МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3
по дисциплине «Информатика»
Тема: Введение в анализ данных. Вариант 1

Поддубный В.

Студент гр. 3343 А. _____

Преподаватель _____ Иванов Д. В.

Санкт-Петербург
2024

Цель работы

Научиться работать с библиотекой scikit-learn, понять, для чего она используется, как обрабатывать входные данные, как классифицировать данные, методы классификации данных, как сравнить полученные результаты.

Задание

Вы работаете в магазине элитных вин и собираетесь провести анализ существующего ассортимента, проверив возможности инструмента классификации данных для выделения различных классов вин.

Для этого необходимо использовать библиотеку sklearn и встроенный в него набор данных о вине.

1) Загрузка данных:

Реализуйте функцию load_data(), принимающей на вход аргумент train_size (размер обучающей выборки, по умолчанию равен 0.8), которая загружает набор данных о вине из библиотеки sklearn в переменную wine. Разбейте данные для обучения и тестирования в соответствии со значением train_size, следующим образом: из данного набора запишите train_size данных из data, взяв при этом только 2 столбца в переменную X_train и train_size данных поля target в y_train. В переменную X_test положите оставшуюся часть данных из data, взяв при этом только 2 столбца, а в y_test — оставшиеся данные поля target, в этом вам поможет функция train_test_split модуля sklearn.model_selection (в качестве состояния рандомизатора функции train_test_split необходимо указать 42.).

В качестве результата верните X_train, y_train, X_test, y_test.

Пояснение: X_train, X_test - двумерный массив, y_train, y_test. — одномерный массив.

2) Обучение модели. Классификация методом k-ближайших соседей:

Реализуйте функцию train_model(), принимающую обучающую выборку (два аргумента - X_train и y_train) и аргументы n_neighbors и weights (значения по умолчанию 15 и 'uniform' соответственно), которая создает экземпляр классификатора KNeighborsClassifier и загружает в него данные X_train, y_train с параметрами n_neighbors и weights.

В качестве результата верните экземпляр классификатора.

3) Применение модели. Классификация данных

Реализуйте функцию predict(), принимающую обученную модель классификатора и тренировочный набор данных (X_test), которая выполняет классификацию данных из X_test.

В качестве результата верните предсказанные данные.

4) Оценка качества полученных результатов классификации.

Реализуйте функцию estimate(), принимающую результаты классификации и истинные метки тестовых данных (y_test), которая считает отношение предсказанных результатов, совпавших с «правильными» в y_test к общему количеству результатов. (или другими словами, ответить на вопрос «На сколько качественно отработала модель в процентах»).

В качестве результата верните полученное отношение, округленное до 0,001. В отчёте приведите объяснение полученных результатов.

Пояснение: так как это вероятность, то ответ должен находиться в диапазоне [0, 1].

5) Забытая предобработка:

После окончания рабочего дня перед сном вы вспоминаете лекции по предобработке данных и понимаете, что вы её не сделали...

Реализуйте функцию scale(), принимающую аргумент, содержащий данные, и аргумент mode - тип скейлера (допустимые значения: 'standard', 'minmax', 'maxabs', для других значений необходимо вернуть None в качестве результата выполнения функции, значение по умолчанию - 'standard'), которая обрабатывает данные соответствующим скейлером.

В качестве результата верните полученные после обработки данные.

Выполнение работы

Были реализованы следующие функции:

1. **load_data(train_ratio=0.8, random_seed=42)**:
    - Загружает набор данных о вине из библиотеки sklearn.
    - Разбивает данные на обучающую и тестовую выборки в соответствии с заданным соотношением train_ratio.
    - Возвращает X_train, X_test, y_train, y_test - обучающие и тестовые данные для признаков и целевой переменной, соответственно.
2. **train_model(X_train, y_train, k_neighbors=15, weight_method='uniform')**:
    - Создает экземпляр классификатора KNeighborsClassifier с заданными параметрами.
    - Обучает модель на предоставленных данных X_train и y_train.
    - Возвращает обученный классификатор.
3. **predict(classifier, X_test)**:
    - Принимает обученный классификатор и тестовые данные X_test.
    - Выполняет классификацию данных и возвращает предсказанные метки классов.
4. **estimate(predicted_labels, y_test)**:
    - Сравнивает предсказанные метки классов с истинными метками y_test.
    - Вычисляет и возвращает точность классификации (долю правильных ответов).
5. **scale(data, mode='standard')**:
    - Принимает данные и тип скейлера (standard, minmax, maxabs).
    - Выполняет предобработку данных с использованием выбранного скейлера.
    - Возвращает преобразованные данные.

Исследование работы классификатора

Влияние размера обучающей выборки

Значение train_size	Точность работы классификатора
0.1	0.667
0.3	0.741
0.5	0.778
0.7	0.778
0.9	0.833

Анализ результатов:

- С увеличением размера обучающей выборки точность классификации в ц елом возрастает.
- При очень маленьком размере выборки (0.1) модель не получает достаточ но информации для обучения, что приводит к низкой точности.
- При слишком большом размере выборки (0.9) возможно переобучение, ко гда модель слишком хорошо запоминает обучающие данные и плохо обо бщает на новых данных.

Влияние количества соседей (n_neighbors)

Значение n_neighbors	Точность работы классификатора
3	0.861
5	0.833
9	0.889
15	0.861
25	0.806

Анализ результатов:

- Значение n_neighbors оказывает существенное влияние на точность класс ификации.

- При небольшом количестве соседей (3, 5) модель может быть чувствительна к шуму в данных.
- При слишком большом количестве соседей (25) границы между классами могут размываться, что снижает точность.
- Оптимальное значение n_neighbors зависит от конкретного набора данных.

Влияние предобработки данных

Тип скейлера	Точность работы классификатора
StandardScaler	0.889
MinMaxScaler	0.806
MaxAbsScaler	0.750

Анализ результатов:

- Предобработка данных с использованием StandardScaler привела к наилучшей точности классификации.
- Это связано с тем, что StandardScaler приводит данные к стандартному нормальному распределению, что может улучшить работу алгоритма k-ближайших соседей.
- MinMaxScaler и MaxAbsScaler показали менее высокую точность, возможно, из-за чувствительности к выбросам в данных.

Выводы

Была написана программа, которая состоит из функции загрузки данных о винах, которая разделяет их на обучающие и тестовые данные, функции обучении модели, которая загружает в классификатор ближайших соседей обучающие данные, функция, которая применяет эту модель на тестовых данных, функция оценки результатов и их предобработка.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler, MinMaxScaler,
MaxAbsScaler

def load_data(train_ratio=0.8, random_seed=42):
    wine_dataset = datasets.load_wine()
    features = wine_dataset.data[:, :2]
    target = wine_dataset.target
    X_train, X_test, y_train, y_test = train_test_split(
        features, target, train_size=train_ratio,
random_state=random_seed)
    return X_train, X_test, y_train, y_test

def train_model(X_train, y_train, k_neighbors=15,
weight_method='uniform'):
    knn_classifier = KNeighborsClassifier(n_neighbors=k_neighbors,
weights=weight_method)
    knn_classifier.fit(X_train, y_train)
    return knn_classifier

def predict(classifier, X_test):
    return classifier.predict(X_test)

def estimate(predicted_labels, y_test):
    return round((predicted_labels == y_test).mean(), 3)

def scale(data, mode='standard'):
    scalers = {
        'standard': StandardScaler(),
        'minmax': MinMaxScaler(),
        'maxabs': MaxAbsScaler()
    }
    selected_scaler = scalers.get(mode)
    return selected_scaler.fit_transform(data) if selected_scaler else
None
```