СЕМЕРИКОВ А. В., ГЛАЗЫРИН М. А. КЛАСТЕРИЗАЦИЯ СТУДЕНТОВ ПО ПРИЗНАКУ УСПЕШНОСТИ ОКОНЧАНИЯ УНИВЕРСИТЕТА

УДК 001.891.57:330.47, ВАК 05.13.01:08.00.05, ГРНТИ 28.17.31

Кластеризация студентов по признаку успешности окончания университета

А. В. Семериков[1], М. А. Глазырин[2]

[1]Ухтинский государственный технический университет, г. Ухта
[2]Вятский государственный университет, г. Киров

В статье представлены результаты построения модели кластеризации студентов с помощью метода главных компонент и метода средних с использованием библиотек Python. Для построения модели использовалась структура DataFrame, в которой содержится обезличенное описание 36830 студентов по 13 признакам. Построение моделей осуществлялось без использования целевого признака (обучение без учителя). В ходе построения моделей было установлена плохая кластеризуемость студентов по признаку успешности окончания университета и низкая оценка предсказуемости отнесения студента к кластерам (успешное окончание университета, отчисление из университета).

Ключевые слова: *большие данные, объекты, кластеризация, главные компоненты, ближайшие соседи, Pandas, библиотека Python.*

Simulation of a process model of functioning of the enterprises for rendering of services

A.V. Semerikov[1], M.A. Glazyrin[2]

[1]Ukhta State Technical University, Ukhta
[2]Vyatka State University, Kirov

The article presents the results of constructing a student clustering model using the principal component method and the average method using Python libraries. To build the model, the DataFrame structure was used, which contains an impersonal description of 36830 students based on 13 features. The construction of the models was carried out without the use of the target attribute (learning without a teacher). During the construction of the models, poor clustering of students was established on the basis of successful graduation from the university and a low assessment of the predictability of attributing a student to clusters (successful graduation from the university, expulsion from the university).

Keywords: *big data, objects, decision tree, features, target feature, Pandas, Python library.*

Введение

После окончания обучения в университете студенту в случае успешного освоения предложенной программы выдают диплом и в персональных данных проставляется индекс 1. В противном случае ему выдают справку о положительных оценках по освоенным им дисциплинам. В персональных данных студента в первом

случае можно поставить индекс 1, а во втором случае можно поставить 0. Таким образом всех студентов можно разделить на два кластера.

При поступлении в ВУЗ студент предоставляет персональные сведения, которые представляют собой набор данных, состоящих из следующих параметров: год поступления, название института, специальность, форма обучения, категория конкурса, сумма баллов, средняя сумма баллов, инвалидность, льготы, должность, страна, регион, город.

Имея большое количество данных о студентах, используя методы машинного обучения, можно построить модели кластеризации данных. В настоящее время имеется целый ряд методов построения кластеров [1, 2, 3]. В предлагаемой статье рассматриваются два подхода: метод главных компонент и метод среднего. На основе кластеризации можно предсказать будущее студента первокурсника, то есть определить наиболее вероятный исход. Будущий студент может оказаться в кластере с индексом 0 или 1.

Экспериментальная часть

В настоящей статье представлено описание построения кластеров студентов с использованием данных в виде таблицы Microsoft Excel в формате «xlsx», в которых строки соответствуют набору признаков для описания отдельного студента, а столбцы соответствуют этим признакам (Таблица 1). Последний столбец «Факт окончания» представляет собой целевой признак, который в процессе построения модели кластеризации не используется (машинное обучение без учителя)

Таблица 1. Исходный набор данных по каждому студенту

Институт	Специальность	Форма обучения	Категория	Средний балл	Пол	Общежитие	Семейное положение	Медаль	Тип школы	Лет после школы	Страна	Город	Факт окончания
СТИ	Электропривод и автоматика промышленных установок и технологических комплексов	очная	общий конкурс	63,3	м	да	не женат	нет медали	школа	5	Россия	Ухта	1
ИнЭУиИТ	Автоматизированные системы обработки информации и управления	заочная	общий конкурс	65	м	да	не женат	серебряная	училище	7	Россия	Ухта	1
…	…	…	…	…	…	…	…	…	…	…	…	…	…

Из таблицы видно, что каждый студент характеризуется 13 признаками.

Для решения задачи по кластеризации студентов все категориальные признаки заменяются на числовые (таблица 2).

Таблица 2. Числовой набор данных по каждому студенту

Институт	Специальность	Форма обучения	Категория	Средний балл	Пол	Общежитие	Семейное положение	Медаль	Тип школы	Лет после школы	Страна	Город	Факт окончания
3	45	2	1	63,3	0	0	1	1	4	5	1	4	1
2	43	1	1	65	1	1	1	3	5	7	1	4	1
…	…	…	…	…	…	…	…	…	…	…	…	…	…

Таким образом дано множество объектов, которые необходимо разбить на несколько групп (кластеров), состоящих из похожих друг на друга объектов. В рассматриваемой задаче кластеризации количество кластеров задано изначально и составляет 2. В тоже время оно может определятся в ходе проведения исследования данных. Близость объектов обычно определяется через расстояние в многомерном пространстве признаков.

Для кластеризации студентов по близости признаков использовалась интерактивная вычислительная среда Jupyter Notebook [4], где представлены библиотеки Python, позволяющие провести машинное обучение и тестирование алгоритма кластеризации:

import numpy as np, import pandas as pd, import matplotlib.pyplot as plt, import seaborn as sns, import warnings, warnings.filterwarnings("ignore"), plt.style.use("ggplot")

Для проведения кластеризации считываем данные из файла ugtuFF.xlsx с помощью библиотеки Pandas:

df=pd.read_excel(io='ugtuFF.xlsx', engine='openpyxl')

Как известно, кластеризация данных будет осуществлена более качественно, если в наборе данных будет меньше попарно сильно коррелированных признаков. Поэтому вначале определим коэффициенты корреляции признаков (Рисунок 1).

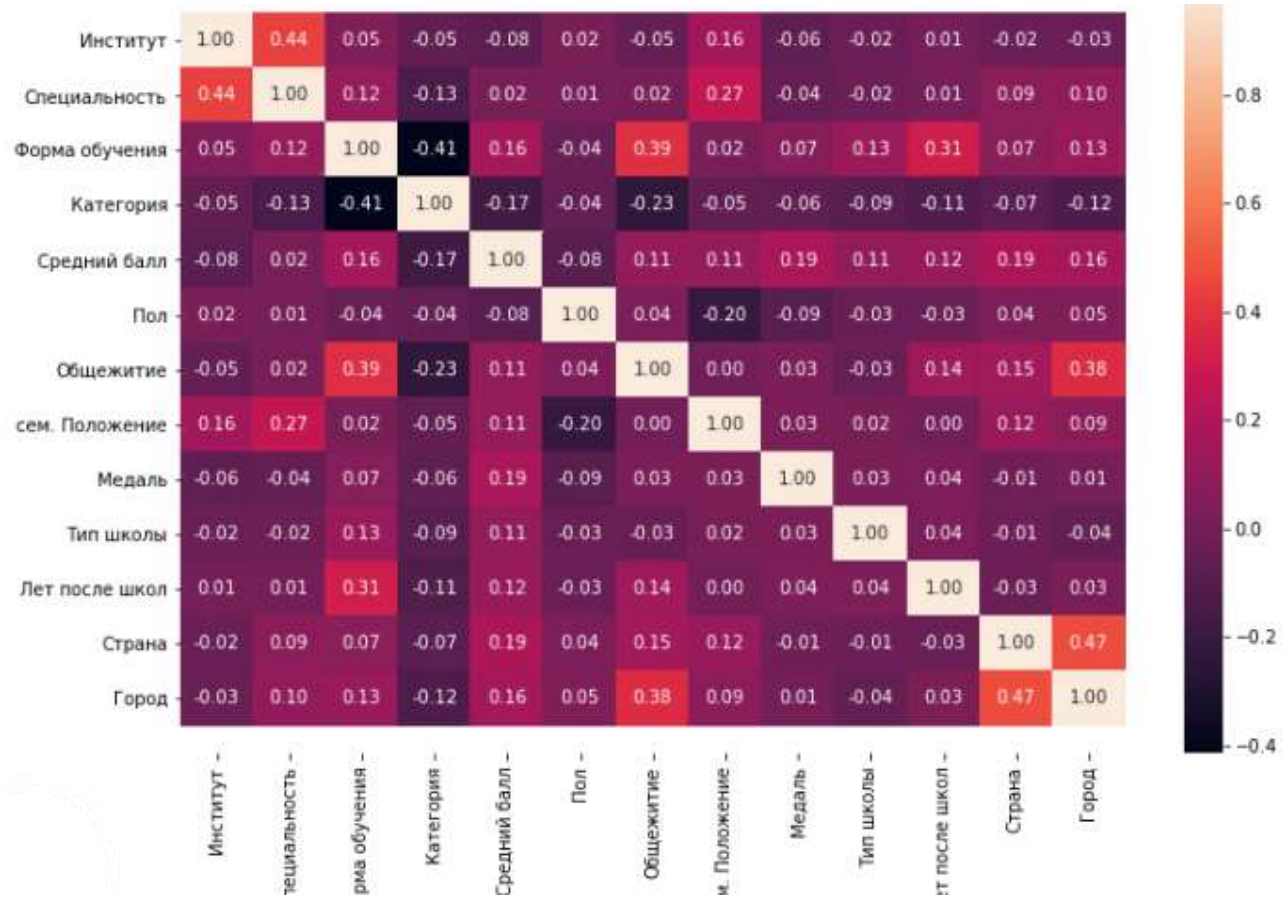

Рисунок 1. Корреляция признаков

Как видно из рисунка 1 между параметрами «Университет» и «Специальность», «Страна» и «Город» имеет место довольно высокий коэффициент корреляции. Поэтому из набора данных удаляются параметры «Страна» и «Институт». Наряду с этим удаляется и целевой признак «Факт окончания», так как он не используется в процессе кластеризации:

df.drop(["Факт окончания", "Страна", "Институт"], axis=1, inplace=True)

Используя библиотеку seaborn sns.pairplot(df, hue="Факт окончания") представляется возможным визуально оценить попарную связь параметров (Рисунок 2).

На основании визуального анализа полученных связей параметров (Рисунок 2) было установлено, что для всех параметров наблюдается слабое разделение объектов по целевому признаку. Наиболее выражено это для признаков «Медаль», «Общежитие», «Категория» и «Форма обучения». По этой причине они так же удаляются из набора данных. Отмеченная особенность рассматриваемых данных свидетельствует о плохой их кластеризации по признаку «Факт окончания». С практической точки зрения эта кластеризация является приоритетной.

df.drop(["Медаль", "Общежитие", "Категория", "Форма обучения"], axis=1, inplace=True)

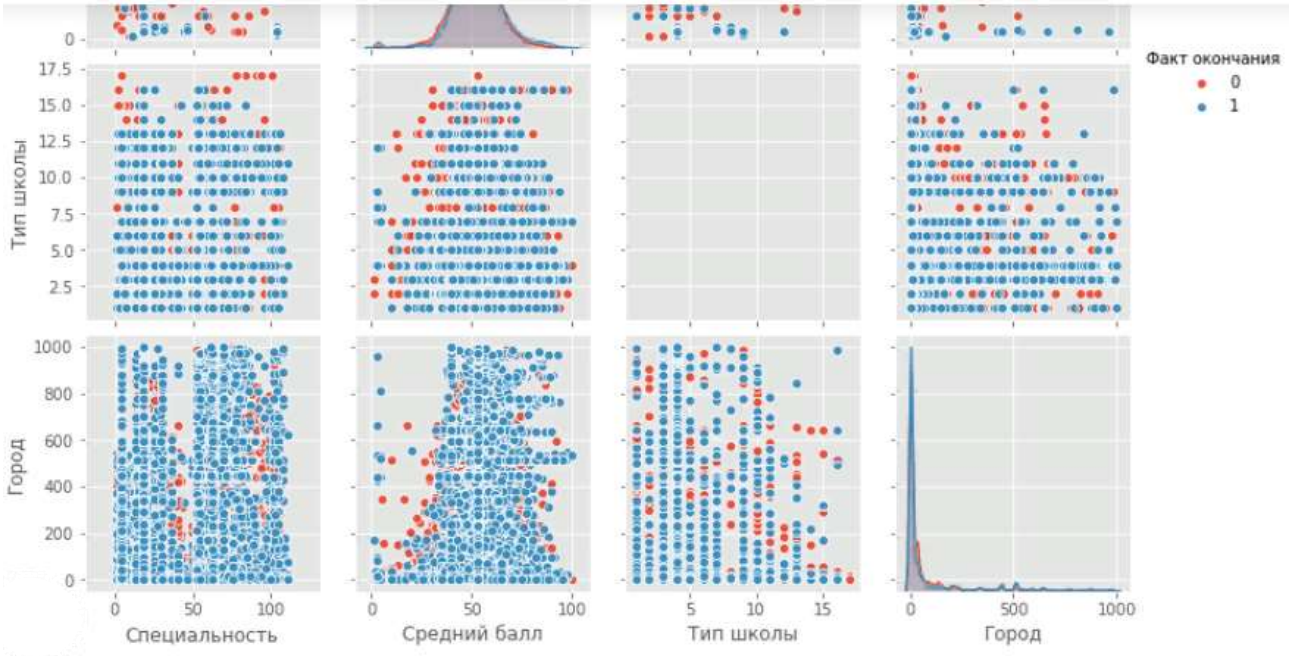

Рисунок 2. Фрагмент изображения связей параметров

Таким образом для проведения кластеризации имеется набор данных из 36830 студентов с 8 параметрами (Рисунок 3).

	Специальность	Средний балл	Пол	сем. Положение	Тип школы	Лет после школ	Город	Факт окончания
0	1	52.7	1	1	1	1	1	0
1	2	26.7	1	1	2	2	2	0
2	3	63.3	1	1	2	1	3	1
3	4	26.3	1	1	3	3	4	1
4	5	44.7	1	1	1	4	5	0

Рисунок 3. Числовой набор данных по каждому студенту в представлении Pandas

Наиболее подходящий набор параметров может быть определен на основе проведения дополнительного исследования. Данный набор можно рассматривать как один из примеров решения задачи кластеризации. В любом случае в наборе данных не должен находится целевой столбец «Факт окончания»:

scal_data, y=StandardScaler().fit_transform(df.drop("Факт окончания", axis=1)), df["Факт окончания"]

Для улучшения качества кластеризации воспользуемся методом главных компонент (PCA), с помощью которого можно снизить размерность путем определения ортогональных направлений, вдоль которых наблюдается наибольший разброс (выборочная дисперсия). Эти направления называются главными компонентами.

Применим метод PCA для представления многомерных данных (восемь измерений) в двухмерном пространстве. Для этого возьмем две первые главные компоненты и спроецируем представленные данные (Рисунок 3) на них. При этом вначале признаки отмасштабируем:

pca=PCA(n_components=2)

Используя библиотеку sklearn и методы fit и transform представим исходное обучающее множестве в формате 2D:

data_pca2D = pca.fit_transform(scal_data)

Используя PCA-представление определим:

```
for i, component in enumerate(pca.components_):
print("{} component: {}% of initial variance".format(i+1, round
(100*pca.explained_variance_ratio_[i], 2)))
print(" + ".join("%.3f x %s" % (value, name) for value, name in
zip(component,df.columns)))
```

В результате две главные компоненты в нашем PCA-представлении данных и процент исходной дисперсии в данных имеют такой вид:

1 component: 20.89% of initial variance
*0.439*Специальность+0.436*Средний балл–0.320*Пол+0.590*сем. Положение+0.128*Тип школы+0.193*Лет после школ+0.337*Город*
2 component: 16.58% of initial variance
*-0.473*Специальность+0.454*Средний балл–0.125*Пол–0.264*сем. Положение+0.514*Тип школы+0.461*Лет после школ–0.088*Город*

На основании полученных расчетов можно сказать, что модель с двумя компонентами объясняет разброс данных на 37,47 %. При этом в первой компоненте все параметры кроме одного имеют одинаковую положительную направленность. Во второй компоненте три параметра имеют отрицательную направленность. Тем самым уменьшая пространство для объектов и уменьшая разброс.

Для визуализации пространства данных изобразим объекты двумя способами в новом 2D пространстве:

plt.scatter(data_pca2D[:,0], data_pca2D[:,1], c=y);

Цвета точек представляют точки из целевого столбца «Факт окончания» (Рисунок 4, Рисунок 5).

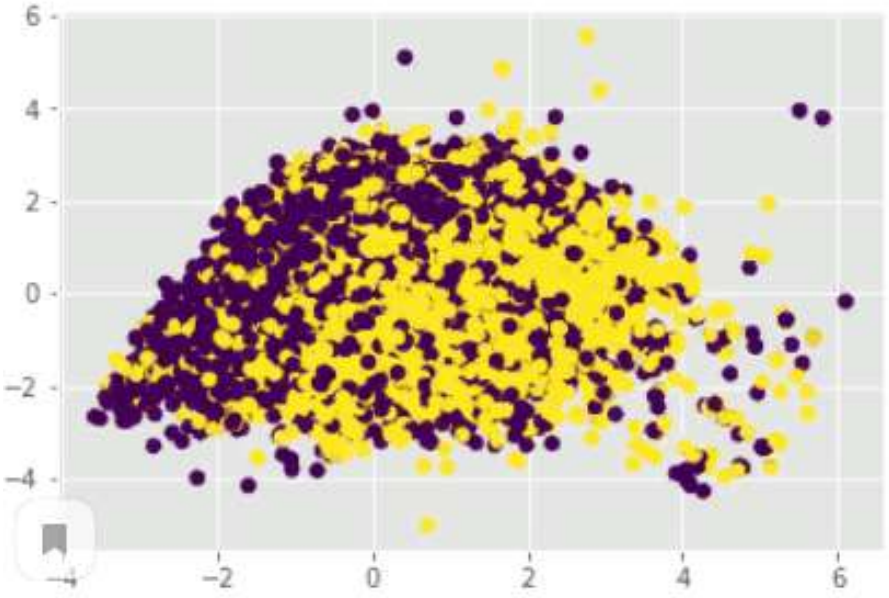

Рисунок 4. Визуальное изображение кластеров согласно целевому столбцу

plt.plot(data_pca2D[y == 0, 0], data_pca2D[y == 0, 1], 'bo', label='Отчислен из университета')
plt.plot(data_pca2D[y == 1, 0], data_pca2D[y == 1, 1], 'go', label='Успешно закончил университет')
plt.legend(loc=0);

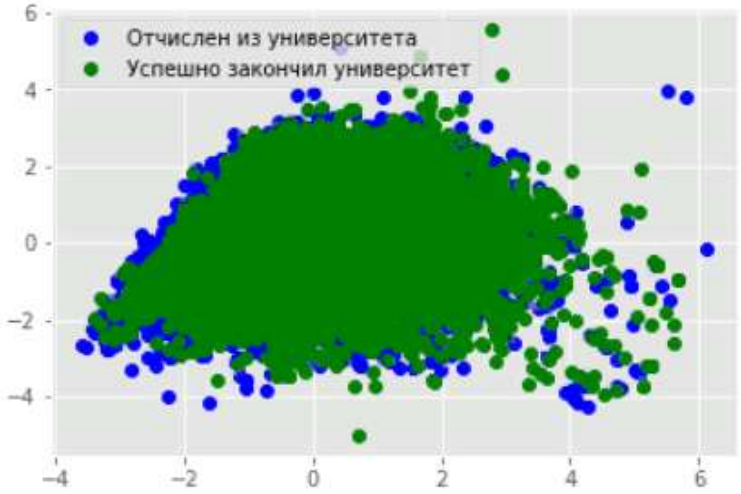

Рисунок 5. Визуальное изображение кластеров согласно целевому столбцу

Как видно из Рисунков 4, 5 использование двух координат не очень хорошо описывает модель кластеризации. Для повышения описания разброса данных возьмем модель с тремя главными координатами

for i, component in enumerate(pca.components_):
print("{} component: {}% of initial variance".format(i + 1, round(100 pca.explained_variance_ratio_[i], 2)))*
print("+".join("%.3f x %s" % (value, name) for value, name in zip(component, df.columns)))

В результате получаем следующие три главные координаты:

1 component: 20.89% of initial variance
*0.439*Специальность+0.436*Средний балл–0.320*Пол+0.590*сем. Положение+0.128*Тип школы+0.193*Лет после школ+0.337*Город*
2 component: 16.58% of initial variance
*-0.473*Специальность+0.454*Средний балл–0.125*Пол–0.264*сем. Положение+0.514*Тип школы+0.461*Лет после школ–0.088*Город*
3 component: 15.9% of initial variance
*0.086*Специальность+0.228*Средний балл+0.636*Пол–0.269*сем. Положение–0.185*Тип школы+0.206*Лет после школ+0.621*Город*

которые описывают 53.37 % разброс данных. Это лучше, чем при двух координатам.

Этот результат лучше предыдущего. Принято считать, что модель хорошо описывает набор данных, если она объясняет 90 % их разброса:

```
pca = PCA(0.9).fit(scal_data)
print('We need %d components to explain 90% of variance' % pca.n_components_)
```

Для объяснения 90 % дисперсии в данном случае потребуется 8 компонент.

Изобразим получившиеся точки в 3D-пространстве (Рисунок 6). Цвета точек соответствуют значениям целевого столбца:

```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(1, figsize=(8, 6))
ax = Axes3D(fig, elev=-150, azim=110)
ax.scatter(data_pca[:, 0], data_pca[:, 1], data_pca[:, 2], c=y, cmap=plt.cm.Set1, edgecolor='k', s=40)
ax.set_title("First three PCA directions")
ax.set_xlabel("1 component")
ax.w_xaxis.set_ticklabels([])
ax.set_ylabel("2 component")
ax.w_yaxis.set_ticklabels([])
ax.set_zlabel("3 component")
ax.w_zaxis.set_ticklabels([])
plt.show()
```

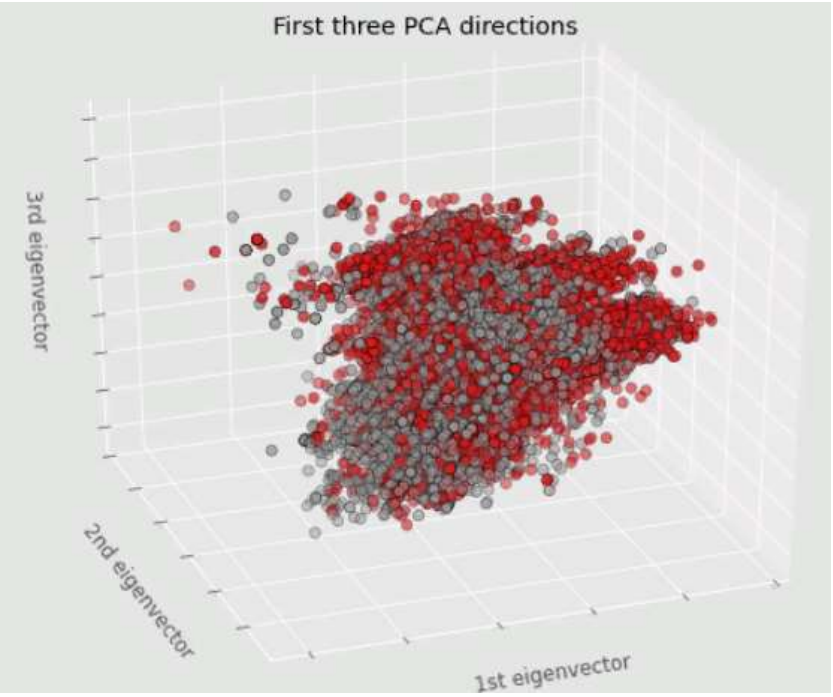

Рисунок 6. 3D изображение кластеров

Как видно из Рисунка 6 кластеризация данных на основе целевого признака со значениями 0 и 1 представляется проблематичной.

Рассматриваемую задачу кластеризации данных можно решить с помощью метода k-средних. Основная идея метода заключается в том, что на каждой итерации пересчитывается центр масс (центроид) для каждого кластера, полученного на предыдущем шаге, затем объекты снова разбиваются на кластеры согласно тому, какой из новых центроидов находится ближе.

Для использования метода k-средних импортируем метод KMeans:

```
from sklearn.cluster import KMeans, MeanShift, AgglomerativeClustering
```

Применим метод k-средних, указывая количество кластеров укажем в параметре n_clusters=2:

```
kmeans=KMeans(n_clusters=2, random_state=50, max_iter=100)
```

Сделаем предсказание на данных, спроектированных в 2D-пространство. Получим принадлежность каждого объекта одному из двух значений целевого столбца:

```
kmeans.fit(data_pca2D)
pred_kmeans = kmeans.fit_predict(data_pca2D)
```

Предсказанное значение столбца:

```
pred_kmeans
array([0, 0, 0, ..., 0, 0, 0])
```

Истинное значение столбца:

y.values
array([0, 0, 1, ..., 1, 0, 0], dtype=int64)

Изобразим кластеризацию, полученную методом k-means. Сравним с рисунком выше, где цвета были взяты по данным целевого столбца:

plt.scatter(data_pca2D[:,0], data_pca2D[:,1], c=pred_kmeans)

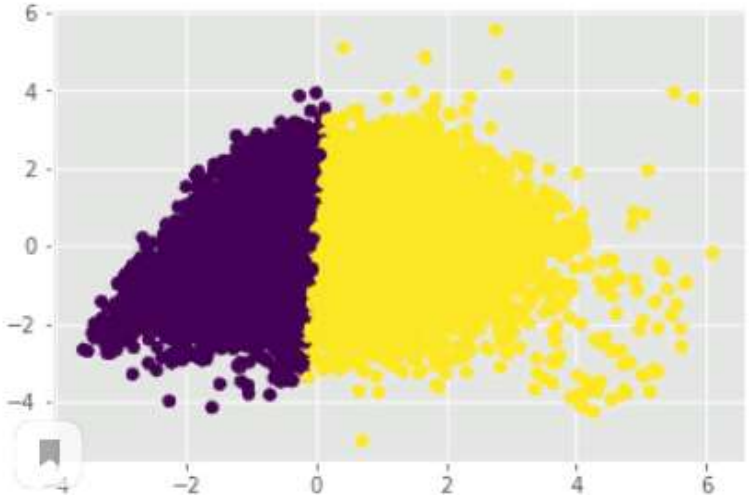

Рисунок 7. Кластеризация методом k-means

Изобразим данные с центроидами (Рисунок 7):

plt.plot(data_pca2D[pred_kmeans ==0,0], data_pca2D[pred_kmeans==0,1], 'bo', label='Отчислен из университета')
plt.plot(data_pca2D[pred_kmeans ==1,0], data_pca2D[pred_kmeans==1,1], 'co', label='Успешно закончил университет')
plt.plot(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],'ro')
plt.legend(loc=0)
plt.title('k-means')

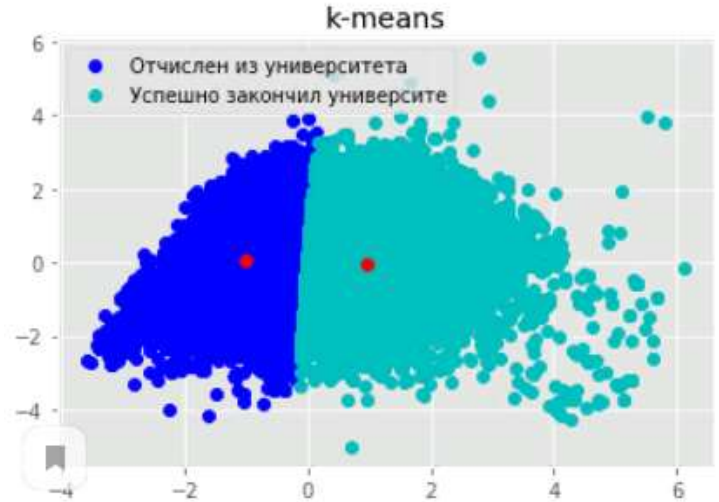

Рисунок 8. Изображение кластеризация методом k-means с центроидами

Оценим качество кластеризации при помощи индекса Adjusted Rand Index (ARI):

from sklearn import metrics
kmeans.fit(data_pca2D)
metrics.adjusted_rand_score(y, kmeans.labels_)
ARI=0.0088

Это значит разбиение на кластеры получилось не совпадающим с истинным разбиением. Они не схожи.

Сделаем кластеризацию методом k-means на данных, полученных при помощи PCA в 3D-пространство (Рисунок 9)**:**

plt.scatter(data_pca[:,0], data_pca[:,1], c=pred_kmeans)

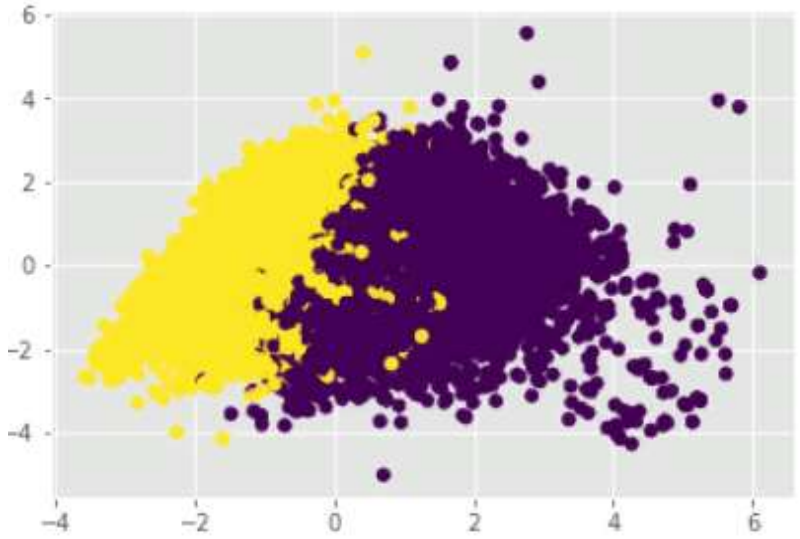

Рисунок 9. Изображение кластеризация методом k-means с использованием пространства тремя координатами

В этом случае оценка качества получилась равной 0.0075, что хуже, чем при 2D-представлении.

Результаты

1. Рассмотрена кластеризации студентов методом средних.
2. Установлено, что эффективность метода средних не улучшается при увеличении количества главных компонент.
3. Установлена невысокая точность предсказания при использовании метода k-средних.
4. Установлена низкая кластеризуемость исходных данных по целевому признаку, который принимает значения 0 и 1, что указывает на необходимость введения их промежуточных значений.

Список использованных источников и литературы

1. Как работает метод главных компонент (PCA) на простом примере [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/304214/ (дата обращения: 24.05.2021).

2. Обучение без учителя: 4 метода кластеризации данных на Python [Электронный ресурс]. – Режим доступа: https://proglib.io/p/unsupervised-ml-with-python (дата обращения: 20.05.2021).

3. Открытый курс по машинному обучению [Электронный ресурс]. – Режим доступа: https://www.youtube.com/watch?v=p9Hny3Cs6rk (дата обращения: 21.05.2021).

List of references

1. How principal component analysis (PCA) works with a simple example, https: //habr.com/ru/post/304214/, accessed May 24, 2021.

2. Unsupervised Learning: 4 Methods for Clustering Data in Python, https: //proglib.io/p/unsupervised-ml-with-python/, accessed May 20, 2021.

3. Open Course in Machine Learning, https://www.youtube.com/ watch?v=p9Hny3Cs6rk, accessed May 21, 2021.