Al-Farabi Kazakh National University

UDC: 004(574)(043)

As a manuscript

KARYUKIN VLADISLAV IGOREVICH

The research and development of a module for an intelligent system for analyzing and evaluating the social mood of society in the media space of the Republic of Kazakhstan

6D070300 – Information systems

Degree dissertation on Doctor of Philosophy (Ph.D.)

The local scientific advisor:
Mutanov Galimkair Mutanovich, Doctor of technical sciences,
Professor, Academician of the National Academy of Science
of the Republic of Kazakhstan, Almaty, Kazakhstan

The foreign scientific advisor:
Matteo Negri, Ph.D. in Philosophy of Language,
Center for Scientific and Technological Research
Fondazione Bruno Kessler, Trento, Italy

Republic of Kazakhstan
Almaty, 2023

CONTENT

REGULATORY REFERENCES

In this dissertation, references were used according to the following standards:

– Law of the Republic of Kazakhstan “On Science” of February 18, 2011, № 407-IV LRK.

– SGSE RK 5.04.034-2011 “State compulsory standard of education of the Republic of Kazakhstan. Postgraduate education. Doctoral studies.” The Ministry of Education and Science of Kazakhstan certified the basic rules. “17” June 2011, №261. Astana 2011.

– “Instructions on the design of dissertation works and abstracts, MES RK, Higher Certification Committee, Almaty, 2004, GOST 7.1-2003. Bibliographic inventory.

ABBREVIATIONS

MM – Mass Media
OMSystem – Opinion Monitoring System
eSM – electronic Social Mood
SMMM – Social media marketing management
BERT – Bidirectional Encoder Representations from Transformers
NLP – Natural Language Processing
SA – Sentiment Analysis
ML – Machine learning
DL – Deep learning
NN – Neural networks
DNN – Deep neural networks
CNN – Convolutional neural networks
RNN – Recurrent neural networks
AUC – Area under the curve
ROC – Receiver operating characteristics
NB – Naïve Bayes
LR – Logistic regression
SVM – Support vector machine
k-NN – k-nearest neighbors
DT – Decision tree
RF – Random Forest
LSTM – Long short-term memory
TF – Term frequency
IDF – Inverse document frequency
SMOTE – Synthetic minority oversampling technique
API – Application programming interface
TP – True positive
TN – True negative
FP – False positive
FN – False negative

INTRODUCTION

Relevance of the work. The development of Internet technologies has contributed to a significant increase in the number of news sites and social networks describing various events in the world [1, p.1, 2]. Posting opinions, thoughts, and ideas about ongoing local and global events on social media has become common. Many social networks, such as Twitter, Facebook, YouTube, and others, remain popular and attract many users [3]. In addition, new platforms like TikTok, Instagram, Pinterest, and others are gaining popularity in social media, covering a massive number of events in the world [4].

Since the number of news topics and user opinions is growing at an incredibly fast pace, there is a significant need to keep track of the most important topics [5] in various areas of life (for example, politics, economics, civil society, education, healthcare, ecology, culture, and sports, etc.). The volume of facts and opinions shared on social media makes such tracking impossible without automated methods, which has made analytics platforms especially important. The main element of these platforms is the sentiment analysis tool [6]. Although algorithms are not able to fully understand human feelings, emotions, culture, and mentality, they allow you to determine the general trend of public opinion on certain events using analytical tools [7]. Manual analysis is a very long and resource-intensive process, leaving uncertainties and ambiguities. The use of algorithms makes it possible to quickly obtain operational analytics and implement various hybrid approaches: vocabulary, machine learning algorithms [8, 9], and neural networks [10].

Currently, there are a number of foreign analytical platforms. Among them, such applications as Sproutsocial [11], Hubspot [12], Buzzsumo [13], Hootsuite [14], IQBuzz [15], Brandmention [16], and Snaplytics [17] stand out. Despite their focus on the business sector, these platforms have similar features, which makes the analysis of socio-political and economic aspects of life underrepresented [18-23]. These platforms also mainly work with resource-rich languages such as English, Spanish, Italian, French, and others. Texts in Russian and Kazakh [24] have a very limited representation [25-29, 1, p.2]. Despite their diverse functionality, these systems are similar to each other in that they are mainly focused on business goals, leaving significant social, economic, and political problems little represented by complex analysis. In addition, most existing platforms focus on resource-rich languages such as English, German, French, Italian, Spanish, and Portuguese. In contrast, texts and comments in resource-poor languages such as Russian and Kazakh are not presented well enough. Therefore, a new information system for monitoring public opinion called the Opinion monitoring system (OMSystem) [30, 31], which pays great attention to various topics taking place in the country, was developed to implement the monitoring of the social media space of Kazakhstan. The OMSystem supports leading Kazakh news portals and popular social networks such as Facebook, VKontakte, Instagram, Twitter, and YouTube. A key element of OMSystem is the social sentiment analysis module, which utilizes a data analysis method with the use of sentiment dictionaries, machine learning models, neural networks, and social marketing indicators.

The collected database of 132000 texts from news portals and social networks of Kazakhstan was used during the development of machine learning models for automatic sentiment detection. The texts underwent preprocessing, stemming, the feature extraction using the *tf-idf* metric and the FastText word embedding method and class balancing to obtain the best classification results. At the classification stage, a number of the most popular machine learning algorithms [32] were used (Support vector machine – SVM, Logistic regression – LR, Decision tree – DT, Random Forest – RF, Naive Bayes – NB, k-nearest neighbors – k-NN, and XGBoost) [33-36] and neural networks (Deep neural networks – DNN, Convolutional neural networks – CNN, and Recurrent neural networks – RNN) [37-41]. The data classification results are presented as summary tables of metrics for evaluating the effectiveness of algorithms: accuracy, precision, recall, and F1-score, curve plots (Area under curve – Receiver operating characteristics – AUC–ROC), and confusion matrices. To analyze the social mood of society, models have been developed using marketing indicators in social networks: the level of interest in the topic in society, the level of activity of the topic's discussion, and the level of social mood [1, p.3].

The effectiveness of the developed models was evaluated by conducting an experiment on the topic of vaccination against Covid-19. The summary analysis presented data on public attitudes toward the vaccination campaign, vaccination policy, and government actions and methods to combat the pandemic. The next step was the development of the electronic Social Mood (eSM) module, which is an application that analyzes data obtained using the OMSystem platform.

The purpose of the dissertation work is to develop a method for assessing the social mood of society in the media space of the Republic of Kazakhstan using machine learning models, neural networks, and marketing technologies.

Research Objectives:

1. Analysis of architecture and functionality of the intelligent OMSystem.
2. Development of a module for analysis and assessment of the social mood of society in the media space of the Republic of Kazakhstan using machine learning models, neural networks, and marketing technologies of the OMSystem.
3. Evaluation of the developed module using the analysis of the theme of vaccination against Covid-19.
4. Development of an electronic Social Mood (eSM) module that analyzes data obtained using the OMSystem system and evaluates the social mood of society.

The object of research: text data, publications, news resources, and social media space of the Republic of Kazakhstan.

Research methods: Data mining, Web mining, Natural language processing (NLP), Sentiment analysis (SA), Machine learning, Neural networks, and Marketing technologies of social analytics.

The theoretical significance of the study: analysis of the architecture and the functionality of the intelligent OMSystem, and estimation of the effectiveness of the social mood evaluation module.

The practical significance of the study: analysis of the social mood of society using the developed module of data processing and analysis of the intelligent OMSystem system.

The scientific novelty of the research conducted and the results obtained:

1. A social mood analysis method, characterized by machine learning models and marketing indicators of the user interest in the topic, topic discussion activity, and level of social mood, has been developed.
2. An integrated model of evaluating social mood, including seven attribute machine learning and four deep learning models, has been developed.
3. A sentiment dictionary for the Kazakh language, which is used for an integrated model of social mood analysis, has been developed.

Decrees for defense:

1. A developed model of analysis of the social mood of society using machine learning methods and marketing indicators that allows the evaluation of various socio-political topics and user responses to governmental campaigns.
2. A developed integrated model of evaluating social mood that includes seven attribute machine learning and four deep learning models.
3. Experimental results on the topic of vaccination against Covid-19 that demonstrate public attitudes and government activities through a model of social mood analysis.

The structure of work. The dissertation work consists of 152 pages and includes 69 figures and 30 tables. The content includes 6 sections.

The introduction describes the relevance, novelty, and main purpose of the dissertation work. A list of the main tasks and objects of the study was given, as well as the theoretical and practical significance of the study.

The first section describes the main aspects of information and analytical systems for monitoring social networks, examines in detail foreign and domestic platforms for monitoring and analyzing the social media space, and highlights their advantages and disadvantages. The description of the main methods for determining the sentiment of texts is given: lexicon-based, machine learning-based, and deep learning-based.

The second section presents the developed analytical OMSystem platform in detail. It specializes in advanced analysis and monitoring of social networks and news portals of the media space of the Republic of Kazakhstan. In addition, OMSystem includes Russian and Kazakh sentiment dictionaries, machine learning and neural network models, and tools for modeling and determining the social mood and well-being of society. This section also presents the development of models for binary and multiclass classification of texts, which is an essential part of the dissertation work. The results are presented as graphs, summary tables, and conclusions. The development of models based on marketing management methods in social networks, which allows to determine the indicators of the social mood of the society on given topics, is presented.

In the third section, an experiment was carried out to analyze the social mood of society regarding vaccination against Covid-19. This topic has gained particular popularity due to the rapid spread of the pandemic in the world. It was actively

discussed in news resources and social networks, and thousands of comments were written under posts devoted to this topic. The evaluation of user opinions was performed with the use of sentiment dictionaries, machine learning models, neural networks, and marketing technologies.

The fourth section presents the electronic Social Mood (eSM) module developed on the Django Python framework, which is an application that analyzes data obtained using the OMSystem platform. This module performs the following main functions: creating the main categories of topics for analyzing the social mood of society, extracting quantitative data on each of the topics from the OMSystem database, calculating the level of topic activity in society, the level of interest in the topic in society and the level of social mood, visual presentation of the results obtained in the form charts and tables.

In conclusion, the theoretical and practical results of this dissertation work are summarized, and its most significant aspects in the analysis of the mood of the society using machine and deep learning methods and indicators of social mood are given.

Personal contribution of the researcher. As a result of the work, a detailed analysis of existing platforms for monitoring social networks was carried out. A detailed description of the architecture, functionality, and features of the analytical OMSystem platform, where the study in this work was carried out, was also given. Experimental studies were conducted on developing ML models and NN to determine the sentiment of text data obtained from the work of the web crawler of the analytical system. The eSM module for assessing social mood was also fully developed.

The degree of validity and reliability of scientific results. The results of the dissertation were presented in 12 scientific papers, of which 2 articles and 1 chapter in the book were published in journals and book series peer-reviewed in the Scopus database, 4 articles in journals recommended by the Committee for Quality Assurance in Education and Science of the Ministry of Education and Science Republic of Kazakhstan, and 2 articles in scientific conferences, peer-reviewed in the Scopus database, and 3 articles in the materials of international conferences:

1. Karyukin V., Mutanov G., Mamykova Z., et al. On the development of an information system for monitoring user opinion and its role for the public // Journal of Big Data. – 2022. – Vol. 9, № 110. – P. 1-45.

2. Mutanov G., Karyukin V., Mamykova Zh. Multi-class Sentiment Analysis of Social Media Data with Machine Learning Algorithms // CMC–Computers, Materials & Continua. – 2021. – Vol. 69, № 1. – P. 913–930.

3. Mutanov G., Mamykova Z., Karyukin V., Yessenzhanova S. The Approach to Building a Context-Dependent Sentiment Dictionary // Digital Transformation in Sustainable Value Chains and Innovative Infrastructures. Studies in Systems. Decision and Control – Springer, Cham, 2022. – P. 11-20. https://doi.org/10.1007/978-3-031-07067-9_1.

4. Мутанов Г.М., Мамыкова Ж.Д., Карюкин В.И., Жақсыкелді А.Ж. Разработка машинно-обучаемого алгоритма определения тональности пользовательского восприятия контента // Вестник КазНИТУ Серия Технические Науки – 2019. – Vol. 135, №5. – P. 479-486.

5. Alimzhanova L.M. Karyukin V.I. A classification model based on decision-making processes // Вестник КазНИТУ Серия Технические Науки. – 2020. – Vol. 138, №2. – P. 183-190.

6. Рахимова Д.Р., Тұрарбек А.Т., Карюкин В.И., Карибаева А.С., Тұрғанбаева А.О. Қазақ тіліне арналған заманауи машиналық аударма технологияларына шолу // Вестник КазНИТУ Серия Технические Науки. – 2020. – Vol. 141, №5. – P. 104-110.

7. Karibayeva A., Karyukin V.I., Turganbayeva A., Turarbek A. The translation quality problems of machine translation systems for the Kazakh language // Journal of Mathematics, Mechanics and Computer Science, Kazakhstan. – 2021. – Vol. 111, № 3. – P. 1-9.

8. Karyukin V., Zhumabekova A., Yessenzhanova S. Machine Learning And Neural Network Methodologies of Analyzing Social Media // Proceedings of the 6th International Conference on Engineering & MIS 2020 (ICEMIS'20). Association for Computing Machinery. – Almaty, 2020. – P. 1-7.

9. Rakhimova D., Karyukin V., Karibayeva A., Turarbek A., Turganbayeva A. The Development of the Light Post-editing Module for English-Kazakh Translation // The 7th International Conference on Engineering & MIS 2021 (ICEMIS'21). Association for Computing Machinery – Almaty, 2021. – P. 1–5.

10. Карюкин В., Есенжанова С. Построение контекстно-зависимого тонального словаря // Международная научная конференция студентов и молодых ученых «ФАРАБИ ӘЛЕМІ». – Алматы, 2020. – P. 1.

11. Карюкин В. Подход к построению приложения eSM // Международная научная конференция студентов и молодых ученых «ФАРАБИ ӘЛЕМІ». – Алматы, 2020. – P. 1

12. Карюкин В. Многоклассовая классификация с применением алгоритмов машинного обучения // Международная научная конференция студентов и молодых ученых «ФАРАБИ ӘЛЕМІ», Алматы, 2021. – P. 1.

Connection of the dissertation with research work. This study was carried out as part of the project for the commercialization of the results of scientific and (or) scientific and technical activities “Opinion Monitoring Information System OMSystem (Opinion monitor system),” 0101-18-GK. (The main role of the Ph.D. student was to develop a module for analyzing and evaluating social mood, machine learning models, and neural networks, conducting an experiment on analyzing social mood on the topic of vaccination against Covid-19, and developing an electronic Social Mood module).

1 THE THEORETICAL OVERVIEW OF SOCIAL MEDIA MONITORING METHODS AND MODELS

1.1 Social and political significance of social media monitoring

1.1.1 Main aspects of social media monitoring

The main task of information and analytical work in the social media space is to monitor social networks to collect information about events and predict and manage various processes. Social networks are free for the open expression of user opinions. The main alternative is a large amount of information on social networks and popular news publications. Social networks' main emphasis is attracting attention to a certain event, often putting professional media in the background. Despite the great emphasis on coverage of new events, social networks also become sources of both unverified and inaccurate information, in most cases being spontaneous.

In this regard, studies of social networks are an important component in the context of building an effective communication model in the "government-society" system. This model seeks to improve the exchange of information and dialogue between civil society and political institutions, to study various socio-political problems in depth, which requires a response from both the government and its citizens.

Social networks allow individuals to assert themselves, express their feelings through certain actions and thoughts, receive support and social approval from others, feel more liberated, and use virtual content to build communication. They also allow socially active users to freely and frankly show their opinions on certain authorities' actions in cities, regions, or the country.

The authorities use social media [42] to inform the public and post important information, photo, and video materials. They also engage in discussions, make and collect proposals, conduct surveys, advise and track the dynamics of the number of comments and subscribers, and increase regular feedback, loyalty, and trust.

The representatives of the government have such tasks as determining the goals of entering the social media space [43], describing the target audience, and controlling activities in social networks [44]. In addition, they use them to interact with the public, study citizens' opinions about the organization, determine the time needed to respond to the user, and harmonize with other Internet resources.

In working with social networks, it is crucial to carry out the following actions: promptly inform citizens and other interested persons about the changes that are taking place, monitor the public using tools such as surveys/questionnaires on various issues of public life, analyze the level of public involvement to assess the civic participation of the population, to explore the views of society on the work of the authorities, quickly interact through the feedback channel, extinguishing negative bursts and increasing the duration of contact with the audience of users, publish posts that will encourage people to leave their comments, allow them to speak out in any negative way, moving everything, if possible, into a positive direction, not to make long-term promises in the fulfillment of which there are significant doubts. Social media [45] as a technology of

electronic participation is widely used in more than 152 countries of the world. For example, the UK government uses a special Civil Pages website. More than 91% of local executive bodies in the USA are registered on Facebook, Twitter – 71%, YouTube – 50%, Skype – 22%, Google Docs – 17%, etc.

The world community uses various approaches to regulate the participation of citizens in public affairs using social networks. Public portals are widely used to discuss amendments to laws. These approaches are aimed at increasing the level of satisfaction of the population with the results of the work of governmental bodies, the quality of public services, and the general standard of living. In addition, the task is to improve public administration through the practical application of the principle of "feedback" with the population over a long period of time.

Monitoring the social media space [46] includes data acquisition and structuring as the main tasks. Data are collected from various sources, such as news portals, social networks, and group accounts on social networks. At the analysis stage of the obtained data, the number of collected texts and user comments, statistical and structural laws of the studied area, and special laws of narrow subject areas is determined.

Data analysis systems perform the following important functions:

– Monitoring, analyzing, and forecasting in social networks.

– Building network structure models (trees, graphs, network nodes, etc.).

– Constructing information transmission models (Markov models, automata, diffusion propagation models, etc.).

– Applying text semantic analysis methods, classification, clustering, SA, etc.

– Selecting social network analysis objects: networks, in general, using aggregated indicators, subnets, certain users, communities, and external nodes (Internet information resources).

– Defining whether to collect the entire amount of data or to collect data on a specific topic.

– The selection of data analysis sources covered news portals (Informburo, Zakon, Nur, etc.), social networks (Vkontakte, Facebook, Instagram, etc.), blogs, forums, and others.

The fast development of data analytics technology has drawn attention to the possibility of using information from social networks in various industries. The combination of structural and content data gives great potential for using social networks to solve many business problems: brand management, advertising, anti-fraud, etc. As a result, the popularity of the technology of "monitoring social networks" (social listening) and content analysis has increased. However, these services generally work well with foreign content and English sentiment dictionaries, making them difficult to use in Russian and Kazakh. Thus, there is an urgent need to create a platform for monitoring and analyzing social networks and content analysis [47], specially adapted for the Russian and Kazakh languages.

Work on monitoring the social network and content analysis will allow:

– Detecting information attacks and stuffing at an early stage;

– Monitoring ways and means of disseminating information;

– Identifying the range of interests and dissemination of information;

– Collecting extended social profiles and automated analysis for signs of specified risks based on human behavior in social networks;

– Identifying existing and potential sources of negativity requiring attention, as well as sources of acute discussion for immediate response;

– Monitoring the dynamics of user involvement in a particular topic;

– Identifying information causes and attitudes;

– Monitoring the mood of Internet users for perceptions of specific topics of discussion [1, p.3].

An example of the importance of monitoring the social media space in 2020-2021 is the active discussion of topics related to the Covid-19 pandemic in news portals and social networks. In addition, the issue of vaccination against coronavirus infection has become a particularly acute issue of political, social, and economic orientation.

Data analysis with monitoring systems makes it possible to understand the public's attitude toward government activities using methods for assessing social mood [48]. It allows to make the right decisions, accelerate the implementation of large-scale government tasks and ensure the preservation of public health.

1.1.2 The concept of the study of monitoring the user opinion perception of political content

Currently, in most cases, the assessment of events in society is carried out using certain indicators that allow you to better analyze and understand people's moods on the most relevant and important topics.

The analysis of opinions of user perception of political content is based primarily on obtaining views from various sources: news portals, social networks, blogs, etc. At the same time, in most cases, a significant amount of data is required to build an objective picture of users' perception of certain events.

Methods of obtaining and accumulating a user database are multiple, ranging from manual monitoring and recording of social media space to the use of complex analytical platforms for the operational accumulation of data from a large number of different sources. Such analytical tools allow you to obtain huge databases of user texts and comments using predefined measures to search and extract data.

At the same time, since the main emphasis is on users' subjective opinions, the emotional coloring of vocabulary in the text, or sentiment [49], plays an important role. The very meaning of the sentiment [50] of the text found its origin in computational linguistics. SA includes tasks such as determining the polarity of text at the document or sentence level and extracting aspects from the emotionally colored text. The three types of text sentiment are most often distinguished: positive, negative, and neutral. Positive sentiment determines texts with a positive connotation, and negative texts mainly include negative events and user opinions. The neutral sentiment means that the text does not contain an emotional component. In addition to sentiment, adapted indicators of social networks, such as the level of interest in the topic in society, the level of activity of the topic's discussion in society, and the level of social mood, are also added to the analysis of social mood (the detailed meaning of these metrics is

disclosed in chapter 2.3.4). A general scheme for monitoring user perception of political content is shown in Figure 1.1.

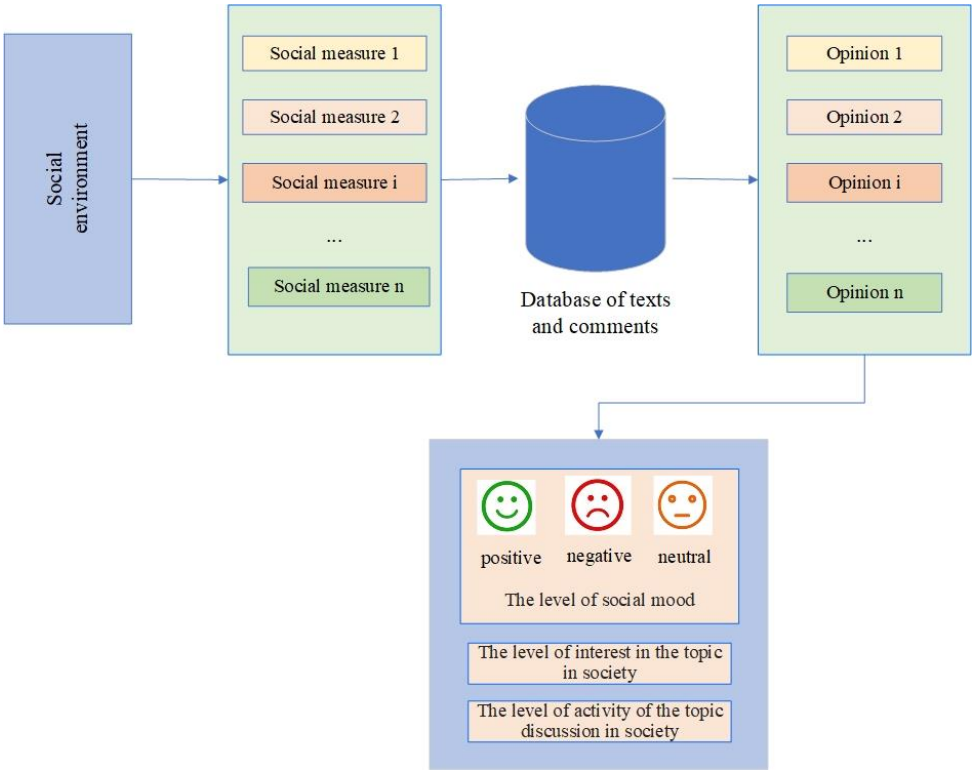

Figure 1.1 – Monitoring scheme

1.2 Analysis of information systems for monitoring social media

1.2.1 Analysis of existing platforms

The rapid growth of the Internet, analysis of data, and social networks made the appearance of different instruments and platforms for the definition of social opinion, evaluation of social well-being, and brand promotion [51]. These tools can be directed to estimate the socio-economic situation in the maintainable development. Modern achievements of foreign countries in the field of social network analysis and brand promotion are represented by a wide range of tools [52]. Among the most significant platforms that occupy an essential place in the study of social networks are Sproutsocial, Hubspot, Buzzsumo, HootSuite, IQBuzz, Brandmention, Snaplytics, and others.

Sproutsocial (https://sproutsocial.com/) [1, p.5] is a powerful analytics platform for presenting various features in social networks. Sproutsocial is a valuable tool for counting Twitter links, measuring Instagram follower growth, evaluating LinkedIn engagement, and other important tasks. This platform collects and tracks posts from Instagram, Facebook, and other social networks. In addition, it collects customer satisfaction data through the Twitter DM survey system. The system uses machine learning models to provide answers to common user questions. In addition, this platform can evaluate the results obtained using visualization. Leading companies such as Subaru,

Chipotle, Zendesk, and others use the social management tools of the Sproutsocial platform.

HubSpot (https://www.hubspot.com/) [1, p.5] is a platform for obtaining data on the level of communication in social networks and presenting previous loads on the main interest of consumers in products. In addition, HubSpot can detect the various effects of social networks on the profits of organizations and create sophisticated reports for fast and efficient analysis. Business managers can use HubSpot to monitor and compare different brands on social media, as the platform allows you to track website activity, display a dashboard of key performance indicators (KPIs), automate sales, and more.

BuzzSumo (https://buzzsumo.com/) [1, p.6] is an advanced platform that reflects social communications on various topics in social networks. It helps to find data from online sources, considering various characteristics, such as likes, comments, and shares. In addition, this tool looks for the most important domain and subject information. It presents the most appropriate destinations for an interested audience. It also collects the most accurate social media content data.

HootSuite (https://www.hootsuite.com/) [1, p.6] is a multifunctional application that provides opportunities for working with various social networks. This platform works with Twitter, Facebook, LinkedIn, as well as WordPress blogs. HootSuite allows you to monitor multiple Twitter accounts at once. Hoot-Suite also offers a wide range of analytics tools, including Google Analytics.

IQBuzz (https://iqbuzz.pro/) [1, p.7] is also one of the most widely used professional tools for monitoring social media and presenting online reputation metrics. IQ-Buzz monitors many sources and platforms, such as Twitter, Yandex, LiveInternet, LiveJournal, various blogs, video hosting sites, such as RuTube and YouTube, various news, entertainment, and specialized services, and thematic and regional portals. One of the key advantages of the service is the ability to connect new sources and Internet resources for monitoring.

Brandmention (https://brandmentions.com/) [1, p.7] is an advanced tool for analytics and search in various social networks. This platform includes a list of the most common monitoring sources, data search keywords, and sentiment analysis tools, including over 100 social networks, services, forums, blogs, and more. Brandmention allows you to set up keywords for social monitoring and search of the company and social analytics tools of its competitors. Some keywords may also be excluded from search results.

Snaplytics (https://thehub.io/startups/snaplytics) [1, p.7] is a cloud-based platform that analyzes the stories of millions of Instagram and Snapchat users, being an effective promotion tool. The main functions of Snaplytics are publishing, monitoring, and analyzing posts. In addition, users can track comments and replies, post stories from various sources, and view ratings. Snaplytics also allows you to create reports and export them to CSV files and other formats.

In Kazakhstan, there are few social analytics platforms. iMAS, Alem Media Monitoring, ExWeb (https://exweb.kz), Media Analytics

(https://nlp.iict.kz/accounts/login/?next=/topicmodelling/topics_list/), and the OM-System are the most famous ones [1, p.7].

iMAS has a wide functionality covering information about the company's brands, evaluating the information field, and identifying threats, critical and negative statements, and disinformation. In addition, this application allows you to monitor the reputation of officials and assesses the level of public confidence in ongoing government programs and reforms. Based on the analysis results, the iMAS platform makes it possible to generate reports in various sections [1, p.7].

Alem Media Monitoring conducts a comprehensive analysis of tension in social networks in various areas of society in Kazakhstan, automatically determines the sentiment of publications and user opinions, assesses the level of social conflicts, defines events with the highest level of discontent, and identifies leaders of public opinion in social networks. The main social networks covered include VK (Vkontakte), Facebook, Twitter, Instagram, Telegram, Odnoklassniki, "Moi Mir," and YouTube. The organization also analyzes topics in such areas of society as healthcare, education, corruption, the judicial system, housing, communal services, business, employment, social protection, transport, infrastructure, ecology, animal protection, public administration, security, protest moods, party system, interethnic and religious relations, etc. [1, p.7].

ExWeb is dedicated to developing methods and tools for identifying extremism in the digital space through the analysis of social media, which is important for the national security of Kazakhstan. The system analyzes the content of Vkontakte, Youtube, and Twitter social networks using pre-trained machine learning models. It determines the connections between network users and their specific groups and the community's leader. The purpose of the system is to conduct a comprehensive study and develop models, algorithms for semantic data analysis and software for detecting extremist content in web resources, methods for identifying involved users and algorithms for graphical visualization of links, creating and researching a model for analyzing cryptocurrency transactions in Darkoin payment systems and its adaptations for identification of funding sources, development of cyber-forensic tools to counter extremism. The system contains general information about religious organizations banned in the country and Kazakhs who left the country and went to war in Syria and Iraq [1, p.7].

Media Analytics is a platform for analyzing text data for management decisions using Big Data processing methods. The platform allows managers and experts to obtain flexible analytical tools for presenting rating publications on various topical topics of social media space and visualize the obtained data in the form of reports. The development of the system included technical requirements such as modularity, scalability, data conversion, and visualization with filtering and full-text search. This system has such functional capabilities as using the teacher-free learning method to search for topics without given keywords, identifying hidden patterns without connecting experts to work, working with a small volume of marked buildings in comparison with other analytical systems, developing predictive models, configuring parsers, analyzing the tonality of Russian and Kazakh texts, etc.

OMSystem [1, p.8] is a new domestic information system for monitoring Kazakhstan's social media space and analyzing society's social mood. This system analyzes leading news portals (Tengrinews, Nur.kz, Zakon.kz, Vesti.kz, etc.), social networks (Facebook, Instagram, Vkontakte, Telegram, Youtube, Twitter), and other resources. OMSystem also allows to estimate users' involvement in a subject, the formed public opinion, to reveal information occasions, analyze dynamics of mentions of various actions and events, and obtain relevant information on the country's political and social situation. The definition of social mood is based on a data processing and analysis module using tone dictionaries, machine learning models, neural networks, and marketing tools. The results of OMSystem work are monitoring the media space over a certain period, analysis of the social mood of society, and prompt access to monitoring output data in the form of reports, graphs, and summary tables.

Despite a significant number of advantages of these platforms, they also have a number of disadvantages. Foreign platforms are effective in promoting brands. Their analytical mechanisms are focused on time-personal business tasks, possessing many marketing tools. Nevertheless, most of them are poorly adapted to social policy, leaving significant social, economic, and political problems unresolved. Moreover, most of them focus on resource-intensive languages, such as English, while texts and commentaries in other languages with limited resources, namely Kazakh and Russian, are not sufficiently represented, which makes them little suitable for monitoring the social media space of Kazakhstan. In addition, Hubspot, Buzzsumo, Sproutsocial, HootSuite, Brandmention, Snaplytics, and IQBuzz require a permanent paid subscription. Consequently, the Kazakh-Stanek platforms' role in social media analytics cannot be underestimated. A comparison of the functionalities of Alem Media Monitoring, iMAS, ExWeb, Media Analytics, and the OMSystem is shown in Table 1.1.

Table 1.1 – Comparative characteristics of Kazakhstan social analytics platforms

№	Name	OMSystem	iMAS	Alem Media Monitoring	ExWeb	Media Analytics
1	2	3	4	5	6	7
1	The Kazakh language support	✓	✓	✓	✓	✓
2	The Russian language support	✓	✓	✓	✓	✓
3	The English language support	✓	✓	✓	✓	✓
4	Social media monitoring	✓	✓	✓	✓	✓
5	Sentiment dictionary	✓	✗	✗	✓	✗
6	Machine learning models	✓	✓	✓	✓	✓
7	Resource catalog configuration service	✓	✓	✓	✓	✓

Continuation of Table 1.1

1	2	3	4	5	6	7
8	Service for adding new resources	✓	✓	✓	✓	✓
9	Rule profile setting service for the search of a topic	✓	✗	✗	✓	✓
10	Analysis detail	✓	✓	✓	✓	✓
11	Data upload settings service	✓	✓	✓	✓	✓
12	A possibility of inviting an expert	✓	✗	✗	✗	✗
13	Social mood modeling	✓	✗	✗	✗	✗
14	Social well-being modeling	✓	✗	✗	✗	✗
15	News feed	✓	✗	✗	✗	✗
16	Geographically interactive opinion map	✗	✓	✗	✓	✗
17	Graph visualization	✗	✗	✗	✓	✗

This table shows that the OMSystem platform has very wide functionality, an important place of which is occupied by sentiment dictionaries, machine learning algorithms, data evaluation by experts, modules of social mood, and social well-being of society. The system also has functionality that is not available in iMAS and Alem Media Monitoring, which makes OMSystem a particularly important platform for analyzing the media space of Kazakhstan.

1.2.2 Algorithms of sentiment analysis

Most social data analytics applications aim to determine the sentiment [53, 54] of text at the level of a document, sentence, paragraph, phrase, or single word. These levels are characterized as follows:

– The document level. Analysis at this level is aimed at determining whether the document expresses a completely positive, neutral, or negative mood. It assumes that the document is a single entity and gives a description or characteristic of one object.

– The paragraph level. In this case, the document is divided into paragraphs as essential components of a single document. Once separated, each paragraph is assigned a specific sentiment value. It is especially true if the document discusses several significantly different objects from each other.

– The sentence level. This level considers the importance of each sentence in determining its sentiment and is crucial if it is required to divide the entire document or paragraph into entities deeply.

– The phrases and individual words level. Analysis at this level moves to the smallest units. It determines each individual word or phrase's sentiment, positive,

neutral, or negative. Usually, the definition of the word sentiment is important when analyzing the sentiment of texts using a lexicon-based approach. Thus, setting the sentiment at this level can be considered the construction of mood vocabulary.

Among the main approaches to determining the sentiment of texts [55, 56] are the following methods:

– Lexicon-based
– Machine learning-based (ML-based)
– Deep learning-based (DL-based)

The lexicon-based approach determines the sentiment of a document based on lexical units and applies a function that calculates the sentiment by the maximum number of positive, negative, and neutral words in the text. Thus, the lexicon is the most important component of this approach. There are three different ways to build them:

a.) A manual approach that depends heavily on human effort. Users label the sentiment of words as positive, neutral, or negative regardless of context. They consider this aspect in labeling in some cases where the context is particularly important. Sometimes a more complex way of labeling is used, considering the sentiment strength by using additional categories. So strongly positive, strongly negative, weakly positive, weakly negative, and neutral sentiments are distinguished. The main disadvantage of the manual approach lies in its labor-intensiveness and time-consuming.

b.) A dictionary approach [57] uses synonyms and antonyms of words in a label sentiment word base. This method works as follows: the initial base of known sentiment words is formed using a manual approach. The set is then expanded by searching an extensive dictionary for synonyms and antonyms, such as WordNet or any other. Found words are added to the existing word base. Next, a new search is performed until the base is no longer replenished or replenished very slowly. The disadvantage of this approach is the dependence on already formed dictionaries and the presence of errors in them. Therefore, manual checking and correcting found errors and inaccuracies are recommended to minimize them.

c.) A corpora-based approach uses the following statement "A document must be positive, neutral, or negative if it contains many positive, neutral, or negative words, and a word must be positive, neutral, or negative if it occurs in many positive, neutral, or negative documents." This approach is often used when an initial list of known sentiment words is present, and it is required to find other sentiment words and their orientation from the domain corpora. Given the set of automatically collected tagged data, it is necessary to determine the polarity of each word. The disadvantage of this approach is also defined by the large dependence on a predetermined list of positive, neutral, and negative words.

Currently, sentiment dictionaries have been developed for English, Spanish, French, Italian, German, Russian, and other languages. Most of these dictionaries are created and maintained by institutions, academic and research organizations, and other development communities. Examples of sentiment dictionaries include NRC Word-Emotion Association Lexicon, SentiWordNet, SenticNet, VADER, AFINN, SentiRuEval-2015, SentiLex, and others. The situation with sentiment dictionaries is

significantly different for a few resource languages. Limited studies present them and are absent in open access. However, to determine the sentiment of texts in Kazakh, you can use public dictionaries and online resources in other languages and translate them into Kazakh. Despite some limitations of this approach related to the peculiarities of the language, it will determine the general emotional color of words and phrases in the context.

Based on this, it can be noted that the effectiveness of the dictionary-based approach depends on the high quality of sentiment dictionaries containing a large corpus of words labeled in the categories mentioned earlier. A notable disadvantage of this approach is the need to include many linguistic resources to search for basic words for SA.

The ML-based approach [58] addresses the task of classifying documents. As with the widespread classification of texts on topics such as politics, economics, security, civil society, sports, etc., in classifying sentiments, words that highlight positive, neutral, and negative opinions play a crucial role. At the same time, the main task is not to determine the sentiment of individual words but the entire document. In general, two phases are distinguished in supervised learning: training and predicting. During a training phase, a corpus of texts labeled by the sentiments is used. The texts go through the stages of preprocessing and vectorization, and then the ML algorithms themselves are applied to them. The trained model is used to label new text documents. The scheme of the ML application is shown in Figure 1.2.

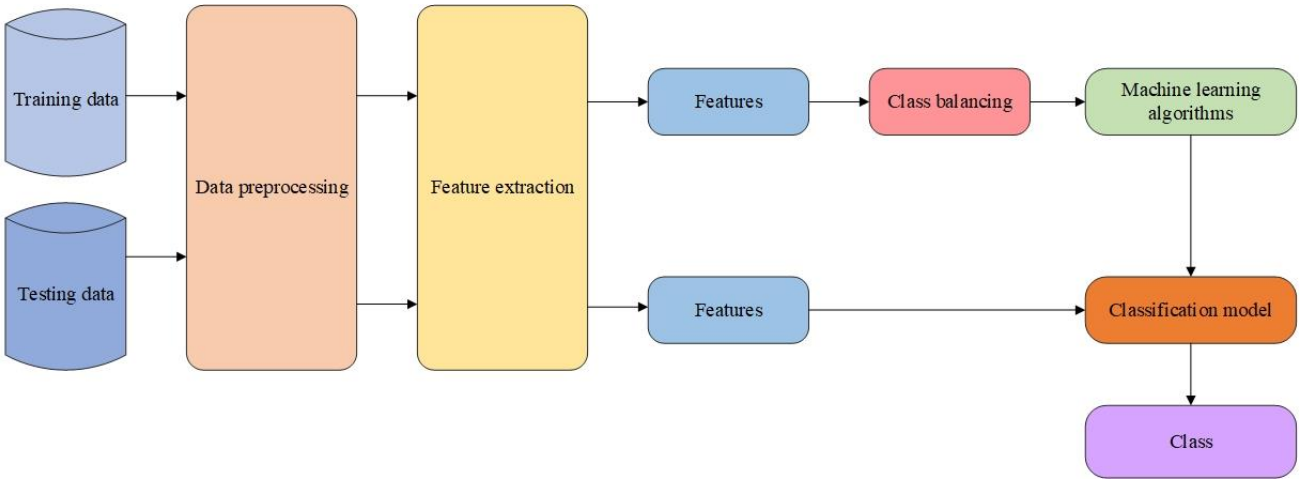

Figure 1.2 – Stages of text classification system training and prediction

In the process of implementing the approach based on machine learning, first of all, text preprocessing is performed, including the removal of unnecessary characters, punctuation marks, links, etc. Stop words, which include words that do not carry a special semantic load, are also removed. Examples of such words are prepositions, conjunctions, pronouns, etc. (“на” – “on,” “в” – “in,” “все” – “all,” “и” – “and,” “но” – “but” and others). The text preprocessing stage also includes important steps such as stemming and lemmatization, which reduce the number of words with similar emotional meanings. The difference between these approaches is that lemmatization converts words to the infinitive form while stemming removes affixes and endings of

words to obtain their roots. Stemming is an easier way to write an algorithm for removing parts of a word. Lemmatization, on the contrary, requires significant efforts to develop rules for reducing words to the infinitive form.

After the text data preprocessing, the feature extraction stage, or vectorization, is performed. Since machine learning algorithms work only with numerical data, textual information must be converted into the appropriate numerical feature set. The most popular vectorization methods are Bag of words, term frequency – inverse document frequency (*tf-idf*), and word embedding methods: Word2vec [59], Glove [60], FastText [61], and others. Although they have different principles of operation, the main task of all of them is to represent text documents in the form of numerical vectors. The *tf-idf* metric is one of the most efficient and commonly used vectorization methods. It includes two components *tf* (term frequency) and *idf* (inverse document frequency).

Tf measures how often a particular word occurs in a document. It is possible to determine how important a word t_i is in a specific document (1.1):

$$tf(t,d) = \frac{n_i}{\sum_{i=1}^{k} n_i}, \tag{1.1}$$

where n_i is the number of occurrences of the word in the document, and $\sum_{i=1}^{k} n_i$ is the total number of words in the document.

Idf measures how common a word is in all the documents. Rare words get a higher weight. *Idf* is calculated by the formula (1.2):

$$idf(t,D) = \log \frac{|D|}{|(d_i \supset t_i)|}, \tag{1.2}$$

where $|D|$ is the number of documents in the corpus; $|(d_i \supset t_i)|$ is the number of documents where t_i occurs.

After calculating the values, *tf* and *idf* parts are multiplied (1.3):

$$tf\text{-}idf = tf \times idf \tag{1.3}$$

Word embedding methods are the other most popular ways to vectorize texts. They are able to capture the context of a word in a document, semantic and syntactic similarities, and relationships with other words.

Word2Vec is one of the methods for constructing such an attachment. It can be obtained in two ways (both using neural networks): Skipgram and Common Bag of Words (CBOW). The CBOW model learns to predict the target word using all the words in its surrounding. On the other hand, the Skipgram model learns to predict a word based on an adjacent word.

The FastText word embedding method uses subword information to create word embeddings. It studies the representations of n-gram symbols and words represented as a sum of n-gram vectors. The FastText method extends the Word2vec type models with subword information to help embeddings understand suffixes and prefixes. Once a word is represented using symbolic n-grams, the Skipgram model is trained to learn embeddings.

The next step after vectorization is the classification itself using ML algorithms. Among them, the following algorithms are distinguished: NB, LR, SVM, k-NN, DT, RF, and XGBoost.

In supervised learning, comparing predictions with available data labels is possible. On the other hand, unsupervised learning has no labeled data, and it is necessary to find data communication patterns. The main problem is the lack of initial information on which characteristic data can be combined and a way to verify the correctness of the data groupings performed. Clustering aims to group unlabeled data and select the most effective ways to find the best criteria.

The most common clustering algorithms are k-means, fuzzy c-means, and hierarchical clustering. The k-means method is one of the most popular clustering methods that groups similar data points around centroids representing cluster centers and discovers their commonalities. The algorithm seeks to minimize the function of quadratic errors and is expressed by formula (1.4):

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i^{(j)} - c_j \|^2 , \quad (1.4)$$

where $\| x_i^{(j)} - c_j \|^2$ is a selected measure of the distance between the data point x_i and the cluster center c_j and determines the distance of the n data points from their respective cluster centers.

In the fuzzy c-means method, each point has the probability of belonging to each cluster and not wholly belonging to only one cluster, as in the case of traditional k-means. Moreover, this method tries explicitly to solve the problem when the points are between centers or otherwise ambiguous.

In hierarchical clustering, the clustering of elements takes place in several stages. First, each element is assigned to one cluster (if there are N elements, then there will be N clusters). A similar pair of clusters is then searched and merged into one cluster, reducing the number of clusters by one value. Then the distance between the new cluster and each of the old clusters is calculated. The operation is performed cyclically. The clustering scheme is shown in Figure 1.3.

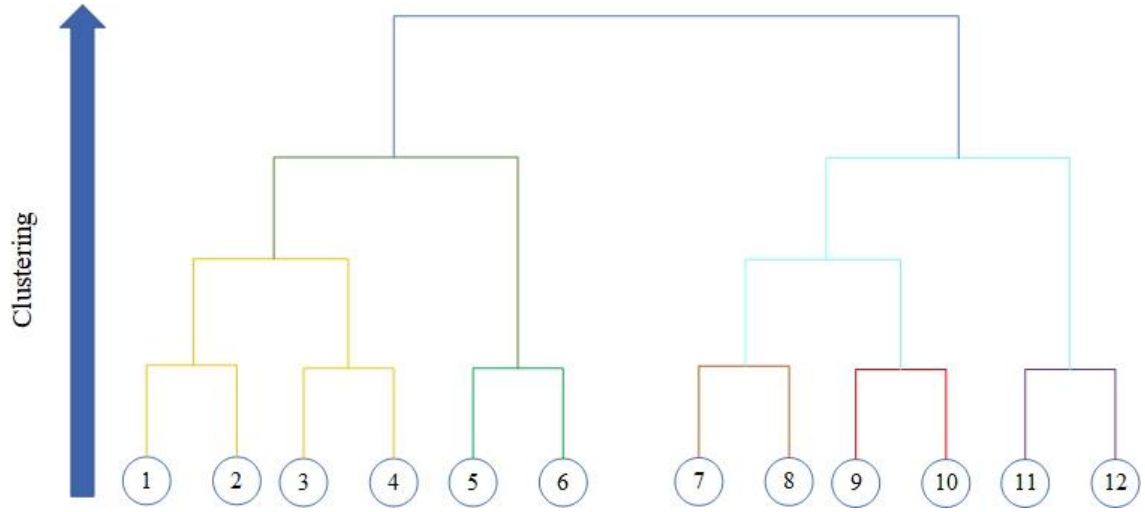

Figure 1.3 – Hierarchical clustering

A number of recent studies have focused on the in-depth learning approach, which focuses on improving the effectiveness of text classification through its superiority in terms of accuracy in learning from a significant amount of data. To this end, the use of Deep neural network (DNN), Recurrent neural network (RNN), and Convolutional neural network (CNN) is well documented in the literature. DNN is a type of NN that includes several layers: an input layer that processes the representation of input data, hidden layers that abstract from this representation, and an output layer that predicts a class based on internal abstraction. CNN is a DNN consisting of convolutional and unifying layers. While convolutional layers filter input data to retrieve objects, combining layers reduces the dimension of objects. The last layer reduces the dimension of the vector to the length of the categorical representation of the class. RNN is a network in which connections between neurons create a directional loop that forms feedback loops. This type of network can store the previous calculation steps and reuse the information in the next input serial. Recently, deep learning models based on the Transformer architecture (BERT, ROBERT, GPT-3, XLNet, T5, and others) have become widespread. This architecture allows you to achieve impressive results in the field of natural language processing.

A study of the effectiveness of machine learning algorithms and neural networks for analyzing text tonality using the example of a publicly available IMDB dataset is demonstrated in [62, 63]. The data classification model training showed the following accuracy values: ML (NB, LR, SVM) – 80% -84%, LSTM – 86.96%, CNN – 85.04%, Transformers (BERT, ROBERT, ELECTRA) – 91.58%. By training time and testing, the study results showed the following values: ML (training – 15 s, testing – 9 ms), CNN (training – 30 s, testing – 10 ms), LSTM (training – 1 m 40 s, testing – 15 ms), Transformers (training – 27 m 58 s, testing – 35 ms). The overall values of the experimental results showed that Transformers surpassed ML, CNN, and LSTM models in classification accuracy. However, they are very resource-intensive and time-consuming, which also needs to be considered when introduced into social mood analysis platforms.

1.2.3 Machine learning algorithms

The ML-based approach uses a number of algorithms, including NB, LR, SVM, k-NN, DT, RF, and XGBoost.

An NB [64, 1, p. 13] is one of the simplest and most commonly used machine learning algorithms for text classification, using a probabilistic approach based on the Bayes theorem with strong data independence assumptions. NB considers each feature independently of other features and evaluates the likelihood of each affecting the final result. In the text classification, NB is trained on documents for each class, where the conditional probability that document *d* belongs to class *c* is calculated. This formula is represented by expression (1.5) [63]:

$$P(c \mid d) = \frac{P(c) \times P(d \mid c)}{P(d)}, \tag{1.5}$$

where $d = \{x_1, x_2, ..., x_n\}$; x_i is the weight of the i^{th} word in the document d; c is the class of the document.

SVM [65, 1, p. 13] is another popular ML algorithm. The algorithm uses a feature space shared by a hyperplane located at the maximum distance from the nearest points of the two classes of training data. The wider the boundary, the smaller the classifier error, and the more efficient data separation is achieved.

The hyperplane equation is written as follows (1.6) [65]:

$$y_i(\vec{w} \times \vec{x} + b) \geq 0, \tag{1.6}$$

where $\vec{x} = (x_1, x_2, ..., x_n)$ is the feature vector; $\vec{w} = (w_1, w_2, ..., w_n)$ is the weight vector; y_i is an output value; b is a shift. If the value is greater than or equal to zero, it belongs to the positive class. Otherwise, it belongs to the negative class.

The SVM separating hyperplane (Figure 1.4) mainly works with two-class classifiers. However, it is easily adapted to multiclass classification, using a set of classifiers, “one vs. all.” In addition to linear classification, this algorithm efficiently performs nonlinear classification using so-called kernels, implicitly mapping its input data to multivariate feature spaces.

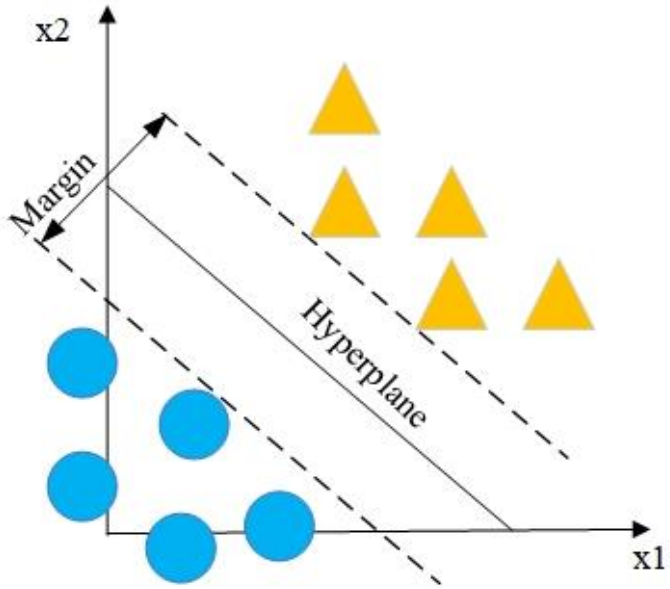

Figure 1.4 – Separating Hyperplane

The classification uses a model that predicts the probability of an independent variable in the interval [0,..., 1]. LR predicts outcome using a logistic function (1.7) [66, 1, p. 13]:

$$p(x) = \frac{1}{1+e^{-f(x)}}, \tag{1.7}$$

where $f(x) = w_0 + w_1 x_1 + ... + w_r x_r$ is the linear function of the classifier; $\vec{x} = (x_1, x_2, ..., x_n)$ is the vector of features; $\vec{w} = (w_1, w_2, ..., w_n)$ is the vector of weights. The logistic function $p(x)$ is in the form of a sigmoid (Figure 1.5) with probability values from 0 to 1. Document *d* belongs to the first class if $p(x)$ is close to zero. Otherwise, it is placed in the second class.

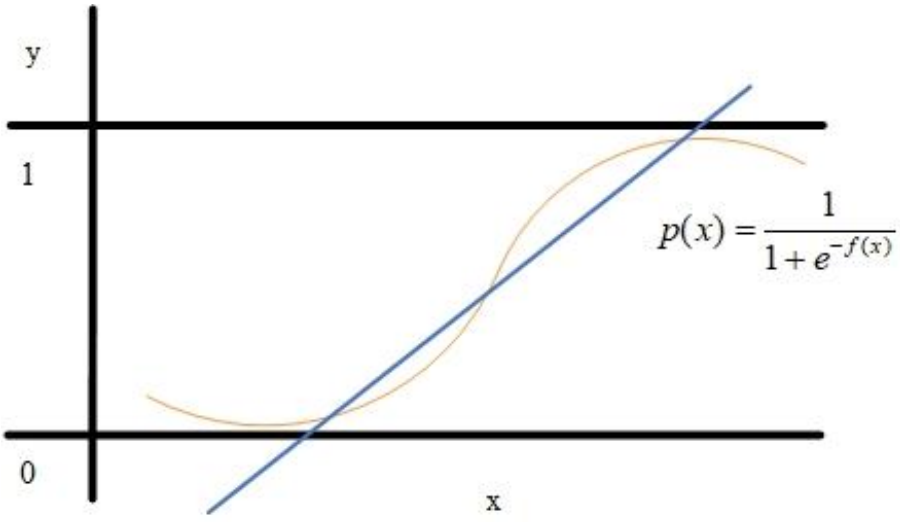

Figure 1.5 – Logistic function

In the case of multiclass classification, a one vs. one (OvO) approach is used to identify a particular class. In this approach, a multiclass dataset is broken down into several binary classification tasks. Each binary classifier is trained on instances belonging to one class and instances belonging to another. The one vs. all (OvA) method is also used. Here, a set of binary classifiers is trained to distinguish instances of one class from all other instances. The advantage of OvO over OvA is that the datasets of all the individual classifiers are balanced when the entire multiclass dataset is balanced.

The k-NN [67, 1, p. 14] method is one of the simplest data classification algorithms. It calculates the distances between vectors and assigns points to the class of its k nearest neighboring points. This algorithm typically classifies documents using the most widely used distance measure, called the Euclidean distance, which is defined as (1.8) [67]:

$$d(x,y) = \sqrt{\sum_{i=1}^{N}(a_{ix} - a_{iy})^2}, \tag{1.8}$$

where $d(x,y)$ the distance between the two documents; a_{ix} and a_{iy} are the weights of i the term in the documents x and y, respectively, N is the number of the unique word in a set of documents. The k-NN method memorizes the feature vectors and their class labels during learning. Where class labels are unknown, the distance between the new observation and the previously stored vectors is determined. Then the k nearest vectors are chosen, and the new object belongs to the class to which the majority belongs. The choice of the parameter k's value is ambiguous and requires experimental approaches. The classification accuracy improves with its increase, but the boundaries between classes become less clear. This method shows good classification results, but its main disadvantage is the high computational complexity with an increase in the training sample size.

A DT [68, 1, p. 14] is a supervised learning method that uses a set of rules to make decisions the same way a person makes. This method divides the data into subsets depending on certain features, answering specific questions until all data points belong to a particular class. Thus, a tree structure is formed with the addition of a node for each question (Figure 1.6). The first node is the root node. When classifying documents, the first step is to select a word, and all documents containing it are placed on one side, and documents that do not contain it are placed on the other side. As a result, two datasets are formed. After that, a new word is selected in these datasets, and all previous steps are repeated. It continues until the entire dataset is split and assigned to end nodes. If all data points in a leaf node uniquely correspond to the same class, then the class of the node is well-defined. In the case of mixed nodes, the algorithm assigns the given node the class with the largest number of data points related to it.

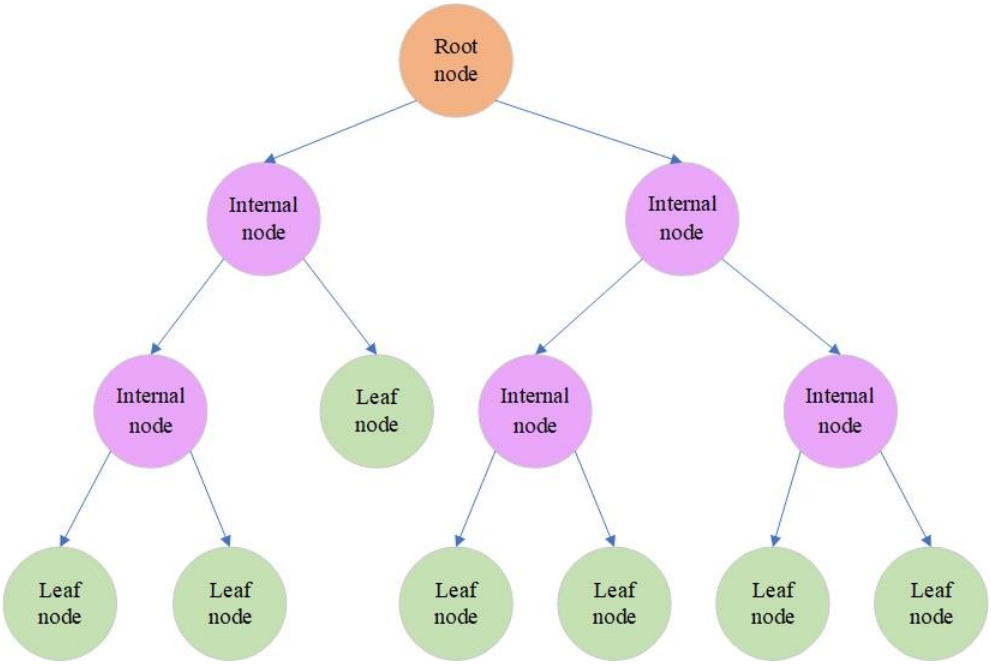

Figure 1.6 – Decision Tree

An RF [69, 1, p. 15] is a popular machine-learning algorithm based on the concept of ensemble learning. This concept combines multiple classifiers to improve the model's performance. An RF consists of not one but many decision trees (Figure 1.7). In classification problems, each document is independently classified by all trees. The class of the document is determined based on the highest number of votes among all trees.

The RF algorithm has the following number of features and advantages:

– Fast training.
– Efficiently processing data sets with a large number of characteristics.
– Performing data prediction with very high accuracy.
– Showing good performance even if there is a large number of data gaps.
– Processing both continuous and discrete features is a good way.
– High scalability.

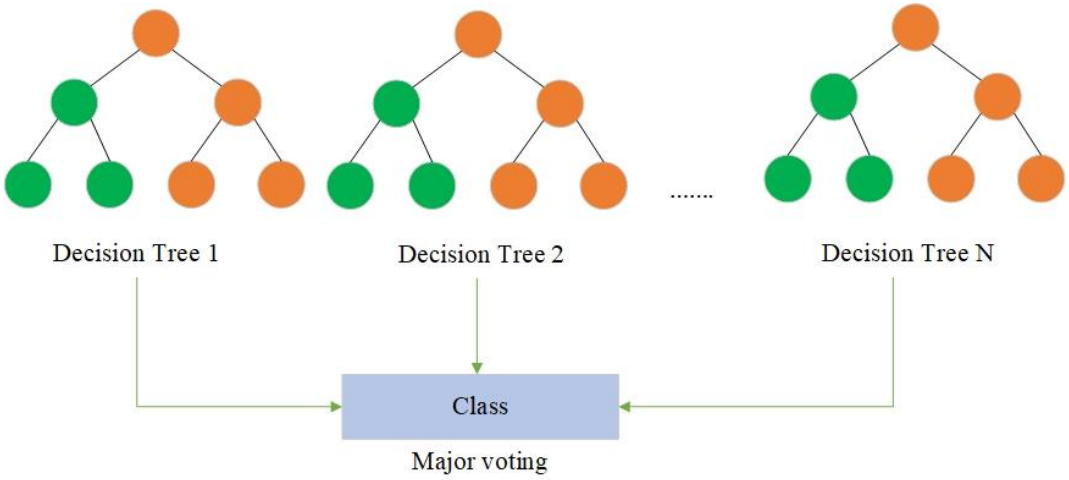

Figure 1.7 – Random Forest

XGboost (eXtreme Gradient Boosting) [1, p. 15] is an advanced machine-learning algorithm that uses the boosting principle. It has good performance and solves most regression and classification problems. Enhancement is an ensemble technique in which previous errors are eliminated in the new model. Deviations of the trained ensemble predictions are calculated on the training set on each iteration. Thus, optimization is performed by adding new tree forecasts to the ensemble, reducing the average model deviation. This procedure continues until the required error level, or "early stopping" criterion, is reached.

1.2.4 Neural network algorithms

Traditional machine learning algorithms allow you to achieve good data classification results. Nevertheless, in the last decade, the deep neural network approach has become dominant in the field of artificial intelligence, showing high accuracy in tasks such as speech and image recognition, text classifications, and natural language processing. Several types of NN are often used: DNN, CNN, and RNN.

Deep neural networks (DNN) [70] are a model of NN with two or more hidden layers. The neural network consists of an input layer containing input data, hidden layers including nodes called neurons, and an output layer containing one or more neurons (Figure 1.8).

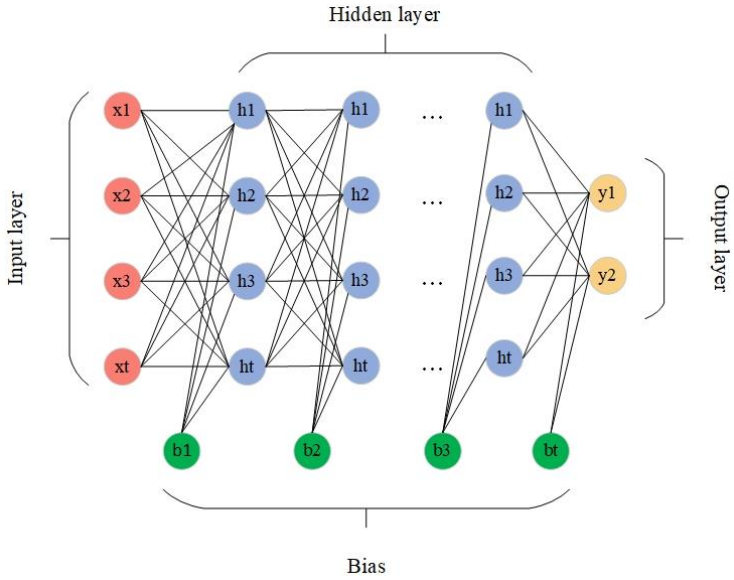

Figure 1.8 – Deep neural network

In this case, $x = x_1, x_2, ..., x_f$ is the input vector; $w_1, w_2, ..., w_i$ is the connection weights vector of each level; $b_1, b_2, ..., b_i$ is the displacement vector. Levels l_2 to l_{n-1} form hidden layers, and the l_n level is represented by the corresponding $y_1, y_2, ..., y_m$ output vector. Elements of hidden and output layers are called neurons. They are represented by the activation functions responsible for the nonlinear functional mapping

between the input data and the response variable. The most popular activation functions are the sigmoid function, the hyperbolic tangent function (*tanh*), the rectified linear unit (*ReLu*), and softmax. The sigmoid function is mainly used on the output layer in the binary classification since it defines the output value as 0 or 1. The *tanh* function is an improved version of the sigmoid function with the difference only in that the *tanh* output values range from -1 to 1. Hidden layers most often use the *ReLu* activation function. It results in an output value of 0 if it receives a negative input of *x*. Otherwise, for positive inputs, it returns *x* unchanged, like a linear function. Mathematically, this is denoted by $f(x) = \max(0, x)$. A softmax function is used in multiclass classification, calculating the probability that each occurrence belongs to a predetermined class and correcting the output values for each class so that they are in the range from 0 to 1. The softmax function is usually used only for the output layer.

Convolutional neural networks (CNN) [71] (Figure 1.9) are one of the popular and often used types of NN, which have gained great popularity due to their use in classification and image recognition tasks. They have become more popular due to their use in image classification, where the filter moves through the image. CNNs are also common in speech recognition, natural language processing, and SA. When working with text data, it is required to remember that words have different lengths, and in a vector representation, they must be brought to the same dimension. Word occurrences such as Word2Vec, Glove, and FastText are commonly used for vector transformation. Each word is converted to a vector of the E size. If there are S words, then a matrix of $S \times E$ size is formed. Then a filter $C_1 \times E$ is applied to the matrix, and the scalar product of the matrix with the filter elements is performed. The resulting value is transferred to the next layer. Transformations result in feature vectors of size $S \times 1$. On the next layer, the max-pooling process is typically applied to the resulting feature vectors to extract the largest value from each vector. Finally, the vector is passed to a classifier, for example, softmax, which reduces the dimension to $m \times 1$, where m is the number of classes of the dataset.

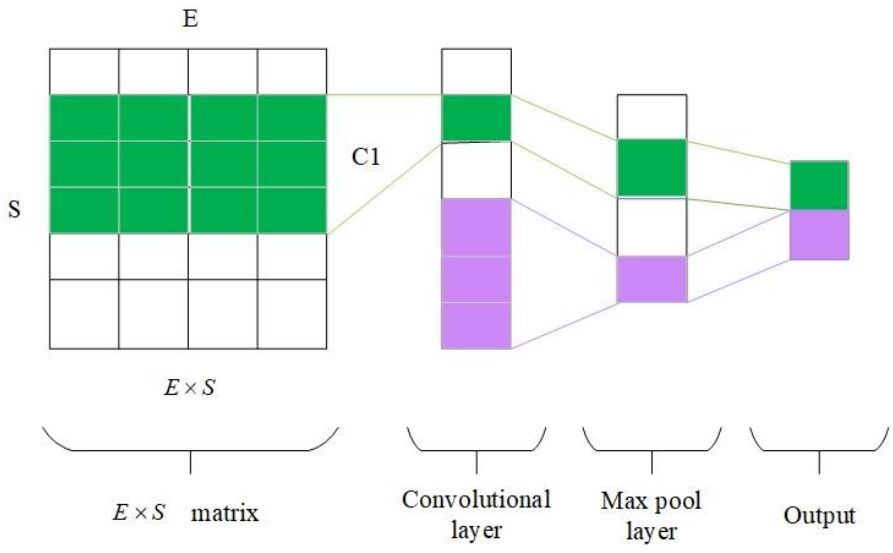

Figure 1.9 – Convolutional neural network

Recurrent neural networks (RNN) [72] are a type of neural network that sequentially processes data and generates current conclusions based on previous calculations. In a given neural network, the latent state h_t is determined using input data x_i at the current time t and represents previous latent states h_{t-1}. The expression h_t can then be evaluated as (1.9):

$$h_t = \varphi(W^{t-1}h_{t-1} + W^t x_t), \tag{1.9}$$

where W^{t-1} and W^t are the weights of the previous hidden state h_{t-1} and the current input x_t. The output y_t is given by the following equation (1.10):

$$y_t = \varphi(W^y h_t). \tag{1.10}$$

The latent state h_t is represented as memory in the network, and the result y_t depends only on the memory h_t at a time t and the matrix of weights W^y.

One of the most popular models of RNN is the Long short-term memory (LSTM) architecture (Figure 1.10). The LSTM structure includes four main components: an input element, an output element, a forgetting element, and a memory cell. Initially, at a time step t, the activation function gate φ decides on the information to be written to the cell state. It takes input values x_t and gets the output of the previous hidden state h_{t-1}. The following formula is used to calculate the probability of storing the current information (1.11):

$$\varphi_t = \rho(W^{\phi}x_t + U^{\varphi}h_{t-1}). \tag{1.11}$$

Subsequently, the LSTM decides which new information should be stored in the cell state. In addition, the input gate (sigmoid) determines the values to be updated based on equation (1.12):

$$i_t = \rho(W^i x_t + U^i h_{t-1}). \tag{1.12}$$

On the other hand, the *tanh* function generates a candidate vector $\overline{C_t}$, which then updates the current state of the cell (1.13):

$$\overline{C_t} = \tanh(W^n x_t + U^n h_{t-1}). \tag{1.13}$$

At the output, the sigmoidal function decides which part of the state of the cell should be returned using equation (1.14):

$$o_t = \rho(W^o x_t + U^o h_{t-1}). \tag{1.14}$$

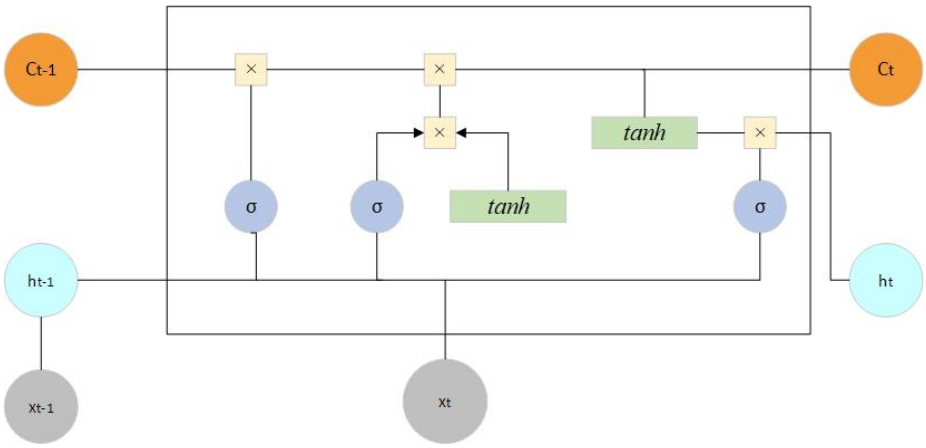

Figure 1.10 – Recurrent neural network

Bidirectional Encoder Representations from Transformers (BERT) [73] is a Transformer neural network represented by a Bidirectional encoder in which each output element is connected to an input element. BERT was introduced by Google in 2018 for the best results in the field of NLP. This model processes texts in both directions, capturing context and understanding the ambiguity of words and phrases. The neural network has been trained on colossal text corpora like Wikipedia, Brown Corpus, etc. BERT is effectively used in the tasks of text classification, sentiment analysis, word ambiguity elimination, question-answering systems, chatbots, text translators, etc. A pre-trained BERT model configuration step called Fine-tuning is performed to solve a specific problem. In this work, the pre-trained BERT model is configured to determine the sentiment of texts from the social media space. BERT is shown in Figure 1.11.

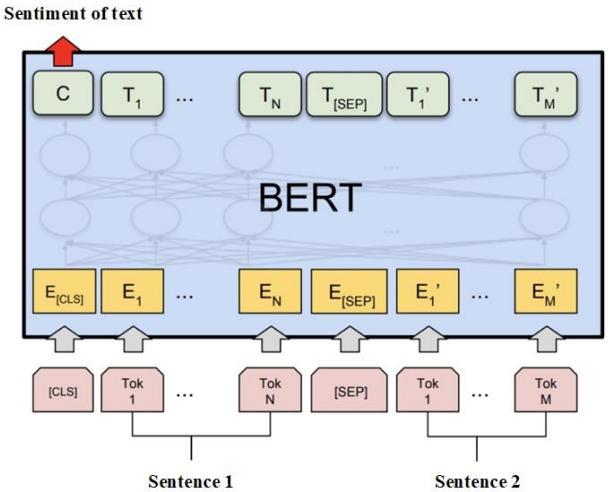

Figure 1.11 – Classification with BERT

1.3 Analysis of marketing technologies of social media platforms

In the marketing part of the social media space [74, 1, p. 22], social media indicators play a large role in marketing management. Brand efficiency is based on socioeconomic effectiveness. This approach is used to evaluate the effectiveness of marketing costs. The efficiency itself is determined by many characteristics and assessments that show the company's level of achievement of marketing tasks. The practice shows that involving customers in interaction with the brand is very important in the marketing plan.

An indicator of the level of audience engagement plays a special role. The number of likes, reposts, and user comments is counted to determine its value. The number of subscribers on social networks is also taken into account. Engagement helps you effectively evaluate marketing campaigns and their effectiveness based on user feedback and suggestions for improving products and services. There are several types of user engagement levels:

– Total engagement level is defined as the ratio of the total number all likes, comments, and reposts to the number of subscribers (1.15):

$$R_{CE}\_general = \frac{L + R + C}{CS} \times 100\%, \tag{1.15}$$

where L is the number of likes; C is the number of comments; R is the number of reposts; CS is the number of subscribers.

– The average level of engagement is the ratio of the sum of likes, comments, and reposts to the product of the number of subscribers and posts (1.16):

$$R_{CE}\_average = \frac{L + R + C}{CS \times CP} \times 100\%, \tag{1.16}$$

where L is the number of likes; C is the number of comments; R is the number of reposts; CS is the number of subscribers; CP is the number of texts found on a specific topic.

– The daily engagement level is defined in the same way as the overall engagement level, with only the difference that it is calculated for each day.

Currently, there is no single generally accepted way to calculate the level of engagement. The formula may vary depending on the requirements of social analytics and methods for analyzing the reaction of a user audience.

Similar to the number of engagements, coverage by the number of record views is also used (1.17):

$$Engagement\_Rate\_View = \frac{Engagement\_Rate}{Views} * 100\%, \tag{1.17}$$

where $Engagement\_Rate$ is the engagement level; $Views$ is the number of views.

Other metrics, such as attractiveness, sociability, and coverage, are also applied. The attractiveness level is defined as the ratio of the number of likes to the number of subscribers (1.18):

$$Attractiveness\_Rate = \frac{L}{CS} * 100\%, \quad (1.18)$$

where L is the number of likes; CS is the number of subscribers.

The sociability level is calculated as the ratio of the number of comments to the number of subscribers (1.19):

$$Sociability\_Rate = \frac{C}{CS} * 100\%, \quad (1.19)$$

where C is the number of comments, CS is the number of subscribers.

Coverage determines the number of users who have at least one contact with a publication.

1.4 Conclusions on Chapter 1

Chapter 1 discusses the main tasks of analyzing and monitoring the social media space. Since news portals and social networks contain a large amount of information presented by world and Kazakh news and also act as a free platform for expressing user opinions, their research has a very important aspect in the formation of analysis of the mood of society on various relevant topics.

Monitoring the social media space includes gathering and structuring data as the main tasks. The data is collected from various sources such as news portals, social networks, and group social media accounts. To accumulate data from Internet sources, various approaches are used, such as manual monitoring (rather long and painstaking work), the use of small programs for parsing data, and the development of complex analytical platforms that perform not only the collection but also data analytics on a set of various aspects. Since great emphasis in the analysis is placed on the emotional features of texts, the sentiment analysis of texts and user comments is very important. At the analysis stage of the received data, the number of collected texts and user comments, statistical and structural patterns of the study area, and special patterns of narrow subject areas is determined. Sentiment analysis, in most cases, is performed at the level of a document, sentence, paragraph, phrase, or single word. The main approaches for determining sentiment are vocabulary, ML, and NN approaches. The lexicon-based approach is mainly aimed at building a sentiment dictionary of words. In it, each word is labeled according to the sentiment class. Most often, 2 or 3 classes of sentiment are distinguished: binary (positive or negative) and multiclass (positive, negative, or neutral). The generated dictionary is further replenished by searching a large dictionary of

synonyms and antonyms, such as WordNet or any other. The largest number of words of a certain class determines the sentiment of a document in most cases. Since the manual formation of a dictionary requires a lot of work, constant replenishment of the base of sentiment words, ML, and NN approaches are now widely used.

In this chapter, the mechanisms of work of the following most popular machine learning algorithms were discussed in detail: NB, LR, SVM, k-NN method, DT, RF, and XGBoost. In addition, the following types were presented in the description of NN: DNN, CNN, and RNN.

2 DEVELOPMENT OF THE OMSYSTEM MONITORING DATA PROCESSING AND ANALYSIS MODULE

2.1 The purpose and objectives of the data processing and analysis module

The opinion monitoring information system OMSystem has been developed as part of the project for the commercialization of the results of scientific and (or) scientific and technical activities "Opinion Monitoring Information System OMSystem (Opinion monitor system)," 0101-18-GK. The main work of the dissertation student was to develop a module for processing and analyzing data, models for determining the social mood of society, conducting an experiment to analyze public opinion on the topic of vaccination against coronavirus infection, and developing an electronic Social Mood module using the OMSystem text database.

The OMSystem platform monitors web resources and social networks. It has very wide functionality, including the calculation of social well-being, determining the sentiment of Russian and Kazakh texts using dictionaries, machine learning models, and neural networks, and building social analytics using marketing technologies. OMSystem covers many media space resources in Kazakhstan, including all popular and widely used news sites and social networks such as Vkontakte, Facebook [75], Twitter, Instagram, YouTube, etc. The system allows you to search and analyze society's most important and relevant topics [76, 77], choosing a time range (day, week, month, half a year, year, etc.). OMSystem builds beautiful visual reports through graphs and charts (histograms, piechart, bubble charts, etc.). At the same time, the platform provides ways to identify the profile of a social network participant by reading the data and counting the participant's activity on the topic by the number of comments, likes, and reposts.

2.2 Designing the architecture and functionalities of the data processing and analysis module

The development of OMSystem includes several critical stages to achieve all the goals set. In the first stage, an API module is created that allows you to connect to social networks and a database for storing the results obtained using the system's parsers [1, p. 9]. Then the sentiment dictionaries in the Russian and Kazakh languages were designed to evaluate the sentiment on the analyzed topics. The SA module was further extended with ML modules trained on the texts labeled by human annotators, sentiment dictionaries, and marketing technologies of social analytics. Information in the system is presented using convenient graphical data visualization. An extended role-playing policy also significantly impacted the system's functionality. Finally, the interface and design of the system have been improved to meet the modern demands of web application development.

The administration subsystem is responsible for setting up social networking APIs, search robots, servers, databases, and resource directories. The main data sources are news portals, blogs, and social networks. The linguistic constructor includes

Kazakh and Russian sentiment dictionaries containing words belonging to the positive, negative, and neutral classes. The data analysis and processing module uses machine learning and neural network models to label texts with sentiment and marketing tools for social analytics. As a result, the obtained quantitative analysis of the system operation makes it possible to use machine learning and neural network models to determine the sentiment of texts and comments in social networks, to model the social mood of society using a production model, and to visualize data in the form of reports, graphs, and charts. The machine learning module uses algorithms that demonstrate the best data classification results regarding the accuracy, precision, recall, and F1-score metric values. The system uses the following machine learning algorithms: NB, LR, SVM, k-NN, DT, RF, and XGBoost, and neural networks: DNN, CNN, and RNN. Marketing indicators are also used to determine the mood of the society in terms of socio-political content using analytical formulas for the level of interest in the topic in society, the level of activity of the topic's discussion in society, and the level of social mood. The general view of the client-server architecture of the system is shown in Figure 2.1. The system's functionality components include:

– Data sources: They are represented by news portals, blogs, and social networks.

– Connector module: It connects to data sources.

– The linguistic constructor module: It contains Russian and Kazakh sentiment dictionaries, which include positive, negative, and neutral words.

– Data analysis and social analytics module. It defines social mood with sentiment dictionaries, ML models, and marketing technologies.

– Results module: It forms reports, tables, and graphics of data social analytics results.

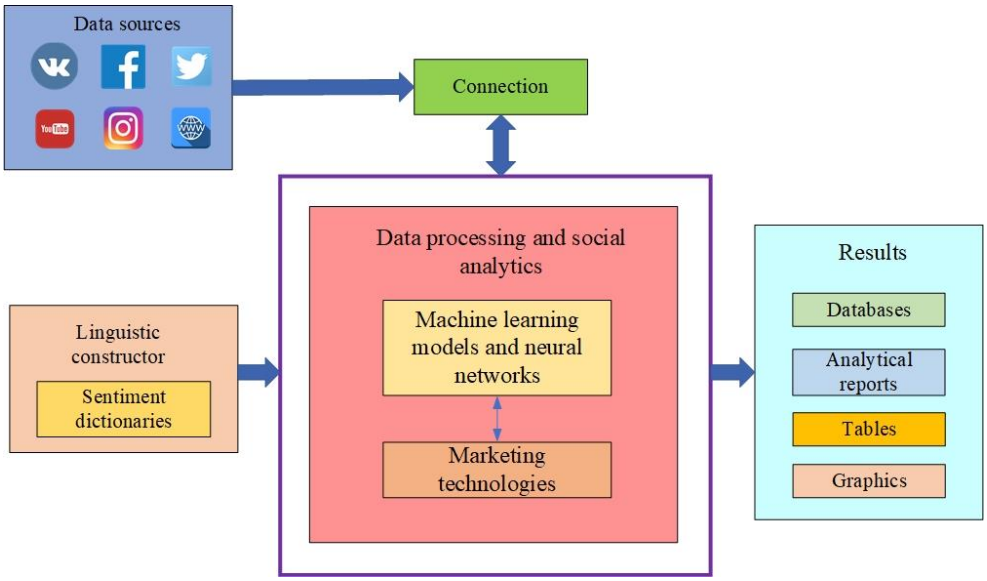

Figure 2.1 – OMSystem architecture

The OMSystem includes flexible functionality for creating search mechanisms for news portals and social networks [78-80], sentiment dictionaries, modeling social mood and well-being (socio-economic indicators and calculation of social well-being), and visualization of the results using tables, graphs, and histograms.

The functional model aims to represent business processes using natural and graphical languages with the IDEF0 methodology. This methodology represents the creation of hierarchical diagrams that comprise separate fragments of the system. First, a general description of the system is given (context diagram), then decomposition is performed, in which the entire system is divided into subsystems (decomposition diagram). Further, the subsystem is divided into even smaller subsystems, which continue until the required level of detail is achieved. IDEF0 has blocks and arcs. Blocks describe the system's functions, and arcs show their connections. Function blocks are shown as rectangles of named processes and tasks. The sides of the blocks have their own special purposes. The left side is for inputs; the right side is for outputs; the top is for control; the bottom is for mechanisms.

There are five types of arrows in IDEF0:

1. Input – objects that are used to obtain the result. The entry arrow is drawn as entering the left side of the job.

2. Control – information that controls the actions of the work. These arrows represent information indicating what the job is doing.

3. Output – objects into which the inputs are converted. Each job has at least one exit arrow, which is drawn as coming from the right side of the job.

4. Mechanism – the resources that perform the work. The mechanism arrow is drawn as entering the bottom face of the work.

In the IDEF0 model of the OMSystem, the input data are search parameters in the social media space: name, category, subcategory, period, keywords, and resource list. Control data is represented by API configuration, sentiment dictionaries, ML algorithms, and data visualization libraries. The mechanisms are system users and administrators and "system monitoring," which determines how data is presented and stored. The output is generated reports, graphs, and data visualization diagrams. The IDEF0 model of the system is shown in Figures 2.2 and 2.3.

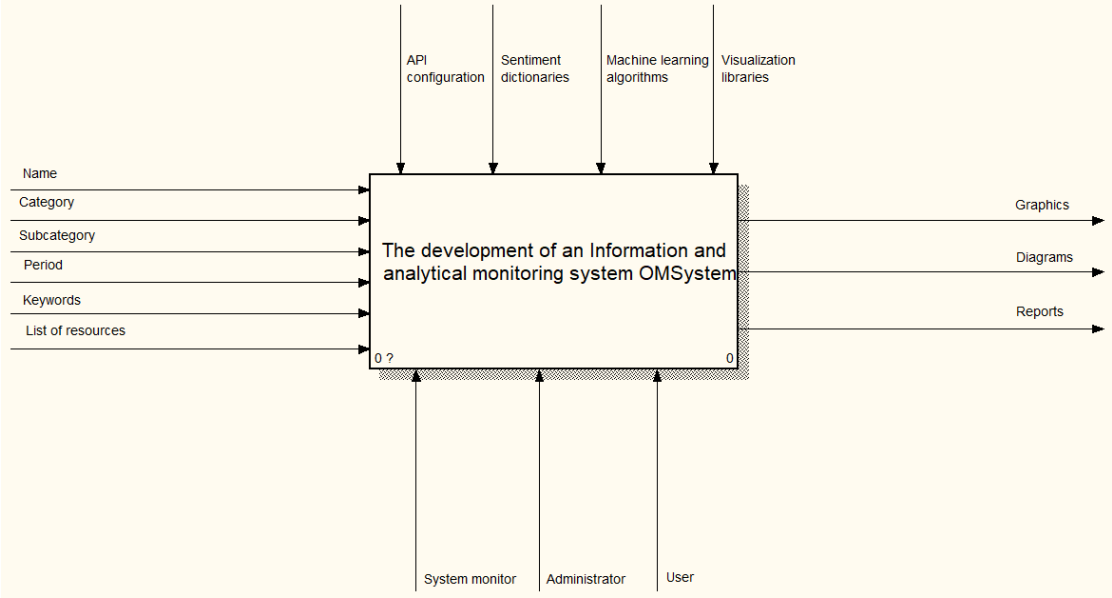

Figure 2.2 – The IDEF0 diagram of the OMSystem

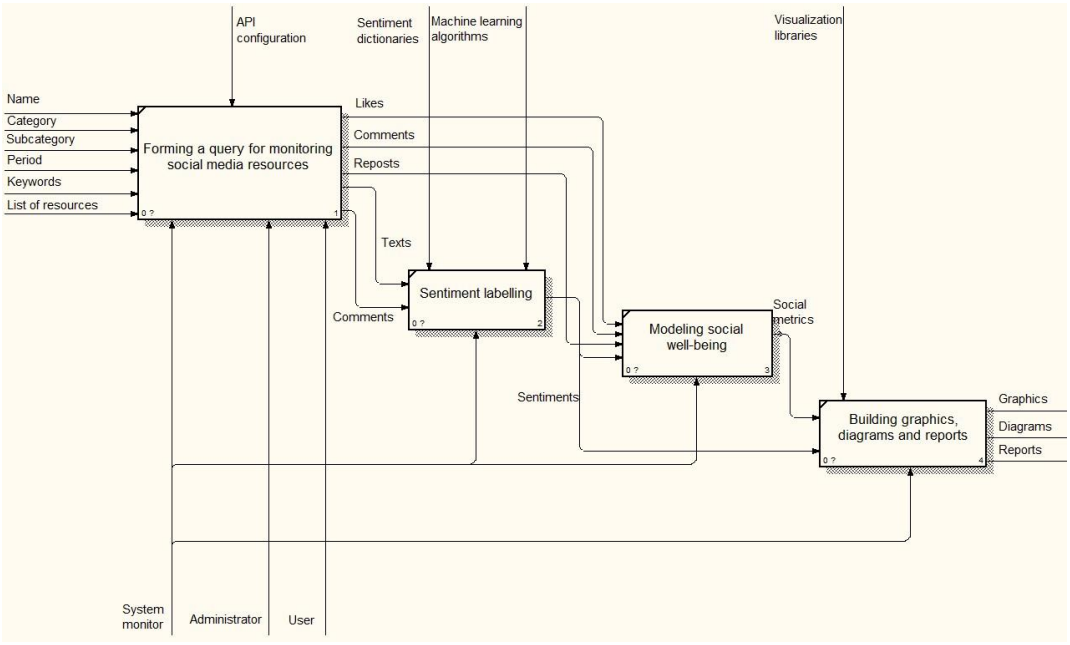

Figure 2.3 – The OMSystem model decomposition

The Infological design provides a formal, human-readable description of the field of study and the information stored in the database. The infological model of the system database provides access to data for all modules of the system. However, there are restrictions on the interaction (reading/writing/deleting/editing) of different tables for each module. When designing the system, the database levels were set: the external level of access for system users to interact with data, which is mainly represented by authorizing users and gaining access to data; the internal level that processes data and issues the results of the system; the conceptual layer that provides independence between layers for presenting data from the inner layer to the outer layer. The main functionality of the data processing system is presented in the form of a Data Flow Diagram (DFD) in Figure 2.4.

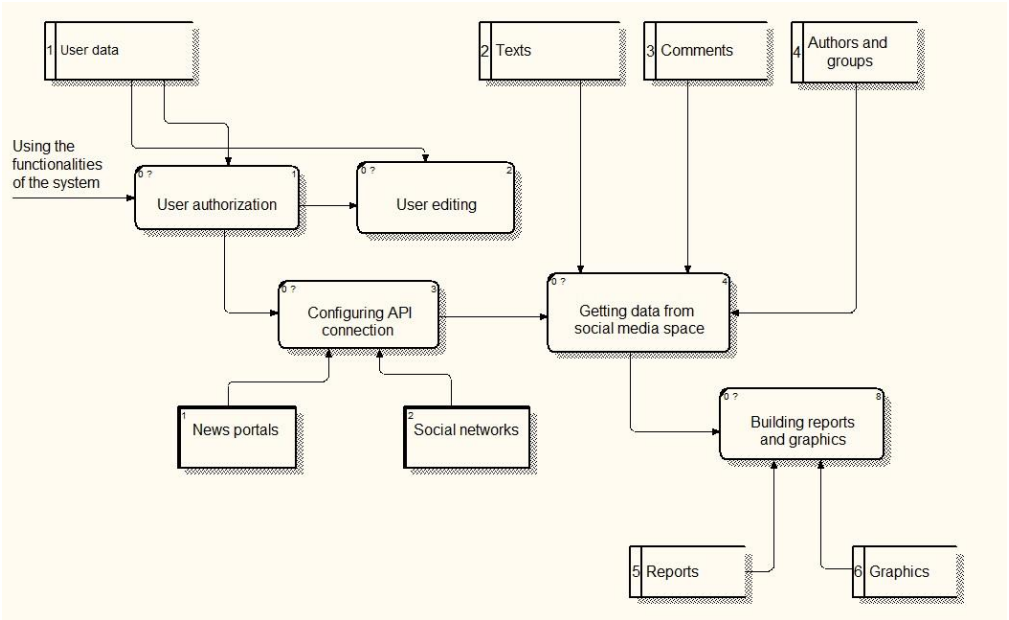

Figure 2.4 – The DFD diagram of the OMSystem

The OMSystem has certain technical requirements to keep the system stable. It is necessary to select the appropriate technical equipment that supports many simultaneous users to maintain the system's stable operation. Also important are the computing abilities of the server, which determine how many requests it can process per unit of time. The speed of the web crawler is also essential for quickly replenishing the database with texts and comments from news portals and social networks.

Thus, the following features are important for the work of the system project:

– The number of requests/responses from social networking APIs.
– The number of requests/responses from the database.
– Multiprocessor search engine startup.

The server on which the system design is published has the following technical characteristics: 16 GB of RAM, 8-core processor Core i7 10700, 1 TB HDD, and RTX 2080 graphics card.

The following steps are performed to publish a system project:

a.) Performing server configuration. Registering a user with a high level of access as a root user is required to ensure full and secure server operation.

b.) Developing a new project. Because the project is in a closed repository, it is necessary to install the Git package and copy the project from the repository.

c.) Performing database configuration. It is required to back up the current local project database and place it on the server to associate a project with a database on the server. The changes are to be made in settings.py project file.

d.) Publishing project to the server. The Nginx + Uwsgi + Postgresql + Supervisor bundle is used to publish Django projects to the server. Here Nginx is used as a web server that meets static content, implements the Uwsgi communication protocol, and is responsible for dynamic Python content. The main DBMS is Postgresql.

e.) Starting server with the project. The project starts in debug mode from port 80 or 443 (depending on the SSL certificate).

2.3 Development of computational kernel models and algorithms

SA models [81, 82] and algorithms [83, 84] allow for determining the sentiment of words, phrases, paragraphs, and documents. The OMSystem uses a lexical approach based on sentiment dictionaries, ML algorithms, and NN. The sentiment dictionary effectively labels texts obtained using a web crawler from news portals and social networks.

The algorithm for determining the sentiment of texts using a dictionary is presented below:

1. Enter text – T, enter sentiment dictionary – D.
2. Split T into a list of words – w_list = T.split().
3. Create three-word containers – w_pos_list, w_neg_list, w_neut_list.
4. Determine if the words from the text T are positive, negative, or neutral and add them to the appropriate lists

```
for w in w_list:
   for key, value in D.items():
  if w == key:
```

```
if value == ’positive’:
    w_pos_list.add(w)
if value == ‘negative’:
   w_neg_list.add(w)
else:
 w_neut_list.add(w)
```

5. Calculate the sentiment of the text by the maximum number of words of a certain sentiment

$$S_t = Max(len(w\_pos\_list), len(w\_neg\_list), len(w\_neut\_list))$$

For effective text processing, sentiment dictionaries must include various word forms. At the same time, user comments contain many grammar, spelling, and punctuation errors, which require the expansion of dictionaries by adding misspelled words. In the OMSystem, the sentiment dictionaries were developed in the following steps:

1. Forming a sentiment vocabulary, which is labeled on the basis of feelings and emotions. The sentiment dictionary consists of such elements as words, phrases, misspelled words, and slang forms of words, each of which has its own emotional aspect.

2. Creating words with errors in the Russian and Kazakh languages will increase the search results. The words with errors are formed by substituting, inserting, and deleting symbols. Examples of such errors in the Kazakh language are letter substitutions, such as ә → a, і → и, ң →н, ғ → г, ү → у, ұ → у, қ → к, ө → о and others. The words “отыргызу,” “отырыкшы,” “турмыс,” “турли,” etc. are added to the sentiment dictionary. In Russian words, various errors are also present. The most frequent word substitutions are о → а, е → и, у → ю, и → ы, ч → щ, etc. The following words, “ужосы,” “сщастье,” “заказщик,” “бугалтер,” etc. are included in the sentiment dictionary.

3. Filling the generated dictionary with new words using the NRC Emotion Lexicon, which is in the public domain. The description of this dictionary indicates that it is suitable for more than 100 languages, including Russian and Kazakh. Furthermore, it only requires the use of available translators. Among the most popular and advanced is Google Translate. The link to this sentiment dictionary is given here (https://public.tableau.com/views/NRC-Emotion-Lexicon-viz1/NRCEmotionLexicon-viz1?:showVizHome=no).

Currently, in the OMSystem, the Russian sentiment dictionary includes 44381 words, and the Kazakh sentiment dictionary includes 29654 words. The sentiment dictionaries in the OMSystem database are shown in Figures 2.5 and 2.6.

id integer	word text	name text
710	беженцев	Отрицательная
711	бежит	Нейтральная
712	без	Отрицательная
713	безбожный	Отрицательная
714	безболезненно	Положительная
715	безболезненный	Отрицательная
716	безвкусный	Отрицательная
717	безвозвратно	Отрицательная
718	безвредный	Положительная
719	безвременно	Отрицательная
720	безвыходное положение	Отрицательная
721	безвыходность	Отрицательная

Figure 2.5 – The Russian sentiment dictionary

id integer	word text	name text
25906	бақылау тізімі	Положительная
25907	бақылаусыз	Отрицательная
25908	бақылаушы	Нейтральная
25909	бақылдады	Нейтральная
25910	бақыраш	Нейтральная
25911	бақыру	Отрицательная
25912	бақыт	Положительная
25913	бақытгүл	Нейтральная
25914	бақытсыз	Отрицательная
25915	бақытсыздық	Отрицательная
25916	бақытты	Положительная

Figure 2.6 – The Kazakh sentiment dictionary

In addition to sentiment dictionaries, ML algorithms are used to label text data. The OMSystem uses the algorithms NB, LR, SVM, k-NN, DT, RF, and XGBoost described in section 1.3.2. Machine learning algorithms are aimed at strengthening the OMSystem analytical module. Well-trained models will make it possible both to obtain high-quality labeling by sentiment classes and, in the future, to eliminate the labor-intensive sentiment dictionary approach that requires constant replenishment with new lists of words.

The algorithm for determining the sentiment of texts using machine learning models is presented below:

1. Enter a set of texts – T_list.

For each text T, perform the following steps.

2. Remove extra characters with a regular expression

T = re.sub('[^a-zA-Za-яA-ЯӘІҢҒҮҰҚӨәіңғүұқөh]', ' ', T)

3. Tokenization of T

$$words = word\_tokenize(T)$$

4. Eliminate stop-words

$$for\ w\ in\ words:$$

$$if\ w\ not\ in\ stop\_words:$$

$$words.remove(w)$$

5. Do stemming of words

$$words\_stem = []$$

$$for\ w\ in\ words:$$

$$w = stem(w)$$

$$words\_stem.append(w)$$

6. Combine words in a text

$$T = \text{‘ ’}.join(words\_stem)$$

7. Vectorize texts

$$T\_vect\_list = vectorize(T\_list)$$

8. Split texts into training and testing parts

$$T\_train,\ T\_test = train\_test\_split(T\_vect\_list)$$

9. Classify with machine learning algorithms

$$ml\_classifier.fit(T\_train)$$

$$S_t = ml\_classifier.predict(T\_test)$$

The database of ML-labeled texts is shown in Figure 2.7.

	id integer	name text	name text	text text
448	108615	Русский	Положительная	Распоряжением акима города Алмас Батанов назначен руководителем управления стратегии и бюдж
449	115882	Казахский	Положительная	Қазақстан президенті Қасым-Жомарт Тоқаев 29 маусымда видеоконференция кезінде үкімет мүшел
450	92612	Русский	Положительная	НУР-СУЛТАН. КАЗИНФОРМ – Минсельхоз РК планирует с 1 июня снять все ограничения и квоты на в
451	103575	Казахский	Положительная	Алматы облысында сауда үйлері қайтадан жабылуда. Қоғамдық шаралар өткізуге қатаң тыйым сал
452	85252	Русский	Положительная	В Мактааральском районе снизился уровень воды. Число вернувшихся домой жителей выросло до 2
453	85253	Русский	Отрицательная	В Алматы по программе "Дорожная карта занятости" уже создано 468 рабочих мест по 12 проектам.
454	85259	Русский	Положительная	В Алматы начали обработку парков и скверов от клещей, чтобы обезопасить горожан от укусов насе
455	85254	Русский	Положительная	Авиакомпания FlyArystan сообщила о приостановлении авиасообщении в Тараз из Алматы и Нур-Сул
456	85255	Русский	Положительная	Теперь всем, кто въезжает или выезжает из Алматы нужно иметь отрицательный результат ПЦР-тес
457	115883	Русский	Положительная	Предварительное слушание по делу активистки Асии Тулесовой назначено на 7 июля
458	85256	Русский	Положительная	В отчете разведчиков сказано, что причиной мог стать коронавирус
459	115884	Казахский	Отрицательная	ДТП в Сатпаеве
460	85257	Русский	Положительная	Полицейские Алматы посвятили песню медицинским работникам и военнослужащим, задействован
461	85258	Русский	Положительная	Сегодня в акимате Нур-Султана состоялась торжественная презентация книги «Великая Победа в па
462	92613	Русский	Положительная	НУР-СУЛТАН. КАЗИНФОРМ – Министр индустрии и инфраструктурного развития РК Бейбут Атамкуло
463	85260	Русский	Положительная	В Алматинской области продолжается работа блокпостов. Сотрудники правоохранительных органов

Figure 2.7 – Labeled texts

Although ML algorithms show excellent results in classifying text data, as the size of the data set increases in the system, DL methods based on NN become more relevant. They show high efficiency, and the OMSystem uses models based on DNN, CNN, and RNN.

The algorithm for determining the sentiment of texts using neural networks is presented below:

1. Enter a set of texts – T_list.

For each text T, perform the following steps.

2. Remove extra characters with a regular expression

T = re.sub('[^a-zA-Za-яA-ЯӘІҢҒҮҰҚӨәіңғүұқөh]', ' ', T)

3. Tokenization of T

$words = word\_tokenize(T)$

4. Eliminate stop-words

$for\ w\ in\ words:$

$if\ w\ not\ in\ stop\_words:$

$words.remove(w)$

5. Do stemming of words

$words\_stem = []$

$for\ w\ in\ words:$

$w = stem(w)$

$words\_stem.append(w)$

6. Combine words in a text

$T = \text{' '}.join(words\_stem)$

7. Vectorize texts

$T\_vect\_list = vectorize(T\_list)$

8. Split texts into training and testing parts

$T\_train,\ T\_test = train\_test\_split(T\_vect\_list)$

9. Classify with neural networks

$neural\_classifier.fit(T\_train)$

$S_t = neural\_classifier.predict(T\_test)$

The OMSystem enables a multifaceted analysis of texts in the media space of Kazakhstan using ML models and NN to determine social mood and evaluate society for various events. Social media marketing management measurement models known as Social Media Marketing Management (SMMM) are used to accomplish these tasks.

As part of this work, calculations were presented based on these indicators using formulas that are the author's methodology [1, p. 23]:

– The level of interest in the topic (R_{CT});

– The level of topic discussion (R_{CE});

– The level of social mood (R_{TS}).

The level of interest in the topic R_{CT} is calculated using as (2.1):

$$R_{CT} = \frac{CT \times 100\%}{\max_{CT}}, \tag{2.1}$$

where CT is the number of texts or comments found on a specific topic; $\max_{CT}$ is the maximum number of texts or comments on a certain topic (set by the expert for a specific time). The value range starts at 0% and is not limited. If the value exceeds 100%, then this topic is of great interest.

R_{CE} defines interaction in social networks and shows the level of activity in discussing a topic in society. This indicator assesses how differently the audience responds to categories of events in society. It is calculated by the formula (2.2):

$$R_{CE} = \frac{L + R + C}{CS \times CP} \times 100\%, \tag{2.2}$$

where CS is the sum of the number of subscribers; CP is the number of found texts on a certain topic; C is the number of comments; L is the number of likes; R is the number of reposts. The value range starts at 0% and is not limited. Since each news portal or group in a social network has many and all users and subscribers cannot discuss all of them, the level of activity in discussing the topic is usually small.

R_{TS} is the level of social mood, which is determined by the maximum value of the sum of positive, neutral, and negative texts or comments on a certain topic (2.3):

$$R_{TS} = Max < CP\_pos, CP\_neg, CP\_neut >, \tag{2.3}$$

where CP_pos is the number of positive texts; CP_neg is the number of negative texts; CP_neut is the number of neutral texts.

2.4 Software implementation of the data processing and analysis module

2.4.1 Preprocessing, vectorization, and class balancing

Software implementation of the data analysis and processing module includes the construction of an ML module and a module for analysis of the social mood of society. The ML module includes data preprocessing, vectorization, class balancing, and classification. The Python programming language is best suited for the implementation of classification, with easy and convenient syntax, wide application, and the presence of a large number of libraries, especially for data processing and ML tasks. The text base, replenished with a web crawler, includes raw data. Two tasks, binary and multiclass classification, were completed. In the binary classification, texts and user comments were labeled by experts. The datasets are presented in Tables 2.1 and 2.2.

Table 2.1 – Texts

Language	Positive	Negative
Russian	520	520
Kazakh	217	250

Table 2.2 – Comments

Language	Positive	Negative
Russian	3757	3757
Kazakh	173	173

Python programming language, which has a large number of NLP libraries, especially NLTK and ML libraries, was chosen to solve the problem of building a data processing module [85]. NLTK is one of the leading libraries for creating programs in the field of NLP. It runs on Windows, Linux, and macOS operating systems. Its main advantage is free distribution, open source, good documentation, and an active community. As a result, it is easy to find answers to most of the problems and errors on websites and forums.

The received texts and user comments must be processed, vectorized, and classified using ML algorithms. All words are translated into lowercase at the preprocessing stage, and extra words, characters, punctuation marks, and links are deleted. Then it is also necessary to delete the so-called stop words, which are words that do not carry much semantic content. Examples of such words are prepositions, unions, pronouns, etc. (“on,” “in,” “all,” “and,” “but,” and others). Another important step is to reduce the number of words with similar meanings. These methods are called stemming and lemmatization. The first method removes affixes and word endings to obtain the root part. In lemmatization, the words are given in an indeterminate form. Stemming is a simpler way to write an algorithm for deleting parts of a word [1, p. 17].

On the contrary, Lemmatization requires significant efforts to develop rules for bringing words to infinitive form. The NLTK library includes excellent stemmers for Russian and English. Unfortunately, it does not yet contain the same developed stemmer for the Kazakh language. Thus, its own “KazakhStemmer” stemmer was written when developing the program. The implementation of preprocessing of text data in Python and using the NLTK library is shown in Figure 2.8 [1, p. 17].

After preprocessing of texts, a vectorization step is performed, where the Bag of Words (BOW) and term frequency-inverse document frequency (*tf-idf*) [86] algorithms are widely used. The Bag of Words method is a simple and convenient way to extract features. It does not take into account the order, structure of words, and other features of the text. This method simply determines the presence or absence of each known word in the document. The dictionary of words is made up of all the words found in all documents [1, p. 17]. For example, given a number of documents and their corresponding vector representations:

– I read – [1, 1, 0, 0, 0]
– I read the book – [1, 1, 1, 0, 0]
– I read the book at the table – [1, 1, 1, 1, 1]

Vectorization involves counting the number of words in each document (Table 2.3).

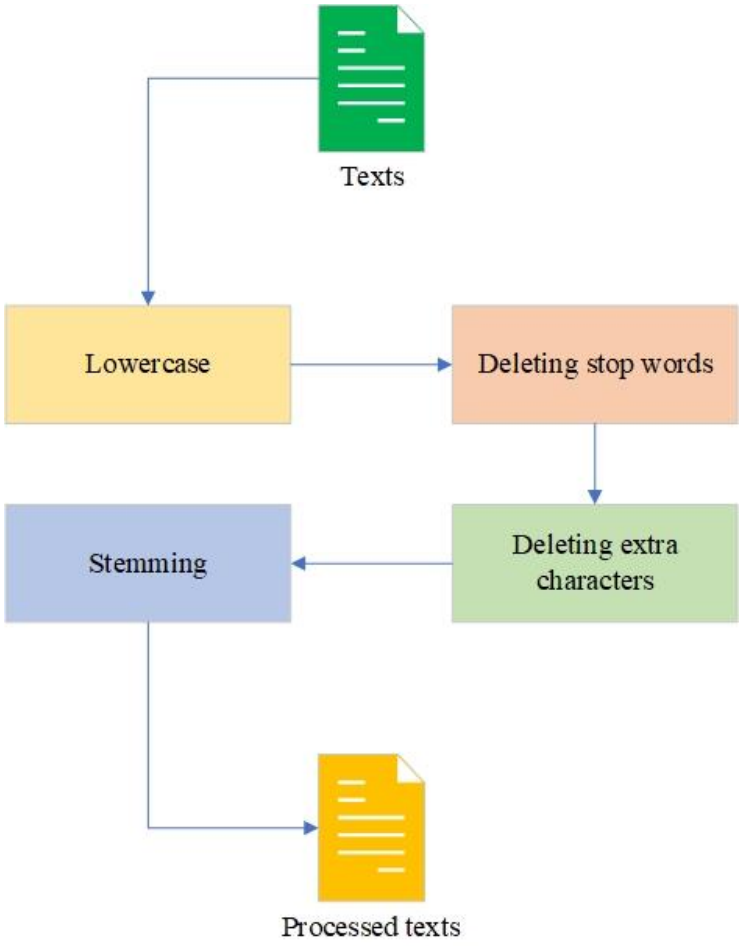

Figure 2.8 – Preprocessing of text data

Table 2.3 – Vectorization with BOW

Documents	I	read	the	book	on	table
I read	1	1	0	0	0	0
I read the book	1	1	1	1	0	0
I read the book on the table	1	1	2	1	1	1

Despite its simplicity, the BOW method has a significant drawback associated with an increase in the size of the vectors in the case of a large number of documents and the presence of many zeros. To solve this problem, the *tf-idf* method, which is a statistical measure that evaluates the importance of a word in a document, is used. The weight of a word is proportional to the frequency of its use in the document and inversely proportional to the frequency of its use in other documents in the collection.

Tf measures how often a particular word occurs in a document. It is possible to determine how important a word t_i is in a specific document (1.1):

$$tf(t,d) = \frac{n_i}{\sum_{i=1}^{k} n_i}, \tag{1.1}$$

where n_i is the number of occurrences of the word in the document, and $\sum_{i=1}^{k} n_i$ is the total number of words in the document.

Idf measures how common a word is in all the documents. Rare words get a higher weight. *Idf* is calculated by the formula (1.2):

$$idf(t,D) = \log \frac{|D|}{|(d_i \supset t_i)|}, \tag{1.2}$$

where $|D|$ is the number of documents in the corpus; $|(d_i \supset t_i)|$ is the number of documents where t_i occurs.

After calculating the values, *tf* and *idf* parts are multiplied (1.3):

$$tf\text{-}idf = tf \times idf \tag{1.3}$$

This experimental part used a *tf-idf* metric to vectorize text data [1, p. 18].

When training classifiers, the imbalance of classes for binary and multiclass classification causes a big problem. The good value of the correct classification is achieved by the fact that all instances are labeled with the most represented class. However, poor accuracy, precision, recall, and F1-score results are caused by the erroneous classification of the less-represented classes. Therefore, class balancing methods, such as random undersampling, random oversampling, and synthetic minority oversampling technique (SMOTE) [1, p. 18-19], are used to solve this problem.

The random undersampling technique results in the size of the smallest class, discarding some of the data of the largest classes. Although classes are balanced, a significant disadvantage of such a method will be discarding valuable information in other classes. In the random oversampling technique, on the contrary, the sizes of all classes are adjusted to the largest class by repeatedly copying its data. Although all valuable information remains, it will only be duplicated without adding new data. The random undersampling and oversampling techniques are shown in Figure 2.9.

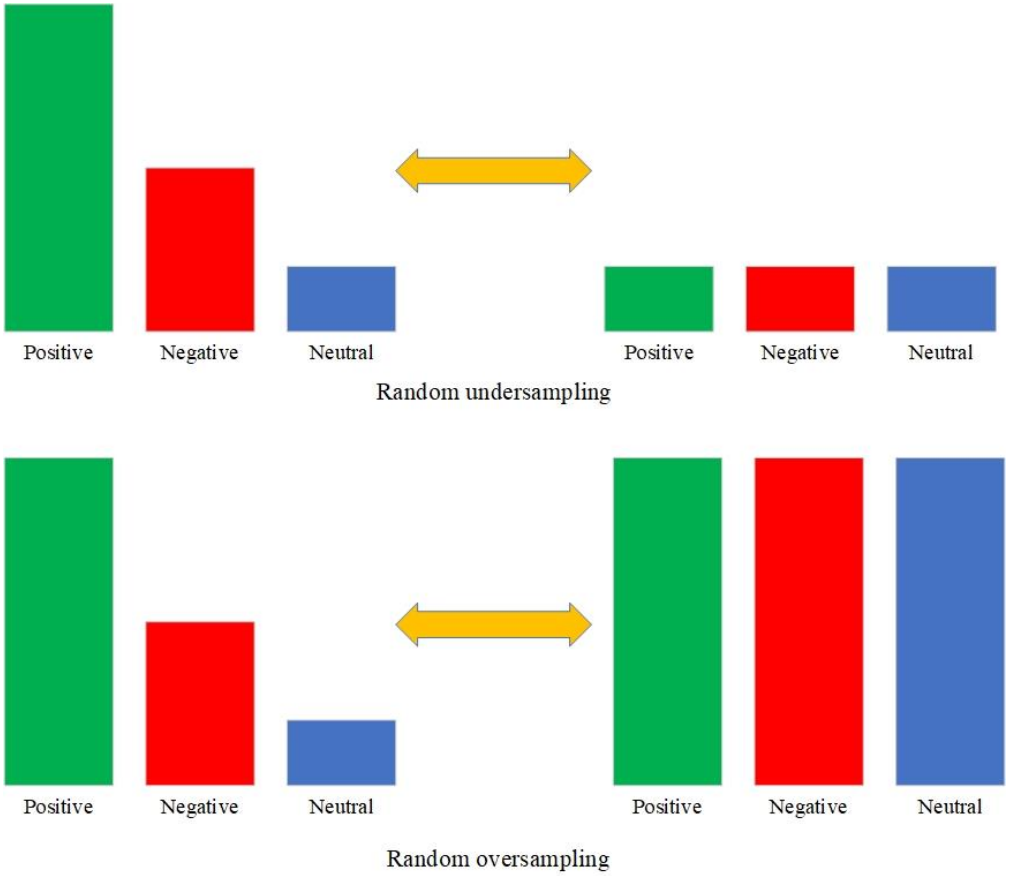

Figure 2.9 – Class balancing – (a) random undersampling and (b) random oversampling

In the SMOTE method (Figure 2.10), new points are synthesized between existing ones. The procedure is usually treated as a hypercube between each point of the minority class and its nearest neighboring points. Inside the hypercube, new artificial points are created. This solution has a significant advantage in preserving useful information and even in increasing its size.

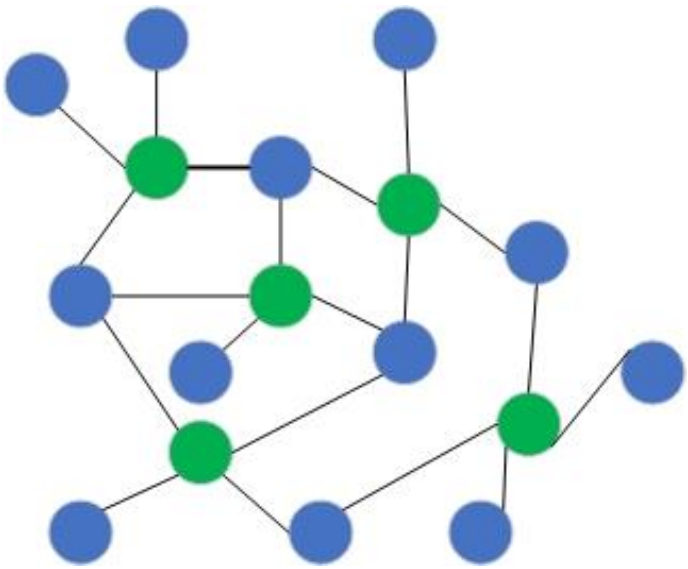

Figure 2.10 – SMOTE

2.4.2 Implementing binary classification

Vectorized data are classified with ML algorithms using the Python – Scikit-learn library. The results were also visualized using Matplotlib and Seaborn libraries. The following metrics were used to assess the effectiveness of data classification: accuracy, precision, recall, and F1-score [1, p. 19-20]. They are expressed by the following formulas (2.7 – 2.10):

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}, \tag{2.7}$$

$$precision = \frac{TP}{TP + FP}, \tag{2.8}$$

$$recall = \frac{TP}{TP + FN}, \tag{2.9}$$

$$F\_score = 2\frac{precision \times recall}{precision + recall}, \tag{2.10}$$

where TP (true positive) indicates a test instance correctly classified by the positive class; TN (true negative) indicates a test instance correctly classified by the negative class; FP (false positive) result indicates that the test instance is erroneously assigned to the positive class; FN (false negative) result indicates that the test instance is erroneously classified as negative.

There is also a useful graphical measure for effectively evaluating the algorithms. It is called an Area under the curve – Receiver operating characteristics (AUC–ROC). The AUC–ROC is very convenient for visualizing classification results. It represents an area under the curve on the axes plane from zero to one. The axes of the planes show TruePositiveRate and FalsePositiveRate, which are calculated by the following formulas (2.11, 2.12):

$$TruePositiveRate = \frac{TP}{TP + FN}, \tag{2.11}$$

$$FalsePositiveRate = \frac{FP}{FP + TN}. \tag{2.12}$$

The greater the value of an area, the better the classification model's performance is. The data classification results are presented in Tables 2.4 and 2.5.

Table 2.4 – Binary classification of texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.78	0.81	0.77	0.74	0.71	0.82	0.79	0.77
Precision	0.74	0.78	0.72	0.80	0.68	0.80	0.75	0.75
Recall	0.82	0.81	0.83	0.58	0.70	0.83	0.83	0.77
F-score	0.78	0.80	0.77	0.67	0.69	0.81	0.78	0.76
Average	0.78	0.80	0.77	0.70	0.70	0.82	0.79	

Table 2.5 – Binary classification of comments

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.68	0.67	0.68	0.61	0.61	0.64	0.61	0.64
Precision	0.68	0.67	0.68	0.65	0.60	0.63	0.59	0.64
Recall	0.69	0.66	0.66	0.43	0.63	0.65	0.70	0.63
F-score	0.68	0.67	0.67	0.52	0.61	0.64	0.64	0.63
Average	0.68	0.67	0.67	0.55	0.61	0.64	0.64	

Classification model evaluation graphs are set using AUC–ROC curves with area values under the curve and are shown in Figures 2.11 and 2.12.

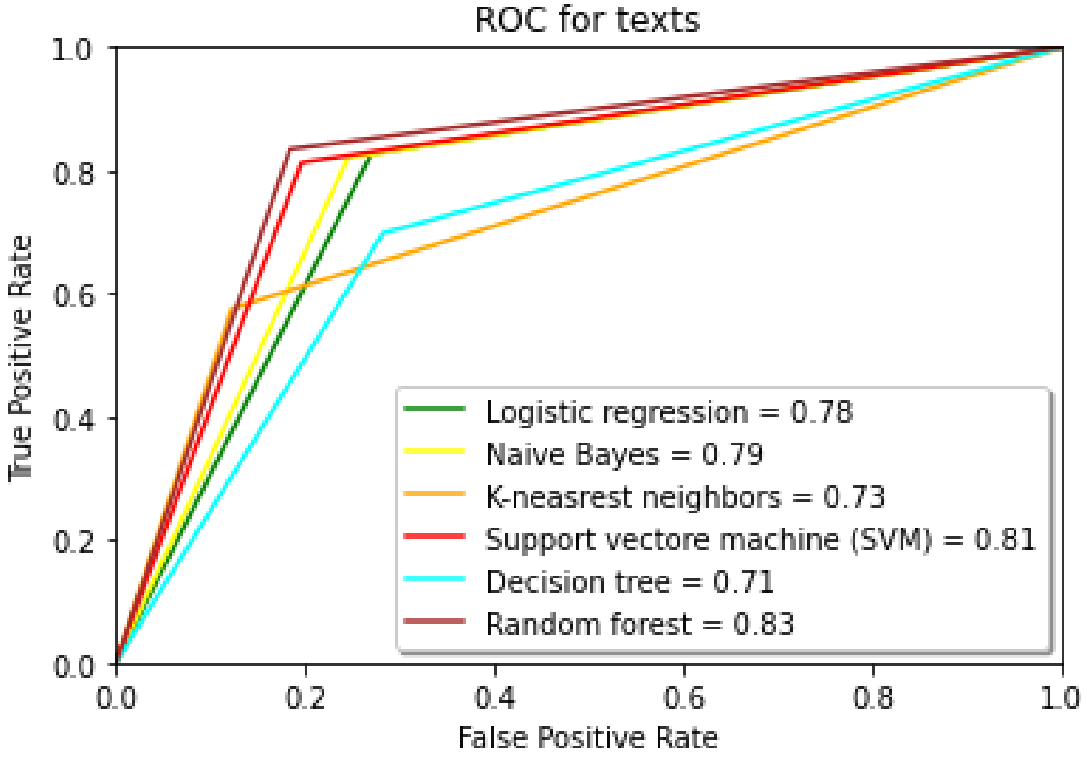

Figure 2.11 – AUC–ROC curve graph for texts

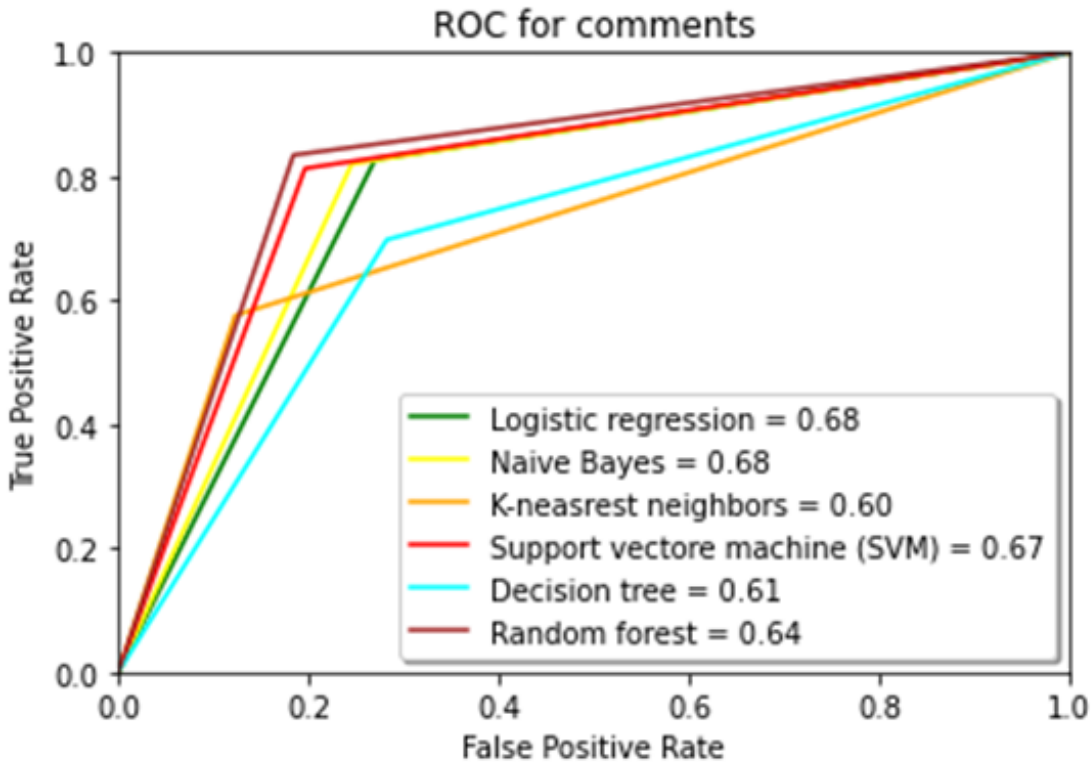

Figure 2.12 – AUC–ROC curve graph for comments

2.4.3 Implementation of multiclass classification with ML algorithms

Metrics for evaluating the effectiveness of multiclass classification [87, 88] differ from metrics for binary classification. Here the following metrics are implemented: accuracy, precision-macro, precision-micro, precision-weighted, recall-macro, recall-micro, recall-weighted, F1-score macro, F1-score micro, and F1-score weighted. Precision-macro is the arithmetic mean of all class accuracy estimates. Precision-micro is the sum of all true positives for all classes divided by all positive predictions. These formulas are shown in (2.13, 2.14):

$$precision\_macro = \frac{precision_1 + precision_2 + precision_3}{3}, \tag{2.13}$$

$$precision\_micro = \frac{TP_1 + TP_2 + TP_3}{TP_1 + TP_2 + TP_3 + FP_1 + FP_2 + FP_3}. \tag{2.14}$$

Recall-macro and recall-micro are defined in a similar manner (2.15, 2.16) [29]:

$$recall\_macro = \frac{recall_1 + recall_2 + recall_3}{3}, \tag{2.15}$$

$$recall\_micro = \frac{TP_1 + TP_2 + TP_3}{TP_1 + TP_2 + TP_3 + FN_1 + FN_2 + FN_3}. \quad (2.16)$$

The weighted average is calculated as macro-weighted. Each class has its own weight by the number of elements included. Precision-weighted and recall-weighted are calculated as follows (2.17, 2.18):

$$precision\_weighted = \frac{w_1 \times precision_1 + w_2 \times precision_2 + w_3 \times precision_3}{3}, \quad (2.17)$$

$$recall\_weighted = \frac{w_1 \times recall_1 + w_2 \times recall_2 + w_3 \times recall_3}{3}, \quad (2.18)$$

where w_1, w_2, w_3 are the weights of the respective classes.

The confusion matrix is a convenient way to show the distribution of predictions across different classes in the graphical representation shown in Figure 2.13.

True \ Predicted	Positive	Negative	Neutral
Positive	T(pos)	F(pos)	F(pos)
Negative	F(neg)	T(neg)	F(neg)
Neutral	F(neut)	F(neut)	T(neut)

Figure 2.13 – A confusion matrix

The datasets parsed by the OMSystem's web crawler were distributed in the following way by the languages and sentiment classes (Table 2.6). The multiclass classification [89-91] took a complete database of texts from news sources and social networks. The texts were labeled into three sentiment classes: positive, neutral, and negative. The initial labeling is made using a sentiment dictionary. The labeling was then checked manually and corrected by experts.

Table 2.6 – Distribution of texts by classes

Language	Negative	Positive	Neutral
Russian	24636	82360	4919
Kazakh	1732	18234	642

The dataset's volume is 132523 for 2021-2022 for creating ML models and NN. The datasets for the Russian and Kazakh languages have been preprocessed, vectorized with the *tf-idf* metric, and resampled with the Random oversampling, Random undersampling, and SMOTE techniques. Then the datasets were randomly split into training, and testing sets as 70% and 30%, respectively, and classified with NB, SVM, LR, k-NN, DT, RF, and XGBoost ML algorithms.

The results of the classification of imbalanced Russian and Kazakh datasets are shown in Tables 2.7 and 2.8.

Table 2.7 – The classification metrics for the imbalanced Russian texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.75	0.74	0.80	0.76	0.73	0.81	0.76	0.76
Precision-macro	0.80	0.25	0.78	0.62	0.58	0.79	0.72	0.65
Precision-micro	0.75	0.74	0.80	0.76	0.73	0.81	0.76	0.76
Precision-weighted	0.76	0.54	0.79	0.74	0.79	0.81	0.73	0.74
Recall-macro	0.39	0.33	0.52	0.44	0.60	0.57	0.42	0.47
Recall-micro	0.75	0.74	0.80	0.76	0.73	0.81	0.76	0.76
Recall-weighted	0.75	0.74	0.80	0.76	0.73	0.81	0.76	0.76
F1-score-macro	0.38	0.28	0.57	0.46	0.54	0.63	0.45	0.47
F1-score-micro	0.75	0.74	0.80	0.76	0.73	0.81	0.76	0.76
F1-score-weighted	0.67	0.63	0.78	0.70	0.75	0.79	0.70	0.72
Average	0.68	0.57	0.74	0.68	0.69	0.76	0.68	

Table 2.8 – The classification metrics for the imbalanced Kazakh texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.89	0.89	0.89	0.89	0.87	0.91	0.89	0.89
Precision-macro	0.43	0.30	0.67	0.59	0.61	0.83	0.75	0.60
Precision-micro	0.89	0.89	0.89	0.89	0.87	0.91	0.89	0.89
Precision-weighted	0.82	0.79	0.87	0.87	0.90	0.91	0.88	0.86
Recall-macro	0.33	0.33	0.37	0.47	0.61	0.50	0.37	0.43
Recall-micro	0.89	0.89	0.89	0.89	0.87	0.91	0.89	0.89
Recall-weighted	0.89	0.89	0.89	0.89	0.87	0.91	0.89	0.89
F1-score-macro	0.32	0.31	0.38	0.50	0.57	0.57	0.38	0.43
F1-score-micro	0.89	0.89	0.89	0.89	0.87	0.91	0.89	0.89
F1-score-weighted	0.83	0.83	0.85	0.87	0.88	0.89	0.85	0.86
Average	0.72	0.70	0.76	0.78	0.79	0.83	0.77	

Analysis of the results showed that classes with an imbalanced distribution showed the lowest values of the precision-macro, recall-macro, and F1-score-macro metrics when using SVM. NB, LR, k-NN, and XGBoost also performed poorly on the

recall-macro and F1-score-macro metrics. At the same time, RF, LR, and DT showed the best average values for imbalanced Russian texts, and RF, DT, and k-NN turned out to be the best for imbalanced Kazakh texts. Generally, RF showed the best results among all ML algorithms for both datasets. Graphs of AUC–ROC curves for the RF algorithm for Russian and Kazakh texts [1, p. 23,24] are shown in Figures 2.14 and 2.15.

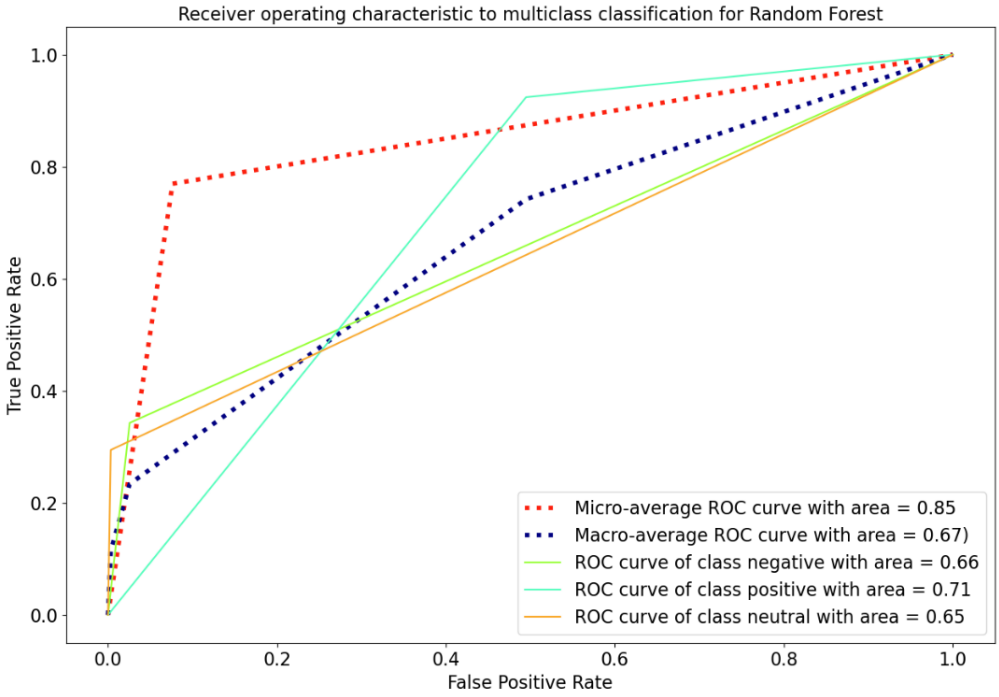

Figure 2.14 – AUC–ROC curves for Russian texts of an RF algorithm

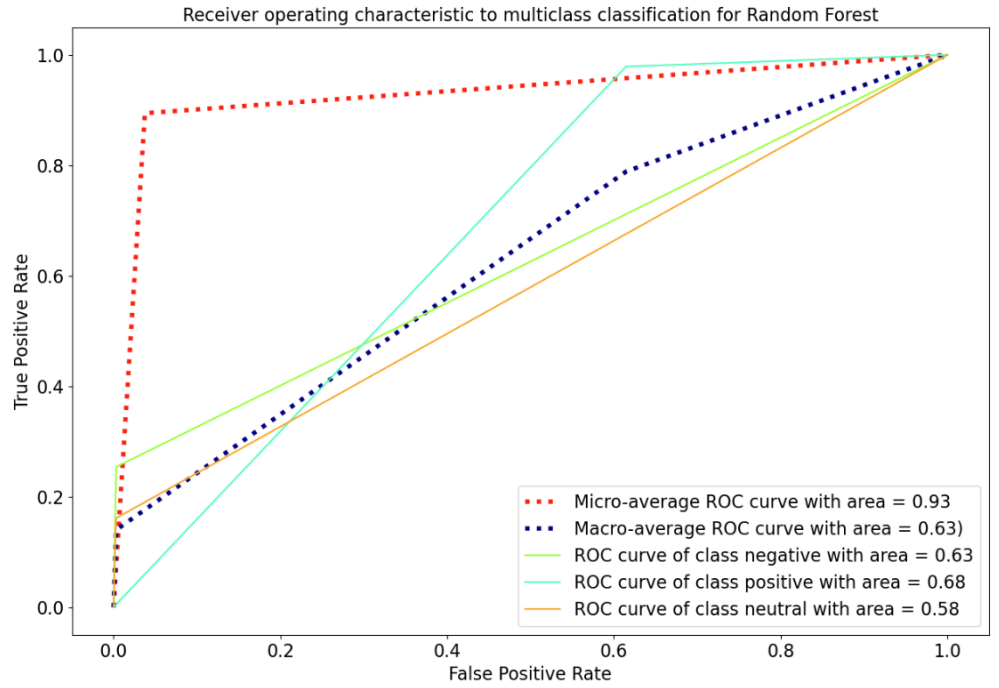

Figure 2.15 – AUC–ROC curves for Kazakh texts of an RF algorithm

AUC–ROC curve plots show that the micro-average metrics are the best, while other metrics show lower results. This problem is caused by class imbalance. Metric values for Russian and Kazakh texts have minor differences.

The results of the classification of the oversampled Russian and Kazakh datasets [1, p. 25,26] are shown in Tables 2.9 and 2.10.

Table 2.9 – The classification metrics for the oversampled Russian texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.71	0.60	0.84	0.67	0.91	0.95	0.64	0.76
Precision-macro	0.73	0.61	0.84	0.77	0.91	0.95	0.64	0.78
Precision-micro	0.71	0.60	0.84	0.67	0.91	0.95	0.64	0.76
Precision-weighted	0.73	0.61	0.84	0.77	0.91	0.95	0.64	0.78
Recall-macro	0.71	0.60	0.84	0.66	0.91	0.95	0.64	0.76
Recall-micro	0.71	0.60	0.84	0.67	0.91	0.95	0.64	0.76
Recall-weighted	0.71	0.60	0.84	0.67	0.91	0.95	0.64	0.76
F1-score-macro	0.71	0.59	0.84	0.65	0.90	0.95	0.63	0.75
F1-score-micro	0.71	0.60	0.84	0.67	0.91	0.95	0.64	0.76
F1-score-weighted	0.71	0.59	0.84	0.65	0.90	0.95	0.63	0.75
Average	0.71	0.60	0.84	0.69	0.91	0.95	0.64	

Table 2.10 – The classification metrics for the oversampled Kazakh texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.84	0.59	0.93	0.93	0.96	0.99	0.73	0.85
Precision-macro	0.85	0.59	0.93	0.94	0.96	0.99	0.73	0.86
Precision-micro	0.84	0.59	0.93	0.93	0.96	0.99	0.73	0.85
Precision-weighted	0.84	0.59	0.93	0.94	0.96	0.99	0.73	0.85
Recall-macro	0.84	0.59	0.93	0.93	0.96	0.99	0.73	0.85
Recall-micro	0.84	0.59	0.93	0.93	0.96	0.99	0.73	0.85
Recall-weighted	0.84	0.59	0.93	0.93	0.96	0.99	0.73	0.85
F1-score-macro	0.84	0.55	0.93	0.93	0.96	0.99	0.73	0.85
F1-score-micro	0.84	0.59	0.93	0.93	0.96	0.99	0.73	0.85
F1-score-weighted	0.84	0.54	0.93	0.93	0.96	0.99	0.73	0.85
Average	0.84	0.58	0.93	0.93	0.96	0.99	0.73	

The results showed that the oversampling technique significantly improved the metrics values for all ML algorithms. Among them, DT and RF were essentially superior to others in the Russian texts. DT, RF, k-NN, and LR were all good for classifying the Kazakh texts. The graphics of confusion matrices for a DT algorithm for the Russian and Kazakh texts are shown in Figures 2.16 and 2.17.

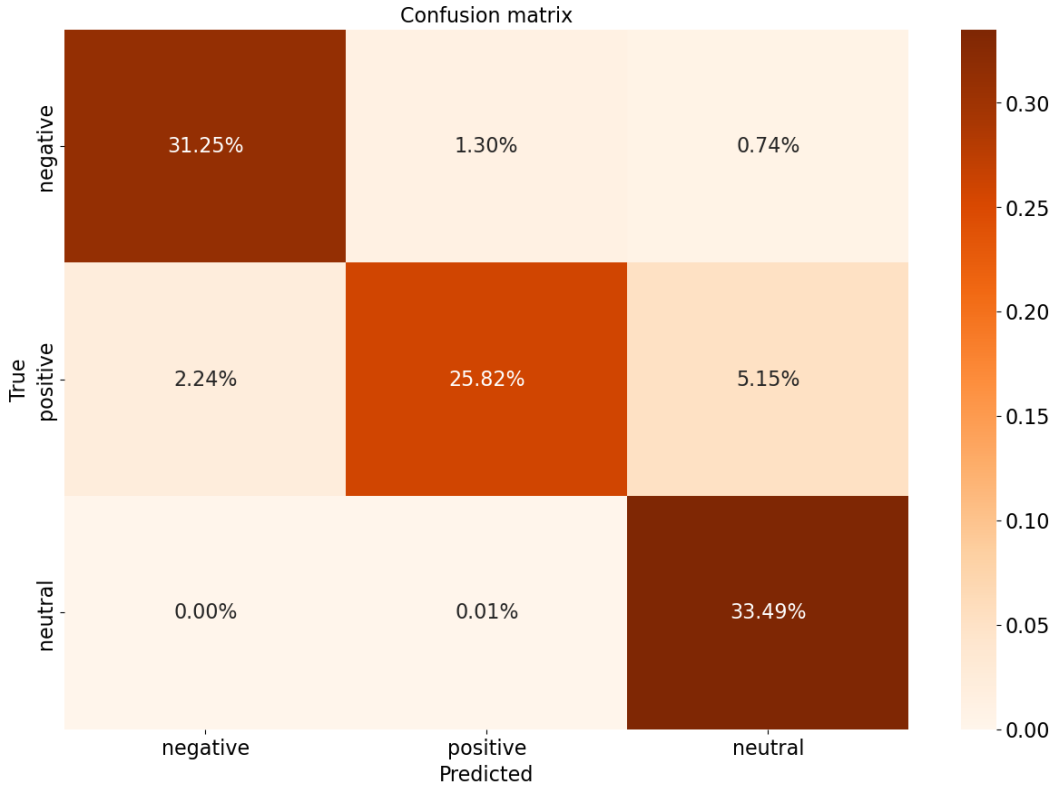

Figure 2.16 – A confusion matrix for Russian texts of a DT algorithm

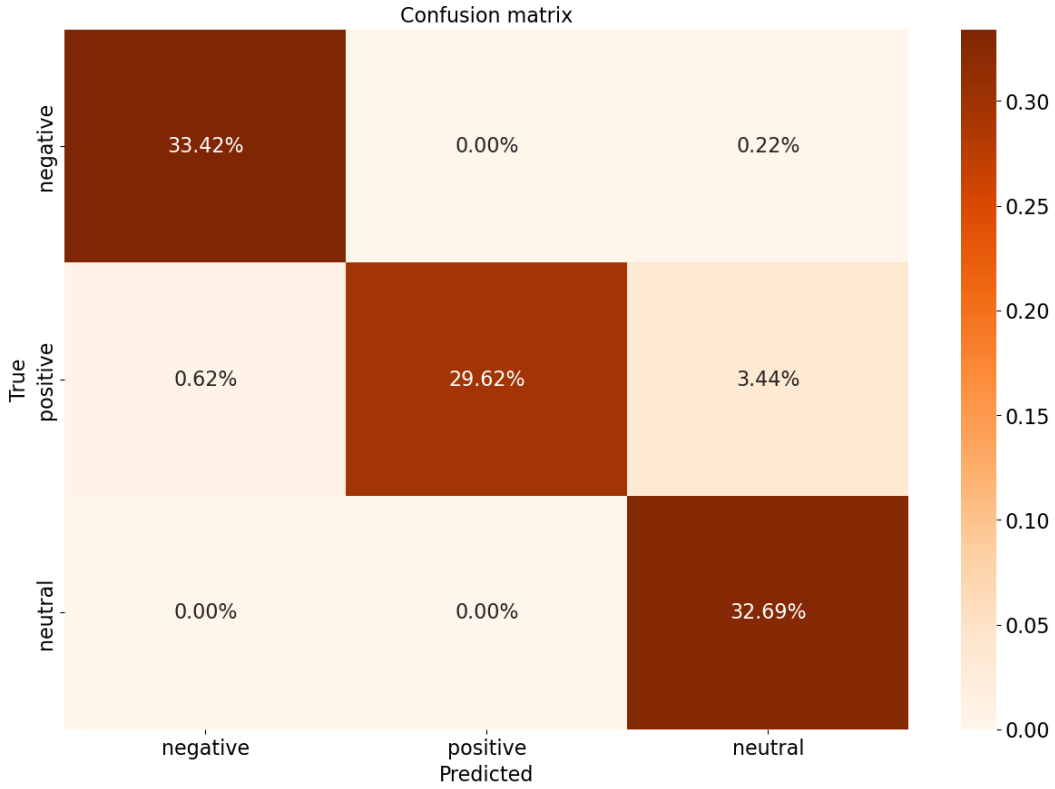

Figure 2.17 – A confusion matrix for Kazakh texts of a DT algorithm

The confusion matrix clearly demonstrates that the DT classification algorithm predicts the classes of the test dataset with a high accuracy value for both Russian and Kazakh texts [1, p. 27,28], which indicates the high quality of the classifier.

The results of the classification of the SMOTE Russian and Kazakh datasets are shown in Tables 2.11 and 2.12.

Table 2.11 – The classification metrics for the SMOTE Russian texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.67	0.64	0.85	0.69	0.83	0.91	0.69	0.75
Precision-macro	0.71	0.63	0.85	0.79	0.84	0.91	0.68	0.77
Precision-micro	0.67	0.64	0.85	0.69	0.83	0.91	0.69	0.75
Precision-weighted	0.71	0.63	0.85	0.79	0.84	0.91	0.68	0.77
Recall-macro	0.67	0.64	0.85	0.69	0.83	0.91	0.69	0.75
Recall-micro	0.67	0.64	0.85	0.69	0.83	0.91	0.69	0.75
Recall-weighted	0.67	0.64	0.85	0.69	0.83	0.91	0.69	0.75
F1-score-macro	0.67	0.63	0.85	0.65	0.82	0.91	0.68	0.74
F1-score-micro	0.67	0.64	0.85	0.69	0.83	0.91	0.69	0.75
F1-score-weighted	0.67	0.63	0.85	0.65	0.82	0.91	0.68	0.74
Average	0.68	0.64	0.85	0.70	0.83	0.91	0.69	

Table 2.12 – The classification metrics for the SMOTE Kazakh texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.85	0.58	0.93	0.78	0.92	0.98	0.72	0.82
Precision-macro	0.85	0.58	0.93	0.84	0.92	0.98	0.72	0.83
Precision-micro	0.85	0.58	0.93	0.78	0.92	0.98	0.72	0.82
Precision-weighted	0.85	0.58	0.93	0.84	0.92	0.98	0.72	0.83
Recall-macro	0.85	0.58	0.93	0.79	0.92	0.98	0.72	0.82
Recall-micro	0.85	0.58	0.93	0.78	0.92	0.98	0.72	0.82
Recall-weighted	0.85	0.58	0.93	0.78	0.92	0.98	0.72	0.82
F1-score-macro	0.85	0.54	0.93	0.75	0.92	0.98	0.71	0.81
F1-score-micro	0.85	0.58	0.93	0.78	0.92	0.98	0.72	0.82
F1-score-weighted	0.85	0.54	0.93	0.75	0.92	0.98	0.71	0.81
Average	0.85	0.57	0.93	0.79	0.92	0.98	0.72	

The results demonstrated that the SMOTE technique also improved the metrics values as the Random oversampling technique. DT and RF outperformed other ML algorithms in classifying the datasets. The graphics of AUC–ROC curves for an RF algorithm for the Russian and Kazakh texts are shown in Figures 2.18 and 2.19.

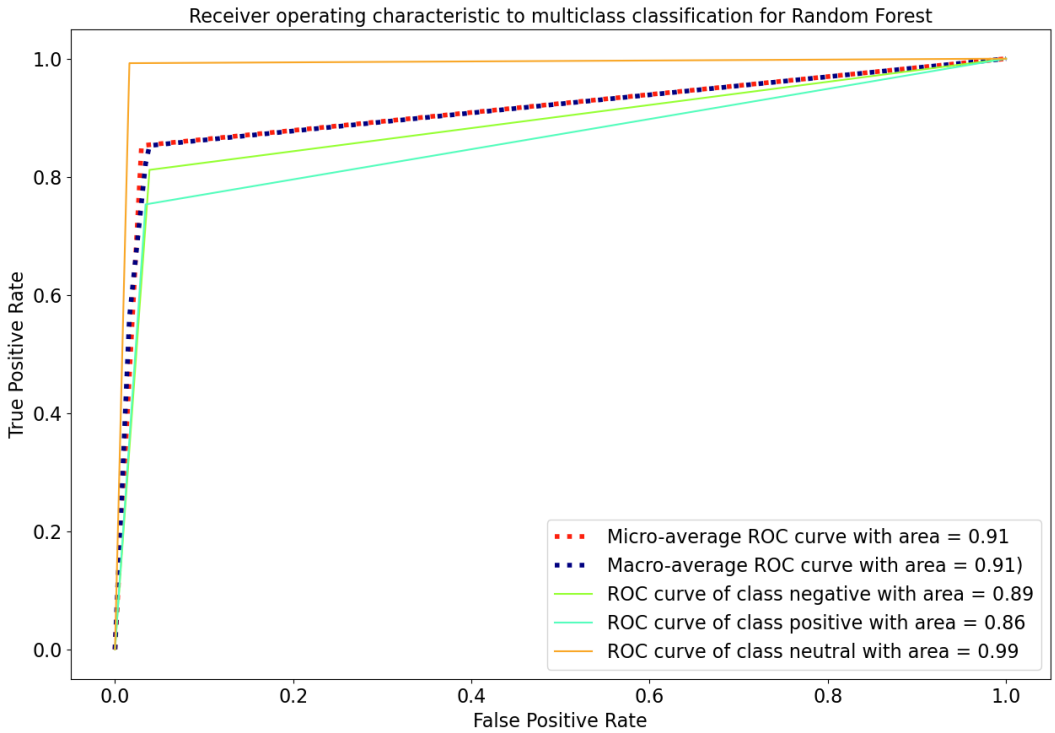

Figure 2.18 – AUC–ROC curves for Russian texts of an RF algorithm

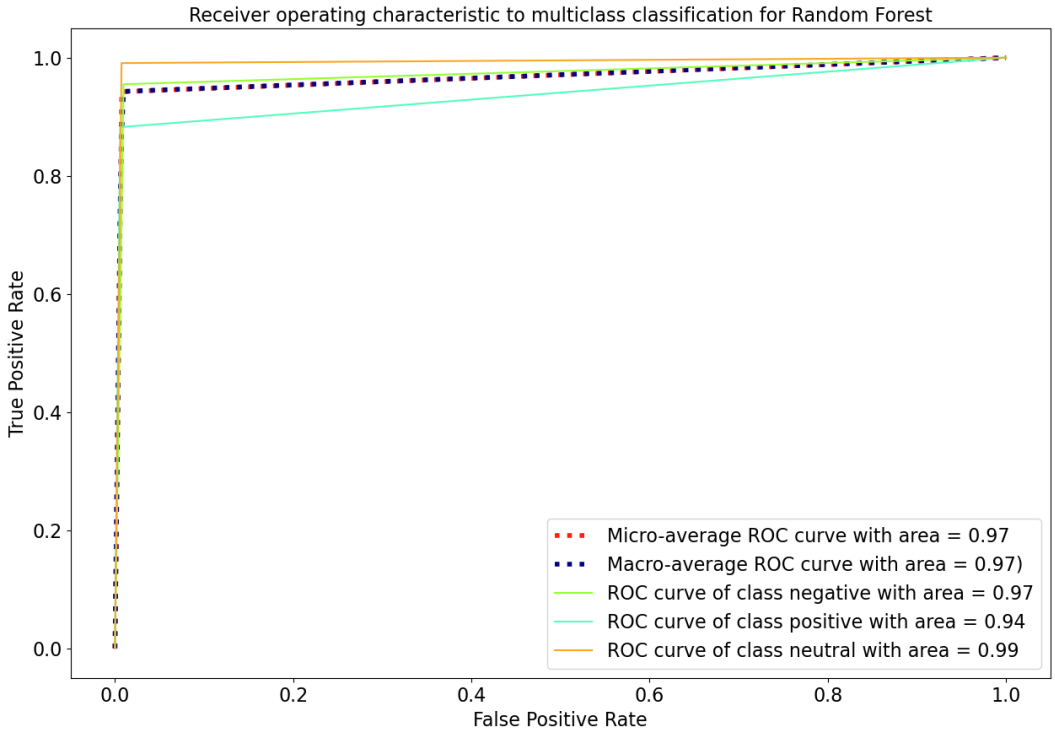

Figure 2.19 – AUC–ROC curves for Kazakh texts of an RF algorithm

Graphs of the AUC–ROC curves demonstrate that the RF classification algorithm shows high values of all classification efficiency metrics for both Russian and Kazakh texts, which indicates the importance of the use of the SMOTE method for class balancing.

The results of the classification of the undersampled Russian and Kazakh datasets [1, p. 29,30] are shown in Tables 2.13 and 2.14.

Table 2.13 – The classification metrics for the undersampled Russian texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.57	0.53	0.72	0.39	0.63	0.71	0.62	0.60
Precision-macro	0.66	0.55	0.72	0.60	0.65	0.72	0.62	0.65
Precision-micro	0.57	0.53	0.72	0.39	0.63	0.71	0.62	0.60
Precision-weighted	0.66	0.55	0.72	0.61	0.65	0.72	0.63	0.65
Recall-macro	0.57	0.54	0.72	0.40	0.63	0.72	0.63	0.60
Recall-micro	0.57	0.53	0.72	0.39	0.63	0.71	0.62	0.60
Recall-weighted	0.57	0.53	0.72	0.39	0.63	0.71	0.62	0.60
F1-score-macro	0.56	0.44	0.71	0.30	0.62	0.71	0.62	0.57
F1-score-micro	0.57	0.53	0.72	0.39	0.63	0.71	0.62	0.60
F1-score-weighted	0.56	0.44	0.71	0.30	0.62	0.71	0.61	0.56
Average	0.59	0.52	0.72	0.42	0.63	0.71	0.62	

Table 2.14 – The classification metrics for the undersampled Kazakh texts

Classifier	NB	SVM	LR	k-NN	DT	RF	XGBoost	Average
Accuracy	0.53	0.58	0.72	0.63	0.70	0.74	0.67	0.65
Precision-macro	0.66	0.61	0.72	0.63	0.72	0.76	0.67	0.68
Precision-micro	0.53	0.58	0.72	0.63	0.70	0.74	0.67	0.65
Precision-weighted	0.67	0.61	0.72	0.63	0.71	0.76	0.67	0.68
Recall-macro	0.54	0.58	0.72	0.63	0.70	0.74	0.67	0.65
Recall-micro	0.53	0.58	0.72	0.63	0.70	0.74	0.67	0.65
Recall-weighted	0.53	0.58	0.72	0.63	0.70	0.74	0.67	0.65
F1-score-macro	0.51	0.54	0.71	0.62	0.69	0.73	0.67	0.64
F1-score-micro	0.53	0.58	0.72	0.63	0.70	0.74	0.67	0.65
F1-score-weighted	0.51	0.54	0.71	0.62	0.69	0.73	0.67	0.64
Average	0.55	0.58	0.72	0.63	0.70	0.74	0.67	

In the results, it could be seen that the values of the undersampled datasets dropped compared with the oversampled and SMOTE datasets. It is caused by the significant decrease in the sizes of the positive and negative classes to make them equal to the negative class. The DT and RF classifiers showed the best results as in previous experiments. The graphics of confusion matrices for a DT algorithm for the Russian and Kazakh texts are shown in Figures 2.20 and 2.21.

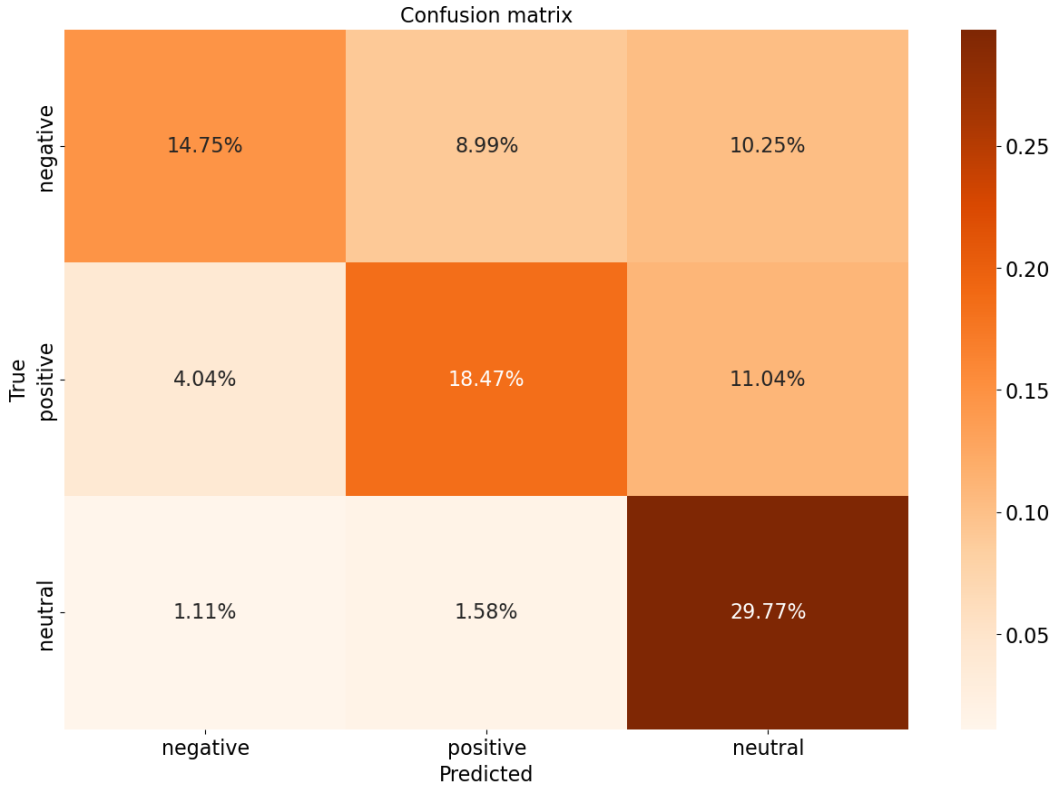

Figure 2.20 – A confusion matrix for Russian texts of a DT algorithm

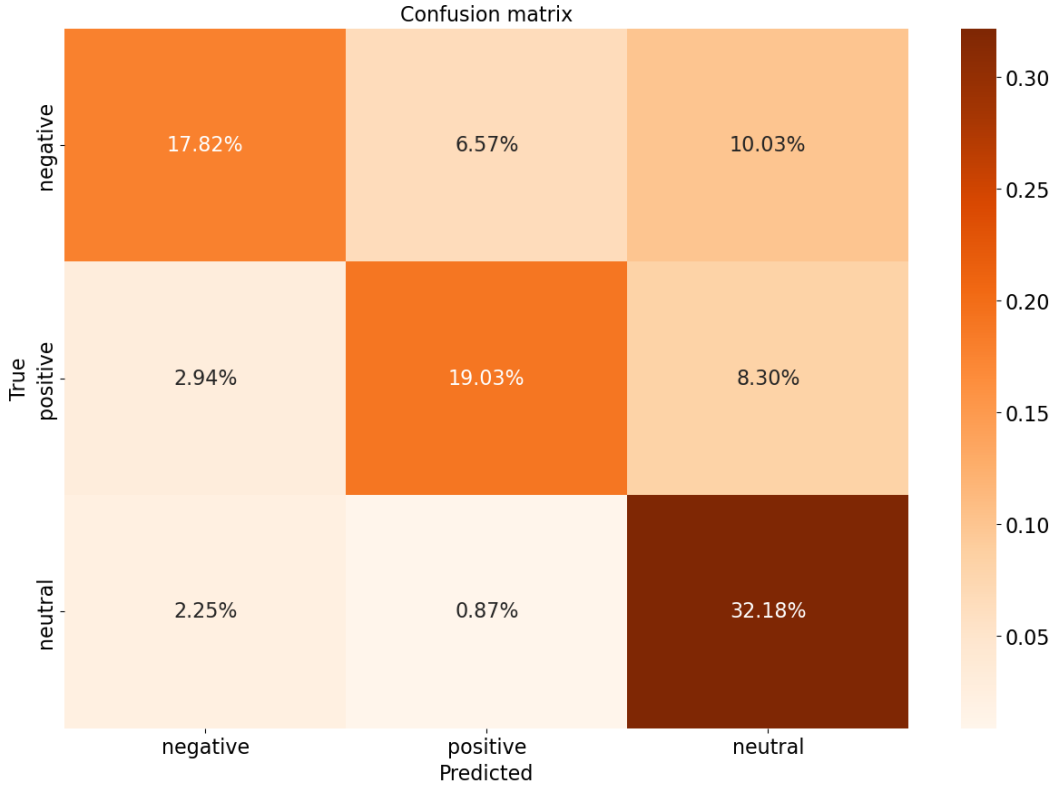

Figure 2.21 – A confusion matrix for Kazakh texts of a DT algorithm

The confusion matrix shows that the DT classification algorithm predicts the classes of the test dataset with a high accuracy value for both Russian and Kazakh texts but is inferior to similar values of the Random oversampling and SMOTE methods,

predicting a large number of neutral classes. It is because the Random undersampling method removes much of the useful information needed to train efficient classifiers.

All the built classification models showed that models trained on the imbalanced datasets achieved the lowest performance. The Random undersampling method gave average values of metrics. It is because the resulting models cannot fully use the entire dataset being significantly decreased in size. The Random oversampling and SMOTE models expectedly demonstrated the best results. LR, DT, and RF showed the best results among ML algorithms. Although the Naive Bayes classifier shows good results, it is worth noting that the algorithm suffers from known limitations associated with the assumption that all its functions are mutually independent. Despite its simplicity, the k-NN method achieves satisfactory results on small datasets. However, it tends to be slower and less accurate with larger datasets. Since RF uses several independent DTs, it is obvious that its performance is higher than that of a single DT. In the previous study, singular number decomposition was applied to texts classified using SVM and XGBoost. It was done to speed up the training of algorithms, so this is one of the reasons why these classifiers are ineffective compared to others. The results of a classification in Russian and Kazakh are relatively equal, with slightly better performance for the latter in oversampled and SMOTE datasets, which have a smaller test size. Thus, it can be seen that large balanced datasets derived from oversampling and SMOTE approaches are the best and preferred for use on social analytics platforms.

2.4.4 Implementation of multiclass classification with neural networks

The next experiment carried out classification using convolutional and recurrent NN. At first, the TF-IDF metric was used to vectorize texts. Datasets are also divided into 70% training and 30% testing sets. The results of the classification of imbalanced data are presented in Table 2.15. Multiclass classification metrics, a confusion matrix, and accuracy and loss graphs of Russian texts with DNN are shown in Figures 2.22, 2.23, and 2.24.

Table 2.15 – Classification of imbalanced datasets

	Russian texts			Kazakh texts		
Classifier	DNN	CNN	LSTM	DNN	CNN	LSTM
Accuracy	0.81	0.73	0.73	0.91	0.89	0.89
Precision-macro	0.69	0.24	0.24	0.67	0.30	0.30
Precision-micro	0.81	0.73	0.73	0.91	0.89	0.89
Precision-weighted	0.80	0.54	0.54	0.90	0.79	0.79
Recall-macro	0.64	0.33	0.33	0.63	0.33	0.33
Recall-micro	0.81	0.73	0.73	0.91	0.89	0.89
Recall-weighted	0.81	0.73	0.73	0.91	0.89	0.89
F1-score macro	0.66	0.28	0.28	0.65	0.31	0.31
F1-score micro	0.81	0.73	0.73	0.91	0.89	0.89
F1-score weighted	0.80	0.62	0.62	0.90	0.84	0.84

The classification results show that DNN has the best metrics compared to CNN and LSTM for both Russian and Kazakh texts. At the same time, DNN values are more aligned for all metrics, while CNN and LSTM have low values for the precision-macro and F1-score macro metrics.

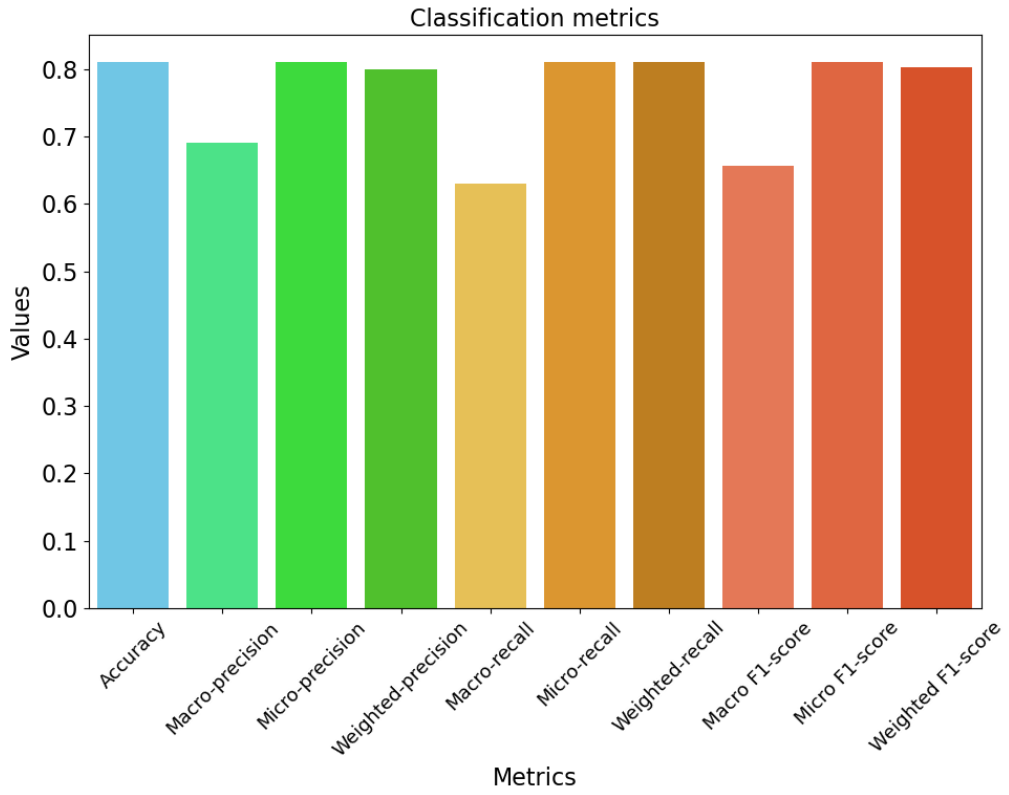

Figure 2.22 – DNN classification metrics for imbalanced Russian texts

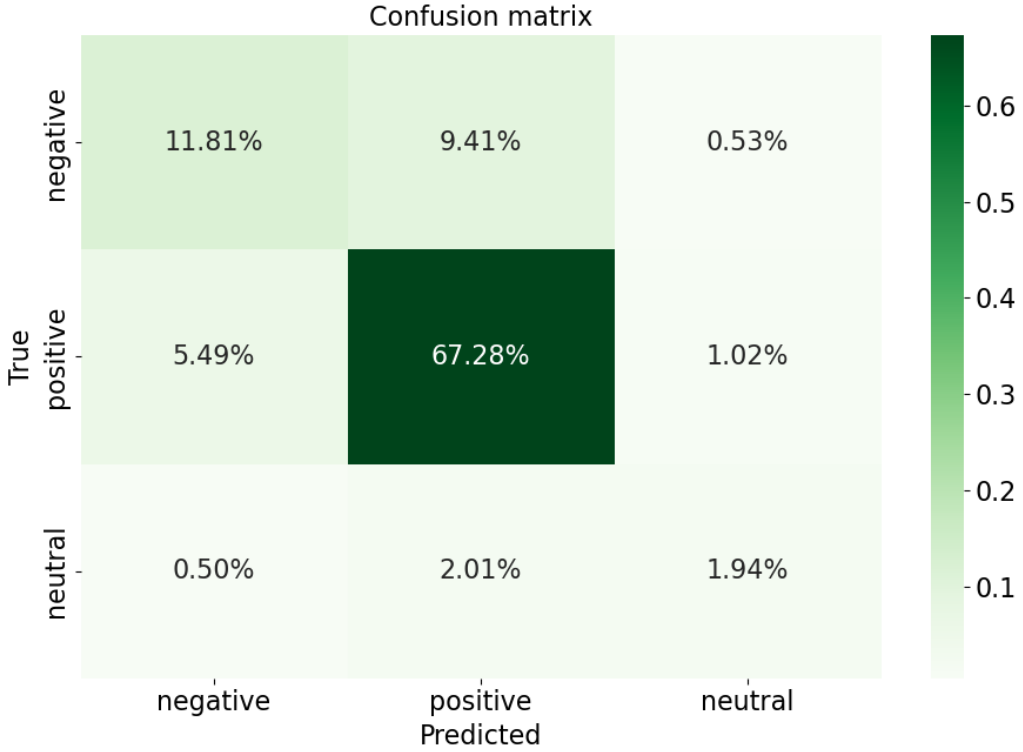

Figure 2.23 – DNN confusion matrix of imbalanced Russian texts

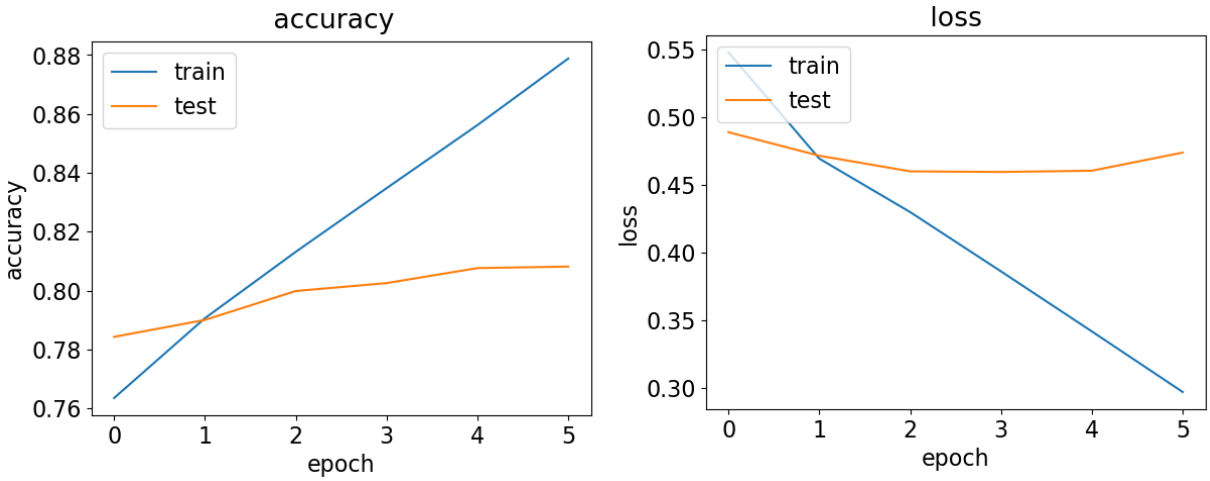

Figure 2.24 – Graphs of accuracy and loss of DNN imbalanced Russian texts

The graphs show that DNN predicts imbalanced texts well, reaching values of 0.80–0.81 for some metrics.

The results of the classification of the oversampled dataset are presented in Table 2.16. Multiclass classification metrics, a confusion matrix, and accuracy and loss graphs of Russian texts with CNN are presented in Figures 2.25, 2.26, and 2.27.

Table 2.16 – Classification of oversampled datasets

Russian texts				Kazakh texts		
Classifier	DNN	CNN	LSTM	DNN	CNN	LSTM
Accuracy	0.92	0.52	0.51	0.97	0.51	0.52
Precision-macro	0.92	0.49	0.45	0.98	0.49	0.54
Precision-micro	0.92	0.52	0.51	0.97	0.51	0.52
Precision-weighted	0.92	0.49	0.45	0.98	0.49	0.54
Recall-macro	0.92	0.51	0.50	0.97	0.51	0.52
Recall-micro	0.92	0.52	0.51	0.97	0.51	0.52
Recall-weighted	0.92	0.52	0.51	0.97	0.51	0.52
F1-score macro	0.92	0.49	0.42	0.97	0.44	0.43
F1-score micro	0.92	0.52	0.51	0.97	0.51	0.52
F1-score weighted	0.92	0.49	0.42	0.97	0.44	0.43

In the classification of oversampled datasets, DNN significantly outperforms both CNN and LSTM for both Russian and Kazakh texts. DNN metric values are greater than 0.90.

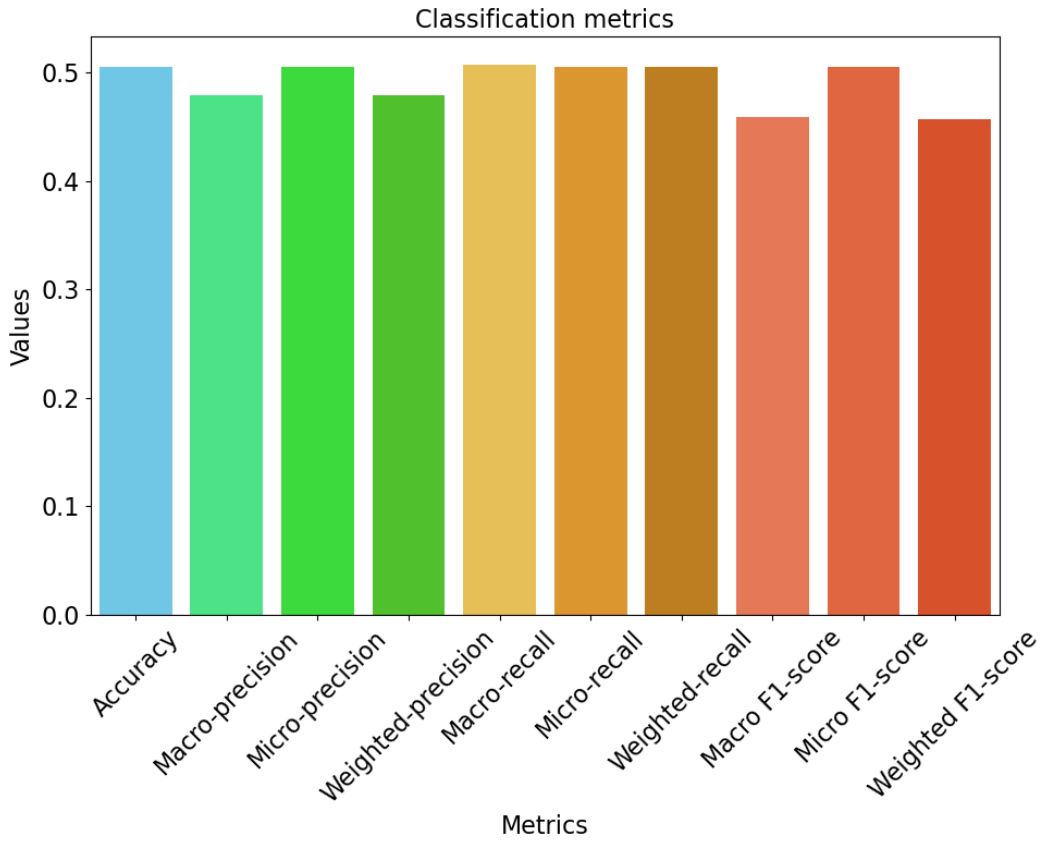

Figure 2.25 – CNN classification metrics of oversampled Kazakh texts

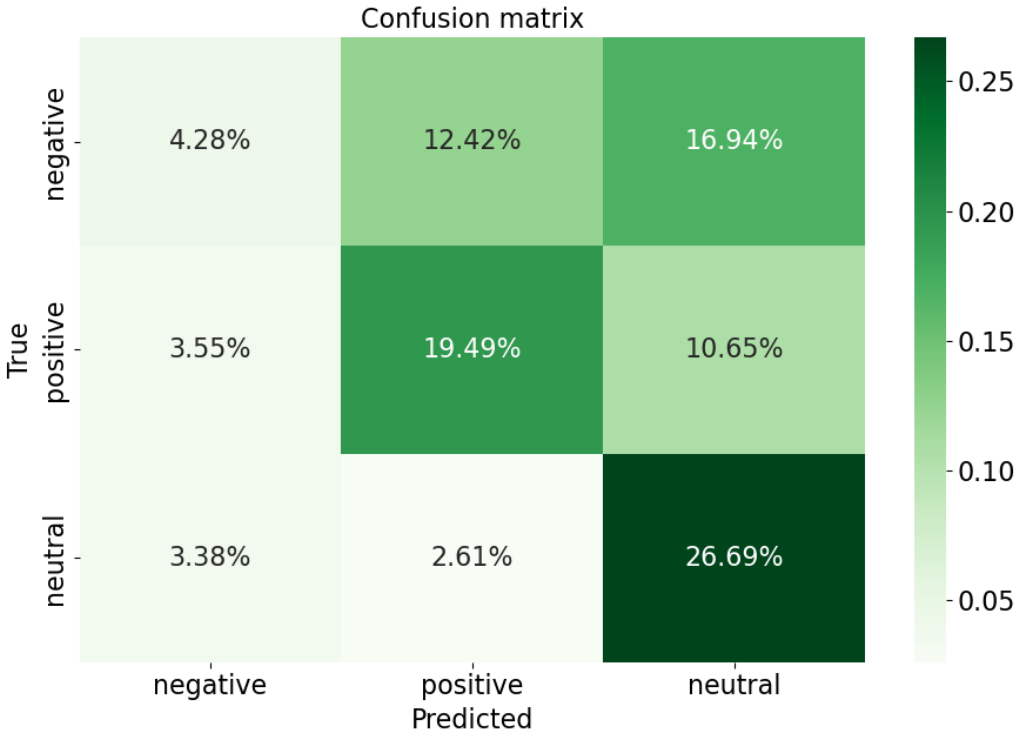

Figure 2.26 – CNN confusion matrix of oversampled Kazakh texts

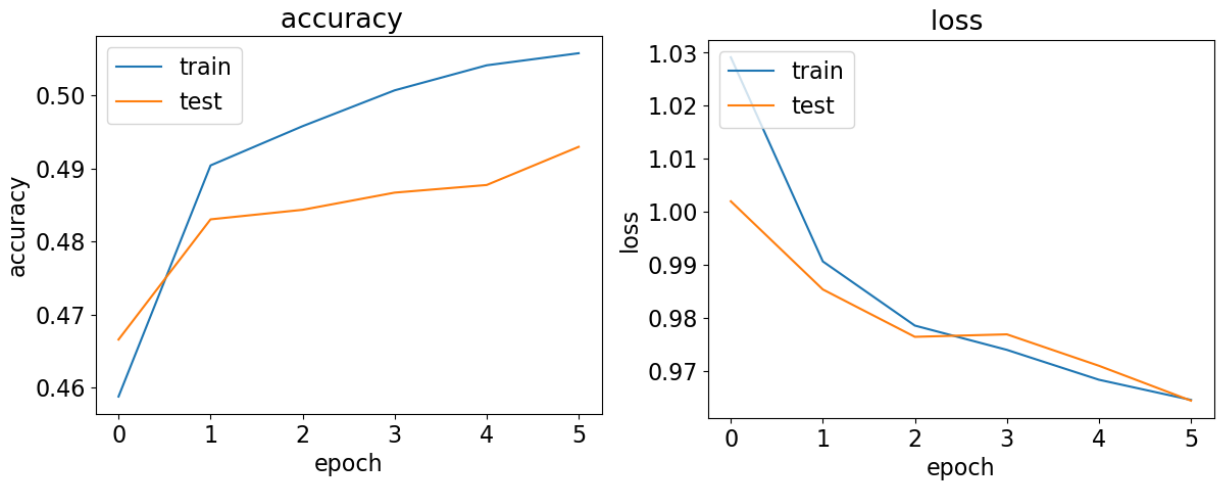

Figure 2.27 – Graphs of accuracy and loss of CNN oversampled Kazakh texts

The graphs demonstrate that CNN does not show good classification results even on oversampled datasets, reaching values no more than 0.50–0.52.

Results of SMOTE data classification are presented in Table 2.17. Multiclass classification metrics, a confusion matrix, and accuracy and loss graphs of Russian texts with CNN are shown in Figures 2.28, 2.29, and 2.30.

Table 2.17 – Classification of SMOTE datasets

	Russian texts			Kazakh texts		
Classifier	DNN	CNN	LSTM	DNN	CNN	LSTM
Accuracy	0.91	0.52	0.36	0.98	0.51	0.35
Precision-macro	0.91	0.53	0.33	0.98	0.53	0.32
Precision-micro	0.91	0.52	0.36	0.98	0.51	0.35
Precision-weighted	0.91	0.53	0.33	0.98	0.53	0.32
Recall-macro	0.91	0.52	0.35	0.98	0.51	0.35
Recall-micro	0.91	0.52	0.36	0.98	0.51	0.35
Recall-weighted	0.91	0.52	0.36	0.98	0.51	0.35
F1-score macro	0.91	0.48	0.22	0.98	0.51	0.21
F1-score micro	0.91	0.52	0.36	0.98	0.51	0.35
F1-score weighted	0.91	0.48	0.22	0.98	0.51	0.21

As in the oversampled dataset classification, DNN performs better in the SMOTE classification, reaching metric values of 0.91–0.98.

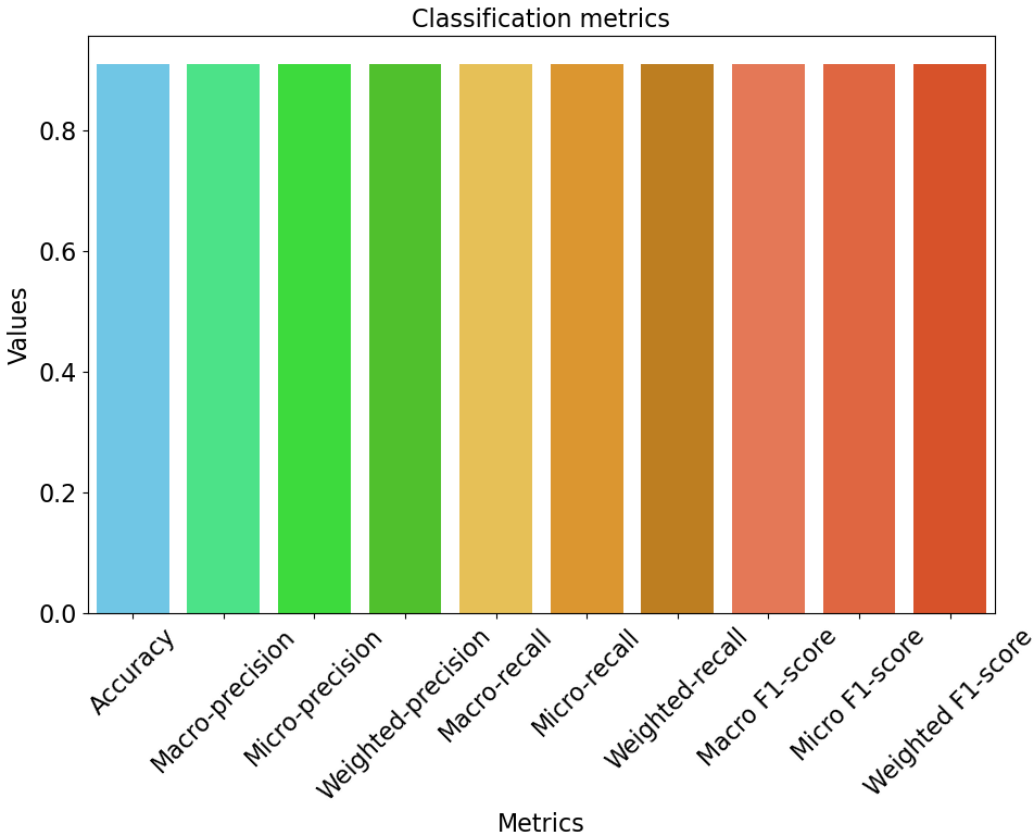

Figure 2.28 – DNN classification metrics of SMOTE Russian texts

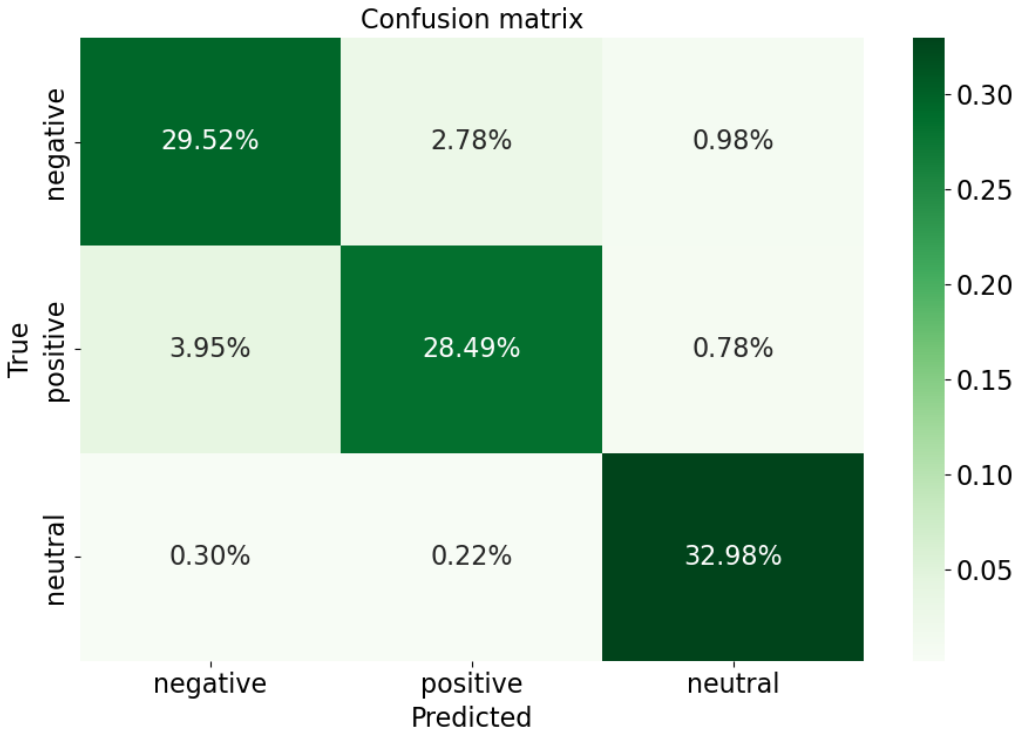

Figure 2.29 – DNN confusion matrix of SMOTE Russian texts

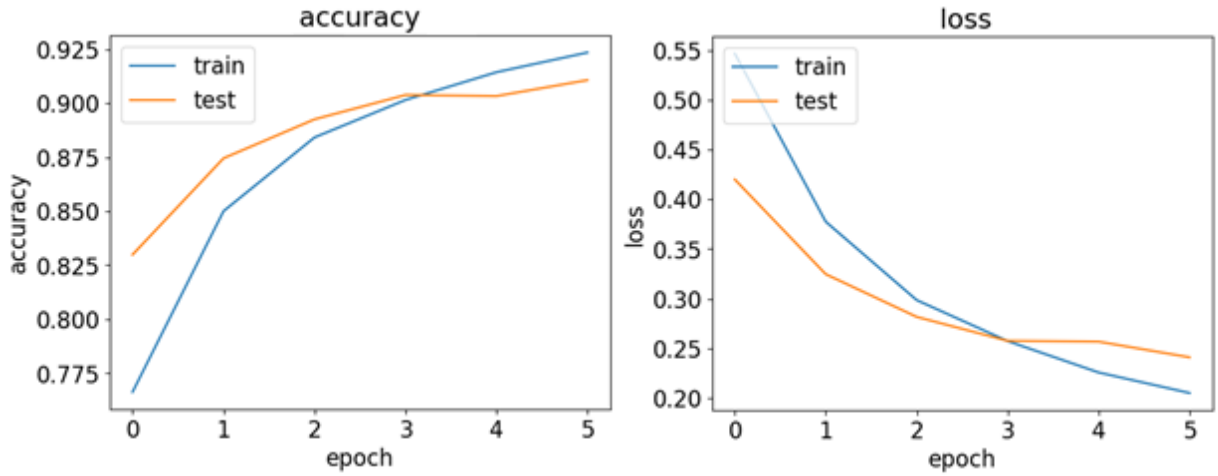

Figure 2.30 – Graphs of accuracy and loss of DNN SMOTE Russian texts

The graphs show that DNN is effective in classifying SMOTE texts, achieving very high accuracy values exceeding 0.90.

The results of the classification of the undersampled datasets are shown in Table 2.18. Multiclass classification metrics, a confusion matrix, and accuracy and loss graphs of Russian texts with DNN are presented in Figures 2.31, 2.32, and 2.33.

Table 2.18 – Classification of undersampled datasets

Russian texts				Kazakh texts		
Classifier	DNN	CNN	LSTM	DNN	CNN	LSTM
Accuracy	0.72	0.50	0.33	0.76	0.48	0.34
Precision-macro	0.72	0.48	0.11	0.76	0.45	0.11
Precision-micro	0.72	0.50	0.33	0.76	0.48	0.34
Precision-weighted	0.72	0.48	0.11	0.76	0.44	0.11
Recall-macro	0.72	0.50	0.33	0.76	0.48	0.33
Recall-micro	0.72	0.50	0.33	0.76	0.48	0.34
Recall-weighted	0.72	0.50	0.33	0.76	0.48	0.33
F1-score macro	0.72	0.47	0.17	0.76	0.43	0.17
F1-score micro	0.72	0.50	0.33	0.76	0.48	0.34
F1-score weighted	0.72	0.47	0.16	0.76	0.43	0.17

The classification results show that the metrics for DNN, CNN, and LSTM using the undersampling method are lower than those for the oversampling and SMOTE methods. However, DNN still outperforms CNN and LSTM.

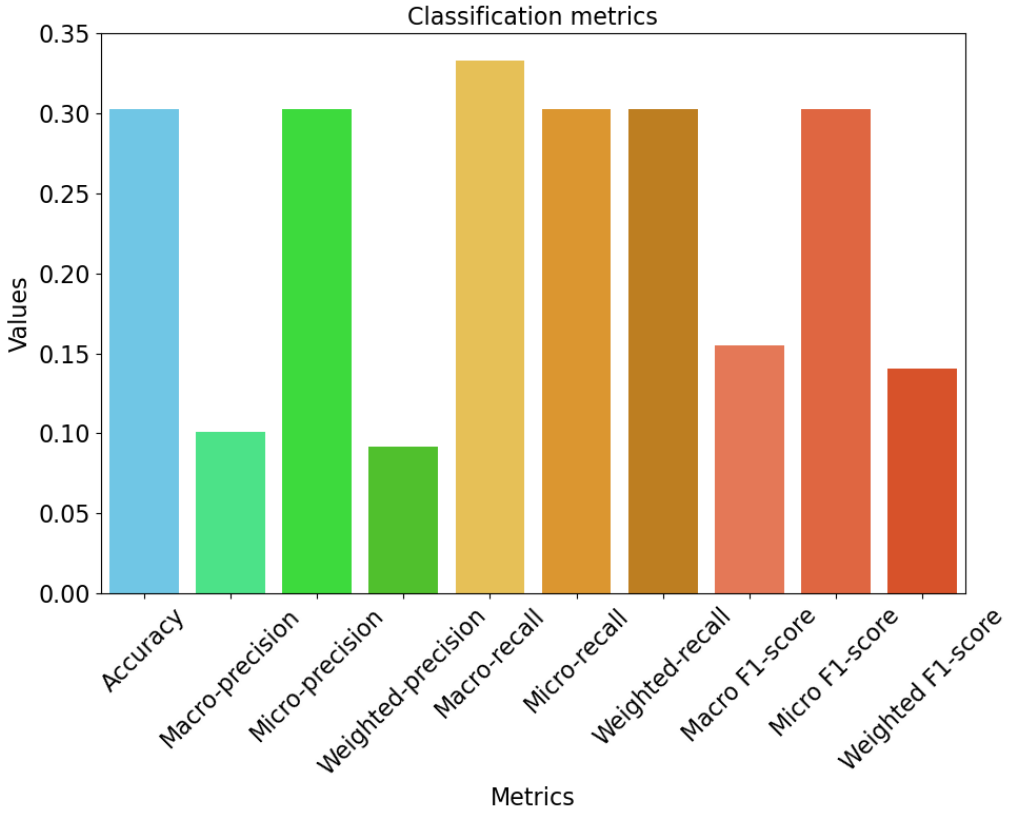

Figure 2.31 – LSTM classification metrics of undersampled Kazakh texts

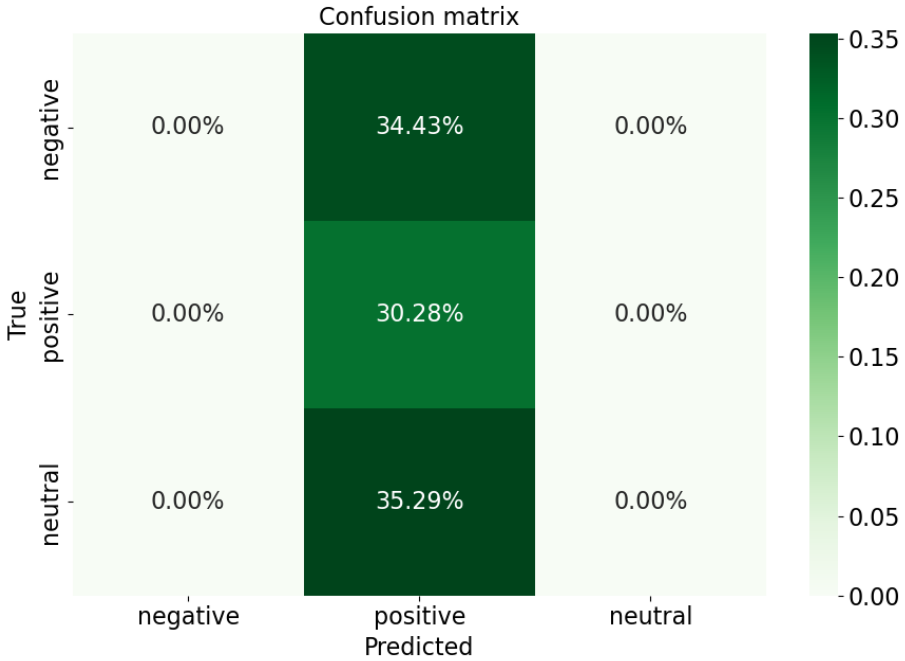

Figure 2.32 – LSTM confusion matrix of undersampled Kazakh texts

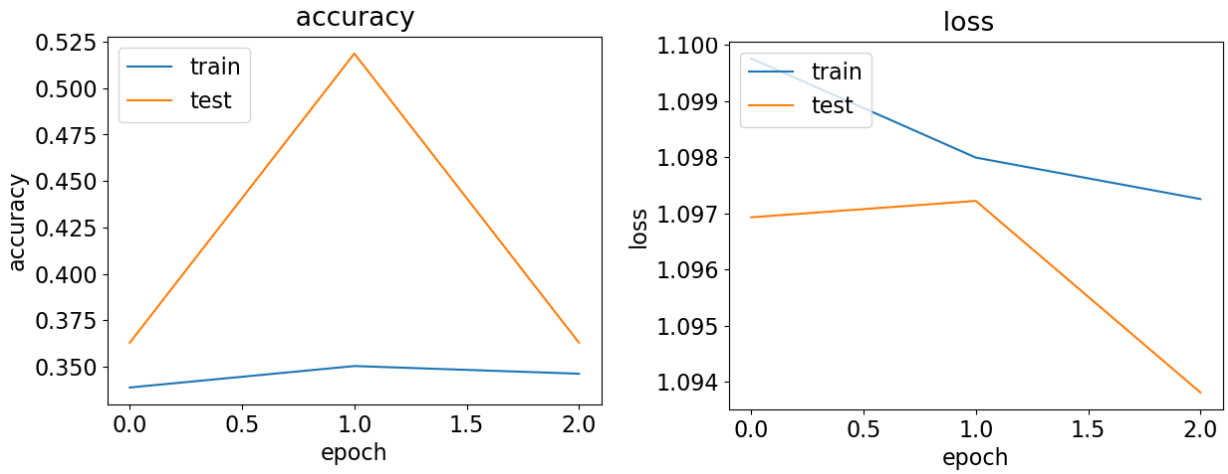

Figure 2.33 – Graphs of accuracy and loss of LSTM undersampled Russian texts

Classification graphics show that LSTM performs very poorly, with metric values in the range of 0.11–0.34. In this case, the neural network predicts only the positive class.

The classification of *tf-idf* metric vectorized texts with NN showed that DNN demonstrated significantly better results than convolutional and recurrent NN. These results take place in all four class balancing experiments. Methods for classifying imbalanced and undersampled datasets in the same way as in experiments with ML algorithms showed worse results than oversampling and SMOTE models.

Then, the Fasttext word embedding method was used for vectorization. The results of the classification of imbalanced data are presented in Table 2.19. Multiclass classification metrics, a confusion matrix, and accuracy and loss graphs of Russian texts with DNN are shown in Figures 2.36, 2.37, and 2.38.

Table 2.19 – Classification of imbalanced datasets

	Russian texts			Kazakh texts		
Classifier	DNN	CNN	LSTM	DNN	CNN	LSTM
Accuracy	0.73	0.81	0.74	0.89	0.92	0.89
Precision-macro	0.24	0.71	0.74	0.30	0.70	0.30
Precision-micro	0.73	0.81	0.74	0.89	0.92	0.89
Precision-weighted	0.54	0.81	0.71	0.79	0.91	0.79
Recall-macro	0.33	0.67	0.37	0.33	0.61	0.33
Recall-micro	0.73	0.81	0.74	0.89	0.92	0.89
Recall-weighted	0.73	0.81	0.74	0.89	0.92	0.89
F1-score macro	0.28	0.69	0.36	0.31	0.65	0.31
F1-score micro	0.73	0.81	0.74	0.89	0.92	0.89
F1-score weighted	0.62	0.81	0.65	0.84	0.91	0.84

In classification with the word embedding method, CNN shows better metrics than DNN and LSTM, reaching values of 0.65–0.81.

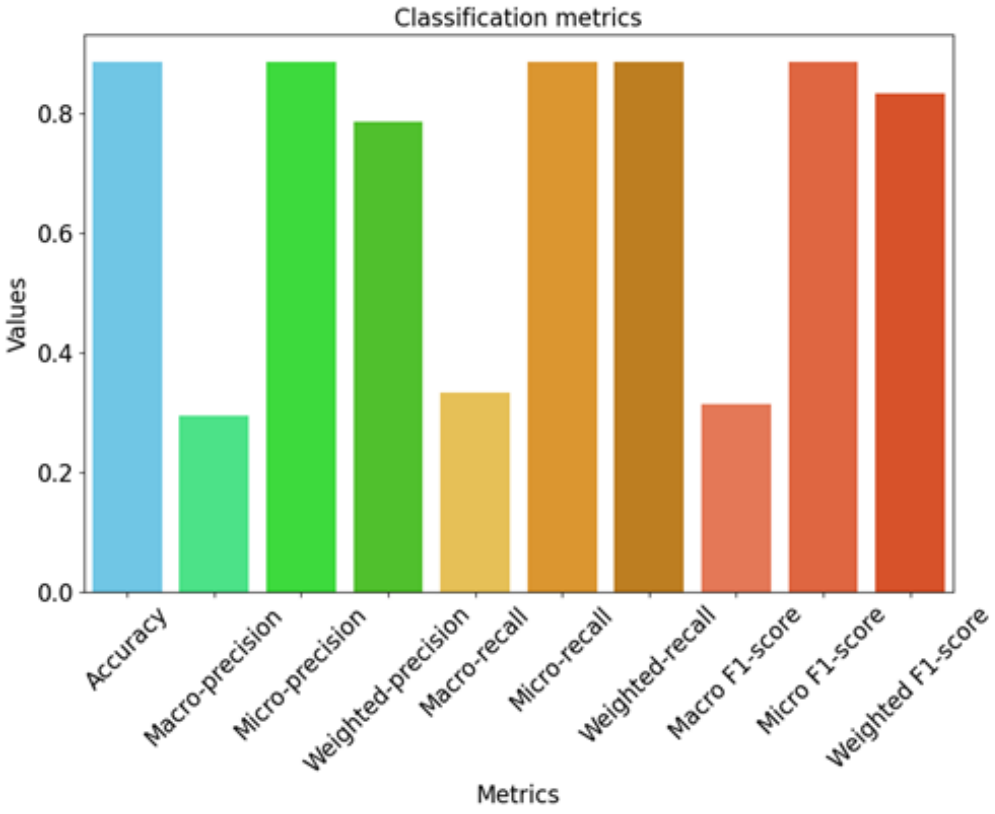

Figure 2.34 – DNN classification metrics of imbalanced Russian texts

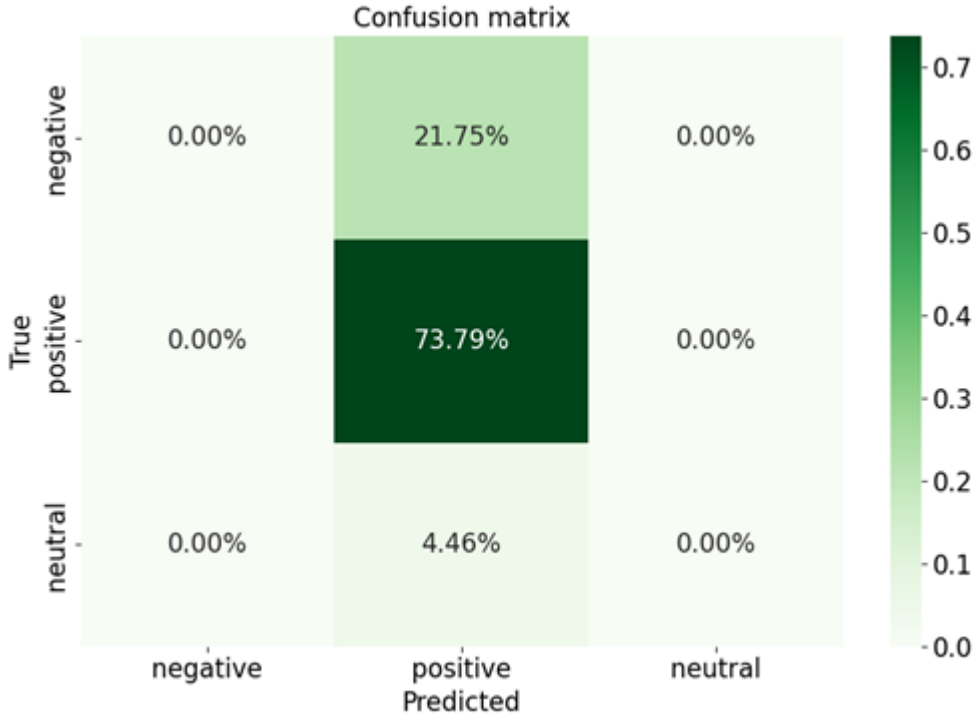

Figure 2.35 – DNN confusion matrix of imbalanced Russian texts

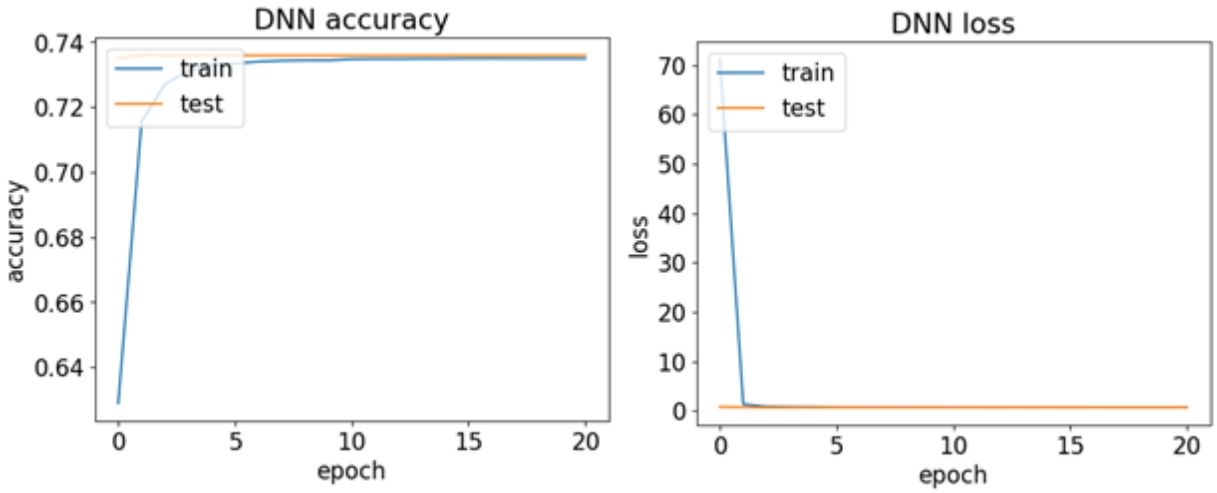

Figure 2.36 – Graphs of accuracy and loss of DNN imbalanced Russian texts

Classification graphs show that the precision-macro, recall-macro, and F1-score macro metrics have low values, inferior to other indicators, as well as CNN and LSTM metrics.

The results of the classification of the oversampled datasets are presented in Table 2.20. Multiclass classification metrics, a confusion matrix, and accuracy and loss graphs of Russian texts with CNN are shown in Figures 2.37, 2.38, and 2.39.

Table 2.20 – Classification of oversampled datasets

Russian texts				Kazakh texts		
Classifier	DNN	CNN	LSTM	DNN	CNN	LSTM
Accuracy	0.62	0.93	0.85	0.75	0.98	0.83
Precision-macro	0.61	0.93	0.85	0.82	0.98	0.83
Precision-micro	0.62	0.93	0.85	0.75	0.98	0.83
Precision-weighted	0.61	0.93	0.85	0.82	0.98	0.83
Recall-macro	0.62	0.93	0.85	0.75	0.98	0.83
Recall-micro	0.62	0.93	0.85	0.75	0.98	0.83
Recall-weighted	0.62	0.93	0.85	0.75	0.98	0.83
F1-score macro	0.59	0.93	0.85	0.76	0.98	0.82
F1-score micro	0.62	0.93	0.85	0.75	0.98	0.83
F1-score weighted	0.59	0.93	0.85	0.76	0.98	0.83

The classification of oversampled datasets shows that CNN and LSTM significantly outperform DNN for Russian and Kazakh texts.

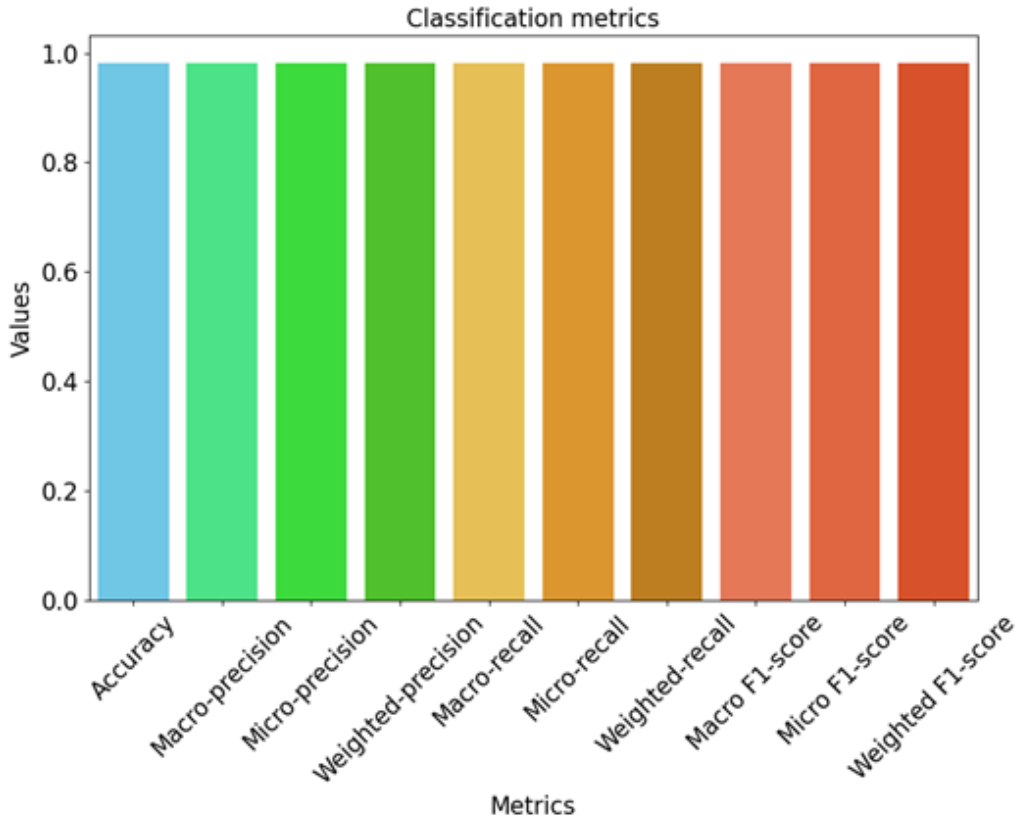

Figure 2.37 – Classification metrics of CNN oversampled Kazakh texts

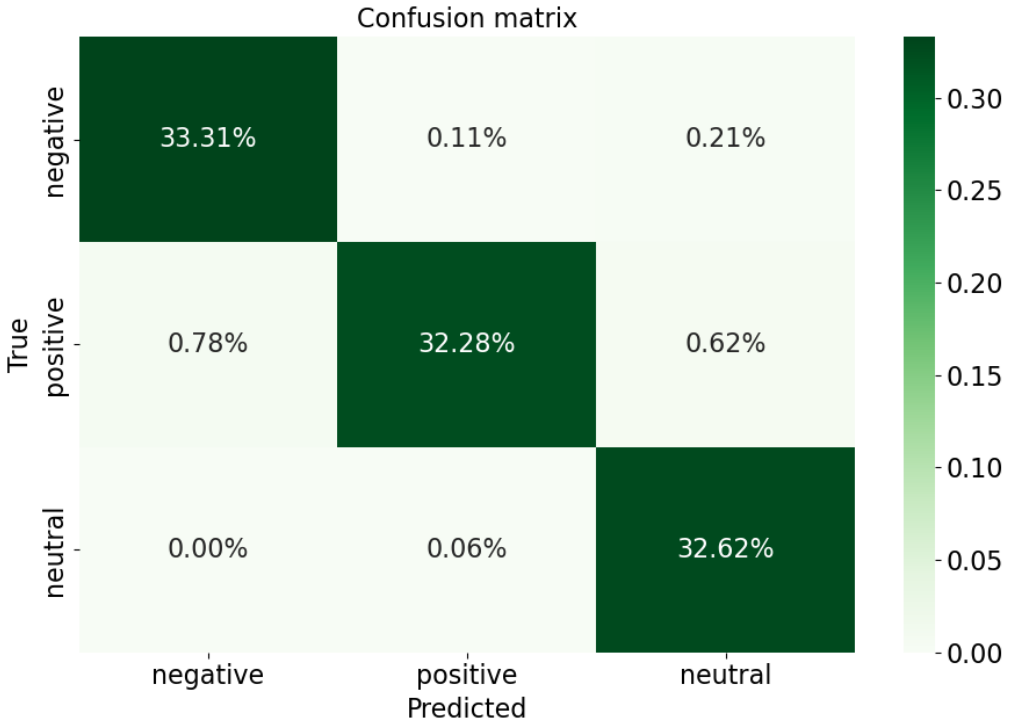

Figure 2.38 – CNN confusion matrix of oversampled Kazakh texts

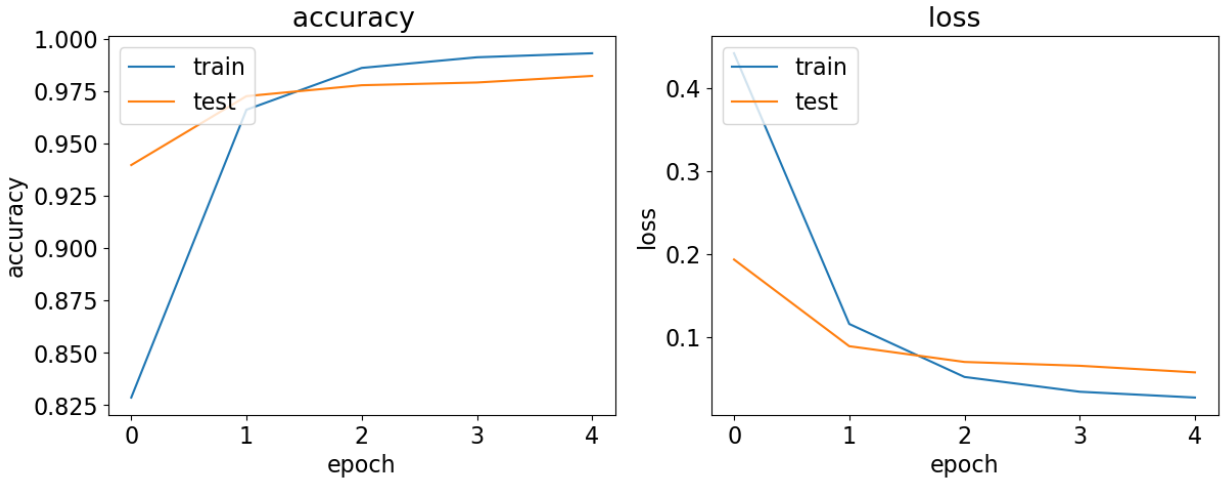

Figure 2.39 – Graphs of accuracy and loss of CNN oversampled Kazakh texts

Classification graphs show very high CNN classification accuracy values over 0.90.

Results of the classification of SMOTE datasets are presented in Table 2.21. Multiclass classification metrics, a confusion matrix, and accuracy and loss graphs of Russian texts with CNN are shown in Figures 2.40, 2.41, and 2.42.

Table 2.21 – Classification of SMOTE datasets

	Russian texts			Kazakh texts		
Classifier	DNN	CNN	LSTM	DNN	CNN	LSTM
Accuracy	0.54	0.79	0.77	0.80	0.83	0.64
Precision-macro	0.51	0.79	0.77	0.82	0.83	0.66
Precision-micro	0.54	0.79	0.77	0.80	0.83	0.64
Precision-weighted	0.51	0.79	0.77	0.82	0.83	0.66
Recall-macro	0.54	0.79	0.77	0.80	0.83	0.64
Recall-micro	0.54	0.79	0.77	0.80	0.83	0.64
Recall-weighted	0.54	0.79	0.77	0.80	0.83	0.64
F1-score macro	0.46	0.79	0.77	0.80	0.83	0.64
F1-score micro	0.54	0.79	0.77	0.80	0.83	0.64
F1-score-weighted	0.46	0.79	0.77	0.80	0.83	0.64

The results of SMOTE data classification show that DNN is inferior to CNN and LSTM for Russian texts. For Kazakh texts, DNN classification results are close to those of CNN and outperform LSTM.

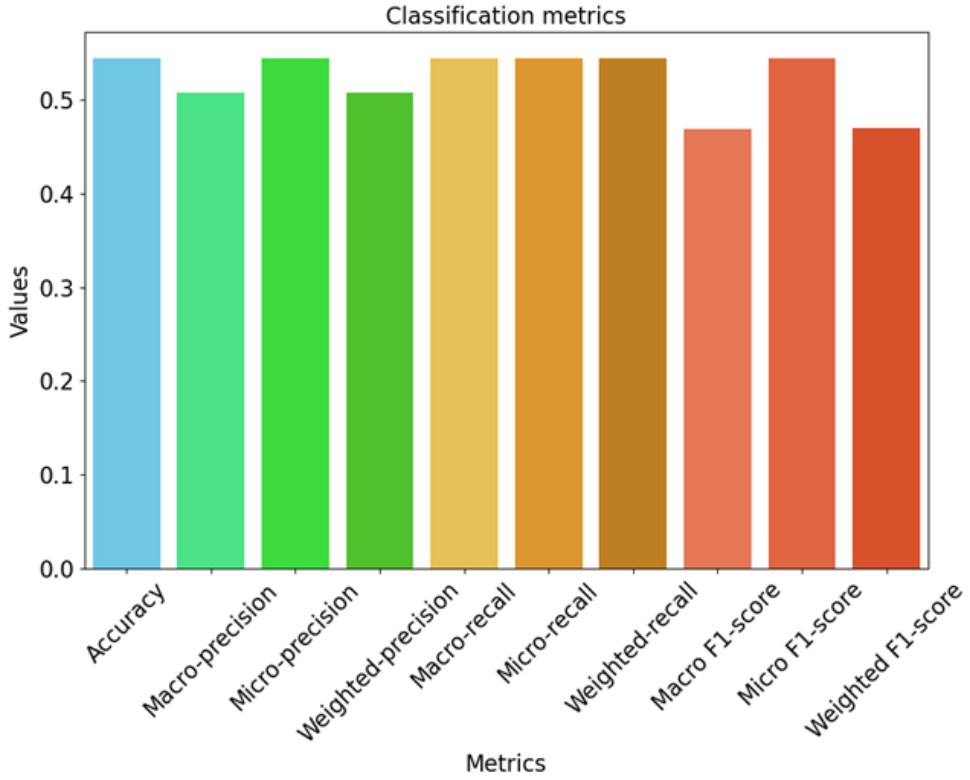

Figure 2.40 – Classification metrics of DNN SMOTE Russian texts

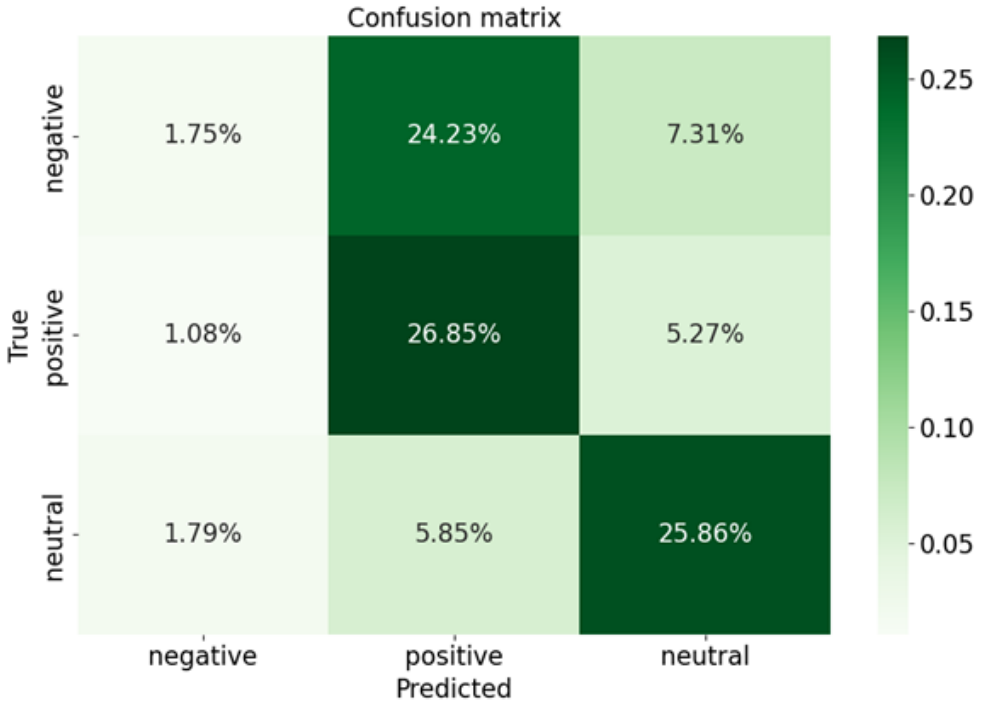

Figure 2.41 – DNN confusion matrix of SMOTE Russian texts

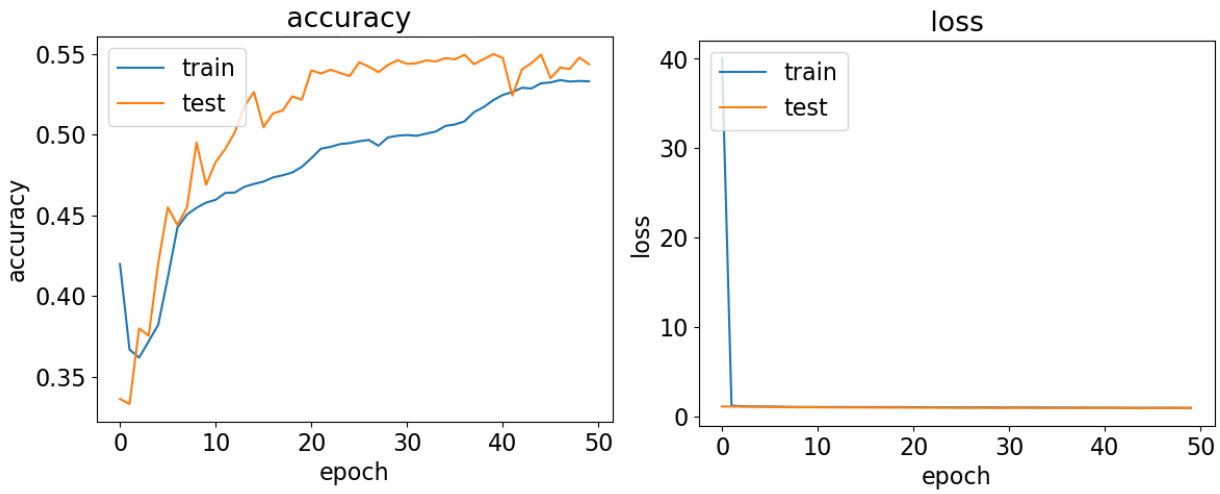

Figure 2.42 – Graphs of accuracy and loss of DNN SMOTE Russian texts

The results of the classification of the undersampled datasets method are shown in Table 2.22. Multiclass classification metrics, a confusion matrix, and accuracy and loss graphs of Russian texts with DNN are presented in Figures 2.43, 2.44, and 2.45.

Table 2.22 – Classification of the undersampled datasets

	Russian texts			Kazakh texts		
Classifier	DNN	CNN	LSTM	DNN	CNN	LSTM
Accuracy	0.32	0.75	0.39	0.62	0.74	0.40
Precision-macro	0.11	0.75	0.67	0.61	0.74	0.45
Precision-micro	0.32	0.75	0.39	0.62	0.74	0.44
Precision-weighted	0.11	0.75	0.67	0.61	0.74	0.44
Recall-macro	0.33	0.75	0.39	0.62	0.74	0.39
Recall-micro	0.32	0.75	0.39	0.62	0.74	0.40
Recall-weighted	0.32	0.75	0.39	0.62	0.74	0.40
F1-score macro	0.16	0.75	0.28	0.61	0.73	0.27
F1-score micro	0.32	0.75	0.39	0.62	0.74	0.40
F1-score weighted	0.16	0.75	0.28	0.61	0.73	0.28

The classification results show that the metrics for DNN, CNN, and LSTM using the undersampling method are lower than those for the resampling and SMOTE methods. At the same time, CNN shows the best classification results among the three neural networks.

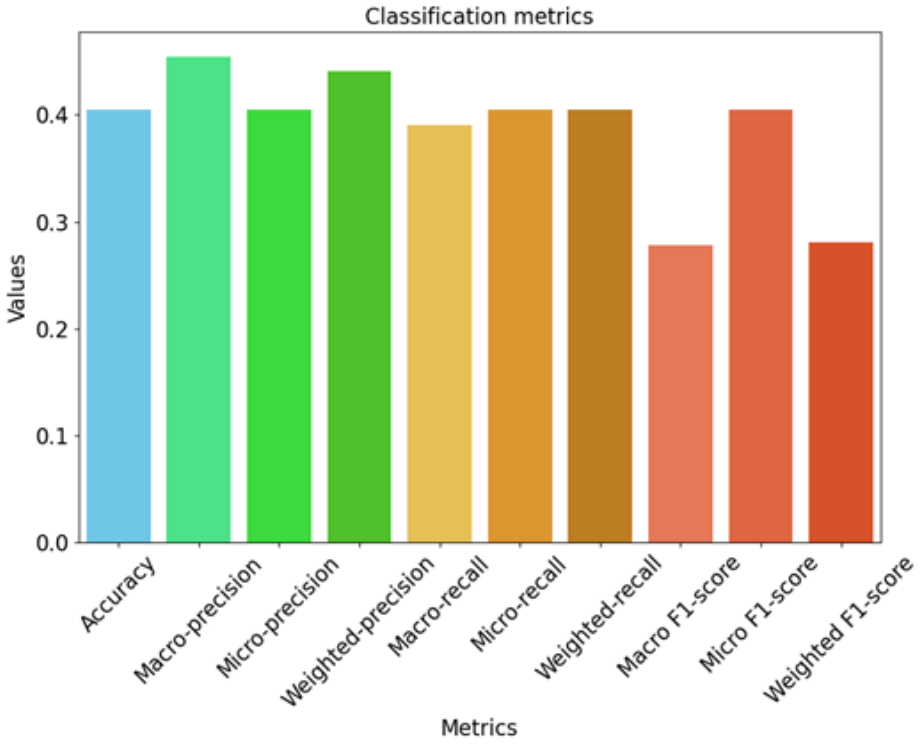

Figure 2.43 – Classification metrics of LSTM undersampled Kazakh texts

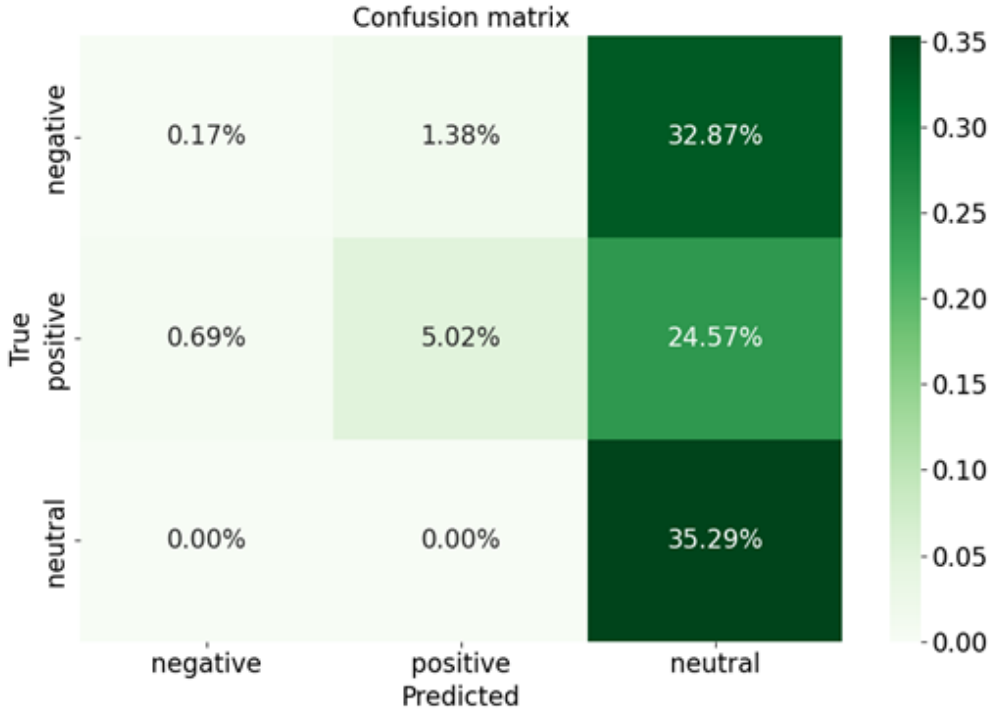

Figure 2.44 – LSTM confusion matrix of undersampled Kazakh texts

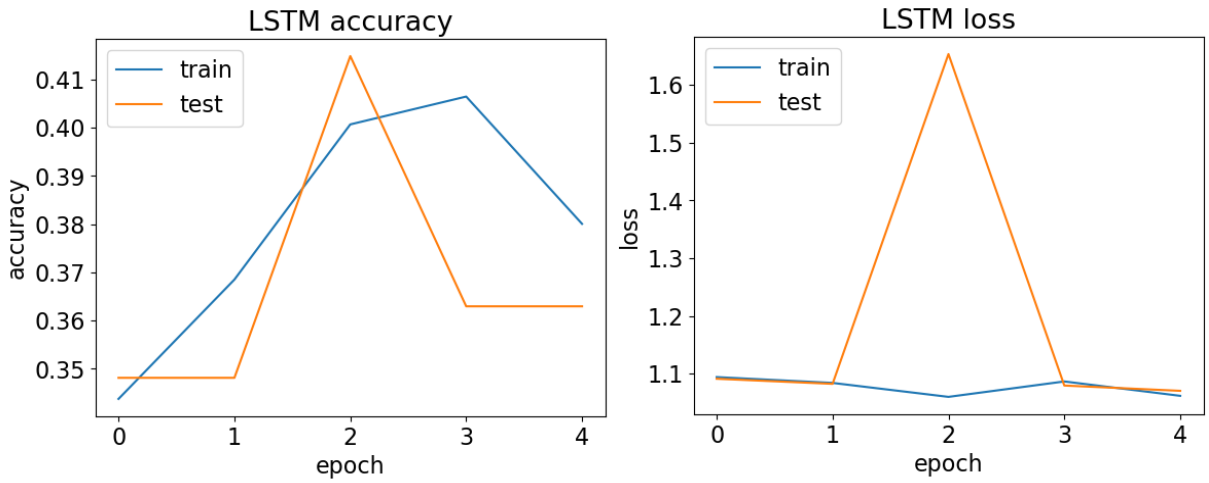

Figure 2.45 – Graphs of accuracy and loss of LSTM undersampled Kazakh texts

Classification graphs demonstrate that LSTM shows acceptable results, reaching values of 0.58–0.60.

The classification of Fasttext-based vectorized texts by NN showed inverse results compared to vectorization with TF-IDF, where convolutional and recurrent NN surpassed the deep neural network. Such values are also relevant for all class balancing experiments. The oversampled and SMOTE models among the balancing methods showed the best results, as in previous experiments. The values close to them also showed imbalanced datasets, demonstrating the effectiveness of CNN for classifying texts vectorized with word embeddings.

After processing text data with deep, convolutional, and recurrent neural networks, classification was implemented using BERT. Before being fed into the neural network, the texts were tokenized using BertTokenizer and converted into tensor vectors. For training, the TFBertModel transformer model was used. The imbalanced and balanced text data classification results were presented in Tables 2.23 and 2.24. In addition, classification graphs for imbalanced Russian and Kazakh texts are shown in Figures 2.46, 2.47, 2.48, 2.49, 2.50, and 2.51.

Table 2.23 – Classification of Russian texts by the BERT neural network

	Imbalanced	Oversampled	SMOTE	Undersampled
Accuracy	0.76	0.92	0.87	0.90
Precision-macro	0.76	0.93	0.87	0.90
Precision-micro	0.76	0.92	0.87	0.90
Precision-weighted	0.78	0.94	0.89	0.91
Recall-macro	0.52	0.92	0.87	0.90
Recall-micro	0.76	0.92	0.87	0.90
Recall-weighted	0.76	0.92	0.87	0.90
F1-score macro	0.43	0.92	0.87	0.90
F1-score micro	0.76	0.92	0.87	0.90
F1-score weighted	0.71	0.92	0.87	0.90

Table 2.24 – Classification of Kazakh texts by the BERT neural network

	Imbalanced	Oversampled	SMOTE	Undersampled
Accuracy	0.96	0.97	0.97	0.94
Precision-macro	0.90	0.97	0.97	0.93
Precision-micro	0.96	0.98	0.97	0.94
Precision-weighted	0.97	0.98	0.98	0.95
Recall-macro	0.88	0.97	0.97	0.94
Recall-micro	0.96	0.97	0.97	0.94
Recall-weighted	0.96	0.97	0.97	0.94
F1-score macro	0.84	0.97	0.97	0.94
F1-score micro	0.96	0.97	0.97	0.94
F1-score weighted	0.95	0.97	0.97	0.94

The BERT classification results showed the effectiveness of this transformer model in sentiment analysis. High accuracy, precision, recall, and F1-score values were achieved for both Russian and Kazakh texts. In addition, BERT deals well with imbalanced data, which the values of the obtained metrics have demonstrated.

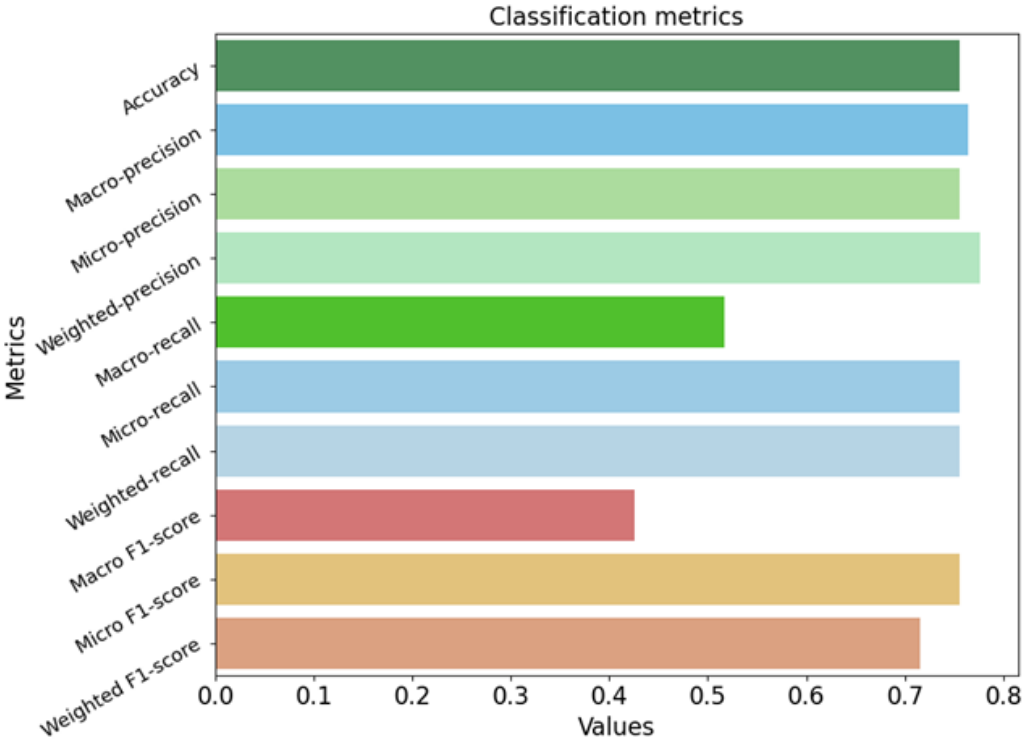

Figure 2.46 – The classification metrics of BERT of imbalanced Russian texts

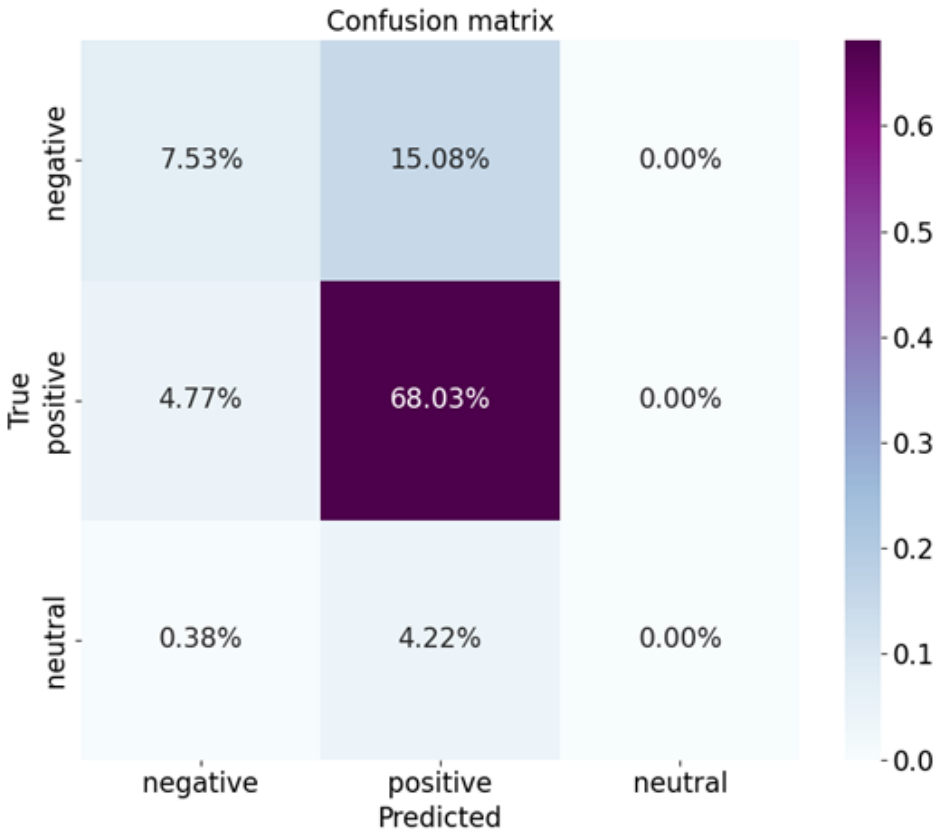

Figure 2.47 – The confusion matrix of BERT of imbalanced Russian texts

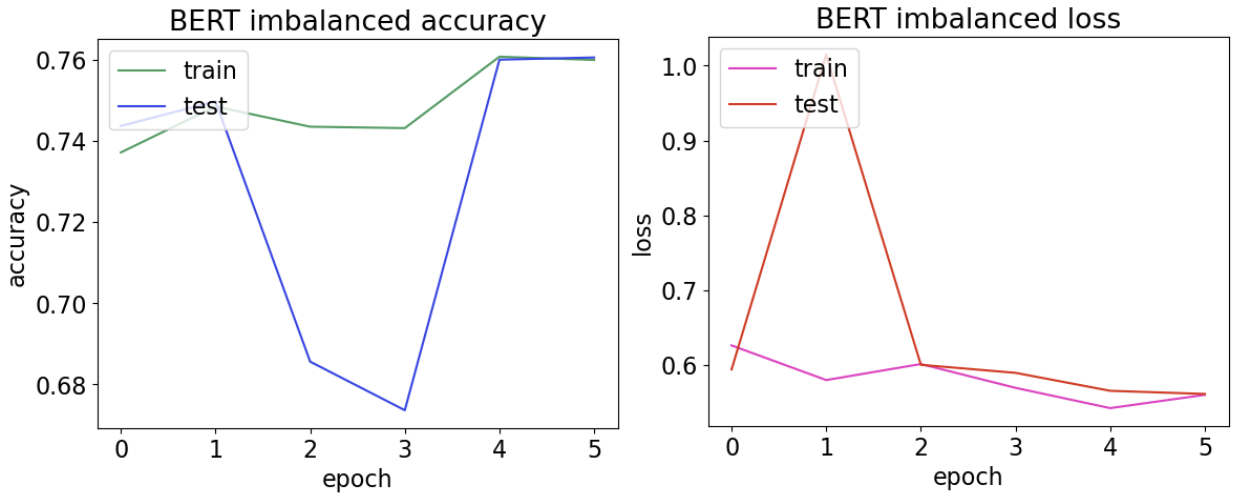

Figure 2.48 – Graphics of accuracy and loss of BERT of imbalanced Russian texts

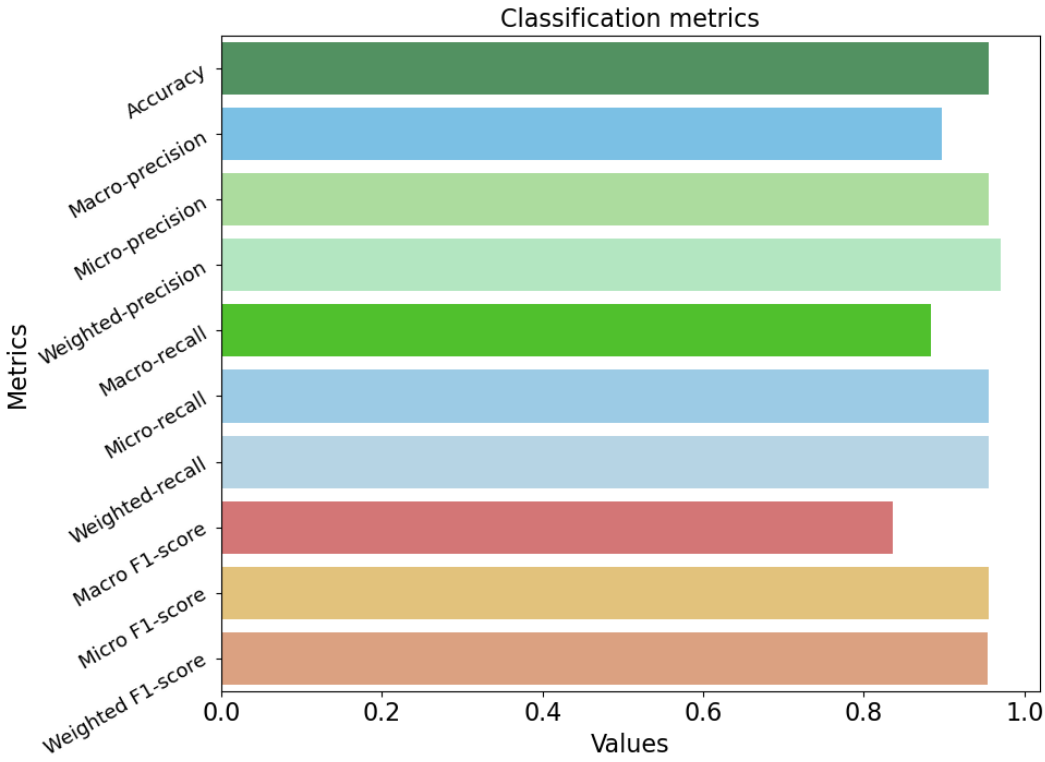

Figure 2.49 – The classification metrics of BERT of imbalanced Kazakh texts

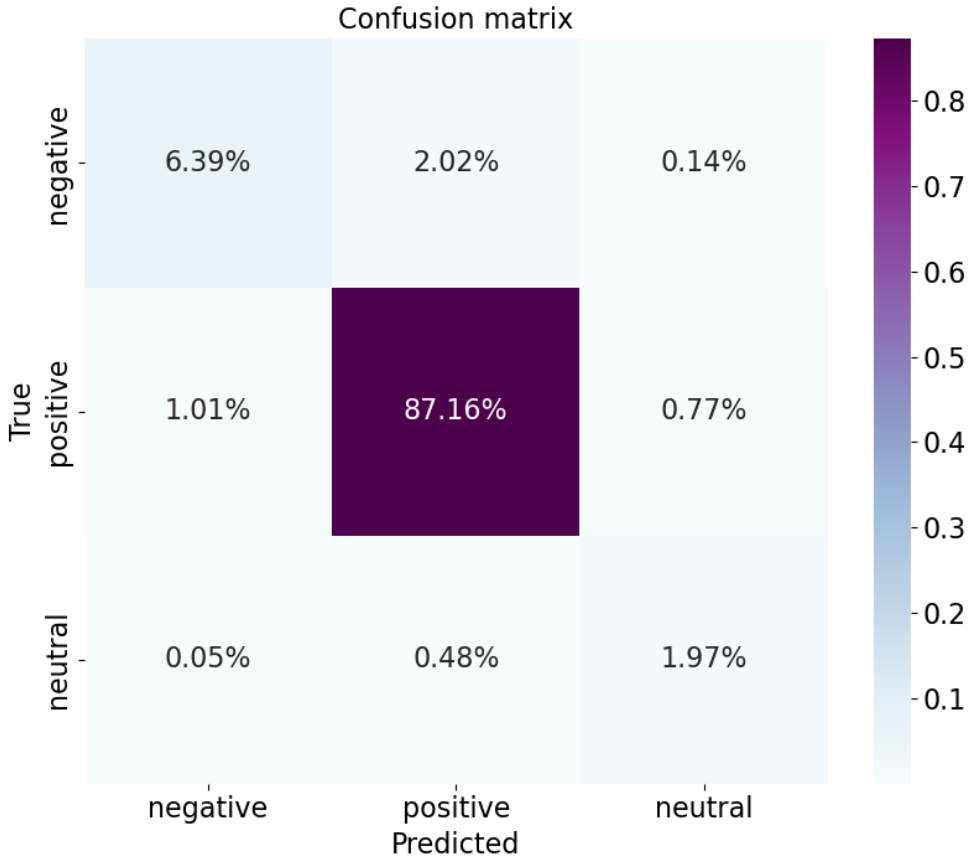

Figure 2.50 – The confusion matrix of BERT of imbalanced Kazakh texts

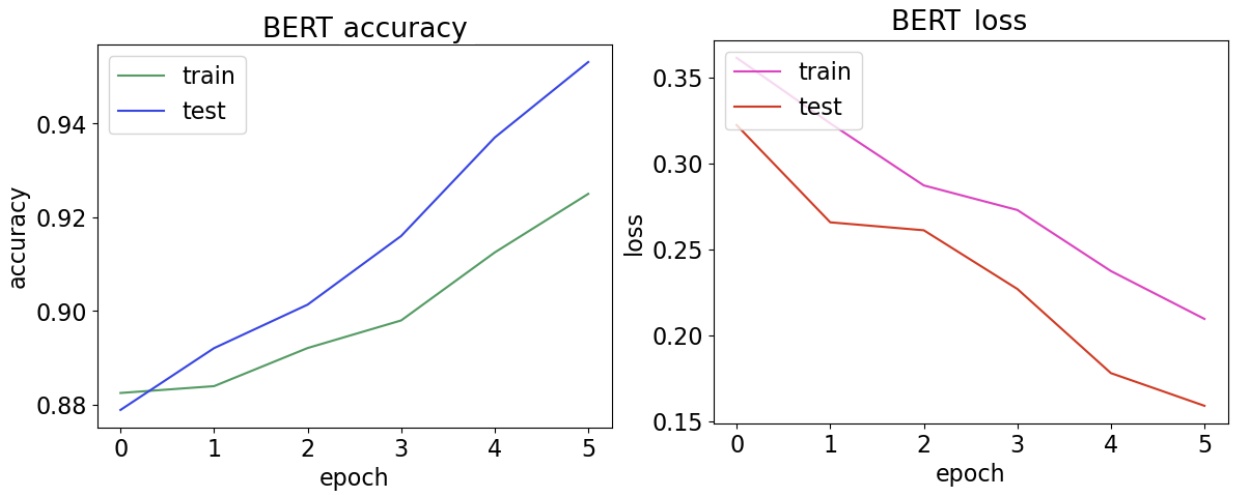

Figure 2.51 – Graphics of accuracy and loss of BERT of imbalanced Kazakh texts

2.5 Conclusions on Chapter 2

Chapter 2 described OMSystem, a multi-faceted system for analyzing user opinions from news portals and social networks. This platform monitors web resources and social networks, uses analytical tools to determine social mood, and supports Kazakh and Russian sentiment dictionaries and machine learning models to find the tone of texts. OMSystem covers the most popular Kazakhstani news portals and social networks, such as Facebook, Vkontakte, Instagram, Twitter, and YouTube.

Next, the system architecture was discussed, which includes modules for platform administration, data collection, processing, analysis, and modeling. The administration subsystem is responsible for configuring social networking APIs, web crawlers, servers, databases, and resource directories. The main data sources are news portals, blogs, and social networks. The data collection subsystem includes a linguistic constructor consisting of sentiment words of the Russian and Kazakh languages belonging to positive, negative, and neutral classes, socio-economic indicators for calculating the social mood index, and a search engine for collecting data from news sites and social networks. The data processing subsystem gives the ability to integrate results from news portals and social networks into the body of a text array using a search engine. Also, this subsystem allows preparing ML models. The analysis and modeling subsystem gets a quantitative analysis of the results of the system, simulates social moods using a production model, visualizes data in the form of reports, charts, and graphs, and uses ML algorithms to determine the sentiment of texts and comments in social networks.

After describing the OMSystem modules, the entity-relationship and functional models of the system were given, as well as the main technical requirements for its operation were described. Then, the basic models and algorithms of SA, including ML algorithms and NN developed for the system, were presented. Next, the basic steps for implementing the data classification module training have been described in detail. These steps consisted of initial data preprocessing, vectorization, class balancing, and data classification. The experimental parts conducted binary and multiclass

classifications with ML algorithms and NN. All metrics and graphs output were presented for each of the experiments performed. The best RF and DT ML models on the oversampled and SMOTE datasets were saved in the files using the Python pickle library. Then a script file that processed a new parsed text with the saved classification model was implemented. In this script, a new text is input data; the saved ML model is a data processing tool; a defined sentiment class of the text is output data. The output data is saved in the corresponding table of the PostgreSQL database of the OMSystem. If it is required to change the trained model, simple corrections to the script are to be made. When the database has grown significantly, the classification models need to be retrained, and the models are saved again.

3 EXPERIMENTAL STUDIES OF THE DEVELOPMENT OF METHODS AND ALGORITHMS

3.1 The computational experiment on the "Vaccination" topic

3.1.1 The purpose of the computational experiment

A relevant topic of vaccination against coronavirus infection is taken for analysis in the experimental part [92-99]. This topic is crucial due to the active vaccination of people worldwide and in Kazakhstan. A large number of news articles have been written on this topic, and users actively comment on various issues related to it. Users' opinions stand out with positive, neutral, and negative sentiments. The experimental part chooses a list of keywords and phrases in the Russian language to monitor the related topics. All words and phrases originally in Russian are translated into English for convenience and proper understanding. These keywords and phrases are Covid [100], Coronavirus [101, 102], "Vaccination in Kazakhstan," "Fear of vaccination," "Vaccine rejection," "Lack of confidence in the vaccine," "Vaccine effectiveness," "Choice of the vaccine," "Russian vaccine," Pfizer [103], Sinopharm [104], Sinovac, QazVac, Hayat, "Sputnik V," and Tsoi [1, p. 31].

3.1.2 Data for the computational experiment

To analyze the "Vaccination" topic, data were obtained from news resources and social networks, such as Vkontakte, Facebook, Instagram, and Youtube, using the OMSystem web crawler. The OMSystem [105] performs calculations for two periods: the 10th of January, 2021 to the 30th of May, 2021 (Table 3.1) and the 1st of July, 2021 to the 12th of August, 2021 (Table 3.3), and two groups of cities: Almaty (the largest city of Kazakhstan) and Nur-Sultan (Astana), and large regional cities of Kazakhstan. The choice of these cities for analysis was made due to several facts. First, the population of Almaty, Nur-Sultan, and other large cities is almost 100% covered with information technologies. Citizens of these cities are also the most active users of social networks, and their opinions are very important, reflecting the general trend in the country. It is also important to get the public's opinion from different regional cities because the epidemiological situation with vaccination and the availability of vaccines significantly varied in all the regions of Kazakhstan. The stated monitoring dates were chosen because the start of vaccination against the Covid-19 campaign started in January 2021. The first phase of vaccination finished by the beginning of summer. Only two vaccines, "Sputnik V" and QazVac, were available in the first phase. Then in May and June 2021, three more vaccines, Hayat-Vax, Sinovac, and Sinopharm, were imported. Nevertheless, these vaccines quickly ran out in Almaty, Nur-Sultan (Astana), and some other cities in the second vaccination phase. In addition, it resulted in a large number of negative user comments. So, it was essential to monitor these two periods of the vaccination campaign to estimate the level of interest and social mood in this topic.

3.1.3 Models and algorithms of the computational experiment

The steps for the OMSystem functionality are shown in Figure 3.1. When launched, the OMSystem web crawler analyzes user texts and comments from a given list of sources (Kazakh news portals, social networks, and blogs). The parsed texts are collected in the created PostgreSQL database.

The aggregated PostgreSQL database texts and comments on the topic of vaccination have been applied the several steps:

– Preprocessing text
– Stemming
– Vectorization
– Assigning sentiment labels

In the preprocessing step, all words are transformed to the lowercase register. Then punctuation marks, digits, and other special symbols without significant meaning are removed. Additionally, it is required to delete frequent words (i.e., stop words such as ‘and,’ ‘or,’ ‘in,’ ‘on,’ ‘at,’ ‘for,’ etc.) that do not bring any significant meaning. However, the ‘to be’ and ‘is’ stop words are left because they are met in expressions such as “to be vaccinated,” “is vaccinated,” and others, which are important for the analyzed topic. The stemming step reduces the number of words with similar meanings by eliminating affixes and endings to gain their roots. Russian words are processed by ‘SnowballStemmer’ from the Python NLTK library. The text vectorization step transforms texts into a numeric vector representation to which ML algorithms are applied. The vectorization is done using the *tf-idf* metric that considers the importance of words in the text. After the texts are vectorized, the trained ML models are applied to label them in three sentiment classes. Those texts and user comments whose sentiment was not determined by the dictionary approach and machine learning algorithms were labeled as undefined sentiment. Next, the number of words in texts and comments is counted, and the most frequently used ones are displayed in summary tables.

3.1.4 Evaluation of the accuracy of the computational experiment

A computational experiment was performed over two monitoring periods. The values of the level of interest in the topic, the level of activity of the topic discussion, and the level of social mood are presented in Tables 3.1 and 3.3 [1, p. 33,34].

Table 3.1 – Topic analysis for period 1

Resource Set	News portals, Vkontakte, Facebook, Instagram, Youtube	Vkontakte, Facebook, Instagram, Youtube
1	2	3
Search Period	from “10-01-2021” to “30-05-2021”	
Location	Cities of Almaty and Nur-Sultan (Astana)	Large regional cities of Kazakhstan

Continuation of Table 3.1

1	2		3	
Number of results (texts + comments)	19340		1228	
Number of texts	4919		122	
Number of comments	14421		1106	
The level of social mood by results	Positive	8944	Positive	396
	Negative	8152	Negative	683
	Neutral	1082	Neutral	66
	Undefined	1162	Undefined	83
The level of social mood by texts	Positive	3829	Positive	56
	Negative	960	Negative	43
	Neutral	123	Neutral	11
	Undefined	7	Undefined	12
The level of social mood by comments	Positive	5115	Positive	340
	Negative	7192	Negative	640
	Neutral	959	Neutral	55
	Undefined	1155	Undefined	71
The level of topic discussion activity in society	0.48%		0.08%	
The level of interest in the topic in society	491%		12.2%	
Engagement level			Engagement level	

Continuation of Table 3.1

1	2			3			
Views	9M			341K			
Comments	14K			1K			
Reposts	2K			249			
Likes	32K			2K			
Dislikes	2K			305			
Total Engagement Level	9M			345K			
Popular words				Popular words			
by texts		by comments		by texts		by comments	
Word	Frequency of use	Word	Frequency of use	Word	Frequency of use	Word	Frequency of use
Coronavirus	2374 (3.51%)	To be	1598 (1.38%)	Coronavirus	118 (1.29%)	To be	148 (1.52%)
Kazakhstan	1811 (2.68%)	Vaccine	1112 (0.96%)	To be	113 (1.24%)	Person	138 (1.42%)
Vaccine	824 (1.22%)	Person	1097 (0.94%)	Area	112 (1.23%)	Vaccine	105 (1.08%)
Person	653 (0.96%)	Can	564 (0.48%)	Kazakhstan	101 (1.11%)	People	92 (0.94%)
Covid-19	540 (0.80%)	Is	532 (0.46%)	Vaccine	68 (0.74%)	Kazakhstan	62 (0.63%)
Day	526 (0.77%)	Kazakhstan	477 (0.41%)	Aktyubinsk	59 (0.64%)	Year	53 (0.54%)
Vaccination	524 (0.77%)	Necessary	472 (0.40%)	Year	55 (0.60%)	Necessary	44 (0.45%)
News	502 (0.74%)	Year	432 (0.37%)	Person	53 (0.58%)	Virus	42 (0.43%)
Almaty	443 (0.65%)	People	415 (0.35%)	Vaccination	52 (0.57%)	Country	41 (0.42%)
New	433 (0.64%)	Virus	370 (0.32%)	Tenge	49 (0.53%)	Can	40 (0.41%)
Country	427 (0.63%)	To speak	337 (0.29%)	Reference	48 (0.52%)	Vaccination	40 (0.41%)
To be	394 (0.58%)	Country	327 (0.28%)	Zone	46 (0.50%)	Power	35 (0.36%)
Case	371 (0.54%)	To do	327 (0.28%)	Case	42 (0.46%)	Good	29 (0.29%)
The first	358 (0.53%)	Vaccination	325 (0.28%)	To attach	40 (0.43%)	Russia	29 (0.29%)
Area	325 (0.48%)	To tell	317 (0.27%)	Later	39 (0.42%)	Simply	29 (0.29%)
Zone	310 (0.45%)	To want	312 (0.27%)	Child	36 (0.39%)	Covid	29 (0.29%)
Ministry of Health	282 (0.41%)	To know	297 (0.25%)	Can	34 (0.37%)	Child	28 (0.28%)
To reveal	281 (0.41%)	Money	286 (0.24%)	Doctor	33 (0.36%)	Inoculation	28 (0.28%)
Tsoi	274 (0.40%)	Another	281 (0.24%)	Healthcare	32 (0.35%)	World	27 (0.27%)
Year	271 (0.40%)	Good	279 (0.24%)	Region	31 (0.34%)	Quarantine	27 (0.27%)

The sentiment graphs of the 1st period for the cities of Almaty and Nur-Sultan (Astana) and large regional cities are presented in Figure 3.1.

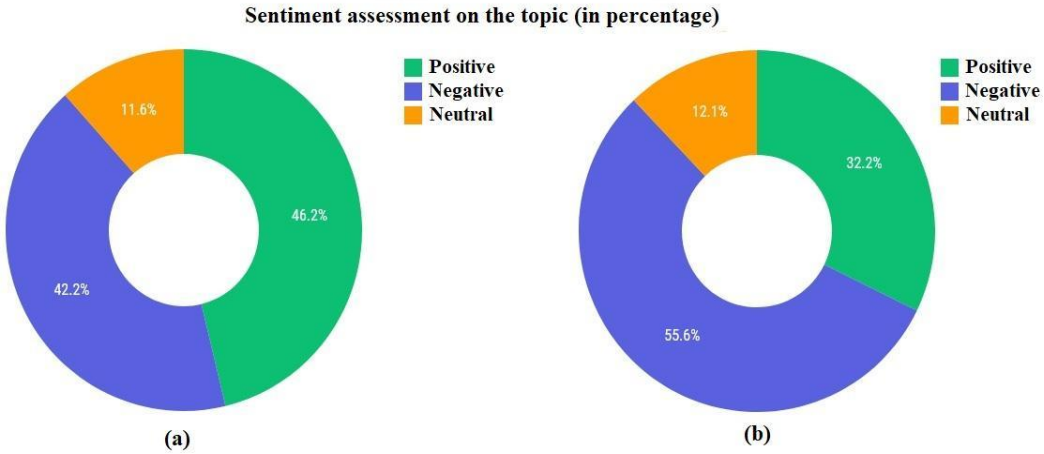

Figure 3.1 – Sentiment analysis for the 1st period – (a) Almaty and Nur-Sultan (Astana), (b) large regional cities

As a result of the analysis of Table 3.1, the content of texts and comments was evaluated, considering the list of the most frequently used words. Having studied the analysis of popular words in the context of regional cities, one can notice their similarity with the content in the cities of Almaty and Nur-Sultan (Astana). According to the results obtained, the level of interest in this topic is significantly higher in the cities of Almaty and Nur-Sultan (Astana) (491%) than in other large regional cities (12.2%). In addition, the topic's discussion level is also higher in the two main cities of the country (0.48%) than in others (0.08%). However, the social mood of texts and comments differs significantly: texts mostly express positive sentiments, while comments are mostly negative. This indicates that texts on social networks are positively covering the topic of vaccination, while people's attitude towards it is the opposite. After the analysis was completed, the system created a final report, where the received texts and comments were also manually examined. Examples of texts and comments are presented in Table 3.2. The public reacted extremely negatively to all government measures related to vaccination in the winter and spring periods [1, p. 37].

Table 3.2 – Texts and comments for period 1

№	Date	Sentiment	Text	Sentiment	Comments
1	2	3	4	5	6
1	19–01–2021	positive	The Head of the Government instructed the Ministry of Health of the Republic of Kazakhstan to ensure the readiness of medical organizations for the start of the mass vaccination of the population with the "Sputnik V" vaccine from February 1.	negative	We do not need your vaccine; go to poison others with these chemicals
				negative	Look for idiots elsewhere. Madhouse

Continuation of Table 3.2

1	2	3	4	5	6
				negative	Experiments on humans are like this, especially when all sane scientists deny the vaccine's effectiveness. So it is time to be vaccinated!
				negative	Madness! Even in Russia, they did not really test it. Did they decide to test it on the Kazakhs? What a madhouse?
				positive	Ready to become a test subject for a fee. Where to go?
2	30–01–2021	positive	First of all, vaccines against COVID-19 will be sent to health workers in eight regions of Kazakhstan, in which there is a high incidence of coronavirus. On Monday, February 1, vaccination against coronavirus starts, but vaccines have not yet been brought to the Aktobe region.	negative	Now we will look before the vaccine and after.
				positive	Soon, vaccinations will start everywhere, and it will be much more difficult for the coronavirus to spread. So the epidemic will end.
				negative	Why do not we start with the deputies?
				positive	The entire Government with their families must be at the forefront. May they get the best, we undoubtably agree this time
3	08–05–2021	positive	In Kazakhstan, 34% of residents have changed their attitude towards vaccination against COVID-19 for the better. At the same time, 23% are skeptical against vaccinations. 9% have recently changed their minds in a negative direction. 4% do not trust the vaccine, 3% do not dare to get vaccinated, as they recently got sick. Who do you belong to?	negative	The data is not exact. More than half of the population of Kazakhstan do not believe
				negative	Where do these statistics come from? For example, no one asked me)
				negative	I did it because I wanted to save my family. The opinions of others do not matter, but my health and safety do.
				neutral	There is no way
				negative	Suicidal people are getting vaccinated
4	01–02–2021	positive	Kazakhstanis are intended to be classified by color in terms of whether they passed the polymerase chain reaction (PCR) test and what the result was, zakon.kz reports. According to the press service of the Republic of Kazakhstan, the data will be reflected in the "Ashyq" application developed by the Ministry of Digital Development, Innovation and Aerospace Industry jointly with the Ministry of Health of the Republic of Kazakhstan.	negative	I am crazy with all sorts of this bullshit to torture the people
				negative	Well! It is straight racism: yellow and red. I disagree
				negative	It is a total control under the guise of coronavirus
				negative	"Divide and conquer" is a working scheme from the ancient time
				negative	Scumbugs! I knew it would come to this!

Continuation of Table 3.2

1	2	3	4	5	6
5	26–01–2021	positive	Mass vaccination of the population against COVID-19 will begin in Kazakhstan on February 1, Kazakh Health Minister Alexei Tsoi said during a government meeting on Tuesday. It is planned to vaccinate up to six million people by the end of the year	negative	Are we their guinea pigs or what? Go away. Check your vaccine to the end first, then to the people.
				negative	It is necessary to start with the ministers and deputies. Whoever survives will remain in office, who does not survive, and to hell with them!
				negative	They want to test the effectiveness of the vaccine on us
				negative	Let them first try this vaccine on themselves. We did not invent this infection; it was not for us to die for it.
				positive	Yes. Nevermind. As they said, it will be “finally,” and vaccination is already in full swing

Table 3.3 – Analysis by topics for period 2

Resource Set	News portals, Vkontakte, Facebook, Instagram, Youtube		Vkontakte, Facebook, Instagram, Youtube	
1	2		3	
Search Period	from “07/01/2021” to “08/12/2021”			
Location	Cities of Almaty and Nur-Sultan (Astana)		Large regional cities of Kazakhstan	
Number of results (texts + comments)	2133		157	
Number of texts	1285		69	
Number of comments	848		88	
The level of social mood by results	Positive	1029	Positive	52
	Negative	955	Negative	58
	Neutral	80	Neutral	10
	Undefined	69	Undefined	37
The level of social mood by texts	Positive	739	Positive	21
	Negative	544	Negative	18
	Neutral	1	Neutral	5
	Undefined	1	Undefined	25
The level of social mood by comments	Positive	290	Positive	31
	Negative	411	Negative	40
	Neutral	79	Neutral	5
	Undefined	68	Undefined	12

Continuation of Table 3.3

1	2			3			
The level of topic discussion activity in society	0.01%			0.03%			
The level of interest in the topic in society	128%			6.9%			
Engagement level				Engagement level			
Views	34K			42K			
Comments	848			97			
Reposts	825			46			
Likes	2K			123			
Dislikes	35			0			
Total Engagement Level	38K			42K			
Popular words				Popular words			
by texts		by comments		by texts		by comments	
Word	Frequency of use	Word	Frequency of use	Word	Frequency of use	Word	Frequency of use
To be	1786 (1.00%)	Vaccine	143 (1.59%)	Coronavirus	52 (2.15%)	Person	11 (1.59%)
Kazakhstan	1630 (0.91%)	Person	82 (0.91%)	Reference	44 (1.82%)	Vaccine	11 (1.59%)
Person	1493 (0.83%)	To be	63 (0.70%)	To attach	40 (1.66%)	To be	11 (1.59%)
Year	1268 (0.71%)	Vaccination	46 (0.51%)	Area	36 (1.49%)	Simply	8 (1.16%)
Coronavirus	1213 (0.68%)	Kazakhstan	39 (0.43%)	Strain	22 (0.91%)	People	6 (0.87%)
Vaccination	1201 (0.67%)	Later	38 (0.42%)	Pavlodar	22 (0.91%)	Though	4 (0.58%)
Vaccine	1018 (0.57%)	Child	33 (0.36%)	Kazakhstan	21 (0.87%)	Level	4 (0.58%)
Case	808 (0.45%)	Can	33 (0.36%)	Heading	20 (0.83%)	To buy	4 (0.58%)
Country	750 (0.42%)	To speak	33 (0.36%)	Vaccine	20 (0.83%)	Proper	4 (0.58%)
Infection	724 (0.40%)	Necessary	30 (0.33%)	Url	20 (0.83%)	Virus	4 (0.58%)
Covid-19	722 (0.40%)	To know	30 (0.33%)	To be	19 (0.78%)	Guilty	4 (0.58%)
Can	692 (0.38%)	Is	30 (0.33%)	Year	17 (0.70%)	Strain	3 (0.43%)
Area	674 (0.37%)	Covid	29 (0.32%)	Person	16 (0.66%)	To make	3 (0.43%)
July	638 (0.35%)	People	27 (0.30%)	Can	14 (0.58%)	Inoculation	3 (0.43%)
More	635 (0.35%)	To do	24 (0.26%)	Vaccination	14 (0.58%)	In a row	3 (0.43%)
Day	633 (0.35%)	Year	24 (0.26%)	Health care	13 (0.53%)	Life	3 (0.43%)
Work	631 (0.35%)	Doctor	24 (0.26%)	Pavlodar	12 (0.49%)	Any	3 (0.43%)
New	630 (0.35%)	To be ill	24 (0.26%)	Doctor	12 (0.49%)	Small	3 (0.43%)
Coronavirus	612 (0.34%)	To tell	23 (0.25%)	To work	11 (0.45%)	To know	3 (0.43%)
Patient	603 (0.33%)	Virus	21 (0.23%)	To become	10 (0.41%)	Delta	3 (0.43%)

The sentiment graphs of the 2nd period for the cities of Almaty and Nur-Sultan (Astana) and large regional cities are presented in Figure 3.2.

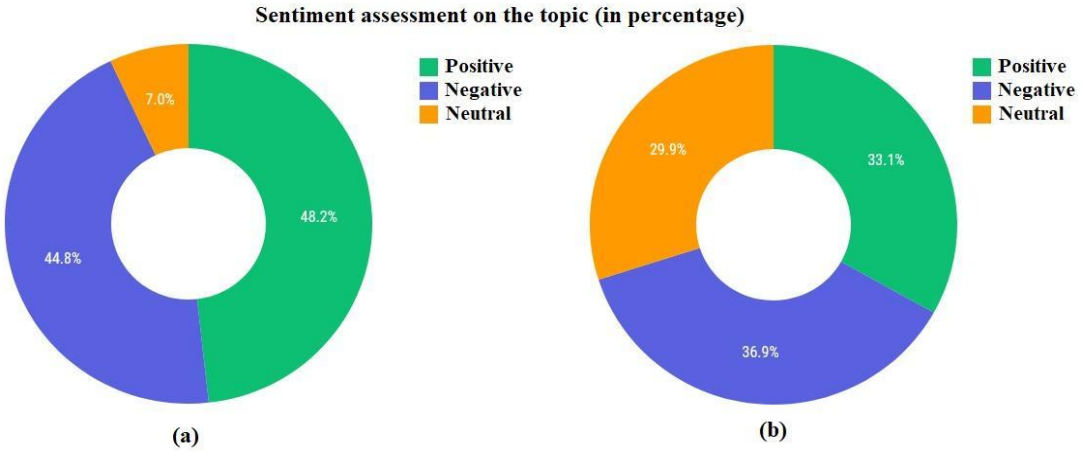

Figure 3.2 – Evaluation of the second period – (a) Almaty and Nur-Sultan (Astana), (b) major regional cities

The analysis of Table 3.3 suggests that there remains a high level of public interest in the topic during the summer. The level of interest in this topic is higher in the cities of Almaty and Nur-Sultan (Astana) (128%) than in large regional cities (6.9%). The level of topic discussion activity is lower than in period 1. It is caused by fewer comments on the considered topics during a shorter time of monitoring. The following values are gained in the context of cities: 0.01% for Almaty and Nur-Sultan and 0.03% for the large regional cities. The level of the social mood of texts and comments shows a situation similar to period 1. This period's obtained texts and comments were also manually analyzed to reveal interesting points. It is noted that texts cover the planned children's vaccination topics, the appearance of new strains of coronavirus, the increase in the number of cases of unvaccinated people's disease, and the supply of a new Chinese vaccine to the country. The corresponding examples of the texts and comments are presented in Table 3.4 [1, p. 39,40].

Table 3.4 – Texts and comments for period 2

№	Date	Sentiment	Text	Sentiment	Comments
1	2	3	4	5	6
1	21–07–2021	positive	It is planned to start vaccination of children against coronavirus in Kazakhstan at the end of this year. What do they plan to vaccinate with, and will vaccination be voluntary?	negative	People stand up to protect children. Healthcare is not able to protect children from vaccine refusal

Continuation of Table 3.4

1	2	3	4	5	6
				positive	In the USA, all children are vaccinated. If we can protect our children, why not? One of my acquaintances received the 2nd dose. She is a girl of 14 years old. Everything is fine. Everybody there voluntarily vaccinates children. We always lag behind. They tell us to take a step ahead, but we take two steps back. Sadly. Therefore, we do not grow, and we do not develop
				negative	You must vaccinate yours!!! If you do not have brains, your children do not have one either!
2	26–07–2021	positive	We have 84% of our intensive care beds filled. They are loaded with patients who have not received vaccination against coronavirus infection and are now in a severe condition – 248 patients. Of these, 77 people are in extremely serious condition. This number scares us as doctors. We are reaching the peak that was last summer," said the head of the public health department of the capital, Timur Muratov.	negative	It is for those anti-vaccinators who can read and hear not only their cries about freedom. As soon as each of them understands the inhuman basis of personal freedom, opposed to the freedom of others, or rather other people, he / she is obliged to think.
				negative	The relatives of the deceased can sue the Shymkent anti-vaccinator (I forgot her name, sorry), which actively urges everyone to refuse vaccination
				negative	Let us gather money for the monuments to the killer doctors! Who sold out for premiums and killed people with the vaccine!!! They also lie!!! I am waiting for the heavenly punishment for you !!!
3	10–08–2021	positive	The first lot of the Chinese vaccine Sinopharm arrived in Kazakhstan on August 10, 2021. Following the negotiations with the People's Republic of China, an aircraft with the first batch of Sinopharm vaccine against coronavirus arrived in Almaty at the warehouses of the SK-Pharmacy Single Distributor.	positive	Good vaccine! It was recognized by WHO and Europe. Vaccinate. Health to all
				positive	Hayat is a good vaccine, so this one too. It is judging by my own example.
				negative	I doubt very much that WHO is responsible for our health and life.
				negative	Perhaps the quality of this vaccine is good (I do not argue). Just answer what it is made of, what is included in the composition?
				positive	Hooray, I'm going to put it. Do not miss the vaccine that came at the expense of the people.

Continuation of Table 3.4

1	2	3	4	5	6
4	11–08–2021	negative	The Ministry of Health of the Republic of Kazakhstan notes that 99.9% of the incidence of Covid-19 falls on unvaccinated citizens. In assessing the effectiveness of vaccination, it was found that 99.9% of the incidence of coronavirus infection falls on unvaccinated, while the proportion of patients after vaccination was only 0.1%, such data reported today by the Minister of Health Alexei Tsoi at a meeting of the Government.	negative	And it is true! Three friends are now in the hospital. There are no vaccinated people in the wards.
				negative	What is the percentage of re-illnesses? If such statistics do not even exist, then this means that there are no more patients, and then the question arises, why vaccinate those who have already been ill?
5	02–07–2021	negative	The "Indian" strain was found in all regions of Kazakhstan and the cities of Nur-Sultan, Almaty, Shymkent, zakon.kz reports. According to the Ministry of Health, the department carried out PCR screening of positive laboratory samples obtained from patients with coronavirus infection (CVI).	negative	There is no Indian strain. They said officially. It is ours who are lying to make people run to shoot up drugs. The day before yesterday, it was in 4 regions, and yesterday it was in all. Walked in the wind
				positive	Well, there is no point in getting vaccinated!
				negative	Do not write Indian. People in India know how upset it is
				negative	The Hindus themselves say there is no such thing.
				positive	These viruses appear abroad, but they come to us to die.

3.2 Conclusions on Chapter 3

The results of the experiments were carefully studied and analyzed to understand the cause of negative public sentiment. Based on the data obtained by the OMSystem [104, 105], it was concluded that Kazakhstanis mostly do not trust governmental methods to combat the pandemic. It should also be noted that users of social networks cannot identify fake news or trust unverified information. Thus, an experiment on vaccination against coronavirus disease makes it possible to understand the public's attitude and the government's activities by assessing SA and the semantic content of comments. As a result, this will allow you to conduct a research policy for the

population correctly, determine the style of submission of information material, accelerate the introduction of such large-scale state tasks, and ensure the preservation of public health. In addition, the OMSystem is used as a serious analytical tool for assessing users' perceptions of social and economic life, which will make it possible to quickly explain to the population, identify alarming factors of the public, and evaluate social mood.

4 DEVELOPMENT OF THE ESM SOCIAL MOOD ANALYSIS MODULE

4.1 The main purpose of the application

The OMSystem provides an opportunity to effectively assess the level of public mood using metrics based on special social media marketing management formulas. Previously, a detailed description of these indicators was given: the level of interest in the topic in society, the level of activity of the topic discussion, and the level of social mood. These indicators proved effective for the analysis of mood on certain given topics. The previous chapter provided a detailed analysis of topics related to coronavirus vaccination. The OMSystem [106–108] calculated metrics of this topic by texts containing search keywords. Based on the analysis results, summary tables containing metric data values for all key keywords were compiled, and general values of the topic assessment for specified time intervals were presented.

To represent the analysis of the mood of society by topic, the eSM social mood analysis module was also developed on the Django Python framework, which is an application that analyzes data obtained with the OMSystem platform [109–111]. The interaction between OMSystem and eSM is shown in Figure 4.1.

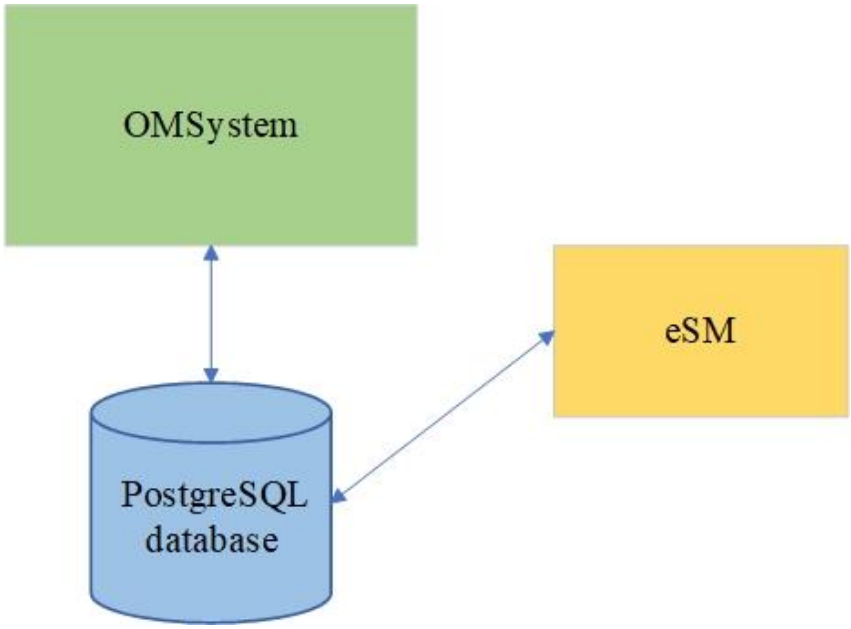

Figure 4.1 – Interaction of OMSystem and eSM

This module performs the following basic functions:

1. Creating basic categories of social mood analysis topics.
2. Retrieving quantitative data for each of the topics from the OMSystem database [112, 113].
3. Counting the level of interest in the topic in society, the level of activity of the topic discussion, and the level of social mood.
4. Visualizing the results as graphs and a summary table.

When creating the application, the main categories of social topics in society were identified. They were given meaningful topics, which also searched keywords in the

database. The main categories are presented as follows: political system, civil society, civil security, public safety, economy, education, health care, culture and sport, ecology.

By default, each category already has a list of topics changed directly in the program itself. In this text, these words are translated into English for convenience.

– Politics – [‘akimat,’ ‘corruption,’ ‘government,’ ‘bribes,’ ‘parliament,’ ‘elections,’ ‘rallies,’ ‘majilis,’ ‘senate,’ ‘constitution,’ ‘authorities,’ ‘region,’ ‘Nazarbayev,’ ‘Tokayev’].

– Civil society – [‘society,’ ‘politics,’ ‘civil society,’ ‘social policy,’ ‘state structures,’ ‘constitution,’ ‘institution of power,’ ‘democracy,’ ‘elections,’ ‘voting,’ ‘equality,’ ‘rights,’ ‘legal state,’ ‘parties,’ ‘electoral bodies’].

– Civil security – [‘defence,’ ‘arms,’ ‘crime,’ ‘robbery,’ ‘murder,’ ‘war,’ ‘terrorism,’ ‘justice,’ ‘security,’ ‘policemen,’ ‘terrorists,’ ‘threats,’ ‘danger,’ ‘clashes,’ ‘riots’].

– Economics – [‘microeconomics,’ ‘macroeconomics,’ ‘inflation,’ ‘capital,’ ‘devaluation,’ ‘crisis,’ ‘default,’ ‘prices,’ ‘monopoly,’ ‘currency,’ ‘money supply,’ ‘exchange rate,’ ‘GDP,’ ‘profit,’ ‘taxes’].

– Education – [‘school,’ ‘university,’ ‘higher education,’ ‘Ministry of Education,’ ‘teaching,’ ‘classes,’ ‘school students,’ ‘students,’ ‘school cafeterias,’ ‘teachers,’ ‘lecturers,’ ‘director,’ ‘books,’ ‘training,’ ‘office supplies’].

– Healthcare – [‘polyclinic,’ ‘hospital,’ ‘health,’ ‘coronavirus,’ ‘medicine,’ ‘disease,’ ‘doctor,’ ‘medical care,’ ‘medical services,’ ‘paid services,’ ‘treatment,’ ‘infection,’ ‘virus,’ ‘pathogens,’ ‘COVID-19’].

– Culture and sports – [‘sports grounds,’ ‘stadium,’ ‘competitions,’ ‘sections,’ ‘gym,’ ‘health,’ ‘physical education,’ ‘children’s sports,’ ‘university sports,’ ‘pool,’ ‘championship,’ ‘football,’ ‘hockey,’ ‘Europa League,’ ‘Champions League’].

– Public safety – [‘mothers of many children,’ ‘large families,’ ‘orphanages,’ ‘pension contributions,’ ‘social support,’ ‘social protection,’ ‘social policy,’ ‘social institutes,’ ‘labor market,’ ‘people’s rights,’ ‘charity,’ ‘privileges,’ ‘grants,’ ‘salary,’ ‘insurance’].

– Ecology – [‘ecology,’ ‘biology,’ ‘conservation,’ ‘nature management,’ ‘reserve,’ ‘garden,’ ‘pollution,’ ‘air,’ ‘water,’ ‘release,’ ‘waste,’ ‘endangered species,’ ‘populations,’ ‘animals,’ ‘plants’].

4.2 Application Functionality

About 10-15 of the most important and relevant topics are initially set for each category. Their number can be effectively varied both in software code and in the form of the main application window, developed on the Django framework in Python, which uses MVC (Model – View – Controller) design technology. The main program window

is shown in Figure 4.2, and the form of topic selection by categories is shown in Figure 4.3.

Figure 4.2 – Topic categories

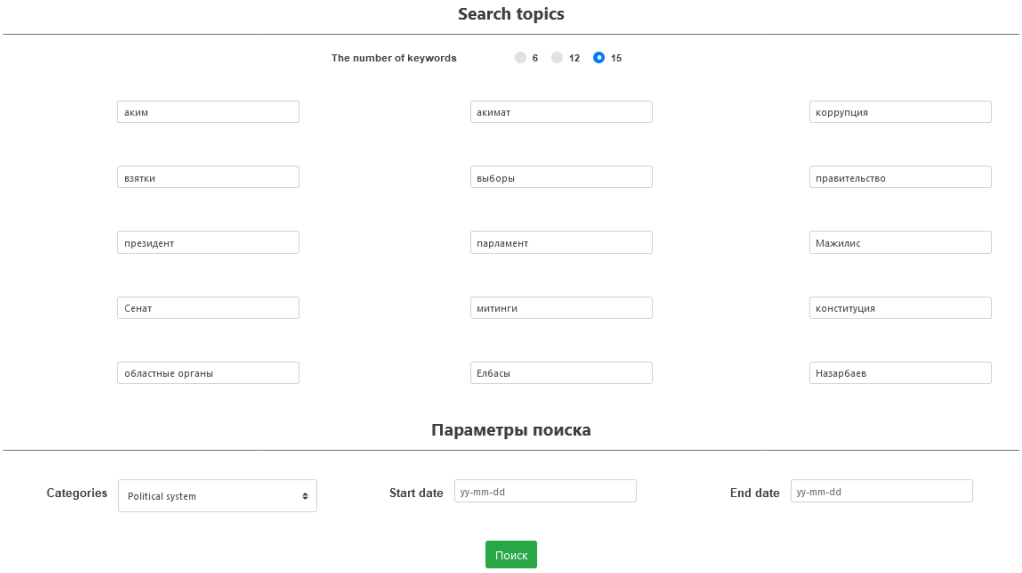

Figure 4.3 – Topic category selection form

In the form itself, it is possible to switch the number of themes to 6, 12, or 15. You can also change theme names in text fields and categories in the drop-down list. The default start and end dates are also defined in the program. The start date is January 1, 2018, and the end date is today, at the time of the application's launch.

After submitting the form, the database searches for the given topics and also calculates indicators of the level of interest in the topic in society, the level of activity of the topic discussion, and the level of social mood for each of the topics. The following formulas are used to calculate these indicators for the entire theme category (4.1, 4.2):

$$R_{CT\_AVG} = \frac{R_{CT\_SUM}}{len(R_{CT\_LIST})}, \quad (4.1)$$

where R_{CT_SUM} is the sum of the values of the interest level for each topic; $len(R_{CT\_LIST})$ is the length of the list of values.

$$R_{CE\_AVG} = \frac{R_{CE\_SUM}}{len(R_{CE\_LIST})}, \quad (4.2)$$

where R_{CE_SUM} is the sum of the values of the discussion activity level for each topic; $len(R_{CE\_LIST})$ is the length of the list of values.

The value of the level of the social mood of the category of topics is determined by the maximum number of positive, negative, neutral, or undefined topics (4.3):

$$R_{TS\_MAX} = Max < T\_pos, T\_neg, T\_neut, T\_non\_def >, \quad (4.3)$$

where T_pos is the number of positive topics; T_neg is the number of negative topics; T_neut is the number of neutral topics; T_non_def is the number of undefined topics.

The application also added the function of determining the general social mood by the selected category of topics. A conditional approach based on the values of the specified metrics shown in Table 4.1 is used to do it.

Table 4.1 – The general social mood of society

The sentiment of the topic category	The level of interest in the topic category	The level of activity of the topic category discussion	The general level of social mood
Positive	>= 0.0015	>= 200	Very good social mood
Positive	>= 0.0015	< 200	Good social mood
Positive	< 0.0015	>= 200	Good social mood
Positive	< 0.0015	< 200	Moderately good social mood
Neutral	>= 0.0015	>= 200	Stable social mood
Neutral	< 0.0015	>= 200	Satisfactory social mood
Neutral	>= 0.0015	< 200	Satisfactory social mood
Neutral	< 0.0015	< 200	Weak social mood
Negative	< 0.0015	< 200	Small social tensions
Negative	>= 0.0015	< 200	Average social tension
Negative	< 0.0015	>= 200	Average social tension
Negative	>= 0.0015	>= 200	High social tensions

If the sentiment is undefined, an uncertain situation of social mood is highlighted. The thresholds are approximate and may vary depending on the size of the database.

4.3 Software implementation of the application

The Chart.js library in JavaScript is used to render graphs. A bubble diagram of the number of found texts on a particular topic is shown in Figure 4.4. At the same time, the radius of each bubble in pixels grows with an increase in the number of found texts and comments on topics using keywords.

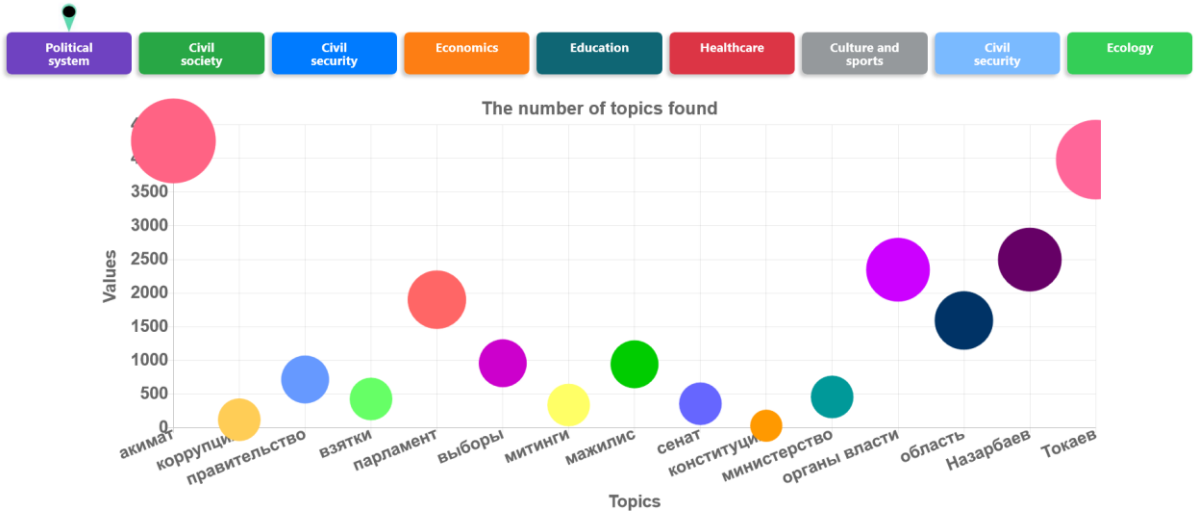

Figure 4.4 – The number of texts by topic

When you hover over each item, a pop-up window shows the topic's name and the number of texts found on it. Graphs of the level of interest in the topic and the level of activity of the topic discussion are shown in Figures 4.5 and 4.6.

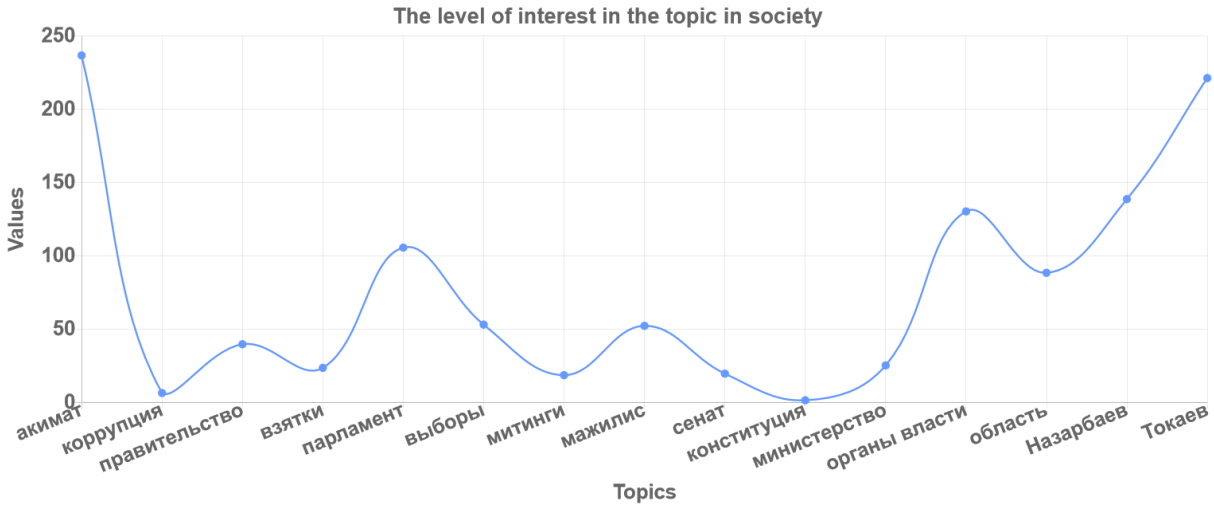

Figure 4.5 – The level of interest in the topic

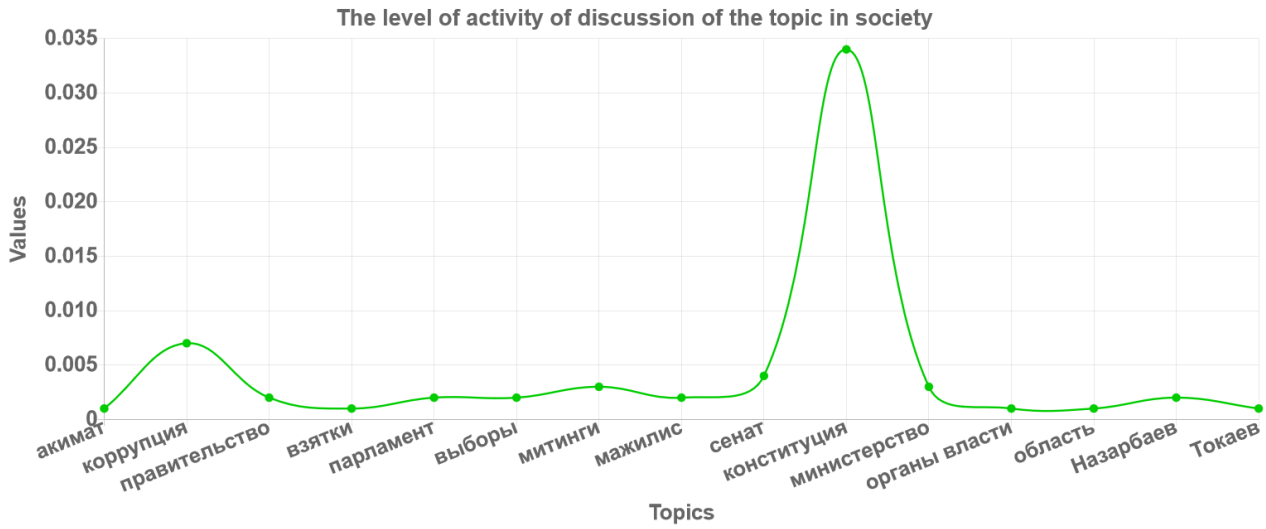

Figure 4.6 – The level of activity of the topic discussion

According to the schedules, it can be observed that the scale of interest level is proportional to the number of found texts on the topic, which is explained by the calculation formula R_{CT}, where the number of found texts is divided by the maximum number of texts or comments for a certain period set by the expert. At the discussion level, the scale is in a small range of values, which is also explained by the formula R_{CE}, where the total number of likes, comments, and reposts is divided by the total number of found texts and the number of subscribers to the data source.

The level of social mood for each topic is represented by the maximum number of certain text sentiments. The corresponding sentiment table is shown in Figure 4.7.

№	Topic	Sentiment
0	акимат	positive
1	коррупция	positive
2	правительство	positive
3	взятки	negative
4	парламент	positive
5	выборы	positive
6	митинги	positive
7	мажилис	positive
8	сенат	positive
9	конституция	positive
10	министерство	positive
11	органы власти	positive
12	область	positive
13	Назарбаев	positive
14	Токаев	positive

Figure 4.7 – The level of social mood by topic

The total social mood value for the selected topic category is displayed in the selection panel of all categories as a pop-up window (Figure 4.8).

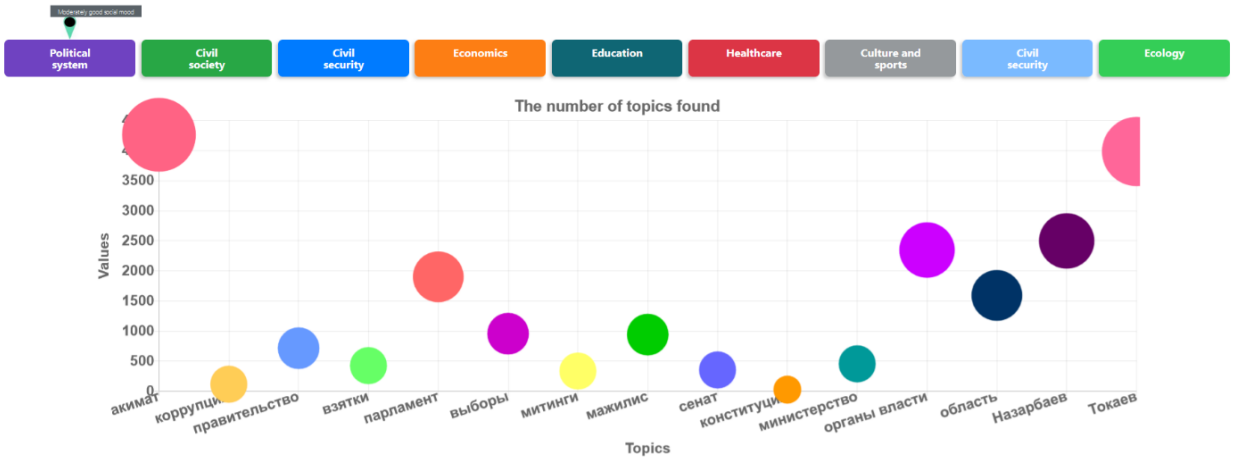

Figure 4.8 – General social mood by selected topic category

4.4 Conclusions on Chapter 4

In this chapter, we discussed the eSM module [114] for analyzing data developed on the Django Python framework. This system allows getting statistical information on the entered topic from the main OMSystem database. While the web crawler of the main platform searches for topics from the list of specified Internet sources, this module is aimed at searching for already obtained data, which significantly speeds up the construction of operational analysis. The function interface of the module allows you to quickly and conveniently change search topics and switch between the main categories of topics themselves. At the same time, a quick calculation of social analytics indicators is also carried out, which have already been discussed in detail in previous sections: the level of interest in the topic, the level of activity of the topic discussion, and the sentiment of the topic. In the future, it is also planned to expand the functionality of this web application.

CONCLUSION

In this dissertation work, an analysis of analytical platforms, design, architecture, and functionality of the developed OMSystem platform for analyzing the social media space of Kazakhstan was presented. In addition, a module for machine learning, neural networks, and marketing technologies has been developed to determine the mood of society in terms of socio-political content.

The following results of work and scientific novelty have been achieved:

1. The analysis of the architecture and functionality of the OMSystem social media analysis platform was carried out.

2. A module for processing and analyzing Big data of the OMSystem platform has been developed using special data processing tools (dictionaries, machine learning models, neural networks, and marketing technologies), which allows determining the social mood of the society.

3. The module's effectiveness was evaluated on a real example of the analysis of the population's reaction to the current government policy on vaccination against Covid-19.

4. An electronic Social Mood (eSM) module has been developed that analyzes data obtained using the OMSystem platform and evaluates the social mood of the society.

In general, this work's theoretical and practical results aim to strengthen the role of social analytics and machine and deep learning methods. In the future, it is planned to conduct further experiments in this area and increase the importance of social analytics as one of the important areas in natural language processing.

REFERENCES

1 Karyukin V., Mutanov G., Mamykova Z., et al. On the development of an information system for monitoring user opinion and its role for the public // Journal of Big Data. – 2022. – Vol. 9, № 110. – P. 1-45.

2 Esteban OO. The rise of social media. Our World in Data https://ourworldindata.org/rise-of-social-media/. 18.09.2019.

3 Chaffey D. Global social media statistics research summary 2022. Smart Insights https://www.smartinsights.com/social-media-marketing/social-media-strategy/new-global-social-media-research/. 01.06.2022.

4 Zhang H., Zang Zh., Zhu H., Uddin M.I., Amin M.A. Big data-assisted social media analytics for business model for business decision making system competitive analysis // Information Processing & Management. – 2022. – Vol. 59, №1. – P. 1–12.

5 Zhuang Y., et al. Intelligent Algorithm of Semantic Analysis Based on BP Neural Network // The 2021 International Conference on Smart Technologies and Systems for Internet of Things. – STSIoT, 2021. – P. 497-504.

6 Lekshmi S., Anoop V.S. Sentiment Analysis on COVID-19 News Videos Using Machine Learning Techniques // Proceedings of International Conference on Frontiers in Computing and Systems. – Springer, 2022. – P. 551-560.

7 Nguyen J., Armisen A., Agell N., Saz-Carranza A. Comparing global news sentiment using hesitant linguistic terms // International Journal of Intelligent Systems. – 2021. – Vol. 37, № 4. – P. 1-17.

8 Siva Rama Rao A.V.S., Kuchu S.S., Thota D., Chennam V.S., Yantrapragada H. Sentiment Analysis: Twitter Tweets Classification Using Machine Learning Approaches // Information and Communication Technology for Competitive Strategies. – Springer, 2021. – P. 411-419.

9 Patil R.S., Kolhe S.R. Supervised classifiers with TF-IDF features for sentiment analysis of Marathi tweets // Social Network Analysis and Mining. – 2022. – Vol. 12, № 51.

10 Thara S., Poornachandran P. Social media text analytics of Malayalam–English code-mixed using deep learning // Journal of Big Data. – 2022. – Vol. 9, № 45. – P. 1-25.

11 Sproutsocial. The most intuitive social media platform. https://sproutsocial.com/. 27.11.2021.

12 Hubspot. A CRM platform with all the software, integrations, and resources you need to connect marketing, sales, content management, and customer service. https://www.hubspot.com/. 27.11.2021.

13 Buzzsumo. See how much interest brands and content generate in the wild. https://buzzsumo.com/. 27.11.2021.

14 Hootsuite. Save time and get REAL results on social media. https://www.hootsuite.com/. 27.11.2021.

15 IQBuzz. Social media monitoring system. https://iqbuzz.pro/. 27.11.2021.

16 Brandmention. We dig every corner of the internet to find all the relevant mentions about anyone or anything. https://brandmentions.com/. 27.11.2021.

17 Snaplytics. Analyze & report social media results with a social listening tool. https://thehub.io/startups/snaplytics. 27.11.2021.

18 Radicioni T., Saracco F., Pavan E., Squartini T. Analysing Twitter semantic networks: the case of 2018 Italian elections // Scientific Reports. – 2021. – Vol. 11, № 13207. – P. 1-22.

19 Huq M.R., Ali A., Rahman A. Sentiment Analysis on Twitter Data using KNN and SVM // International Journal of Advanced Computer Science and Applications (IJACSA). – 2017. – Vol. 8, №6. – P. 1-7.

20 Znovarev A., Bilyi A. A comparison of machine learning methods of sentiment analysis based on Russian language Twitter data // 11th Majorov International Conference on Software Engineering and Computer Systems (MICSECS). – Saint Petersburg, Russian Federation, 2020. – P. 1-7.

21 Belcastro L., Branda F., Cantini R., et al. Analyzing voter behavior on social media during the 2020 US presidential election campaign // Social Network Analysis and Mining. – 2022. – Vol. 12, №83. – P. 1-16.

22 Negrete J.C.M., Iano Y., Negrete P.D.M., Vaz G.C., de Oliveira G.G. Sentiment and Emotions Analysis of Tweets During the Second Round of 2021 Ecuadorian Presidential Election // Proceedings of the 7th Brazilian Technology Symposium (BTSym'21). – Springer, 2021. – P. 257-268.

23 Oussous A., Boulouard Z., Zahra B.F. Prediction and Analysis of Moroccan Elections Using Sentiment Analysis // In AI and IoT for Sustainable Development in Emerging Countries. Lecture Notes on Data Engineering and Communications Technologies – Springer, Cham. – P. 597-609.

24 Tukeyev U., Karibayeva A., Zhumanov Zh. Morphological segmentation method for Turkic language neural machine translation // Cogent Engineering. – 2022. – Vol. 7, № 1. – P. 1-15.

25 Hamada M.A., Sultanbek K., Alzhanov B., Tokbanov B. Sentimental text processing tool for Russian language based on machine learning algorithms // Proceedings of the 5th International Conference on Engineering and MIS. – Astana, Kazakhstan, 2019. P. 1-6.

26 Yergesh B., Bekmanova G., Sharipbay A. Sentiment analysis of Kazakh text and their polarity // Web Intelligence. – 2019. – Vol. 17, №1 – P. 9-15.

27 Bekmanova G., Yelibayeva G., Aubakirova S., Dyussupova N., Sharipbay A., Niyazova N. Methods for analyzing polarity of the Kazakh texts related to the terrorist threats // 19th International Conference on Computational Science and Its Applications (ICCSA). – Saint Petersburg, Russian Federation, 2019. P. 717–730.

28 iMAS. Monitoring system for media, social networks, blogs, web resources. https://imas.kz/. 27.11.2021.

29 Alem Media Monitoring. Media and social networks monitoring system based on artificial intelligence. https://alem.kz/en/monitoring-smi/. 27.11.2021.

30 Mutanov G., Karyukin V., Mamykova Zh. Multi-class Sentiment Analysis of Social Media Data with Machine Learning Algorithms // CMC–Computers, Materials & Continua. – 2021. – Vol. 69, № 1. – P. 913–930.

31 Kadyrbek N., Sundetova Zh., Torekul S. Information Monitoring System of Social Wellness Opinions // IEEE 8th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE). – Vilnius, Lithuania, 2021. – P. 1-4.

32 Pamuksuz U., Yun J.T., Humphreys A. A Brand-New Look at You: Predicting Brand Personality in Social Media Networks with Machine Learning // Journal of Interactive Marketing. – 2021. – Vol. 56, № 1. – P. 55–69.

33 Chandra Sekhar Reddy N., Subhashini V., Rai D., Sriharsha Vittal B., Ganesh S. Product rating estimation using machine learning // 6th International Conference on Communication and Electronics Systems (ICCES). – Coimbatre, India, 2021. – P. 1366–1369.

34 Dangi D., Bhagat A., Dixit D.K. Emerging Applications of Artificial Intelligence, Machine learning and Data Science // CMC-Computers, Materials & Continua. – 2022. – Vol. 70, № 3. – P. 5399–5419.

35 Akpatsa S.K., Lei H., Li X., Kofi Setornyo Obeng V.H. Evaluating Public Sentiments of Covid-19 Vaccine Tweets Using Machine Learning Techniques // Informatica. – 2022. – Vol. 46, № 1, – P. 69-75.

36 Mussiraliyeva Sh., Omarov B., Yoo P., Bolatbek M. Applying Machine Learning Techniques for Religious Extremism Detection on Online User Contents // CMC-Computers, Materials & Continua. – 2021. – Vol. 70, № 1. – P. 915-934.

37 Röchert D., Neubaum G., Stieglitz S. Identifying Political Sentiments on YouTube: A Systematic Comparison Regarding the Accuracy of Recurrent Neural Network and Machine Learning Models. // Disinformation in Open Online Media (MISDOOM). – Springer: Cham, 2020. – P. 107-121.

38 Dang N.C., Moreno-García M.N., De la Prieta F. Sentiment Analysis Based on Deep Learning: A Comparative Study // Electronics. – 2020. – Vol. 9, № 3. – P. 1-29.

39 Thara S., Poornachandran P. Social media text analytics of Malayalam–English code-mixed using deep learning // Journal of Big Data. – 2022. – Vol. 9, № 45. – P. 1-25.

40 Ombabi A.H., Ouarda W., Alimi A.M. Deep learning CNN–LSTM framework for Arabic sentiment analysis using textual information shared in social networks // Social Network Analysis and Mining. – 2020. – Vol. 10, №53. – P. 1-13.

41 Alzahrani H., Acharya S., Duverger P., Nguyen Nam P. Contextual polarity and influence mining in online social networks // Computational Social Networks. – 2021. – Vol. 8, № 21. – P. 1-27.

42 Karamouzas D., Mademlis I., Pitas I. Public opinion monitoring through collective semantic analysis of tweets // Social Network Analysis and Mining. – 2022. – Vol. 12, № 91. – P. 1-21.

43 Ali K., Hamilton M., Thevathayan C., et al. Big social data as a service (BSDaaS): a service composition framework for social media analysis // Journal of Big Data. – 2022. – Vol. 9, № 64. – P. 1-27.

44 Benedetto F., Tedeschi A. Big Data Sentiment Analysis for Brand Monitoring in Social Media Streams by Cloud Computing // Sentiment Analysis and

Ontology Engineering. Studies in Computational Intelligence. – Springer: Cham, 2016. – P. 341–377.

45 Schinas M., Papadopoulos S., Apostolidis L., Kompatsiaris Y., Mitkas P.A. Open-Source Monitoring, Search and Analytics Over Social Media // Internet Science. INSCI, 2017. – P. 361–369.

46 Pellert M., Metzler H., Matzenberger M., et al. Validating daily social media macroscopes of emotions // Scientific Reports. – 2022. – Vol. 12, № 11236. – P. 1-8.

47 Singh H., Yadav A., Bansal R., Mala S. Understanding Brand Authenticity Sentiments using Big Data Analytics // 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence). – Noida, India, 2021. – P. 304-308.

48 Camacho D., Luzón M.V., Cambria E. New research methods & algorithms in social network analysis // Future Generation Computer Systems. – 2021. – Vol. 114, №1. – P. 290–293.

49 Zarzour H., Al shboul B., Al-Ayyoub M., Jararweh Y. Sentiment Analysis Based on Deep Learning Methods for Explainable Recommendations with Reviews // 12th International Conference on Information and Communication Systems (ICICS). – Valencia, Spain, 2021. – P. 452–456.

50 Qureshi M.A., Asif M., Hassan M.F., Mustafa G., Ehsan M.K., Ali A., Sajid U. A Novel Auto-Annotation Technique for Aspect Level Sentiment Analysis // CMC-Computers, Materials & Continua. – 2022. – Vol. 70, № 3. – P. 4987–5004.

51 Deng Q., Hine M.J., Ji Sh., Wang Y. Understanding consumer engagement with brand posts on social media: The effects of post linguistic styles // Electronic Commerce Research and Applications. – 2021. – Vol. 48, № 101068. – P. 1-17.

52 Rahmatulloh A., Shofa R.N., Darmawan I., Ardiansah. Sentiment Analysis of Ojek Online User Satisfaction Based on the Naïve Bayes and Net Brand Reputation Method // 9th International Conference on Information and Communication Technology (ICoICT). – Yogyakarta, Indonesia, 2021. – P. 337–341.

53 Nandwani P., Verma R. A review on sentiment analysis and emotion detection from text // Social Network Analysis and Mining. – 2021. – Vol. 11, № 81. – P. 1-19.

54 Hartmann J., Huppertz J., Schamp C., Heitmann M. Comparing automated text classification methods // International Journal of Research in Marketing. – 2021. – vol. 36, №1. – P. 20–38.

55 Weber D., Nasim M., Mitchell L., Falzon L. Exploring the effect of streamed social media data variations on social network analysis // Social Network Analysis and Mining. – 2021. – Vol. 11, № 62. – P. 1-38.

56 Chaudhary K., Alam M., Al-Rakhami M.S., Gumaei A. Machine learning-based mathematical modelling for prediction of social media consumer behavior using big data analytics // Journal of Big Data. – 2021. – Vol. 8, №73. – P. 1-20.

57 Munnes S., Harsch C., Knobloch M., Vogel J.S., Hipp L., Schilling E. Examining Sentiment in Complex Texts. A Comparison of Different Computational Approaches // Frontiers in Big Data. – 2022. – Vol. 5, № 886362. – P. 1-16.

58 Jawale S., Sawarkar S.D. Sentiment Analysis and Vector Embedding: A Comparative Study // Smart Trends in Computing and Communications. – Springer, Singapore, 2023. – P. 311-321.

59 Bensalah N., Ayad H., Adib A., Farouk A.I.E. Combining Static and Contextual Features: The Case of English Tweets // Emerging Trends in Intelligent Systems & Network Security. – Springer, Cham, 2022. – P. 168-175.

60 Oscar Deho B, William Agangiba A., Felix Aryeh L., Jeffery Ansah A. Sentiment Analysis with Word Embedding // 7th International Conference on Adaptive Science & Technology (ICAST). – Accra, Ghana, 2018. – P. 1-4.

61 Setyanto A., Laksito A., Alarfaj F., Alreshoodi M., Kusrini Oyong I., Hayaty M., Alomair A., Almusallam N., Kurniasari L. Arabic Language Opinion Mining Based on Long Short-Term Memory (LSTM) // Applied Sciences. – 2022. Vol. 12, № 4140. – P. 1-18.

62 Balci S., Demirci G.M., Demirhan H., Sarp S. Sentiment Analysis Using State of the Art Machine Learning Techniques // Digital Interaction and Machine Intelligence. (MIDI). – Springer, Cham, 2022. – P. 34-42.

63 Tripathi J., Tiwari S., Saini A., Kumari S. Prediction of movie success based on machine learning and twitter sentiment analysis using internet movie database data // Indonesian Journal of Electrical Engineering and Computer Science. – 2023. – Vol. 29, №3. – P. 1-8.

64 Abdurrahim A., Lailis S., Lestandy M. Sentiment analysis of Covid-19 vaccine tweets utilizing Naïve Bayes // AIP Conference Proceedings. – 2022. – Vol. 2453, №1.

65 Hadwan M., Al-Sarem M., Saeed F., Al-Hagery M.A. An Improved Sentiment Classification Approach for Measuring User Satisfaction toward Governmental Services' Mobile Apps Using Machine Learning Methods with Feature Engineering and SMOTE Technique // Applied Sciences. – 2022. – Vol. 12, № 5547. – P. 1-25.

66 Sharma T., Diwakar M., Singh P., Lamba S., Kumar P., Joshi K. Emotion Analysis for predicting the emotion labels using Machine Learning approaches // IEEE 8th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON). – Dehradun, India, 2021. – P. 1-6.

67 Qiu Y., Song Z., Chen Z. Short-term stock trends prediction based on sentiment analysis and machine learning // Soft Computing. – 2022. – Vol. 26, №1. – P. 2209–2224.

68 Frankel R., Jennings J., Lee J. Disclosure Sentiment: Machine Learning vs. Dictionary Methods // Management Science. – 2021. – Vol. 68, №7. – P. 5514-5532.

69 Sood D., Kapoor N., Singh P. Voting Classification Approach for Sentiment Analysis of Twitter Data // International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES). – Greater Noida, India, 2022. – P. 307-313.

70 Alshuwaier F., Areshey A., Poon J. Applications and Enhancement of Document-Based Sentiment Analysis in Deep learning Methods: Systematic Literature

Review // Intelligent Systems with Applications. – 2022. – Vol. 15, № 200090. – P. 1-25.

71 Das S., Kolya A.K. Predicting the pandemic: sentiment evaluation and predictive analysis from large-scale tweets on Covid-19 by deep convolutional neural network // Evolutionary Intelligence. – 2022. – Vol. 15, №1. – P. 1913–1934.

72 Bharti P., Sagar V., Wadhwa B. An Analysis on Sentiments Using Deep Learning Approaches // International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES). – Greater Noida, India, 2022. – P. 355-360.

73 Kim K., Park S. AOBERT: All-modalities-in-One BERT for multimodal sentiment analysis // Information Fusion. – 2023. – Vol. 92, №1. – P. 37-45.

74 Engagement Rate: A Metric You Can Count On. https://www.socialbakers.com/blog/1427-engagement-rate-a-metric-you-can-count-on. 27.11.2021.

75 Praet S., Van Aelst P., Van Erkel P., Van der Veeken S., Martens D. Predictive modeling to study lifestyle politics with Facebook likes // EPJ Data Science. – 2021. – Vol. 10, № 50 – P. 1-25.

76 Beltrán J., Jara-Reyes R., Faure A. The Emotions of the Outbreak. Topics, Sentiments and Politics on Twitter During Chilean October // Communication and Smart Technologies. – Springer, Singapore, 2021. – P. 216–226.

77 Usero B., Hernández V., Quintana C. Social Media Mining for Business Intelligence Analytics: An Application for Movie Box Office Forecasting // Intelligent Computing. Lecture Notes in Networks and Systems. – Springer: Cham, 2022. – P. 981–999.

78 Bhatnagar S., Choubey N. Making sense of tweets using sentiment analysis on closely related topics // Social Network Analysis and Mining. – 2021. – Vol. 11, № 44. – P. 1-11.

79 Khalil E.A.H., Houby E.M.F.E., Mohamed H.K. Deep learning for emotion analysis in Arabic tweets // Journal of Big Data. – 2021. – Vol. 8, № 136. – P. 1-15.

80 Domalewska D. An analysis of COVID-19 economic measures and attitudes: evidence from social media mining // Journal of Big Data. – 2021. – Vol. 8, № 42. – P. 1-14.

81 Bhatnagar S., Choubey N. Making sense of tweets using sentiment analysis on closely related topics // Social Network Analysis and Mining. – 2021. – Vol. 11, № 44. – P. 1-11.

82 Ramya G.R., Bagavathi Sivakumar P. An incremental learning temporal influence model for identifying topical influencers on Twitter dataset // Social Network Analysis and Mining. – 2021. – Vol. 11, № 27. – P. 1-16.

83 Heidari M., Shamsinejad P. Producing An Instagram Dataset For Persian Language Sentiment Analysis Using Crowdsourcing Method // 6th International Conference on Web Research (ICWR). – Tehran, Iran, 2020. – P. 284–287.

84 Ng L.H.X., Loke J.Y. Analyzing Public Opinion and Misinformation in a COVID-19 Telegram Group Chat. IEEE Internet Computing. – 2021. – Vol. 25, №2. – P. 84–91.

85 Kulchitskaya D.Y., Folts A.O. Between politics and show business: Public discourse on social media regarding ksenia sobchak, the only female candidate in the 2018 Russian presidential election // Monitoring Obshchestvennogo Mneniya: Ekonomicheskie i Sotsial'nye Peremeny. – 2020. – № 4. – P. 176–199.

86 Chen J., Chen Y., He Y., et al. A classified feature representation three-way decision model for sentiment analysis // Applied Intelligence. – 2022. – Vol. 52. – P. 7995–8007.

87 Buzea M.C., Stefan T.M., Traian R. Automatic Fake News Detection for Romanian Online News // Information. – 2022. – Vol. 13, № 3. – P. 1-13.

88 Kadam V.P., Khandale K.B., Mahender C.N. Text Analysis and Classification for Preprocessing Phase of Automatic Text Summarization Systems // Soft Computing and its Engineering Applications – Springer, Cham, 2021.

89 Manjunath T.N., Yogish D., Mahalakshmi S., Yogish H.K. Smart question answering system using vectorization approach and statistical scoring method // Materials Today. – 2021. – Vol. 80, № 3. – P. 3719-3725.

90 Didi Y., Ahlam W., Ali W. COVID-19 Tweets Classification Based on a Hybrid Word Embedding Method // Big Data and Cognitive Computing. – 2022. – Vol. 6, № 2. – P. 1-20.

91 Aldawod A., Alsakran R., Alrasheed H. Understanding Entertainment Trends during COVID-19 in Saudi Arabia // Information. – 2022. – Vol. 13, № 308. – P. 1-11.

92 Vigneshwaran P., Prasath N., Sindhuja M., Islabudeen M.M., Ragaventhiran J., Muthu K.B. A comprehensive analysis of consumer decisions on Twitter dataset using machine learning algorithms // International Journal of Artificial Intelligence. – 2022. – Vol. 11, № 3. – P. 1-9.

93 Hassan F., Hicham El M., Hicham L., Ali Y. Sentiment Analysis of Arabic Comments Using Machine Learning and Deep Learning Models // Indian Journal of Computer Science and Engineering. – 2022. – Vol. 13, № 3, 2022. – P. 1-9.

94 Mousa G.A., Elamir E.A.H, Hussainey K. Using machine learning methods to predict financial performance: Does disclosure tone matter? // International Journal of Disclosure and Governance. – 2022. – Vol. 19. – P. 93–112.

95 Jain P.K., Pamula R., Yekun E.A. A multi-label ensemble predicting model to service recommendation from social media contents // The Journal of Supercomputing. – 2022. – Vol. 78. – P. 5203–5220.

96 Aljabri M., Aljameel S.S., Khan I.U., Aslam N., Charouf S.M.B., Alzahrani N. Machine Learning Model for Sentiment Analysis of COVID-19 Tweets // International Journal on Advanced Science, Engineering and Information Technology. – 2022. – Vol. 12, № 3. – P. 1206-1214.

97 Yeasmin N., Mahbub N.I., Baowaly M.K., Singh B.C., Alom Z., Aung Z., Azim M.A. Analysis and Prediction of User Sentiment on COVID-19 Pandemic Using Tweets // Big Data and Cognitive Computing. – 2022. – Vol. 6, № 2. – P. 1-15.

98 Daradkeh M. Analyzing Sentiments and Diffusion Characteristics of COVID-19 Vaccine Misinformation Topics in Social Media: A Data Analytics

Framework // International Journal of Business Analytics. – 2022. – Vol. 9, № 3. – P. 1-22.

99 Mishra S., Verma A., Meena K.. et al. Public reactions towards Covid-19 vaccination through twitter before and after second wave in India // Social Network Analysis and Mining. – 2022. – Vol. 12, № 57. – P. 1-16.

100 Iwendi C., Mohan S., Khan S., Ibeke E., Ahmadian A., Ciano T. Covid-19 fake news sentiment analysis // Computers and Electrical Engineering. – 2022. – Vol. 101, № 107967. – P. 1-22.

101 Porreca A., Scozzari F., Di Nicola M. Using text mining and sentiment analysis to analyze YouTube Italian videos concerning vaccination // BMC Public Health. – 2020. – Vol. 20, № 259. – P. 1-9.

102 Karami A., Zhu M., Goldschmidt B., Boyajieff H.R., Najafabadi M.M. COVID-19 Vaccine and Social Media in the US: Exploring Emotions and Discussions on Twitter // Vaccines. – 2021 – Vol. 9, № 10. – P. 1-15.

103 Marcec R., Likic R. Using Twitter for sentiment analysis towards AstraZeneca/Oxford, Pfizer/BioNTech and Moderna COVID-19 vaccines // Postgraduate Medical Journal. – 2021. – Vol. 98, № 1161. – P. 544–550.

104 Sahraian M.A., Ghadiri F., Azimi A., Moghadasi A.N. Adverse events reported by Iranian patients with multiple sclerosis after the first dose of Sinopharm BBIBP-CorV // Vaccine. – 2021. – Vol. 39, №43. – P. 6347-6350.

105 Mutanov G., Mamykova Z., Karyukin V., Yessenzhanova S. The Approach to Building a Context-Dependent Sentiment Dictionary // Digital Transformation in Sustainable Value Chains and Innovative Infrastructures. Studies in Systems. Decision and Control – Springer, Cham, 2022. – P. 11-20. https://doi.org/10.1007/978-3-031-07067-9_1.

106 Мутанов Г.М., Мамыкова Ж.Д., Карюкин В.И., Жақсыкелді А.Ж. Разработка машинно-обучаемого алгоритма определения тональности пользовательского восприятия контента // Вестник КазНИТУ Серия Технические Науки – 2019. – Vol. 135, №5. – P. 479-486.

107 Alimzhanova L.M. Karyukin V.I. A classification model based on decision-making processes // Вестник КазНИТУ Серия Технические Науки. – 2020. – Vol. 138, №2. – P. 183-190.

108 Рахимова Д.Р., Тұрарбек А.Т., Карюкин В.И., Карибаева А.С., Тұрғанбаева А.О. Қазақ тіліне арналған заманауи машиналық аударма технологияларына шолу // Вестник КазНИТУ Серия Технические Науки. – 2020. – Vol. 141, №5. – P. 104-110.

109 Karibayeva A., Karyukin V.I., Turganbayeva A., Turarbek A. The translation quality problems of machine translation systems for the Kazakh language // Journal of Mathematics, Mechanics and Computer Science, Kazakhstan. – 2021. – Vol. 111, № 3. – P. 1-9.

110 Karyukin V., Zhumabekova A., Yessenzhanova S. Machine Learning And Neural Network Methodologies of Analyzing Social Media // Proceedings of the 6th International Conference on Engineering & MIS 2020 (ICEMIS'20). Association for Computing Machinery. – Almaty, 2020. – P. 1-7.

111 Rakhimova D., Karyukin V., Karibayeva A., Turarbek A., Turganbayeva A. The Development of the Light Post-editing Module for English-Kazakh Translation // The 7th International Conference on Engineering & MIS 2021 (ICEMIS'21). Association for Computing Machinery – Almaty, 2021. – P. 1–5.

112 Карюкин В., Есенжанова С. Построение контекстно-зависимого тонального словаря // Международная научная конференция студентов и молодых ученых «ФАРАБИ ӘЛЕМІ». – Алматы, 2020. – Р. 1.

113 Карюкин В. Подход к построению приложения eSM // Международная научная конференция студентов и молодых ученых «ФАРАБИ ӘЛЕМІ». – Алматы, 2020. – Р. 1

114 Карюкин В. Многоклассовая классификация с применением алгоритмов машинного обучения // Международная научная конференция студентов и молодых ученых «ФАРАБИ ӘЛЕМІ», Алматы, 2021. – Р. 1.

APPENDIX A

Author’s certificate of the Republic of Kazakhstan

Author’s certificate of entering information into the state register of rights to objects protected by copyright No. 32914 dated February 22, 2023 “Electronic Social Mood (eSM) social mood analysis module”.

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ

РЕСПУБЛИКА КАЗАХСТАН

СВИДЕТЕЛЬСТВО

О ВНЕСЕНИИ СВЕДЕНИЙ В ГОСУДАРСТВЕННЫЙ РЕЕСТР ПРАВ НА ОБЪЕКТЫ, ОХРАНЯЕМЫЕ АВТОРСКИМ ПРАВОМ

№ 32914 от «22» февраля 2023 года

Фамилия, имя, отчество, (если оно указано в документе, удостоверяющем личность) автора (ов):
КАРЮКИН ВЛАДИСЛАВ ИГОРЕВИЧ

Вид объекта авторского права: **программа для ЭВМ**

Название объекта: **Модуль анализа социального настроения общества electronic Social Mood (eSM)**

Дата создания объекта: **28.02.2020**

Құжат түпнұсқалығын http://www.kazpatent.kz/ru сайтының "Авторлық құқық" бөлімінде тексеруге болады. https://copyright.kazpatent.kz

Подлинность документа возможно проверить на сайте kazpatent.kz в разделе «Авторское право» https://copyright.kazpatent.kz

Подписано ЭЦП

Е. Оспанов

APPENDIX B

Text preprocessing and classification program code

1. Text preprocessing

```
import nltk
import pandas as pd
import re
import Kazakh_Stemmer as kz_stemmer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem.snowball import SnowballStemmer
from nltk.stem.porter import PorterStemmer

topics_all = pd.read_csv('Texts_all_with_lan.csv')
topics_rus = pd.read_csv("Texts_russian_all.csv")
topics_kaz = pd.read_csv("Texts_kazakh_all.csv")

class Topics_processing:
    def __init__(self):
      self.kazakh_cleaned_text = self.kaz_text_process()
      self.russian_cleaned_text = self.rus_text_process()

    def rus_text_process(self):
      data = pd.read_csv("Texts_russian_all.csv")
      data['cleaned_text'] = ''
      tokens = []
      tokens2 =[]
      data['text'] = data['text'].astype(str)
      data['text'] = data['text'].apply(lambda x: x.lower())
      stop_words_russian = set(stopwords.words('russian'))
      stop_words_english = set(stopwords.words('english'))
      words_to_remove = []
      for i in stop_words_russian:
         if i=='не' or i=='ни':
            words_to_remove.append(i)
      for i in words_to_remove:
         stop_words_russian.remove(i)
      russian_stemmer = SnowballStemmer('russian')
      english_stemmer = PorterStemmer()

      for index, row in data.iterrows():
         row['text'] = re.sub('[^a-zA-Za-яА-ЯӘІҢҒҮҰҚӨәіңғүұқөh]',' ',row['text'])
         tokens = word_tokenize(row['text'])
         tokens2 = [t for t in tokens if t not in (stop_words_russian and stop_words_english)]
         tokens2 = [russian_stemmer.stem(t) for t in tokens2]
         tokens2 = [english_stemmer.stem(t) for t in tokens2]
         data.at[index, 'cleaned_text'] = ' '.join(tokens2)
      data = data.iloc[:,1:]
```

```
        data.to_csv("Texts_russian_all.csv")

    def kaz_text_process(self):
        data = pd.read_csv("Texts_kazakh_all.csv")
        data['text'] = data['text'].astype(str)
        data['cleaned_text'] = ''
        for index, row in data.iterrows():
            stemmer = kz_stemmer.Kazakh_Stemmer(row['text'])
            data.at[index, 'cleaned_text'] = stemmer.text
        data = data.iloc[:, 1:]
        data.to_csv("Texts_kazakh_all.csv")

processing = Topics_processing()
```

2. Text classification

```
import re
import keras
import pickle
import nltk
import sklearn
import seaborn as sns
import numpy as np
import pandas as pd
import pickle
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from nltk import pos_tag, word_tokenize
from sklearn.decomposition import PCA
from sklearn.model_selection import StratifiedKFold
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.multiclass import OneVsRestClassifier
from sklearn.decomposition import TruncatedSVD
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_ma-
trix, classification_report
from sklearn.model_selection import train_test_split
from scipy.sparse import random as sparse_random
from imblearn.over_sampling import SMOTE
from keras.utils import np_utils
from collections import Counter
from nltk.stem.wordnet import WordNetLemmatizer
from nltk.corpus import words
from keras import backend as K
```

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Conv1D, MaxPooling1D
from keras.layers import Flatten
from keras.layers import Embedding
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.layers import Dense, Dropout, Activation
from keras.layers import LSTM
import matplotlib.pyplot as plt
from itertools import cycle
from sklearn.metrics import roc_curve, auc
from sklearn import preprocessing
from sklearn.preprocessing import label_binarize

topics = pd.read_csv("Topics_rus.csv")
topics = topics.dropna()
print(topics.shape)

topics = topics[topics['id.1'] != 4.0]
print(topics.shape)

topics['tonal_name'] = ''
topics.loc[topics['name'] == 'Нейтральная', 'tonal_name'] = 'Neutral'
topics.loc[topics['name'] == 'Положительная', 'tonal_name'] = 'Positive'
topics.loc[topics['name'] == 'Отрицательная', 'tonal_name'] = 'Negative'

topics = topics.drop(['Unnamed: 0', 'id', 'lan', 'name'], axis=1)
topics['tonal_id'] = ''

topics.loc[topics['tonal_name'] == 'Neutral', 'tonal_id'] = 2
topics.loc[topics['tonal_name'] == 'Positive', 'tonal_id'] = 1
topics.loc[topics['tonal_name'] == 'Negative', 'tonal_id'] = 0

topics.tonal_name.value_counts()
tonal_labels = [0, 1, 2]

get_ipython().run_line_magic('matplotlib', 'inline')

plt.figure(figsize=[8,8])
topics.tonal_name.value_counts().plot(kind='bar', width=0.9, color=['green', 'red', 'blue'])
plt.xlabel('Class',fontsize=14)
plt.ylabel('Count',fontsize=14)
plt.title('Distribution',fontsize=14)
plt.xticks(rotation='horizontal', fontsize=14)
plt.yticks(fontsize=14)
plt.show()

classifiers = [LogisticRegression(max_iter=1000), SVC(kernel='linear'),  KNeighborsClassi-
fier(n_neighbors=5),
DecisionTreeClassifier(criterion='gini', splitter='best', max_depth=None),
```

```
RandomForestClassifier(n_estimators = 10),
GradientBoostingClassifier(random_state=42), MultinomialNB(alpha=1.0, fit_prior=True,
class_prior=None)]

topics["tonal_id"] = pd.to_numeric(topics["tonal_id"])

x = topics['cleaned_text']
y = topics['tonal_id']

algorithms = ['Logistic Regression', 'Support vector machine', 'K-nearest neighbors', 'Decision tree',
'Random Forest', 'Gradient boosting', 'Naive Bayes']

vectorizer = TfidfVectorizer()
x_tfidf = vectorizer.fit_transform(x)

pickle.dump(vectorizer, open('Tf_idf_rus.pickle', "wb+"))

svd = TruncatedSVD(n_components=10, random_state=42)

x_tfidf_truncated = svd.fit_transform(x_tfidf)
y_bin = label_binarize(y, classes=[0, 1, 2])
tonal_names = ['negative', 'positive', 'neutral']

x_tfidf_train, x_tfidf_test, y_train, y_test = train_test_split(x_tfidf, y, test_size=0.3, ran-
dom_state=42)
x_tfidf_train, x_tfidf_test, y_train_bin, y_test_bin = train_test_split(x_tfidf, y_bin, test_size=0.3,
random_state=42)
x_tfidf_truncated_train, x_tfidf_truncated_test, y_train, y_test = train_test_split(x_tfidf_truncated,
y, test_size=0.3, random_state=42)
x_tfidf_truncated_train, x_tfidf_truncated_test, y_train_bin, y_test_bin =
train_test_split(x_tfidf_truncated, y_bin, test_size=0.3, random_state=42)

print(x_tfidf_train.shape)
print(x_tfidf_test.shape)
print(x_tfidf_truncated_train.shape)
print(x_tfidf_truncated_test.shape)

n_classes = 3
k = 0
for i in classifiers:
    if algorithms[k] in ['Support vector machine', 'Gradient boosting']:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_truncated_train, y_train_bin)
        y_pred_bin = classifier.predict(x_tfidf_truncated_test)
    else:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_train, y_train_bin)
        y_pred_bin = classifier.predict(x_tfidf_test)

    fpr = dict()
    tpr = dict()
```

```
    roc_auc = dict()
    for i in range(n_classes):
        fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_pred_bin[:, i])
        roc_auc[i] = auc(fpr[i], tpr[i])

    fpr["micro"], tpr["micro"], _ = roc_curve(y_test_bin.ravel(), y_pred_bin.ravel())
    roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

    all_fpr = np.unique(np.concatenate([fpr[i] for i in range(n_classes)]))
    mean_tpr = np.zeros_like(all_fpr)
    for i in range(n_classes):
        mean_tpr += np.interp(all_fpr, fpr[i], tpr[i])

    mean_tpr /= n_classes

    fpr["macro"] = all_fpr
    tpr["macro"] = mean_tpr
    roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])

    plt.rcParams['font.size'] = 16
    fig, axs = plt.subplots(figsize=(15, 10))
    axs.set_title('Receiver operating characteristic to multiclass classification for '+algorithms[k],
fontsize = 16)
    axs.set_xlabel("False Positive Rate")
    axs.set_ylabel("True Positive Rate")
    colors = cycle(['#96ff33',  '#54fdbd', '#f9a727', '#f1fc0e', '#b40efc'])

    axs.plot(fpr["micro"], tpr["micro"],
    label="Micro-average ROC curve with area = {0:0.2f}".format(roc_auc["micro"]),
    color="#fc200e",
    linestyle=":",
    linewidth=4,)

    axs.plot(fpr["macro"], tpr["macro"],
    label="Macro-average ROC curve with area = {0:0.2f})".format(roc_auc["macro"]),
    color="navy",
    linestyle=":",
    linewidth=4,)

    for i, color in zip(range(n_classes), colors):
        axs.plot(fpr[i], tpr[i],
        color=color,
        label="ROC curve of class {0} with area = {1:0.2f}".format(tonal_names[i], roc_auc[i]),)

    axs.legend(loc="lower right")

    plt.show()
    k = k + 1

k = 0
for i in classifiers:
```

```
    if algorithms[k] in ['Support vector machine', 'Gradient boosting']:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_truncated_train, y_train)
        y_pred = classifier.predict(x_tfidf_truncated_test)
    else:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_train, y_train)
        y_pred = classifier.predict(x_tfidf_test)

    accuracy = accuracy_score(y_test, y_pred)
    precision_macro = precision_score(y_test, y_pred, average='macro')
    recall_macro = recall_score(y_test, y_pred, average='macro')
    f1_macro = f1_score(y_test, y_pred, average='macro')

    precision_micro = precision_score(y_test, y_pred, average='micro')
    recall_micro = recall_score(y_test, y_pred, average='micro')
    f1_micro = f1_score(y_test, y_pred, average='micro')

    precision_weighted = precision_score(y_test, y_pred, average='weighted')
    recall_weighted = recall_score(y_test, y_pred, average='weighted')
    f1_weighted = f1_score(y_test, y_pred, average='weighted')

    metrics_list = [('Accuracy', accuracy),('Macro-precision', precision_macro),('Micro-precision',
precision_micro),
                    ('Weighted-precision',precision_weighted),('Macro-recall', recall_macro),('Micro-re-
call', recall_micro),
                    ('Weighted-recall',recall_weighted), ('Macro F1-score', f1_macro), ('Micro F1-score',
f1_micro), ('Weighted F1-score', f1_weighted)]

    y_test_name = []
    y_pred_name = []
    for i in y_test:
        if i==2:
            y_test_name.append('neutral')
        elif i==1:
            y_test_name.append('positive')
        elif i==0:
            y_test_name.append('negative')

    for i in y_pred:
        if i==2:
            y_pred_name.append('neutral')
        elif i==1:
            y_pred_name.append('positive')
        elif i==0:
            y_pred_name.append('negative')
        else:
            print(i)

    cf_matrix = confusion_matrix(y_test_name, y_pred_name, tonal_names)
```

```
    cl_report = classification_report(y_test, y_pred, target_names=tonal_names)
    plt.rcParams['font.size'] = 16
    fig, axes = plt.subplots(figsize=(15, 10))

    axes.set_title('Confusion matrix', fontsize = 16)
    sns.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%', cmap='Oranges')
    axes.set_xticklabels(tonal_names, fontsize = 16)
    axes.set_yticklabels(tonal_names, fontsize = 16)
    axes.set_xlabel('Predicted',fontsize = 16)
    axes.set_ylabel('True', fontsize = 16)

    print("Evaluation metrics of " + algorithms[k]+" algorithm: ")
    print('Accuracy: ', accuracy)
    print('Precision macro: ', precision_macro)
    print('Precision micro: ', precision_micro)
    print('Precision weighted: ', precision_weighted)
    print('Recall macro: ', recall_macro)
    print('Recall micro: ', recall_micro)
    print('Recall weighted: ', recall_weighted)
    print('F1-score macro: ', f1_macro)
    print('F1-score micro: ', f1_micro)
    print('F1-score weighted: ', f1_weighted)
    print('Classification report:')
    print(cl_report)
    plt.show()
    k = k + 1

from imblearn.over_sampling import RandomOverSampler
ros = RandomOverSampler(random_state=42)
x_tfidf_ros, y_ros = ros.fit_resample(x_tfidf, y)
x_tfidf_ros, y_ros_bin = ros.fit_resample(x_tfidf, y_bin)

svd = TruncatedSVD(n_components=10, n_iter=7, random_state=42)
x_tfidf_ros_truncated = svd.fit_transform(x_tfidf_ros)

print(x_tfidf_ros_truncated.shape)
print(x_tfidf_ros.shape)

x_tfidf_ros_train, x_tfidf_ros_test, y_ros_train, y_ros_test = train_test_split(x_tfidf_ros, y_ros,
test_size=0.3, random_state=42)
x_tfidf_ros_train, x_tfidf_ros_test, y_ros_train_bin, y_ros_test_bin = train_test_split(x_tfidf_ros,
y_ros_bin, test_size=0.3, random_state=42)
x_tfidf_ros_truncated_train, x_tfidf_ros_truncated_test, y_ros_train, y_ros_test =
train_test_split(x_tfidf_ros_truncated, y_ros, test_size=0.3, random_state=42)
x_tfidf_ros_truncated_train, x_tfidf_ros_truncated_test, y_ros_train_bin, y_ros_test_bin =
train_test_split(x_tfidf_ros_truncated, y_ros_bin, test_size=0.3, random_state=42)

classifiers[4].fit(x_tfidf_ros_train, y_ros_train)
pickle.dump(classifiers[4], open('RF_oversampled_rus.h5', 'wb+'))

k = 0
```

```
for i in classifiers:
    if algorithms[k] in ['Support vector machine', 'Gradient boosting']:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_ros_truncated_train, y_ros_train_bin)
        y_ros_pred_bin = classifier.predict(x_tfidf_ros_truncated_test)
    else:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_ros_train, y_ros_train_bin)
        y_ros_pred_bin = classifier.predict(x_tfidf_ros_test)

    fpr = dict()
    tpr = dict()
    roc_auc = dict()
    for i in range(n_classes):
        fpr[i], tpr[i], _ = roc_curve(y_ros_test_bin[:, i], y_ros_pred_bin[:, i])
        roc_auc[i] = auc(fpr[i], tpr[i])

    fpr["micro"], tpr["micro"], _ = roc_curve(y_ros_test_bin.ravel(), y_ros_pred_bin.ravel())
    roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

    all_fpr = np.unique(np.concatenate([fpr[i] for i in range(n_classes)]))
    mean_tpr = np.zeros_like(all_fpr)
    for i in range(n_classes):
        mean_tpr += np.interp(all_fpr, fpr[i], tpr[i])

    mean_tpr /= n_classes

    fpr["macro"] = all_fpr
    tpr["macro"] = mean_tpr
    roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])

    plt.rcParams['font.size'] = 16
    fig, axs = plt.subplots(figsize=(15, 10))
    axs.set_title('Receiver operating characteristic to multiclass classification for '+algorithms[k],
fontsize = 16)
    axs.set_xlabel("False Positive Rate")
    axs.set_ylabel("True Positive Rate")
    colors = cycle(['#96ff33',  '#54fdbd', '#f9a727', '#f1fc0e', '#b40efc'])

    axs.plot(fpr["micro"], tpr["micro"],
    label="Micro-average ROC curve with area = {0:0.2f}".format(roc_auc["micro"]),
    color="#fc200e",
    linestyle=":",
    linewidth=4,)

    axs.plot(fpr["macro"], tpr["macro"],
    label="Macro-average ROC curve with area = {0:0.2f})".format(roc_auc["macro"]),
    color="navy",
    linestyle=":",
    linewidth=4,)
```

```
    for i, color in zip(range(n_classes), colors):
        axs.plot(fpr[i], tpr[i],
        color=color,
        label="ROC curve of class {0} with area = {1:0.2f}".format(tonal_names[i], roc_auc[i]),)

    axs.legend(loc="lower right")

    plt.show()
    k = k + 1

k = 0
for i in classifiers:

    if algorithms[k] in ['Support vector machine', 'Gradient boosting']:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_ros_truncated_train, y_ros_train)
        y_ros_pred = classifier.predict(x_tfidf_ros_truncated_test)
    else:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_ros_train, y_ros_train)
        y_ros_pred = classifier.predict(x_tfidf_ros_test)

    accuracy = accuracy_score(y_ros_test, y_ros_pred)
    precision_macro = precision_score(y_ros_test, y_ros_pred, average='macro')
    recall_macro = recall_score(y_ros_test, y_ros_pred, average='macro')
    f1_macro = f1_score(y_ros_test, y_ros_pred, average='macro')

    precision_micro = precision_score(y_ros_test, y_ros_pred, average='micro')
    recall_micro = recall_score(y_ros_test, y_ros_pred, average='micro')
    f1_micro = f1_score(y_ros_test, y_ros_pred, average='micro')

    precision_weighted = precision_score(y_ros_test, y_ros_pred, average='weighted')
    recall_weighted = recall_score(y_ros_test, y_ros_pred, average='weighted')
    f1_weighted = f1_score(y_ros_test, y_ros_pred, average='weighted')

    metrics_list = [('Accuracy', accuracy),('Macro-precision', precision_macro),('Micro-precision',
precision_micro),
                ('Weighted-precision',precision_weighted),('Macro-recall', recall_macro),('Micro-re-
call', recall_micro),
                ('Weighted-recall',recall_weighted), ('Macro F1-score', f1_macro), ('Micro F1-score',
f1_micro), ('Weighted F1-score', f1_weighted)]

    y_ros_test_name = []
    y_ros_pred_name = []
    for i in y_ros_test:
        if i==2:
            y_ros_test_name.append('neutral')
        elif i==1:
            y_ros_test_name.append('positive')
        elif i==0:
            y_ros_test_name.append('negative')
```

```
    for i in y_ros_pred:
        if i==2:
            y_ros_pred_name.append('neutral')
        elif i==1:
            y_ros_pred_name.append('positive')
        elif i==0:
            y_ros_pred_name.append('negative')
        else:
            print(i)

    cf_matrix = confusion_matrix(y_ros_test_name, y_ros_pred_name, tonal_names)
    cl_report = classification_report(y_ros_test, y_ros_pred, target_names=tonal_names)

    plt.rcParams['font.size'] = 16
    fig, axes = plt.subplots(figsize=(15, 10))
    axes.set_title('Confusion matrix', fontsize = 16)
    sns.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%', cmap='Oranges')
    axes.set_xticklabels(tonal_names, fontsize = 16)
    axes.set_yticklabels(tonal_names, fontsize = 16)
    axes.set_xlabel('Predicted',fontsize = 16)
    axes.set_ylabel('True', fontsize = 16)

    print("Evaluation metrics of " + algorithms[k]+" algorithm: ")
    print('Accuracy: ', accuracy)
    print('Precision macro: ', precision_macro)
    print('Precision micro: ', precision_micro)
    print('Precision weighted: ', precision_weighted)
    print('Recall macro: ', recall_macro)
    print('Recall micro: ', recall_micro)
    print('Recall weighted: ', recall_weighted)
    print('F1-score macro: ', f1_macro)
    print('F1-score micro: ', f1_micro)
    print('F1-score weighted: ', f1_weighted)
    print('Classification report:')
    print(cl_report)
    plt.show()
    k = k + 1

from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=42)

x_tfidf_sm, y_sm = sm.fit_resample(x_tfidf, y)
x_tfidf_sm, y_sm_bin = sm.fit_resample(x_tfidf, y_bin)
x_tfidf_sm_truncated = svd.fit_transform(x_tfidf_sm)

print(x_tfidf_sm.shape)
print(y_sm.shape)

x_tfidf_sm_train, x_tfidf_sm_test, y_sm_train, y_sm_test = train_test_split(x_tfidf_sm, y_sm,
test_size=0.3, random_state=42)
```

```
x_tfidf_sm_train, x_tfidf_sm_test, y_sm_train_bin, y_sm_test_bin = train_test_split(x_tfidf_sm, y_sm_bin, test_size=0.3, random_state=42)
x_tfidf_sm_truncated_train, x_tfidf_sm_truncated_test, y_sm_train, y_sm_test = train_test_split(x_tfidf_sm_truncated, y_sm, test_size=0.3, random_state=42)
x_tfidf_sm_truncated_train, x_tfidf_sm_truncated_test, y_sm_train_bin, y_sm_test_bin = train_test_split(x_tfidf_sm_truncated, y_sm_bin, test_size=0.3, random_state=42)

k = 0
for i in classifiers:
    if algorithms[k] in ['Support vector machine', 'Gradient boosting']:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_sm_truncated_train, y_sm_train_bin)
        y_sm_pred_bin = classifier.predict(x_tfidf_sm_truncated_test)
    else:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_sm_train, y_sm_train_bin)
        y_sm_pred_bin = classifier.predict(x_tfidf_sm_test)

    fpr = dict()
    tpr = dict()
    roc_auc = dict()
    for i in range(n_classes):
        fpr[i], tpr[i], _ = roc_curve(y_sm_test_bin[:, i], y_sm_pred_bin[:, i])
        roc_auc[i] = auc(fpr[i], tpr[i])

    fpr["micro"], tpr["micro"], _ = roc_curve(y_sm_test_bin.ravel(), y_sm_pred_bin.ravel())
    roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

    all_fpr = np.unique(np.concatenate([fpr[i] for i in range(n_classes)]))
    mean_tpr = np.zeros_like(all_fpr)
    for i in range(n_classes):
        mean_tpr += np.interp(all_fpr, fpr[i], tpr[i])

    mean_tpr /= n_classes

    fpr["macro"] = all_fpr
    tpr["macro"] = mean_tpr
    roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])

    plt.rcParams['font.size'] = 16
    fig, axs = plt.subplots(figsize=(15, 10))
    axs.set_title('Receiver operating characteristic to multiclass classification for '+algorithms[k], fontsize = 16)
    axs.set_xlabel("False Positive Rate")
    axs.set_ylabel("True Positive Rate")
    colors = cycle(['#96ff33',  '#54fdbd', '#f9a727', '#f1fc0e', '#b40efc'])

    axs.plot(fpr["micro"], tpr["micro"],
    label="Micro-average ROC curve with area = {0:0.2f}".format(roc_auc["micro"]),
    color="#fc200e",
    linestyle=":",
```

```
    linewidth=4,)
    axs.plot(fpr["macro"], tpr["macro"],
    label="Macro-average ROC curve with area = {0:0.2f})".format(roc_auc["macro"]),
    color="navy",
    linestyle=":",
    linewidth=4,)

    for i, color in zip(range(n_classes), colors):
        axs.plot(fpr[i], tpr[i],
        color=color,
        label="ROC curve of class {0} with area = {1:0.2f}".format(tonal_names[i], roc_auc[i]),)

    axs.legend(loc="lower right")

    plt.show()
    k = k + 1

k = 0
for i in classifiers:

    if algorithms[k] in ['Support vector machine', 'Gradient boosting']:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_sm_truncated_train, y_sm_train)
        y_sm_pred = classifier.predict(x_tfidf_sm_truncated_test)
    else:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_sm_train, y_sm_train)
        y_sm_pred = classifier.predict(x_tfidf_sm_test)

    accuracy = accuracy_score(y_sm_test, y_sm_pred)
    precision_macro = precision_score(y_sm_test, y_sm_pred, average='macro')
    recall_macro = recall_score(y_sm_test, y_sm_pred, average='macro')
    f1_macro = f1_score(y_sm_test, y_sm_pred, average='macro')

    precision_micro = precision_score(y_sm_test, y_sm_pred, average='micro')
    recall_micro = recall_score(y_sm_test, y_sm_pred, average='micro')
    f1_micro = f1_score(y_sm_test, y_sm_pred, average='micro')

    precision_weighted = precision_score(y_sm_test, y_sm_pred, average='weighted')
    recall_weighted = recall_score(y_sm_test, y_sm_pred, average='weighted')
    f1_weighted = f1_score(y_sm_test, y_sm_pred, average='weighted')

    metrics_list = [('Accuracy', accuracy),('Macro-precision', precision_macro),('Micro-precision',
precision_micro),
                ('Weighted-precision',precision_weighted),('Macro-recall', recall_macro),('Micro-re-
call', recall_micro),
                ('Weighted-recall',recall_weighted), ('Macro F1-score', f1_macro), ('Micro F1-score',
f1_micro), ('Weighted F1-score', f1_weighted)]

    y_sm_test_name = []
    y_sm_pred_name = []
```

```
    for i in y_sm_test:
        if i==2:
            y_sm_test_name.append('neutral')
        elif i==1:
            y_sm_test_name.append('positive')
        elif i==0:
            y_sm_test_name.append('negative')

    for i in y_sm_pred:
        if i==2:
            y_sm_pred_name.append('neutral')
        elif i==1:
            y_sm_pred_name.append('positive')
        elif i==0:
            y_sm_pred_name.append('negative')
        else:
            print(i)

    cf_matrix = confusion_matrix(y_sm_test_name, y_sm_pred_name, tonal_names)
    cl_report = classification_report(y_sm_test, y_sm_pred, target_names=tonal_names)

    plt.rcParams['font.size'] = 16
    fig, axes = plt.subplots(figsize=(15, 10))
    axes.set_title('Confusion matrix', fontsize = 16)
    sns.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%', cmap='Oranges')
    axes.set_xticklabels(tonal_names, fontsize = 16)
    axes.set_yticklabels(tonal_names, fontsize = 16)
    axes.set_xlabel('Predicted',fontsize = 16)
    axes.set_ylabel('True', fontsize = 16)

    print("Evaluation metrics of " + algorithms[k]+" algorithm: ")
    print('Accuracy: ', accuracy)
    print('Precision macro: ', precision_macro)
    print('Precision micro: ', precision_micro)
    print('Precision weighted: ', precision_weighted)
    print('Recall macro: ', recall_macro)
    print('Recall micro: ', recall_micro)
    print('Recall weighted: ', recall_weighted)
    print('F1-score macro: ', f1_macro)
    print('F1-score micro: ', f1_micro)
    print('F1-score weighted: ', f1_weighted)
    print('Classification report:')
    print(cl_report)
    plt.show()
    k = k + 1

from imblearn.under_sampling import RandomUnderSampler
rus = RandomUnderSampler(random_state=0, replacement=True)

x_tfidf_rus, y_rus = rus.fit_resample(x_tfidf, y)
x_tfidf_rus, y_rus_bin = rus.fit_resample(x_tfidf, y_bin)
```

```
x_tfidf_rus_truncated = svd.fit_transform(x_tfidf_rus)

print(x_tfidf_rus.shape)
print(y_rus.shape)

x_tfidf_rus_train, x_tfidf_rus_test, y_rus_train, y_rus_test = train_test_split(x_tfidf_rus, y_rus,
test_size=0.3, random_state=42)
x_tfidf_rus_train, x_tfidf_rus_test, y_rus_train_bin, y_rus_test_bin = train_test_split(x_tfidf_rus,
y_rus_bin, test_size=0.3, random_state=42)
x_tfidf_rus_truncated_train, x_tfidf_rus_truncated_test, y_rus_train, y_rus_test =
train_test_split(x_tfidf_rus_truncated, y_rus, test_size=0.3, random_state=42)
x_tfidf_rus_truncated_train, x_tfidf_rus_truncated_test, y_rus_train_bin, y_rus_test_bin =
train_test_split(x_tfidf_rus_truncated, y_rus_bin, test_size=0.3, random_state=42)

k = 0
for i in classifiers:
    if algorithms[k] in ['Support vector machine', 'Gradient boosting']:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_rus_truncated_train, y_rus_train_bin)
        y_rus_pred_bin = classifier.predict(x_tfidf_rus_truncated_test)
    else:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_rus_train, y_rus_train_bin)
        y_rus_pred_bin = classifier.predict(x_tfidf_rus_test)

    fpr = dict()
    tpr = dict()
    roc_auc = dict()
    for i in range(n_classes):
        fpr[i], tpr[i], _ = roc_curve(y_rus_test_bin[:, i], y_rus_pred_bin[:, i])
        roc_auc[i] = auc(fpr[i], tpr[i])

    fpr["micro"], tpr["micro"], _ = roc_curve(y_rus_test_bin.ravel(), y_rus_pred_bin.ravel())
    roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

    all_fpr = np.unique(np.concatenate([fpr[i] for i in range(n_classes)]))
    mean_tpr = np.zeros_like(all_fpr)
    for i in range(n_classes):
        mean_tpr += np.interp(all_fpr, fpr[i], tpr[i])

    mean_tpr /= n_classes

    fpr["macro"] = all_fpr
    tpr["macro"] = mean_tpr
    roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])

    plt.rcParams['font.size'] = 16
    fig, axs = plt.subplots(figsize=(15, 10))
    axs.set_title('Receiver operating characteristic to multiclass classification for '+algorithms[k],
fontsize = 16)
    axs.set_xlabel("False Positive Rate")
```

```
    axs.set_ylabel("True Positive Rate")
    colors = cycle(['#96ff33',  '#54fdbd', '#f9a727', '#f1fc0e', '#b40efc'])

    axs.plot(fpr["micro"], tpr["micro"],
    label="Micro-average ROC curve with area = {0:0.2f}".format(roc_auc["micro"]),
    color="#fc200e",
    linestyle=":",
    linewidth=4,)

    axs.plot(fpr["macro"], tpr["macro"],
    label="Macro-average ROC curve with area = {0:0.2f})".format(roc_auc["macro"]),
    color="navy",
    linestyle=":",
    linewidth=4,)

    for i, color in zip(range(n_classes), colors):
        axs.plot(fpr[i], tpr[i],
        color=color,
        label="ROC curve of class {0} with area = {1:0.2f}".format(tonal_names[i], roc_auc[i]),)

    axs.legend(loc="lower right")
    plt.show()
    k = k + 1

k = 0
for i in classifiers:
    if algorithms[k] in ['Support vector machine', 'Gradient boosting']:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_rus_truncated_train, y_rus_train)
        y_rus_pred = classifier.predict(x_tfidf_rus_truncated_test)
    else:
        classifier = OneVsRestClassifier(i)
        classifier.fit(x_tfidf_rus_train, y_rus_train)
        y_rus_pred = classifier.predict(x_tfidf_rus_test)

    accuracy = accuracy_score(y_rus_test, y_rus_pred)
    precision_macro = precision_score(y_rus_test, y_rus_pred, average='macro')
    recall_macro = recall_score(y_rus_test, y_rus_pred, average='macro')
    f1_macro = f1_score(y_rus_test, y_rus_pred, average='macro')

    precision_micro = precision_score(y_rus_test, y_rus_pred, average='micro')
    recall_micro = recall_score(y_rus_test, y_rus_pred, average='micro')
    f1_micro = f1_score(y_rus_test, y_rus_pred, average='micro')

    precision_weighted = precision_score(y_rus_test, y_rus_pred, average='weighted')
    recall_weighted = recall_score(y_rus_test, y_rus_pred, average='weighted')
    f1_weighted = f1_score(y_rus_test, y_rus_pred, average='weighted')

    metrics_list = [('Accuracy', accuracy),('Macro-precision', precision_macro),('Micro-precision',
precision_micro),
                ('Weighted-precision',precision_weighted),('Macro-recall', recall_macro),('Micro-
```

```
recall', recall_micro),
                ('Weighted-recall',recall_weighted), ('Macro F1-score', f1_macro), ('Micro F1-score',
f1_micro), ('Weighted F1-score', f1_weighted)]

    y_rus_test_name = []
    y_rus_pred_name = []
    for i in y_rus_test:
        if i==2:
            y_rus_test_name.append('neutral')
        elif i==1:
            y_rus_test_name.append('positive')
        elif i==0:
            y_rus_test_name.append('negative')

    for i in y_rus_pred:
        if i==2:
            y_rus_pred_name.append('neutral')
        elif i==1:
            y_rus_pred_name.append('positive')
        elif i==0:
            y_rus_pred_name.append('negative')
        else:
            print(i)

    cf_matrix = confusion_matrix(y_rus_test_name, y_rus_pred_name, tonal_names)
    cl_report = classification_report(y_rus_test, y_rus_pred, target_names=tonal_names)

    plt.rcParams['font.size'] = 16
    fig, axes = plt.subplots(figsize=(15, 10))
    axes.set_title('Confusion matrix', fontsize = 16)
    sns.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%', cmap='Oranges')
    axes.set_xticklabels(tonal_names, fontsize = 16)
    axes.set_yticklabels(tonal_names, fontsize = 16)
    axes.set_xlabel('Predicted',fontsize = 16)
    axes.set_ylabel('True', fontsize = 16)

    print("Evaluation metrics of " + algorithms[k]+" algorithm: ")
    print('Accuracy: ', accuracy)
    print('Precision macro: ', precision_macro)
    print('Precision micro: ', precision_micro)
    print('Precision weighted: ', precision_weighted)
    print('Recall macro: ', recall_macro)
    print('Recall micro: ', recall_micro)
    print('Recall weighted: ', recall_weighted)
    print('F1-score macro: ', f1_macro)
    print('F1-score micro: ', f1_micro)
    print('F1-score weighted: ', f1_weighted)
    print('Classification report:')
    print(cl_report)
    plt.show()
    k = k + 1
```

APPENDIX C

Python Django eSM application

1. forms.py

```
from django import forms
from crispy_forms.helper import FormHelper
from crispy_forms.layout import Layout, Submit, Div, Fieldset, Row, Column
from crispy_forms.bootstrap import Tab, TabHolder, Field, Div, InlineCheckboxes, InlineRadios

class TagsForm(forms.Form):
    CATEGORIES = (
        ('politics', 'Политическая система'),
        ('society', 'Гражданское общество'),
        ('security', 'Гражданская безопасность'),
        ('economics', 'Экономика'),
        ('education', 'Образование'),
        ('health', 'Здравоохранение'),
        ('sport', 'Культура и спорт'),
        ('defence', 'Общественная безопасность'),
        ('ecology', 'Экология')
    )

    KEYWORDCOUNT = [('1','6'), ('2', '12'), ('3', '15')]

    tag1 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag2 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag3 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag4 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag5 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag6 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag7 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag8 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag9 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag10 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag11 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag12 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
```

```
    tag13 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag14 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))
    tag15 = forms.CharField(label='', required=False, widget=forms.TextInput(attrs={'placeholder':
'ключевое слово'}))

    choice_field = forms.ChoiceField(label = 'Количество ключевых слов', choices=KEYWORD-
COUNT, initial='3')
    categories = forms.ChoiceField(label = 'Категории', choices=CATEGORIES, initial='politics')
    dateStart = forms.CharField(label = 'Начальная дата', widget=forms.TextInput(attrs={'place-
holder': 'yy-mm-dd'}))
    dateEnd = forms.CharField(label = 'Конечная дата', widget=forms.TextInput(attrs={'placehold-
er': 'yy-mm-dd'}))

    def __init__(self, *args, **kwargs):
        super(TagsForm, self).__init__(*args, **kwargs)
        self.helper = FormHelper()
        self.helper.form_method = 'POST'
        self.helper.field_class = 'col-6'
        self.helper.form_class = 'form-horizontal'

        self.helper.layout = Layout(Fieldset("Темы поиска",
                Div(css_class="line"),
                InlineRadios('choice_field', id='radio_id'),
                Div(Field('tag1', css_class="form-control form-control-sm"),
                Field('tag2', css_class="form-control form-control-sm"),
                Field('tag3', css_class="form-control form-control-sm"), css_class='row-fluid',
css_id="tag_row_1"),
                Div(Field('tag4', css_class="form-control form-control-sm"),
                    Field('tag5', css_class="form-control form-control-sm"),
                    Field('tag6', css_class="form-control form-control-sm"), css_class='row-fluid',
css_id="tag_row_2"),
                Div(Field('tag7', css_class="form-control form-control-sm"),
                    Field('tag8', css_class="form-control form-control-sm"),
                    Field('tag9', css_class="form-control form-control-sm"), css_class='row-fluid',
css_id='tag_row_3'),
                Div(Field('tag10', css_class="form-control form-control-sm"),
                    Field('tag11', css_class="form-control form-control-sm"),
                    Field('tag12', css_class="form-control form-control-sm"), css_class='row-fluid',
css_id='tag_row_4'),
                 Div(Field('tag13', css_class="form-control form-control-sm"),
                     Field('tag14', css_class="form-control form-control-sm"),
                     Field('tag15', css_class="form-control form-control-sm"), css_class='row-fluid',
css_id='tag_row_5')),

                Fieldset("Параметры поиска", Div(css_class="line"),
                    Div(Field('categories', css_class="form-control form-control-sm"),
                        Field('dateStart', css_class="form-control form-control-sm"),
                        Field('dateEnd', css_class="form-control form-control-sm"), css_class='row-flu-
id')),
```

```
            Submit('submit', 'Поиск', css_class='btn-success'))

    class Meta:
        fields = ['tag1', 'tag2', 'tag3', 'tag4', 'tag5', 'tag6', 'tag7', 'tag8', 'tag9', 'tag10', 'tag11', 'tag12',
'tag13', 'tag14', 'tag15',
                  'choice_field', 'categories', 'dateStart', 'dateEnd']

    def clean(self):
        cleaned_data = self.cleaned_data
        tag1 = self.cleaned_data.get("tag1")
        tag2 = self.cleaned_data.get("tag2")
        tag3 = self.cleaned_data.get("tag3")
        tag4 = self.cleaned_data.get("tag4")
        tag5 = self.cleaned_data.get("tag5")
        tag6 = self.cleaned_data.get("tag6")
        tag7 = self.cleaned_data.get("tag7")
        tag8 = self.cleaned_data.get("tag8")
        tag9 = self.cleaned_data.get("tag9")
        tag10 = self.cleaned_data.get("tag10")
        tag11 = self.cleaned_data.get("tag11")
        tag12 = self.cleaned_data.get("tag12")
        tag13 = self.cleaned_data.get("tag13")
        tag14 = self.cleaned_data.get("tag14")
        tag15 = self.cleaned_data.get("tag15")
        categories = self.cleaned_data.get("categories")
        dateStart = self.cleaned_data.get("dateStart")
        dateEnd = self.cleaned_data.get("dateEnd")
        choice = self.cleaned_data.get("choice_field")
        print(dateStart)
        print(dateEnd)

        if choice == '1':
            if tag1 == "":
                self.add_error('tag1', 'Заполните данное поле')
            if tag2 == "":
                self.add_error('tag2', 'Заполните данное поле')
            if tag3 == "":
                self.add_error('tag3', 'Заполните данное поле')
            if tag4 == "":
                self.add_error('tag4', 'Заполните данное поле')
            if tag5 == "":
                self.add_error('tag5', 'Заполните данное поле')
            if tag6 == "":
                self.add_error('tag6', 'Заполните данное поле')

            if tag1 == "" or tag2 == "" or tag3 == "" or tag4 == "" or tag5 == "" or tag6 == "":
                raise forms.ValidationError("Заполнены не все поля формы")

        if choice == '2':
            if tag1 == "":
                self.add_error('tag1', 'Заполните данное поле')
```

```
            if tag2 == "":
                self.add_error('tag2', 'Заполните данное поле')
            if tag3 == "":
                self.add_error('tag3', 'Заполните данное поле')
            if tag4 == "":
                self.add_error('tag4', 'Заполните данное поле')
            if tag5 == "":
                self.add_error('tag5', 'Заполните данное поле')
            if tag6 == "":
                self.add_error('tag6', 'Заполните данное поле')
            if tag7 == "":
                self.add_error('tag7', 'Заполните данное поле')
            if tag8 == "":
                self.add_error('tag8', 'Заполните данное поле')
            if tag9 == "":
                self.add_error('tag9', 'Заполните данное поле')
            if tag10 == "":
                self.add_error('tag10', 'Заполните данное поле')
            if tag11 == "":
                self.add_error('tag11', 'Заполните данное поле')
            if tag12 == "":
                self.add_error('tag12', 'Заполните данное поле')
            if tag1 == "" or tag2 == "" or tag3 == "" or tag4 == "" or tag5 == "" or tag6 == "" or tag7 ==
"" or \
                    tag8 == "" or tag9 == "" or tag10 == "" or tag11 == "" or tag12 == "" :
                raise forms.ValidationError("Заполнены не все поля формы")

        if choice == '3':
            if tag1 == "":
                self.add_error('tag1', 'Заполните данное поле')
            if tag2 == "":
                self.add_error('tag2', 'Заполните данное поле')
            if tag3 == "":
                self.add_error('tag3', 'Заполните данное поле')
            if tag4 == "":
                self.add_error('tag4', 'Заполните данное поле')
            if tag5 == "":
                self.add_error('tag5', 'Заполните данное поле')
            if tag6 == "":
                self.add_error('tag6', 'Заполните данное поле')
            if tag7 == "":
                self.add_error('tag7', 'Заполните данное поле')
            if tag8 == "":
                self.add_error('tag8', 'Заполните данное поле')
            if tag9 == "":
                self.add_error('tag9', 'Заполните данное поле')
            if tag10 == "":
                self.add_error('tag10', 'Заполните данное поле')
            if tag11 == "":
                self.add_error('tag11', 'Заполните данное поле')
            if tag12 == "":
```

```
            self.add_error('tag12', 'Заполните данное поле')
        if tag13 == "":
            self.add_error('tag13', 'Заполните данное поле')
        if tag14 == "":
            self.add_error('tag14', 'Заполните данное поле')
        if tag15 == "":
            self.add_error('tag15', 'Заполните данное поле')

        if tag1 == "" or tag2 == "" or tag3 == "" or tag4 == "" or tag5 == "" or tag6 == "" or tag7 ==
"" \
                or tag8 == "" or tag9 == "" or tag10 == "" or tag11 == "" or tag12 == "" or tag13 == ""
or tag14 == "" or tag15 == "":
            raise forms.ValidationError("Заполнены не все поля формы")

    return self.cleaned_data
```

2. views.py

```
from django.shortcuts import render
from django.shortcuts import redirect
from django.http import HttpResponseRedirect
from django.db import connection
from django.http import JsonResponse
from .forms import TagsForm
from nltk.tokenize import word_tokenize
from nltk.stem.snowball import SnowballStemmer
from nltk.stem.porter import PorterStemmer
from datetime import datetime

category = ''
date_start = '2018-01-01'
date_end = datetime.today().strftime('%Y-%m-%d')
category_tonality = 0
tag_list_politics = ['акимат', 'коррупция', 'правительство', 'взятки', 'парламент', 'выборы',
'митинги',
                     'мажилис', 'сенат', 'конституция', 'министерство', 'органы власти', 'область',
'Назарбаев', 'Токаев']
tag_list_society = ['гражданское общество', 'социальная политика', 'государственные структу-
ры', 'конституция',
                    'общество', 'политика', 'институт власти', 'демократия', 'выборы', 'голосование',
'равенство',
                    'права', 'правовое государство', 'партии', 'избирательные органы']
tag_list_security = ['оборона', 'оружие', 'преступление', 'ограбление', 'убийство', 'война',
'терроризм', 'правосудие',
                     'безопасность', 'полицейские', 'террористы', 'угрозы', 'опасность', 'столкновения',
'беспорядки']
tag_list_economics = ['микроэкономика', 'макроэкономика', 'инфляция', 'капитал',
'девальвация', 'кризис', 'дефолт',
                      'цены', 'монополия', 'валюта', 'денежная масса', 'курс', 'ВВП', 'прибыль', 'налоги']
tag_list_education = ['школа', 'университет', 'высшее образование', 'министерство образования',
'преподавание',
```

```
                'занятия', 'школьники', 'студенты', 'школьные столовые', 'учителя',
'преподаватели', 'директор',
                'книги', 'обучение', 'канцелярские товары']
tag_list_health = ['поликлиника', 'больница', 'здоровье', 'коронавирус', 'медицина',
'заболевание', 'врач',
               'медицинская помощь', 'медицинские услуги', 'платные услуги', 'лечение',
'инфекция', 'вирус',
               'возбудители', 'COVID-19']
tag_list_sport = ['спортивные площадки', 'стадион', 'соревнования', 'секции', 'спортзал',
'здоровье', 'физкультура',
              'детский спорт', 'университетский спорт', 'бассейн', 'чемпионат',
              'футбол', 'хоккей', 'Лига Европы', 'Лига чемпионов']
tag_list_defence = ['многодетные матери', 'многодетные семьи', 'детские дома', 'пенсионные
отчисления',
               'социальная поддержка', 'социальная защита', 'социальная политика', 'социальные
институты',
               'рынок труда', 'права людей', 'благотворительность', 'льготы', 'пособия', 'зарплата',
'страхование']
tag_list_ecology = ['экология', 'биология', 'охрана природы', 'природопользование',
'заповедник', 'сад',
               'загрязнение', 'воздух', 'водоем', 'выброс', 'отход', 'вымирающие виды',
'популяции', 'животные',
               'растения']
tag_list_general = []
post_form = False

def index(request):
    global post_form
    if request.method == 'POST':
        form = TagsForm(request.POST)
        if form.is_valid():
            post_form = True
            del tag_list_general[:]
            category = ''
            if(form.cleaned_data['choice_field'] == '1'):
                tag_list_general.append(form.cleaned_data['tag1'])
                tag_list_general.append(form.cleaned_data['tag2'])
                tag_list_general.append(form.cleaned_data['tag3'])
                tag_list_general.append(form.cleaned_data['tag4'])
                tag_list_general.append(form.cleaned_data['tag5'])
                tag_list_general.append(form.cleaned_data['tag6'])
            elif(form.cleaned_data['choice_field'] == '2'):
                tag_list_general.append(form.cleaned_data['tag1'])
                tag_list_general.append(form.cleaned_data['tag2'])
                tag_list_general.append(form.cleaned_data['tag3'])
                tag_list_general.append(form.cleaned_data['tag4'])
                tag_list_general.append(form.cleaned_data['tag5'])
                tag_list_general.append(form.cleaned_data['tag6'])
                tag_list_general.append(form.cleaned_data['tag7'])
                tag_list_general.append(form.cleaned_data['tag8'])
                tag_list_general.append(form.cleaned_data['tag9'])
```

```
            tag_list_general.append(form.cleaned_data['tag10'])
            tag_list_general.append(form.cleaned_data['tag11'])
            tag_list_general.append(form.cleaned_data['tag12'])
        elif(form.cleaned_data['choice_field'] == '3'):
            tag_list_general.append(form.cleaned_data['tag1'])
            tag_list_general.append(form.cleaned_data['tag2'])
            tag_list_general.append(form.cleaned_data['tag3'])
            tag_list_general.append(form.cleaned_data['tag4'])
            tag_list_general.append(form.cleaned_data['tag5'])
            tag_list_general.append(form.cleaned_data['tag6'])
            tag_list_general.append(form.cleaned_data['tag7'])
            tag_list_general.append(form.cleaned_data['tag8'])
            tag_list_general.append(form.cleaned_data['tag9'])
            tag_list_general.append(form.cleaned_data['tag10'])
            tag_list_general.append(form.cleaned_data['tag11'])
            tag_list_general.append(form.cleaned_data['tag12'])
            tag_list_general.append(form.cleaned_data['tag13'])
            tag_list_general.append(form.cleaned_data['tag14'])
            tag_list_general.append(form.cleaned_data['tag15'])

        global date_start
        global date_end
        if form.cleaned_data['categories'] == 'politics':
            category = 'politics'
        elif form.cleaned_data['categories'] == 'society':
            category = 'society'
        elif form.cleaned_data['categories'] == 'security':
            category = 'security'
        elif form.cleaned_data['categories'] == 'economics':
            category = 'economics'
        elif form.cleaned_data['categories'] == 'education':
            category = 'education'
        elif form.cleaned_data['categories'] == 'health':
            category = 'health'
        elif form.cleaned_data['categories'] == 'sport':
            category = 'sport'
        elif form.cleaned_data['categories'] == 'defence':
            category = 'defence'
        elif form.cleaned_data['categories'] == 'ecology':
            category = 'ecology'

        date_start = form.cleaned_data['dateStart']
        date_end = form.cleaned_data['dateEnd']
        return redirect('graphics', category=category)

    else:
        form = TagsForm()

    return render(request, 'index.html', {'form': form})

def database_sql(tag_list):
```

```
    with connection.cursor() as cursor:
        Rct_list = []
        Rce_list = []
        Rts_list = []
        CT_list = []
        tag_data = []
        Rce_sum = 0
        Rct_sum = 0
        data = [[] for i in range(7)]
        sql_queries(cursor, Rct_list, Rce_list, Rts_list, CT_list, tag_data, tag_list)
        for i in Rce_list:
            Rce_sum += i
        for i in Rct_list:
            Rct_sum += i
        Rct_avg = round(Rct_sum / len(Rct_list), 3) #расчет среднего значения уровня заинтересованности темой в обществе
        Rce_avg = round(Rce_sum / len(Rce_list), 3) #расчет среднего значения уровня активности обсуждения темы в обществе
        pos_counter = 0
        neg_counter = 0
        neut_counter = 0
        non_def_counter = 0

        for i in Rts_list:
            if i == 'положительная':
                pos_counter += 1
            elif i == 'отрицательная':
                neg_counter += 1
            elif i == 'нейтральная':
                neut_counter += 1
            else:
                non_def_counter += 1

        Rts_max = max(pos_counter, neg_counter, neut_counter, non_def_counter) #поиск максимального значения тональности по категории тем
        if Rts_max == pos_counter:
            category_tonality = 'положительная'
        elif Rts_max == neg_counter:
            category_tonality = 'отрицательная'
        elif Rts_max == neut_counter:
            category_tonality = 'нейтральная'
        else:
            category_tonality = 'неопределенная'

        for tag, count in zip(tag_data, CT_list):
            data[0].append({'name': tag, 'quantity': count})
        for tag, count in zip(tag_list, Rct_list):
            data[1].append({'name': tag, 'count': count})
        for tag, count in zip(tag_list, Rce_list):
            data[2].append({'name': tag, 'count': count})
        for tag, count in zip(tag_list, Rts_list):
```

```
            data[3].append({'name': tag, 'tonality': count})

        data[4].append(category_tonality)
        data[5].append(Rce_avg)
        data[6].append(Rct_avg)
        print("The sentiment of a category = " + category_tonality)
        print("Rct_avg = " + str(Rct_avg))
        print("Rce_avg = " + str(Rce_avg))
        return data

def sql_queries(cursor, Rct_list, Rce_list, Rts_list, CT_list, tag_data, tag_list):
    for tag in tag_list:
        russian_stemmer = SnowballStemmer('russian')
        tag_tokens = word_tokenize(tag)

        whereClause = ''
        i = 1
        for t in tag_tokens:
            if i == 1:
                whereClause = " and (r.text LIKE '% " + t + "%' OR r.text LIKE '%" + t + " %' OR r.text
LIKE '% " + t + " %'"
            else:
                whereClause += " or r.text LIKE '% " + t + "%' or r.text LIKE '%" + t + " %' OR r.text
LIKE '% " + t + " %'"
            i += 1

        cursor.execute("select cast(count(*) as int) from results as r inner join resources as res "\
                    "on r.author = res.account where date between '" + date_start + "' and '" + date_end +
"' " + whereClause + ")")
        # поиск записей по теме в базе данных

        record = cursor.fetchall()
        if record[0][0] != 0:
            CT = int(record[0][0])
            maxCT = 1800  # максимальное предполагаемое количество текстов по заданной теме
            Rct = round((CT * 100) / maxCT, 3)  # уровень заинтересованности темой в обществе
            C, R, L, V = 0, 0, 0, 0
            CS, CP = 1, 1
            CT_list.append(CT)  # добавляем в первый список количество найденных текстов по
теме
            tag_data.append(tag)  # добавляем во второй список названия тем
            Rct_list.append(Rct)  # добавляем в список значение уровня заинтересованности по
теме

            CP = int(record[0][0])
            cursor.execute("SELECT sum(agg.comm_count), sum(agg.likes_count), sum(agg.re-
posts_count), "\
                "sum(agg.views_count), sum(agg.members) "\
                "FROM (SELECT r.author, res.account, sum(r.comm_count) AS comm_count, "\
                "sum(r.likes_count) AS likes_count, sum(r.reposts_count) AS reposts_count, "\
                "sum(r.views_count) AS views_count, round(avg(res.members), 0) AS members "\
```

```
            "FROM results r inner JOIN resources res ON r.author = res.account "\
            "where r.date >= '" + date_start + "' AND r.date <= '" + date_end +"' " + whereClause + ")
GROUP BY r.author, res.account ORDER BY r.author, res.account ) as agg")

        # поиск комментариев, лайков, репостов, просмотров и подписчиков по теме на ресурсах
        record = cursor.fetchall()
        if record[0][0] == None:
            C = 0
        else:
            C = int(record[0][0])  # количество комментариев по теме
        if record[0][1] == None:
            L = 0
        else:
            L = int(record[0][1])  # количество лайков по теме
        if record[0][2] == None:
            R = 0
        else:
            R = int(record[0][2])  # количество репостов по теме
        if record[0][3] == None:
            V = 0
        else:
            V = int(record[0][3])  # количество просмотров по теме
        if record[0][4] == None or record[0][4] == 1:
            CS = 1
        else:
            CS = int(record[0][4])  # количество подписчиков на ресурсах по теме

        Rce = round((L + R + C ) /CS / CP * 100, 3)  # уровень активности обсуждения темы в обществе
        Rce_list.append(Rce)
        cursor.execute("select count(*) "\
                    "FROM results as r inner join resources res ON r.author = res.account "\
                    "WHERE r.date >= '" + date_start + "' AND r.date <= '" + date_end + "'"\
                    "AND r.ml_tonal_id = 2" + whereClause +")")

        record = cursor.fetchall()
        pos = int(record[0][0])  # количество текстов по теме, имеющих положительную тональность
        cursor.execute("select count(*) " \
                    "FROM results as r inner join resources res ON r.author = res.account "\
                    "WHERE r.date >= '" + date_start + "' AND r.date <= '" + date_end + "'"\
                    "AND r.ml_tonal_id = 1" + whereClause + ")")

        record = cursor.fetchall()
        neg = int(record[0][0])  # количество текстов по теме, имеющих отрицательную тональность
        cursor.execute("select count(*) " \
                    "FROM results as r inner join resources res ON r.author = res.account "\
                    "WHERE r.date >= '%" + date_start + "%' AND r.date <= '" + date_end+ "'"\
                    "AND r.ml_tonal_id = 3" + whereClause + ")")
```

```
        record = cursor.fetchall()
        neut = int(record[0][0])  # количество текстов по теме, имеющих нейтральную тональность

        cursor.execute("select count(*) " \
                    "FROM results as r inner join resources res ON r.author = res.account "\
                    "WHERE r.date >= '%" + date_start + "%' AND r.date <= '" + date_end+ "'"\
                    "AND r.ml_tonal_id is NULL" + whereClause + ")")

        record = cursor.fetchall()
        non_def = int(record[0][0])  # количество текстов по теме, имеющих неопределенную тональность
        print(record[0][0])

        print('Topic - '+ tag)
        print('pos = '+ str(pos))
        print('neg = '+ str(neg))
        print('neut = '+ str(neut))
        print('non_def = '+ str(non_def))
        largest = max([neg, pos, neut, non_def])

        if largest == pos:
            Rts = "положительная"
            Rts_list.append(Rts)
        elif largest == neg:
            Rts = "отрицательная"
            Rts_list.append(Rts)
        elif largest == neut:
            Rts = "нейтральная"
        else:
            Rts = "неопределенная"
            Rts_list.append(Rts)
    else:
        Rct_list.append(0)
        Rce_list.append(0)
        Rts_list.append("неопределенная")

def get_data(request, category):
    data_metrics = 0
    global post_form
    if post_form == False:
        if category == 'politics':
            data_metrics = database_sql(tag_list_politics)
        elif category == 'society':
            data_metrics = database_sql(tag_list_society)
        elif category == 'security':
            data_metrics = database_sql(tag_list_security)
        elif category == 'economics':
            data_metrics = database_sql(tag_list_economics)
        elif category == 'education':
```

```
            data_metrics = database_sql(tag_list_education)
        elif category == 'health':
            data_metrics = database_sql(tag_list_health)
        elif category == 'sport':
            data_metrics = database_sql(tag_list_sport)
        elif category == 'defence':
            data_metrics = database_sql(tag_list_defence)
        elif category == 'ecology':
            data_metrics = database_sql(tag_list_ecology)
    else:
        data_metrics = database_sql(tag_list_general)
        post_form = False

    return JsonResponse(data_metrics, safe=False)

def graphics(request, category):
    context = {}
    context["category"] = category
    context["date_start"] = date_start
    context["date_end"] = date_end
    return render(request, 'graphics.html', context)
```

3. index.html

```
{% extends "base.html"%}
{% load static %}
{% block content %}
<div class="main_groups" style="margin: auto;">
    <div class="group1" align="center">
        <a id="politics" class="block-primary" href="{% url 'graphics' 'politics' %}">
            Политическая <br>
            система
        </a>
        <a id="society" class="block-primary" href="{% url 'graphics' 'society' %}">
            Гражданское <br>
            общество
        </a>
        <a id="security" class="block-primary" href="{% url 'graphics' 'security' %}">
            Гражданская <br>
            безопасность
        </a>
    </div>
    <div class="group2" align="center">
        <a id="economics" class="block-primary" href="{% url 'graphics' 'economics' %}">
            Экономика
        </a>
        <a id="education" class="block-primary" href="{% url 'graphics' 'education'%}">
            Образование
        </a>
        <a id="health" class="block-primary" href="{% url 'graphics' 'health'%}">
            Здравоохранение
```

```
      </a>
   </div>
   <div class="group3" align="center">
      <a id="sport" class="block-primary" href="{% url 'graphics' 'sport'%}">
         Культура и <br> спорт
      </a>
      <a id="defence" class="block-primary" href="{% url 'graphics' 'defence'%}">
         Общественная <br> безопасность
      </a>
      <a id="ecology" class="block-primary" href="{% url 'graphics' 'ecology'%}">
         Экология <br>  
      </a>
   </div>
</div>

<div id="search" align="center">
   {% load crispy_forms_tags %}
   <div id="tags" align="center">
      {% crispy form form.helper %}
   </div>
</div>
{% endblock content %}
```

4. base.html

```
{% load static %}
<html>
<head>
   <meta name="http-equiv" content="charset=windows-1251">
</head>
<body id="body" style="margin: 0; padding: 10px">
   <div id="panel" style="display: flex; padding: 10px; box-shadow: 0 0 4px 0 rgba(0, 0, 0, .5);">
      <div id="logo" style="margin: 0 auto;"><img class="bg_image" src="{% static 'Im-
ages/eSM_logo2.png' %}" style="height: 40px; margin: 0 auto;"/></div>
   </div>
   <div id = "main_content" style="text-align: center;">
      {% block content %}
      {% endblock %}
   </div>
   <footer>
      <div class="footer" style="text-align: center; height: 40px; border-top: 1px solid rgba(0, 0, 0,
0.5); line-height: 40px; clear: both">
         <a href="#">Пользовательское соглашение</a> | <a href="#">Политика конфиденци-
альности</a>
      </div>
   </footer>
   {% block scripts %}
         <script src="{% static 'JS/jquery-3.4.1.min.js' %}"></script>
         <link rel="stylesheet" href="{% static 'CSS/bootstrap.min.css'%}">
         <script src="{%  static 'JS/bootstrap.min.js'%}"></script>
         <link rel="stylesheet" href="{% static 'CSS/bootstrap-datepicker.min.css' %}">
```

```
        <script src="{% static 'JS/bootstrap-datepicker.min.js' %}"></script>
        <script src="{% static 'JS/bootstrap-datepicker.ru.min.js' %}"></script>
        <script src="{% static 'JS/main_script.js' %}"></script>
        <link rel="stylesheet" href="{% static 'CSS/main.css'%}">
    {% endblock%}
</body>
</html>
```

5. graphics.html

```
{% extends "base.html"%}
{% block content %}
<!--<div id="date_start" align="left"> Начальная дата - {{ date_start }}</div>
<div id="date_end" align="left">Конечная дата - {{ date_end }}</div> -->
<div id = "category_panel" align="center">

    <div id="ext_politics" class="panel-blocks">
        <div id="politics" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'politics' %}">
                Политическая <br> система </a></div>
    </div>

    <div id="ext_society" class="panel-blocks">
        <div id="society" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'society' %}">
                Гражданское <br> общество
            </a></div>
    </div>

    <div id="ext_security" class="panel-blocks">
        <div id="security" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'security' %}">
                Гражданская <br> безопасность </a></div>
    </div>

    <div id="ext_economics" class="panel-blocks">
        <div id="economics" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'economics' %}">
                 Экономика <br>  </a></div>
    </div>

    <div id="ext_education" class="panel-blocks">
        <div id="education" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'education' %}">
                Образование <br>   </a></div>
    </div>

    <div id="ext_health" class="panel-blocks">
        <div id="health" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'health' %}">
                Здравоохранение <br>   </a></div>
```

```
    </div>

    <div id="ext_sport" class="panel-blocks">
        <div id="sport" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'sport' %}">
                Культура и <br> спорт </a></div>
    </div>

    <div id="ext_defence" class="panel-blocks">
        <div id="defence" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'defence' %}">
                Общественная <br> безопасность </a></div>
    </div>

    <div id="ext_ecology" class="panel-blocks">
        <div id="ecology" class="int-blocks">
            <a class="graph_links" href="{% url 'graphics' 'ecology' %}">
                Экология <br>  </a></div>
    </div>

    <div class="sign_text">

    </div>
</div>

{% csrf_token %}
<div id="category" category-name="{{ category }}" category-done="{% url 'data' cate-
gory %}"></div>

<div class="ajaxProgres" style="text-align: center;">
    {% load static %}
    <img style="margin-top: 300px;" src="{% static 'Images/loader.gif' %}"/>
</div>

<div id="charts" class="charts">
    <div id="chartContainer" align="center" style="width: 80%; height: 800px; margin:0 auto; mar-
gin-bottom: 30px; margin-top: 30px;">
        <canvas id="chart1" align="center" style="margin:0 auto; margin-bottom: 30px; margin-top:
30px;"></canvas>
    </div>

    <div id="chartContainer2" align="center" style="width: 80%; height: 800px; margin:0 auto; mar-
gin-bottom: 30px;">
        <canvas id="chart2" align="center" style="margin:0 auto; margin-bottom: 30px; margin-top:
30px;"></canvas>
    </div>

    <div id="chartContainer3" style="width: 80%; height: 800px; margin:0 auto; margin-bottom:
30px;">
        <canvas id="chart3" align="center" style="margin:0 auto; margin-bottom: 30px; margin-top:
30px;"></canvas>
```

```
    </div>
</div>

<div id="table" style="margin-bottom:30px;">
    <table align="center" style="width: 80%">
        <thead>
            <tr>
                <th>№</th>
                <th>Тема</th>
                <th>Тональность</th>
            </tr>
        </thead>
        <tbody>
        </tbody>
    </table>
</div>

{% load static %}
<script>
    var sign_very_good = "{% static 'Images/Very_good_mood.jpg' %}";
    var sign_good = "{% static 'Images/Good_mood.jpg' %}";
    var moderately_good = "{% static 'Images/Moderately_good.jpg' %}"
    var stable = "{% static 'Images/Stable_mood.jpg' %}"
    var satisfied = "{% static 'Images/Satisfied_mood.jpg' %}"
    var weak = "{% static 'Images/Weak_mood.jpg' %}"
    var small_tension = "{% static 'Images/Small_social_tension.jpg' %}"
    var middle_tension = "{% static 'Images/Middle_social_tension.jpg' %}";
    var large_tensions = "{% static 'Images/Large_social_tension.jpg' %}";
    var undefined = "{% static 'Images/Undefined_social_mood.jpg' %}";
    var main_page = "<div id='back'><a href = {% url 'index' %} style='display: inline-
block;'>Назад</a></div>";
</script>
<script src="{% static 'JS/jquery-3.4.1.min.js' %}"></script>
<script src="{% static 'JS/jquery.canvasjs.min.js' %}"></script>
<script src="{% static 'JS/Chart.min.js' %}"></script>
<script src="{% static 'JS/bubble_chart.js' %}" charset=windows-1251></script>
<link rel="stylesheet" href="{% static 'CSS/table_style.css' %}">
{% endblock content %}
```

6. bubble_chart.js

```
// JavaScript source code

$(document).ready(function () {
    var points = [];
    var points2 = [];
    var points3 = [];
    var json_data;
    var trHTML = "";
    var category = "";
```

```
    $('.ajaxProgres').show();
    $("#panel").prepend(main_page);
    $("#back").attr("style", "margin: auto 0; text-align: center; width:10%; color:#000000; font-family: 'Calibri'; font-size: 24px;");
    $('.bg_image').css('margin-left', '-150px');
    var bubbleColors = ['rgb(255, 99, 132)', 'rgb(255, 205, 86)', 'rgb(102, 153, 255)', 'rgb(102, 255, 102)',
    'rgb(255, 102, 102)', 'rgb(204, 0, 204)', 'rgb(255, 255, 102)', 'rgb(0,204,0)', 'rgb(102, 102, 255)',
    'rgb(255, 153, 0)', 'rgb(0, 153, 153)', 'rgb(204, 0, 255)', 'rgb(0, 51, 102)', 'rgb(102, 0, 102)', 'rgb(255, 102, 153)']

    function drawChart() {
        var j = 0;
        var xLabels = [];
        var yLabels = [];
        for(var i = 0, len = points.length; i < len; i++)
        {
            xLabels.push(points[i].x);
            yLabels.push(points[i].y);
        }

        while (bubbleColors.length < points.length)
        {
            if(j <= bubbleColors.length)
            {
                bubbleColors.push(bubbleColors[j]);
                j++;
            }
            else
                j = 0;
        }

        var ctx = document.getElementById('chart1').getContext("2d");
        var myLine = new Chart(ctx, {
            type: 'bubble',
            data: {
                xLabels: xLabels,
                yLabels: yLabels,
                datasets: [{
                    label: "Data",
                    data: points,
                    fill: false,
                    showLine: false,
                    borderColor: bubbleColors,
                    backgroundColor: bubbleColors
                }]
            },

            options: {
                responsive: true,
                maintainAspectRatio: false,
```

```
title: {
    display: true,
    fontSize: 34,
    fontStyle: 'bold',
    text: 'Количество найденных тем'
},
legend: {
    display: false
},
scales: {
    xAxes: [{
        type: 'category',
        display: true,
        scaleLabel: {
            display: true,
            labelString: 'Темы',
            fontSize: 32,
            fontStyle: 'bold'
        },
        ticks: {
            fontSize: 32,
            fontStyle: 'bold'
        }
    }],
    yAxes: [{
        position: 'left',
        display: true,
        scaleLabel: {
            display: true,
            labelString: 'Значения',
            fontSize: 32,
            fontStyle: 'bold'
        },
        ticks: {
            fontSize: 32,
            fontStyle: 'bold'
        },
    }],
},
tooltips: {
    custom: function (tooltip) {
        tooltip.displayColors = false;
    },

    callbacks: {
        label: function (t, d) {
            return 'Тема - ' + t.xLabel + '\n' + ' Количество - ' + t.yLabel;
        },
        footerFontSize: {
            size: 40
        }
```

```
            }
        }
    }
});
}

function drawChart2() {
    var j = 0;
    var xLabels = [];
    var yLabels = [];
    for (var i = 0, len = points2.length; i < len; i++) {
        xLabels.push(points2[i].x);
        yLabels.push(points2[i].y);
    }

    var ctx = document.getElementById('chart2').getContext("2d");
    var myLine = new Chart(ctx, {
        type: 'line',
        data: {
            xLabels: xLabels,
            yLabels: yLabels,
            datasets: [{

                label: "Data",
                data: points2,
                fill: false,
                showLines: true,
                borderColor: 'rgb(102, 153, 255)',
                backgroundColor: 'rgb(102, 153, 255)',
                pointRadius: 6
            }]
        },

        options: {
            responsive: true,
            maintainAspectRatio: false,
            title: {
                display: true,
                fontSize: 34,
                fontStyle: 'bold',
                text: 'Уровень заинтересованности темой в обществе'
            },
            legend: {
                display: false
            },
            scales: {
                xAxes: [{
                    type: 'category',
                    display: true,
                    scaleLabel: {
                        display: true,
```

```
                    labelString: 'Темы',
                    fontSize: 32,
                    fontStyle: 'bold'
                },
                ticks: {
                    fontSize: 32,
                    fontStyle: 'bold'
                }
            }],
            yAxes: [{
                position: 'left',
                display: true,
                stacked: true,
                scaleLabel: {
                    display: true,
                    labelString: 'Значения',
                    fontSize: 32,
                    fontStyle: 'bold'
                },
                ticks: {
                    fontSize: 32,
                    fontStyle: 'bold'
                },
            }],
        },
        tooltips: {
            custom: function(tooltip){
                tooltip.displayColors = false;
            },
            fontSize: 32,
            callbacks: {
                label: function (t, d) {
                    return 'Тема - ' + t.xLabel + '\n' + ' Уровень заинтересованности - ' + t.yLabel;
                }
            }
        }
    }
});
}

    function drawChart3() {
    var j = 0;
    var xLabels = [];
    var yLabels = [];
    for (var i = 0, len = points3.length; i < len; i++) {
        xLabels.push(points3[i].x);
        yLabels.push(points3[i].y);
    }

    var ctx = document.getElementById('chart3').getContext("2d");
    var myLine = new Chart(ctx, {
```

```
type: 'line',
data: {
    xLabels: xLabels,
    yLabels: yLabels,
    datasets: [{
        label: "Data",
        data: points3,
        fill: false,
        showLines: true,
        borderColor: 'rgb(0,204,0)',
        backgroundColor: 'rgb(0,204,0)',
        pointRadius: 6
    }]
},

options: {
    responsive: true,
    maintainAspectRatio: false,
    title: {
        display: true,
        fontSize: 34,
        fontStyle: 'bold',
        text: 'Уровень активности обсуждения темы в обществе'
    },
    legend: {
        display: false
    },
    scales: {
        xAxes: [{
            type: 'category',
            display: true,
            scaleLabel: {
                display: true,
                labelString: 'Темы',
                fontSize: 32,
                fontStyle: 'bold'
            },
            ticks: {
                fontSize: 32,
                fontStyle: 'bold'
            }
        }],
        yAxes: [{
            position: 'left',
            display: true,
            stacked: true,
            scaleLabel: {
                display: true,
                labelString: 'Значения',
                fontSize: 32,
                fontStyle: 'bold'
```

```
                },
                ticks: {
                    fontSize: 32,
                    fontStyle: 'bold'
                },
            }],
        },
        tooltips: {
            custom: function(tooltip){
                tooltip.displayColors = false;
            },
            fontSize: 32,
            callbacks: {
                label: function (t, d) {
                    return 'Тема - ' + t.xLabel + '\n' + ' Уровень активности обсуждения - ' + t.yLa-
bel;
                }
            }
        }
    }
});
}

$.ajax(
    {
        url: $('#category').attr('category-done'),
        type: "GET",
        success: function (data) {
            json_data = data;
            var radius;
            var stepSize = 5;
            for (var i = 0; i < Object.keys(json_data[0]).length; i++) {
                if (json_data[0][i].quantity <= 10)
                    radius = Math.round(json_data[0][i].quantity) + stepSize;
                else if (json_data[0][i].quantity > 10 && json_data[0][i].quantity <= 25)
                    radius = Math.round(json_data[0][i].quantity / 2 + stepSize * 2);
                else if (json_data[0][i].quantity > 25 && json_data[0][i].quantity <= 50)
                    radius = 30;
                else if (json_data[0][i].quantity > 50 && json_data[0][i].quantity <= 100)
                    radius = 35;
                else if (json_data[0][i].quantity > 100 && json_data[0][i].quantity <= 500)
                    radius = 40;
                else if (json_data[0][i].quantity > 500 && json_data[0][i].quantity <= 1000)
                    radius = 45;
                else if (json_data[0][i].quantity > 1000 && json_data[0][i].quantity <= 1500)
                    radius = 50;
                else if (json_data[0][i].quantity > 1500 && json_data[0][i].quantity <= 2000)
                    radius = 55;
                else if (json_data[0][i].quantity > 2000 && json_data[0][i].quantity <= 2500)
                    radius = 60;
                else if (json_data[0][i].quantity > 2500 && json_data[0][i].quantity <= 3000)
```

```
                radius = 65;
            else if (json_data[0][i].quantity > 3000 && json_data[0][i].quantity <= 3500)
                radius = 70;
            else if (json_data[0][i].quantity > 3500 && json_data[0][i].quantity <= 4000)
                radius = 75;
            else if (json_data[0][i].quantity > 4000 && json_data[0][i].quantity <= 4500)
                radius = 80;
            else if (json_data[0][i].quantity > 4500 && json_data[0][i].quantity <= 5000)
                radius = 85;
            else if (json_data[0][i].quantity > 5000)
                radius = 90;
            points.push({x: json_data[0][i].name, y: json_data[0][i].quantity, r: radius});
        }
        for (var i = 0; i < Object.keys(json_data[1]).length; i++) {
            points2.push({x: json_data[1][i].name, y: json_data[1][i].count});
        }
        for (var i = 0; i < Object.keys(json_data[2]).length; i++) {
            points3.push({x: json_data[2][i].name, y: json_data[2][i].count});
        }
        for (var i = 0; i < Object.keys(json_data[3]).length; i++) {
            trHTML += '<tr><td>' + i + '</td><td style="text-decoration: underline;">' +
json_data[3][i].name + '</td><td>' + json_data[3][i].tonality + '</td></tr>';
        }
        sign_text = '';
        category = $('#category').attr('category-name')
        if (data[4][0] == 'положительная' && data[5][0] >= 0.015 && data[6][0] >= 200) {
            $('#ext_' + category).prepend('<img id="sign_image" src="' + sign_very_good + '"/>');
            sign_text = 'Очень хорошее социальное настроение';
        }
        else if (data[4][0] == 'положительная' && data[5][0] >= 0.015 && data[6][0] < 200) {
            $('#ext_' + category).prepend('<img id="sign_image" src="' + sign_good + '"/>');
            sign_text = 'Хорошее социальное настроение';
        } else if (data[4][0] == 'положительная' && data[5][0] < 0.015 && data[6][0] >= 200) {
            $('#ext_' + category).prepend('<img id="sign_image" src="' + sign_good + '"/>');
            sign_text = 'Хорошее социальное настроение';
        } else if (data[4][0] == 'положительная' && data[5][0] < 0.015 && data[6][0] < 200) {
            $('#ext_' + category).prepend('<img id="sign_image" src="' + moderately_good +
'"/>');
            sign_text = 'Умеренно хорошее социальное настроение';
        }
        else if (data[4][0] == 'нейтральная' && data[5][0] >= 0.015 && data[6][0] >= 200) {
            $('#ext_' + category).prepend('<img id="sign_image" src="' + stable + '"/>');
            sign_text = 'Стабильное социальное настроение';
        }
        else if (data[4][0] == 'нейтральная' && data[5][0] < 0.015 && data[6][0] >= 200) {
            $('#ext_' + category).prepend('<img id="sign_image" src="' + satisfied + '"/>');
            sign_text = 'Удовлетворительное социальное настроение';
        }
        else if (data[4][0] == 'нейтральная' && data[5][0] >= 0.015 && data[6][0] < 200) {
            $('#ext_' + category).prepend('<img id="sign_image" src="' + satisfied + '"/>');
            sign_text = 'Удовлетворительное социальное настроение';
```

```
            }
            else if (data[4][0] == 'нейтральная' && data[5][0] < 0.015 && data[6][0] < 200) {
                $('#ext_' + category).prepend('<img id="sign_image" src="' + weak + '"/>');
                sign_text = 'Слабо выраженное социальное настроение';
            }
            else if (data[4][0] == 'отрицательная' && data[5][0] < 0.015 && data[6][0] < 200) {
                $('#ext_' + category).prepend('<img id="sign_image" src="' + small_tension + '"/>');
                sign_text = 'Небольшая социальная напряженность';
            }
            else if (data[4][0] == 'отрицательная' && data[5][0] >= 0.015 && data[6][0] < 200) {
                $('#ext_' + category).prepend('<img id="sign_image" src="' + middle_tension + '"/>');
                sign_text = 'Средняя социальная напряженность';
            }
            else if (data[4][0] == 'отрицательная' && data[5][0] < 0.015 && data[6][0] >= 200) {
                $('#ext_' + category).prepend('<img id="sign_image" src="' + middle_tension + '"/>');
                sign_text = 'Средняя социальная напряженность';
            }
            else if (data[4][0] == 'отрицательная' && data[5][0] >= 0.015 && data[6][0] >= 200) {
                $('#ext_' + category).prepend('<img id="sign_image" src="' + large_tensions + '"/>');
                sign_text = 'Высокая социальная напряженность';
            }
            else {
                $('#ext_' + category).prepend('<img id="sign_image" src="' + undefined + '"/>');
                sign_text = 'Неопределенная ситуация';
            }
            $('.ajaxProgres').hide();
            $("#table").show();
            $("#charts").show();
            //$("#date_start").show();
            //$("#date_end").show();
            $('#category_panel').css('display', 'block').css('vertical-align', 'middle');
            $('.sign_text').append(sign_text);
            $('#ext_' + category).mouseover(function () {
                $(".sign_text").css('display', 'block').offset({
                    top: $(this).offset().top - 20,
                    left: $(this).offset().left + 100
                });
            });
            $('#ext_' + category).mouseout(function () {
                $(".sign_text").css('display', 'none');
            });
            drawChart();
            drawChart2();
            drawChart3();
            $('tbody').append(trHTML);
        }
    });
});
```