Лабораторная работа №2.

Анализ предпочтений пользователей социальных сетей.

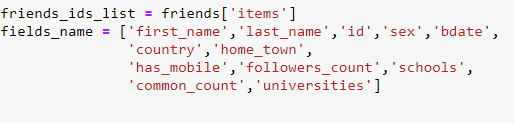
План:

1. Получить как можно больше данных пользователей социальной сети vk.com. Это может быть: Имя (first\_name), Фамилия (last\_name), Уникальный номер пользователя (id), Пол (sex), Дата рождения (bdate), Страна (country), Родной город (home\_town), Наличие телефона (has\_mobile), Количество подписчиков (followers\_count), Школы (schools), Количество общих друзей (common\_count), Университеты (universities) и обязательно группы и предпочтения.
2. Провести предобработку данных.
3. Провести кластерный анализ получившихся данных. Описать полученные кластеры.
4. Спроектировать автоматизированную систему, позволяющую определенным кластерам «навязывать» (рекомендовать) ту или иную информацию.

# 

## Описание исследуемых данных

Будут рассмотрены данные о моих друзьях в социальной сети VK (ВКонтакте). Для каждого пользователя собирается информация по некоторым признакам: Имя (first\_name), Фамилия (last\_name), Уникальный номер пользователя (id), Пол (sex), Дата рождения (bdate), Страна (country), Родной город (home\_town), Наличие телефона (has\_mobile), Количество подписчиков (followers\_count), Школы (schools), Количество общих друзей (common\_count), Университеты (universities)



Для удобной работы с VK API удобно использовать библиотеку в питоне VK. Далее передаем в API параметры access\_token и version (индивидуальный токен и версия API соответственно). Ниже продемонстрирован отрывок кода для отправки запроса на серверы VK.



Из выявленных в ходе исследования проблем можно выделить ограничение количества запросов со стороны VK, который составляет не более одного запроса в секунду для каждого поля. В данной работе совершается запрос по 12 полям для 240 друзей, и время парсинга составляет 48 минут для не очень большого датасета. Чтобы избежать ограничения в коде, используется функция sleep (1), которая после каждого запроса приостанавливает работу кода на одну секунду. Для парсинга размером в 508 страниц потребуется почти полтора часа времени.

В результате получаем таблицу с необработанными данными. Пример представлен ниже:

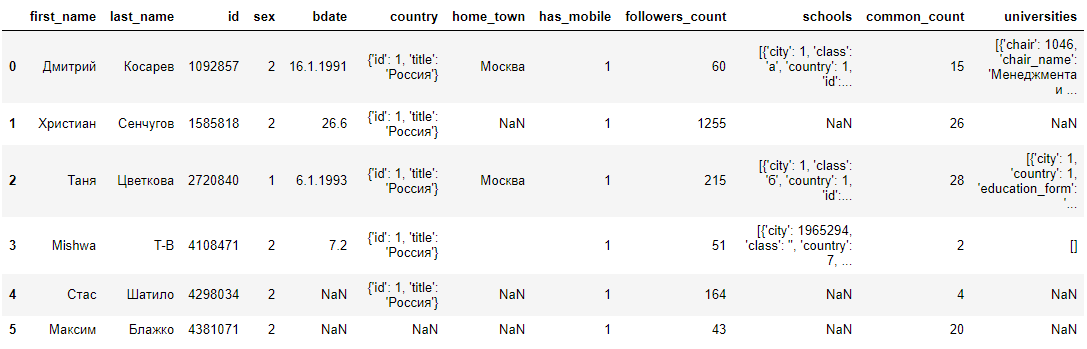


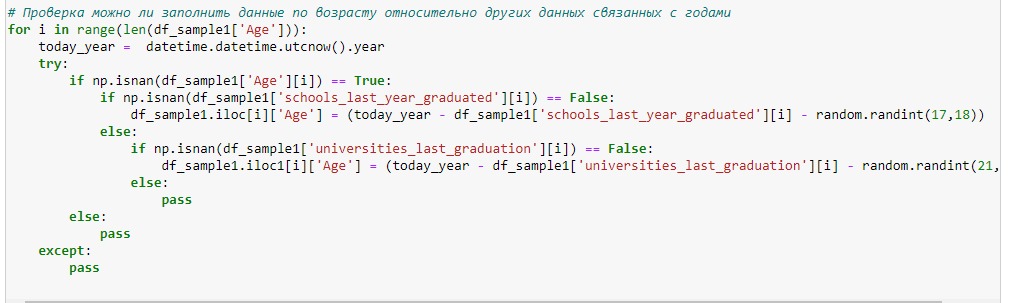
Таблица данных 1

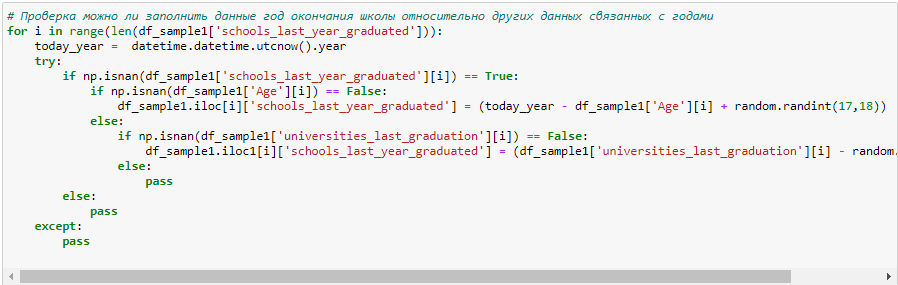
После анализа первых результатов стало ясно, что для выполнения работы придётся вводить дополнительные столбцы для данных о школе, университете. У некоторых пользователей отмечено несколько школ и университетов, однако рассматриваться будут только последние (по дате) указанные поля, подразумевая, что скорее всего, если человек есть у вас в друзьях, то он относительно недавно к вам добавился (многие со временем удаляют страницы, друзей). Как следствие, сформируем новые столбцы: Город последней школы (schools\_last\_city), Уникальный номер последней школы (school\_last\_id), Год выпуска из последней школы (schools\_last\_year\_graduated), Город последнего университета (universities\_last\_city), Страна последнего университета (universities\_last\_country), Выпуск из последнего университета (universities\_last\_graduation), Уникальный номер факультета последнего университета (universities\_last\_faculty), Уникальный номер последнего университета (universities\_last\_id).

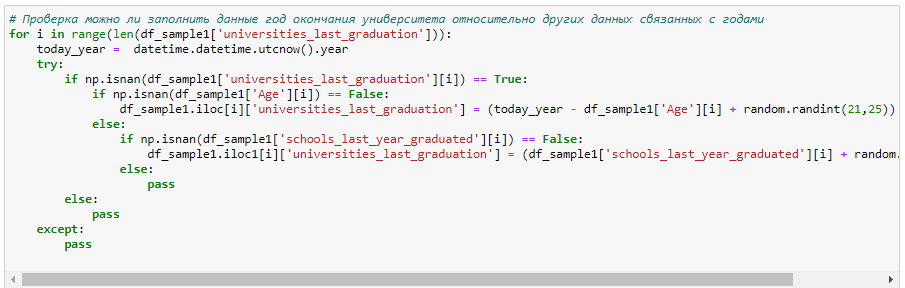
Некоторые пользователи ограничили в настройках доступ к информации об их годе рождения. Напишем функцию для перевода даты рождения в возраст (если указаны все данные, иначе не заполняем поле):



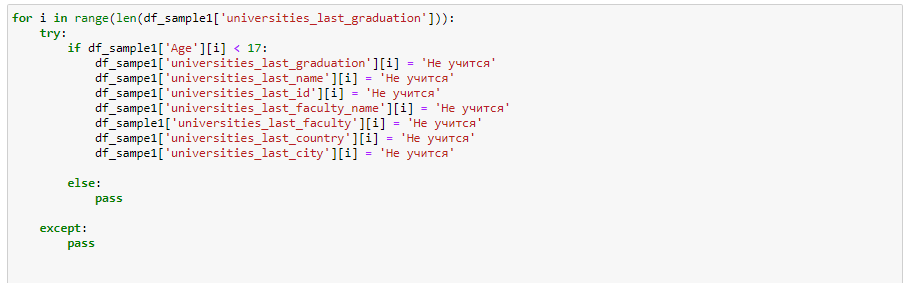
Довольно большое количество людей всё-таки не указывает свой год рождения, однако для работы с моделями нам нужно полностью заполнить все поля датасета, поэтому постараемся получить необходимую информацию через приблизительную оценку их возраста, используя данные о годе выпуска из школы, университета. Также можем приблизительно оценить, в каком году человек закончил школу и университет. Для этого выполним код:



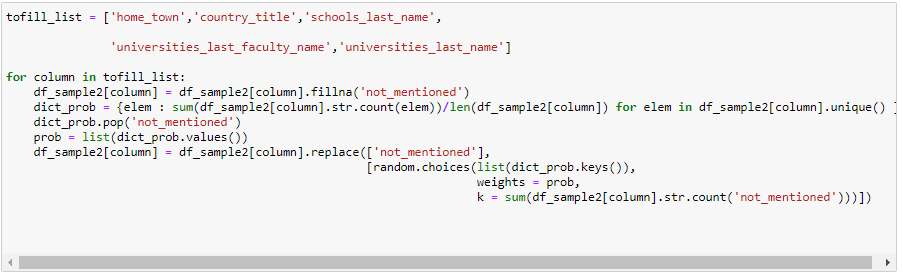




Помимо этого, если пользователь моложе 17 лет, то во всех полях с университетом ставим значение “Не учится”. Код представлен ниже.



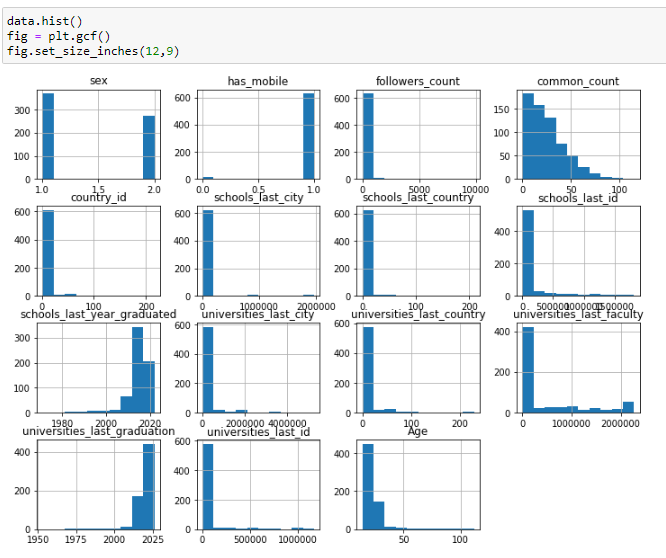
Нерешённой остаётся проблема с неуказанными городами, университетами, факультетами, школами, поэтому на основе данных, которые имеем, будем заполнять отсутсвующие поля с определенной вероятностью (на основе остальных данных, которыми обладаем):



После того как заполнили все поля, исследуемые данные готовы к препроцессингу, выявлению аномалий.

Для начала построим гистограммы для каждого численного столбца нашего датасета. Поле sex принимает значения 1 и 2 для женского и мужского пола соответсвенно, Has\_mobile – 0 и 1 означает наличие телефона, остальные поля имеют уникальные номера, поэтому их нельзя анализировать на предмет выбросов и аномалий.

Стоит обратить внимание на поля followers\_count, common\_count, Age.



Гистограмма признаков 1

Построим диаграмму рассеивания (scatter plot) для признаков followers\_count по оси OX, common\_count по оси OY.

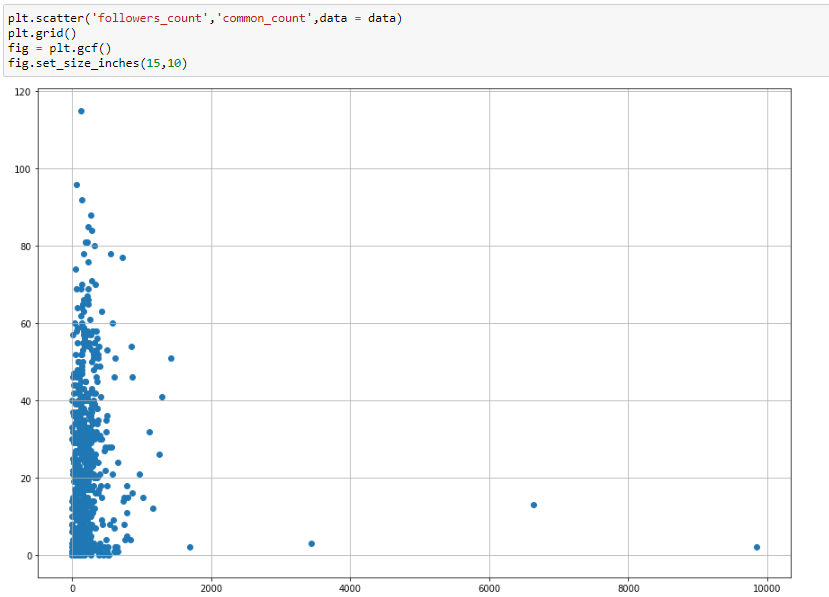
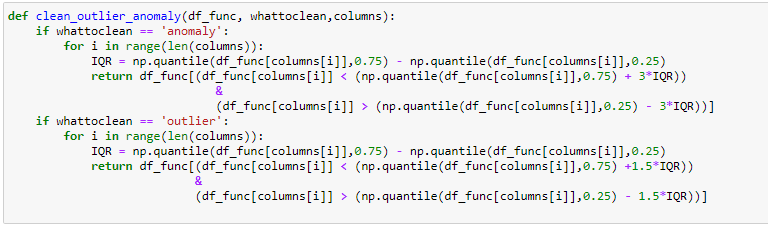
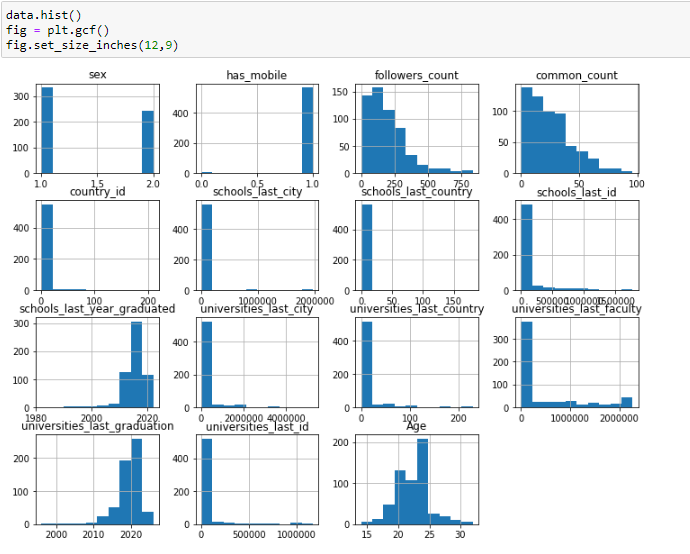


Диаграмма рассеивания данных 1

Можем заметить некоторые выделяющиеся данные, такие как число подписчиков (около 10000, 6400, 3750) и общие друзья (120). Напишем функицю для выявления и очистки данных от аномалий:



В качестве основы алгоритма использовали IQR (межквантильный размах). Удаляем значения, которые превышают значения квантиль 75 % + 3\* IQR и меньше квантиля 25% - 3\* IQR. Посмотрим на результат:

**

Гистограмма признаков 2

В итоге можно заметить, как поменялись гистограммы. Гистограмма Age стала похожа на нормальное распределение (но таковым не является). Посмотрим, как преобразовалась диаграмма рассеивания:

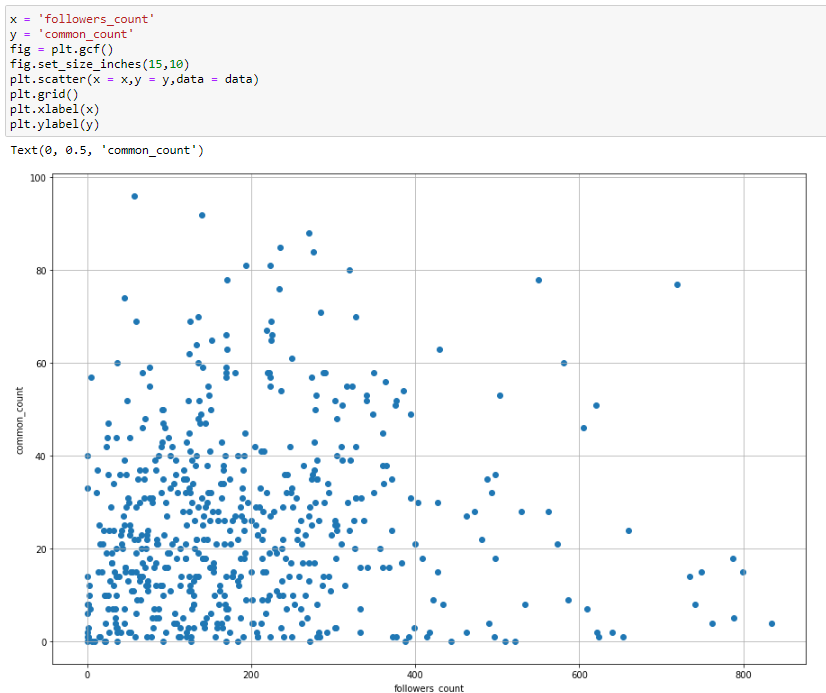
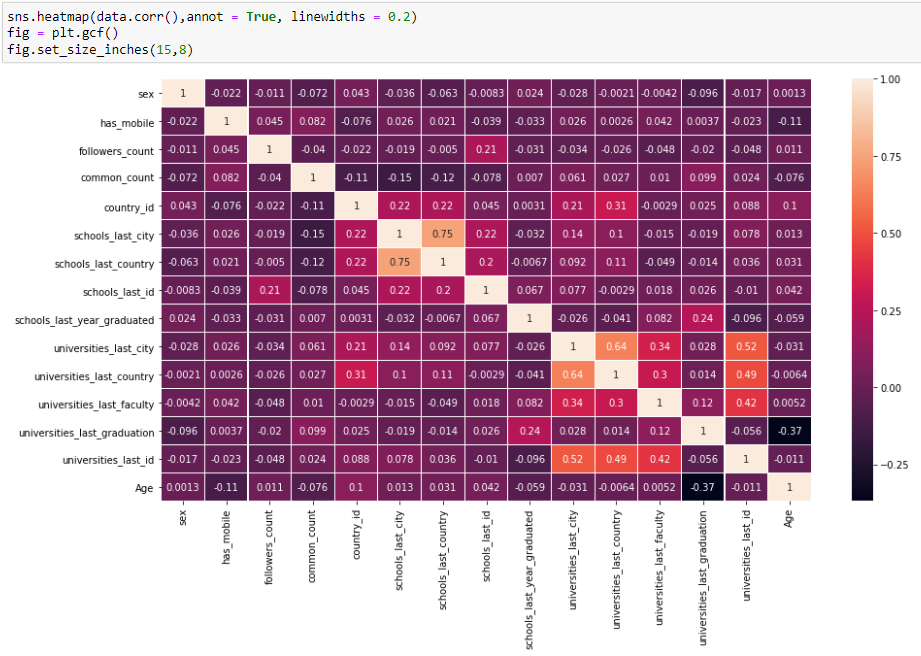


Диаграмма рассеивания данных 2

Предпологаемые значения оказались аномальными и следует их удалить.

Проверим корелляцию и визуализируем её с помощью тепловой карты

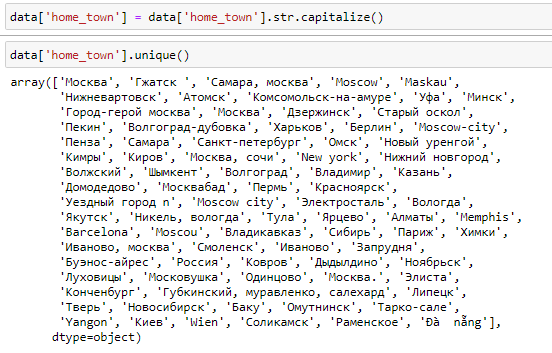


Тепловая карта корреляций признаков 1

Можно заметить высокую корелляцию между городом и страной университета (что весьма логично), городом и страной школы. Корелляция больше 0.2 принимается между университетскими и школьными признаками.

## Глава 1.2. Представление категориальных данных для компьютерной обработки

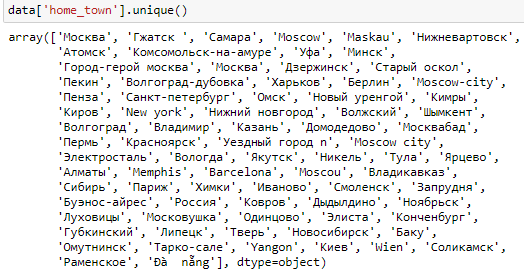
Первым этапом решения задачи кластеризации будет перевод категориальных данных в понятный для компьютера вид (числовые признаки). В нашем случае это столбец home\_town. Рассмотрим его уникальные значения:



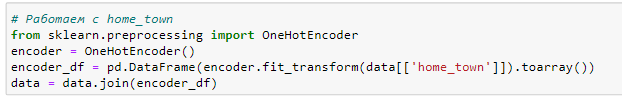
Некоторые люди указывают несколько городов в качестве своих родных (home\_town), однако будут учитываться только первое значение. В нескольких случаях пользователи указывали названия городов с точкой в конце, что в данной ситуации мешает нашему исследованию. Попробуем это исправить. Для этого воспользуемся алгоритмами Python 3.7.



Уникальные значения городов стали принимать вид:



Далее воспользуемся OneHotEncoder. Этот кодировщик берёт столбец с категориальными данными и для каждого уникального значения списка создаёт столбец с бинарным значениями 0 и 1, которые несут значения истины и лжи. В языке программирования Python 3.7. данная реализация предусмотрена в библиотеке sklearn в модуле preprocessing и имеет название OneHotEncoder.

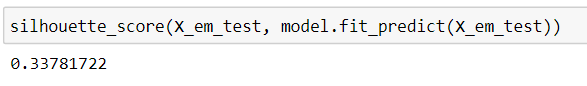


В результате проведенных операций, получили данные, полностью подгтовленные к работе с алгоритмами машинного обучения.

В данной работе будут использованы алгоритмы машинного обучения без учителя – кластеризации. Согласно своему определению, кластеризация – это задача группировки множества объектов на подмножества таким образом, чтобы объекты из одного кластера были более похожи друг на друга, чем объекты из других кластеров, по какому-либо критерию. Для оценки качества моделей были использованы следующие метрики: Fit time, Silhouette score, Davies Bouldin score, Inertia. Выбор метрик обусловлен тем фактом, что именно эти метрики показывают качество моделей для неразмеченных данных, каковыми являются исследуемые данные.

Fit time показывает время обучения модели. Эту метрику будем стараться минимизировать. Относительно этого датасета показатель времени довольно низкий, но при повышении размера выборки время будет увеличиваться в зависимости от сложности алгоритма, поэтому эту метрику следует проанализировать.

Silhouette score используется в задачах кластеризации для изучения расстояния между полученными кластерами. Эта метрика принимает значения от -1 до 1. Коэффициент близкий к 1 указывает на то, что образец находится далеко от соседних кластеров. Значение 0 указывает, что выборка находится на границе между двумя соседними кластерами, а отрицательные значения указывают на то, что эти выборки могли быть назначены не в тот кластер, поэтому это значение будем стараться максимизировать. Данная метрика реализуется на языке программирования Python 3.7. в библиотеке sklearn в модуле metrics.

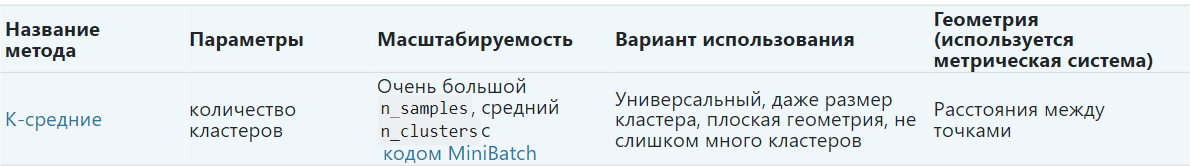


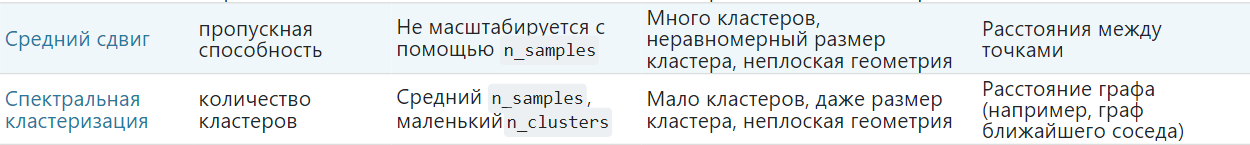
Davies Bouldin score используется как средняя мера сходства каждого кластера с наиболее похожим кластером, где сходство – это отношение расстояний внутри кластера и между кластерами. Минимальная оценка равна нулю, более низкие значения указывают на лучшую кластеризацию, поэтому будем минимизировать это значение. Данная метрика реализуется на языке программирования Python 3.7. в библиотеке sklearn в модуле metrics.



Inertia показывает сумму квадратов расстояний от образцов до ближайшего. Этот параметр будем стараться минимизировать.

Для данной работы были отобраны некоторые модели кластеризации: KMeans, MiniBatchKMeans, MeanShift, SpectralClustering, AgglomerativeClustering. Основные различия можно отметить по нижеприведенной таблице:





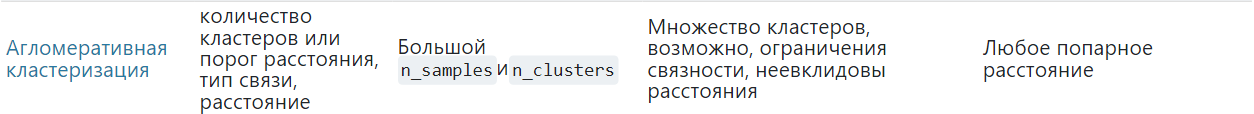
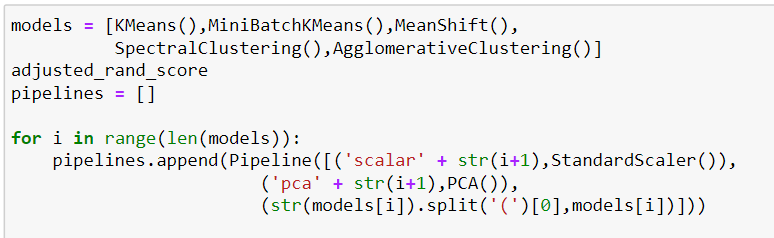


Таблица алгоритмов кластеризации 1

Для их сравнения между собой создадим контейнер, который включает в себя нормализацию данных (масштабирование), метод главных компонентов (для понижения размерности) и саму модель.



Проверим метрики для алгоритмов кластеризации на тестовых данных. Данные были раздлены в соотношении 80/20, где 80% составляют обучающие данные, а 20% – тестовые. Подобное разделение обосновывется не очень большим размером исследуемых данных, в результате чего возможно неодобучение модели. В итоге на одних и тех же данных получили следующие результаты обучения. Ниже представлена таблица результатов замеров:

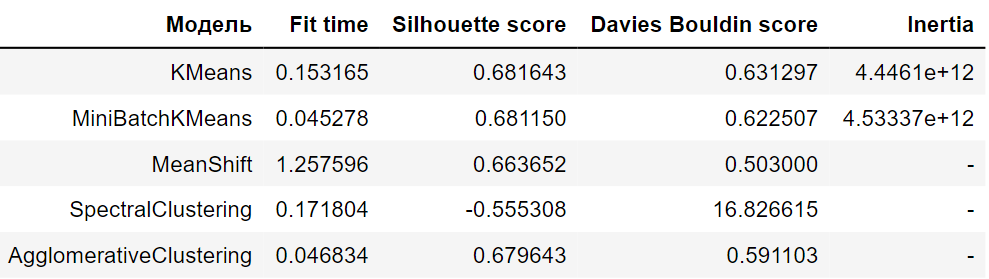
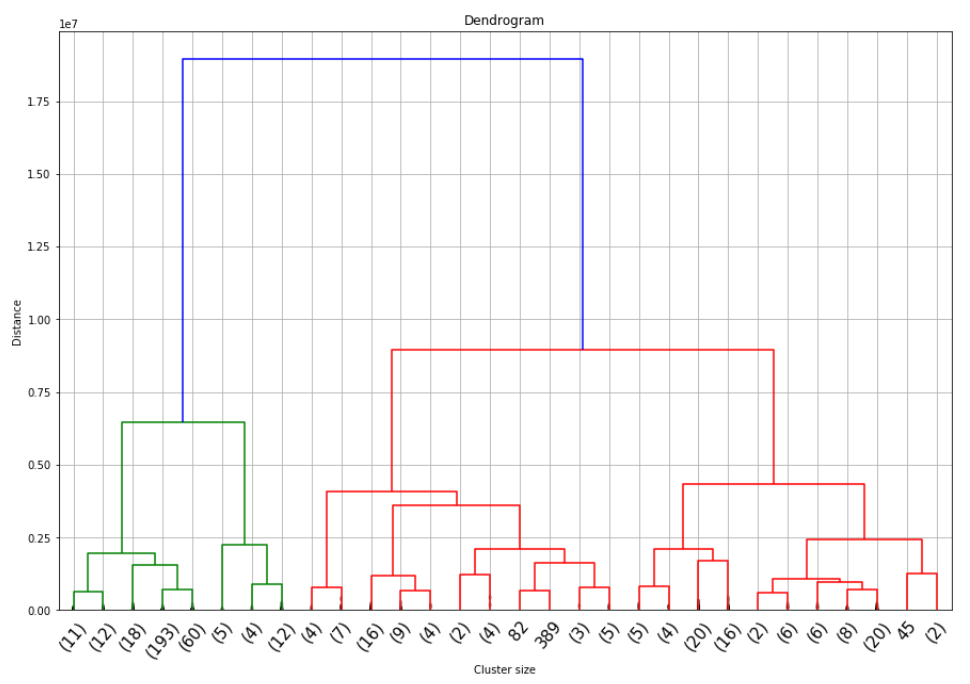


Таблица метрик алгоритмов 1

Как можно заметить, лучшее время обучения составило 0.045278 секунд. Лучшиее значение для Silhouette score – 0.681643, а для Davies Bouldin score – 0.503000. В итоге, выбираем модель MiniBatchKMeans за высокую скорость и относительно хорошие другие метрики (0.681150 и 0.622507 для метрик Silhouette score и Davies Bouldin score соответсвенно). MiniBatchKMeans также подходит для анализа небольших выборок (которыми являются исследуемые данные), поэтому в дальнейшем будем работать именно с ней. Попробуем подобрать оптимальные параметры и построим графики по получившимся результатам.

Построим дендрограмму и через евклидовые расстояния оценим, как будут распределяться кластеры.

Ниже представлена получившаяся дендрограмма:



Дендрограмма 1

Как можем заметить, видны два полноценных кластера (красный и синий), однако стоит обратить внимание на отдельное деление внутри красного кластера. Попробуем оценить рекомендуемое количество кластеров методом локтя. Для решения этой задачи построим график, где по оси OX будет количество кластеров, а по оси OY – условное расстояние между прямыми. Оптимальное количество кластеров будет при минимальном расстоянии между этими прямыми. Попробуем оценить по графику:

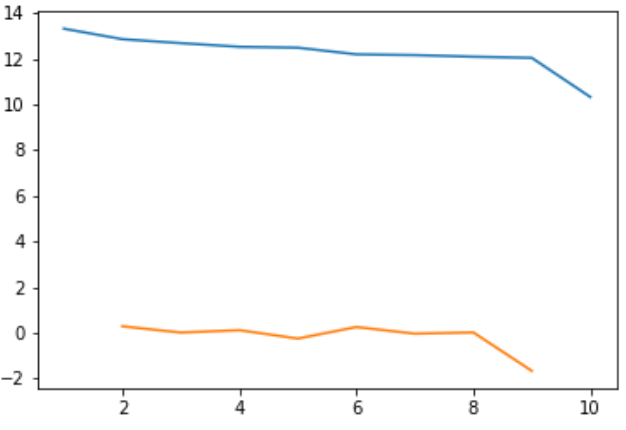
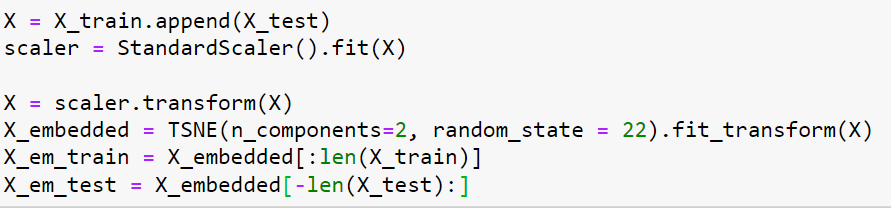


График метода локтя 1

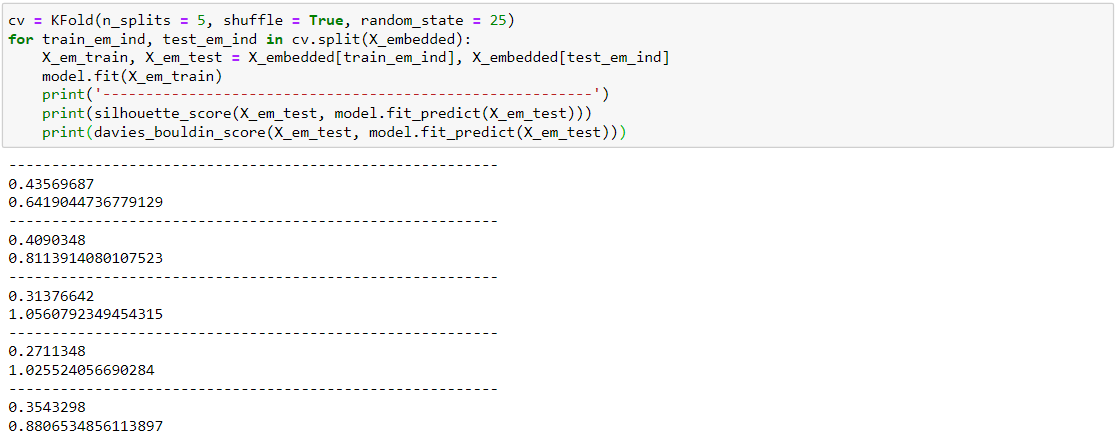
Можно отметить, что оптимальное количество кластеров – 2, однако, построив дендрограмму, можем выделить 3-ий кластер внутри красного. Для 3 кластеров сможем заметить, что расстояние увеличилось, однако виден “зубчик” вверх для 4 кластеров. В совокупности данных факторов останавливаем наш выбор на четырех кластерах.

Перед переходом к обучению итоговой модели следует провести нормализацию и понижение размерности. Эта операция используется для снижения затрат на вычисления, так как после нормализации значения исследуемых принимают значения от 0 до 1 (в определенном масштабе). Для графического представления результатов и упрощения работы алгоритма воспользуемся методом понижения размерности TSNE (стохастическое вложение соседей с t-распределением). Данный метод основывается на нелинейной технике снижения размерности, задача которой состоит в минимизации расстояния Кульбака-Лейблера между двумя распределениями с учетом положения точек.

Реализация в коде:

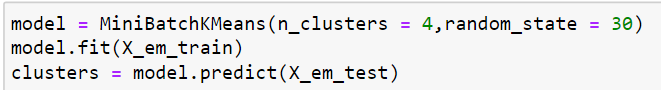


После алгоритма TSNE наша размерность понизилась с 90 до 2.

Перед обучением модели следует провести валидацию для того, чтобы избежать ситуацию, при которой исследуемые данные разбились бы неравномерно в момент разделения на обучающую и тестовую выборку. Для этого оригинальная выборка перемешивается на 5 подвыборок, затем метрики проходят проверку. Код и результат представлен ниже: 

Как итог, можно отметить, что есть небольшие колебания метрик, однако они не критичны. Следовательно, можем переходить к обучению данных.

Обучение и тест на данных:



Получаем диаграмму рассеивания (цветом обозначены кластеры):

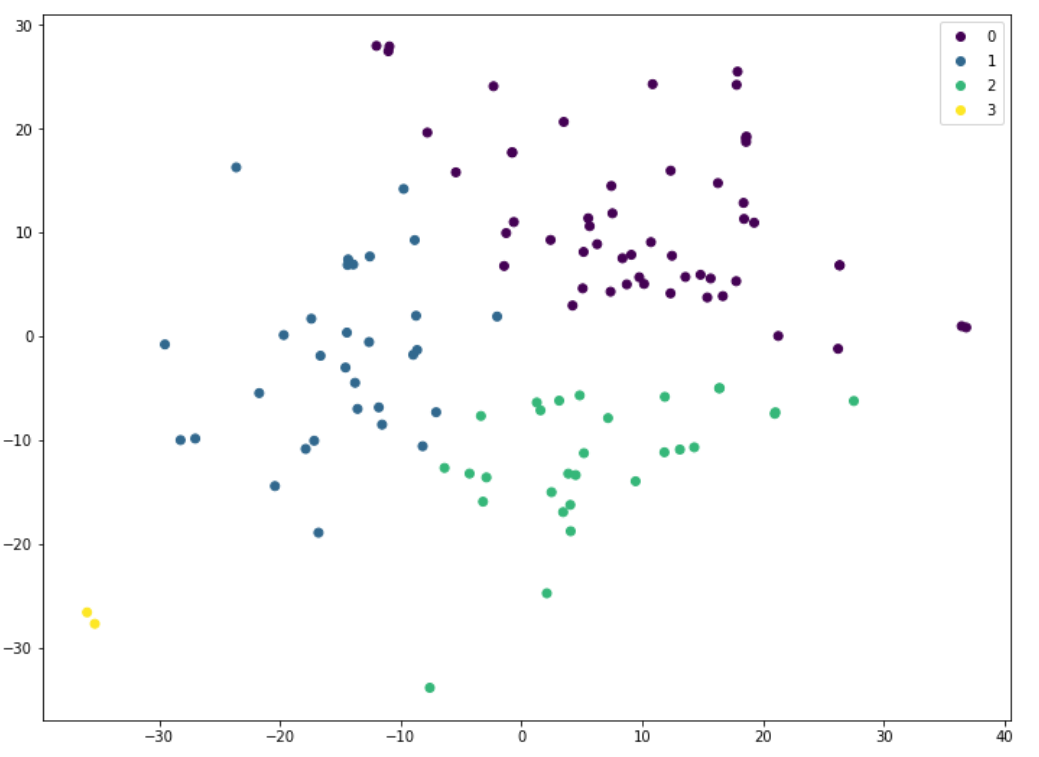
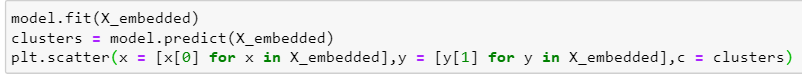


Диаграмма рассеивания кластеров 1

Метрики показали значения 0.046877 секунд для времени обучения, 0.39123 для Silhouette score и 0.8756 для Davies Bouldin score.

После построения диаграммы и проверки метрик попробуем кластеризировать весь датасет:



Итоговая диаграмма рассеивания для всего датасета (цветом обозначены кластеры):

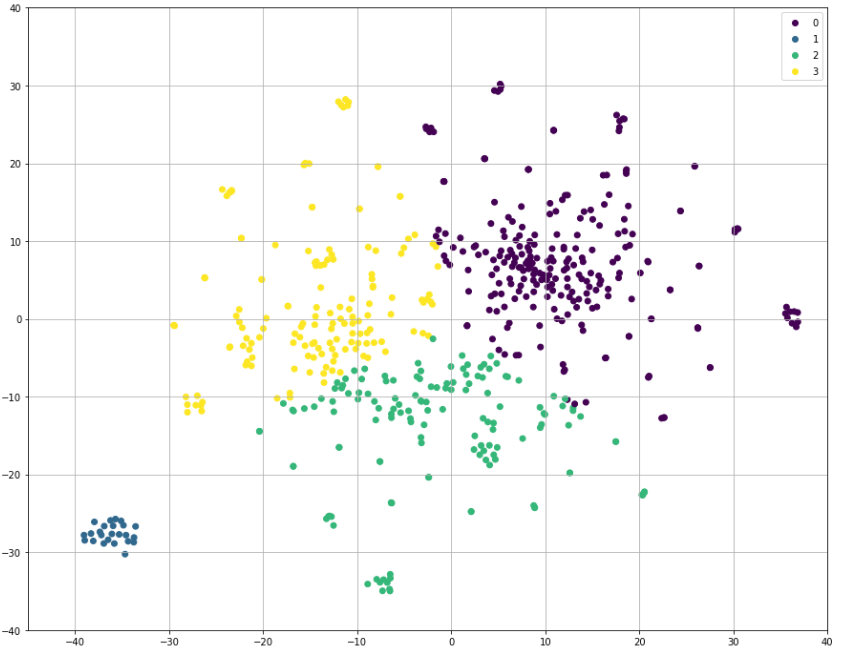


Диаграмма рассеивания кластеров 2

Оценим получившиеся метрики. Silhouette score равен 0.3912, Davies Bouldin score равен 0.87562. Можем заметить, что получившиеся показатели только уменьшились, это связано с тем, что при понижении размерности с 90 до 2 произошла минимизация расстояний, которая в итоге и снизила исследуемые метрики, а также с заполнением отсутсвтующих данных с относительной вероятностью. На графике не видно выбросов и аномалий что может означать о хорошей кластеризации, в то время как метрики указывают на наличие дополнительных оптимизаций для улучшения алгоритма.

Рассмотрим получившиеся кластеры. Рассматриваемые признаки не позволяют построить обычные закономерности и правила кластеров (оценка про среднему значению, моде), поэтому рассмотрим некоторые из них. В кластер под номером 1 попало большинство друзей из музыкальной школы. Ниже представлена таблица пользователей первого кластера.

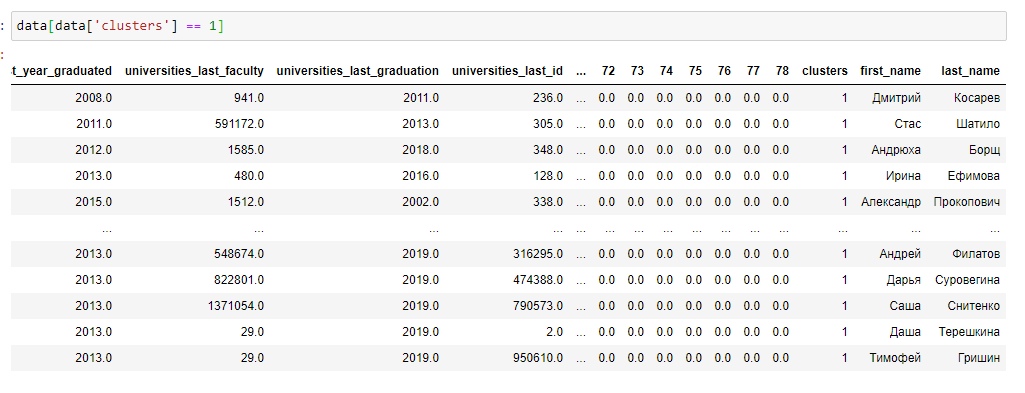


Таблица результатов 1

В кластер под номером 3 попали все друзья из школы и университета. Ниже представлена таблица пользователей третьего кластера.

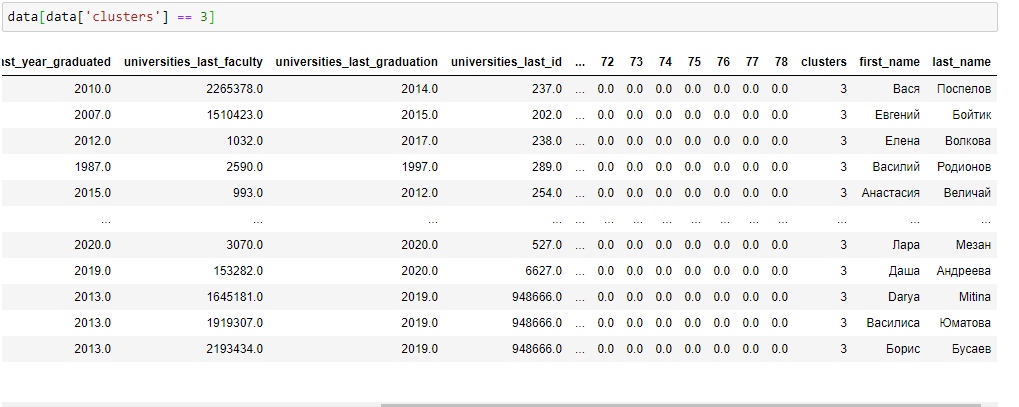


Таблица результатов 2

Во второй кластер вошли люди с которыми я познакомился случайно и общих друзей не имеется.