Contents

Objective

We are gone inspect some data on startups and decide which features are most important to build a successful company.

Data

We are given data with 472 data entries, where each row represents some info about Startup (Investors, founding date, etc.), it's Co-Founders (Education, involvement in startups earlier, etc.). The overall number of features (excluded company name and target column) is 114. The target column is "Dependent-Company Status", which represents if the start-up is succeeded or not.

Cleaning Data

We are given data with such data types (Fig 2).

First, we change datatypes from categorical to numerical as much as we can, to work better with different models. After some lines of code, which are well described in jupyter notebook we get result like this (Fig 1).

After data type conversion, we handle NaN data. In given dataset portion of NaN values for every column is given in (Fig 3)

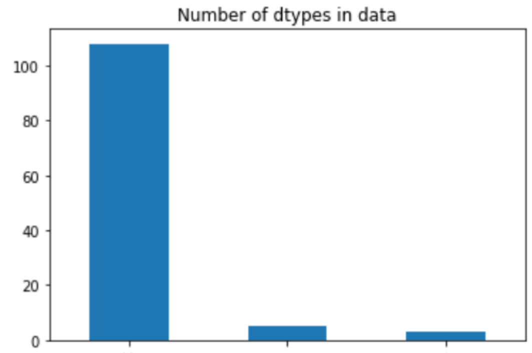


Fig 2 Quantity of data types before change

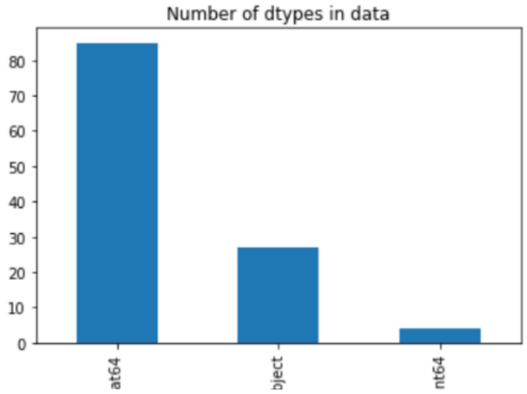


Fig 1 Quantity of data types after changes

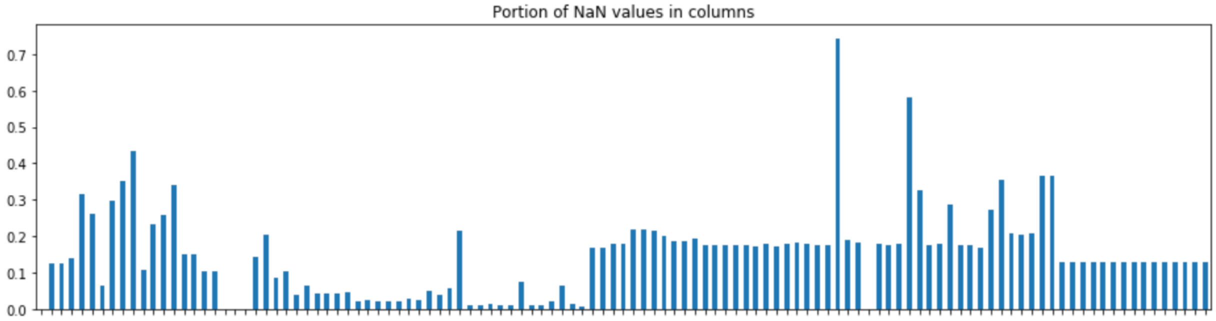


Fig 3 Distribution of NaN values in data columns

For numerical features we fill NaN values with mean values of the column, rounded to near integer (for discrete values). For categorical features we fill NaN's with close valid values (front and back fill), we assume, that for example some investors could also invest in those companies where we don't know the name of an investor, and it doesn't affect our feature importance.

Feature Selection

First, we inspect our features for mutual correlations, it 2 features has strong correlation that does mean that one of them doesn't contribute much prediction, because this 2 are collinear vectors, and dropping one of them wouldn't affect the prediction. For correlation threshold we select **0.8,** so if correlation value between two different features are higher than this threshold, we will drop one of them, which has less correlation with target column. There are 2 pairs like this: "Team size all employees" and "Have been part of startups in the past?", so we will drop them.

We will continue to building our sample model, but before to work better with categorical features, we do one hot encoding. This part increases number of out features, but we don't mind, because after feeding the data to our model it will give some of them fewer weights, so we can ignore them.

Model Training

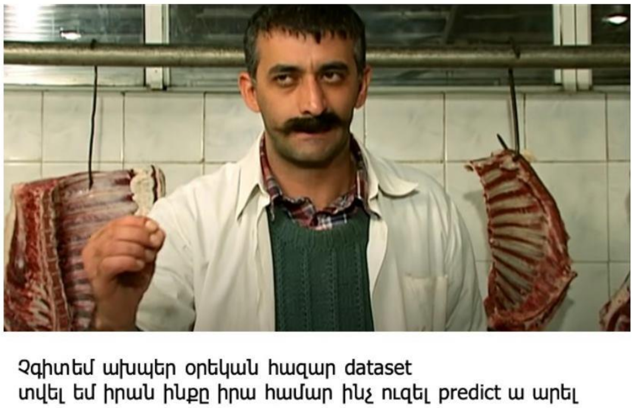
We divide our dataset to 2 parts: training and testing, with 0.7/0.3 portions.

The model I selected for training on these data is Gradient Boosting Decision Trees.

I choose GBDT, because they are fast, well interpreted, and work good with large number of features, and don’t tend to overfit if we don’t set number of trees higher than our data points.

To avoid overfitting, and for interpretation of model, we do Cross Validation, we implement Grid Search on GBDTs to also find the best parameters of trees. For given features we the best parameters of GBDTs are learning_rate = 0.4, n_estimators = 30

Score on test set we obtain as in (Table 1).


Չգիտեմ ախպեր օրեկան հազար dataset
տվել եմ իրան ինքը իրա համար ինչ ուզել predict ա արել

If you ask me how my model works

Table 1 Metric scores of GBDTs

Metric	Score
R^2	0.6281
MSE	0.07746
Accuracy	0.9225
Precision	0.9406
Recall	0.95
F-1 Score	0.9453

Our model returned good metrics on test set, so we can assume that we can select best features from this model.

This is done by *feature_importance_* method of our classifier, it returns scores for each feature, higher - better. At this point we have 1751 features, but model gives 0 weights 1663 of those, the model assumes that they are not related features, so we will do the same.

Then we plot our 1751 feature to see how score changes (Fig 4).

We set threshold of best features: it’s the 0.9 quartile of score values, it means that in score values 10% of scores which are higher than 0.9 quartile do matter. The features which scores are below red line in (Fig 4), we won’t take into account.

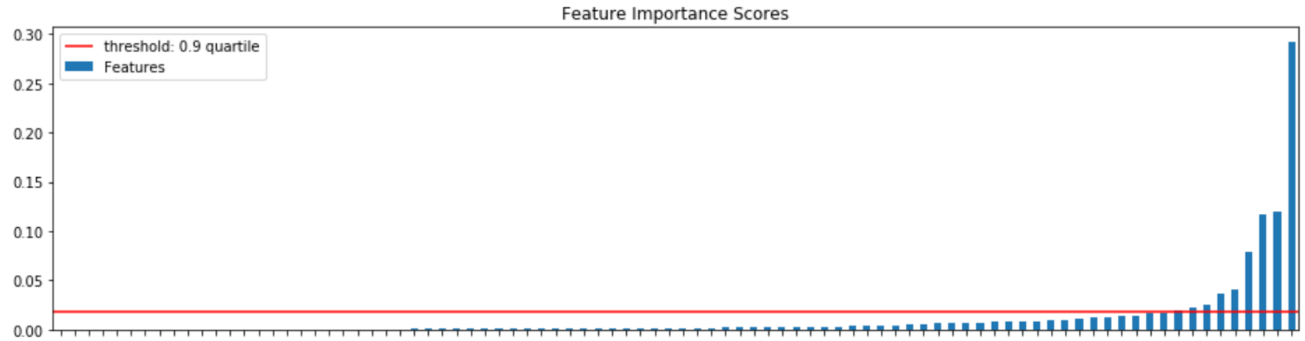


Fig 4 Feature importance score comparison for each column

In (Fig 5) we can see those most important features with their corresponding scores.

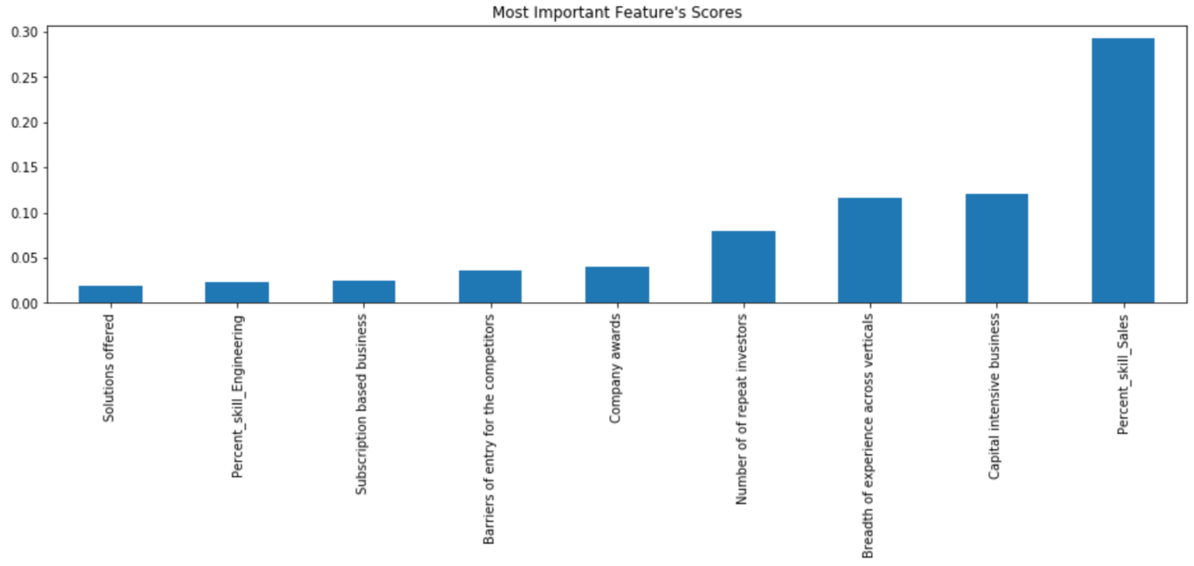


Fig 5 Top 9 Features with their scores

So, there are 9 most important features according to our model, which decide the future of the start-up. Feature names and their scores are represented also in (Table 2).

Table 2 Top 9 Features with their scores

Feature	Score
Percent_skill_Sales	0.292505
Capital intensive business	0.120403
Breadth of experience across verticals	0.116438
Number of of repeat investors	0.079109
Company awards	0.040122
Barriers of entry for the competitors	0.036378
Subscription based business	0.024763
Percent_skill_Engineering	0.022555
Solutions offered	0.019429

To check our assumptions, we build a new model with Random Forest Classifier, with parameters *max_depth = 4, min_samples_leaf = 4, n_estimators = 50*, which gave grid search on random forest. We get scores in (Table 3).

Table 3 Metric Scores on Random Forest with most important features

Metric	Score
R^2	-0.0481
MSE	0.2183
Accuracy	0.7817
Precision	0.8
Recall	0.92
F-1 Score	0.8558

Those are worst then in our first model, but it was predictable, because we make prediction with fewer features, but we assume that accuracy **0.78** and F-1 score **0.86** are not bad for this kind of small dataset.

Conclusion

We inspected data with different kind of information about startups, our main goal was to inspect which features mostly influence startups to succeed. Using correlation matrix and feature selection from Gradient Boosting Decision Trees, where we set threshold of importance 0.9 quartile, we achieved our main goal of this research. These features are posted in (Table 2). Speaking about some of the features, it was predictable that "Solutions offered" will find his place in top 10, everyone knows that if start-ups solve a difficult problem, they become valuable for most of the investors, because if start-up solves a problem for a business, they get paid more by those businesses, and thus create more value, which returns investment to main investors, and they repeat their investment ("Number of of repeat investors").

"Company awards" are point in that direction, that problem solved by the start-up is valuable, that even people will give prizes for that. If solution is valuable to people who give prizes, it also has high chances to be valuable for many more companies and people, so that means company succeeded.

“Percent_skill_Sales” and “Percent_skill_Engineering” shows that co-founders with engineering backgrounds can solve difficult problems, which is valuable, but if they couldn’t sale their solution or find people who can, they will probably fail. The most iconic duo is Steve Wozniak and Steve Jobs, engineering skills of Woz, was higher than of Jobs’, but Steve Jobs could sell computers like no one else.

We can talk about features not only from top 9, but also from worst 9, but our goal was to inspect features which affect to successes **or** failures, so failure part is behind of the scope of the report.

Acknowledges

I would like to express my special thanks to **metric** team, who gave me opportunity to work on such interesting data, and also for extending my deadline.