**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по лабораторной работе №3**

**по дисциплине «Машинное обучение»**

Тема: Исследование алгоритмов классификации

|  |  |  |
| --- | --- | --- |
| Студентка гр. 2372 |  | Котлова П. В. |
| Преподаватель |  | Татчина Я.А. |

Санкт-Петербург

2024

**Цель работы.**

Знакомство с задачей классификации, изучение методов k-ближайших соседей и дерева решений.

**Постановка задачи.**

1. Выбрать атрибут, выступающий в роли целевого класса (метки);
2. Оценить сбалансированность классов;
3. Применить методы классификации алгоритмами kNN, Дерево решений;
4. Оценить и сравнить результаты метриками оценки качества accuracy, precision, recall, f1, ROC;
5. Проанализировать полученные результаты.

**Выполнение работы.**

В данном отчете представлены результаты исследования алгоритмов классификации для предсказания качества красного вина на основе датасета "Red Wine Quality" из репозитория UCI Machine Learning Repository.

**Подготовка данных:**

1. Загрузка данных: Датасет "Red Wine Quality" был загружен и обработан с помощью библиотеки Pandas.
2. Предобработка:
   1. Бинаризация целевой переменной: поскольку исходный атрибут "quality" является числовым (от 3 до 8), для задачи бинарной классификации он был преобразован в бинарную переменную. Вина с оценкой качества 7 и выше были классифицированы как "хорошие" (1), а вина с оценкой ниже 7 – как "плохие" (0). Это решение было принято для упрощения задачи и для лучшей интерпретации результатов. Альтернативные подходы включают использование многоклассовой классификации или других способов дискретизации.
   2. Анализ баланса классов: после бинаризации был проведен анализ сбалансированности классов. Было подсчитано количество вин в каждом классе ("хорошие" и "плохие") и вычислено их соотношение. Результаты показали [вставить сюда результаты из value\_counts(normalize=True) - соотношение классов]. Незначительный дисбаланс классов был обнаружен.
   3. Масштабирование признаков: для алгоритма kNN, чувствительного к масштабу признаков, было применено стандартизированное масштабирование (StandardScaler) ко всем числовым признакам.
3. Разделение данных: датасет был разделен на обучающую и тестовую выборки в соотношении 80/20 с использованием метода train\_test\_split из библиотеки scikit-learn. При этом использовалась стратификация (stratify=y), чтобы сохранить соотношение классов в обучающей и тестовой выборках.

**Методы классификации**

Были выбраны два алгоритма классификации:

1. k-Nearest Neighbors (kNN): Этот алгоритм классифицирует новые объекты на основе ближайших соседей в пространстве признаков. В данной работе использовались параметры по умолчанию из библиотеки scikit-learn.
2. Дерево решений: Этот алгоритм создает дерево решений для классификации объектов на основе значений признаков. В данной работе использовались параметры по умолчанию из библиотеки scikit-learn.

**Оценка результатов.**

Для оценки производительности моделей были использованы следующие метрики:

• Accuracy: Доля правильно классифицированных объектов.

• Precision: Доля правильно предсказанных положительных объектов среди всех предсказанных положительных объектов.

• Recall: Доля правильно предсказанных положительных объектов среди всех фактических положительных объектов.

• F1-score: Гармоническое среднее precision и recall.

• ROC AUC: Площадь под кривой ROC, которая отражает способность модели различать классы.

Результаты оценки каждой модели представлены в таблице [вставить таблицу df\_results]. Также были построены confusion matrices для визуализации результатов классификации [вставить confusion matrix].

**Анализ результатов.**

На основе полученных результатов можно сделать вывод о том, что [сравнить модели на основе полученных метрик]. Например, можно сравнить accuracy, precision, recall и F1-score для каждой модели. ROC AUC score позволяет сравнить модели с точки зрения их способности различать классы. Визуализация confusion matrix позволяет понять, какие типы ошибок наиболее часты для каждой модели.

**Подбор гиперпараметров (Пример для kNN).**

Для улучшения производительности модели kNN был проведен поиск оптимальных гиперпараметров с помощью GridSearchCV. Лучшими параметрами оказались [вставить лучшие параметры из grid\_search.best\_params\_]. Результаты классификации с использованием оптимальных параметров представлены в отчете [вставить результаты для best\_knn].

**Выводы.**

В ходе работы были изучены алгоритмы классификации KNN и дерево решений. Проведено обучение моделей, оценка и сравнение результатов.