

**Московский государственный технический  
университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Теория машинного обучения»

Отчет по лабораторной работе №3

Выполнил:  
студент группы ИУ5-64  
Кузьмин Роман

Подпись и дата:

Проверил:  
преподаватель каф. ИУ5  
Гапанюк Ю.Е.

Подпись и дата:

Москва, 2022 г.

## Описание задания

1. Выберите набор данных (датасет) для решения задачи классификации или регрессии.
2. С использованием метода `train_test_split` разделите выборку на обучающую и тестовую.
3. Обучите модель ближайших соседей для произвольно заданного гиперпараметра  $K$ . Оцените качество модели с помощью подходящих для задачи метрик.
4. Произведите подбор гиперпараметра  $K$  с использованием `GridSearchCV` и/или `RandomizedSearchCV` и кросс-валидации, оцените качество оптимальной модели. Желательно использование нескольких стратегий кросс-валидации.
5. Сравните метрики качества исходной и оптимальной моделей.

## Текст программы и её результаты

```
from sklearn import datasets, svm
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
import seaborn as sns
import pandas as pd
import numpy as np
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import precision_score, recall_score
```

```
data = pd.read_csv("/content/diabet.csv", index_col=False, header=None,
                  names=['PREG', 'GLUK', 'PRES', 'THIC', 'INS', 'MAS', 'FUN', 'AGE', 'CLAS'])
data.head()
```

	PREG	GLUK	PRES	THIC	INS	MAS	FUN	AGE	CLAS
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
[ ] X_train, X_test, y_train, y_test = train_test_split(
    data.drop('CLAS', 1), data["CLAS"], test_size=0.3, random_state=1)
```

```
[ ] clf = KNeighborsClassifier(n_neighbors=55)
    clf.fit(X_train, y_train)
    target1_0 = clf.predict(X_train)
    target1_1 = clf.predict(X_test)
    accuracy_score(y_train, target1_0), accuracy_score(y_test, target1_1)

(0.7359635811836115, 0.7522935779816514)
```

```
parameters = {'n_neighbors': range(3, 100, 5)}
clf2 = KNeighborsClassifier()
grid = GridSearchCV(clf, parameters)

skf = StratifiedKFold(n_splits=7)
X = data.drop('CLAS', 1)
y = data["CLAS"]
for train_index, test_index in skf.split(X, y):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    grid.fit(X_train, y_train)
    print("Параметры: ", grid.best_params_)
```

/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:6: FutureWarning

```
Параметры: {'n_neighbors': 13}
Параметры: {'n_neighbors': 13}
Параметры: {'n_neighbors': 8}
Параметры: {'n_neighbors': 13}
Параметры: {'n_neighbors': 28}
Параметры: {'n_neighbors': 8}
Параметры: {'n_neighbors': 13}
```

```
[ ] clf2 = KNeighborsClassifier(n_neighbors=13)
    clf2.fit(X_train, y_train)
    target1_0 = clf2.predict(X_train)
    target1_1 = clf2.predict(X_test)
    accuracy_score(y_train, target1_0), accuracy_score(y_test, target1_1)

(0.7860394537177542, 0.7339449541284404)
```

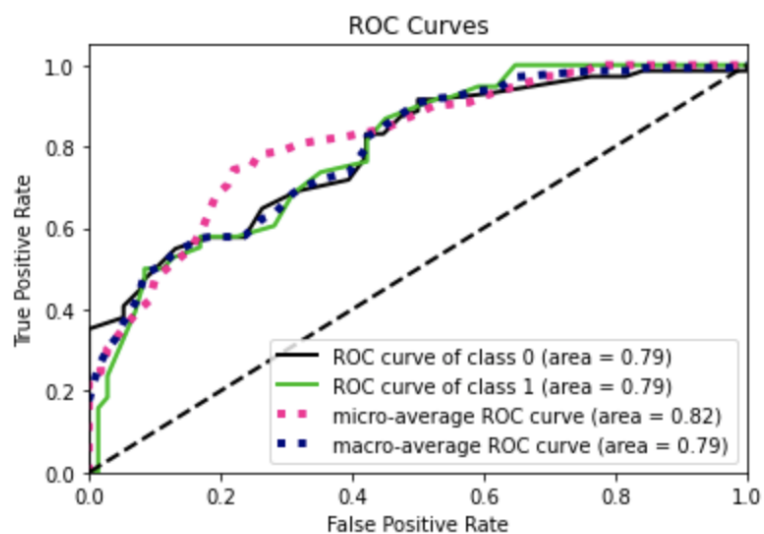
```
[ ] probs1 = clf.predict_proba(X_test)
    probs2 = clf2.predict_proba(X_test)
```

```
[ ] %pip install scikit-plot
```

```
▶ import scikitplot as skplt
   import matplotlib.pyplot as plt
```

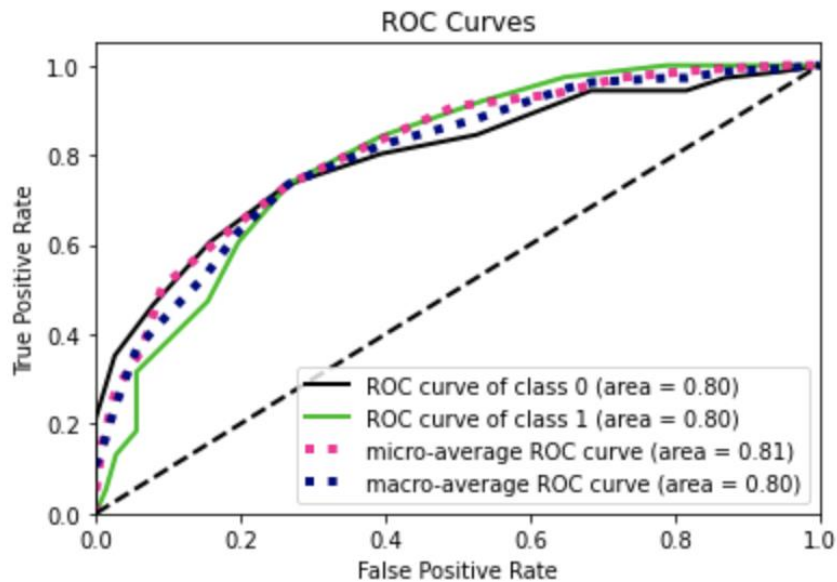
```
skplt.metrics.plot_roc_curve(y_test, probs1)
plt.show()
```

```
👤 /usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87
   warnings.warn(msg, category=FutureWarning)
```



```
[ ] skplt.metrics.plot_roc_curve(y_test, probs2)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation
warnings.warn(msg, category=FutureWarning)
```



```
▶ pred1 = clf.predict(X_test)
  pred2 = clf2.predict(X_test)
  print("Модель 1:")
  print(precision_score(pred1, y_test))
  print(recall_score(pred1, y_test))
  print("Модель 2:")
  print(precision_score(pred2, y_test))
  print(recall_score(pred2, y_test))
```

```
👤 Модель 1:
  0.5
  0.7037037037037037
  Модель 2:
  0.6052631578947368
  0.6216216216216216
```