**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**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3344 |  | Бажуков С.В. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2024

**Цель работы**

Получение базовых навыков работы с инструменты для анализа данных на языке программирования Python.

**Задание**

Вариант 1.

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, X\_test, y\_train, 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 c 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**

load\_data:

Эта функция загружает данные о вине из библиотеки sklearn, разделяя их на обучающую и тестовую выборки. Пользователь может настроить размер обучающей выборки с помощью параметра train\_size.

train\_model:

В данной функции применяется метод K-ближайших соседей для обучения классификатора. Пользователь может указать количество соседей (n\_neighbors) и весовую функцию (weights) для классификатора.

predict:

Эта функция использует обученную модель для предсказания меток классов на тестовой выборке.

estimate:

Данная функция оценивает точность предсказаний, используя метки истинных классов и предсказанные моделью метки. Она вычисляет точность классификации с помощью метрики accuracy\_score и возвращает результат, округленный до трех знаков после запятой.

scale:

Функция scale позволяет масштабировать входные данные в соответствии с выбранным режимом масштабирования: стандартным (standard), мини-максимальным (minmax) или масштабированием по максимальному абсолютному значению (maxabs). Возвращает масштабированный массив данных или None, если режим масштабирования недопустим.

Исследование работы классификатора.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| train\_size | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
| Точность | 0.379 | 0.8 | 0.843 | 0.815 | 0.722 |

С увеличением размера выборки растет и точность классификатора. Однако когда размер выборки достигает 0,7, точность начинает снижаться. Таким образом, можно сделать вывод, что слишком большая выборка также может быть неэффективна для классификации, так как может привести к переобучению модели и увеличению времени обучения.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n\_neighbors | 3 | 5 | 9 | 15 | 25 |
| Точность | 0.861 | 0.833 | 0.861 | 0.861 | 0.833 |

Точность работы классификаторов при разных значениях параметра n\_neighbors изменяется незначительно. Наивысшая точность достигается при значениях n\_neighbors равных 3, 9 и 15, составляя 0.861. При значениях n\_neighbors равных 5 и 25 точность немного ниже, составляя 0.833. Оптимальными значениями являются 3, 9, 15

|  |  |
| --- | --- |
| Scaler | Точность |
| StandardScaler | 0.417 |
| MinMaxScaler | 0.417 |
| MaxAbsScaler | 0.278 |

Точность классификации варьируется в зависимости от выбранного метода масштабирования данных. При использовании стандартного и минимакс-масштабирования точность составляет 0.417, в то время как при максимальном абсолютном масштабировании точность снижается до 0.278.

Исходный код см. в приложении A

**Выводы**

Получены практические навыки использования библиотек, которые включают основные инструменты для анализа данных. Был получен опыт написания программ на Python для анализа данных.

**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.metrics import accuracy\_score

from sklearn.preprocessing import StandardScaler, MinMaxScaler, MaxAbsScaler

def load\_data(train\_size=.8):

wine\_dataset = datasets.load\_wine()

features = wine\_dataset.data

labels = wine\_dataset.target

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

features[:, [0, 1]], labels, train\_size=train\_size, random\_state=42)

return X\_train, X\_test, y\_train, y\_test

def train\_model(X\_train, y\_train, n\_neighbors=15, weights='uniform'):

classifier = KNeighborsClassifier(n\_neighbors=n\_neighbors, weights=weights)

classifier.fit(X\_train, y\_train)

return classifier

def predict(clf, X\_test):

predictions = clf.predict(X\_test)

return predictions

def estimate(res, y\_test):

accuracy = accuracy\_score(y\_true=y\_test, y\_pred=res)

return round(accuracy, 3)

def scale(X, mode="standard"):

if mode not in ["standard", "minmax", "maxabs"]:

return None

scaler = StandardScaler()

if mode == "minmax":

scaler = MinMaxScaler()

elif mode == "maxabs":

scaler = MaxAbsScaler()

scaled = scaler.fit\_transform(X)

return scaled