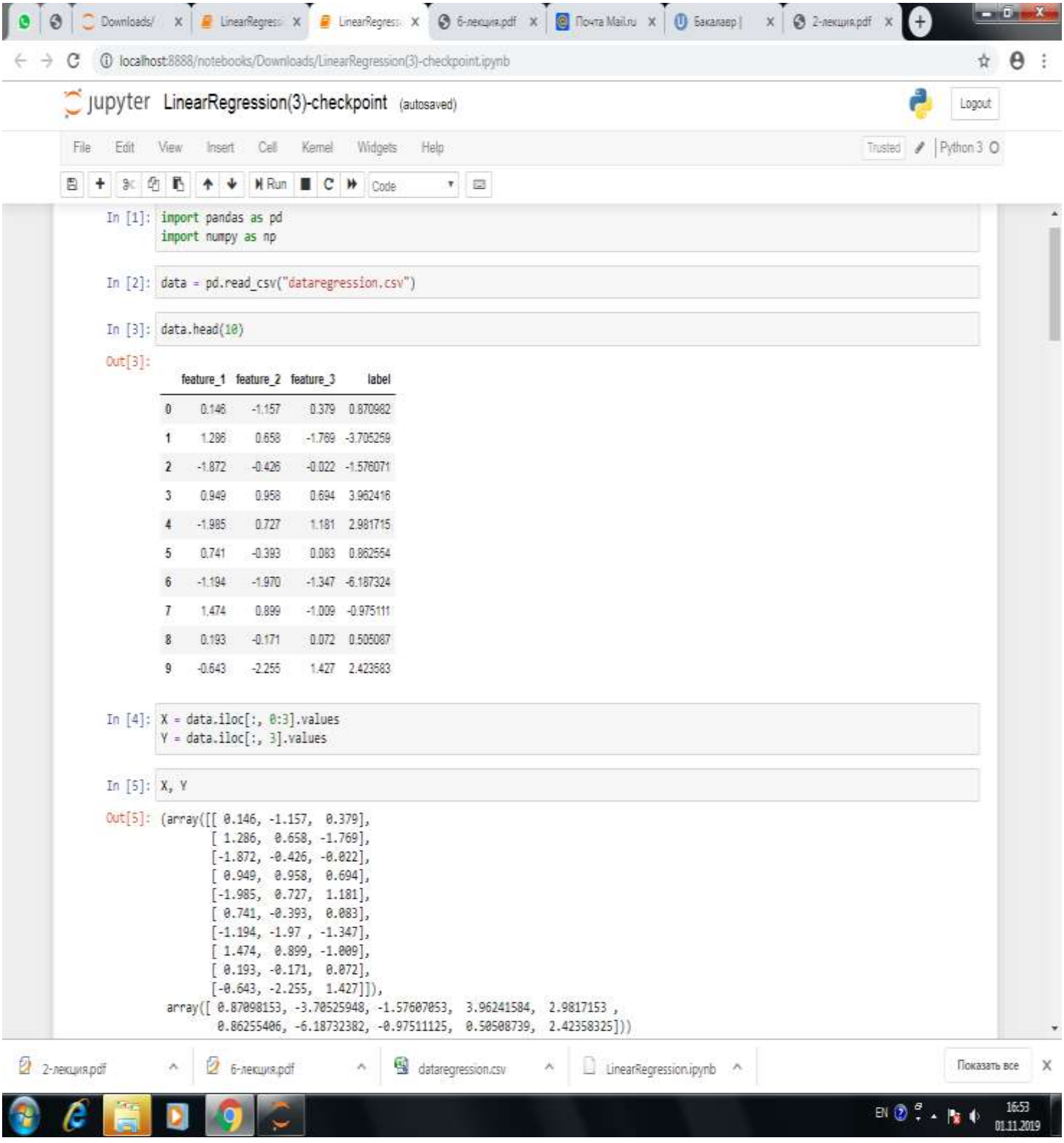
Лабораторное занятие 7. Метод логистической регрессии на Python -е

7-зертханалық сабақ. Python да логистикалық регрессия әдісін жүзеге асыру.



jupyter LinearRegression(3)-checkpoint (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [6]:
```
X = np.hstack([ np.ones((10, 1)), X])
```

In [7]:
```
X
```

Out[7]:
```
array([[ 1.   ,  0.146, -1.157,  0.379],
       [ 1.   ,  1.286,  0.658, -1.769],
       [ 1.   , -1.872, -0.426, -0.022],
       [ 1.   ,  0.949,  0.958,  0.694],
       [ 1.   , -1.985,  0.727,  1.181],
       [ 1.   ,  0.741, -0.393,  0.083],
       [ 1.   , -1.194, -1.97 , -1.347],
       [ 1.   ,  1.474,  0.899, -1.009],
       [ 1.   ,  0.193, -0.171,  0.072],
       [ 1.   , -0.643, -2.255,  1.427]])
```

In [8]:
```
X_tr = np.transpose(X)
X_tr_X = np.dot(X_tr, X)
X_tr_X_inv = np.linalg.inv(X_tr_X)
coef = np.dot(X_tr_X_inv, X_tr)
params_analytic = np.dot(coef, Y)
```

In [9]:
```
X_tr_X, X_tr_X_inv
```

Out[9]:
```
(array([[10.      , -0.905   , -3.13    , -0.311   ],
        [-0.905   , 14.618413,  5.74384 , -4.585205],
        [-3.13    ,  5.74384 , 13.357198, -1.586011],
        [-0.311   , -4.585205, -1.586011, 10.030775]]),
 array([[ 0.10850148, -0.00174249,  0.02698602,  0.00683441],
        [-0.00174249,  0.09436244, -0.03655694,  0.03730016],
        [ 0.02698602, -0.03655694,  0.09684311, -0.00056169],
        [ 0.00683441,  0.03730016, -0.00056169,  0.1168667 ]]))
```

In [10]:
```
params_analytic
```

Out[10]:
```
array([0.31749375, 0.78032738, 0.74979223, 3.08462768])
```

In [11]:
```
teta_grad_start = np.zeros(4)
learning_rate = 0.001
epsilon = 0.00001
max_iter = 10000
```

```
def step(teta):
    return teta - learning_rate * (1.0/10) * np.dot(np.transpose(X), (np.dot(X, teta) - Y))

teta_GD = teta_grad_start
for i in range(max_iter):
    teta_GD_new = step(teta_GD)
    if np.sum(np.abs(teta_GD_new - teta_GD)) <= epsilon:
        break
    else:
        teta_GD = teta_GD_new

print(i)
print(teta_GD)
```

```
8454
[0.31673068 0.77492892 0.75208069 3.07801993]
```

In [12]:
```
from sklearn.linear_model import LinearRegression
```

In [13]:
```
LR = LinearRegression()
```

In [14]:
```
X = data.iloc[:, 0:3].values
Y = data.iloc[:, 3].values
LR.fit(X, Y)
```

Out[14]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [15]:
```
LR.coef_, LR.intercept_
```

Out[15]:
```
(array([0.78032738, 0.74979223, 3.08462768]), 0.31749374659996565)
```

In [16]:
```
Y_pred=LR.predict(X)
```

In [17]:
```
Y_pred
```

Out[17]:
```
array([ 0.73298582, -3.64234832, -1.53055241,  3.917057  ,  2.95658814,
        0.85707209, -6.24630133, -0.97062981,  0.56197565,  2.52672546])
```

In [18]:
```
eps=Y-Y_pred
eps
```

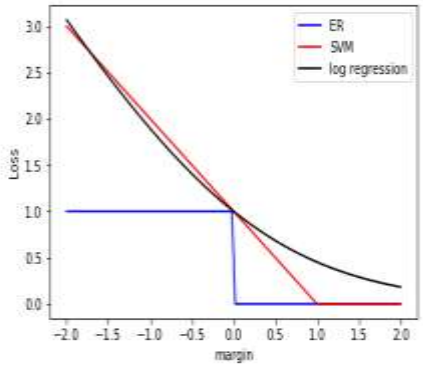
In [18]:
```
eps=Y-Y_pred
eps
```

Out[18]:
```
array([ 0.13799571, -0.06291116, -0.04551812,  0.04535884,  0.02512716,
        0.00548197,  0.05897751, -0.00448144, -0.05688826, -0.10314221])
```

In [24]:
```
from sklearn import metrics
print("r2=", metrics.r2_score(Y,Y_pred))
```

```
r2= 0.9994844216048888
```

In [26]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
X = np.linspace(-20, 20, 100) / 10.0
plt.plot(X, [1 if x < 0 else 0 for x in X], color = [0, 0, 1], label="ER")
plt.plot(X, [0 if ((1 - x) < 0) else (1 - x) for x in X], color = [1, 0, 0], label="SVM"
)
plt.plot(X, np.log2(1 + np.exp(-1 * X)), color = [0, 0, 0], label="log regression")
plt.legend()
plt.xlabel("margin")
plt.ylabel("Loss")
plt.show()
```



In [27]:
```
X = np.linspace(-100, 100, 100) / 10.0
plt.plot(X, 1 / (1 + np.exp(-1 * X)), color = [1, 0, 0])
```

Downloads/ | LinearRegress | LinearRegress | 6-лекция.pdf | Почта Mail.ru | Бакалавр | 2-лекция.pdf

localhost:8888/notebooks/Downloads/LinearRegression(3)-checkpoint.ipynb

jupyter LinearRegression(3)-checkpoint (autosaved)

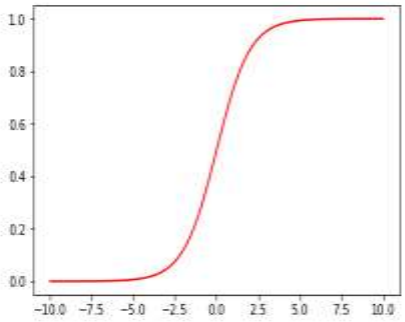
Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

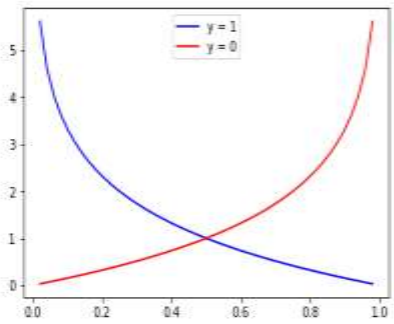
Run Code

Out[27]: [<matplotlib.lines.Line2D at 0xb579f28>]



In [28]:
```
X = np.linspace(0, 10, 50) / 10.0
X = X[1:-1]
plt.plot(X, np.log2(X) * -1, color = [0, 0, 1], label="y = 1")
plt.plot(X, np.log2(1 - X) * -1, color = [1, 0, 0], label="y = 0")
plt.legend()
```

Out[28]: <matplotlib.legend.Legend at 0xb5a8ba8>



In [29]:
```
from sklearn import linear_model, datasets
from sklearn.model_selection import train_test_split
```

2-лекция.pdf | 6-лекция.pdf | dataregression.csv | LinearRegression.ipynb | Показать все

EN 16:58 01.11.2019

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import pandas as pd
```

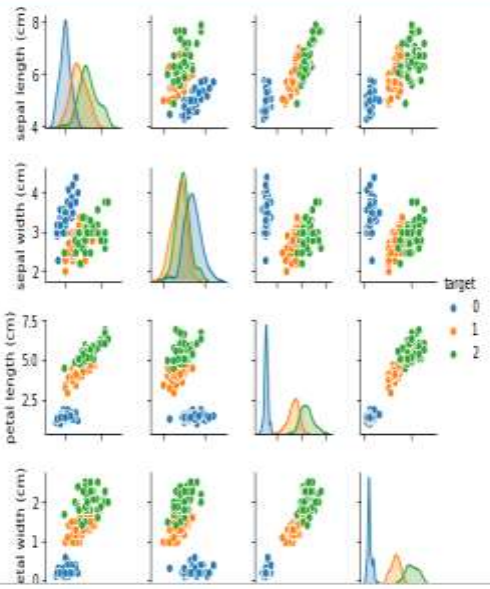
In [30]:
```
iris = datasets.load_iris()
```

In [31]:
```
iris.feature_names
```

Out[31]:
```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

In [35]:
```
X_DF=pd.DataFrame(data=iris.data,columns=iris.feature_names)
X_DF["target"]=iris.target
g=sns.pairplot(X_DF,hue="target",vars=iris.feature_names,size=1.5)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\axisgrid.py:2065: UserWarning: The `size` parameter has been renamed to `hei
ght`; pleaes update your code.
  warnings.warn(msg, UserWarning)
```



jupyter LinearRegression(3)-checkpoint (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted | Python 3

Run Code

In [37]:
```
X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state=42)
```

In [39]:
```
logreg = linear_model.LogisticRegression()
logreg.fit(X_train, y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: Default solver will be changed
to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:460: FutureWarning: Default multi_class will be cha
nged to 'auto' in 0.22. Specify the multi_class option to silence this warning.
  "this warning.", FutureWarning)
```

Out[39]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

In [40]:
```
y_pred = logreg.predict(X_test)
```

In [41]:
```
confusion_matrix(y_test, y_pred)
```

Out[41]:
```
array([[29,  0,  0],
       [ 0, 21,  2],
       [ 0,  0, 23]], dtype=int64)
```

In [42]:
```
logreg.coef_, logreg.intercept_
```

Out[42]:
```
(array([[ 0.34727293,  1.1961666 , -1.92844924, -0.85983715],
        [ 0.45226594, -1.40595322,  0.34170865, -0.88352211],
        [-1.3541608 , -1.23978952,  2.01033825,  1.82990522]]),
 array([ 0.22596203,  0.66625886, -0.87817104]))
```

In [ ]: