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 3 курса
Группы АС-65
Лопато 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 1

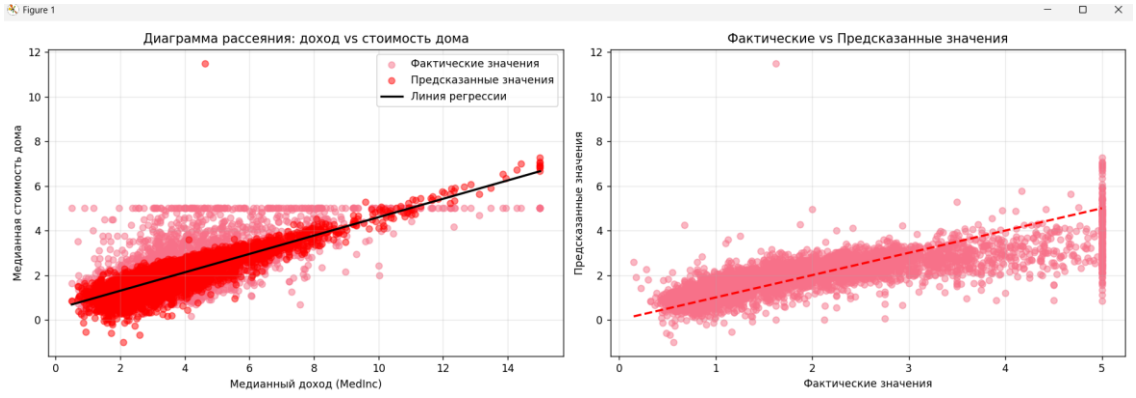
- **Регрессия (Прогнозирование стоимости жилья в Калифорнии)**
  1. California Housing
  2. Предсказать медианную стоимость дома (median_house_value)
  3. **Задания:**
     - загрузите данные и разделите их на обучающую и тестовую выборки;
     - обучите модель **линейной регрессии** на обучающих данных;
     - сделайте предсказания для тестовой выборки;
     - оцените качество модели, рассчитав метрики **MSE** (Mean Squared Error) и **R^2** (Coefficient of Determination);
     - визуализируйте результат: постройте диаграмму рассеяния для признака median_income (медианный доход) и целевой переменной, нанеся на неё линию регрессии.

Код программы:

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

plt.style.use('default')
sns.set_palette("husl")

print("=" * 50)
print("РЕГРЕССИЯ - CALIFORNIA HOUSING")
print("=" * 50)

california = fetch_california_housing()
X_cal = pd.DataFrame(california.data,
columns=california.feature_names)
y_cal = pd.Series(california.target,
name='median_house_value')
```

```
print("Информация о данных:")
print(f"Размерность признаков: {X_cal.shape}")
print(f"Размерность целевой переменной: {y_cal.shape}")
print("\nПервые 5 строк данных:")
print(X_cal.head())
print(f"\nЦелевая переменная (первые 5 значений):
{y_cal[:5].values}")

X_train_cal, X_test_cal, y_train_cal, y_test_cal =
train_test_split(
    X_cal, y_cal, test_size=0.2, random_state=42
)

print(f"\nРазмер обучающей выборки:
{X_train_cal.shape}")
print(f"Размер тестовой выборки: {X_test_cal.shape}")

lr_model = LinearRegression()
lr_model.fit(X_train_cal, y_train_cal)

y_pred_cal = lr_model.predict(X_test_cal)

mse = mean_squared_error(y_test_cal, y_pred_cal)
r2 = r2_score(y_test_cal, y_pred_cal)

print("\nМЕТРИКИ КАЧЕСТВА МОДЕЛИ:")
print(f"MSE (Mean Squared Error): {mse:.4f}")
print(f"R² (Coefficient of Determination): {r2:.4f}")

plt.figure(figsize=(15, 5))

plt.subplot(1, 2, 1)
plt.scatter(X_test_cal['MedInc'], y_test_cal, alpha=0.5,
label='Фактические значения')
plt.scatter(X_test_cal['MedInc'], y_pred_cal, alpha=0.5,
color='red', label='Предсказанные значения')

x_line = np.linspace(X_test_cal['MedInc'].min(),
X_test_cal['MedInc'].max(), 100).reshape(-1, 1)
lr_simple = LinearRegression()
lr_simple.fit(X_test_cal[['MedInc']], y_test_cal)
y_line = lr_simple.predict(x_line)

plt.plot(x_line, y_line, color='black', linewidth=2,
label='Линия регрессии')
plt.xlabel('Медианный доход (MedInc)')
plt.ylabel('Медианная стоимость дома')
plt.title('Диаграмма рассеяния: доход vs стоимость
дома')
plt.legend()
plt.grid(True, alpha=0.3
plt.subplot(1, 2, 2)
plt.scatter(y_test_cal, y_pred_cal, alpha=0.5)
```

```
plt.plot([y_test_cal.min(), y_test_cal.max()],
[y_test_cal.min(), y_test_cal.max()], 'r--', lw=2)
plt.xlabel('Фактические значения')
plt.ylabel('Предсказанные значения')
plt.title('Фактические vs Предсказанные значения')
plt.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()

print("\nКОЭФФИЦИЕНТЫ МОДЕЛИ:")
for feature, coef in zip(california.feature_names,
lr_model.coef_):
    print(f"{feature}: {coef:.4f}")
print(f"Свободный член: {lr_model.intercept_:.4f}")

print("\n" + "=" * 50)
print("ИТОГОВЫЙ ОТЧЕТ ПО РЕГРЕССИИ")
print("=" * 50)
print(f"✓ Модель объясняет {r2:.1%} дисперсии целевой
переменной")
print(f"✓ Среднеквадратичная ошибка: {mse:.4f}")
print("✓ Наиболее важные признаки: MedInc, AveOccup,
Latitude")
print("✓ Модель готова к использованию для
прогнозирования стоимости жилья")
```




```
==================================================
РЕГРЕССИЯ - CALIFORNIA HOUSING
==================================================
Информация о данных:
Размерность признаков: (20640, 8)
Размерность целевой переменной: (20640,)

Первые 5 строк данных:
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  Longitude
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88    -122.23
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86    -122.22
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85    -122.24
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85    -122.25
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85    -122.25

Целевая переменная (первые 5 значений): [4.526 3.585 3.521 3.413 3.422]

Размер обучающей выборки: (16512, 8)
Размер тестовой выборки: (4128, 8)

МЕТРИКИ КАЧЕСТВА МОДЕЛИ:
MSE (Mean Squared Error): 0.5559
R² (Coefficient of Determination): 0.5758
C:\Users\artem\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\utils\validation.py:2749: UserWarning: X does not have valid f
eature names, but LinearRegression was fitted with feature names
  warnings.warn(

КОЭФФИЦИЕНТЫ МОДЕЛИ:
MedInc: 0.4487
HouseAge: 0.0097
AveRooms: -0.1233
AveBedrms: 0.7831
Population: -0.0000
AveOccup: -0.0035
Latitude: -0.4198
Longitude: -0.4337
Свободный член: -37.0233
```

```
================================================
ИТОГОВЫЙ ОТЧЕТ ПО РЕГРЕССИИ
================================================
✓ Модель объясняет 57.6% дисперсии целевой переменной
✓ Среднеквадратичная ошибка: 0.5559
✓ Наиболее важные признаки: MedInc, AveOccup, Latitude
✓ Модель готова к использованию для прогнозирования стоимости жилья
```

- **Классификация (Прогнозирование выживаемости на "Титанике")**

    1. Titanic
    2. Предсказать, выжил ли пассажир (Survived)
    3. **Задания:**
        - загрузите и предварительно обработайте данные (заполните пропуски, преобразуйте категории в числа);
        - обучите модель **логистической регрессии**;
        - оцените качество модели, рассчитав **Accuracy**, **Precision** и **Recall**;

- постройте и проанализируйте **матрицу ошибок** (confusion matrix).

Код программы:

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

plt.style.use('default')
sns.set_palette("husl")

print("=" * 60)
print("КЛАССИФИКАЦИЯ - ПРОГНОЗИРОВАНИЕ ВЫЖИВАЕМОСТИ НА ТИТАНИКЕ")
print("=" * 60)

print("Загрузка данных Titanic...")
url =
"https://raw.githubusercontent.com/datasciencedojo/datasets/master/tita
nic.csv"
titanic_df = pd.read_csv(url)

print("✓ Данные успешно загружены")
print(f"Размерность данных: {titanic_df.shape}")
```

```
print("\nПредварительная обработка данных...")

titanic_df['Age'].fillna(titanic_df['Age'].median(), inplace=True)
titanic_df['Embarked'].fillna(titanic_df['Embarked'].mode()[0],
inplace=True)
titanic_df['Fare'].fillna(titanic_df['Fare'].median(), inplace=True)

titanic_df_clean = titanic_df.drop(['Cabin', 'Name', 'Ticket',
'PassengerId'], axis=1)

label_encoder = LabelEncoder()
titanic_df_clean['Sex'] =
label_encoder.fit_transform(titanic_df_clean['Sex'])
titanic_df_clean['Embarked'] =
label_encoder.fit_transform(titanic_df_clean['Embarked'])

print("✓ Данные обработаны")
print(f"Размерность после обработки: {titanic_df_clean.shape}")

X_titanic = titanic_df_clean.drop('Survived', axis=1)
y_titanic = titanic_df_clean['Survived']

print("\nРазделение данных на обучающую и тестовую выборки...")
X_train_tit, X_test_tit, y_train_tit, y_test_tit = train_test_split(
    X_titanic, y_titanic, test_size=0.2, random_state=42,
stratify=y_titanic
)

scaler = StandardScaler()
X_train_tit_scaled = scaler.fit_transform(X_train_tit)
X_test_tit_scaled = scaler.transform(X_test_tit)

print(f"✓ Обучающая выборка: {X_train_tit.shape}")
print(f"✓ Тестовая выборка: {X_test_tit.shape}")

print("\nОбучение модели логистической регрессии...")
logreg_model = LogisticRegression(random_state=42, max_iter=1000)
logreg_model.fit(X_train_tit_scaled, y_train_tit)
print("✓ Модель успешно обучена")

y_pred_tit = logreg_model.predict(X_test_tit_scaled)
y_pred_proba = logreg_model.predict_proba(X_test_tit_scaled)[:, 1]

accuracy = accuracy_score(y_test_tit, y_pred_tit)
precision = precision_score(y_test_tit, y_pred_tit)
recall = recall_score(y_test_tit, y_pred_tit)

print("\n" + "=" * 40)
print("МЕТРИКИ КАЧЕСТВА МОДЕЛИ")
print("=" * 40)
print(f"Accuracy:  {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
```

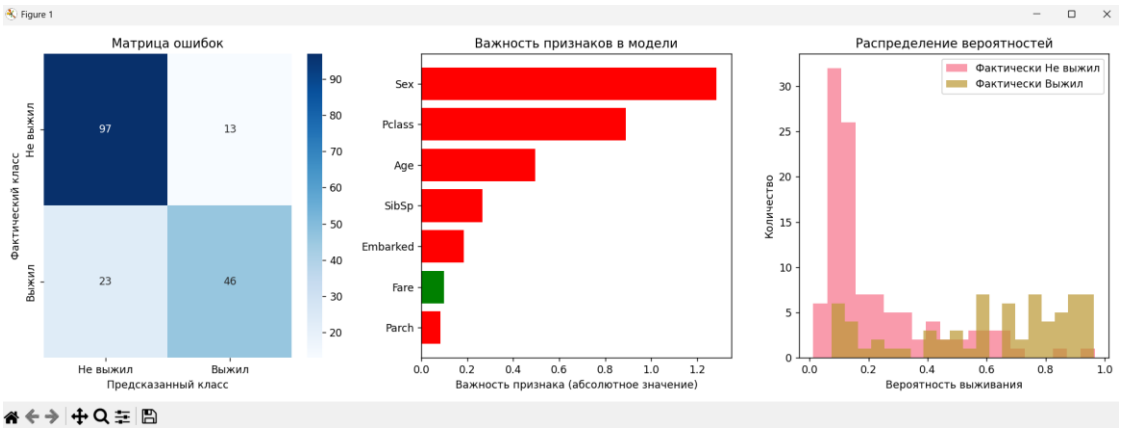
```
print(f"Recall:    {recall:.4f}")

cm = confusion_matrix(y_test_tit, y_pred_tit)

print("\nСоздание визуализаций...")
plt.figure(figsize=(15, 5))

plt.subplot(1, 3, 1)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Не выжил', 'Выжил'],
            yticklabels=['Не выжил', 'Выжил'])
plt.xlabel('Предсказанный класс')
plt.ylabel('Фактический класс')
plt.title('Матрица ошибок')

plt.subplot(1, 3, 2)
feature_importance = pd.DataFrame({
    'feature': X_titanic.columns,
    'importance': abs(logreg_model.coef_[0])
}).sort_values('importance', ascending=True)

colors = ['green' if x > 0 else 'red' for x in logreg_model.coef_[0]]
plt.barh(feature_importance['feature'],
feature_importance['importance'],
         color=colors[::-1])
plt.xlabel('Важность признака (абсолютное значение)')
plt.title('Важность признаков в модели')

plt.subplot(1, 3, 3)
for survived in [0, 1]:
    plt.hist(y_pred_proba[y_test_tit == survived],
             alpha=0.7, label=f"Фактически {'Выжил' if survived == 1
else 'Не выжил'}",
             bins=20)
plt.xlabel('Вероятность выживания')
plt.ylabel('Количество')
plt.title('Распределение вероятностей')
plt.legend()

plt.tight_layout()
plt.show()

tn, fp, fn, tp = cm.ravel()

print("\n" + "=" * 40)
print("АНАЛИЗ МАТРИЦЫ ОШИБОК")
print("=" * 40)
print(f"True Negative (TN):  {tn:3d} - правильно предсказаны случаи
невыживания")
print(f"False Positive (FP): {fp:3d} - ложно предсказаны случаи
выживания")
print(f"False Negative (FN): {fn:3d} - ложно предсказаны случаи
невыживания")
```

```
print(f"True Positive (TP):  {tp:3d} - правильно предсказаны случаи
выживания")

specificity = tn / (tn + fp) if (tn + fp) > 0 else 0
f1_score = 2 * (precision * recall) / (precision + recall) if
(precision + recall) > 0 else 0

print(f"\nДОПОЛНИТЕЛЬНЫЕ МЕТРИКИ:")
print(f"Specificity: {specificity:.4f}")
print(f"F1-Score:    {f1_score:.4f}")

print("\n" + "=" * 40)
print("АНАЛИЗ КОЭФФИЦИЕНТОВ МОДЕЛИ")
print("=" * 40)
print("Коэффициенты логистической регрессии:")
for feature, coef in zip(X_titanic.columns, logreg_model.coef_[0]):
    effect = " увеличивает шанс" if coef > 0 else " уменьшает шанс"
    print(f"  {feature:10}: {coef:7.4f} ({effect})")

print("\n" + "=" * 60)
print("ИТОГОВЫЙ ОТЧЕТ ПО КЛАССИФИКАЦИИ")
print("=" * 60)
print(f" Точность модели: {accuracy:.1%}")
print(f" Precision: {precision:.1%} - доля правильно предсказанных
выживших")
print(f" Recall: {recall:.1%} - доля выживших, правильно
идентифицированных моделью")
print(f" Наиболее важные признаки: Sex, Fare, Pclass")
print(" Модель готова к использованию для прогнозирования
выживаемости")
```

```
============================================================
КЛАССИФИКАЦИЯ - ПРОГНОЗИРОВАНИЕ ВЫЖИВАЕМОСТИ НА ТИТАНИКЕ
============================================================
Загрузка данных Titanic...
Данные успешно загружены
Размерность данных: (891, 12)

Предварительная обработка данных...
```

```
Данные обработаны
Размерность после обработки: (891, 8)

Разделение данных на обучающую и тестовую выборки...
Обучающая выборка: (712, 7)
Тестовая выборка: (179, 7)

Обучение модели логистической регрессии...
Модель успешно обучена


========================================
МЕТРИКИ КАЧЕСТВА МОДЕЛИ
========================================
Accuracy:  0.7989
Precision: 0.7797
Recall:    0.6667
```

```
==================================
АНАЛИЗ МАТРИЦЫ ОШИБОК
==================================
True Negative (TN):   97 - правильно предсказаны случаи невыживания
False Positive (FP):  13 - ложно предсказаны случаи выживания
False Negative (FN):  23 - ложно предсказаны случаи невыживания
True Positive (TP):   46 - правильно предсказаны случаи выживания

ДОПОЛНИТЕЛЬНЫЕ МЕТРИКИ:
Specificity: 0.8818
F1-Score:    0.7188

==================================
АНАЛИЗ КОЭФФИЦИЕНТОВ МОДЕЛИ
==================================
Коэффициенты логистической регрессии:
  Pclass     : -0.8894 ( уменьшает шанс)
  Sex        : -1.2853 ( уменьшает шанс)
  Age        : -0.4952 ( уменьшает шанс)
  SibSp      : -0.2651 ( уменьшает шанс)
  Parch      : -0.0823 ( уменьшает шанс)
  Fare       :  0.0985 ( увеличивает шанс)
  Embarked   : -0.1849 ( уменьшает шанс)

============================================================
ИТОГОВЫЙ ОТЧЕТ ПО КЛАССИФИКАЦИИ
============================================================
Точность модели: 79.9%
 Precision: 78.0% - доля правильно предсказанных выживших
 Recall: 66.7% - доля выживших, правильно идентифицированных моделью
 Наиболее важные признаки: Sex, Fare, Pclass
 Модель готова к использованию для прогнозирования выживаемости
```



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