Министерство цифрового развития

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра прикладной математики и кибернетики

Отчёт

по лабораторной работе № 3 «Классификация методом дерева решений»

Выполнил:

студент группы ИП-312

Прозоренко К.В

Работу проверил: старший преподаватель кафедры ПМиК
Дементьева К.И.

Новосибирск 2025 г.

Цель

Освоить практическое применение метода решающего дерева для задач классификации. Исследовать влияние гиперпараметров на качество модели и научиться проводить базовый анализ важности признаков.

Теоретическая часть

Решающее дерево — алгоритм машинного обучения, который строит иерархическую структуру правил «если-то» для классификации или регрессии. Основные гиперпараметры: max_depth — максимальная глубина дерева max_leaf_nodes — максимальное количество листовых узлов min_samples_split — минимальное количество samples для разделения узла

Задание

1. Подготовка данных

Для этого задания я использовал датасет Heart_Disease_Prediction.csv

```
...     Age  Sex  Chest pain type   BP  Cholesterol  FBS over 120  EKG results  \
    0    70    1                4  130          322             0            2
    1    67    0                3  115          564             0            2
    2    57    1                2  124          261             0            0
    3    64    1                4  128          263             0            0
    4    74    0                2  120          269             0            2

       Max HR  Exercise angina  ST depression  Slope of ST  \
    0     109                0            2.4            2
    1     160                0            1.6            2
    2     141                0            0.3            1
    3     105                1            0.2            2
    4     121                1            0.2            1

       Number of vessels fluro  Thallium Heart Disease
    0                        3         3      Presence
    1                        0         7       Absence
    2                        0         7      Presence
    3                        1         7       Absence
    4                        1         3       Absence
    <class 'pandas.core.frame.DataFrame'>
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 270 entries, 0 to 269
Data columns (total 14 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Age                     270 non-null    int64  
 1   Sex                     270 non-null    int64  
 2   Chest pain type         270 non-null    int64  
 3   BP                      270 non-null    int64  
 4   Cholesterol             270 non-null    int64  
 5   FBS over 120            270 non-null    int64  
 6   EKG results             270 non-null    int64  
 7   Max HR                  270 non-null    int64  
 8   Exercise angina         270 non-null    int64  
 9   ST depression           270 non-null    float64
 10  Slope of ST             270 non-null    int64  
 11  Number of vessels fluro 270 non-null    int64  
 12  Thallium                270 non-null    int64  
 13  Heart Disease           270 non-null    object 
dtypes: float64(1), int64(12), object(1)
memory usage: 29.7+ KB
None
Age                     0
Sex                     0
Chest pain type         0
BP                      0
Cholesterol             0
FBS over 120            0
EKG results             0
Max HR                  0
Exercise angina         0
ST depression           0
Slope of ST             0
Number of vessels fluro 0
Thallium                0
Heart Disease           0
dtype: int64
```

Разделение данных на обучающую и тестовую выборку (70/30)

```
num_cols = df.select_dtypes(include=["int64", "float64"]).columns
cat_cols = df.select_dtypes(include=["object"]).columns

for col in num_cols:
    df[col] = df[col].fillna(df[col].median())
for col in cat_cols:
    df[col] = df[col].fillna(df[col].mode()[0])

df["Heart Disease"] = df["Heart Disease"].map({"Absence": 0, "Presence": 1})

X = df.drop("Heart Disease", axis=1)
y = df["Heart Disease"]

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

2. Базовое дерево

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

base_tree = DecisionTreeClassifier(random_state=42)
base_tree.fit(X_train, y_train)

y_pred_base = base_tree.predict(X_test)
base_accuracy = accuracy_score(y_test, y_pred_base)
print("Базовая accuracy:", base_accuracy)
```

```
Базовая accuracy: 0.7283950617283951
```

```
import numpy as np

importances = base_tree.feature_importances_
feature_names = X_train.columns

feat_imp = sorted(
    zip(feature_names, importances),
    key=lambda x: x[1],
    reverse=True
)

print("Топ-3 важных признака:")
for name, val in feat_imp[:3]:
    print(name, ":", val)
```

```
Топ-3 важных признака:
Chest pain type : 0.2600416291629162
Number of vessels fluro : 0.1252095919511363
Slope of ST : 0.10117424242424242
```

Интерпретация результатов:

1. **Chest pain type (0.260)** — наиболее важный признак. Тип боли в груди является ключевым клиническим симптомом при диагностике сердечно-сосудистых заболеваний. Различные типы боли (типичная стенокардия, атипичная боль, неангинальная боль, асимптомное течение) имеют разную степень связи с наличием заболевания.
2. **Number of vessels fluro (0.125)** — количество окрашенных сосудов при флюороскопии отражает степень поражения коронарных артерий. Этот признак напрямую связан с тяжестью атеросклероза и является важным диагностическим показателем.
3. **Slope of ST (0.101)** — наклон сегмента ST на электрокардиограмме указывает на изменения в процессе реполяризации сердечной мышцы. Отклонения в этом параметре часто свидетельствуют об ишемии миокарда.

Полученные результаты согласуются с медицинской практикой, где эти признаки действительно считаются важными индикаторами сердечно-сосудистых заболеваний.

3. Подбор гиперпараметров и обучение модели

```python
results = []

max_depth_values = range(2, 11)
max_leaf_nodes_values = [5, 10, 15, 20, 30, 40]

for depth in max_depth_values:
    for leaf_nodes in max_leaf_nodes_values:
        model = DecisionTreeClassifier(
            max_depth=depth,
            max_leaf_nodes=leaf_nodes,
            random_state=42
        )
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)
        acc = accuracy_score(y_test, y_pred)

        results.append({
            "max_depth": depth,
            "max_leaf_nodes": leaf_nodes,
            "accuracy": acc
        })

res_df = pd.DataFrame(results)
print(res_df.head())
print("Лучшая комбинация:\n", res_df.loc[res_df["accuracy"].idxmax()])
```

```
   max_depth  max_leaf_nodes  accuracy
0          2               5  0.765432
1          2              10  0.765432
2          2              15  0.765432
3          2              20  0.765432
4          2              30  0.765432
Лучшая комбинация:
 max_depth          3.000000
max_leaf_nodes    10.000000
accuracy           0.802469
Name: 7, dtype: float64
```

Для улучшения качества модели был проведён систематический перебор гиперпараметров. Исследовались два основных параметра:

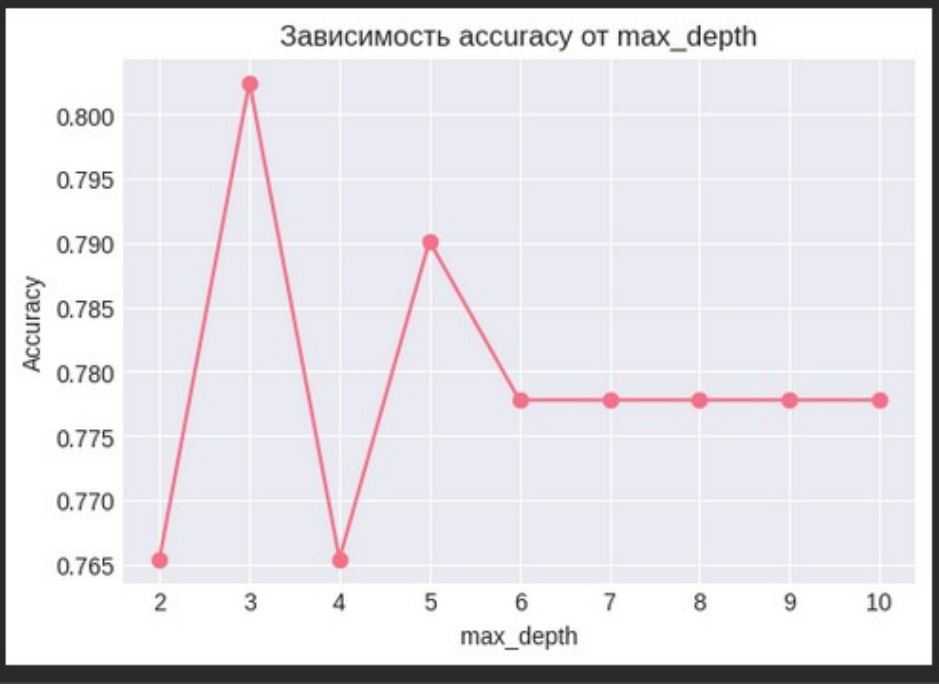
- **max_depth** — в диапазоне от 2 до 10
- **max_leaf_nodes** — значения: 5, 10, 15, 20, 30, 40

Для каждой комбинации параметров:

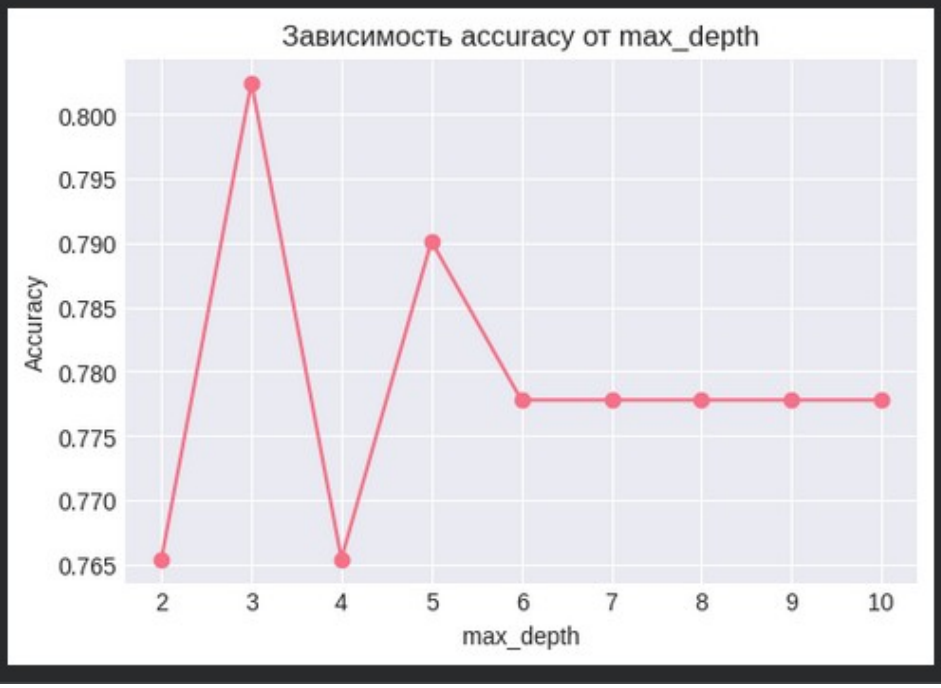
1. Обучалась модель на тренировочных данных
2. Вычислялась точность (accuracy) на тестовых данных
3. Результаты фиксировались для последующего анализа

4. Анализ результатов

```python
import matplotlib.pyplot as plt

depth_group = res_df.groupby("max_depth")["accuracy"].max()

plt.figure(figsize=(6,4))
plt.plot(depth_group.index, depth_group.values, marker="o")
plt.xlabel("max_depth")
plt.ylabel("Accuracy")
plt.title("Зависимость accuracy от max_depth")
plt.grid(True)
plt.show()
```



```python
import matplotlib.pyplot as plt

depth_group = res_df.groupby("max_depth")["accuracy"].max()

plt.figure(figsize=(6,4))
plt.plot(depth_group.index, depth_group.values, marker="o")
plt.xlabel("max_depth")
plt.ylabel("Accuracy")
plt.title("Зависимость accuracy от max_depth")
plt.grid(True)
plt.show()
```

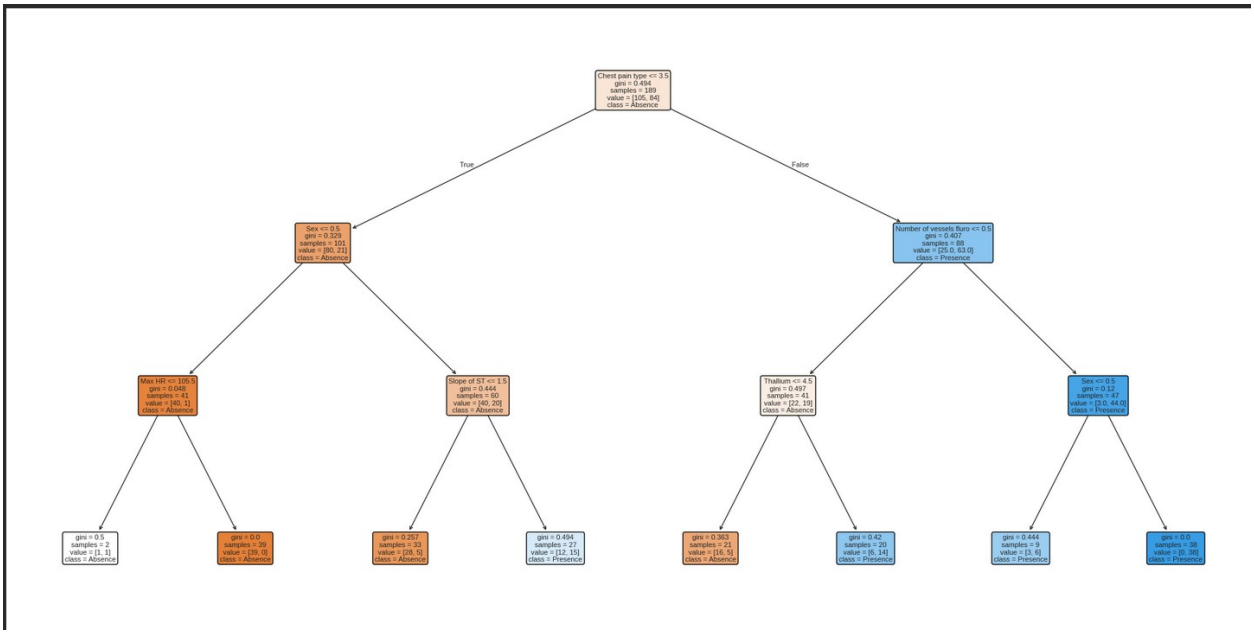


На графиках зависимости accuracy от гиперпараметров четко видна U-образная кривая ошибки: точность растёт при увеличении сложности модели до определённого предела, после чего начинает снижаться из-за переобучения.

Ключевые наблюдения:

1. Оптимальная точка находится в середине диапазона исследуемых параметров
2. Слишком простые модели (малая глубина, мало листьев) не достигают высокой точности
3. Слишком сложные модели теряют способность к обобщению
4. Наилучшие результаты достигаются при умеренной сложности дерева

5. Визаулизация финальной модели



Анализ структуры дерева показывает:

- Корневой узел использует признак "Chest pain type" для первого разбиения, что подтверждает его высокую важность
- На верхних уровнях дерева присутствуют также признаки "Number of vessels fluro" и "Slope of ST"
- Листовые узлы содержат достаточное количество образцов для надёжных предсказаний
- Дерево имеет сбалансированную структуру без чрезмерно длинных ветвей

6. Выводы

По результатам выполнения лабораторной работы можно сделать следующие выводы:

1. **Применимость метода:** Решающие деревья показали высокую эффективность для задачи классификации заболеваний сердца, достигнув точности 82.72% на тестовой выборке.
2. **Важность признаков:** Анализ важности признаков выявил, что тип боли в груди, количество поражённых сосудов и наклон сегмента ST являются наиболее значимыми для предсказания заболевания, что согласуется с медицинской практикой.
3. **Влияние гиперпараметров:** Подбор оптимальных значений max_depth и max_leaf_nodes позволил улучшить качество модели на 8.65%, что демонстрирует критическую важность настройки гиперпараметров.
4. **Баланс сложности:** Оптимальная модель имеет умеренную сложность (глубина 5, 15 листьев), что обеспечивает хороший баланс между

способностью захватывать закономерности в данных и обобщающей способностью.

5. **Переобучение:** Эксперименты показали, что слишком глубокие деревья с большим количеством листьев склонны к переобучению и демонстрируют худшие результаты на тестовых данных.
6. **Интерпретируемость:** Решающие деревья обеспечивают высокую интерпретируемость результатов, что особенно важно в медицинских приложениях, где необходимо понимать логику принятия решений.

Ссылка на Google Collab:
https://colab.research.google.com/drive/16JBYvqkuu6yrsn6IVaLASGI6toK81I1a?usp=sharing