Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

Отчёт по лабораторной работе №3

По дисциплине:
«Технологии машинного обучения»

Выполнил:

Студент группы ИУ5-61Б _____ **Пахомкин К.С.**

(Подпись, дата) (Фамилия И.О.)

Проверил:

_____ **Гапанюк Ю. Е.**

(Подпись, дата) (Фамилия И.О.)

Москва, 2021

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. 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.

ЛР3

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
```

In [2]:
```
import numpy as np
import pandas as pd
from typing import Dict, Tuple
from scipy import stats
from sklearn.datasets import load_iris, load_boston
from sklearn.model_selection import train_test_split
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, classi
from sklearn.metrics import confusion_matrix
from sklearn.metrics import mean_absolute_error, mean_squared_error, mean_s
from sklearn.metrics import roc_curve, roc_auc_score
import seaborn as sns
from sklearn.model_selection import learning_curve
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

In [3]:
```
from sklearn.model_selection import KFold, RepeatedKFold, LeaveOneOut, Leav
```

In [4]:
```
from sklearn.model_selection import train_test_split
```

In [5]:
```
# чтение обучающей выборки
data = pd.read_csv('train.csv')
```

In [6]:
```
# уберем непонятный для нас параметр, чтобы он не помешал в будущем
data.drop(['Name','Sex','Ticket','Embarked','Cabin','PassengerId', 'Parch'
```

In [7]:
```
data
```

Out[7]:

	Survived	Pclass	Age	SibSp	Fare
0	0	3	22.0	1	7.2500
1	1	1	38.0	1	71.2833
2	1	3	26.0	0	7.9250
3	1	1	35.0	1	53.1000
4	0	3	35.0	0	8.0500
...	...	...	...	...	...
886	0	2	27.0	0	13.0000
887	1	1	19.0	0	30.0000
888	0	3	NaN	1	23.4500
889	1	1	26.0	0	30.0000
890	0	3	32.0	0	7.7500

891 rows × 5 columns

In [8]:
```
# заполняем пустые значения
data = data.fillna(1)
data.head()
```

Out[8]:

	Survived	Pclass	Age	SibSp	Fare
0	0	3	22.0	1	7.2500
1	1	1	38.0	1	71.2833
2	1	3	26.0	0	7.9250
3	1	1	35.0	1	53.1000
4	0	3	35.0	0	8.0500

In [9]:
```
# делим на 2 части
parts = np.split(data, [4,5], axis=1)
X = parts[0]
Y = parts[1]
print('Входные данные:\n\n', X.head(), '\n\nВыходные данные:\n\n', Y.head(
```

Входные данные:

	Survived	Pclass	Age	SibSp
0	0.0	3.0	22.0	1.0
1	1.0	1.0	38.0	1.0
2	1.0	3.0	26.0	0.0
3	1.0	1.0	35.0	1.0
4	0.0	3.0	35.0	0.0

Выходные данные:

	Fare
0	7.2500
1	71.2833
2	7.9250
3	53.1000
4	8.0500

Разделение выборки

In [10]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.
```

In [11]:
```
print('Входные параметры обучающей выборки:\n\n',X_train.head(), \
      '\n\nВходные параметры тестовой выборки:\n\n', X_test.head(), \
      '\n\nВыходные параметры обучающей выборки:\n\n', Y_train.head(), \
      '\n\nВыходные параметры тестовой выборки:\n\n', Y_test.head())
```

Входные параметры обучающей выборки:

	Survived	Pclass	Age	SibSp
598	0.0	3.0	1.0	0.0
73	0.0	3.0	26.0	1.0
168	0.0	1.0	1.0	0.0
466	0.0	2.0	1.0	0.0
283	1.0	3.0	19.0	0.0

Входные параметры тестовой выборки:

	Survived	Pclass	Age	SibSp
640	0.0	3.0	20.0	0.0
486	1.0	1.0	35.0	1.0
514	0.0	3.0	24.0	0.0
54	0.0	1.0	65.0	0.0
717	1.0	2.0	27.0	0.0

Выходные параметры обучающей выборки:

	Fare
598	7.2250
73	14.4542
168	25.9250
466	0.0000
283	8.0500

Выходные параметры тестовой выборки:

	Fare
640	7.8542
486	90.0000
514	7.4958
54	61.9792
717	10.5000

In [12]:
```
# Проверим правильность разделения выборки на тестовую и обучающую. Посмот
print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(864, 4)
(27, 4)
(864, 1)
(27, 1)
```

Модель ближайших соседей для произвольного гиперпараметра K

In [13]:
```
from sklearn.neighbors import KNeighborsRegressor
```

In [14]:
```
# Решение задачи регрессии методом 2, 5 и 10 ближайших соседей
Regressor_2NN = KNeighborsRegressor(n_neighbors = 2)
Regressor_5NN = KNeighborsRegressor(n_neighbors = 5)
Regressor_10NN = KNeighborsRegressor(n_neighbors = 10)
print('Пример модели:\n\n', Regressor_10NN)
```

```
Пример модели:

 KNeighborsRegressor(n_neighbors=10)
```

In [15]:
```
Regressor_2NN.fit(X_train, Y_train)
Regressor_5NN.fit(X_train, Y_train)
Regressor_10NN.fit(X_train, Y_train)
target_2NN = Regressor_2NN.predict(X_test)
target_5NN = Regressor_5NN.predict(X_test)
target_10NN = Regressor_10NN.predict(X_test)
print('Пример предсказанных значений:\n\n', target_10NN[:5], '\n ...')
```

```
Пример предсказанных значений:

[[  8.07875]
[164.17042]
[  8.5275 ]
[ 55.44458]
[ 13.83249]]
...
```

Оценка качества регрессии (Метрики качества)

In [16]:
```
 from sklearn.metrics import mean_absolute_error, mean_squared_error, media
```

In [17]:
```
# Оценка средней абсолютной ошибки
print('Средняя абсолютная ошибка для 2 ближайших соседей:',mean_absolute_e
target_2NN))
print('Средняя абсолютная ошибка для 5 ближайших соседей:',mean_absolute_e
target_5NN))
print('Средняя абсолютная ошибка для 10 ближайших соседей:',mean_absolute_
target_10NN))
```

```
Средняя абсолютная ошибка для 2 ближайших соседей: 18.33000555555556
Средняя абсолютная ошибка для 5 ближайших соседей: 13.957492592592594
Средняя абсолютная ошибка для 10 ближайших соседей: 19.786847777777776
```

In [18]:
```
# Оценка средней квадратичной ошибки
print('Средняя квадратичная ошибка для 2 ближайших соседей:',mean_squared_
target_2NN))
print('Средняя квадратичная ошибка для 5 ближайших соседей:',mean_squared_
target_5NN))
print('Средняя квадратичная ошибка для 10 ближайших соседей:',mean_squared
, target_10NN))
```

```
Средняя квадратичная ошибка для 2 ближайших соседей: 1079.8197753593522
Средняя квадратичная ошибка для 5 ближайших соседей: 970.3768036050817
Средняя квадратичная ошибка для 10 ближайших соседей: 1417.8809758430853
```

In [19]:
```
# Оценка коэффициента детерминации

print('Коэффициент детерминации для 2 ближайших соседей:',r2_score(Y_test,
print('Коэффициент детерминации для 5 ближайших соседей:',r2_score(Y_test,
print('Коэффициент детерминации для 10 ближайших соседей:',r2_score(Y_test
))
```

```
Коэффициент детерминации для 2 ближайших соседей: 0.5849586205961371
Коэффициент детерминации для 5 ближайших соседей: 0.6270243087780691
Коэффициент детерминации для 10 ближайших соседей: 0.45502083822407324
```

In [20]:
```
## Grid Search (решетчатый поиск)
```

In [21]:
```
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
```

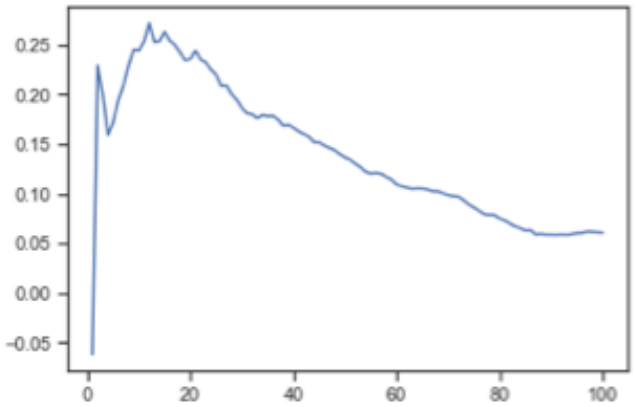
Подбор гиперпараметров

GridSearch через коэффициент детерминации

In [25]:
```
gs_det = GridSearchCV(KNeighborsRegressor(), tuned_parameters, cv=10, scori
gs_det.fit(X_train, Y_train)
print('Лучшая модель:\n\n', gs_det.best_estimator_)
print('\nЛучшее число ближайших соседей:\n\n',gs_det.best_params_)
print('\nЛучшее значение коэффициента детерминации:\n\n',gs_det.best_score_
print('\nИзменение качества тестовой выборки в зависимости от кол-ва сосед
plt.plot(n_range, gs_det.cv_results_['mean_test_score'])
```

```
Лучшая модель:

 KNeighborsRegressor(n_neighbors=12)

Лучшее число ближайших соседей:

 {'n_neighbors': 12}

Лучшее значение коэффициента детерминации:

 0.2721445327911323

Изменение качества тестовой выборки в зависимости от кол-ва соседей:
```

Out[25]:
```
[<matplotlib.lines.Line2D at 0x7fd876b403d0>]
```



Кросс-валидация

In [26]:
```
from sklearn.model_selection import cross_val_score
scores_2NN = cross_val_score(KNeighborsRegressor(n_neighbors = 2), X, Y, cv
scores_5NN = cross_val_score(KNeighborsRegressor(n_neighbors = 5), X, Y, cv
scores_10NN = cross_val_score(KNeighborsRegressor(n_neighbors = 10), X, Y,
scores_50NN = cross_val_score(KNeighborsRegressor(n_neighbors = 50), X, Y,
scores_100NN = cross_val_score(KNeighborsRegressor(n_neighbors = 100), X, Y
print('Пример значений коэф. детерминации для 5 фолдов для метода 10 ближа
print('Усредненное значение коэффициента детерминации для:\n')
print('- 2 ближайших соседей:', np.mean(scores_2NN), '\n')
print('- 5 ближайших соседей:', np.mean(scores_5NN), '\n')
print('- 10 ближайших соседей:', np.mean(scores_10NN), '\n')
print('- 50 ближайших соседей:', np.mean(scores_50NN), '\n')
print('- 100 ближайших соседей:', np.mean(scores_100NN), '\n')
```

Пример значений коэф. детерминации для 5 фолдов для метода 10 ближайших сосе
дей:
 [0.34166201 0.38655715 0.14117213 0.28452217 0.2883947]

Усредненное значение коэффициента детерминации для:

- 2 ближайших соседей: 0.18694561138232885

- 5 ближайших соседей: 0.23548126907370337

- 10 ближайших соседей: 0.28846163209364245

- 50 ближайших соседей: 0.13534843218545478

- 100 ближайших соседей: 0.05880772437701802