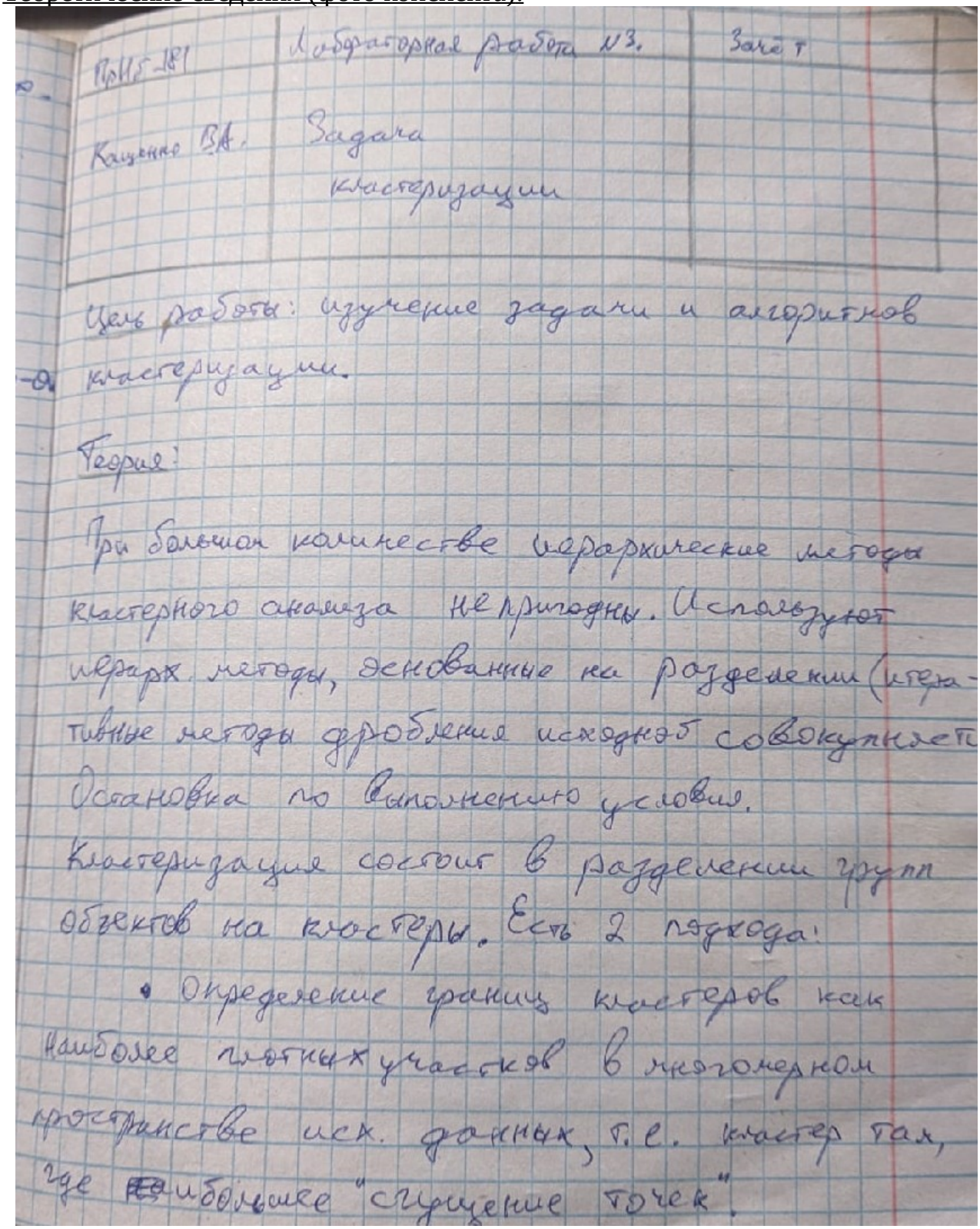
ПрИб-181	Лабораторная работа №3	Зачёт
Кащенко В. А.	Задача кластеризации.	

Цель работы: Изучить задачу и алгоритмы кластеизации.

Теоретические сведения (фото конспекта):



• Минимизация меры различия объектов.

1. Иерархич. алгоритмы.

Популярны алгоритмы, основанные на поиске оптимального в опр. смысле разбиения ~~точек~~ множества данных на ~~группы (точки)~~ кластеры. Из-за множества достоинств используются алгоритмы разбиения. Они пытаются кластеризовать данные т. образом, чтобы целевая ф-я достигла экстремума (минимума)

Алгоритмы кластеризации, основанные на методе разбиения (будут рассмотрены).

Базовые понятия:

- обучающее множество M (входное), кот. разбивается;
- метрика расстояния:

$$d_A^2(m_j, c^{(i)}) = \|m_j - c^{(i)}\|^2 = (m_j - c^{(i)})^t A (m_j - c^{(i)}),$$

где матрица A - способ вычисления расстояния

Например для единичной матрицы используется расстояние по Евклиду:

- вектор центров кластеров C;
- матрица разбиения по кластерам U;
- целевая ф-я: $J = J(M, d, C, U)$;
- набор ограничений.

Метрика расстояния - одно из важнейших понятие кластеризации. С помощью расстояния между входными векторами определяются сходства и различия. Есть много способов вычислить расстояние: Евклидово, Хэмминга, Махаланобиса и др. Выбор метода зависит от природы исследуемых объектов и имеет нагаг.

Целевая ф-я, кот. даёт решение задачи кластеризации. Алгоритм кластеризации - последовательность действий поиска минимума.

Матрица разбиения – р-ат иерархии. кластеризации

Это таблица, где # строки содержит основание матрицы и получается итогов. разбиение. Центр кластера – вектор, степень принадлежности которого заданному классу максимальна. Как правило, центров нет в исходном мн-ве.

Набор ограничений связан с условиями, налагаемые на значения элементов матрицы принадлежности. Они определяются алгоритмом кластеризации.

2. Алгоритм K-средних (K-means)

Он распространён и также носит имя быстрого кластерного анализа. Описание: работа Хартигана и Вонга (Hartigan and Wong, 1978). В отл. от иерарх. методов, кот. не требуют предв. предположений отн. числа кластеров, для возм-ти

использование необх-мо иметь гипотезу о наиб. вероятном кол-ве кластеров.

Алгоритм k-средних строит K-кластеров, расположенных на возможно больших расстояниях друг от друга. Осн. тип задач — наличие предположений (гипотез) отн. числа кластеров, при кот. они должны быть различны настолько, насколько это возможно.

Выбор числа K может базироваться на р-ах предмет. исследований, теорет соображ-ях и интуиции. Общ. идея — k кластеров (число кластеров) составляется так, что средние в кластере макс-но возможно отличаются друг от друга.

K-means хорошо работает, если данные можно разделить на компактные, примерно сферич. группы.

Данный алгоритм — прообраз всех алгоритмов нечеткой кластеризации, и его формализация позволяет путем понимания принципов, заложенных в более сложные алгоритмы.

Базов.-е понятия:

– обуч. множество $M = \{m_j\}_{j=1}^{d}$ (d – кол-во точек/векторов данных);

– метрика расстояния:

$$d_A^2(m_j, c^{(i)}) = \|m_j - c^{(i)}\|^2 = (m_j - c^{(i)})^t A (m_j - c^{(i)});$$

– вектор центров кластеров $C = \{c^{(i)}\}_{i=1}^{c}$, где

$$c^{(i)} = \frac{\sum_{j=1}^{d} u_{ij} m_j}{\sum_{j=1}^{d} u_{ij}}, \quad 1 \le i \le c$$

– матрица разбиения $U = \{u_{ij}\}$, где

$$u_{ij}^{(l)} = \begin{cases} 1, \text{ при } d(m_j, c_k^{(l)}) = \min\limits_{1 \le k \le c} d(m_j, c_k^{(l)}) \\ 0, \text{ в ост. случаях.} \end{cases}$$

– целевая ф-я:

$$J(M, U, C) = \sum_{(i)}^{c} \sum_{j=1}^{d} u_{ij} d_A^2(m_j, c^{(i)})$$

– набор ограничений:

$u_{ij} \in \{0, 1\}$

$$\sum_{i=1}^{c} u_{ij} = 1; \quad 0 < \sum_{j=1}^{d} u_{ij} < d, \text{ кол. опред-ей}$$

что # вектор данных может принадлежать одному кластеру и не принадлежать остальным

В # кластере не менее 1 точки, но не более общ. кол-ва точек.

Конструктивно алгоритм — итерационная процедура:

1. Инициализация начального разбиения (random for example), выбрать точность δ (используется в условии завершения алгоритма), инициализация номера итерации $t = 0$.

2. Центры кластеров:

$$c_i^{(t)} = \frac{\sum_{j=1}^{d} u_{ij}^{(t-1)} \cdot m_j}{\sum_{j=1}^{d} u_{ij}^{(t-1)}}, \quad 1 \le i \le C$$

3. Обновить матрицу разбиения с целью минимизации квадратов ошибок:

$$u_{ij}^{(t)} = \begin{cases} 1, \text{ при } d(m_i, c_k^{(t)}) = \min\limits_{1 \le k \le C} d(m_i, c_k^{(t)}) \\ 0, \text{ в ост. случаях} \end{cases}$$

4. Проверить условие $\|u^{(t)} - u^{(t-1)}\| < \delta$

Осн. недостаток — большой размер пространства разбиения

Один из способов устранения недостатка - представление элементов матрицы разбиения числами единичного интервала, т.е. принадлежность должна определяться ф-ей принадлежности

Данный подход нашел воплощение в алгоритме нечёткой кластеризации Fuzzy C-Means

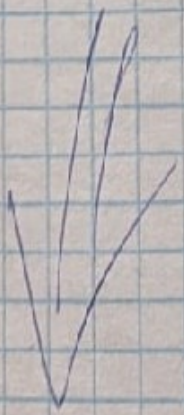
3. Кластеризация с библиотекой Python sklearn.cluster

Алгоритм бывает 2-х вариантов

- класс с методом fit для изучения на обучающем множестве и ф-я, кот. учитывая данные обуч. множества возвращает массив целочисл. меток, соотв кластерам. Метки для класса можно найти в атрибуте labels_

Входные данные:

- алгоритмы в качестве вх. данных могут принимать разл. виды матриц.
- # метод принимает стандарт. матрицы данных размера [n_samples, n_features]
Их можно получить с помощью sklearn_feature_extraction.
- Для Affinity Propagation, SpectralClustering и DBSCAN можно ввести матрицы подобия [n_samples, n_samples]
Их можно получить с помощью sklearn.metrics.pairwise.



методы пакета sklearn

Название метода	Параметры	Масштабируемость	Вариант использования	Геометрия (система)
K-means	Кол-во кластеров	Большие n_samples, средние n_clusters с опцией Mini Batch	Универсальный не слишком много кластеров плоск. геометрия	Расстояние между точками
Affinity propagation	демпфирование предпочтит. образцы	Не масштаб	Много кластеров неравномерный размер кластеров	Расстояние графа (между ближ. соседа)
Ward hierarchical clustering	Кол-во кластеров или порог. расстояние	Большое знач-е n_samples и n_clusters	много кластеров огр. связанности	Расстояние между точками
Agglomerative clustering	Кол-во кластеров, вычисл. расст-е тип связи: расст-е	Большое знач-е n_samples и n_clusters	Много кластеров огр. связанности неравном. размер	Любое парное расстояние
DBSCAN	Размер области	Больш. знач. n_samples сред. знач. n_clusters	Неплоск. геом. Неравномерн. размер кластер	Расст-е между ближ. точками
OPTICS	Мин. кол-во в кластере	ср. больш. знач. n_samples больше n_clusters	Неплоск. геом. Неравном. разм. кластеров, перем. плотность кл.	Расст-е между точками
Gaussian mixtures	Много парам.	Не масштабир-ся	Плоск. геом. Удобно оценка плотн.	Расст-е Махаланобиса от центров
Birch	Фактор ветвления	Больш. знач-е n_clusters и n_samples	Большой набор удаление выбросов сокращ-е данных	Евклидово расстояние между точками

Кластеризация с нелин. границ. возникает ситуация, когда кластеры имеют вытянутую форму, имеют нелинейное многообразие и стандартное евклидово расстояние не является правильной метрикой

Ход работы:

1 Подобран CSV-файл - Banknote-authentication.csv

2 Выполнен алгоритм K-means (Scikit-learn)

3. Для сравнения выполнен алгоритм Mean-Shift (Сдвиг среднего значения)

4. Алгоритм среднего сдвига кластеризации не делает никаких предположений, в отличие от K-means,

Мы видим результат работы параметрического (K-means) и непараметрического алгоритма (Mean-Shift)

Вывод: Изучена задача и алгоритмы кластеризации.

Постановка задачи:

Выполнить кластеризацию методом K-means на примере датасета о банкнотах (banknote-authentication.csv).

Это датасет данных об изображениях подлинных и поддельных образцов банкнот. Для оцифровки использовалась промышленная камера, которая обычно используется для проверки печати. Окончательные изображения имеют размер 400x 400 пикселей. За счет линзы объекта и расстояния до исследуемого объекта были получены полутоновые изображения с разрешением около 660 dpi. Применялось пакетное вейвлет-преобразование (WPT) для проверки подлинности защищенных документов, в частности банкнот.

Практическая часть (код программы):

```
import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib import rcParams
import pandas as pd
from sklearn.cluster import KMeans, MeanShift, estimate_bandwidth

# Reading csv file
df = pd.read_csv('banknote-authentication.csv')

# Deleting class column
df = df.drop('class', axis=1)
processed_df = df.to_numpy()[:, :]

# K-means clustering
cluster_num = 2
k_means = KMeans(init='k-means++', n_clusters=cluster_num, n_init=12)
k_means.fit(processed_df)
k_means_labels = k_means.labels_

# Mean-shift clustering
bandwidth = estimate_bandwidth(processed_df, quantile=0.183, n_samples=1000)
m_shift = MeanShift(bandwidth=bandwidth)
m_shift.fit(processed_df)
m_shift_labels = m_shift.labels_
df['ms_cluster'] = m_shift.labels_

# Setting up mpl params
mpl.rcParams['font.family'] = 'Times New Roman'
mpl.rcParams['axes.unicode_minus'] = False
rcParams.update({'figure.autolayout': True})

fig = plt.figure(figsize=(12, 8))

colors = list(map(lambda x: '#002bff' if x == 1 else '#f7ab05', k_means_labels))

ax1 = fig.add_subplot(1, 2, 1, projection = '3d')
```

```
#ax1.view_init(azim=0, elev=90)
ax1.scatter(processed_df[:, 0], processed_df[:, 1], processed_df[:, 2], c=colors, alpha=0.8)
ax1.set_zlabel('кратность изображения')
ax1.set_xlabel('дисперсия изображения')
ax1.set_ylabel('асимметрия изображения')
ax1.set_title('K-Means')

colors = list(map(lambda x: '#002bff' if x == 1 else '#f7ab05', m_shift_labels))

ax2 = fig.add_subplot(1, 2, 2, projection = '3d')
#ax2.view_init(azim=0, elev=90)
ax2.scatter(processed_df[:, 0], processed_df[:, 1], processed_df[:, 2], c=colors, alpha=0.8)
ax2.set_zlabel('кратность изображения')
ax2.set_xlabel('дисперсия изображения')
ax2.set_ylabel('асимметрия изображения')
ax2.set_title('Mean-shift')

# images showing and saving
plt.savefig('banknote-authentication_clustering_done.svg')
plt.show()
```

Этапы выполнения работы:

1. Подключаем необходимые библиотеки (в том числе и sklearn):

```
import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib import rcParams
import pandas as pd
from sklearn.cluster import KMeans, MeanShift, estimate_bandwidth
```

2. Далее читаем csv-файл с помощью библиотеки pandas и удаляем столбец class. Также происходит приведение типов данных для корректной работы:

```
# Reading csv file
df = pd.read_csv('banknote-authentication.csv')

# Deleting class column
df = df.drop('class', axis=1)
processed_df = df.to_numpy()[:, :]
```

3. Затем используем алгоритм K-means. Укажем при инициализации k-means++. Это означает разумное определение значения K и начальных центральных точек кластера K, что влияет на результаты работы метода. .fit(processed_df) выполняет саму кластеризацию, далее происходит работа с ярлыками:

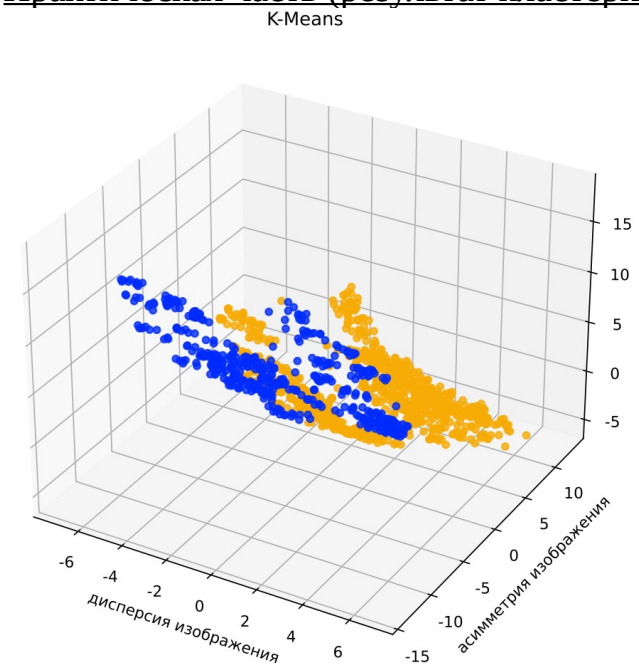
```
# K-means clustering
cluster_num = 2
k_means = KMeans(init='k-means++', n_clusters=cluster_num, n_init=12)
k_means.fit(processed_df)
k_means_labels = k_means.labels_
```

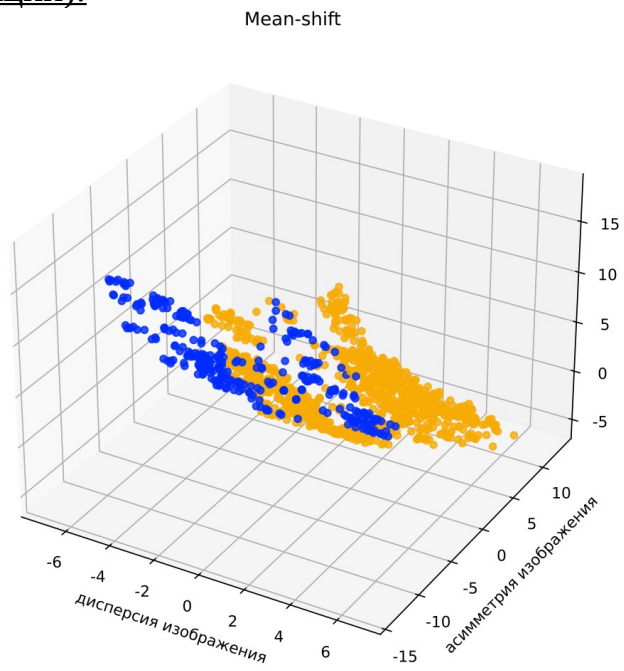
4. Для сравнения используем метод Mean-shift, который характеризуется пропускной способностью текст. В параметрах передаем датасет, медиану попарных расстояний и количество образцов (функция estimate_bandwidth() инициализирует поля bandwidth), кластеризацию выполняем аналогично k-means:

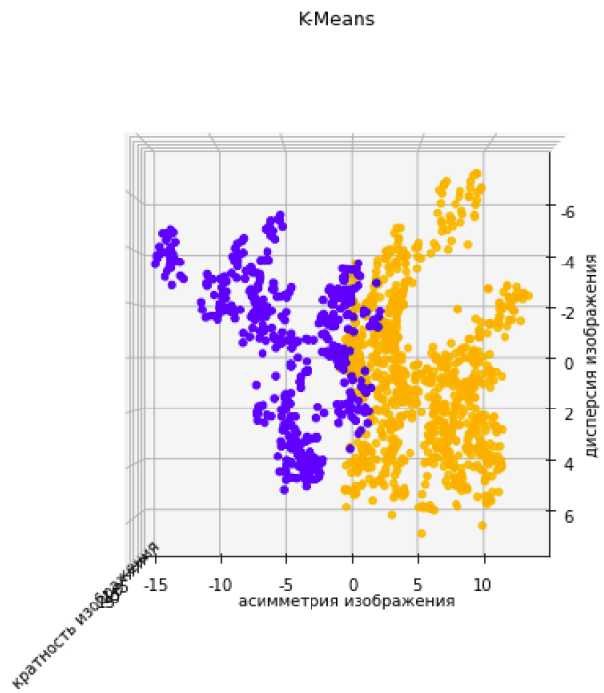
```
# Mean-shift clustering
bandwidth = estimate_bandwidth(processed_df, quantile=0.183, n_samples=1000)
m_shift = MeanShift(bandwidth=bandwidth)
m_shift.fit(processed_df)
m_shift_labels = m_shift.labels_
df['ms_cluster'] = m_shift.labels_
```

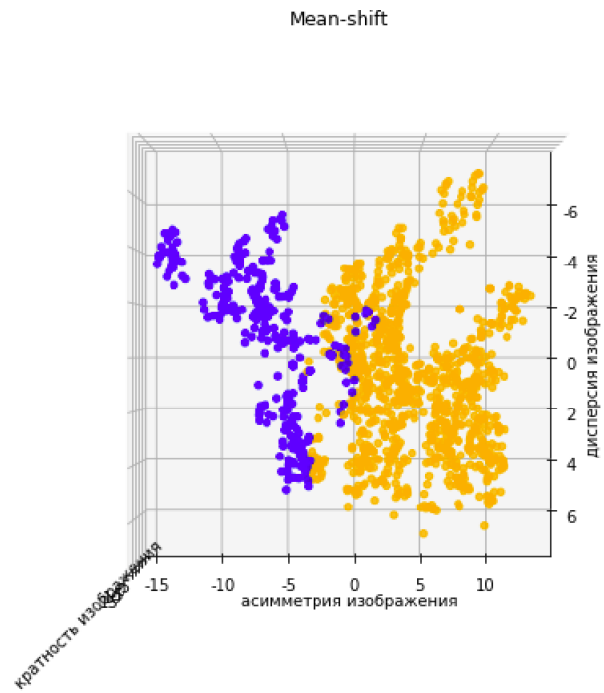
5. Оставшийся код применяется для настройки отображения результата кластеризации.

Практическая часть (результат кластеризации):









На графиках виден похожий результат выполнения двух алгоритмов, однако метод Mean-shift дает более детальный результат. По полученным сведениям можно отличить поддельные банкноты от настоящих.

Вывод: изучена задача и алгоритмы кластеризации.