Московский государственный технический

университет им. Н. Э. Баумана

Факультет «Радиотехнический»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Отчет по лабораторной работе №4
«Линейные модели, SVM и деревья решений»

Выполнил:
студент группы РТ5-61Б

Алиев Тимур

Подпись и дата:

Проверил:
доцент 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.

Jupyter notebook

Лабораторная работа №4. Алиев Тимур РТ5-61Б.

Подготовка данных

Ввод [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import PolynomialFeatures, MinMaxScaler, StandardScaler
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.tree import DecisionTreeRegressor, export_graphviz, export_text
from sklearn.svm import SVR
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
from sklearn.model_selection import train_test_split, GridSearchCV
from IPython.display import Image
from IPython.core.display import HTML
```

Ввод [2]:
```
data = pd.read_csv('WineQT.csv')
data.head()
```

Out[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	0
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5	1
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5	2
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6	3
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	4

Ввод [3]:
```
# Корреляция с алкоголем по модулю - top
best_params = data.corr()['alcohol'].map(abs).sort_values(ascending=False)[1:]
best_params = best_params[best_params.values > 0.35]
best_params
```

Out[3]:
```
density    0.494727
quality    0.484866
Name: alcohol, dtype: float64
```

Разделение выборки на обучающую и тестовую

Ввод [4]:
```
x_train, x_test, y_train, y_test = train_test_split(data[best_params.index], data['alcohol'], test_size=0.3, random_state=3)
```

Обучение моделей

1) Линейная регрессия

Ввод [5]:
```
def print_metrics(y_test, y_pred):
    print(f"R^2: {r2_score(y_test, y_pred)}")
    print(f"MSE: {mean_squared_error(y_test, y_pred)}")
    print(f"MAE: {mean_absolute_error(y_test, y_pred)}")

linear_model = LinearRegression()
linear_model.fit(x_train, y_train)
y_pred_linear = linear_model.predict(x_test)
print_metrics(y_test, y_pred_linear)
```

```
R^2: 0.4537565878619284
MSE: 0.6403004607616753
MAE: 0.6514450393971108
```

2) Пополиномиальная регрессия

Ввод [6]:
```
poly_model = PolynomialFeatures(degree=3)
x_train_poly = poly_model.fit_transform(x_train)
x_test_poly = poly_model.fit_transform(x_test)
linear_model = LinearRegression()
linear_model.fit(x_train_poly, y_train)
y_pred_poly = linear_model.predict(x_test_poly)
print_metrics(y_test, y_pred_poly)
```

```
R^2: 0.5045011976737863
MSE: 0.5808181927439674
MAE: 0.5979541713845272
```

3) SVM

Ввод [7]:
```
scaler = StandardScaler().fit(x_train)
x_train_scaled = pd.DataFrame(scaler.transform(x_train), columns=x_train.columns)
x_test_scaled = pd.DataFrame(scaler.transform(x_test), columns=x_train.columns)
x_train_scaled.describe()
```

Out[7]:

	density	quality
count	8.000000e+02	8.000000e+02
mean	-7.923662e-15	-5.240253e-16
std	1.000626e+00	1.000626e+00
min	-3.396295e+00	-3.330798e+00
25%	-6.281010e-01	-8.038767e-01
50%	-1.634535e-02	4.595838e-01
75%	5.750184e-01	4.595838e-01
max	3.271841e+00	2.986505e+00

Ввод [8]:
```
params = {'C': np.concatenate([np.arange(0.1, 2, 0.1), np.arange(2, 15, 1)])}
svm_model = SVR(kernel='linear')
grid_cv = GridSearchCV(estimator=svm_model, param_grid=params, cv=10, n_jobs=-1, scoring='r2')
grid_cv.fit(x_train_scaled, y_train)
print(grid_cv.best_params_)
```

```
{'C': 1.9000000000000001}
```

Ввод [9]:
```
best_svm_model = grid_cv.best_estimator_
best_svm_model = SVR(kernel='linear', C=11)
best_svm_model.fit(x_train_scaled, y_train)
y_pred_svm = best_svm_model.predict(x_test_scaled)
print_metrics(y_test, y_pred_svm)
```

```
R^2: 0.44738317687446305
MSE: 0.647771302333795
MAE: 0.6408125766617059
```

4) Дерево решений

Ввод [10]:
```
params = {'min_samples_leaf': range(3, 30)}
tree = DecisionTreeRegressor(random_state=3)
grid_cv = GridSearchCV(estimator=tree, cv=5, param_grid=params, n_jobs=-1, scoring='neg_mean_absolute_error')
grid_cv.fit(x_train, y_train)
print(grid_cv.best_params_)
```

```
{'min_samples_leaf': 28}
```

Ввод [11]:
```
best_tree = grid_cv.best_estimator_
best_tree.fit(x_train, y_train)
y_pred_tree = best_tree.predict(x_test)
print_metrics(y_test, y_pred_tree)
```

```
R^2: 0.48707752514637803
MSE: 0.6012420281615745
MAE: 0.614506439367307
```

4) Дерево решений

Ввод [10]:
```
params = {'min_samples_leaf': range(3, 30)}
tree = DecisionTreeRegressor(random_state=3)
grid_cv = GridSearchCV(estimator=tree, cv=5, param_grid=params, n_jobs=-1, scoring='neg_mean_absolute_error')
grid_cv.fit(x_train, y_train)
print(grid_cv.best_params_)
```

```
{'min_samples_leaf': 28}
```

Ввод [11]:
```
best_tree = grid_cv.best_estimator_
best_tree.fit(x_train, y_train)
y_pred_tree = best_tree.predict(x_test)
print_metrics(y_test, y_pred_tree)
```

```
R^2: 0.48707752514637803
MSE: 0.6012420281615745
MAE: 0.614506439367307
```

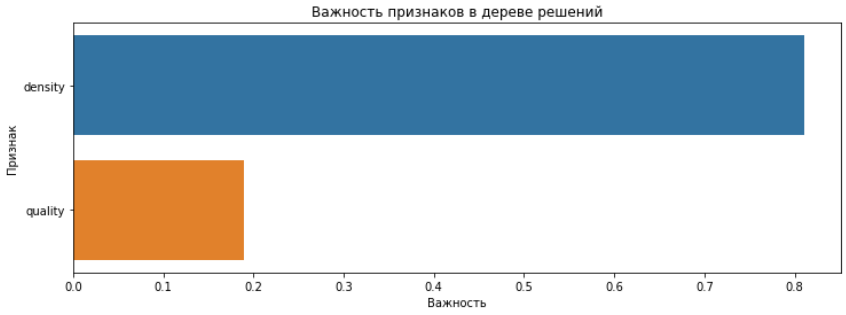
Важность признаков в дереве решений

Ввод [12]:
```
importances = pd.DataFrame(data=zip(x_train.columns, best_tree.feature_importances_), columns=['Признак', 'Важность'])
print('Важность признаков в дереве решений\n')
for row in importances.sort_values(by='Важность', ascending=False).values:
    print(f'{row[0]}: {round(row[1], 3)}')
```

```
Важность признаков в дереве решений

density: 0.81
quality: 0.19
```

Ввод [13]:
```
plt.figure(figsize=(12, 4))
sns.barplot(data=importances.sort_values(by='Важность', ascending=False), y='Признак', x='Важность', orient='h', )
plt.title('Важность признаков в дереве решений')
plt.show()
```



Сравнение моделей

Ввод [14]:
```
print('Линейная регрессия')
print_metrics(y_test, y_pred_linear)

print('\nПолиномиальная регрессия')
print_metrics(y_test, y_pred_poly)

print('\nМетод опорных векторов')
print_metrics(y_test, y_pred_svm)

print('\nДерево решений')
print_metrics(y_test, y_pred_tree)
```

```
Линейная регрессия
R^2: 0.4537565878619284
MSE: 0.6403004607616753
MAE: 0.6514450393971108

Полиномиальная регрессия
R^2: 0.5045011976737863
MSE: 0.5808181927439674
MAE: 0.5979541713845272

Метод опорных векторов
R^2: 0.44738317687446305
MSE: 0.647771302333795
MAE: 0.6408125766617059

Дерево решений
R^2: 0.48707752514637803
MSE: 0.6012420281615745
MAE: 0.614506439367307
```