Выбор датасета.

В качестве датасета для анализа и построения графа связей я выбрала социальную сеть ВКонтакте и своих друзей. Я решила воспользоваться именно этой базой данных по нескольким причинам. Во-первых, частое использование сети делает анализ более показательным и интересным для обработки. Во-вторых, данные предоставлены явно и информативны, а поэтому удобны для анализа и построения графов. В-третьих, анализ социальной сети ВКонтакте открывает возможности к проверке новых данных и построению новых графов. Например, можно увидеть следующую информацию:

- Количество моих друзей ВКонтакте, их общие друзья
- Информация о самих друзьях (нодах), либо в качестве списка признаков, либо также визуализированная на самом графе
- Кластера объединенных (близко расположенных друг к другу) друзей и их значения
- Возможные значения: объединение по организации, по месту проживания, итд
- Разные другие закономерности связей между людьми
- Подтверждение теории о 5 рукопожатиях, а именно:
    - Найти популярных людей, с кем я знакома через 1, 2, 3, 4 и 5 рукопожатий
    - Определить популярность человека через количество подписчиков, лайков, количество подарков ВКонтакте

Как видно, есть множество возможностей для анализа и новых получения новых данных. Поэтому я и решила остановиться на выборе графа друзей ВКонтакте. Некоторые из предложенных выше пунктов предполагаются как дальнейшие пути развития.

Сбор данных.

Для сбора данных я написала свой код, а не брала готовый датасет. Чтобы достать данные о друзьях ВКонтакте и связях, я воспользовалась предоставленным ВК API. Использовались такие методы, как:

- friends.get()
- friends.getMutual()
- users.get()

Первым делом, я получила список своих друзей и общих друзей между ними. После чего составила матрицу смежности, отсортировав друзей по их id. Получилась следующая таблица (часть ее, pandas):

	0	1	2	3	4	5	6	7	8	9	...	207	208	209	210	211	212	213	214	215	216
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	1	1	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
212	0	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	0
213	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
214	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0
215	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	1	0	0
216	1	0	0	0	0	0	0	0	0	0	...	0	0	1	0	0	0	0	0	0	0

Далее с помощью API я достала информацию о своих друзьях и составила отдельную таблицу. А именно, были созданы следующие поля (если таковые имелись):

name	Имя
last_name	Фамилия
sex	Пол
bdate	Дата рождения
city	Город (первый указанный)
country	Страна (первая указанная)
has_mobile	Наличие мобильной версии (бинарное)
status	Статус
followers_count	Количество подписчиков (число)
nickname	Отчество
common_count	Количество общих друзей (число)
has_photo	Наличие фотографии в профиле (бинарное)
screen_name	Ник (при его отсутствие - id)

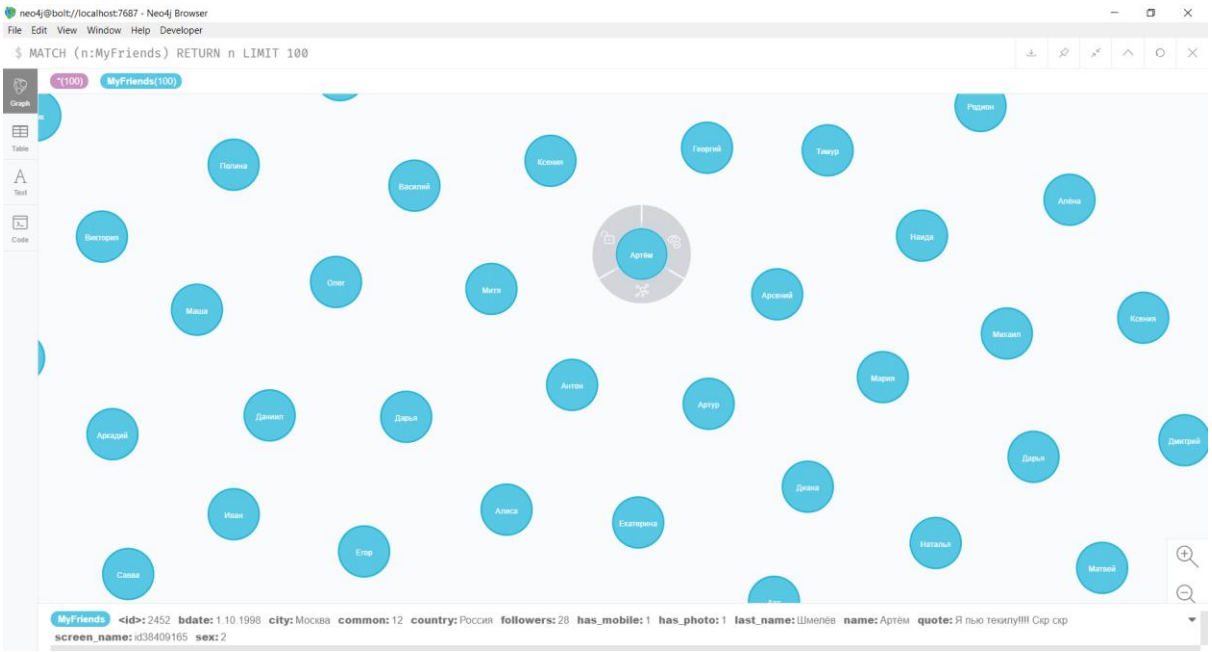
Таблица с результатами (pandas) выглядит так:

	names	last_names	sex	bdate	city	country	has_mobile	status	followes_count	nickname	common_count	has_photo	screen_name
0	Дмитрий	Степанов	2	8.5	Москва	Россия	1		78		15	1	id726499
1	Алексей	Чернов	2	8.8.1985	Москва	Россия	1		380		17	1	alexey.chernov
2	Александра	Ковтунова	1	28.11.1988	Москва	Россия	1		67		3	1	kovtounova
3	Тарас	Сулима	2	28.2	Москва	Россия	1	Once you find your center - you are sure to win	82		2	1	futility
4	Наташа	Дигас	1	11.10	Madrid	Испания	1		93		1	1	id2808168

Полученные данные я загрузила в neo4j и визуализировала. Первое, я визуализировала матрицу смежности. Но из-за большого ее масштаба решила показать только маленькую ее часть (матрица 5*5, хотя друзей гораздо больше). Полученная визуализация и ее запрос в neo4j:

```
1 LOAD CSV FROM "file:///adj_matrix.csv" as line
2 WITH line[1] as first_id, line[2] as second_id, line[3] as third_id, line[4] as
  fourth_id, line[5] as fifth_id
3 RETURN first_id, second_id, third_id, fourth_id, fifth_id LIMIT 6
```

Далее решила визуализировать таблицу с данными о друзьях. А именно - сразу построить граф. Однако, пока что без связей между нодами. Но чтобы каждая нода представляла собой полноценную информацию о человеке и о его признаках. Полученный граф (а точнее, его часть) представлен далее:



Далее я буду пробовать другие алгоритмы визуализации на моих данных и построю новые графы связей, либо дополню существующие.