Лабораторное занятие 8. Решение задач на L2 регулирование.

8-зертханалық сабақ. L2 реттеу ге есептер шығару.

```
import sklearn.model_selection as GridSearchCV
# отключим всякие предупреждения Anaconda
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
from matplotlib import pyplot as plt
import seaborn as sns

import numpy as np
import pandas as pd
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LogisticRegression, LogisticRegressionCV
from sklearn.model_selection import cross_val_score, StratifiedKFold
from sklearn.model_selection import GridSearchCV
data=pd.read_csv("dataregression2.csv")
data.head(10) X = data.ix[:,:3].values
y = data.ix[:,3].values
plt.scatter(X[y == 1, 0], X[y == 1, 1], c='green', label='Выпущен')
plt.scatter(X[y == 0, 0], X[y == 0, 1], c='red', label='Бракован')
plt.xlabel("Тест 1")
plt.ylabel("Тест 2")
plt.title('2 теста микрочипов')
plt.legend();
```

Определяем функцию для отображения разделяющей кривой классификатора

```
import sklearn.linear_model
import sklearn.svm
from sklearn.linear_model import LinearRegression
clf=LogisticRegression()
from future import print_function, division
#Создадим объект sklearn, который добавит в матрицу X полиномиальные
#признаки вплоть до степени 7 и обучим логистическую регрессию с
#параметром регуляризации C=10−2. Изобразим разделяющую границу.
poly = PolynomialFeatures(degree=7)
X_poly = poly.fit_transform(X)
C = 1
logit = LogisticRegression(C=C, n_jobs=-1, random_state=17)
logit.fit(X_poly, y)

plot_boundary(logit, X, y, grid_step=.005, poly_featurizer=poly)

plt.scatter(X[y == 1, 0], X[y == 1, 1], c='green', label='Выпущен')
plt.scatter(X[y == 0, 0], X[y == 0, 1], c='red', label='Бракован')
plt.xlabel("Тест 1")
plt.ylabel("Тест 2")
plt.title('2 теста микрочипов. Логит с C=1')
plt.legend();

print("Доля правильных ответов классификатора на обучающей выборке:",
round(logit.score(X_poly, y), 3))
C = 1e4
logit = LogisticRegression(C=C, n_jobs=-1, random_state=17)
```

```
logit.fit(X_poly, y)


plot_boundary(logit, X, y, grid_step=.005, poly_featurizer=poly)


plt.scatter(X[y == 1, 0], X[y == 1, 1], c='green', label='Выпущен')
plt.scatter(X[y == 0, 0], X[y == 0, 1], c='red', label='Бракован')
plt.xlabel("Тест 1")
plt.ylabel("Тест 2")
plt.title('2 теста микрочипов. Логит с C=10k')
plt.legend();


print("Доля правильных ответов классификатора на обучающей выборке:",
round(logit.score(X_poly, y), 3))
#
```

Настройка параметра регуляризации

Теперь найдем оптимальное (в данном примере) значение параметра регуляризации C. Сделать это можно с помощью LogisticRegressionCV – перебора параметров по сетке с последующей кросс-валидацией. Этот класс создан специально для логистической регрессии (для нее известны эффективные алгоритмы перебора параметров), для произвольной модели мы бы использовали GridSearchCV, RandomizedSearchCV или, например, специальные алгоритмы оптимизации гиперпараметров, реализованные в hyperopt.

```
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=17)
c_values = np.logspace(-2, 3, 500)
logit_searcher = LogisticRegressionCV(Cs=c_values, cv=skf, verbose=1, n_jobs=-1)
logit_searcher.fit(X_poly, y)
```