

МГТУ им. Н. Э. Баумана, кафедра ИУ5
курс “Технологии машинного обучения”

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

ВЫПОЛНИЛ:

Ерохин И.А.

Группа: ИУ5-61Б

ПРОВЕРИЛ:

Гапанюк Ю.Е.

Москва 2020

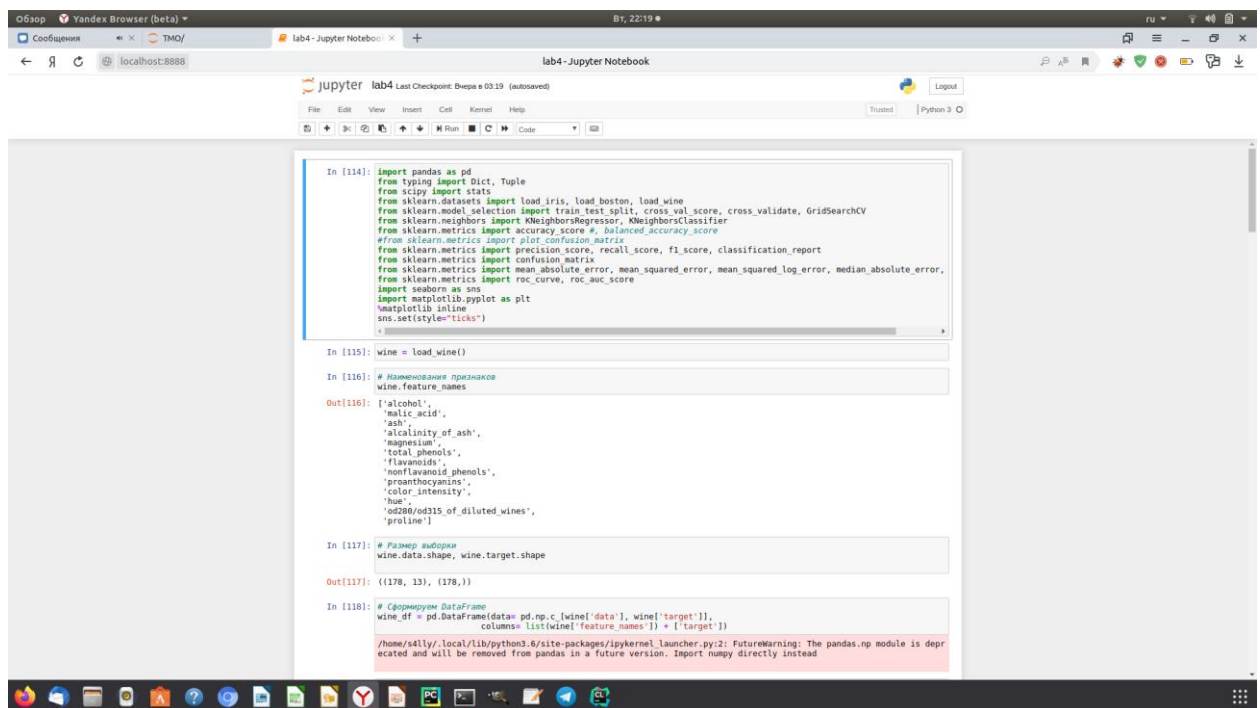
Цель лабораторной работы: изучение сложных способов подготовки выборки и подбора гиперпараметров на примере метода ближайших соседей.

Задание:

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. Постройте модель и оцените качество модели с использованием кросс-валидации.
5. Произведите подбор гиперпараметра K с использованием `GridSearchCV` и кросс-валидации.

Выполненная работа:

Загрузка и первичный анализ данных



```
In [114]: import pandas as pd
from typing import Dict, Tuple
from scipy import stats
from sklearn.datasets import load_iris, load_boston, load_wine
from sklearn.model_selection import train_test_split, cross_val_score, cross_validate, GridSearchCV
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import precision_score, recall_score, f1_score, classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import mean_absolute_error, mean_squared_error, mean_squared_log_error, median_absolute_error
from sklearn.metrics import roc_curve, roc_auc_score
import seaborn as sns
import matplotlib.pyplot as plt
sns.set(style="ticks")

In [115]: wine = load_wine()

In [116]: # Наименование признаков
wine.feature_names

Out[116]: ['alcohol',
'malic_acid',
'ash',
'alcalinity_of_ash',
'magnesium',
'total_phenols',
'flavonoids',
'nonflavonoid_phenols',
'proanthocyanins',
'color_intensity',
'hue',
'od280/od315_of_diluted_wines',
'proline']

In [117]: # Размер выборки
wine.data.shape, wine.target.shape

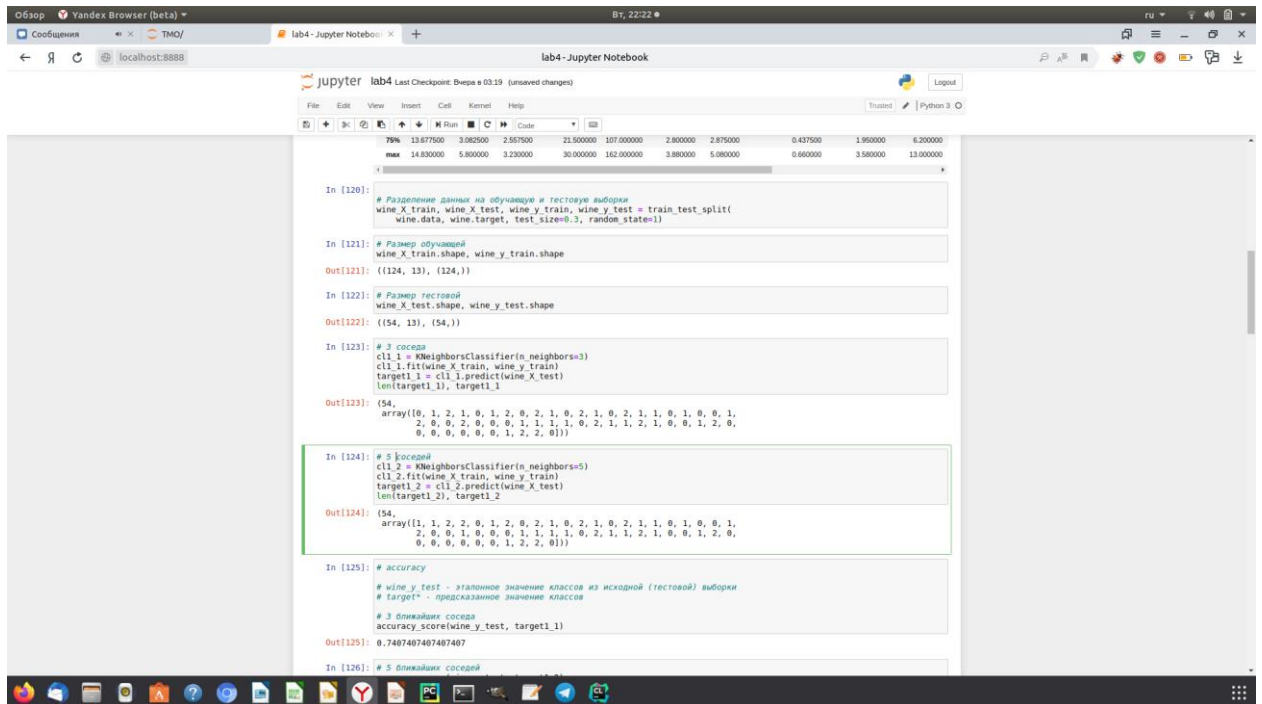
Out[117]: ((178, 13), (178,))

In [118]: # Создать pandas DataFrame
wine_df = pd.DataFrame(data=pd.c_[wine['data'], wine['target']],
                      columns=list(wine['feature_names']) + ['target'])

/home/s41ly/.local/lib/python3.6/site-packages/ipykernel_launcher.py:2: FutureWarning: The pandas.np module is deprecated and will be removed from pandas in a future version. Import numpy directly instead
```

Разделение данных на обучающую и тестовую выборки и построение базовых моделей на основе метода ближайших соседей и подсчет метрик качества классификации

1) Accuracy Precision, recall и F-мера



The screenshot shows a Jupyter Notebook interface in a browser. The notebook is titled 'lab4 - Jupyter Notebook' and contains several code cells. The first cell (In [120]) splits the 'wine' dataset into training and testing sets using `train_test_split`. The second cell (In [121]) prints the shapes of the training and testing data. The third cell (In [122]) prints the shapes of the training and testing targets. The fourth cell (In [123]) creates a `KNeighborsClassifier` with `neighbors=3` and predicts the target for the training set. The fifth cell (In [124]) creates a `KNeighborsClassifier` with `neighbors=5` and predicts the target for the testing set. The sixth cell (In [125]) calculates the accuracy score for the testing set using `accuracy_score`. The seventh cell (In [126]) prints the accuracy score.

```
In [120]: # Разделение данных на обучающую и тестовую выборки
wine_X_train, wine_X_test, wine_y_train, wine_y_test = train_test_split(
    wine.data, wine.target, test_size=0.3, random_state=1)

In [121]: # Размер обучающей
wine_X_train.shape, wine_y_train.shape
Out[121]: ((124, 13), (124,))

In [122]: # Размер тестовой
wine_X_test.shape, wine_y_test.shape
Out[122]: ((54, 13), (54,))

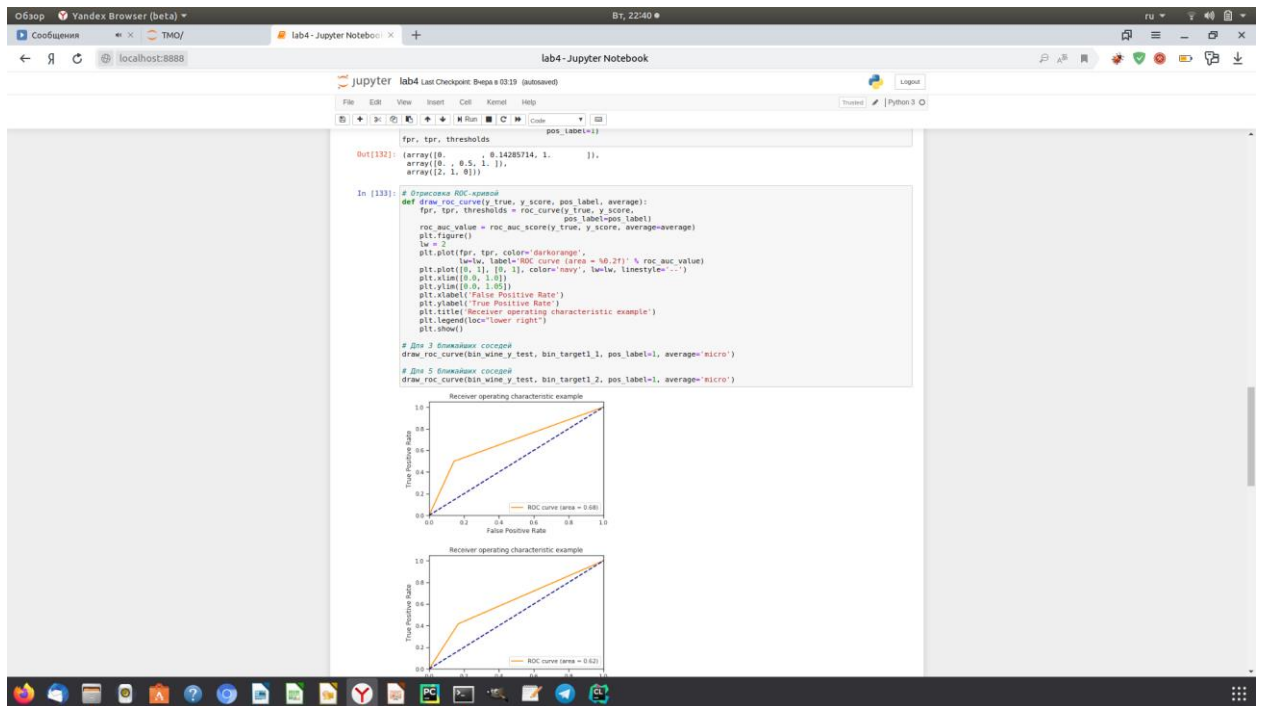
In [123]: # 3 соседа
cl1 = KNeighborsClassifier(n_neighbors=3)
cl1.fit(wine_X_train, wine_y_train)
target1_1 = cl1.predict(wine_X_test)
len(target1_1)
Out[123]: (54,
array([0, 1, 2, 1, 0, 1, 2, 0, 2, 1, 0, 2, 1, 0, 2, 1, 0, 1, 0, 0, 1,
       2, 0, 0, 2, 0, 0, 0, 1, 1, 1, 1, 0, 2, 1, 1, 2, 1, 0, 0, 1, 2, 0,
       0, 0, 0, 0, 0, 1, 2, 2, 0]))

In [124]: # 5 соседей
cl1_2 = KNeighborsClassifier(n_neighbors=5)
cl1_2.fit(wine_X_train, wine_y_train)
target1_2 = cl1_2.predict(wine_X_test)
len(target1_2), target1_2
Out[124]: (54,
array([1, 1, 2, 2, 0, 1, 2, 0, 2, 1, 0, 2, 1, 0, 2, 1, 1, 0, 1, 0, 0, 1,
       2, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 2, 1, 1, 2, 1, 0, 0, 1, 2, 0,
       0, 0, 0, 0, 0, 1, 2, 2, 0]))

In [125]: # accuracy
# wine_y_test - эталонное значение классов из исходной (тестовой) выборки
# target* - предсказанное значение классов
# 3 ближайших соседа
accuracy_score(wine_y_test, target1_1)
Out[125]: 0.7407407407407407

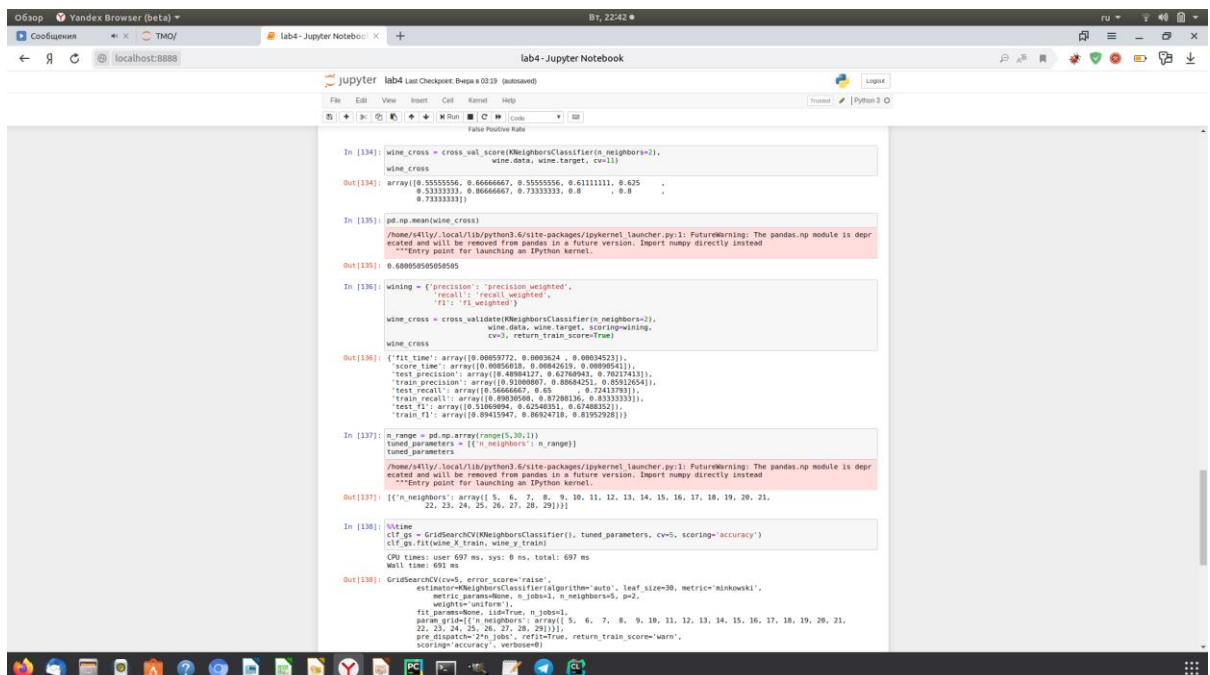
In [126]: # 5 ближайших соседей
```

2) ROC-кривая и ROC AUC

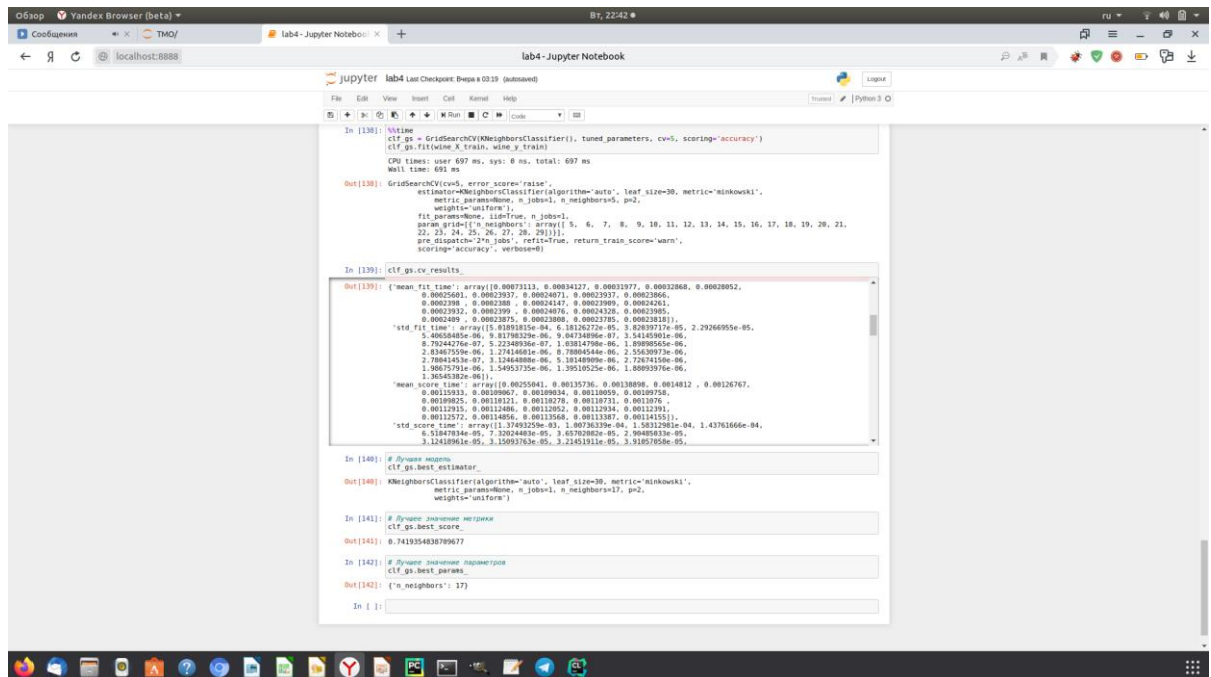


Проанализировав результаты полученных метрик качества классификации, можно судить о среднем качестве классификации.

Разбиение выборки на k частей с помощью кросс-валидации. Наиболее простым способом кросс-валидации является вызов функции `cross_val_score`. В этом случае стратегия кросс-валидации определяется автоматически



Нахождение наилучшего гиперпараметра К с использованием GridSearchCV и кросс-валидации



Как видно, лучшее найденное значение гиперпараметра = 17. При этом гиперпараметре получено наилучшее значение метрики = 0.74