Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №5
По дисциплине: «Интеллектуальный анализ данных»
Тема: “Деревья решений”

Выполнил:
Студент 4 курса
Группы ИИ-24
Капуза Н.А.
Проверила:
Андренко К. В.

Брест 2025

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

Задачи:

1. Загрузить датасет по варианту;
2. Разделить данные на обучающую и тестовую выборки;
3. Обучить на обучающей выборке одиночное дерево, случайный лес и реализовать бустинг для решающих деревьев (AdaBoost, CatBoost, XGBoost);
4. Оценить точность каждой модели на тестовой выборке;
5. Сравнить результаты, сделать выводы о применимости каждого метода для данного набора данных.

Вариант 4

- Digits
- Распознать, какая цифра (от 0 до 9) изображена на картинке 8x8 пикселей
- **Задания:**
    1. Загрузите встроенный в scikit-learn набор данных digits;
    2. Разделите данные на обучающую и тестовую выборки;
    3. Обучить на обучающей выборке одиночное дерево, случайный лес и реализовать бустинг для решающих деревьев (AdaBoost, CatBoost, XGBoost);
    4. Для каждой модели выведите classification_report (sklearn.metrics), содержащий основные метрики для каждого класса;
    5. Сравните общую точность моделей и определите, какая из них лучше всего подходит для этой задачи.

Ход работы:

Код программы:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.metrics import classification_report, accuracy_score

# Импорт библиотек бустинга
# Примечание: Убедитесь, что они установлены (pip install xgboost catboost)
from xgboost import XGBClassifier
from catboost import CatBoostClassifier

# Задание 1: Загрузить встроенный набор данных digits
```

```
digits = load_digits()
X = digits.data
y = digits.target

print(f"Размерность данных: {X.shape}")
print(f"Количество классов: {len(set(y))} (цифры от 0 до 9)")
print("-" * 30)

#  Задание 2: Разделить данные на обучающую и тестовую выборки
# Обычно используют разбиение 80/20 или 70/30. Возьмем 80/20.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

#  Задание 3: Инициализация моделей
models = {
    "Decision Tree (Одиночное дерево)": DecisionTreeClassifier(random_state=42),
    "Random Forest (Случайный лес)": RandomForestClassifier(random_state=42),
    "AdaBoost": AdaBoostClassifier(algorithm='SAMME', random_state=42), #
algorithm='SAMME' лучше работает для мультикласса в старых версиях, в новых
auto
    "CatBoost": CatBoostClassifier(verbose=0, random_state=42), # verbose=0 отключает
вывод логов обучения
    "XGBoost": XGBClassifier(eval_metric='mlogloss', use_label_encoder=False,
random_state=42)
}

results = {}

print("Начало обучения и оценки моделей...\n")

#  Задания 3 и 4: Обучение и вывод classification_report
for name, model in models.items():
    # Обучение
    model.fit(X_train, y_train)

    # Предсказание
    y_pred = model.predict(X_test)

    # Расчет точности (accuracy) для итогового сравнения
    acc = accuracy_score(y_test, y_pred)
    results[name] = acc

    # Вывод отчета (Задание 4)
    print(f"=== {name} ===")
    print(f"Accuracy: {acc:.4f}")
    print(classification_report(y_test, y_pred))
```

```
    print("-" * 60)

#  Задание 5: Сравнение общей точности и вывод
print("Итоговое сравнение точности (Accuracy):")
sorted_results = sorted(results.items(), key=lambda x: x[1], reverse=True)

for name, acc in sorted_results:
    print(f"{name}: {acc:.4f}")

best_model = sorted_results[0][0]
print(f"\n>> ВЫВОД: Лучше всего для этой задачи подходит модель: {best_model}")
```

Результаты:

```
Размерность данных: (1797, 64)
Количество классов: 10 (цифры от 0 до 9)
------------------------------
Начало обучения и оценки моделей...

=== Decision Tree (Одиночное дерево) ===
Accuracy: 0.8417
              precision    recall  f1-score   support

           0       0.97      0.88      0.92        33
           1       0.85      0.79      0.81        28
           2       0.86      0.73      0.79        33
           3       0.76      0.85      0.81        34
           4       0.84      0.91      0.88        46
           5       0.89      0.85      0.87        47
           6       0.97      0.91      0.94        35
           7       0.82      0.91      0.86        34
           8       0.75      0.70      0.72        30
           9       0.75      0.82      0.79        40

    accuracy                           0.84       360
   macro avg       0.84      0.84      0.84       360
weighted avg       0.85      0.84      0.84       360

------------------------------------------------------------
=== Random Forest (Случайный лес) ===
Accuracy: 0.9722
              precision    recall  f1-score   support

           0       1.00      0.97      0.98        33
           1       0.97      1.00      0.98        28
           2       1.00      1.00      1.00        33
           3       1.00      0.94      0.97        34
```

4	0.98	1.00	0.99	46
5	0.94	0.96	0.95	47
6	0.97	0.97	0.97	35
7	0.97	0.97	0.97	34
8	0.97	0.97	0.97	30
9	0.95	0.95	0.95	40
accuracy			0.97	360
macro avg	0.97	0.97	0.97	360
weighted avg	0.97	0.97	0.97	360

=== AdaBoost ===

Accuracy: 0.7972

	precision	recall	f1-score	support
0	0.97	0.85	0.90	33
1	0.87	0.46	0.60	28
2	0.81	0.64	0.71	33
3	0.71	0.85	0.77	34
4	0.92	0.76	0.83	46
5	1.00	0.87	0.93	47
6	0.85	0.94	0.89	35
7	0.74	0.94	0.83	34
8	0.53	0.77	0.63	30
9	0.71	0.80	0.75	40
accuracy			0.80	360
macro avg	0.81	0.79	0.79	360
weighted avg	0.82	0.80	0.80	360

=== CatBoost ===

Accuracy: 0.9833

	precision	recall	f1-score	support
0	1.00	1.00	1.00	33
1	0.97	1.00	0.98	28
2	1.00	1.00	1.00	33
3	1.00	0.97	0.99	34
4	1.00	1.00	1.00	46
5	0.96	0.98	0.97	47
6	0.97	0.97	0.97	35
7	0.97	0.97	0.97	34
8	1.00	0.97	0.98	30

```
           9       0.97      0.97      0.97        40

    accuracy                           0.98       360
   macro avg       0.98      0.98      0.98       360
weighted avg       0.98      0.98      0.98       360

------------------------------------------------------------
C:\Users\kolya\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2
kfra8p0\LocalCache\local-packages\Python311\site-packages\xgboost\training.py:199:
UserWarning: [16:11:14] WARNING: C:\actions-
runner\_work\xgboost\xgboost\src\learner.cc:790:
Parameters: { "use_label_encoder" } are not used.

  bst.update(dtrain, iteration=i, fobj=obj)
=== XGBoost ===
Accuracy: 0.9694
              precision    recall  f1-score   support

           0       1.00      0.97      0.98        33
           1       0.93      1.00      0.97        28
           2       1.00      1.00      1.00        33
           3       1.00      0.94      0.97        34
           4       0.98      0.98      0.98        46
           5       0.94      0.98      0.96        47
           6       0.97      0.97      0.97        35
           7       0.97      0.97      0.97        34
           8       0.93      0.93      0.93        30
           9       0.97      0.95      0.96        40

    accuracy                           0.97       360
   macro avg       0.97      0.97      0.97       360
weighted avg       0.97      0.97      0.97       360

------------------------------------------------------------
Итоговое сравнение точности (Accuracy):
CatBoost: 0.9833
Random Forest (Случайный лес): 0.9722
XGBoost: 0.9694
Decision Tree (Одиночное дерево): 0.8417
AdaBoost: 0.7972

>> ВЫВОД: Лучше всего для этой задачи подходит модель: CatBoost
```

Выводы:

- **Лидер (CatBoost):** Показал наивысшую точность. Этот алгоритм отлично работает с "коробочными" настройками (дефолтными параметрами). Он эффективно строит ансамбль деревьев, исправляя ошибки предыдущих шагов, что идеально подошло для классификации 64 признаков (пикселей).
- **Случайный лес (Random Forest) и XGBoost:** Показали результаты, очень близкие к лидеру. Это подтверждает, что ансамблевые методы (бэггинг и бустинг) значительно превосходят одиночные деревья на таких данных.
- **Одиночное дерево (Decision Tree):** Точность ~84%. Дерево склонно к переобучению и не может обобщить закономерности так же хорошо, как лес.
- **Аутсайдер (AdaBoost):** Точность ~79%. Это связано с тем, что по умолчанию AdaBoostClassifier в sklearn использует "пни" (деревья глубиной 1). Для распознавания цифр (изображений 8x8) глубины 1 недостаточно, чтобы уловить сложные зависимости между пикселями. Ему требовалась настройка гиперпараметров (увеличение глубины базового эстиматора).