Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №4
по дисциплине
«Технологии машинного обучения»
на тему
«Подготовка обучающей и тестовой выборки, кросс-валидация и подбор гиперпараметров на примере метода ближайших соседей»

Выполнил:
студент группы ИУ5-63Б
Кривцов Н. А.

_____

Москва — 2020 г.

1. Лабораторная работа №4. Подготовка обучающей и тестовой выборки, кросс-валидация и подбор гиперпараметров на примере метода ближайших соседей.

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, cross_val_score,
 ↪KFold, RepeatedKFold, ShuffleSplit, StratifiedKFold,
 ↪RepeatedStratifiedKFold, StratifiedShuffleSplit, GridSearchCV
from sklearn.neighbors import KNeighborsClassifier
from sklearn.datasets import load_wine
from sklearn.metrics import classification_report, SCORERS
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19:
FutureWarning: pandas.util.testing is deprecated. Use the functions in
 ↪the
public API at pandas.testing instead.
  import pandas.util.testing as tm
```

1.1. Загрузка набора данных. Разбиение на тестовую и обучающую выборки

[ ]:
```
X, y = load_wine(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(X, y,
 ↪random_state=42)
```

[3]:
```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(133, 13)
(45, 13)
(133,)
(45,)
```

1.2. KNN с тремя соседями

[4]:
```
classifier_3 = KNeighborsClassifier(n_neighbors=3)
classifier_3.fit(X_train, y_train)
predicted = classifier_3.predict(X_test)
print(classification_report(y_test, predicted))
```

```
              precision    recall  f1-score   support
```

```
              0       0.87      0.87      0.87        15
              1       0.81      0.72      0.76        18
              2       0.57      0.67      0.62        12

       accuracy                           0.76        45
      macro avg       0.75      0.75      0.75        45
   weighted avg       0.77      0.76      0.76        45
```

1.3. Кросс-валидация

[ ]:
```
kf = KFold(n_splits=5)
rkf = RepeatedKFold(n_splits=5, n_repeats=3)
ss = ShuffleSplit(n_splits=10)
skf = StratifiedKFold(n_splits=5)
rskf = RepeatedStratifiedKFold(n_splits=5, n_repeats=3)
sss = StratifiedShuffleSplit(n_splits=10)

validators = [kf, rkf, ss, skf, rskf, sss]
```

[6]:
```
for val in validators:
  scores = cross_val_score(KNeighborsClassifier(n_neighbors=3), X, y,
 ↪cv=val, scoring="f1_macro")
  print(type(val).__name__)
  print(scores)
  print(scores.mean(), "±", scores.std(), "\n")
```

```
KFold
[0.46268657 0.52163743 0.23030303 0.5037037  0.05263158]
0.3541924614037732 ± 0.18354833477610105

RepeatedKFold
[0.72483455 0.63504274 0.6028462  0.7245671  0.77482239 0.66999197
 0.68668047 0.73214286 0.53459223 0.71895425 0.80064269 0.65925926
 0.61532129 0.77089783 0.63006189]
0.6853771813100387 ± 0.07116962204541094

ShuffleSplit
[0.71282051 0.52096052 0.59259259 0.66045066 0.66045066 0.64057239
 0.56654457 0.61111111 0.92673993 0.72222222]
0.6614465164465165 ± 0.10638068007568899

StratifiedKFold
[0.63053613 0.69075369 0.65873016 0.63174603 0.82666667]
0.6876865356865356 ± 0.07288191106697622

RepeatedStratifiedKFold
[0.55982906 0.6925561  0.73760684 0.6540404  0.74344168 0.64796992
 0.77753623 0.70414295 0.68439898 0.70299145 0.65555556 0.69444011
 0.78101209 0.72996835 0.65555556]
```

```
0.6947363517818592 ± 0.05485285977152599

StratifiedShuffleSplit
[0.52096052 0.81562882 0.82222222 0.72294372 0.55952381 0.76349206
 0.77777778 0.78333333 0.78166278 0.93939394]
0.7486938986938986 ± 0.11743755101794308
```

1.4. Подбор гиперпараметра K

[7]:
```
n_range = np.array(range(1, 11, 1))
tuned_parameters = [{'n_neighbors': n_range}]
tuned_parameters
```

[7]:
```
[{'n_neighbors': array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10])}]
```

[8]:
```
%%time
clf_gs = GridSearchCV(KNeighborsClassifier(), tuned_parameters, cv=5,
 ↪scoring='f1_macro')
clf_gs.fit(X_train, y_train)
```

```
CPU times: user 145 ms, sys: 253 µs, total: 146 ms
Wall time: 147 ms
```

[9]:
```
# Лучший классификатор
clf_gs.best_estimator_
```

[9]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=1,
 ↪p=2,
                     weights='uniform')
```

[10]:
```
# Лучшая F1-метрика
clf_gs.best_score_
```

[10]:
```
0.7438210961152139
```

[11]:
```
# Лучшее значение гиперпараметра
clf_gs.best_params_
```

[11]:
```
{'n_neighbors': 1}
```