Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

Отчет по лабораторной работе № 4

«Линейные модели, SVM и деревья решений.»

по дисциплине «Технологии машинного обучения»

Студент ИУ5-65Б _____ Д.А. Шиленок
(Группа) (Подпись, дата) (И.О.Фамилия)

Преподаватель _____ Ю.Е. Гапанюк
(Подпись, дата) (И.О.Фамилия)

Москва

2025

Цель лабораторной работы: изучение линейных моделей, SVM и деревьев решений.

Задание

1. Выберите набор данных (датасет) для решения задачи классификации или регрессии.
2. В случае необходимости проведите удаление или заполнение пропусков и кодирование категориальных признаков.
3. С использованием метода train_test_split разделите выборку на обучающую и тестовую.
4. Обучите следующие модели:
    - одну из линейных моделей (линейную или полиномиальную регрессию при решении задачи регрессии, логистическую регрессию при решении задачи классификации);
    - SVM;
    - дерево решений.
5. Оцените качество моделей с помощью двух подходящих для задачи метрик. Сравните качество полученных моделей.
6. Постройте график, показывающий важность признаков в дереве решений.
7. Визуализируйте дерево решений или выведите правила дерева решений в текстовом виде.

Текст программы и результаты

Лабораторная работа №4

Линейные модели, SVM и деревья решений.

Шиленок Даниил ИУ5-65Б

```
import seaborn as sns

df = sns.load_dataset('titanic')
df = df.drop(['class','who','adult_male','deck','embark_town','alive'], axis=1)
df = df.dropna()
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

df['sex'] = LabelEncoder().fit_transform(df['sex'])         # male=1, female=0
df['embarked'] = LabelEncoder().fit_transform(df['embarked'])  # C=0, Q=1, S=2

# Разделяем признаки и целевую переменную
X = df.drop(['survived'], axis=1)
y = df['survived']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=1
)
```

```
from sklearn.linear_model import LogisticRegression
log_model = LogisticRegression(max_iter=500)
log_model.fit(X_train, y_train)
log_preds = log_model.predict(X_test)
```

```
from sklearn.metrics import accuracy_score, f1_score

print('Логистическая регрессия:')
print(f'Accuracy: {accuracy_score(y_test, log_preds):.3f}')
print(f'F1-score: {f1_score(y_test, log_preds):.3f}')
```

```python
from sklearn.svm import SVC

svm_model = SVC(kernel='linear',class_weight='balanced')
svm_model.fit(X_train, y_train)
svm_preds = svm_model.predict(X_test)
```

```python
print('SVM:')
print(f'Accuracy: {accuracy_score(y_test, svm_preds):.3f}')
print(f'F1-score: {f1_score(y_test, svm_preds):.3f}')
```

```python
from sklearn.tree import DecisionTreeClassifier

tree_model = DecisionTreeClassifier(max_depth=3, random_state=2)
tree_model.fit(X_train, y_train)
tree_preds = tree_model.predict(X_test)
```

```python
print('Дерево решений:')
print(f'Accuracy: {accuracy_score(y_test, tree_preds):.3f}')
print(f'F1-score: {f1_score(y_test, tree_preds):.3f}')
```

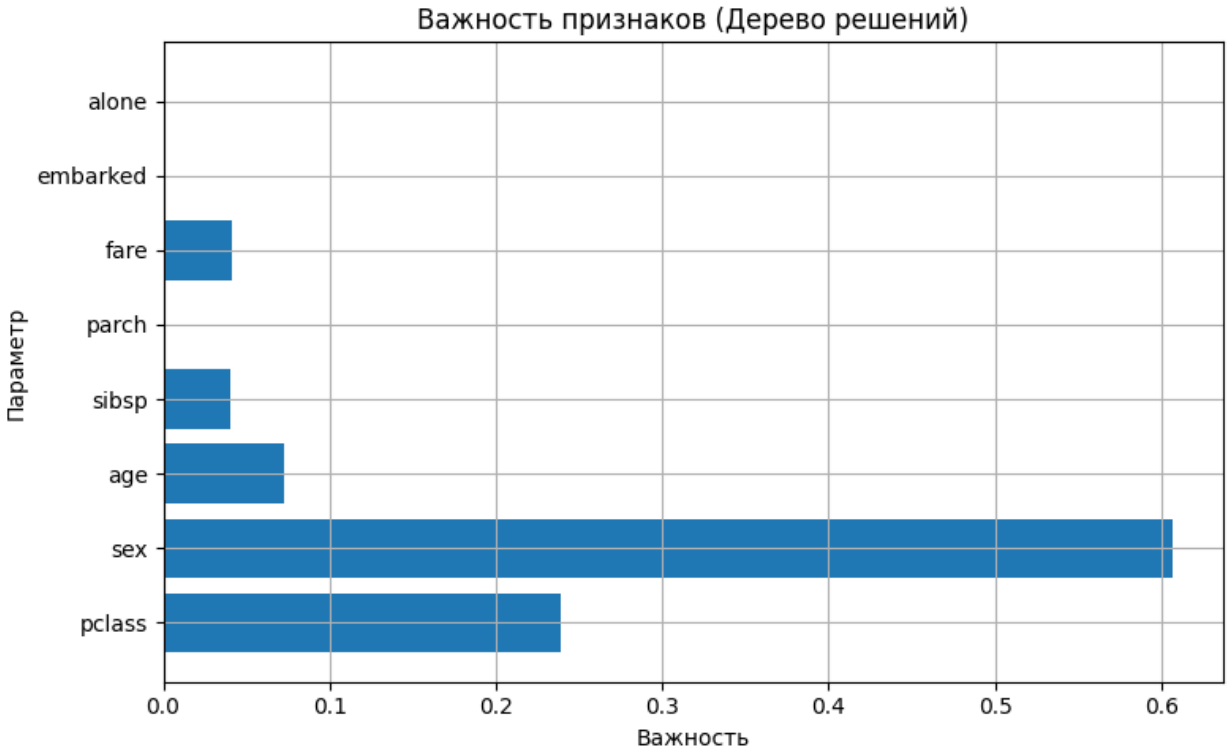
```
Логистическая регрессия:
Accuracy: 0.748
F1-score: 0.695


SVM:
Accuracy: 0.748
F1-score: 0.690


Дерево решений:
Accuracy: 0.790
F1-score: 0.732
```

```python
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 5))
plt.barh(X.columns, tree_model.feature_importances_)
plt.title('Важность признаков (Дерево решений)')
plt.xlabel('Важность')
plt.ylabel('Параметр')
plt.grid(True)
plt.tight_layout()
plt.show()
```




```python
from sklearn.tree import plot_tree
plt.figure(figsize=(20, 10))
plot_tree(
    tree_model,
    feature_names=X.columns,
    class_names=["Не выжил", "Выжил"],
    filled=True,
    rounded=True,
    fontsize=12
)
plt.title("Decision Tree", fontsize=16)
plt.show()
```

Полученное дерево решений: