

**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 №4

Выполнила:
студентка группы ИУ5-64
Бредня Елизавета

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. В случае необходимости проведите удаление или заполнение пропусков и кодирование категориальных признаков.
3. С использованием метода `train_test_split` разделите выборку на обучающую и тестовую.
4. Обучите следующие модели:
 - одну из линейных моделей (линейную или полиномиальную регрессию при решении задачи регрессии, логистическую регрессию при решении задачи классификации);
 - SVM;
 - дерево решений.
5. Оцените качество моделей с помощью двух подходящих для задачи метрик. Сравните качество полученных моделей.
6. Постройте график, показывающий важность признаков в дереве решений.
7. Визуализируйте дерево решений или выведите правила дерева решений в текстовом виде.

Текст программы и её результаты

```
[ ] from sklearn.datasets import fetch_california_housing
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
import warnings
%matplotlib inline

data = fetch_california_housing(as_frame=True)['data']
data["price"] = fetch_california_housing(as_frame=True)['target']
data.head()
```

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	price
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422

Пропусков нет

```
[ ] data.isna().sum()
```

```
MedInc      0
HouseAge    0
AveRooms    0
AveBedrms   0
Population  0
AveOccup    0
Latitude    0
Longitude   0
price       0
dtype: int64
```

```
[ ] X_train, X_test, y_train, y_test = train_test_split(data.drop('price', axis=1), data['price'], test_size=0.3, random_state=42)
```

```
➤ model1 = LinearRegression().fit(X_train, y_train)
model2 = SVR(kernel='rbf', gamma=0.8, C=1.0).fit(X_train, y_train)
model3 = DecisionTreeRegressor(max_depth=3).fit(X_train, y_train)
```

```
➤ for i in [model1, model2, model3]:
    y_test_predict = i.predict(X_test)
    mse = mean_squared_error(y_test, y_test_predict)
    mae = mean_absolute_error(y_test, y_test_predict)
    print(i)
    print('MSE: {}'.format(mse))
    print('MAE: {}'.format(mae))
```

```

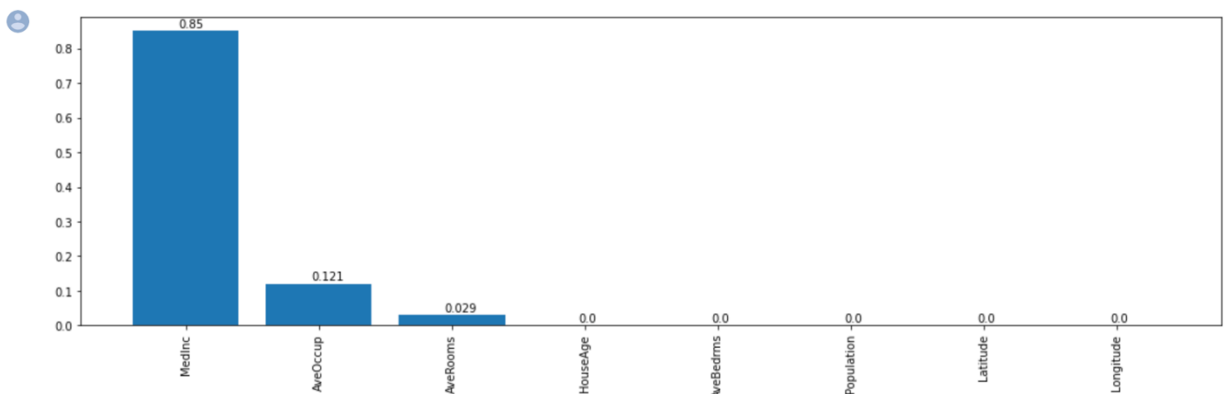
LinearRegression()
MSE: 0.5305677824766758
MAE: 0.5272474538305936
SVR(gamma=0.8)
MSE: 1.3288203961412937
MAE: 0.8763638769611481
DecisionTreeRegressor(max_depth=3)
MSE: 0.6325363214073455
MAE: 0.5987305284617841

```

```

▶ from operator import itemgetter
list_to_sort = list(zip(data.columns.values, model3.feature_importances_))
sorted_list = sorted(list_to_sort, key=itemgetter(1), reverse = True)
# Названия признаков
labels = [x for x, _ in sorted_list]
# Важности признаков
data = [x for _, x in sorted_list]
# Вывод графика
fig, ax = plt.subplots(figsize=(18,5))
ind = np.arange(len(labels))
plt.bar(ind, data)
plt.xticks(ind, labels, rotation='vertical')
# Вывод значений
for a,b in zip(ind, data):
    plt.text(a-0.05, b+0.01, str(round(b,3)))
plt.show()

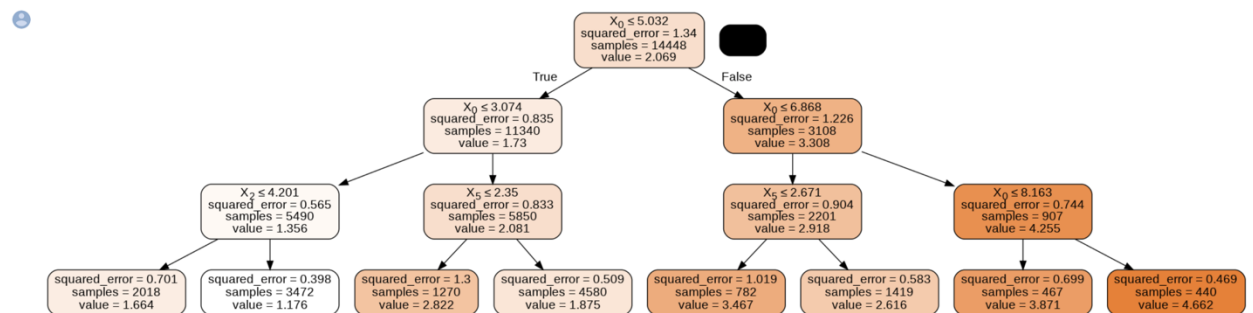
```



```

[ ] from six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dot_data = StringIO()
export_graphviz(model3, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())

```



Дерево решений