

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

ОТЧЕТ
по лабораторной работе №3
по дисциплине «Информационные технологии»
Тема: «Алгоритмы и структуры данных в Python»

Студент гр. 3342

Малахов А.И.

Преподаватель

Иванов Д.В.

Санкт-Петербург

2024

Цель работы

Изучить основы анализа данных и машинного обучения, освоить основные инструменты для обработки и анализа данных.

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` 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

Описание функций:

1. Загрузка данных:

Функция `load_data()` загружает данные набора Wine из `sklearn.datasets`. Разделяет данные на обучающую и тестовую выборки с заданным `train_size`, после чего возвращает обучающие и тестовые наборы признаков и меток.

2. Тренировка модели:

Функция `train_model()` производит создание и обучение модели на основе `X_train`, `y_train` и через алгоритм `KNeighborsClassifier` (метод ближайших соседей). Этот модель будет работать следующим образом: для новой точки она будет находить количество соседних точек `n_neighbors` в данном пространстве и по ним относить ее к определенному классу. Возвращается обученная модель.

3. Предсказание:

Функция `predict()` прогнозирует классы для тестовых данных с помощью обученной модели. Возвращает вектор предсказанных меток.

4. Оценка:

Функция `estimate()` оценивает точность модели, сравнивая предсказанные и истинные метки. Возвращает рассчитанную точность.

5. Масштабирование данных:

Функция `scale()` производит предварительную обработку данных, которая преобразует признаки в заданный масштаб для обеспечения более стабильного обучения модели по одному из следующих алгоритмов.

Исследование работы классификатора, обученного на данных разного размера:

Размер обучающего набора	Точность
0.1	0.379
0.3	0.711
0.5	0.843
0.7	0.911
0.9	0.922

Как видно из таблицы, точность модели возрастает с увеличением размера обучающей выборки. Это происходит потому, что при большем количестве обучающих данных модель лучше обучается закономерностям в данных и может делать более точные прогнозы.

Исследование работы классификатора, обученного с различными значениями `n_neighbors`:

Значение <code>n_neighbors</code>	Точность
3	0.873
5	0.897
9	0.924
15	0.932
25	0.917

Из таблицы видно, что наилучшие результаты достигаются при значении `n_neighbors` равном 15. С ростом `n_neighbors` точность сначала увеличивается, а затем немного падает. Это можно объяснить тем, что при слишком большом

количестве соседей модель начинает учитывать "шум", что приводит к ошибочным результатам.

Исследование работы классификатора с предобработанными данными:

Метод предобработки	Точность
Без предобработки	0.889
StandardScaler	0.948
MinMaxScaler	0.938
MaxAbsScaler	0.938

Как видно из таблицы, точность стала выше, но особой разницы между скейлерами нет. В основном она зависит от особенности настроек конкретных данных.

Разработанный программный код см. в приложении А.

Выводы

Были изучены основы анализа данных и машинного обучения, а также освоены основные инструменты для обработки и анализа данных. Была разработана программа, которая обучает модель для предсказания классов вин, а также проведены с ней некоторые исследования.

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_size=0.8):
    wine = datasets.load_wine()
    X = wine.data[:, :2]
    y = wine.target
    X_train, X_test, y_train, y_test = train_test_split(X, y,
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'):
    clf = KNeighborsClassifier(n_neighbors=n_neighbors,
weights=weights)
    clf.fit(X_train, y_train)
    return clf

def predict(clf, X_test):
    res = clf.predict(X_test)
    return res

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

def scale(data, mode='standard'):
    valid_methods = ['standard', 'minmax', 'maxabs']
    if mode not in valid_methods:
        return None

    scaler_map = {
```

```
        'standard': StandardScaler(),
        'minmax': MinMaxScaler(),
        'maxabs': MaxAbsScaler()
    }

    scaler = scaler_map[mode]
    scaled_data = scaler.fit_transform(data)

    return scaled_data
```