Министерство науки и высшего образования Российской Федерации
Муромский институт (филиал)
Федерального государственного бюджетного образовательного учреждения высшего образования
«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»

Факультет ИТР

Кафедра ПИн

ЛАБОРАТОРНАЯ РАБОТА №6

По Анализ данных

Тема Линейный регрессионный анализ на Python. Построение моделей для задачи регрессии. Линейная регрессия. Регуляризация

Руководитель

Белякова А.С.
(фамилия, инициалы)

(подпись) (дата)

Студент ПИН - 121
(группа)

Ермилов М.В.
(фамилия, инициалы)

(подпись) (дата)

Муром 2024

Лабораторная работа №6

Тема: линейный регрессионный анализ на Python. Построение моделей для задачи регрессии. Линейная регрессия. Регуляризация.

Цели и задачи: изучение способов построения регрессионных моделей.

Ход работы: провести анализ набора данных о заработной плате в области науки о данных, получить следующие данные:

Задание 1) Обучите модель гребневой регрессии Ridge. Выведите коэффициенты. Узнать зануляются ли какие-то? Вычислить MSE.

Листинг кода 1 – подключение библиотек и чтение данных из файла:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
plt.style.use("fivethirtyeight")
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import StandardScaler,MinMaxScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.linear_model import LassoCV, RidgeCV
df = pd.read_csv("D:/ucheba/python/sales.csv")
```

Листинг кода 2 – Обучение модели и вывод коэффициентов:

```
linreg = LinearRegression(fit_intercept = True)
X = df.drop(["ADV"],axis=1)
y = df["ADV"]
df.info()
X.shape, y.shape
X_scal = StandardScaler().fit_transform(X)
x_train, x_test, y_train, y_test = train_test_split(X_scal, y,
        test_size = 0.3, random_state=42)
linreg.fit(x_train, y_train)
y_pred = linreg.predict(X_scal)
y_pred.shape
df["ADV"][33]
y_pred[33]
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34 entries, 0 to 33
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   SALES   34 non-null     int64
 1   PRICE   34 non-null     int64
 2   ADV     34 non-null     int64
dtypes: int64(3)
memory usage: 948.0 bytes

544.6144501435472
```

Рисунок 1 – Вывод коэффициентов

Оценка качества модели при помощи MSE:

Листинг кода 3 – Вычисление MSE:

```
np.sqrt(mean_squared_error(y_test, linreg.predict(x_test)))
X.columns, linreg.coef_
linreg.intercept_
pd.DataFrame(linreg.coef_, X.columns, columns=["coef"]).sort_values(
    by="coef", ascending=False)
linreg.predict(X_scal)[0], df["ADV"][0]
```

```
(412.18215722931865, 200)
```

Рисунок 2 –Результат вычисления MSE

Задание 2) Найти оптимальное значение коэффициента для гребневой регрессии (L2-регуляризация).

Листинг кода 4 – Нахождение оптимального значения коэффициента для гребневой регрессии:

```
rf = RandomForestRegressor(n_estimators=100, max_depth = 5,
                           min_samples_leaf= 5, random_state=42)
rf.fit(x_train,y_train)
rf_pred = rf.predict(x_test)
np.sqrt(mean_squared_error(y_test, rf_pred))
pd.DataFrame(rf.feature_importances_, X.columns,
             columns=["rf_coef"]).sort_values(by="rf_coef", ascending=False)
```

	rf_coef
SALES	0.850907
PRICE	0.149093

Рисунок 3 – Оптимальное значение коэффициента для гребневой регрессии

Задание 3) Загрузите набор данных sales.csv. Рассчитайте коэффициенты множественной регрессии для расчета объема продаж (SALES) по объему рекламы (ADV) и цены батончика (PRICE). При помощи рассчитанных коэффициентов найдите объем продаж в магазине с рекламой 400 долларов в месяц и ценой батончика 79 центов. Деление на тренировочную и тестовую выборку делать не нужно. Масштабирование делать не нужно.

Листинг кода 5 – Нахождение объёма продаж в магазине:

```
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import StandardScaler,MinMaxScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.linear_model import LassoCV, RidgeCV
df = pd.read_csv('sales.csv')
X = df[['ADV', 'PRICE']]
y = df['SALES']
model = LinearRegression()
model.fit(X, y)
adv = 400
price = 79
sales_prediction = model.predict([[adv, price]])
sales_prediction
```

```
array([3078.57440476])
```

Рисунок 4 – Объём продаж в магазине

Задание 4) Загрузить набор данных sandler.csv - набор данных для предсказания дохода от показа фильмов. Применить простую линейную регрессию для предсказания дохода от показа фильмов от количества кинотеатров. Предскажите, чему будет равен доход для 1700 кинотеатров.

Листинг кода 6 – Нахождение предсказанного дохода с помощью простой линейной и множественной линейной регрессий:

```
from sklearn.linear_model import LinearRegression
# Загрузка данных
df = pd.read_csv('sandler.csv')
df['Opening Theaters']=df['Opening Theaters'].apply(lambda x: x.replace('
','',10).replace("$",'',10))
df['Theaters']=df['Theaters'].apply(lambda x: x.replace(' ','',10).replace("$",'',10))
df['Gross']=df['Gross'].apply(lambda x: x.replace(' ','',10).replace("$",'',10))
df['Opening Gross']=df['Opening Gross'].apply(lambda x: x.replace(' ','',10).replace("$",'',10))
df.drop(['Date','Title', 'Genre', 'Studio'], axis=1, inplace=True)
df['Opening Theaters'] = pd.to_numeric( df['Opening Theaters'])
df['Theaters'] = pd.to_numeric( df['Theaters'])
df['Gross'] = pd.to_numeric( df['Gross'])
df['Opening Gross'] = pd.to_numeric( df['Opening Gross'])
X1, y = df.drop(["Gross"], axis = 1), df['Gross']
# Простая линейная регрессия
X_simple = df[['Theaters']]
y_simple = df['Gross']
model_simple = LinearRegression()
```

```
model_simple.fit(X_simple, y_simple)
theaters = 1700
gross_prediction_simple = model_simple.predict([[theaters]])
print("Предсказанный доход для 1700 кинотеатров (простая линейная регрессия):",
gross_prediction_simple)
# Множественная линейная регрессия
X_multiple = df[['Theaters', 'Opening Theaters', 'Opening Gross']]
y_multiple = df['Gross']
model_multiple = LinearRegression()
model_multiple.fit(X_multiple, y_multiple)
theaters = 1700
opening_theaters = 1700
opening_gross = 5000000
gross_prediction_multiple = model_multiple.predict([[theaters, opening_theaters, opening_gross]])
print("Предсказанный доход для 1700 кинотеатров, 1700 открывающих кинотеатров и 5 миллионов дохода
от премьеры (множественная линейная регрессия):", gross_prediction_multiple)
```

```
Предсказанный доход для 1700 кинотеатров (простая линейная регрессия): [26984713.04266509]
Предсказанный доход для 1700 кинотеатров, 1700 открывающих кинотеатров и 5 миллионов дохода от премьеры (множественная линейная регрессия): [16489377.4788352]
```

Рисунок 5 – Предсказанный доход

Задание 5) Открыть набор House3.csv. Предсказать стоимость дома по его площади и наличию камина. Сделать предсказание цены дома, используя модель линейной регрессии для размера дома равного 2, с камином.

Листинг кода 7 – Нахождение предсказанной цены дома:

```
from sklearn.linear_model import LinearRegression
# Загружаем набор данных
data = pd.read_csv('house.csv')
# Закодируем наличие камина как 0/1
data['Kamin'] = data['Kamin'].astype('category').cat.codes
# Создаем модель линейной регрессии
model = LinearRegression()
# Обучаем модель на данных о площади и наличии камина
model.fit(data[['Area', 'Kamin']], data['Price'])
# Делаем предсказание цены дома для площади 2 и наличия камина
predicted_price = model.predict([[2, 1]])[0]
# Выводим предсказанную цену
print('Предсказанная цена дома:', predicted_price)
```

```
Предсказанная цена дома: 86.31513936898799
```

Рисунок 6 – Предсказанная цена дома

Вывод: в ходе работы изучили способы построения регрессионных моделей.