**Липецкий государственный технический университет**

Факультет Автоматизации и Информатики

Кафедра Автоматизированных систем управления

Лабораторная работа №4

по прикладным интеллектуальным системам и экспертным системам

Предварительная обработка текстовых данных

Студент Ледовских Д.В.

Группа М-ИАП-22

Проверил

Доцент Кургасов В.В.

Липецк 2022г.

Цель работы

Получить практические навыки решения задачи кластеризации фактографических данных в среде Jupyter Notebook. Научиться проводить настраивать параметры методов и оценивать точность полученного разбиения.

Задание кафедры

1) Загрузить выборки согласно варианту задания

2) Отобразить данные на графике в пространстве признаков. Поскольку

решается задача кластеризации, то подразумевается, что априорная информация о принадлежности каждого объекта истинному классу неизвестна, соответственно, на данном этапе все объекты на графике должны отображаться одним цветом, без привязки к классу.

3) Провести иерархическую кластеризацию выборки, используя разные

способы вычисления расстояния между кластерами: расстояние ближайшего соседа (single), дальнего соседа (complete), Уорда (Ward). Построить дендрограммы для каждого способа. Размер графика должен быть подобран таким образом, чтобы дендрограмма хорошо читалась.

4) Исходя из дендрограмм выбрать лучший способ вычисления расстояния между кластерами.

5) Для выбранного способа, исходя из дендрограммы, определить количество кластеров в имеющейся выборке. Отобразить разбиение на кластеры и центроиды на графике в пространстве признаков (объекты одного кластера должны отображаться одним и тем же цветом, центроиды всех кластеров – также одним цветом, отличным от цвета кластеров)

6) Рассчитать среднюю сумму квадратов расстояний до центроида, среднюю сумму средних внутрикластерных расстояний и среднюю сумму межкластерных расстояний для данного разбиения. Сделать вывод о качестве разбиения.

7) Провести кластеризацию выборки методом k-средних. для k [1, 10].

8) Сформировать три графика: зависимость средней суммы квадратов расстояний до центроида, средней суммы средних внутрикластерных расстояний и средней суммы межкластерных расстояний от количества кластеров. Исходя из результатов, выбрать оптимальное количество кластеров.

9) Составить сравнительную таблицу результатов разбиения иерархическим методом и методом k-средних.

Вариант 6

|  |  |
| --- | --- |
| Вариант | 6 |
| Вид классов | blobs |
| Random\_state | 68 |
| cluster\_std | 2 |
| Centers | 6 |

Ход работы

Для данной лабораторной работы нам необходимы библиотеки: pandas, numpy, matplotlib, NLTK, itertools, scikit-learn.

Pandas — программная библиотека на языке Python для обработки и анализа данных;

NumPy нужен для работы с большими многомерными массивами и матрицами, в данной библиотеке реализованы различные математические функции и операции с массивами данных;

Matplotlib разработан для визуализации данных в понятном пользователю виде за счет обширного функционала настроек итогового изображение, представленного двумерной или трехмерной графикой.

Scikit-learn – это один из наиболее широко используемых пакетов Python для Data Science и Machine Learning. Он содержит функции и алгоритмы для машинного обучения: классификации, прогнозирования или разбивки данных на группы;

NLTK — пакет библиотек и программ для символьной и статистической обработки естественного языка, написанных на языке программирования Python. Содержит графические представления и примеры данных;

Itertools стандартизирует основной набор быстрых эффективных по памяти инструментов, которые полезны сами по себе или в связке с другими инструментами.

Для выполнения лабораторной работы в среде Jupiter Notebook были импортированы следующие модули:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.datasets import make\_blobs

from sklearn.metrics import classification\_report

from scipy.cluster.hierarchy import linkage, dendrogram, fcluster

from sklearn.cluster import KMeans

from sklearn.metrics.pairwise import euclidean\_distances

from sklearn.cluster import KMeans

from matplotlib.pyplot import xticks

Программный код для загрузки и отображения данных:

X, y = make\_blobs(centers=centers\_count, random\_state=68, cluster\_std=2, n\_samples=100)

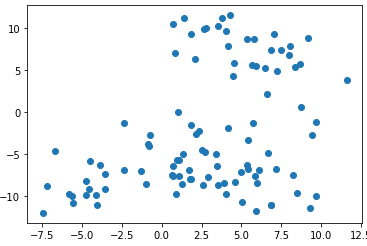


Рисунок 1 – сгенерированная выборка на графике

Выполним кластеризацию загруженной выборки

Реализация расстояния ближайшего соседа выполнена следующим образом:

def cluster\_center(X, c):

result = np.zeros((centers\_count,2))

for i in range(1,centers\_count):

ix = np.where(c == i)

result[i-1,:] = np.mean(X[ix,:], axis=1)

return result

mergings = linkage(X, method='single')

T = fcluster(mergings, centers\_count, criterion='maxclust')

c = cluster\_center(X, T)

dendrogram(mergings)

plt.show()

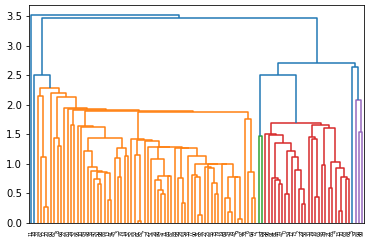


Рисунок 2 – результат кластеризации

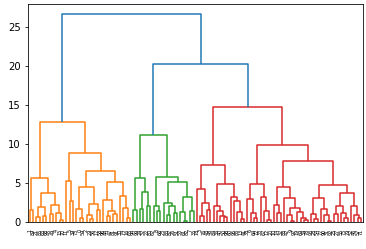


Рисунок 3 – Расстояние дальнего соседа

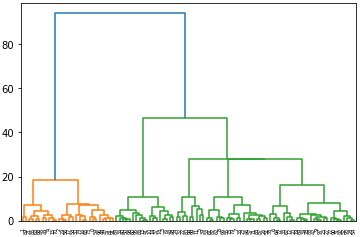


Рисунок 4 – Метод Уорда

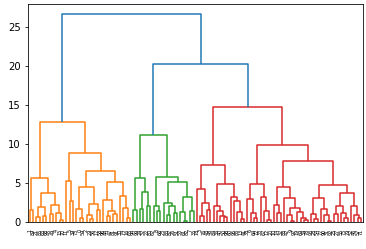


Рисунок 5 – Лучший способ разбиений

Реализация визуализации разбиения на кластеры и центроиды:

plt.scatter(X[:,0], X[:,1], c=clusterf)

plt.scatter(c[:,0], c[:,1], c='red')

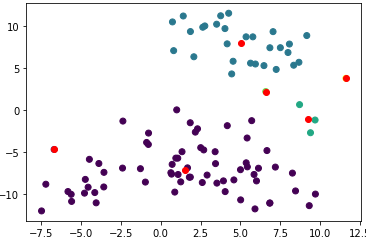


Рисунок 6 – Визуализация разбиения

Расчитаем среднее расстояние до центроидов, среднюю сумму внутрикластерных расстояний и среднюю сумму междукластерных расстояний

Реализация функций для расчета выполнена следующим образом:

def sum\_sq(X, clust, cent):

sums = np.zeros(centers\_count)

for i in range(1,centers\_count+1):

ix = np.where(clust == i)

sums[i-1] = np.sum(euclidean\_distances(\*X[ix,:], [cent[i-1]])\*\*2)

return np.sum(sums) / centers\_count

def sum\_av(X, clust, cent):

sums = np.zeros(centers\_count)

for i in range(1,centers\_count+1):

ix = np.where(clust == i)

sums[i-1] = np.sum(euclidean\_distances(\*X[ix,:], [cent[i-1]])\*\*2)/len(\*X[ix,:])

return np.sum(sums) / centers\_count

def sum\_cl(cent):

sums = np.sum(euclidean\_distances(cent, cent))

return sums / centers\_count

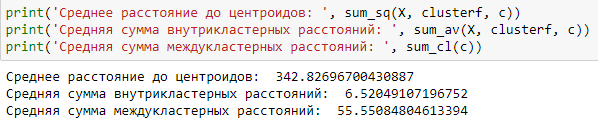


Рисунок 7–Результат выполнения реализованных функций

Проведем кластеризацию методом к-средних

Программный код кластеризации:

clussters = []

dist = []

cent = []

for i in range(1,15):

KM = KMeans(n\_clusters=i)

KM.fit(X)

clussters.append(KM.predict(X))

cent.append(KM.cluster\_centers\_)

dist.append(KM.inertia\_ / i)

xticks(np.arange(1, 15, step=1))

plt.plot(range(1, 15), dist, '-bo')

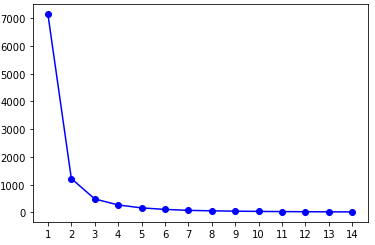


Рисунок 8 – Результат кластеризации методом k-средних

Визуализируем графики зависимость средней суммы квадратов расстояний до центроида, средней суммы средних внутрикластерных расстояний и средней суммы межкластерных расстояний от количества кластеров.

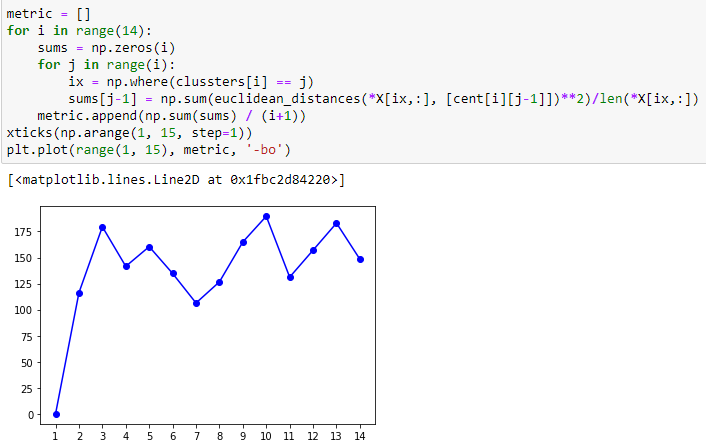


Рисунок 9 – Результат визуализации зависимости средней суммы квадратов расстояний до центроида

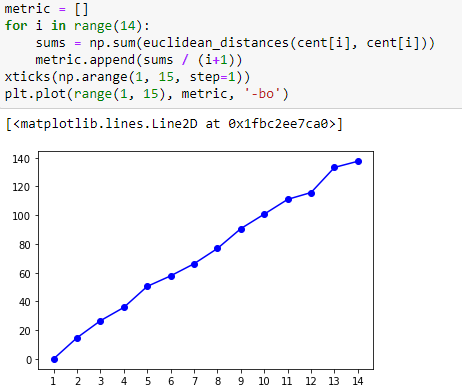


Рисунок 10 – Результат визуализации зависимости средней суммы средних внутрикластерных расстояний

Наилучшие результаты разбиения продемонстрировал, для которого расстояние между кластерами вычислялось методом ближайшего соседа. Количество кластеров при этом подходе составило 6. При этом метод локтя нашёл оптимальное количество кластеров в размере трёх.

Вывод

В ходе выполнения лабораторной работы получены практические навыки решения задачи кластеризации фактографических данных в среде Jupyter Notebook и навыки настройки параметров методов и оценивания точность полученного разбиения.