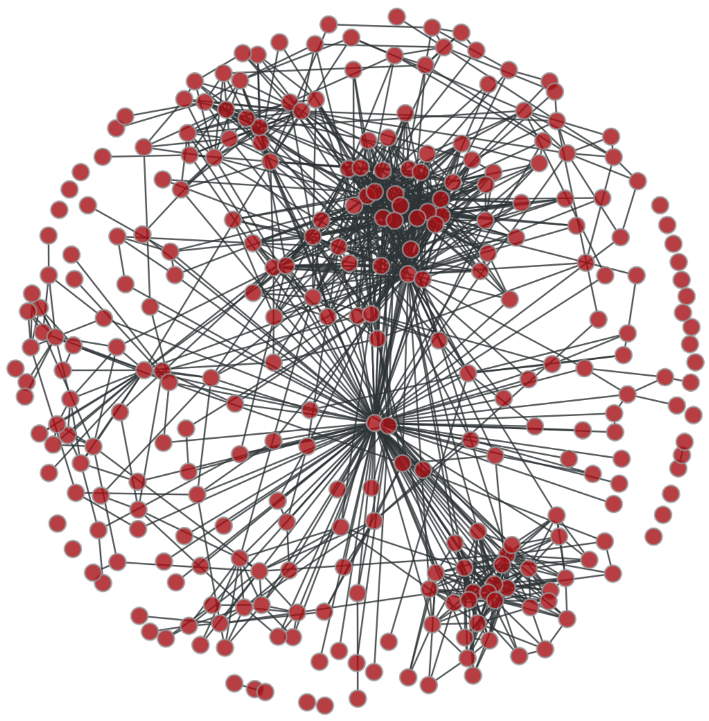
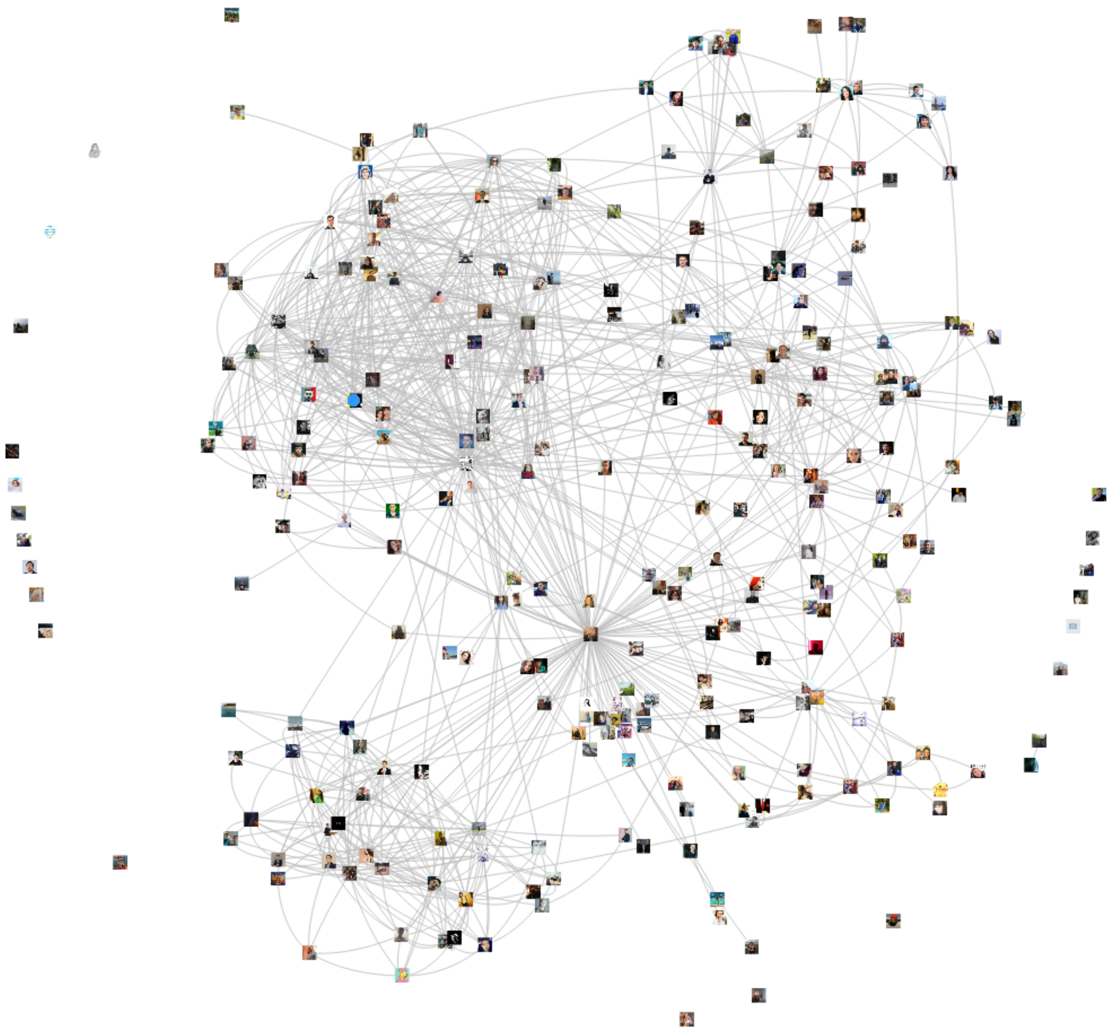
For research, Python language and the GraphTool library (working with graphs) were chosen as the main tools. Python is very comfortable language for researches and rapid development. A lot of useful tools are created for you. There are a big community, so you can find almost all you need or create something there are not exist in quick way. GraphTool is very powerful library for working with graphs.

The first task was to collect a test data-set of 10,000 users from Kazan. VK API provides a method for finding users, but you can only get the first thousand. A asked support about that issue, but they said there are no possibility to get all users from city. So, to obtain a data-set I used a search method in groups (Kazan city group), where I was able to use offset: get first thousand and check city, then get next thousand and check city and so on. After that I got a list of friends for each user. So, for this step I had all vertices I need, but there are no edges (links between users). I went through all users and their friends. When there were intersection (for example I have a friend which consists in this group) I created a link between them. In pic. 1 you can see same graph but with ~400 users for Innopolis city (visualization for 10 000 takes time, so for examples I will use data from Innopolis)



Picture 1. Graph for Innopolis city

I wanted to make graph a bit interactive, so I used d3.js to draw graph in browser (as vertices were used users' avatars). The result you can see in Pic. 2

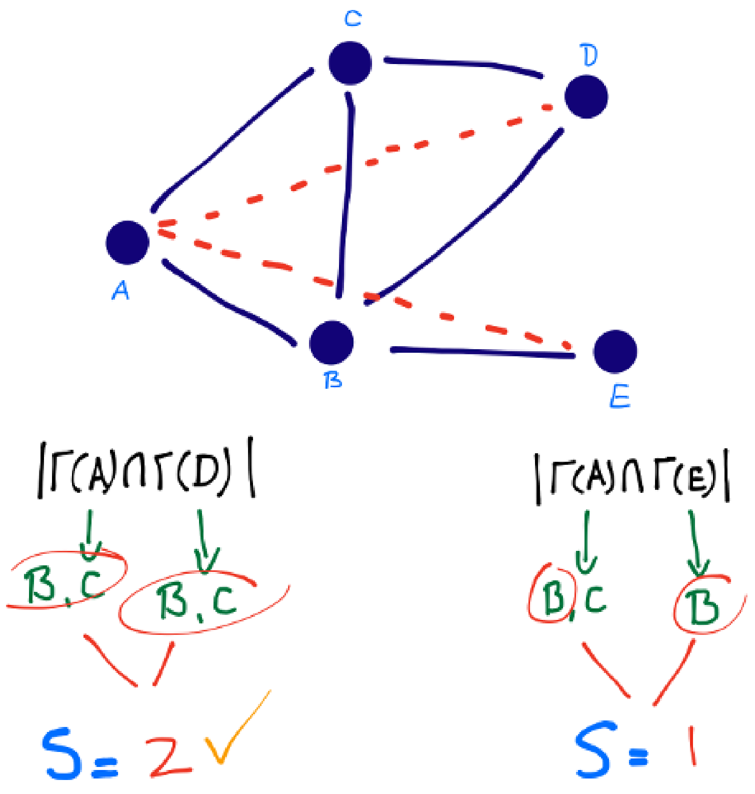


Picture 2. Interactive graph for Innopolis city

Then, 20% of the edges were randomly removed for the application of prediction methods, after which an attempt was made to predict remote edges using the Common Neighbors method and the Adamic / Adar and Jaccard’s coefficients.

Common Neighbors predictor captures the notion that two users who have a common friend may be introduced by that friend. This introduction has the effect of “closing a triangle” in the graph and feels like a common mechanism in real life. Newman has computed this quantity in the context of collaboration networks, verifying a positive correlation between the number of common neighbors of x and y at time t, and the probability that x and y will collaborate at some time after t.

$$\mathrm{Score}(x,y) = |\underbrace{\Gamma(x)}_{\text{Neighbors of } x} \cap \Gamma(y)|$$



Jaccard's Coefficient is a similarity metric that is commonly used in information retrieval— measures the probability that both x and y have a feature f, for a randomly selected feature f that either x or y has. If we take "features" here to be neighbors, then this measure captures the intuitively appealing notion that the proportion of the coauthors of x who have also worked with y (and vice versa) is a good measure of the similarity of x and y.

Common friends

$$\text{Score}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}$$

total friends

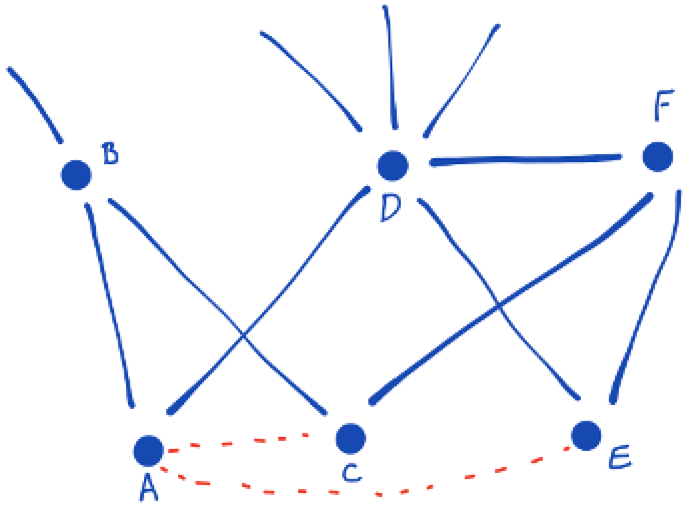
Adamic/Adar (Frequency-Weighted Common Neighbors) measure, which refines the simple counting of common features by weighting rarer features more heavily. The Adamic/Adar predictor formalizes the intuitive notion that rare features are more telling; documents that share the phrase "for example" are probably less similar than documents that share the phrase "clustering coefficient."

If “triangle closing” is a frequent mechanism by which new edges form in a social network, then for x and y to be introduced by a common friend z, person z will have to choose to introduce the pair ⟨x,y⟩ from (choose |Γ(z)| with 2) pairs of his friends; thus an unpopular person (someone with not a lot of friends) may be more likely to introduce a particular pair of his friends to each other.

$$\mathrm{Score}(x,y) = \sum_{z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{\underline{\log|\Gamma(z)|}}$$

Frequency of z

weighting rare features more heavily



$\Gamma(A)\cap\Gamma(C) = B$

$\frac{1}{\log(\Gamma(B))} = \frac{1}{\log 3} = \underline{\underline{2.09}}$

$\Gamma(A)\cap\Gamma(E) = D$

$\frac{1}{\log(\Gamma(D))} = \frac{1}{\log 6} = 1.2$

The result was low - the best predictor with the Adamic / Adar coefficient was 11.9% of the successfully predicted edges. You can see all results in Table 1.

Common Neighbors	**Jaccard’s Coefficient**	**Adamic / Adar**
9.813739234928901	1.722411375926297	10.674944922892049
10.294412177047867	1.0414580412577608	10.815141197676747
9.893851391948727	1.70238333667134	10.614860805127178
10.955337472461446	1.3819347085920288	11.896655317444422
10.27438413779291	1.1415982375325455	10.715001001401962

Table 1. Predictor results

The question arises - why in the VK network such a low level of predictions? A small study was conducted. VK, unlike Facebook, for example, a lot of promotional accounts. You can determine them by several criteria: profile, number of friends, amount of media data, activity. But in order to verify the data of each user you need to make at least 5 requests (for 1 million users - 5 million requests). VK API allows you to do no more than 3 requests per second, when using the execute method - 75 requests per second. There is a big problem of getting data.

For further work, the program code has been optimized. Also work has begun on the design of modules for independent use.

To quickly get users across cities, it was decided to build a local database with all the active users of the network. For this purpose, a user catalog and a parser were used. Next, it is necessary for each user to obtain a city identifier, where there is a problem with the restriction of the VK network.