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
Лысюк Р. А.
**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 4**

- **Регрессия (Определение веса рыбы)**
  1. Fish Market
  2. Предсказать вес рыбы (Weight)
  3. **Задания:**
     - загрузите данные. В качестве признаков используйте Length1, Length2, Length3, Height, Width;
     - обучите модель **линейной регрессии**;
     - оцените качество, рассчитав **$R^2$** и **RMSE** (Root Mean Squared Error);
     - постройте диаграмму рассеяния для Length3 и Weight с линией регрессии.
- **Классификация (Прогнозирование отклика на банковское предложение)**
  1. Bank Marketing UCI
  2. Предсказать, подпишется ли клиент на срочный вклад (y)
  3. **Задания:**
     - загрузите данные, преобразуйте категориальные признаки;
     - обучите модель **логистической регрессии**;
     - рассчитайте **Accuracy**, **Precision** и **Recall** для класса "yes";
     - постройте **матрицу ошибок**.

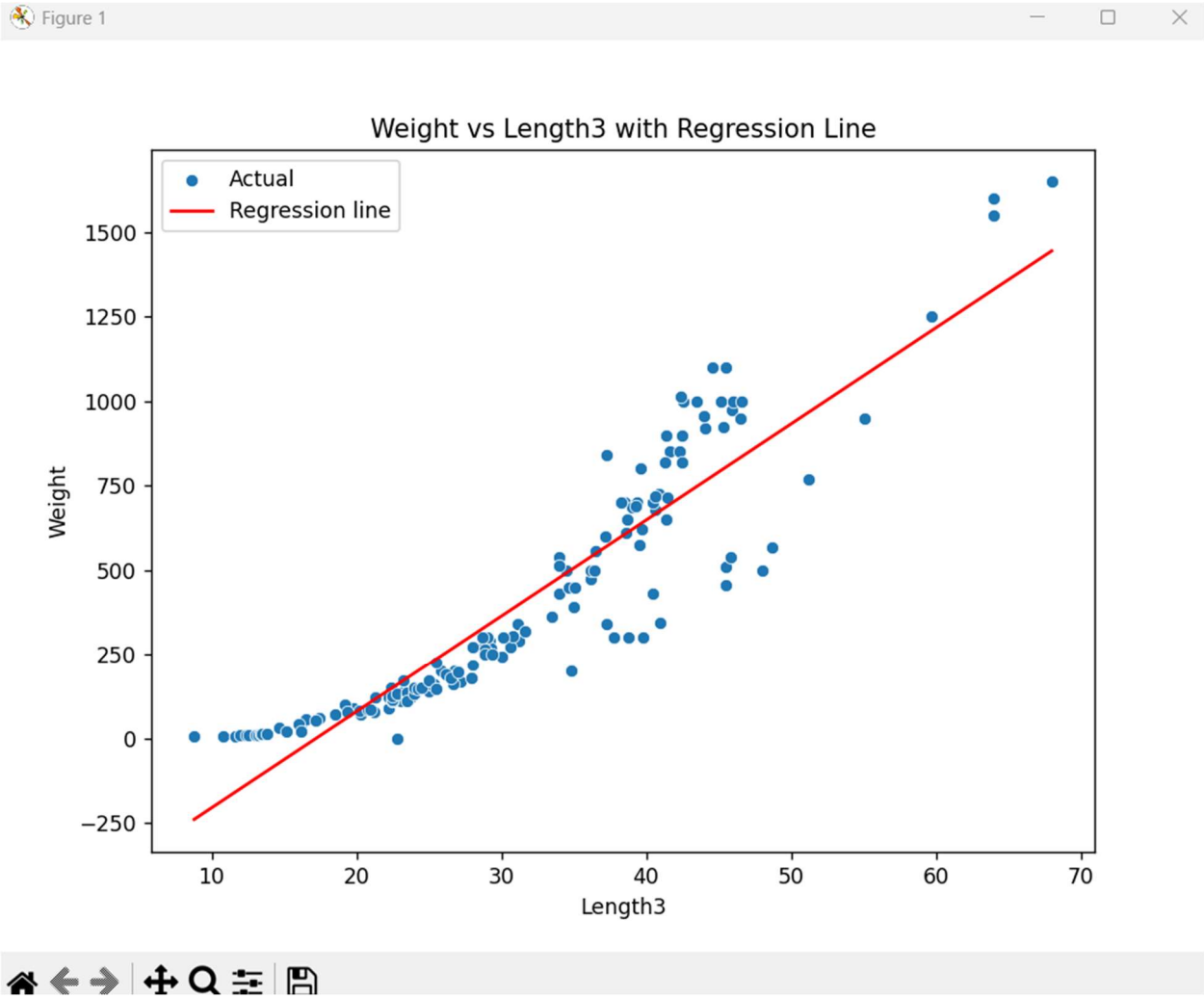
Код программы 1:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_squared_error, r2_score
```

```
# Загрузим файл из текущей папки
df = pd.read_csv('fish.csv')

# Остальной код остается без изменений
print(df.head())

# Объявляем признаки и целевую
features = ['Length1', 'Length2', 'Length3', 'Height', 'Width']
X = df[features]
y = df['Weight']

# Обучение модели линейной регрессии
model = LinearRegression()
model.fit(X, y)

# Предсказания и метрики
y_pred = model.predict(X)
r2 = r2_score(y, y_pred)
rmse = np.sqrt(mean_squared_error(y, y_pred))
print(f'R2: {r2:.4f}')
print(f'RMSE: {rmse:.4f}')

# Построение графика Length3 vs Weight с линией регрессии
plt.figure(figsize=(8, 6))
sns.scatterplot(x=df['Length3'], y=y, label='Actual')
# Обучение модели только по Length3 для линии регрессии
single_feature = df[['Length3']]
model_single = LinearRegression().fit(single_feature, y)
y_pred_single = model_single.predict(single_feature)
sns.lineplot(x=df['Length3'], y=y_pred_single,
             color='red', label='Regression line')
plt.xlabel('Length3')
plt.ylabel('Weight')
```

```
plt.title('Weight vs Length3 with Regression Line')
plt.legend()
plt.show()
```

Диаграмма после выполения программы 1:




Вывод в консоли 1:

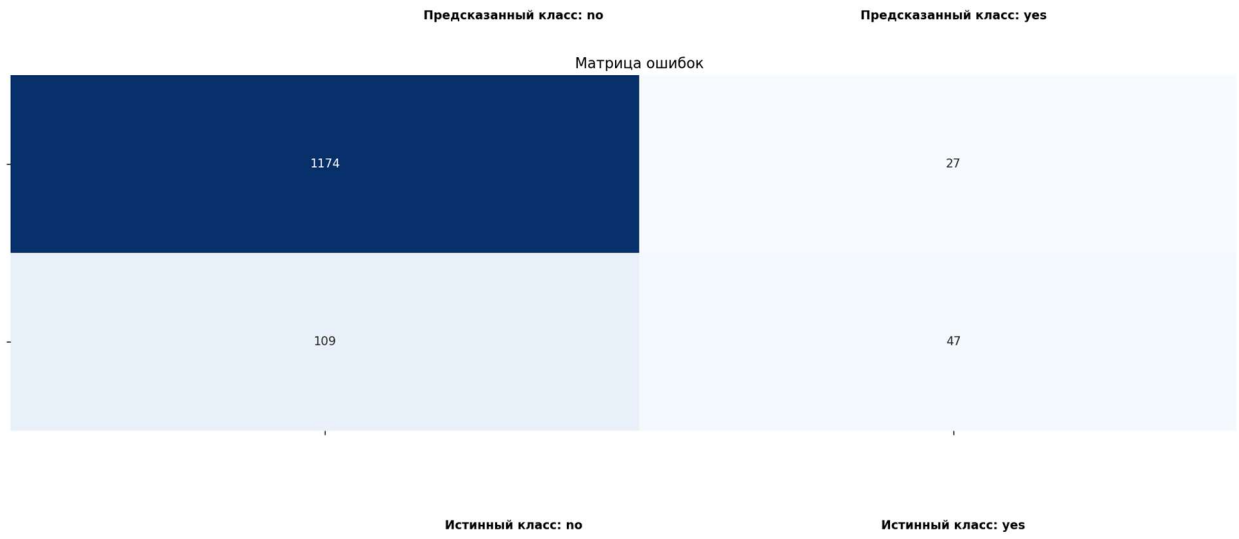
```
  Species  Weight  Length1  Length2  Length3   Height   Width
0   Bream   242.0     23.2     25.4     30.0  11.5200  4.0200
1   Bream   290.0     24.0     26.3     31.2  12.4800  4.3056
2   Bream   340.0     23.9     26.5     31.1  12.3778  4.6961
3   Bream   363.0     26.3     29.0     33.5  12.7300  4.4555
4   Bream   430.0     26.5     29.0     34.0  12.4440  5.1340
R2: 0.8853
RMSE: 120.8631
```

Код программы 2:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Загрузка и подготовка данных
df = pd.read_csv('bank.csv', sep=';')

categorical_cols = df.select_dtypes(include='object').columns.tolist()
target = 'y'
categorical_cols.remove(target)
df_encoded = pd.get_dummies(df, columns=categorical_cols, drop_first=True)

X = df_encoded.drop(target, axis=1)
y = df_encoded[target].map({'no': 0, 'yes': 1})

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42, stratify=y
)

model = LogisticRegression(max_iter=10000, random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print(f"Accuracy: {accuracy_score(y_test, y_pred):.3f}")
print(f"Precision: {precision_score(y_test, y_pred):.3f}")
print(f"Recall: {recall_score(y_test, y_pred):.3f}")

cm = confusion_matrix(y_test, y_pred, labels=[0, 1])
labels = ['no', 'yes']  # метки для классов
```

```
fig, ax = plt.subplots(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False,
            xticklabels=labels, yticklabels=labels, ax=ax)

ax.set_xlabel('')  # убираем стандартные подписи осей
ax.set_ylabel('')

# Убираем подписи по осям — будем добавлять свои
ax.set_xticklabels([])
ax.set_yticklabels([])

# Подписи сверху (предсказанный класс)
ax.text(0.4, 1.15, 'Предсказанный класс: no', ha='center', va='bottom',
        transform=ax.transAxes, fontsize=10, fontweight='bold')
ax.text(0.75, 1.15, 'Предсказанный класс: yes', ha='center', va='bottom',
        transform=ax.transAxes, fontsize=10, fontweight='bold')

# Подписи снизу (истинный класс)
ax.text(0.4, -0.25, 'Истинный класс: no', ha='center', va='top',
        transform=ax.transAxes, fontsize=10, fontweight='bold')
ax.text(0.75, -0.25, 'Истинный класс: yes', ha='center', va='top',
        transform=ax.transAxes, fontsize=10, fontweight='bold')

ax.set_title('Матрица ошибок')

plt.tight_layout()
plt.show()
```

Диаграмма после выполения программы 2:




Вывод в консоли 2 :

```
rgenceWarning: lbfgs failed to converge after 10000 iteration(s) (status=1):
STOP: TOTAL NO. OF ITERATIONS REACHED LIMIT

Increase the number of iterations to improve the convergence (max_iter=10000).
You might also want to scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
Accuracy: 0.900
Precision: 0.635
Recall: 0.301
```

Вывод: Изучил  применение линейной и логистической регрессии для решения  практических задач. Научился  обучать  модели, оценивать их качество с помощью соответствующих метрик и интерпретировать результаты.