

**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

Выполнил:
студент группы ИУ5-64
Кузьмин Роман

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 load_boston
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np
import matplotlib.pyplot as plt
import warnings
%matplotlib inline

warnings.filterwarnings("ignore")
boston = pd.DataFrame(load_boston().data, columns=load_boston().feature_names)
target = pd.DataFrame(load_boston().target, columns=["MEDV"])
boston.head()
```

| | CRIM | ZN | INDUS | CHAS | NOX | RM | AGE | DIS | RAD | TAX | PTRATIO | B | LSTAT |
|---|---------|------|-------|------|-------|-------|------|--------|-----|-------|---------|--------|-------|
| 0 | 0.00632 | 18.0 | 2.31 | 0.0 | 0.538 | 6.575 | 65.2 | 4.0900 | 1.0 | 296.0 | 15.3 | 396.90 | 4.98 |
| 1 | 0.02731 | 0.0 | 7.07 | 0.0 | 0.469 | 6.421 | 78.9 | 4.9671 | 2.0 | 242.0 | 17.8 | 396.90 | 9.14 |
| 2 | 0.02729 | 0.0 | 7.07 | 0.0 | 0.469 | 7.185 | 61.1 | 4.9671 | 2.0 | 242.0 | 17.8 | 392.83 | 4.03 |
| 3 | 0.03237 | 0.0 | 2.18 | 0.0 | 0.458 | 6.998 | 45.8 | 6.0622 | 3.0 | 222.0 | 18.7 | 394.63 | 2.94 |
| 4 | 0.06905 | 0.0 | 2.18 | 0.0 | 0.458 | 7.147 | 54.2 | 6.0622 | 3.0 | 222.0 | 18.7 | 396.90 | 5.33 |

Пропуски отсутствуют:

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

```
CRIM      0
ZN        0
INDUS     0
CHAS      0
NOX       0
RM        0
AGE       0
DIS       0
RAD       0
TAX       0
PTRATIO   0
B         0
LSTAT     0
dtype: int64
```

```
[ ] X_train, X_test, y_train, y_test = train_test_split(boston, target, 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)
```

```

▶ y_test_predict = model1.predict(X_test)
  rmse = (np.sqrt(mean_squared_error(y_test, y_test_predict)))
  r2 = r2_score(y_test, y_test_predict)

  print("-----\nmodel1\n-----")
  print('RMSE: {}'.format(rmse))
  print('R2 score: {}'.format(r2))

  y_test_predict = model2.predict(X_test)
  rmse = (np.sqrt(mean_squared_error(y_test, y_test_predict)))
  r2 = r2_score(y_test, y_test_predict)

  print("-----\nmodel2\n-----")
  print('RMSE: {}'.format(rmse))
  print('R2 score: {}'.format(r2))

  y_test_predict = model3.predict(X_test)
  rmse = (np.sqrt(mean_squared_error(y_test, y_test_predict)))
  r2 = r2_score(y_test, y_test_predict)

  print("-----\nmodel3\n-----")
  print('RMSE: {}'.format(rmse))
  print('R2 score: {}'.format(r2))

```

```

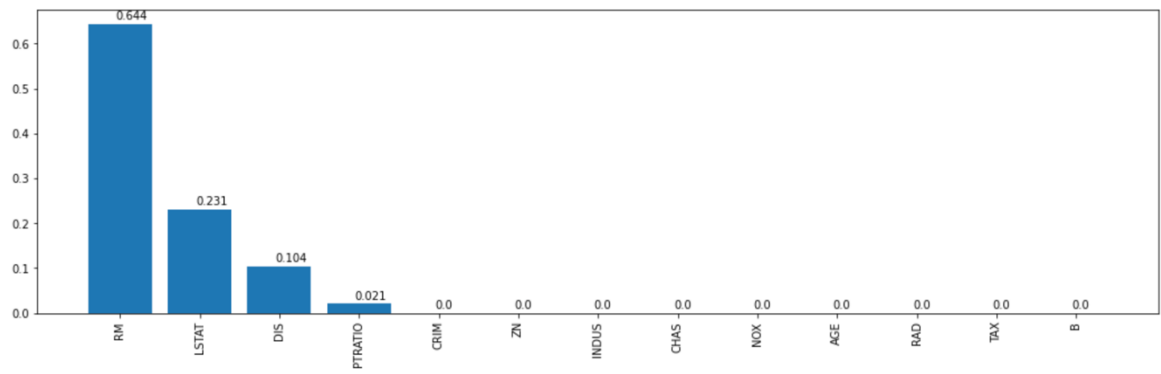
-----
model1
-----
RMSE: 4.638689926172788
R2 score: 0.7112260057484974
-----
model2
-----
RMSE: 8.63201641777932
R2 score: 1.8629356956956222e-05
-----
model3
-----
RMSE: 4.371190517416146
R2 score: 0.7435711616336518

```

```

▶ from operator import itemgetter
  list_to_sort = list(zip(boston.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())

```

