Министерство образования Республики Беларусь
Учреждение образования
«Брестский Государственный технический университет»
Кафедра ИИТ

Лабораторная работа №2
По дисциплине: «Основы машинного обучения»
Тема: «Линейные модели для задач регрессии и классификации»

Выполнил:
Студент 2 курса
Группы АС-66
Колбашко А. В.
Проверил:
Крощенко А. А.

Брест 2025

Цель работы: Изучить применение линейной и логистической регрессии для решения практических задач. Научиться обучать модели, оценивать их качество с помощью соответствующих метрик и интерпретировать результаты.

Ход работы

Общее задание: выполнить задания по варианту (регрессия и классификация), построить все требуемые визуализации и рассчитать метрики, **написать отчет, создать пул-реквест в репозиторий с кодом решения и отчетом в формате pdf**.

Вариант 10

- **Регрессия (Прогнозирование счастья в странах)**
  1. World Happiness Report
  2. Предсказать оценку счастья (Score)
  3. **Задания:**
     - загрузите данные. В качестве признаков используйте GDP per capita, Social support, Healthy life expectancy;
     - обучите модель **линейной регрессии**;
     - рассчитайте **MSE** и **R^2**;
     - визуализируйте зависимость Score от GDP per capita с линией регрессии.
- **Классификация (Прогнозирование оттока клиентов)**

1. Telco Customer Churn
2. Предсказать, уйдёт ли клиент (Churn = 'Yes')
3. **Задания:**
   - загрузите данные, обработайте категориальные признаки;
   - обучите модель **логистической регрессии**;
   - рассчитайте **Accuracy**, **Precision** и **Recall** для класса 'Yes';
   - постройте **матрицу ошибок**.

Код программы:

1)

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np

data = pd.read_csv("world_happiness_report.csv")
```

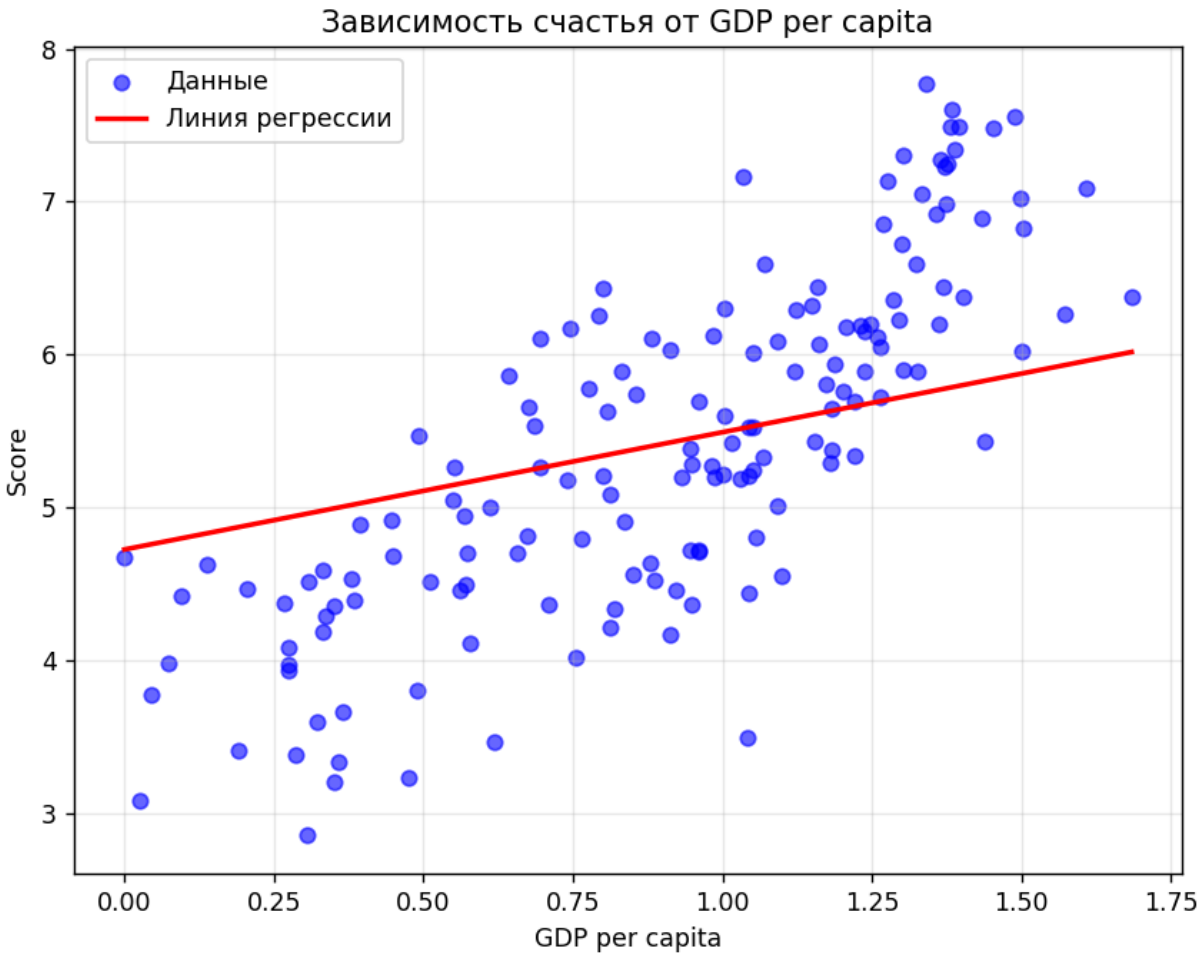
```
X = data[['GDP per capita', 'Social support', 'Healthy life expectancy']]
y = data['Score']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LinearRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"MSE: {mse:.3f}")
print(f"R²: {r2:.3f}")

gdp = data['GDP per capita'].values.reshape(-1, 1)
gdp_sorted_idx = np.argsort(gdp.flatten())
gdp_sorted = gdp[gdp_sorted_idx]

X_line = pd.DataFrame({
    'GDP per capita': gdp_sorted.flatten(),
    'Social support': [data['Social support'].mean()] * len(gdp_sorted),
    'Healthy life expectancy': [data['Healthy life expectancy'].mean()] * len(gdp_sorted)
})

y_line = model.predict(X_line)

plt.figure(figsize=(8, 6))
plt.scatter(data['GDP per capita'], data['Score'], color='blue', alpha=0.6, label='Данные')
plt.plot(gdp_sorted, y_line, color='red', linewidth=2, label='Линия регрессии')
plt.xlabel('GDP per capita')
plt.ylabel('Score')
plt.title('Зависимость счастья от GDP per capita')
plt.legend()
plt.grid(True, alpha=0.3)
plt.show()
```

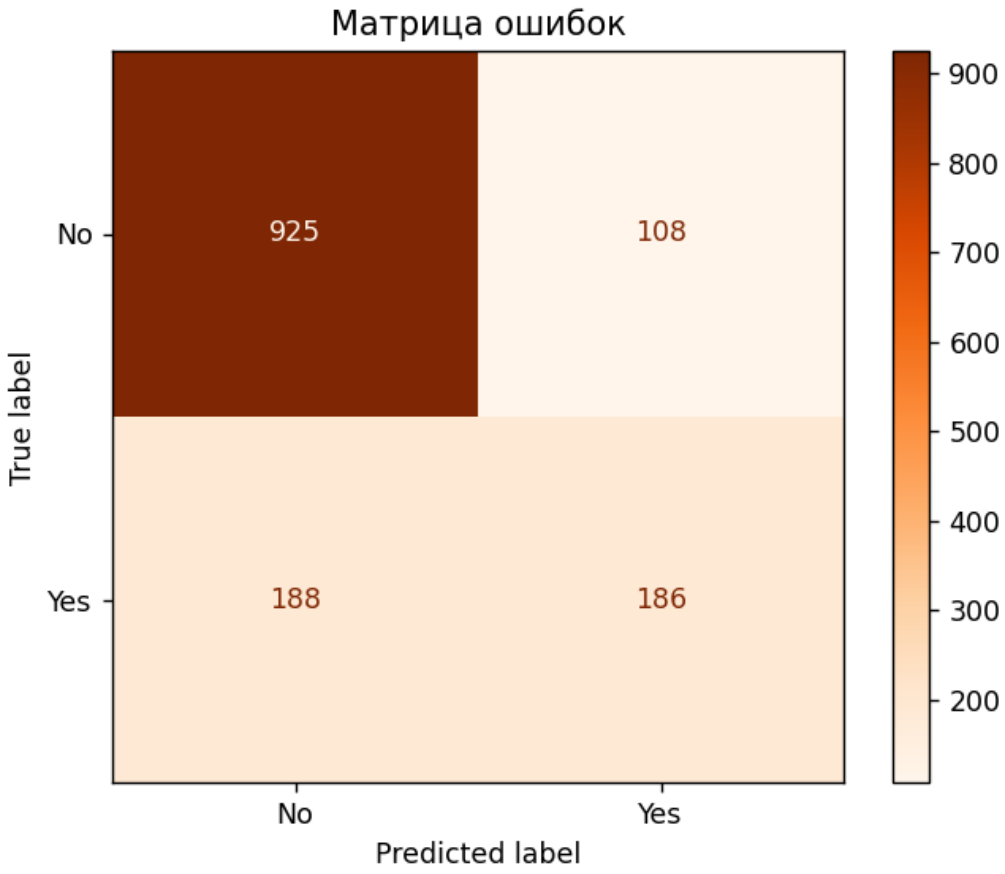
2)

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix,
ConfusionMatrixDisplay
import matplotlib.pyplot as plt

df = pd.read_csv("Telco-Customer-Churn.csv")

if 'customerID' in df.columns:
    df = df.drop('customerID', axis=1)
```

```
df = df.replace(" ", None).dropna()

for col in df.select_dtypes(include='object').columns:
    df[col] = LabelEncoder().fit_transform(df[col])

X = df.drop('Churn', axis=1)
y = df['Churn']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train, y_train)

y_pred = log_reg.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)

print(f"Accuracy: {accuracy:.3f}")
print(f"Precision: {precision:.3f}")
print(f"Recall: {recall:.3f}")

cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=['No', 'Yes'])
disp.plot(cmap='Oranges')
plt.title('Матрица ошибок')
plt.show()
```

Диаграммы после выполнения программы:


Зависимость счастья от GDP per capita
Данные
Линия регрессии
Score
GDP per capita
8
7
6
5
4
3
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75


Матрица ошибок
925
108
188
186
No
Yes
True label
Predicted label
900
800
700
600
500
400
300
200

Консольный вывод:

1)

MSE: 0.464

R^2: 0.554

2)

Accuracy: 0.790

Precision: 0.633

Recall: 0.497