Eugenio Murillo

Final Report
Startup Acquisition Prediction

Problem Statement

The startup environment is considered one of the riskiest investments due to 90% of startups failing in the first year. This 90% can be reduced throughout the life of a startup depending on several factors and if the founders were able to create relationships and get funding to finance the startup operations.The United States of America has the largest startup ecosystem in the world meaning they offer the greatest opportunities for people who want to be successful in the startup environment. The purpose of this project is to be able to predict if a startup is going to be acquired or fail based on several features. I also want to see which are the features that have the most weight on predicting a startup fail/success. This can lead to investors to use the model and decide whether to invest or not being one step ahead of the competitor investors.

Data collection/wrangling and Definitions

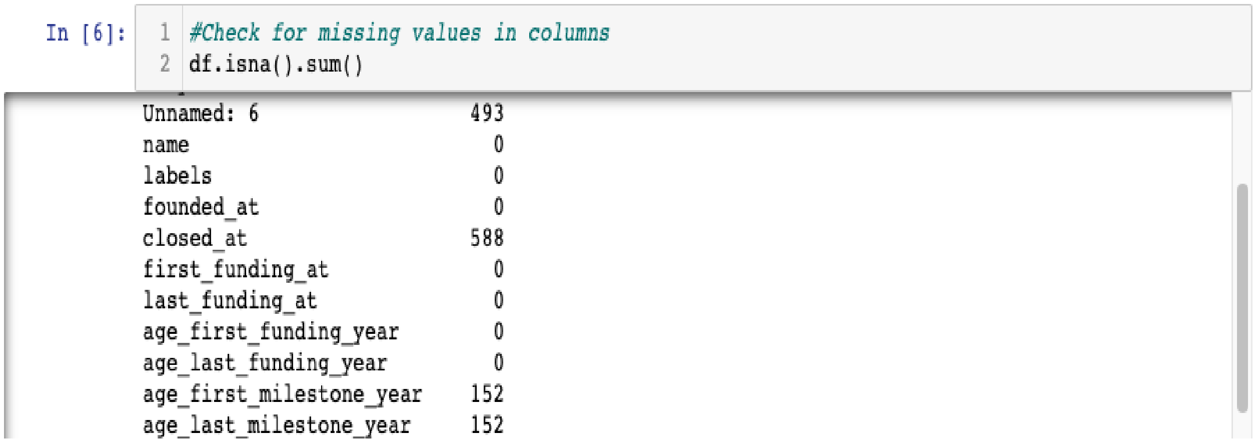
- The dataset was downloaded from Kaggle and is courtesy of Ramkishan Panthena who is a Machine Learning Engineer in GMO.
- The dataset contains the following principal features (excluding one-hot encoding created features) with their following type.
    - age_first_funding_year – quantitative
    - age_last_funding_year – quantitative
    - relationships – quantitative
    - funding_rounds – quantitative
    - funding_total_usd – quantitative
    - milestones – quantitative
    - age_first_milestone_year – quantitative
    - age_last_milestone_year – quantitative
    - state – categorical
    - industry_type – categorical
    - has_VC – categorical
    - has_angel – categorical
    - has_roundA – categorical
    - has_roundB – categorical
    - has_roundC – categorical
    - has_roundD – categorical
    - avg_participants – quantitative

- is_top500 – categorical
- status(acquired/closed) – categorical (the target variable, if a startup is 'acquired' by some other organization, means the startup succeed)

Taking a look at the data I found out there were 923 rows and 49 columns. Which will be referred to as rows -> observations, columns -> features. The dataframe has a semi-clean state, it does not require much cleaning due to dummy variables already created and column types being mostly correct. Still there are some columns which will be manipulated and dropped later while analyzing deeper.

While looking at missing values I observed missing values in columns:

- 'Unnamed: 6',
- 'Closed_at',
- 'age_first_milestone_year',
- 'age_last_milestone_year'



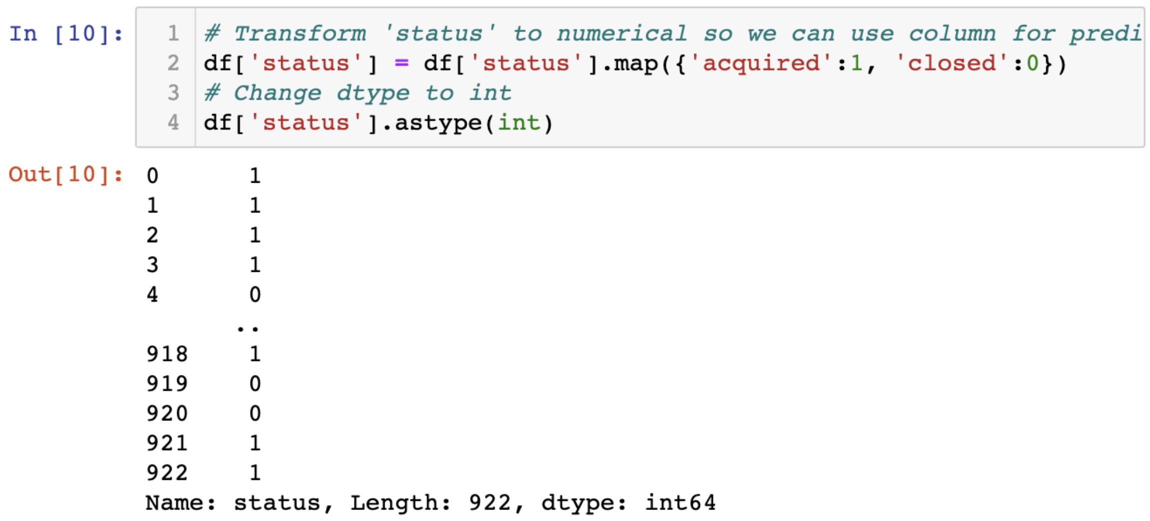
```
In [6]:
#Check for missing values in columns
df.isna().sum()
```

```
Unnamed: 6                 493
name                         0
labels                       0
founded_at                   0
closed_at                  588
first_funding_at             0
last_funding_at              0
age_first_funding_year       0
age_last_funding_year        0
age_first_milestone_year   152
age_last_milestone_year    152
```

I then check for nulls or negative values in the 'age_first_milestone_year' and 'age_last_milestone_year, due to them being columns representing date in integer they shouldnt have negative values, I proceed to replace nulls and negative values with the column mean value to be able to use all the rows in the df. I dropped the 'Unnamed: 6' column and 'Closed_at' due to them being insignificant for the analysis.

Then I checked for duplicated data in the 'name' column due to this column being a unique identifier of each row. Every single 'name' should be different. In this case I have a duplicate row which I proceeded to drop.

I performed a change of values to the 'status' column so I can use it on the modeling part, changing values from 'acquired' to 1 and 'closed' to 0. I drop the columns with the missing values and also the columns that are not useful for analysis and exploration.



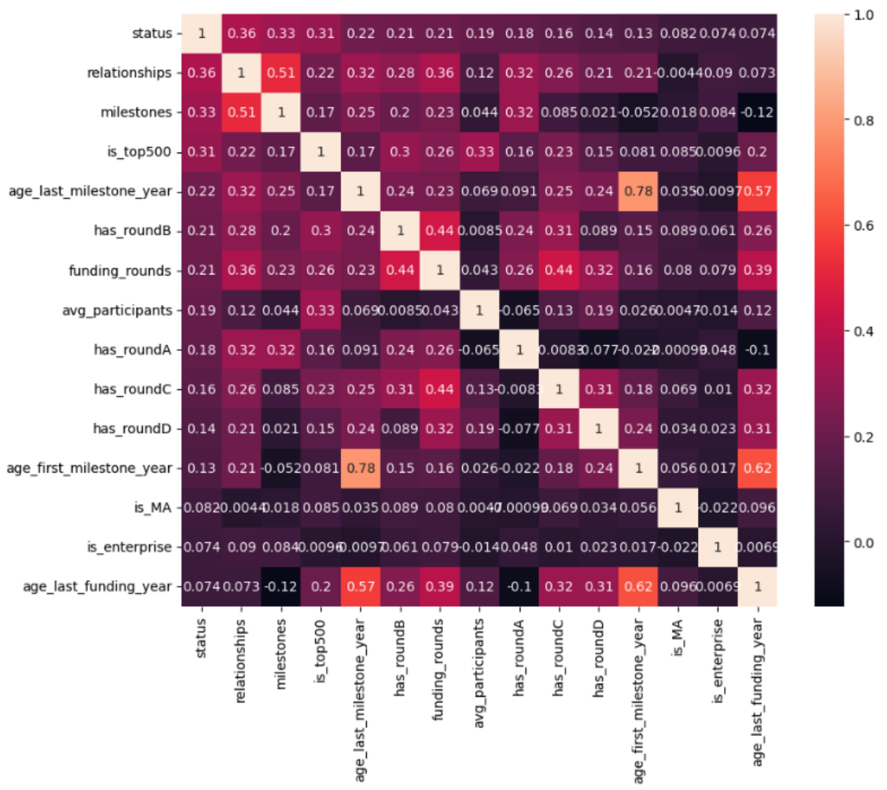
```
In [10]:  # Transform 'status' to numerical so we can use column for predi
          df['status'] = df['status'].map({'acquired':1, 'closed':0})
          # Change dtype to int
          df['status'].astype(int)

Out[10]:  0      1
          1      1
          2      1
          3      1
          4      0
                ..
          918    1
          919    0
          920    0
          921    1
          922    1
          Name: status, Length: 922, dtype: int64
```

For our last cleaning step we fix the 'state_code.1' row which has a null value, we realize this row belongs to CA so we add it to the 'state_code.1' column and fix the last null value in the df. We then proceed with a clean df ready for exploration.

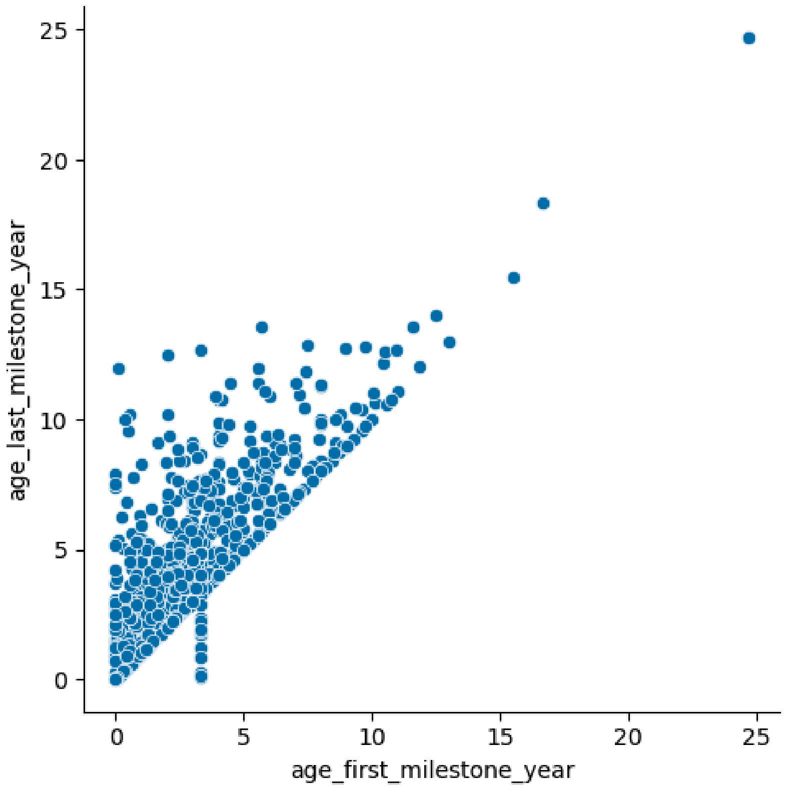
After all these steps I am finished with the data wrangling step so I proceed to save my new cleaned df which has a shape of (922,45).

Exploratory Data Analysis

I plot all of the numerical features correlated vs the ‘status’ feature which will be the predicted feature. We can see that the most correlated features against ‘status’ are ‘relationships’, ‘milestones’ and ‘is_top500’.



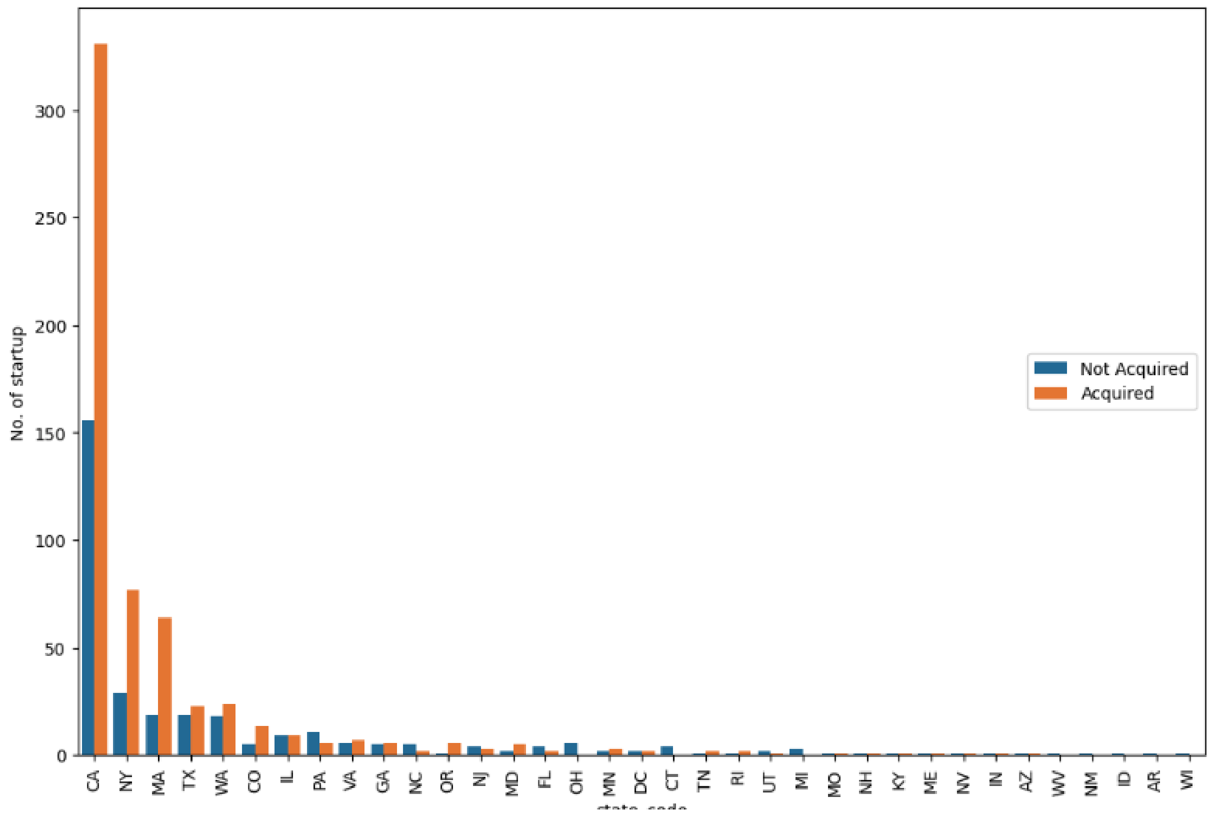
I also spot a strong correlation between 'age_first_milestone_year' 'and age_last_milestone_year' so I deepdive into it.



We can see this relationship is due to some startups having the same 'age_first_milestone_year' and 'and age_last_milestone_year', which means they only ever reached 1 milestone. This might not be important for our analysis, however I will decide in the modeling section which features to drop.

I then take a look at the number of startups within each state_code so we can see which state is the most influential in the startup environment.

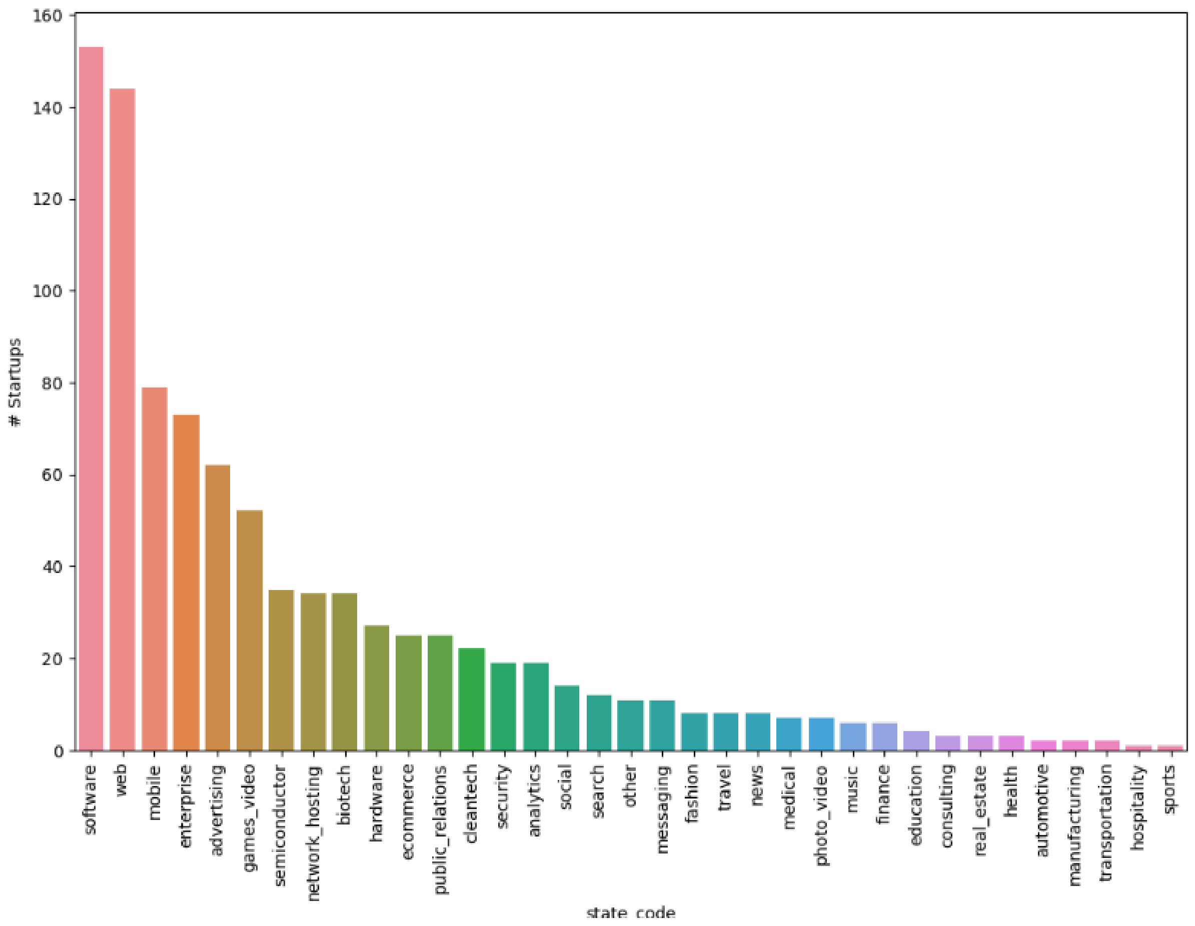
Eugenio Murillo



state_code	Acquired	Not Acquired	total_count	success_rate
CA	331	156	487	67.97%
NY	77	29	106	72.64%
MA	64	19	83	77.11%
WA	24	18	42	57.14%
CO	14	5	19	73.68%
OR	6	1	7	85.71%
MD	5	2	7	71.43%
MN	3	2	5	60.00%
RI	2	1	3	66.67%
TN	2	1	3	66.67%

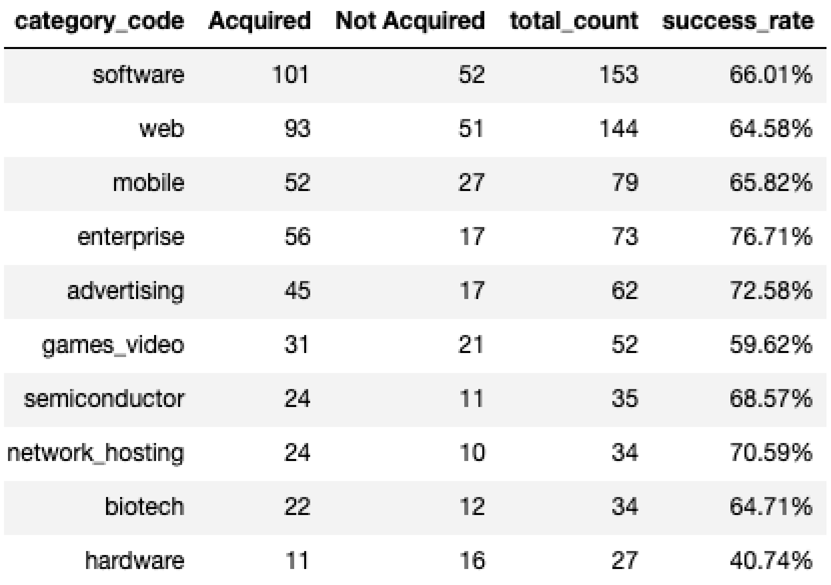
We can see the detail of the states and the total # of startups Acquired or Not Acquired. We create a summary statistic named 'success_rate' which determines # startups successful in each state. The 3 top states with the most count of startups are 'CA', 'NY', and 'MA'. Even though there are startups with higher success rate their count of startups is just a small fraction compared to the top, which is why we focus on the count first.

I then proceed to do a similar analysis with the industries instead of state_code, to see which are the industries with most startups.



We can see that the industries with the most startups belong to software, web, mobile, enterprise. This gives us insight about the technology industry being the most important when it comes to startups. Let's see the success rate based on industry.

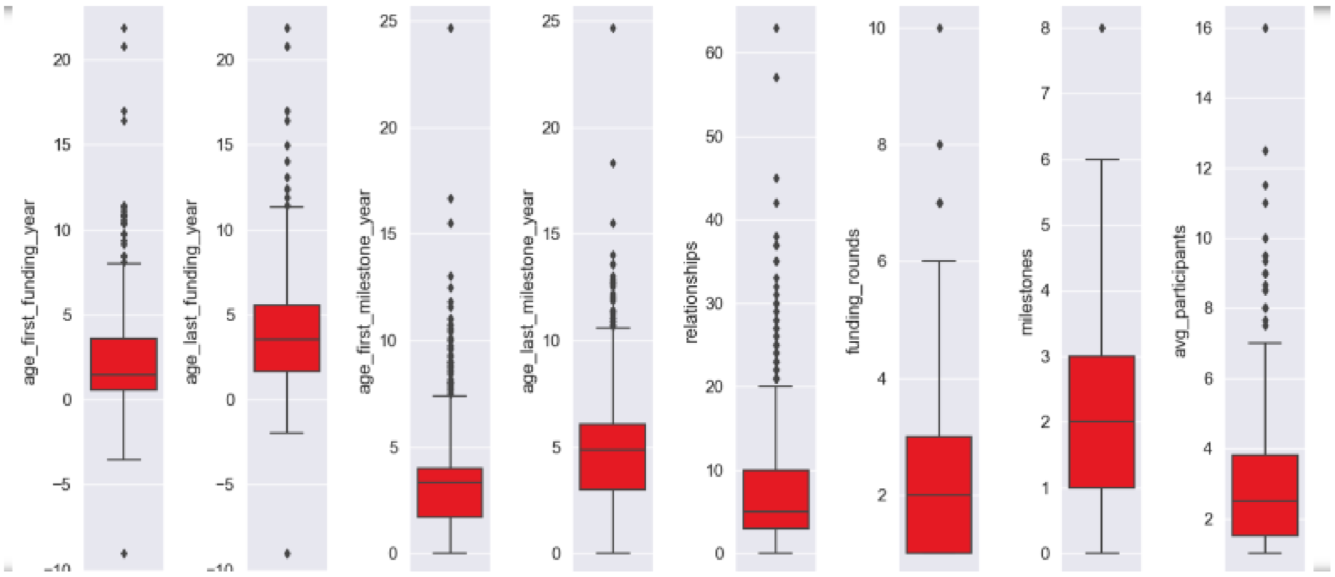
Eugenio Murillo



category_code	Acquired	Not Acquired	total_count	success_rate
software	101	52	153	66.01%
web	93	51	144	64.58%
mobile	52	27	79	65.82%
enterprise	56	17	73	76.71%
advertising	45	17	62	72.58%
games_video	31	21	52	59.62%
semiconductor	24	11	35	68.57%
network_hosting	24	10	34	70.59%
biotech	22	12	34	64.71%
hardware	11	16	27	40.74%

We add the 'success_rate' statistic which represents the % of Acquired startups, we can see positive numbers for most of the tech industry, with advertising having the highest 'success_rate'. software, web and mobile have the highest count of startups with their success_rate being really similar. This tells us that tech companies have a high success rate and also a high count, making it a competitive market but with opportunity.

As the last step of EDA I create some boxplots of the numerical features.



I created this boxplot so we can visualize some summary statistics and outliers of each of the numerical variables. We see that the median year for receiving the first funding year is 1, but still there are a lot of outliers which have gotten their first funding year in 5+ years. I also see that it is not likely for a startup to get more than 6 funding rounds. 99% of milestones reached by startups lie between 0-6.

Pre-processing and Training Data Development

I create a new df called 'df1', which will be of use in the modeling part specifically in models which need scaling, then I take a look at the object type features in our df to separate them from numerical features and proceed to do the train/test split with the numeric features. We already have the object features created as dummies and for that reason we don't need to do one-hot encoding.

Training split has 737 rows
and the corresponding labels have an equal number of values. (737)
Test split has 185 rows
and the corresponding labels have an equal number of values. (185)

I scale the data using our split data and then replace the scaled data with the original data to create our new scaled df. The main reason for scaling data comes in the modeling part, in which I will use some models that require or perform better with scaled data.

I will use the scaled data for Logistic Regression, KNN, SVM. The reason for the use in these models is due to them being distance scaled models.

The original data will be used for the remaining models including Random Forest and Gradient Boosting due to them being models that don't depend on distance meaning observations don't need to be comparable.

The performance of the models will be evaluated using 'classification_report', 'confusion_matrix' and 'accuracy_score'. Given that the classification_report gives us several metrics to analyze for the purpose of this project we will focus on the f1-score given that this metric is a more general one. This metric will be the one to help us choose our best performing model.

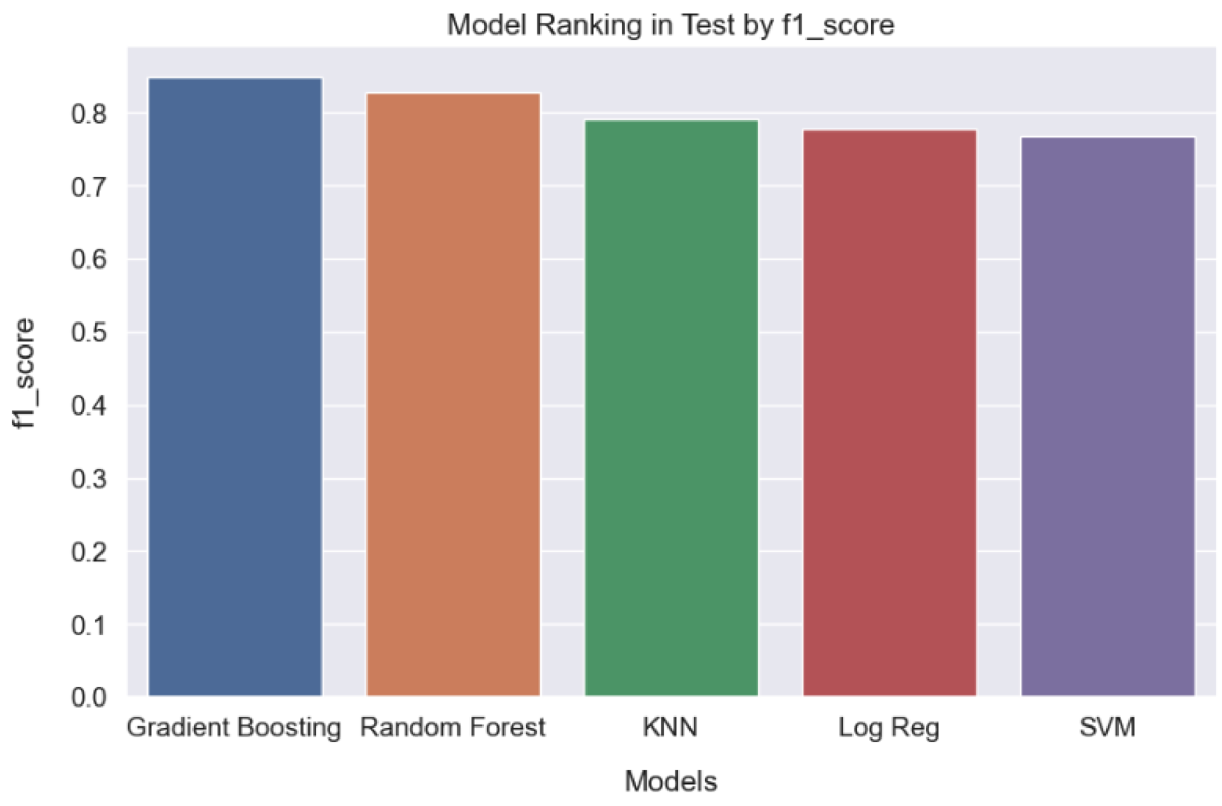
Modeling

After applying all the different models to the data I then proceed to check for the mdoel with the top f1_score and visualize the results. The best performing model will be the one recommended for deployment for predicting if a startup will be acquired or not be acquired.

The models used were:

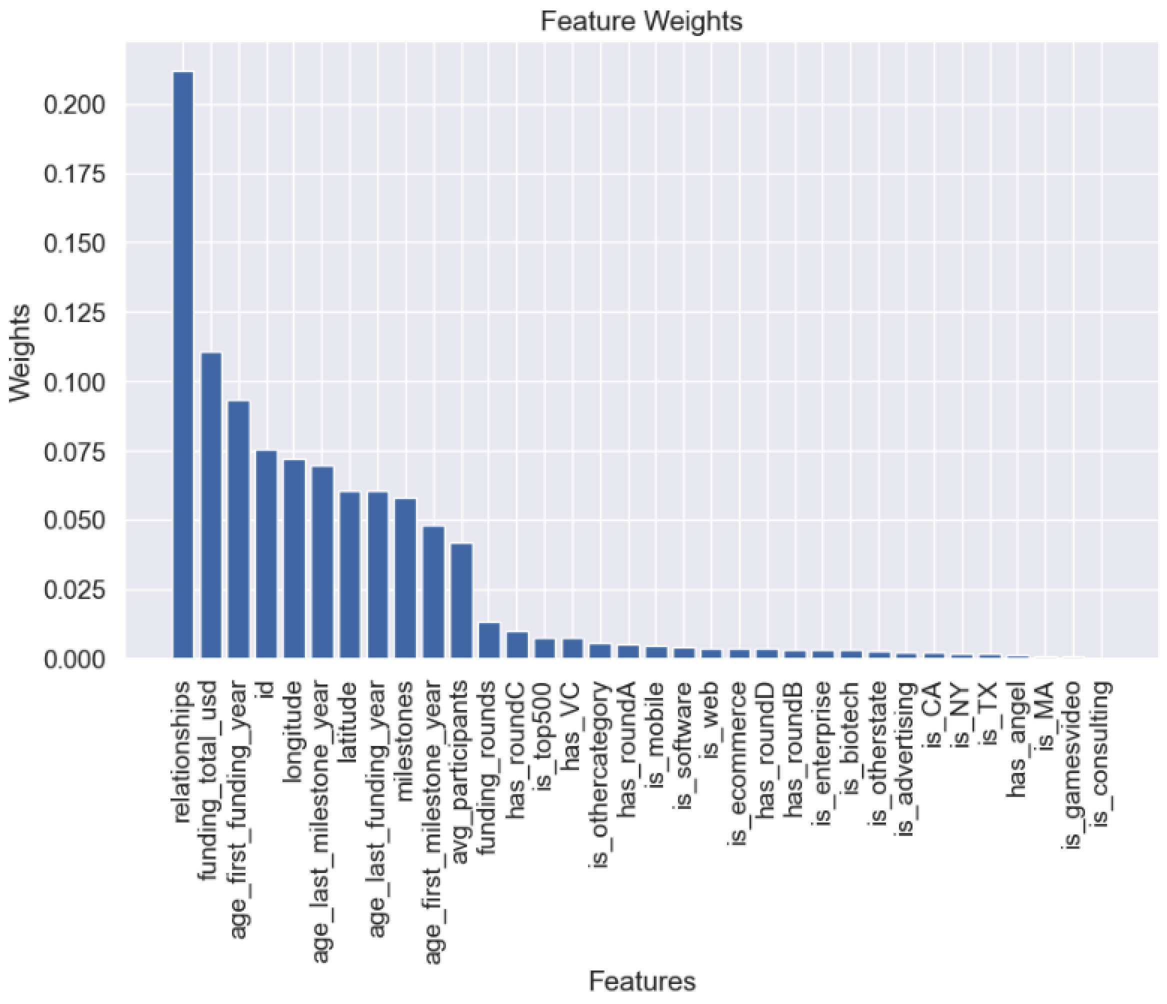
- Logistic Regression
- KNN Classifier
- SVM Classifier
- Random Forest
- Gradient Boosting

And as mentioned before the f1_score was the decisive metric.

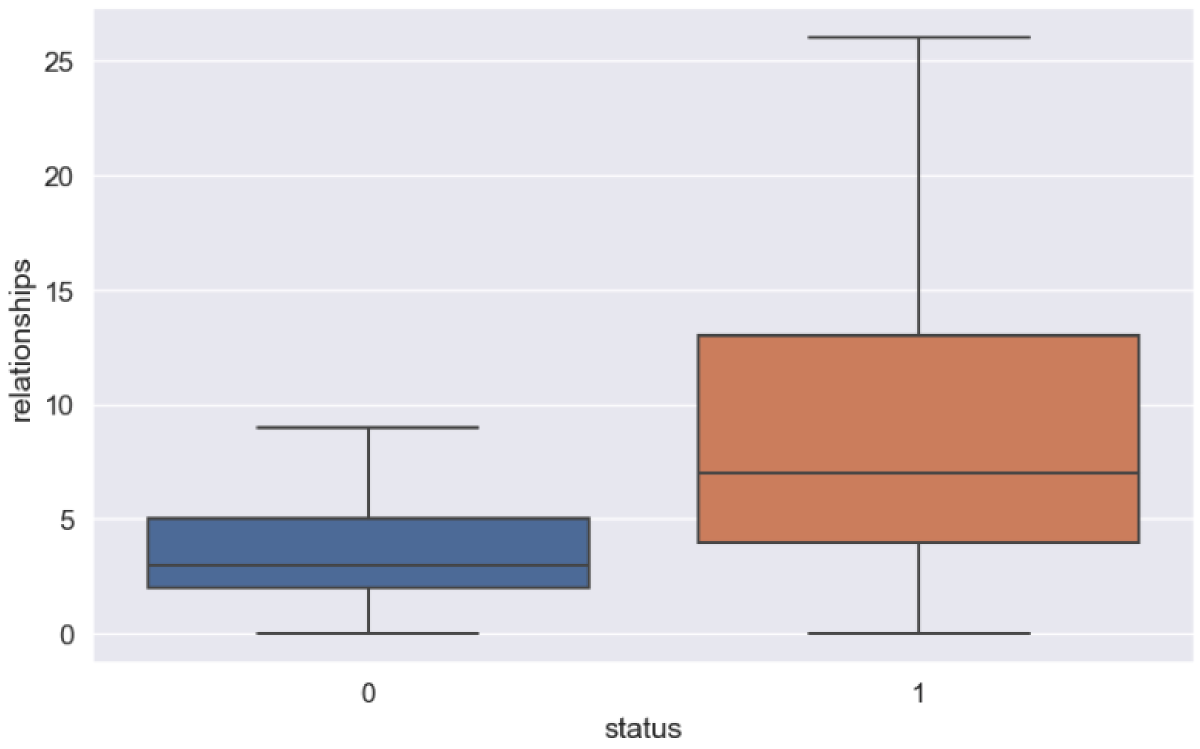


Model	f1_score
Gradient Boosting	0.849206
Random Forest	0.827309
KNN	0.791667
Log Reg	0.778243
SVM	0.767932

Our best performing model is Gradient Boosting with an f1_score of .849, which is significantly higher than the other models. This is the recommended model for deployment and for using if we want to predict if a startup will be acquired or not be acquired. I then proceed to do a feature weight analysis to visualize our top features with more predictive weight.

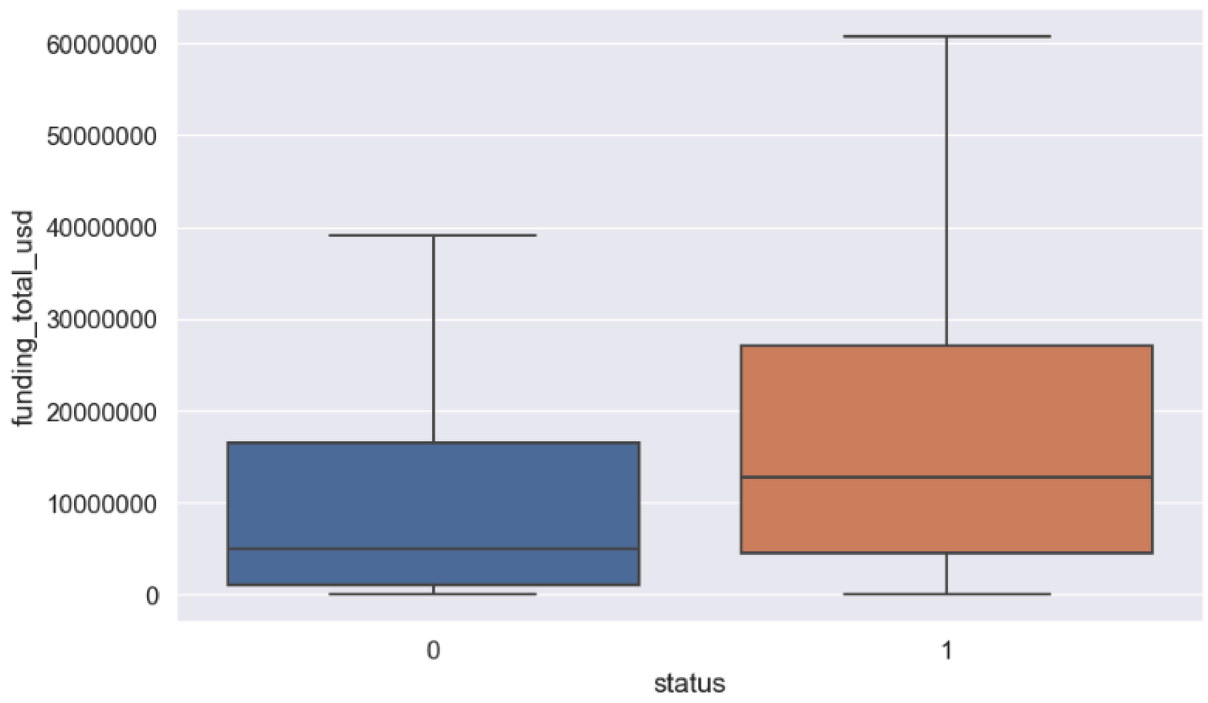


It is clearly visible which are the top features with more predicting weight, the top 3 are 'relationships', 'funding_total_usd' and 'age_first_funding_year'. These features alone represent around 42% of the predictive weight of the modeling, for this reason we will dive deeper into these features.
We create boxplots of the top 3 features with the most weight, we do this by segmenting by groups '0' and '1' which represent 'Not Acquired' and 'Acquired'. We remove outliers to be able to generalize in the interpretation of the data due to outliers being a great disturbance in the data for these specific features.

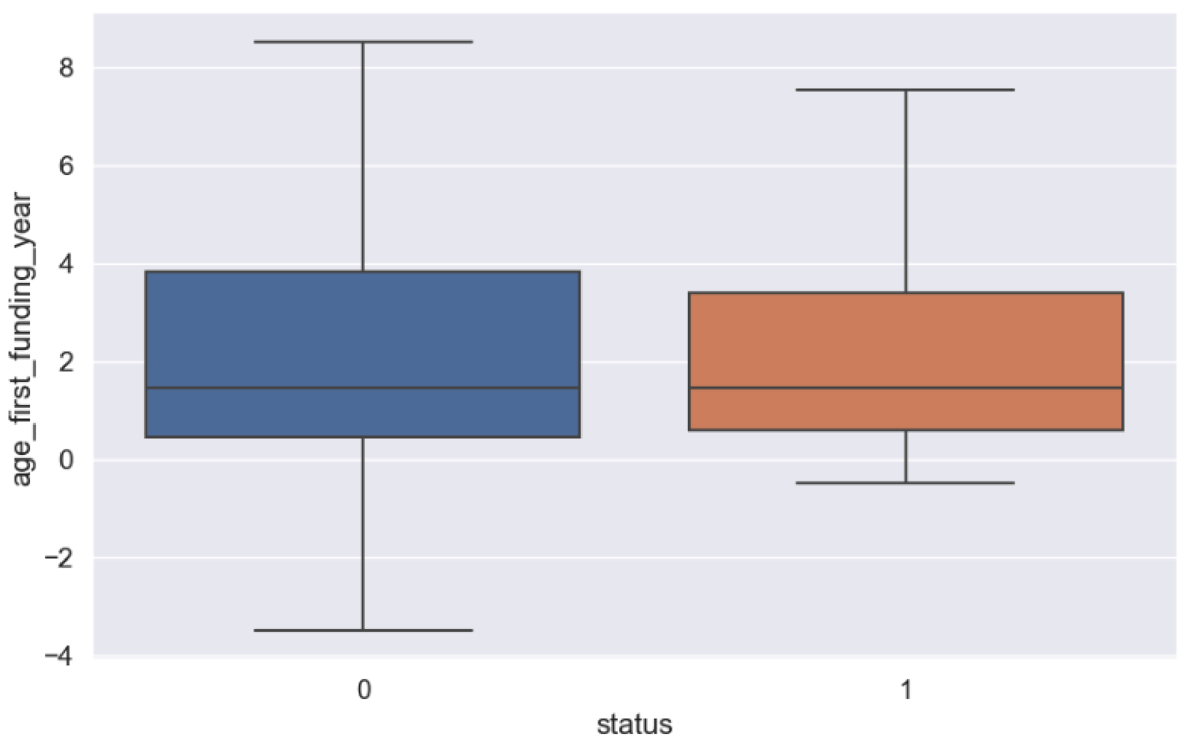


With the boxplot of feature relationship we can observe that Non Acquired Startups have a median of 3 relationships while their IQR range of 25% - 75% goes from 3 - 5 relationships. Acquired Startups have a median of 7 relationships and their IQR range goes from 4 - 13 relationships. Given that the 'relationships' feature is the feature with more importance, we can conclude that having more than 4 relationships is essential to being successful in the startup environment.

Eugenio Murillo



With the boxplot of 'funding_total_usd' we can see a great difference in the amount of funding needed to be successful in the startup environment. The Non Acquired startups have a median of 4,000,000 usd on 'total_funding_usd' while the Acquired startups have a median of 15,000,000 'funding_total_usd'. We can also observe great difference in the 25% - 75% IQR of the boxplots, it is more probable for a startup to be acquired if it has more than 6,000,000 usd total funding.



In the boxplot of 'age_first_funding_year' we can see that most startups receive funding within 1 year of being created. This behavior is independent of groups Not Acquired and Acquired. The Non Acquired group tends to be slower in receiving their first funding with a range of 1 - 4 years. In conclusion, receiving funding in an early stage has a positive impact determining success in the startup environment.

Takeaways

After doing all the analysis we get a model with strong predictive power, our Gradient Boosting model can help an investor or someone analyzing the startup environment to be more prepared and a step further from the competition. This model can be used to accurately decide if a startup will be acquired or not.