ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ  
«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра АСУ

Отчет

о лабораторной работе №2

по дисциплине «Интеллектуальный анализ данных»

на тему: «Применение библиотеки Scikit-Learn для решения задач кластеризации»

Выполнил:

студент группы ИСТ-19а

Деркач К. И.

Проверили:

Васяева Т. А.

Шуватова Е. А.

Донецк – 2022

**Цель работы:** изучение алгоритмов кластеризации, изучение возможностей библиотеки Scikit-Learn для решения задач кластеризации.

Порядок выполнения работы

1. Выбрать одну из баз данных (согласовать с преподавателем), список которых представлен на сайте http http://archive.ics.uci.edu/ml/datasets.php?format=&task=clu&att=&area=&numAtt=&numIns=&type=&sort=nameUp&view=table. Скачать данные с сайта или можно использовать данные, полученные на практике, результаты НИРС или данные для бакалаврской работы.
2. Исходные данные должны храниться в файле. Программным способом считать эти данные и сформировать обучающую и тестовую выборки.
3. Разработать программу на любом языке программирования с использованием любых существующих библиотек. Рекомендован язык Python с использованием библиотек NumPy, Pandas, Scikit-Learn. Программа должна обеспечивать: реализацию метода кластеризации, который указан в задании; визуализацию результатов; тестирование модели; оценка точности модели; сохранение / считывание модели; вывод модели.
4. Согласно варианту заданий (таблица 2.8) построить несколько моделей. В процессе построения моделей, анализируйте ее качество и записывайте результаты в таблицу, указывая значения гиперпараметров.

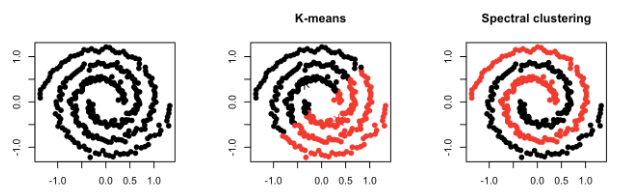
**Индивидуальное задание:** Вариант 4, Spectral clustering

Спектральная кластеризация

Спектральная кластеризация

При спектральной кластеризации точки данных рассматриваются как узлы графа. Затем узлы сопоставляются с низкоразмерным пространством, которое можно легко разделить для формирования кластеров. Спектральная кластеризация использует информацию из собственных значений (спектра) специальных матриц (т.е. матрицы сродства, матрицы степени и матрицы Лапласа), полученных из графика или набора данных.

В таком методе точки, которые соединены или находятся непосредственно рядом друг с другом, помещаются в один и тот же кластер.



Спектральная кластеризация не зависит от количества параметров объектов. Но проблемы могут возникнуть при применении к очень большим наборам данных.

Метод может быть реализован при помощи класса sklearn.cluster.SpectralClustering библиотеки sklearn. Среди наиболее важных параметров можно выделить:

* affinity (метод построения матрицы сходства)

kmeans/discretize

* n\_neighbors (количество соседей для использования при построении матрицы сходства)
* assign\_labels (стратегия назначения меток)

nearest\_neighbors – построить матрицу сходства, вычислив граф ближайших соседей.

rbf – построить матрицу сходства с использованием ядра радиальной базисной функции (RBF).

**Описание набора данных**

Измерение геометрических свойств зерен трех разных сортов пшеницы. Метод мягкого рентгеновского излучения и пакет GRAINS использовались для построения всех семи атрибутов с действительными значениями

Для построения данных были измерены семь геометрических параметров зерен пшеницы:

1. площадь A,

2. периметр P,

3. компактность C = 4\*pi\*A/P^2,

4. длина зерна,

5. ширина зерна. ,

6. коэффициент асимметрии

7. длина борозды ядра.

**Листинг программы**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA

from sklearn.cluster import SpectralClustering

from sklearn.metrics import silhouette\_score

import pickle

# функции сохранения и загрузки модели

def save\_model(path, model):

with open('%s.pickle' % path, 'wb') as f:

pickle.dump(model, f)

def load\_model(path):

with open('%s.pickle' % path, 'rb') as f:

return pickle.load(f)

data = np.loadtxt('d:/seeds\_dataset.txt')

df = pd.DataFrame(data, columns = ['Area', 'Perimeter',

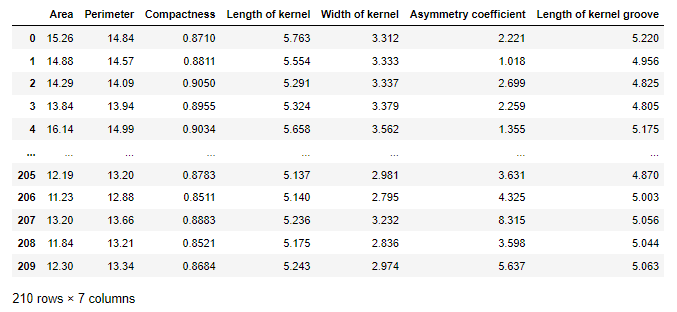
'Compactness','Length of kernel', 'Width of kernel','Asymmetry coefficient','Length of kernel groove', 'Class'])

class\_names = df['Class'].astype(int)

# удаление столбца меток класса из фрейма

df = df.drop('Class', axis=1)

df



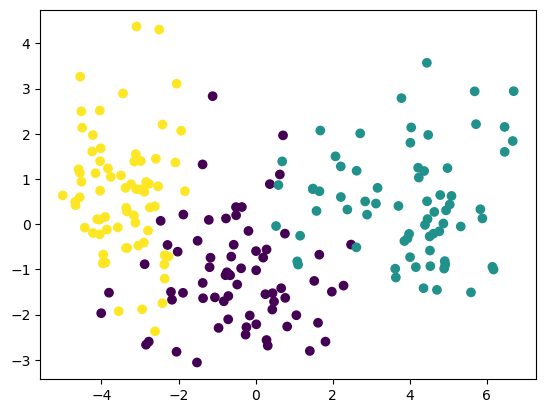
# уменьшение размерности методом главных компонент

pca = PCA(n\_components=2).fit(df)

df2d = pca.transform(df)

plt.scatter(df2d[:,0], df2d[:,1], c=class\_names)

plt.show()



from itertools import product

alist = ['rbf', 'nearest\_neighbors']

nblist = [5,10,20]

allist = ['kmeans', 'discretize']

# перебор гиперпараметров

for a, nb, al in product(alist, nblist, allist):

model = SpectralClustering(n\_clusters=3, affinity=a, n\_neighbors=nb,assign\_labels=al)

labels = model.fit\_predict(df2d)

plt.scatter(df2d[:, 0], df2d[:, 1], c=labels)

plt.show()

'''

Метрика silhouette score (средний коэффициент силуэта всех образцов) относится к методу интерпретации и

проверки согласованности в кластерах данных.

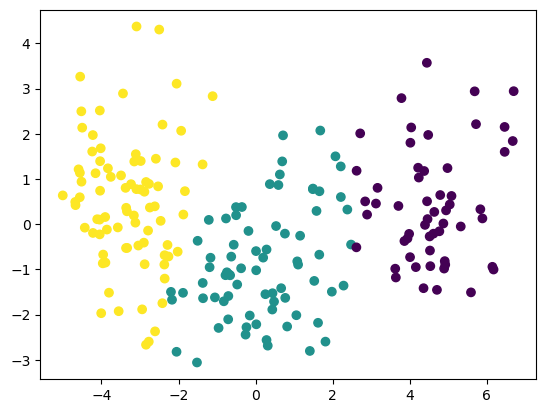
Этот метод обеспечивает краткое представление того, насколько хорошо классифицирован каждый объект.

Коэффициент силуэта рассчитывается с использованием среднего внутрикластерного расстояния

и среднего расстояния до ближайшего кластера для каждого образца.

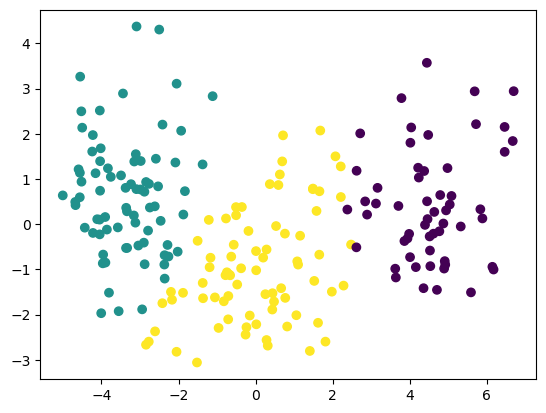
'''

print('affinity =' +a+'; n\_neighbors = '+str(nb)+'; assign\_labels = '+al+'\nsilhouette score = '+ str(silhouette\_score(df2d, labels))+'\n\n\n')



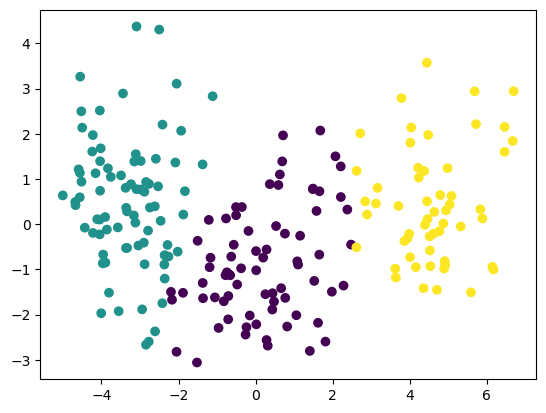
affinity =rbf; n\_neighbors = 5; assign\_labels = kmeans

silhouette score = 0.47194806040596987



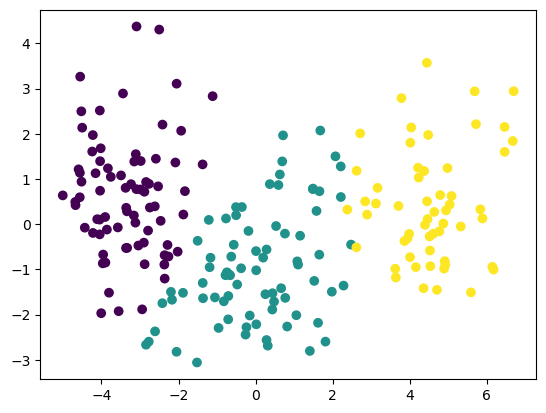
affinity =rbf; n\_neighbors = 5; assign\_labels = discretize

silhouette score = 0.47301351167289907



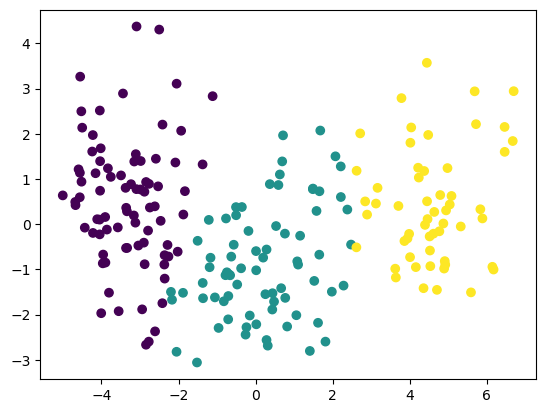
affinity =rbf; n\_neighbors = 10; assign\_labels = kmeans

silhouette score = 0.47194806040596987



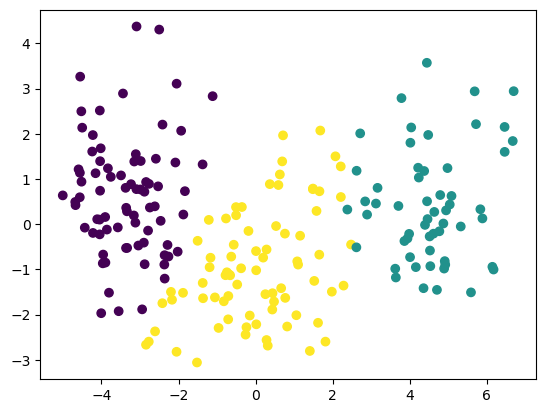
affinity =rbf; n\_neighbors = 10; assign\_labels = discretize

silhouette score = 0.47301351167289907



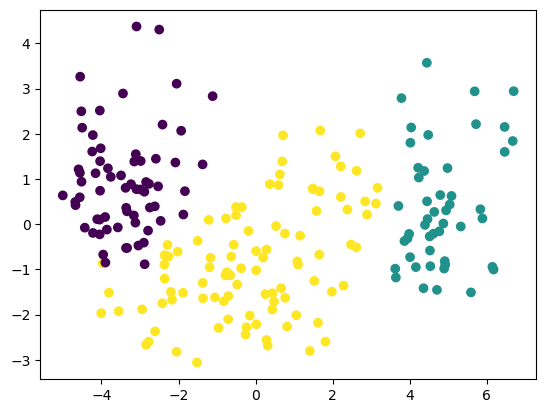
affinity =rbf; n\_neighbors = 20; assign\_labels = kmeans

silhouette score = 0.47194806040596987



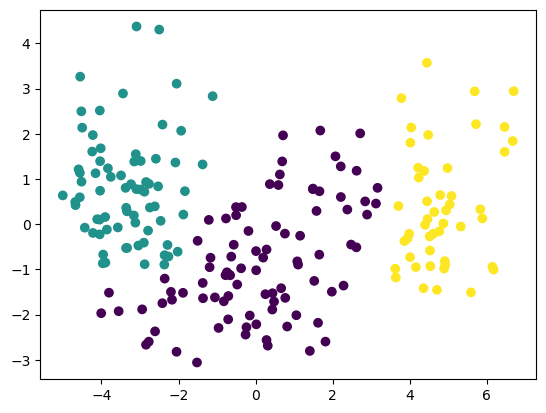
affinity =rbf; n\_neighbors = 20; assign\_labels = discretize

silhouette score = 0.47301351167289907



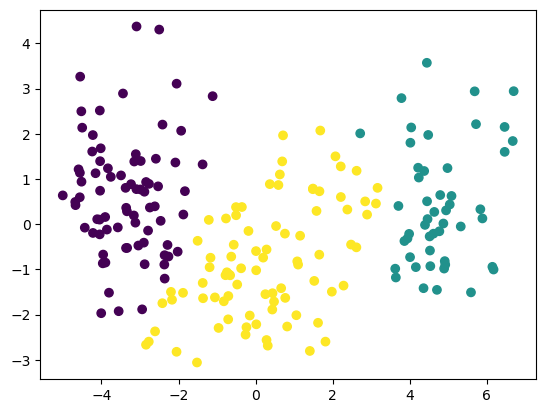
affinity =nearest\_neighbors; n\_neighbors = 5; assign\_labels = kmeans

silhouette score = 0.42736102135432275



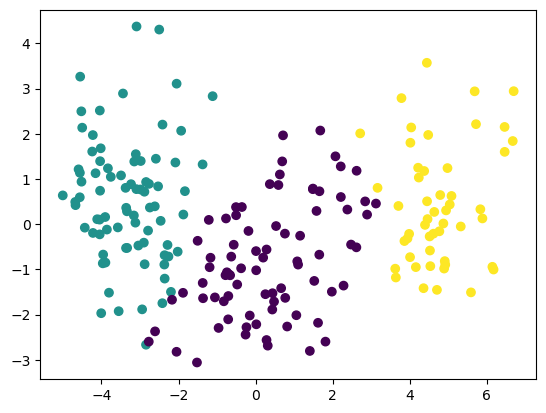
affinity =nearest\_neighbors; n\_neighbors = 5; assign\_labels = discretize

silhouette score = 0.439096563506894



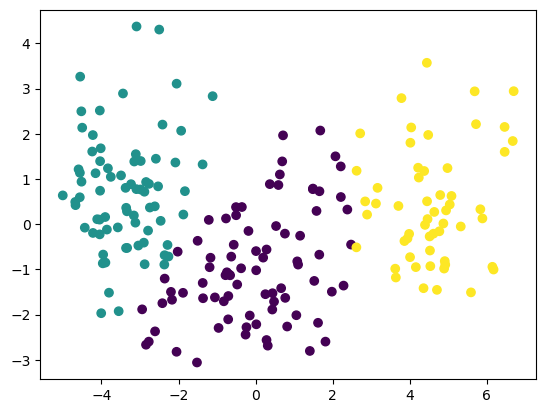
affinity =nearest\_neighbors; n\_neighbors = 10; assign\_labels = kmeans

silhouette score = 0.4562731663898959



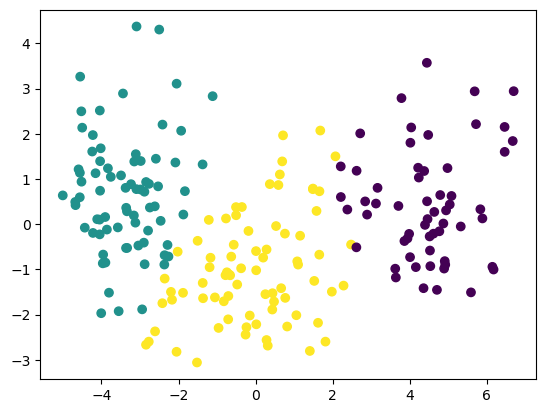
affinity =nearest\_neighbors; n\_neighbors = 10; assign\_labels = discretize

silhouette score = 0.46145344109934083



affinity =nearest\_neighbors; n\_neighbors = 20; assign\_labels = kmeans

silhouette score = 0.46939683633550566



affinity =nearest\_neighbors; n\_neighbors = 20; assign\_labels = discretize

silhouette score = 0.474252625259939