

Московский Государственный Технический Университет им. Н. Э. Баумана
Факультет «Информатика и Системы управления»
Кафедра «Автоматизированные системы обработки информации и
управления»
Дисциплина «Технологии машинного обучения»

Отчёт по лабораторной работе №3
**«Подготовка обучающей и тестовой выборки, кросс-валидация и подбор
гиперпараметров на примере метода ближайших соседей.»**

Выполнил:
Студент группы ИУ5ц-83Б
Костников И.А.
Преподаватель:
Гапанюк Ю.Е.

Москва, 2020 г.

1 Цель работы

Изучение способов подготовки выборки и подбора гиперпараметров на примере метода ближайших соседей.

2 Краткое описание

Подготовка данных

3 Текст программы

Текст программы представлена во втором файле (Lab3wine)

4 Экранные формы с примерами выполнения программы.

```
[80] > wine = load_wine()
[81] > wine.feature_names

['alcohol',
 'malic_acid',
 'ash',
 'alcalinity_of_ash',
 'magnesium',
 'total_phenols',
 'flavanoids',
 'nonflavanoid_phenols',
 'proanthocyanins',
 'color_intensity',
 'hue',
 'od280/od315_of_diluted_wines',
 'proline']

[82] > np.unique(wine.target)

array([0, 1, 2])

[83] > wine.target_names

array(['class_0', 'class_1', 'class_2'], dtype='<U7')
```

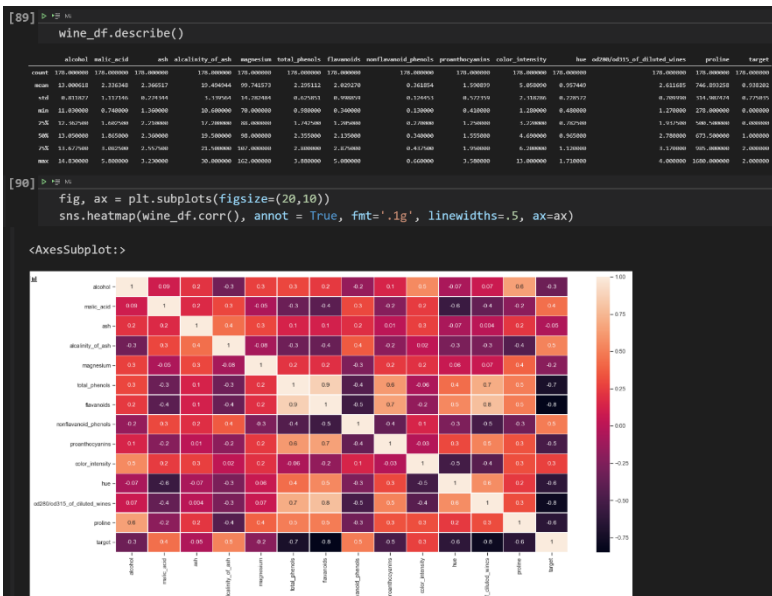
```
[86] > # Сформируем DataFrame
      wine_df = pd.DataFrame(data=np.c_[wine['data'], wine['target']], columns=wine['feature_names'] + ['target'])
[87] > wine_df

      alcohol  malic_acid  ash  alcalinity_of_ash  magnesium  total_phenols  flavanoids  nonflavanoid_phenols  proanthocyanins  color_intensity  hue  od280/od315_of_diluted_wines  proline  target
0      14.23      1.71  2.43         15.6      127.0         2.80         3.06         0.28         2.29         5.64  1.04         3.92  1065.0  0.0
1      13.20      1.78  2.14         11.2      109.0         2.65         2.76         0.26         1.28         4.38  1.05         3.40  1050.0  0.0
2      13.16      2.36  2.67         18.6      101.0         2.80         3.24         0.30         2.81         5.68  1.03         3.17  1185.0  0.0
3      14.17      1.95  2.50         16.8      113.0         3.85         3.49         0.24         2.18         7.80  0.85         3.45  1480.0  0.0
4      13.24      2.59  2.87         21.0      118.0         2.80         2.69         0.39         1.82         4.32  1.04         2.93   735.0  0.0
...      ...      ...      ...      ...      ...      ...      ...      ...      ...      ...      ...      ...      ...
173     13.71      5.05  2.45         20.5       95.0         1.68         0.61         0.52         1.00         7.70  0.64         1.74   740.0  2.0
174     13.40      3.91  2.48         23.0      102.0         1.80         0.75         0.43         1.41         7.30  0.70         1.56   750.0  2.0
175     13.27      4.28  2.26         20.0      120.0         1.59         0.69         0.43         1.35         10.20  0.59         1.56   835.0  2.0
176     13.17      2.59  2.37         20.0      120.0         1.65         0.68         0.53         1.46         9.30  0.60         1.62   840.0  2.0
177     14.13      4.10  2.74         24.5       96.0         2.05         0.76         0.56         1.35         9.20  0.61         1.60   560.0  2.0

178 rows x 14 columns

[88] > wine_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 14 columns):
#   Column              Non-Null Count  Dtype
---  -
0   alcohol              178 non-null    float64
1   malic_acid           178 non-null    float64
2   ash                  178 non-null    float64
3   alcalinity_of_ash    178 non-null    float64
4   magnesium            178 non-null    float64
5   total_phenols        178 non-null    float64
6   flavanoids           178 non-null    float64
7   nonflavanoid_phenols 178 non-null    float64
8   proanthocyanins      178 non-null    float64
9   color_intensity       178 non-null    float64
10  hue                   178 non-null    float64
11  od280/od315_of_diluted_wines 178 non-null    float64
12  proline              178 non-null    float64
13  target                178 non-null    object
```



Построение базовой модели на основе ближайших соседей

```
[97] In [98]: # 2 ближайших соседа
cl1_1 = KNeighborsClassifier(n_neighbors=2)
cl1_1.fit(wine_X_train, wine_y_train)
target1_1 = cl1_1.predict(wine_X_test)
len(target1_1, target1_1)

(89,
array([0, 1, 2, 1, 0, 1, 2, 0, 1, 1, 0, 1, 1, 0, 2, 1, 1, 1, 1, 0, 0, 1,
       1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0,
       0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 2, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 2, 2, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0,
       1]))
```

```
[98] In [99]: # 10 ближайших соседа
cl1_2 = KNeighborsClassifier(n_neighbors=10)
cl1_2.fit(wine_X_train, wine_y_train)
target1_2 = cl1_2.predict(wine_X_test)
len(target1_2, target1_2)

(89,
array([1, 1, 2, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1,
       2, 0, 2, 1, 0, 0, 0, 2, 1, 2, 1, 0, 2, 1, 1, 1, 1, 0, 0, 1, 1, 2,
       0, 2, 2, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 2, 2, 1, 0,
       1, 0, 1, 2, 1, 1, 1, 2, 1, 1, 2, 1, 0, 2, 1, 0, 1, 0, 1, 1, 1, 0,
       1]))
```

Метрики качества классификации

```
[99] In [100]: accuracy_score(wine_y_test, target1_1)

0.6404494382022472
```

```
[100] In [101]: accuracy_score(wine_y_test, target1_2)

0.6629213483146067
```

Произведите подбор гиперпараметра K с использованием GridSearchCV

```
[101] In [101]: print(classification_report(wine_y_test, target1_2))
```

	precision	recall	f1-score	support
0	0.93	0.79	0.85	33
1	0.61	0.82	0.70	34
2	0.33	0.23	0.27	22
accuracy			0.66	89
macro avg	0.62	0.61	0.61	89
weighted avg	0.66	0.66	0.65	89

```
[102] In [102]: def print_gridResults(grid):
                print(f'Подобранный параметр: {grid.best_params_}')
                print(f'Оценка при подобранном параметре: {grid.best_score_}')
                return [grid.best_params_, grid.best_score_]

[103] In [103]: param_grid = {'n_neighbors' : np.arange(1, 25)}
                grid = GridSearchCV(KNeighborsClassifier(), param_grid, cv=5)
                grid.fit(wine_X_train, wine_y_train)
                print_gridResults(grid)

Подобранный параметр: {'n_neighbors': 17}
Оценка при подобранном параметре: 0.7405228758169934

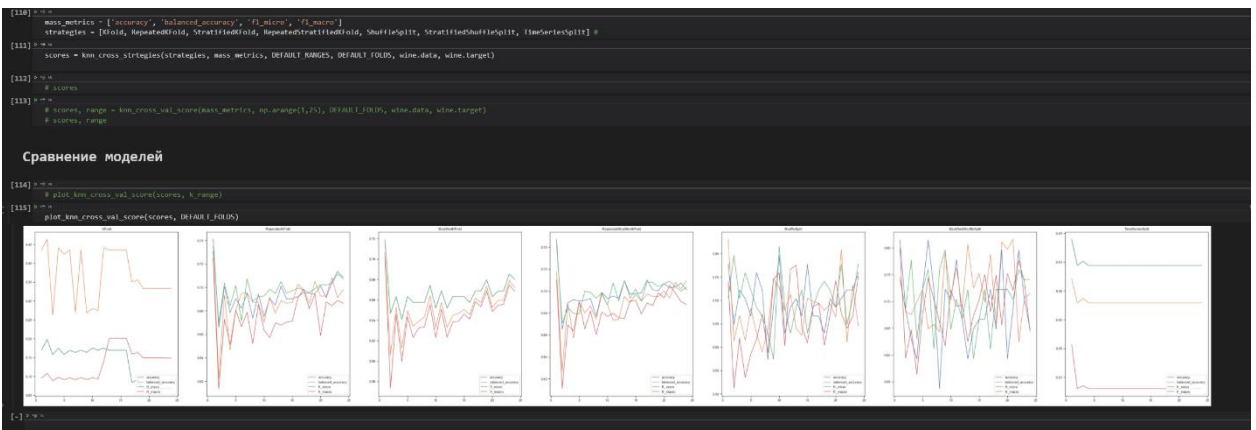
[{'n_neighbors': 17}, 0.7405228758169934]
```

Произведите подбор гиперпараметра K с использованием RandomizedSearchCV

```
[104] In [104]: param_grid = {'n_neighbors' : np.arange(1, 25)}
                grid2 = RandomizedSearchCV(KNeighborsClassifier(), param_grid, cv=5, n_iter=2)
                grid2.fit(wine_X_train, wine_y_train)
                print_gridResults(grid2)

Подобранный параметр: {'n_neighbors': 15}
Оценка при подобранном параметре: 0.7189542483660132

[{'n_neighbors': 15}, 0.7189542483660132]
```



5 Вывод

В данной лабораторной работе я научился разделять данные на обучающую и тестовую. После разделения построил модель на основе алгоритма ближайших соседей. Познакомился с кросс-валидацией и подбором гиперпараметров.