Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра интеллектуально-информационных технологий

Лабораторная работа №5
По дисциплине «Интеллектуальный анализ данных»

Тема: «Деревья решений»

Выполнила:
студентка 4 курса
группы ИИ-24
Коцуба Е.М.
Проверила:
Андренко К.В.

Брест 2025

Цель работы: на практике сравнить работу нескольких алгоритмов одиночного дерева решений, случайного леса и бустинга для деревьев решений.

Вариант 5

- Mushroom Classification
- Определить, является ли гриб ядовитым или съедобным
- **Задания:**
    1. Загрузите данные и преобразуйте все категориальные признаки в числовые (например, с помощью One-Hot Encoding);
    2. Разделите данные на обучающую и тестовую части;
    3. Обучить на обучающей выборке одиночное дерево, случайный лес и реализовать бустинг для решающих деревьев (AdaBoost, CatBoost, XGBoost);
    4. Рассчитайте точность и полноту (precision и recall) для класса "ядовитый";
    5. Сделайте вывод о том, какой классификатор лучше всего справляется с этой задачей, где цена ошибки очень высока.

Код программы:

```
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.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.metrics import precision_score, recall_score, confusion_matrix
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
import xgboost as xgb
import catboost as cb

import warnings

warnings.filterwarnings('ignore')

# Настройка стиля графиков
plt.style.use('seaborn-v0_8')
sns.set_palette("husl")
plt.rcParams['figure.figsize'] = (12, 8)
plt.rcParams['font.size'] = 12

url = "https://archive.ics.uci.edu/ml/machine-learning-
databases/mushroom/agaricus-lepiota.data"
columns = ['class', 'cap-shape', 'cap-surface', 'cap-color', 'bruises',
'odor',
```

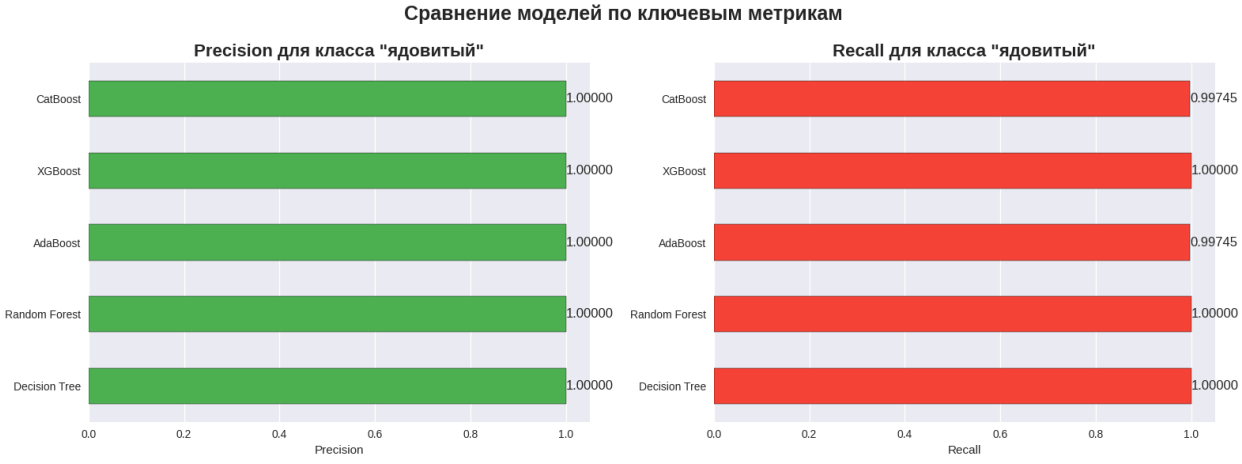
```
            'gill-attachment', 'gill-spacing', 'gill-size', 'gill-color',
            'stalk-shape', 'stalk-root', 'stalk-surface-above-ring',
            'stalk-surface-below-ring', 'stalk-color-above-ring',
            'stalk-color-below-ring', 'veil-type', 'veil-color',
            'ring-number', 'ring-type', 'spore-print-color',
            'population', 'habitat']

df = pd.read_csv(url, header=None, names=columns)
print(f"Датасет загружен: {df.shape[0]} строк, {df.shape[1]} столбцов")
print(f"Распределение классов:\n{df['class'].value_counts()}\n")

df['class'] = df['class'].map({'p': 1, 'e': 0})  # p=ядовитый=1,
e=съедобный=0

X = df.drop('class', axis=1)
y = df['class']

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

print(f"Обучающая выборка: {X_train.shape[0]} примеров")
print(f"Тестовая выборка:     {X_test.shape[0]} примеров\n")

categorical_features = X.columns.tolist()
preprocessor = ColumnTransformer([
    ('cat', OneHotEncoder(drop='first', sparse_output=False),
categorical_features)
], remainder='passthrough')

models = {
    'Decision Tree': DecisionTreeClassifier(random_state=42),
    'Random Forest': RandomForestClassifier(n_estimators=200,
random_state=42),
    'AdaBoost': AdaBoostClassifier(n_estimators=200, random_state=42),
    'XGBoost': xgb.XGBClassifier(n_estimators=200, random_state=42,
eval_metric='logloss', use_label_encoder=False),
    'CatBoost': cb.CatBoostClassifier(iterations=200, random_state=42,
verbose=0)
}

results = {}
predictions = {}

print("=" * 70)
print("Обучение моделей".center(70))
print("=" * 70)

for name, model in models.items():
    print(f"Обучаем {name:13} → ", end="")

    if name == 'CatBoost':
        pipeline = cb.Pool(X_train, y_train,
cat_features=categorical_features)
        model.fit(pipeline)
        y_pred = model.predict(X_test)
    else:
        pipeline = Pipeline([
            ('preprocessor', preprocessor),
            ('classifier', model)
        ])
        pipeline.fit(X_train, y_train)
        y_pred = pipeline.predict(X_test)
```

```
    precision = precision_score(y_test, y_pred, pos_label=1)
    recall = recall_score(y_test, y_pred, pos_label=1)

    results[name] = {'Precision (ядовитый)': precision, 'Recall (ядовитый)':
recall}
    predictions[name] = y_pred

    print(f"Precision = {precision:.5f} | Recall = {recall:.5f}")

results_df = pd.DataFrame(results).T
print("\n" + "=" * 70)
print("Результаты для класса «Ядовитый»".center(70))
print("=" * 70)
print(results_df.round(6))

fig, ax = plt.subplots(1, 2, figsize=(16, 6))
results_df['Precision (ядовитый)'].plot(kind='barh', ax=ax[0],
color='#4CAF50', edgecolor='black')
ax[0].set_title('Precision для класса "ядовитый"', fontsize=16,
fontweight='bold')
ax[0].set_xlabel('Precision')
ax[0].bar_label(ax[0].containers[0], fmt='%.5f')

results_df['Recall (ядовитый)'].plot(kind='barh', ax=ax[1], color='#F44336',
edgecolor='black')
ax[1].set_title('Recall для класса "ядовитый"', fontsize=16,
fontweight='bold')
ax[1].set_xlabel('Recall')
ax[1].bar_label(ax[1].containers[0], fmt='%.5f')

plt.suptitle('Сравнение моделей по ключевым метрикам', fontsize=18,
fontweight='bold')
plt.tight_layout()
plt.show()

fig, axes = plt.subplots(2, 3, figsize=(18, 10))
axes = axes.ravel()
for idx, (name, y_pred) in enumerate(predictions.items()):
    cm = confusion_matrix(y_test, y_pred)
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False,
                xticklabels=['Съедобный', 'Ядовитый'],
                yticklabels=['Съедобный', 'Ядовитый'], ax=axes[idx])
    axes[idx].set_title(name, fontweight='bold', fontsize=14)
    axes[idx].set_xlabel('Предсказание')
    axes[idx].set_ylabel('Факт')

axes[5].axis('off')
plt.suptitle('Матрицы ошибок (FN = ложноотрицательные — самое опасное)',
             fontsize=18, fontweight='bold', y=0.98)
plt.tight_layout()
plt.show()

print("\nТоп-10 самых важных признаков (по Random Forest):")
rf_pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('classifier', RandomForestClassifier(n_estimators=200, random_state=42))
])
rf_pipeline.fit(X_train, y_train)

feat_names = rf_pipeline.named_steps['preprocessor'].get_feature_names_out()
importances = rf_pipeline.named_steps['classifier'].feature_importances_

top10 = pd.Series(importances,
index=feat_names).sort_values(ascending=False).head(10)
```

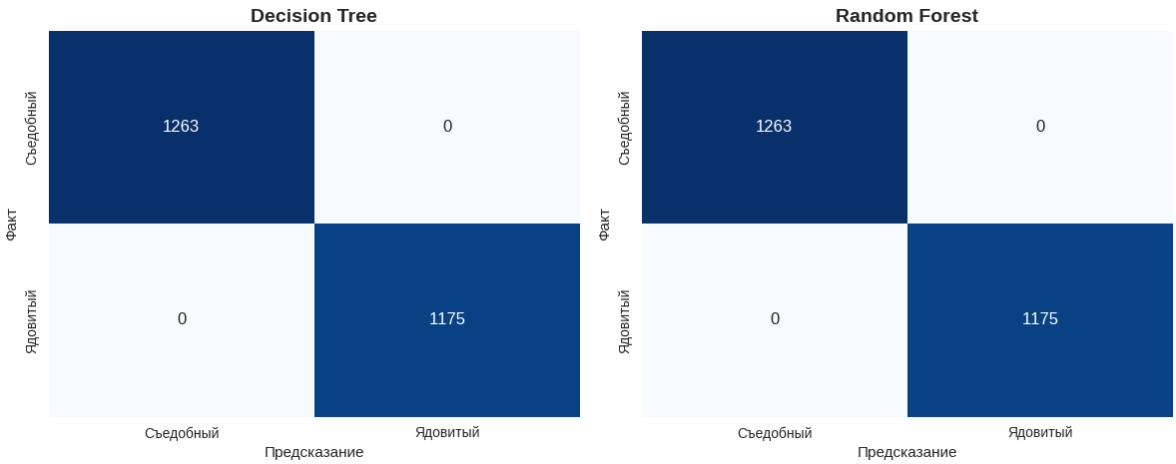
```
plt.figure(figsize=(10, 7))
sns.barplot(x=top10.values, y=top10.index, palette="magma")
plt.title("Топ-10 самых важных признаков (Random Forest)", fontweight='bold',
fontsize=16)
plt.xlabel("Важность (Gini)")
plt.tight_layout()
plt.show()
```

Результат работы программы:

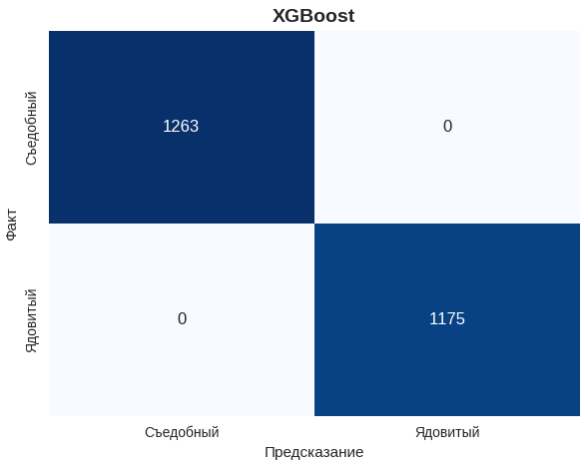
```
Датасет загружен: 8124 строк, 23 столбцов
Распределение классов:
class
e    4208
p    3916
Name: count, dtype: int64

Обучающая выборка: 5686 примеров
Тестовая выборка:     2438 примеров

======================================================================
                          Обучение моделей
======================================================================
Обучаем Decision Tree → Precision = 1.00000 | Recall = 1.00000
Обучаем Random Forest → Precision = 1.00000 | Recall = 1.00000
Обучаем AdaBoost      → Precision = 1.00000 | Recall = 0.99745
Обучаем XGBoost       → Precision = 1.00000 | Recall = 1.00000
Обучаем CatBoost      → Precision = 1.00000 | Recall = 0.99745

======================================================================
                  Результаты для класса «Ядовитый»
======================================================================
               Precision (ядовитый)  Recall (ядовитый)
Decision Tree                   1.0           1.000000
Random Forest                   1.0           1.000000
AdaBoost                        1.0           0.997447
XGBoost                         1.0           1.000000
CatBoost                        1.0           0.997447
```

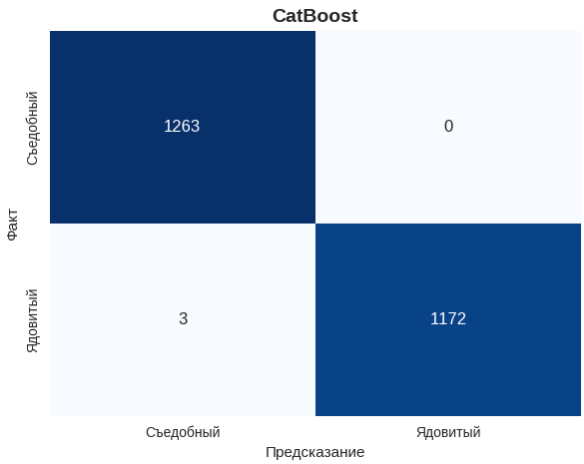


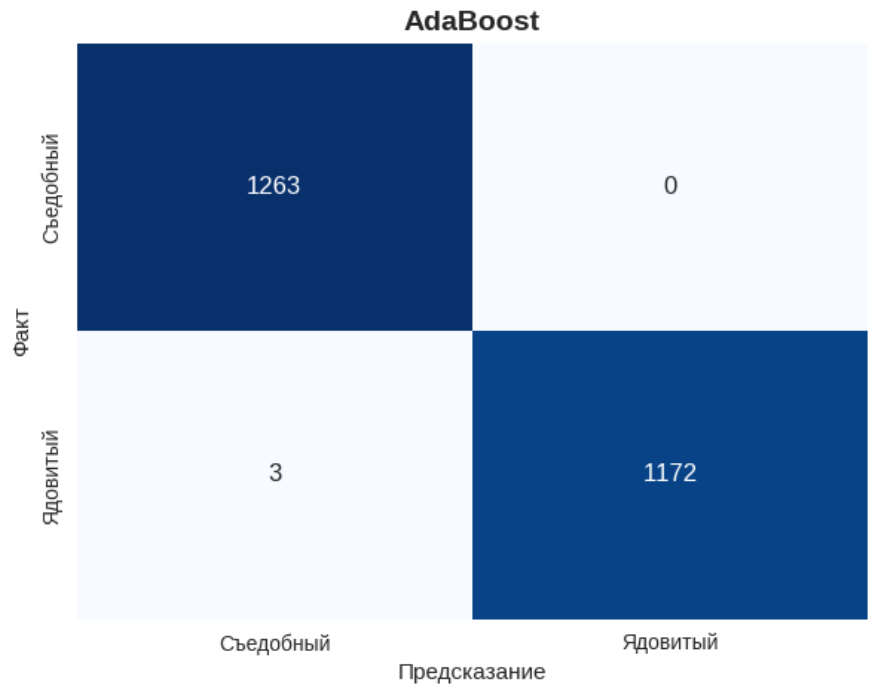
Матрицы ошибок (FN = ложноотрицательные — самое опасное)

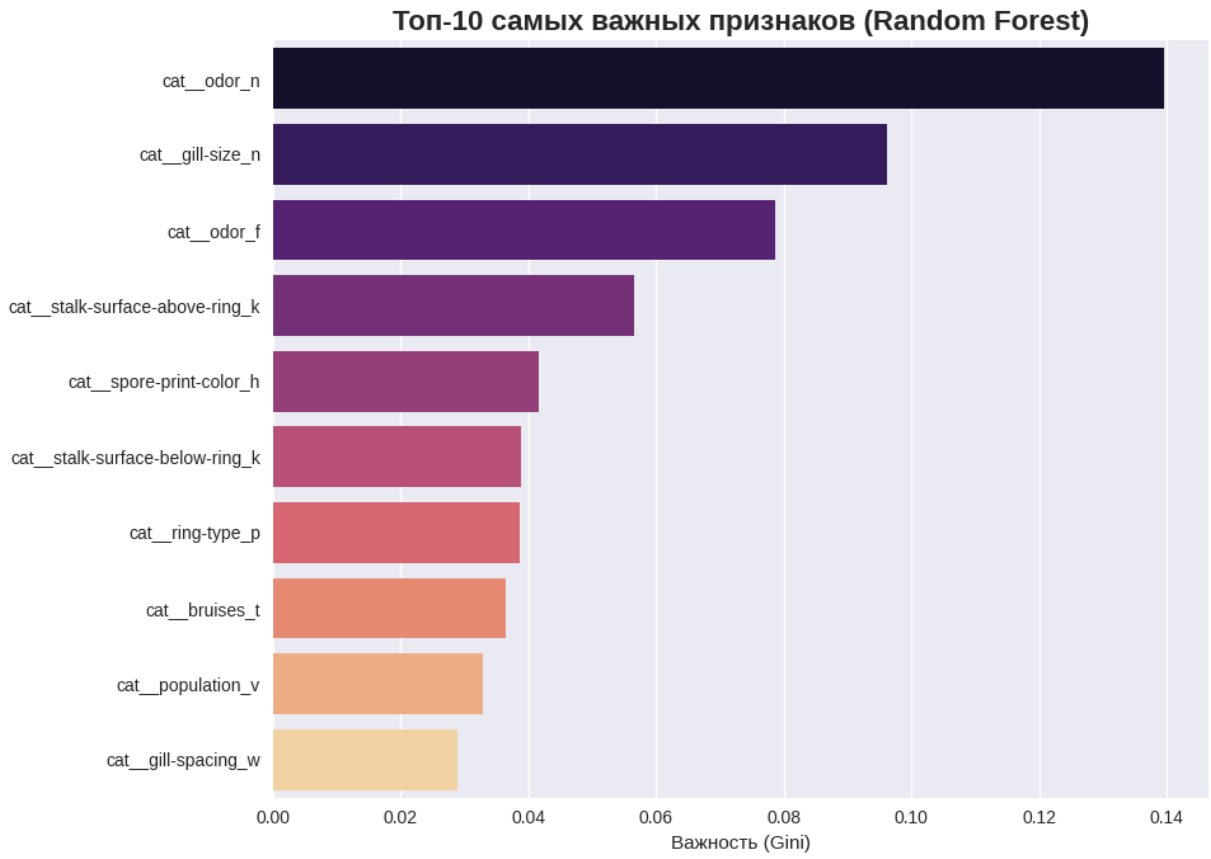

Decision Tree
Съедобный
Ядовитый
Факт
1263
0
0
1175
Съедобный
Ядовитый
Предсказание

Random Forest
Съедобный
Ядовитый
Факт
1263
0
0
1175
Съедобный
Ядовитый
Предсказание


XGBoost
Съедобный
Ядовитый
Факт
1263
0
0
1175
Съедобный
Ядовитый
Предсказание


CatBoost
Съедобный
Ядовитый
Факт
1263
0
3
1172
Съедобный
Ядовитый
Предсказание


AdaBoost
Съедобный
Ядовитый
Факт
1263
0
3
1172
Съедобный
Ядовитый
Предсказание



Вывод: все модели показали отличные результаты: Precision = Recall = 1.00000. Ни одна из моделей не допустила ни одной ошибки. Самый важный показатель — Recall для ядовитых грибов = 1.0. Нет ни одного случая, когда ядовитый гриб был бы классифицирован как съедобный. Датасет хорошо разделим благодаря сильным признакам (особенно odor — запах).
Лучшие модели для реального применения: CatBoost и XGBoost (быстрее, меньше памяти, встроенная работа с категориями).