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-66
Гончерёнок К.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.

Вариант 2

Ход работы

Регрессия (Прогнозирование медицинских расходов)

1. Medical Cost Personal Datasets
2. Предсказать страховые выплаты (charges)
3. Задания:

• загрузите и обработайте категориальные признаки (например, sex, smoker);

• обучите модель линейной регрессии для предсказания charges;

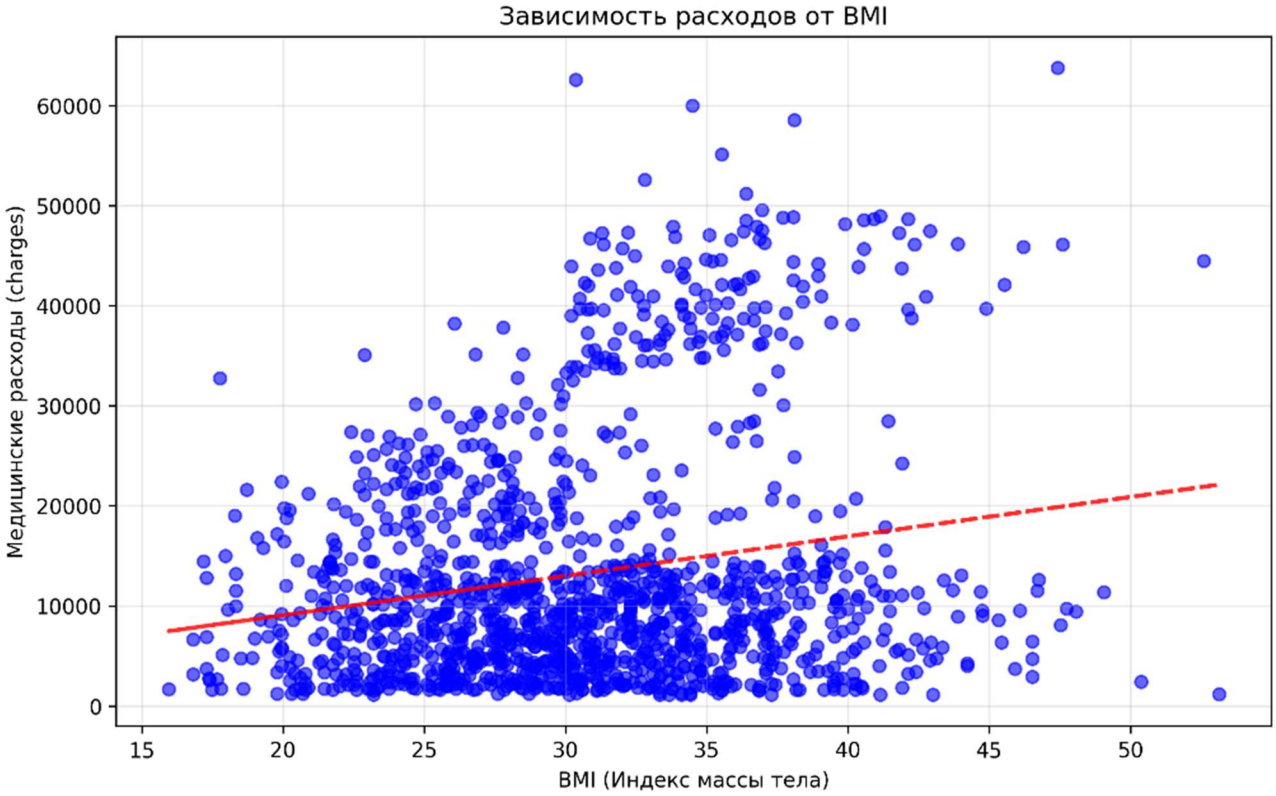
• рассчитайте MAE (Mean Absolute Error) и R2;

• визуализируйте зависимость charges от bmi (индекс массы тела) с помощью диаграммы рассеяния и линии регрессии.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score
from sklearn.preprocessing import LabelEncoder

df_medical = pd.read_csv('medical_cost_personal_dataset.csv')
categorical_columns = ['sex', 'smoker', 'region']
label_encoders = {}
for col in categorical_columns:
    le = LabelEncoder()
    df_medical[col + '_encoded'] = le.fit_transform(df_medical[col])
    label_encoders[col] = le
X_reg = df_medical[['age', 'bmi', 'children', 'sex_encoded', 'smoker_encoded',
'region_encoded']]
y_reg = df_medical['charges']
X_train_reg, X_test_reg, y_train_reg, y_test_reg = train_test_split(
    X_reg, y_reg, test_size=0.2, random_state=42
)
lr_model = LinearRegression()
lr_model.fit(X_train_reg, y_train_reg)
y_pred_reg = lr_model.predict(X_test_reg)
mae = mean_absolute_error(y_test_reg, y_pred_reg)
r2 = r2_score(y_test_reg, y_pred_reg)
print("РЕГРЕССИЯ  МЕДИЦИНСКИЕ РАСХОДЫ  regr.py:34 - regression.py:34")
print(f"MAE: {mae:.2f}  regr.py:35 - regression.py:35")
print(f"R² Score: {r2:.4f}  regr.py:36 - regression.py:36")

plt.figure(figsize=(10, 6))
plt.scatter(df_medical['bmi'], df_medical['charges'], alpha=0.6, color='blue')
z = np.polyfit(df_medical['bmi'], df_medical['charges'], 1)
p = np.poly1d(z)
```

```
plt.plot(df_medical['bmi'], p(df_medical['bmi']), "r--", alpha=0.8, linewidth=2)
plt.xlabel('BMI (Индекс массы тела)')
plt.ylabel('Медицинские расходы (charges)')
plt.title('Зависимость расходов от BMI')
plt.grid(True, alpha=0.3)
plt.savefig('medical_costs_regression.png', dpi=300, bbox_inches='tight')
plt.show()

feature_importance = pd.DataFrame({
    'Feature': X_reg.columns,
    'Coefficient': lr_model.coef_
}).sort_values('Coefficient', key=abs, ascending=False)
print("\nВажность признаков:  regr.py:54 - regression.py:54")
print(feature_importance)
```

```
РЕГРЕССИЯ  МЕДИЦИНСКИЕ РАСХОДЫ  regr.py:34 - regression.py:34
MAE: 4186.51  regr.py:35 - regression.py:35
R² Score: 0.7833  regr.py:36 - regression.py:36

Важность признаков:  regr.py:54 - regression.py:54
          Feature   Coefficient
4  smoker_encoded  23647.818096
2        children    425.091456
1             bmi    335.781491
5  region_encoded   -271.284266
0             age    257.056264
3     sex_encoded    -18.791457
```



Классификация (Диагностика заболеваний сердца)

1. Heart Disease UCI

2. Предсказать наличие у пациента болезни сердца (target)

3. Задания:

• загрузите данные и разделите их на обучающую и тестовую выборки;

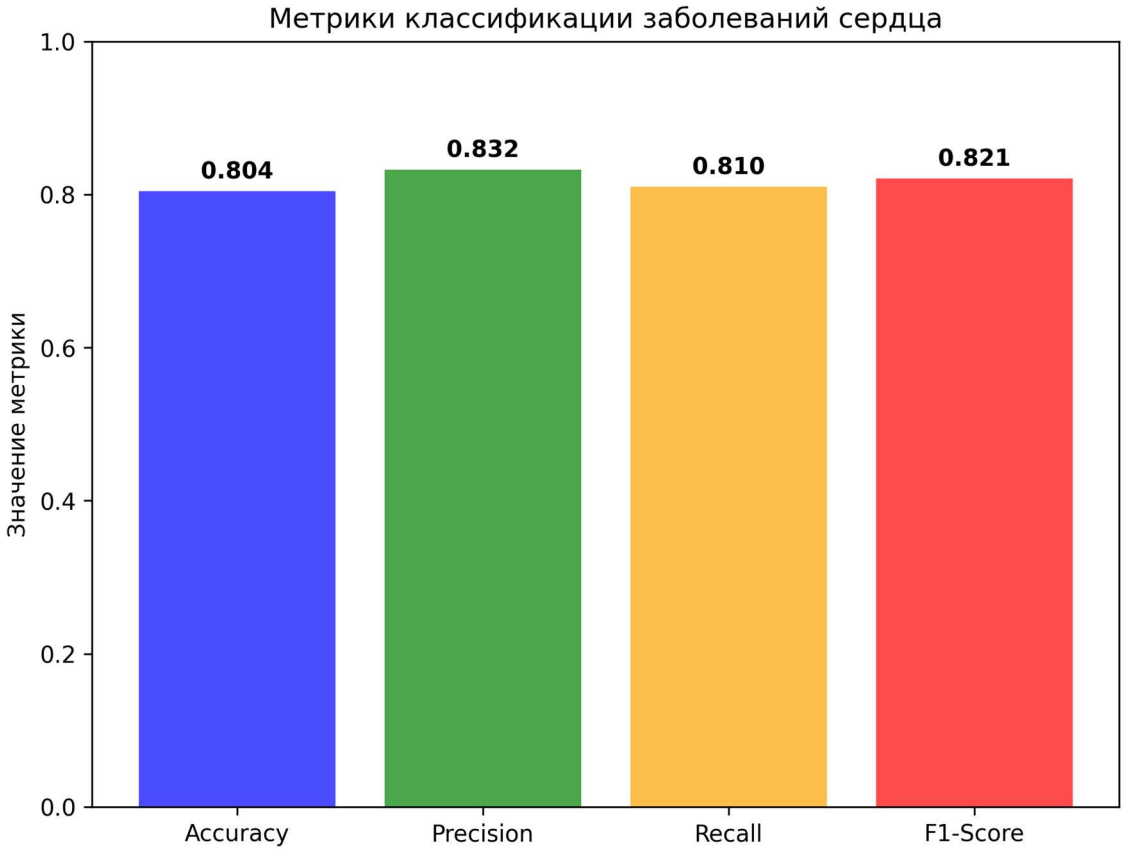
• обучите модель логистической регрессии;

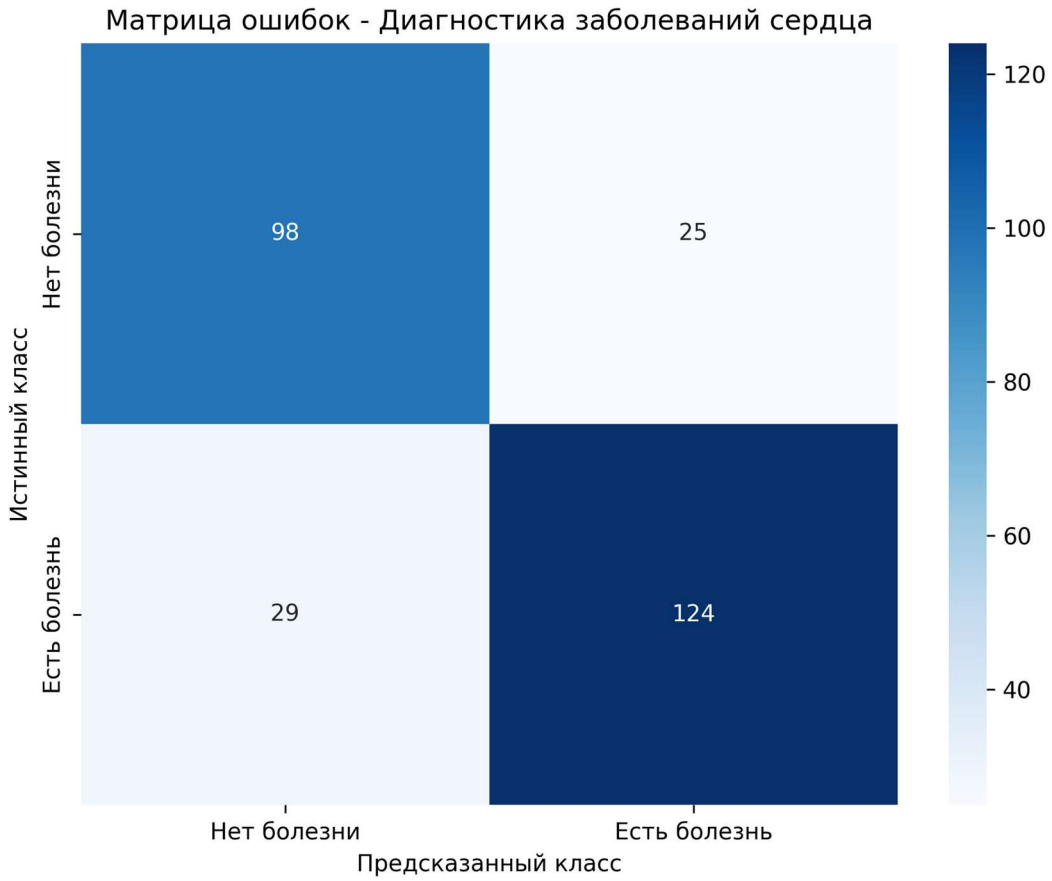
• оцените модель с помощью Accuracy, Precision, Recall и F1-score;

• постройте матрицу ошибок.

```
import pandas as pd
import numpy as np
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, f1_score, confusion_matrix
from sklearn.preprocessing import LabelEncoder, StandardScaler
df_heart = pd.read_csv('heart_disease_uci.csv')
df_heart['heart_disease'] = (df_heart['num'] > 0).astype(int)
numeric_features = ['age', 'trestbps', 'chol', 'thalch', 'oldpeak']
categorical_features = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal']
for col in numeric_features:
    if col in df_heart.columns:
        df_heart[col].fillna(df_heart[col].median(), inplace=True)
for col in categorical_features:
    if col in df_heart.columns:
        df_heart[col].fillna(df_heart[col].mode()[0] if not df_heart[col].mode().empty else 'unknown', inplace=True)
for col in categorical_features:
    if col in df_heart.columns:
        le = LabelEncoder()
        df_heart[col + '_encoded'] = le.fit_transform(df_heart[col].astype(str))
feature_columns = numeric_features + [col + '_encoded' for col in categorical_features]
feature_columns = [col for col in feature_columns if col in df_heart.columns]
X_clf = df_heart[feature_columns]
y_clf = df_heart['heart_disease']
scaler = StandardScaler()
X_clf_scaled = scaler.fit_transform(X_clf)
X_train_clf, X_test_clf, y_train_clf, y_test_clf = train_test_split(
    X_clf_scaled, y_clf, test_size=0.3, random_state=42, stratify=y_clf
)
logreg_model = LogisticRegression(random_state=42, max_iter=1000)
logreg_model.fit(X_train_clf, y_train_clf)
y_pred_clf = logreg_model.predict(X_test_clf)
accuracy = accuracy_score(y_test_clf, y_pred_clf)
precision = precision_score(y_test_clf, y_pred_clf)
recall = recall_score(y_test_clf, y_pred_clf)
f1 = f1_score(y_test_clf, y_pred_clf)
print("КЛАССИФИКАЦИЯ  БОЛЕЗНИ СЕРДЦА - classif.py:53")
print(f"Accuracy: {accuracy:.4f} - classif.py:54")
print(f"Precision: {precision:.4f} - classif.py:55")
print(f"Recall: {recall:.4f} - classif.py:56")
print(f"F1Score: {f1:.4f} - classif.py:57")
```

```
cm = confusion_matrix(y_test_clf, y_pred_clf)
print("\nМатрица ошибок: - classif.py:60")
print(cm)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
           xticklabels=['Нет болезни', 'Есть болезнь'],
           yticklabels=['Нет болезни', 'Есть болезнь'])
plt.xlabel('Предсказанный класс')
plt.ylabel('Истинный класс')
plt.title('Матрица ошибок - Диагностика заболеваний сердца')
plt.savefig('heart_disease_confusion_matrix.png', dpi=300, bbox_inches='tight')
plt.show()
metrics = ['Accuracy', 'Precision', 'Recall', 'F1-Score']
values = [accuracy, precision, recall, f1]
plt.figure(figsize=(8, 6))
bars = plt.bar(metrics, values, color=['blue', 'green', 'orange', 'red'], alpha=0.7)
plt.ylabel('Значение метрики')
plt.title('Метрики классификации заболеваний сердца')
plt.ylim(0, 1)
for bar, value in zip(bars, values):
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.01,
             f'{value:.3f}', ha='center', va='bottom', fontweight='bold')
plt.savefig('heart_disease_metrics.png', dpi=300, bbox_inches='tight')
plt.show()
```





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