МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизация производственных процессов»

Основы разработки СППР на языке Python

Методические указания
с заданиями по контрольной работе
для студентов заочной формы обучения

Ростов-на-Дону
ДГТУ
2022

УДК 681.5

Составитель: Быкадор В.С.

Методические указания. – Ростов-на-Дону : Донской гос. техн. ун-т, 2022. – 16 с.

Методические указания с заданиями по контрольной работе по дисциплине «Основы разработки СППР на языке Python» предназначены для студентов заочной формы обучения по направлению подготовки 15.04.04 «Автоматизация технологических процессов и производств» профиль «Интеллектуальные системы сбора и анализа больших данных»

УДК 681.5

Печатается по решению редакционно-издательского совета
Донского государственного технического университета

В печать ___.___.20__г.
Формат 60x84/16. Объем ____ усл. п. л.
Тираж___ экз. Заказ № ___.

Издательский центр ДГТУ
Адрес университета и полиграфического предприятия:
344000, г. Ростов-на-Дону, пл. Гагарина, 1

Содержание

ОБЩИЕ СВЕДЕНИЯ И ВЫБОР ВАРИАНТА

Контрольная работа состоит из ряда заданий, выполнение которых необходимо для освоения дисциплины. Все задания выполняются вне зависимости от номера варианта.

Задание № 1. Программная реализация и исследование метода k ближайших соседей на основе классов специальной библиотеки по анализу данных языка программирования Python

Цель — программная реализация и исследование свойств метода **k** ближайших соседей.

Описание

Практическая часть:

Требуется разработать модель машинного обучения с использованием метода **k** ближайших соседей для СППР о сорте растения «ирис» с целью дальнейшей классификации ириса по измеренным признакам, которыми являются:

- длина чашелистиков (sepal length), см;
- ширина чашелистиков (sepal width), см;
- длина лепестков (petal length), см;
- ширина лепестков (petal width), см;

Рассматривается три класса (сорта) ириса:

- щетинистый (setosa);
- разноцветный (versicolor);
- виргинский (virginica).

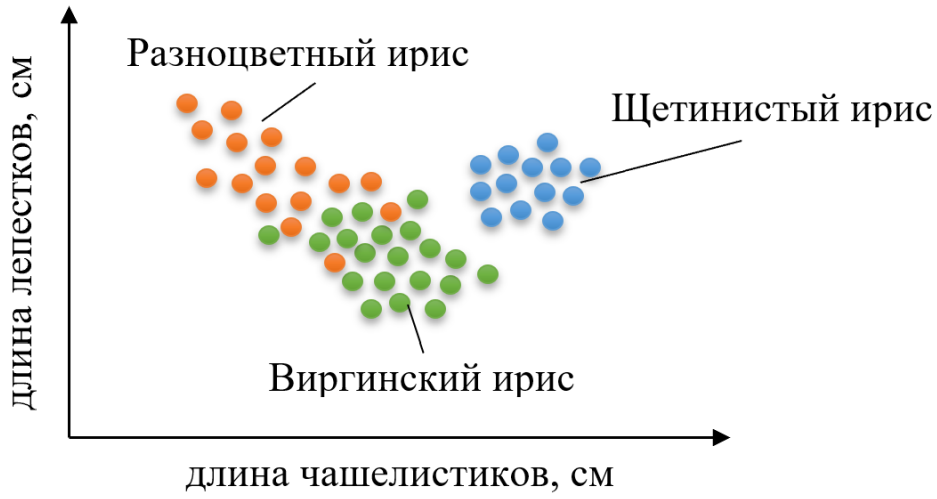
В массиве имеется 150 маркированных (заранее размеченных данных).

1) Подключите набор данных sklearn.datasets.load_iris к файлу программы через инструкцию from…import.
2) Загрузите набор данных iris_dataset = load_iris().
3) Используя функцию train_test_split() требуется разделить маркированные данные на два списка: обучающий набор (75% записей) и тестовый (контрольный) набор (25% записей).
4) Используя для модели класс, реализующий алгоритм **k** ближайших соседей – sklearn.neighbors.KNeighborsClassifier для одного соседа, выполните обучение модели на обучающих признаках и соответствующих им классах, используя метод fit() экземпляра класса KNeighborsClassifier.

5) Оцените правильность полученной модели машинного обучения для заданных параметров классификатора, используя метод score() экземпляра класса KNeighborsClassifier.
6) Проверьте работу модели машинного обучения на каких-либо произвольных значениях признаков (достаточно получить два различных результата), при этом соблюдайте формат входных данных. Для выполнения классификации наблюдаемого объекта используйте метод predict() экземпляра класса KNeighborsClassifier.
7) Доработайте программу до рабочего консольного приложения с предложением пользователю ввести значения признаков, на основании которых пользователь получит ответ о сорте ириса. Интерфейс программы на русском языке.

Исследовательская часть:

1) Постройте области меток каждого из классов по двум признакам (переберите все варианты пар признаков) и оцените на сколько хорошо области меток, для каждого вида ириса, разделены между собой.



Пример, того, как могут выглядеть области меток для сортов ириса по двум признакам

2) Постройте зависимость (график) точности модели машинного обучения (для метода k ближайших соседей) от количества соседей и оцените как их количество влияет на точность результата прогнозирования. Что можно сказать об обобщающей способности модели.

Задание № 2. Программная реализация линейной модели классификации на основе классов специальной библиотеки по анализу данных языка программирования Python

Цель — программная реализация и исследование свойств линейных моделей машинного обучения для мультиклассовой классификации.

Описание

Практическая часть:

Требуется разработать линейные модели машинного обучения для мультиклассовой классификации сортов растения «ирис»

В массиве имеется 150 маркированных (заранее размеченных данных).

1) Подключите набор данных sklearn.datasets.load_iris к файлу программы через инструкцию from…import.
2) Загрузите набор данных iris_dataset = load_iris().
3) Оставьте в наборе два любых признака для каждой записи. Это делается для возможности визуализации классов ириса и результатов работы модели на плоскости. Можно выбрать любые два признака, но можете использовать результаты исследований набора данных из предыдущей работы для выбора наилучшего сочетания признаков с точки зрения разделимости классов.
4) Используя функцию train_test_split() требуется разделить маркированные данные на два списка: обучающий набор (75% записей) и тестовый (контрольный) набор (25% записей).
5) Выполните:
    a. обучение модели используя алгоритм логистической регрессии – класс linear_model.LogisticRegression на обучающем наборе данных.
    b. оцените точность прогнозирования модели, используя тестовый набор данных (метод score()).
    c. оцените точность прогнозирования моделью построением на плоскости точек классов ириса и задаваемых для классификации данных. Выполните вывод результата классификации, полученный моделью машинного обучения в консоль, и проверьте данный результат визуально на построенной плоскости.

d. Добавьте к программе модель на основе алгоритма линейного метода опорных векторов – класс svm.LinearSVC.
e. повторите пункты b и c, но одновременно по двум линейным моделям мультиклассовой классификации.
f. сделайте выводы по полученным результатам.

Исследовательская часть:

Исследуете как параметр **C** в регрессионной и svc моделях влияет на их способность выполнять обобщения. Для представления результатов постройте соответствующие графики зависимостей и сделайте выводы.

Задание № 3. Программная реализация метода деревьев решений на основе классов специальной библиотеки по анализу данных языка программирования Python

Цель — программная реализация и исследование свойств моделей, основанных на методе «деревья решений».

Описание

Практическая часть:

Требуется разработать модель машинного обучения на основе метода «дерева решений».

1) Для лучшей демонстрации работы модели, на основе дерева решений, следует оставить два признака - длина лепестков (petal length) и ширина лепестков (petal width).
2) Используя функцию train_test_split() требуется разделить маркированные данные на два списка: обучающий набор (75% записей) и тестовый (контрольный) набор (25% записей).
3) Выполните:
    a. обучение модели используя алгоритм «дерева решений» – класс DecisionTreeClassifier на обучающем наборе данных с глубинной дерева равной 4.
    b. оцените точность прогнозирования модели, используя тестовый набор данных (метод score()).
    c. оцените точность прогнозирования модели построением на плоскости точек классов ириса и задаваемых для классификации данных. Выполните вывод результата классификации, полученный моделью машинного обучения в консоль, и проверьте данный результат визуально на построенной плоскости.

Исследовательская часть:

1) Исследуете как влияет глубина дерева на обобщающую способность модели машинного обучения, применяемую как обучающим, так и тестовым данным. Сделайте выводы.
2) Выполните визуализацию дерева решений и изучите его.

Задание № 4. Программная реализация методов снижения размерности, выделения признаков и множественного обучения на основе классов специальной библиотеки по анализу данных языка программирования Python

Цель — программная реализация и исследование возможностей снижения размерности признаков набора данных и множественного обучения.

Описание

Практическая часть:

Требуется выполнить снижение размерности признаков набора данных, а также выполнить множественного обучение модели.

1) Снижение размерности признаков набора данных:
    a. загрузите набор данных по видам ириса;
    b. выполните масштабирование данных исходного набора, используя класс StandardScaler(); и его метод fit();
    c. выполните снижение размерности признаков набора данных с 4-х признаков до 2-х признаков, применив метод transform() класса StandardScalter().
    d. используя метод fit() класса PCA() оставьте в наборе данных две главные компоненты;
    e. постройте график распределения классов ириса в плоскости двух компонент.

2) Множественное обучение с помощью алгоритма t-SNE:
    a. загрузите набора данных load_digits;
    b. выполните вывод считанных данных, используя matplotlib.pyplot.imshow(), предварительно разделив область вывода графиков на 10 подграфиков, используя matplotlib.pyplot.subplot();
    c. применив метод fit_transform() класс from sklearn.manifold import TSNE выделите t-SNE признаки из набора данных;
    d. постройте график распределения классов «цифр» в плоскости двух t-SNE признаков.

Исследовательская часть:

Постройте тепловую карту (функция matplotlib.pyplot.matshow()) двух главных компонент набора данных:

a. Что можно сказать о главных компонентах?
b. Как они зависят от признаков?

2) Выполните снижение размерности признаков набора данных load_digits:
   a. используйте класс PCA(), также как в первом задании данной практической работы;
   b. постройте график распределения классов «цифр» в плоскости двух компонент;
   c. сравните с результатами из второго практического задания и сделайте выводы.

Задание № 5. Программная реализация методов кластеризации на основе классов специальной библиотеки по анализу данных языка программирования Python

Цель — программная реализация и исследование возможностей кластеризации по методу ***k***-средних.

Описание

Практическая часть:

1) Требуется выполнить разделение на кластеры исходных данных используя метод ***k***-средних:
   a. загрузите набора данных from sklearn.datasets import make_blobs;
   b. используя метод fit() класса from sklearn.cluster import KMeans, постройте три кластера из исходных данных;
   c. выполните визуализацию данных вместе с центрами кластеров.

2) Требуется выполнить разделение на кластеры исходных данных используя алгоритм DBSCAN:
   a. загрузите набора данных from sklearn.datasets import make_moons;
   b. выполните масштабирование данных, применив методы fit() и transform() класса StandardScalter();
   c. используя метод fit_predict() класса from sklearn.cluster import DBSCAN, постройте два кластера из исходных данных;
   d. выполните визуализацию данных.

Исследовательская часть:

1) Исследуйте как будет работать метод k-средних на наборе данных from sklearn.datasets import make_moons, при разбиении данного набора на два кластера.
2) Сравните полученный результат с данными из второй части практической работы.
3) Какие Вы можете сделать выводы по работе метода ***k***-средних.

Задание № 6. Программная реализация модели текстовых данных в виде "мешка слов" на основе классов специальной библиотеки по анализу данных языка программирования Python

Цель — программная реализация и исследование модели текстовых данных, представленных в виде «мешка слов».

Описание

Практическая часть:

1) Требуется создать «мешок слов» из размеченного набора сообщений для дальнейшего обучения модели:
   a. используя метод fit() экземпляра класса from sklearn.feature_extraction.text import CountVectorizer выполните токенизацию и построения словаря из исходных документов;
   b. используя метод transform() экземпляра класса CountVectorizer получите «мешок слов».

2) Используя любой из классов машинного обучения, которые были рассмотрены в первых, двух работах выполните процедуры:
   a. обучения модели;
   b. анализа её общности;
   c. проверьте работу обученной модели на паре примеров.

Исследовательская часть:

Исследуйте как повысится точность работы модели машинного обучения если выполнить:

1) учет только тех токенов, которые встречаются в нескольких документах (двух…трёх) – параметр min_df в конструкторе класса CountVectorizer;
2) исключите из анализируемого текста так называемые «стоп-слова»;
3) сделайте выводы по повышению общности модели машинного обучения.

Задание № 7. Программная реализация масштабирования данных методом tf-idf на основе классов специальной библиотеки по анализу данных языка программирования Python

Цель — программная реализация и исследование метода tf-idf анализа текстовых данных.

Описание

Практическая часть:

1) Требуется выполнить анализ текстовых данных используя метод tf-idf:
   a. используя метод fit_transform() экземпляра класса from sklearn.feature_extraction.text import TfidfVectorizer выполните токенизацию и построения словаря из исходных документов;

2) Используя любой из классов машинного обучения, которые были рассмотрены в первых, двух работах выполните процедуры:
   a. обучения модели;
   b. анализа её общности;
   c. проверьте работу обученной модели на паре примеров.

Исследовательская часть:

Исследуйте как отличаются результаты, полученные классом CountVectorizer и классом TfidfVectorizer выполните. Сделайте вывод.

Задание № 8. Программная реализация модели текстовых данных в виде "мешка слов" для n-грамм на основе классов специальной библиотеки по анализу данных языка программирования Python

Цель — программная реализация и исследование модели текстовых данных, представленных в виде «мешка слов» для n-грамм.

Описание

Практическая часть:

3) Требуется создать «мешок слов» биграмм и триграмм из размеченного набора сообщений для дальнейшего обучения модели:
   a. используя метод fit() экземпляра класса from sklearn.feature_extraction.text import CountVectorizer выполните токенизацию и построения словаря из исходных документов;
   b. используя метод tran get_feature_names()экземпляра класса CountVectorizer и получите «мешок слов» для биграмм и триграмм.

4) Используя любой из классов машинного обучения, которые были рассмотрены в первых, двух работах выполните процедуры:
   a. обучения модели каждой n-грамм;
   b. анализа её общности каждой n-грамм;
   c. проверьте работу обученной модели на паре примеров для каждой n-грамм.

Исследовательская часть:

Исследуйте как меняется точность работы модели машинного обучения для обычного «мешка слов», биграмм и триграмм.

Учебно-методическое и информационное обеспечение дисциплины

1. Сузи, Р.А. Язык программирования Python: учеб. пособие, М.: Интернет-Ун-т Информ. Технологий: Бином. Лаборатория знаний, 2006.
2. Буйначев, С.К., Боклаг, Н.Ю. Основы программирования на языке Python: учебное пособие, Екатеринбург: Издательство Уральского университета, 2014.
3. Сузи, Р.А. Язык программирования Python: учебное пособие, М.: Интернет- Университет Информационных Технологий (ИНТУИТ), 2016
4. Уэс, Маккинли Python и анализ данных: практическое пособие, Саратов: Профобразование, 2017.
5. Балджы, А.С., Хрипунова, М.Б. Математика на Python: учебно-методическое пособие, М.: Прометей, 2018.
6. Гуриков С. Р. Основы алгоритмизации и программирования на Python: учебное пособие, М.: Издательство "ФОРУМ", 2017.
7. Жуков Р. А. Язык программирования Python: практикум: Учебное пособие, М.: ООО "Научно- издательский центр ИНФРА-М", 2020.