INTERMEDIATE STATUS REPORT

TEAM MEMBERS: Harshitha Mohanraj Radhika, Fnu Maria poulose, Samiksha Talwekar

1. Progress towards the goal achieved so far

Data Collection:

Our project's objective is to classify whether the startups are successful or not. This model will help to measure the performance of the startup and help the investors like venture capitalists to whether invest in such companies or not. We collected the data from the secondary source kaggle. The dataset has 923 rows and 49 columns and the data are labeled. The dataset is imported into the python notebook in google collab and further explored and cleaned.

```
startup_df=pd.read_csv("startup.csv")
startup_df.head(10)
```

ongitude	zip_code	id	city	Unnamed: 6	name	labels	...	object_id	has_VC	has_angel	has_roundA	has_roundB	has_roundC	has_roundD	avg_participants	is
-71.056820	92101	c:6669	San Diego	NaN	Bandsintown	1	...	c:6669	0	1	0	0	0	0	1.0000	
21.973718	95032	c:16283	Los Gatos	NaN	TriCipher	1	...	c:16283	1	0	0	1	1	1	4.7500	
117.192656	92121	c:65620	San Diego	San Diego CA 92121	Plixi	1	...	c:65620	0	0	1	0	0	0	4.0000	
22.050040	95014	c:42668	Cupertino	Cupertino CA 95014	Solidcore Systems	1	...	c:42668	0	0	0	1	1	1	3.3333	
22.419236	94105	c:65806	San Francisco	San Francisco CA 94105	Inhale Digital	0	...	c:65806	1	1	0	0	0	0	1.0000	
22.090370	94043	c:22898	Mountain View	Mountain View CA 94043	Matisse Networks	0	...	c:22898	0	0	0	1	0	0	3.0000	
22.070264	94041	c:16191	Mountain View	NaN	RingCube Technologies	1	...	c:16191	1	0	1	1	0	0	1.6667	
22.513742	94901	c:5192	San Rafael	NaN	ClairMail	1	...	c:5192	0	0	1	1	0	1	3.5000	
-73.203599	1267	c:1043	Williamstown	Williamstown MA 1267	VoodooVox	1	...	c:1043	1	0	1	0	0	1	4.0000	
22.145783	94306	c:498	Palo Alto	NaN	Doostang	1	...	c:498	1	1	1	0	0	0	1.0000	

Data Exploration:

Once the data is imported we started data exploration. There are 49 columns out of which only few are used for modeling. To select them we need to understand each of the columns in the dataset. There are 35 numeric and 14 object columns.

DISPLAYING THE CATEGORICAL OR OBJECT COLUMNS

```
[11] category_df=startup_df.select_dtypes(include='object')
     category_df.head()
```

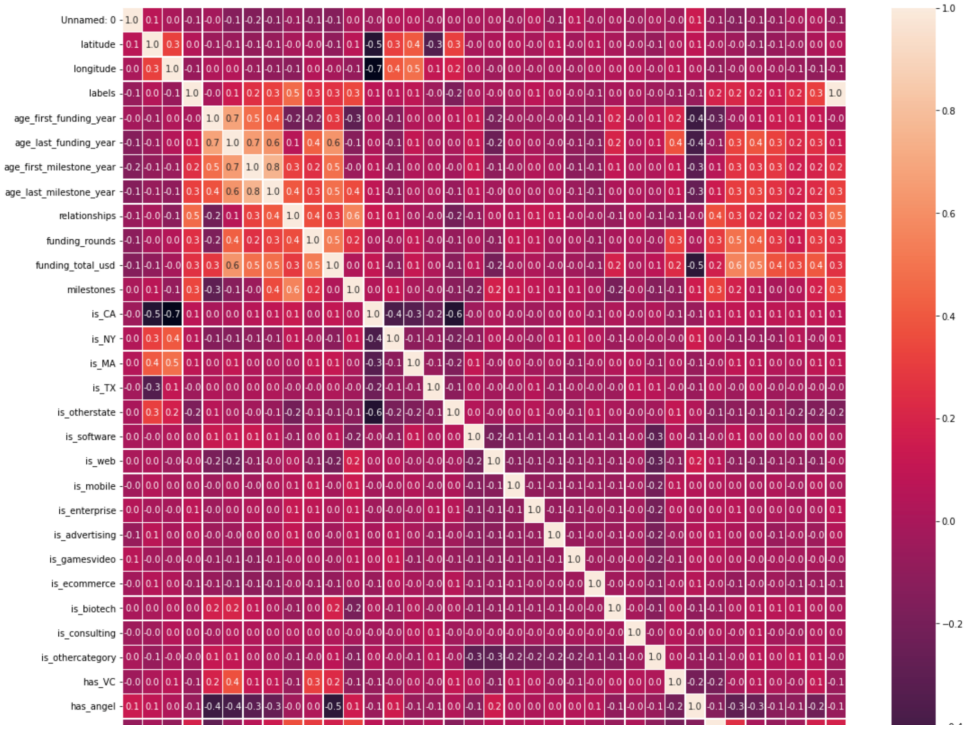
	state_code	zip_code	id	city	Unnamed: 6	name	founded_at	closed_at	first_funding_at	last_funding_at	state_code.1	category_code	object_id	status
0	CA	92101	c:6669	San Diego	NaN	Bandsintown	1/1/2007	NaN	4/1/2009	1/1/2010	CA	music	c:6669	acquired
1	CA	95032	c:16283	Los Gatos	NaN	TriCipher	1/1/2000	NaN	2/14/2005	12/28/2009	CA	enterprise	c:16283	acquired
2	CA	92121	c:65620	San Diego	San Diego CA 92121	Plixi	3/18/2009	NaN	3/30/2010	3/30/2010	CA	web	c:65620	acquired
3	CA	95014	c:42668	Cupertino	Cupertino CA 95014	Solidcore Systems	1/1/2002	NaN	2/17/2005	4/25/2007	CA	software	c:42668	acquired
4	CA	94105	c:65806	San Francisco	San Francisco CA 94105	Inhale Digital	8/1/2010	10/1/2012	8/1/2010	4/1/2012	CA	games_video	c:65806	closed

The status column is the target column where we see the two classes acquired and closed.We are converting the categorical value to numerical value where the acquired is replaced as 1 and closed is replaced as 0 for the purpose of modeling.

```
startup_df.isnull().sum()
```

```
Unnamed: 0                   0
state_code                   0
latitude                     0
longitude                    0
zip_code                     0
id                           0
city                         0
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
relationships                0
funding_rounds               0
funding_total_usd            0
milestones                   0
state_code.1                 1
```

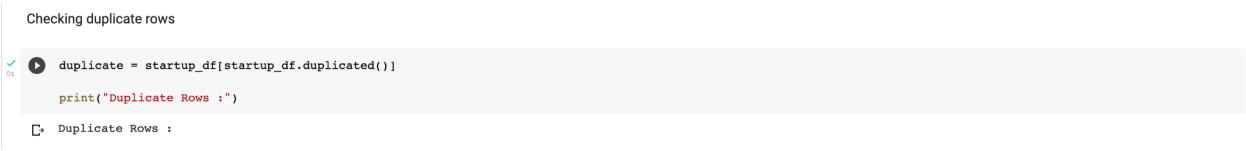
There are 1386 null values in the dataset. More than 50% of the data is missing in the unnamed:6 column and closed_at column. We are removing those columns in the data cleaning part.



The correlation of the columns is displayed as a heat map. Here we can see the strength of relationship between the numerical values.

Data Cleaning:

Checking duplicate values**:** Removing all the duplicate values in the dataset as they do not contribute to the generation of the model.

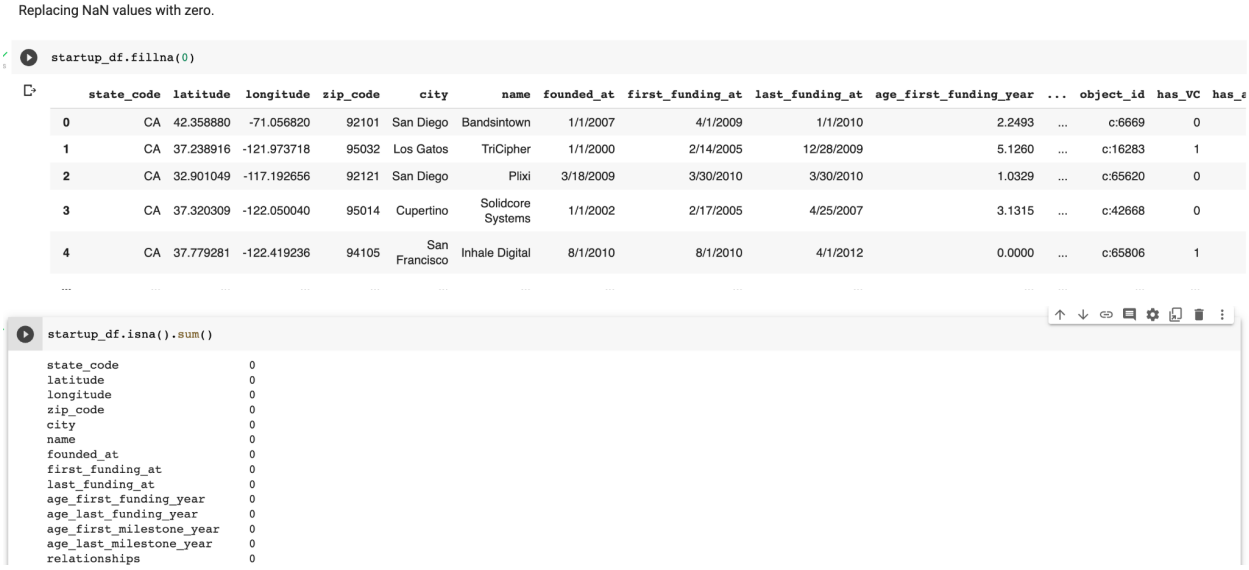


```
Checking duplicate rows

duplicate = startup_df[startup_df.duplicated()]
print("Duplicate Rows :")

Duplicate Rows :
```

Removing the irrelevant columns: The columns which is not included in the factors of predicting the success prediction of the start up. For example: some garbage columns as id, object_id, unammed_columns were identified during analysis as trivial. Hence we removed them.

```
Removing irrelevant columns

[64] startup_df = startup_df.drop(["Unnamed: 0", "Unnamed: 6", "labels","closed_at", "id"], axis=1)
```

Replacing NaN values with zero: Some columns/rows have NaN values, which are not required while computation. Hence, to carry out any operations we convert it into a numeric value such as 0 or any other values relevant.



Replacing NaN values with zero.

```
startup_df.fillna(0)
```

	state_code	latitude	longitude	zip_code	city	name	founded_at	first_funding_at	last_funding_at	age_first_funding_year	...	object_id	has_VC	has_s
0	CA	42.358880	-71.056820	92101	San Diego	Bandsintown	1/1/2007	4/1/2009	1/1/2010	2.2493	...	c:6669	0	
1	CA	37.238916	-121.973718	95032	Los Gatos	TriCipher	1/1/2000	2/14/2005	12/28/2009	5.1260	...	c:16283	1	
2	CA	32.901049	-117.192656	92121	San Diego	Plixi	3/18/2009	3/30/2010	3/30/2010	1.0329	...	c:65620	0	
3	CA	37.320309	-122.050040	95014	Cupertino	Solidcore Systems	1/1/2002	2/17/2005	4/25/2007	3.1315	...	c:42668	0	
4	CA	37.779281	-122.419236	94105	San Francisco	Inhale Digital	8/1/2010	8/1/2010	4/1/2012	0.0000	...	c:65806	1	

```
startup_df.isna().sum()

state_code                0
latitude                  0
longitude                 0
zip_code                  0
city                      0
name                      0
founded_at                0
first_funding_at          0
last_funding_at           0
age_first_funding_year    0
age_last_funding_year     0
age_first_milestone_year  0
age_last_milestone_year   0
relationships             0
```

Replacing NaN values with median value: Some columns factors majorly in computation and having insignificant values can hamper the modelling. Therefore, the Nan values are replaced by the median values.

Filling missing values

```
[67] startup_df['age_first_milestone_year'] = startup_df['age_first_milestone_year'].fillna(startup_df['age_first_milestone_year'].median())
```

```
[68] startup_df['age_last_milestone_year'] = startup_df['age_last_milestone_year'].fillna(startup_df['age_last_milestone_year'].median())
```

Replacing negative values: Some columns like last_year_fundings ,have negative values and cause error in the model generation.

Removing negative values

```
[69] startup_df=startup_df.drop(startup_df[startup_df.age_first_funding_year<0].index)
     startup_df=startup_df.drop(startup_df[startup_df.age_last_funding_year<0].index)
     startup_df=startup_df.drop(startup_df[startup_df.age_first_milestone_year<0].index)
     startup_df=startup_df.drop(startup_df[startup_df.age_last_milestone_year<0].index)
```

Feature Engineering: Created a few new columns by combining multiple columns into one.

```
startup_df['has_RoundABCD'] = np.where((startup_df['has_roundA'] == 1) | (startup_df['has_roundB'] == 1) | (startup_df['has_roundC'] == 1) | (startup_df['has_roundD'
startup_df.head()
```

ıg_at	last_funding_at	age_first_funding_year	...	has_VC	has_angel	has_roundA	has_roundB	has_roundC	has_roundD	avg_participants	is_top500	status	has_RoundABCD
/2009	1/1/2010	2.2493	...	0	1	0	0	0	0	1.0000	0	1	0
/2005	12/28/2009	5.1260	...	1	0	0	1	1	1	4.7500	1	1	1
/2010	3/30/2010	1.0329	...	0	0	1	0	0	0	4.0000	1	1	1
/2005	4/25/2007	3.1315	...	0	0	0	1	1	1	3.3333	1	1	1
/2010	4/1/2012	0.0000	...	1	1	0	0	0	0	1.0000	1	0	0

Creating new column has_investor: It would help us understand the credibility of the startup

```
startup_df['has_Investor'] = np.where((startup_df['has_VC'] == 1) | (startup_df['has_angel'] == 1), 1, 0)
startup_df.head()
```

last_funding_at	age_first_funding_year	...	has_angel	has_roundA	has_roundB	has_roundC	has_roundD	avg_participants	is_top500	status	has_RoundABCD	has_Investor
1/1/2010	2.2493	...	1	0	0	0	0	1.0000	0	1	0	1
12/28/2009	5.1260	...	0	0	1	1	1	4.7500	1	1	1	1
3/30/2010	1.0329	...	0	1	0	0	0	4.0000	1	1	1	0

Using a column invalid start up to discard it as an input to the model

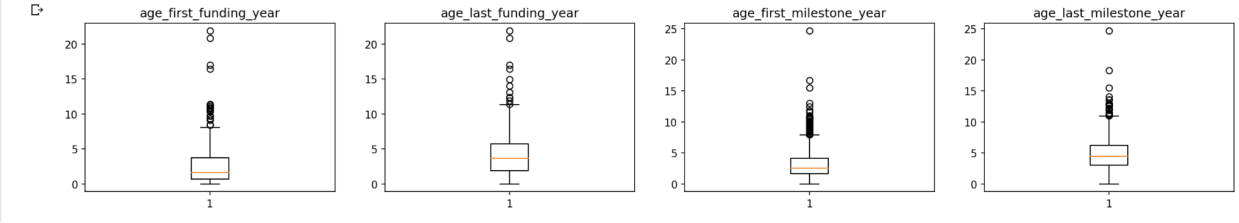
```
startup_df['invalid_startup'] = np.where((startup_df['has_RoundABCD'] == 0) & (startup_df['has_VC'] == 0) & (startup_df['has_angel'] == 0), 1, 0)
startup_df.head()
```

_funding_at	age_first_funding_year	...	has_roundB	has_roundC	has_roundD	avg_participants	is_top500	status	has_RoundABCD	has_Investor	has_Seed	invalid_startup
1/1/2010	2.2493	...	0	0	0	1.0000	0	1	0	1	1	0
12/28/2009	5.1260	...	1	1	1	4.7500	1	1	1	1	0	0
3/30/2010	1.0329	...	0	0	0	4.0000	1	1	1	0	0	0

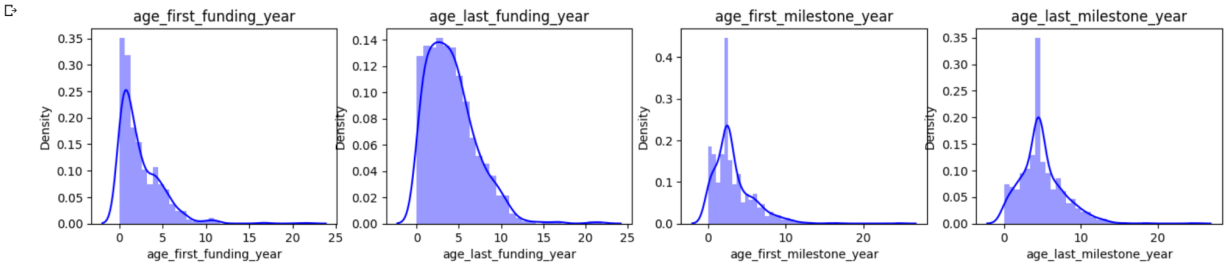
Handling Outliers: During data analysis, we identified outliers using box plots, one of the effective methods to spot outliers is to visualise on the graph. Use those data points and by using normalisation, we spread the data points accordingly.

Handling outliers

```
variable=["age_first_funding_year","age_last_funding_year","age_first_milestone_year","age_last_milestone_year"]

plt.figure(figsize=(20,3),dpi=150)
for i in range(len(variable)):
    plt.subplot(1,4,i+1)
    plt.title("{}". format(variable[i]))
    plt.boxplot(startup_df[variable[i]]);
```




```
variable=["age_first_funding_year","age_last_funding_year","age_first_milestone_year","age_last_milestone_year"]

plt.figure(figsize=(17,3),dpi=100)
for i in range(len(variable)):
    plt.subplot(1,4,i+1)
    plt.title("{}". format(variable[i]))
    sns.distplot(startup_df[variable[i]], color="blue");
```



Scaling the data: To remove bias towards a certain feature having higher magnitude and smoothning the flow of gradient descent.

```
from sklearn.preprocessing import StandardScaler

scale= StandardScaler()
scaled_data = scale.fit_transform(X)
```

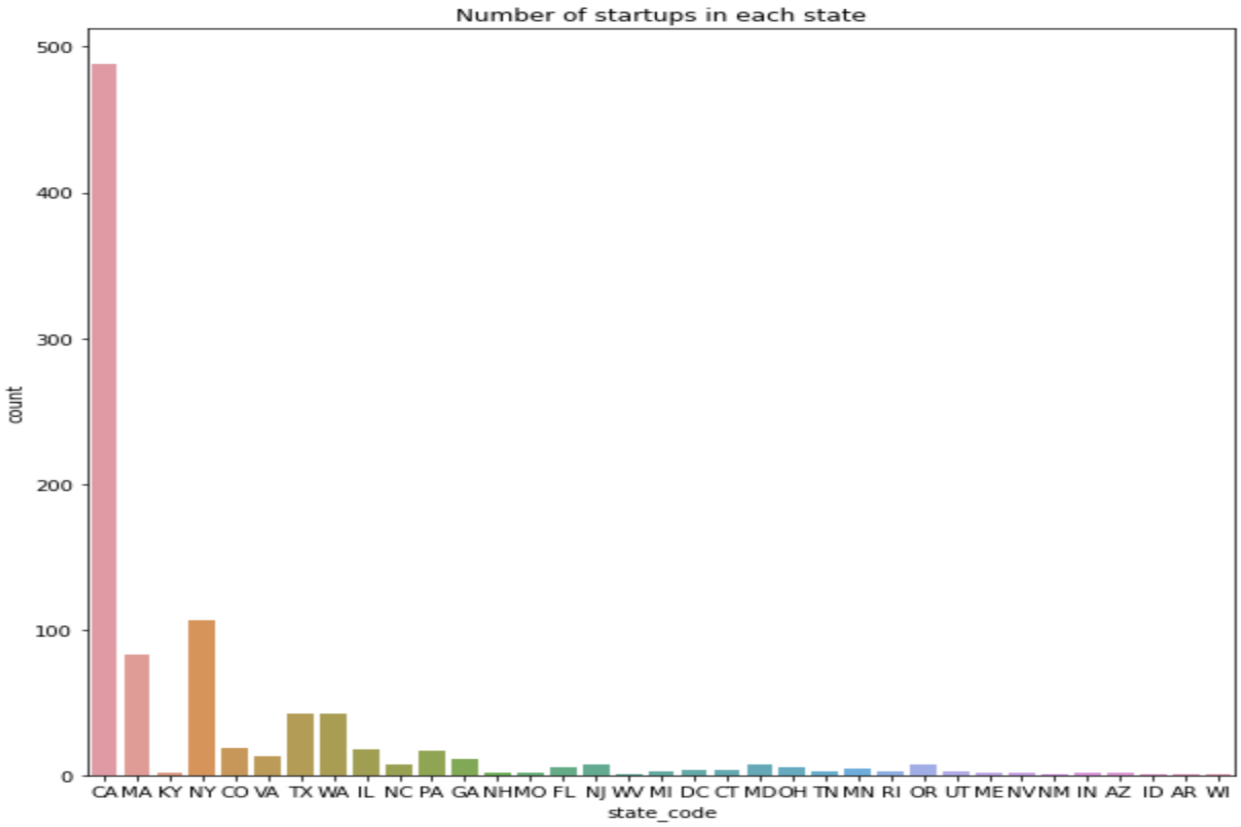
2. Findings / results so far:

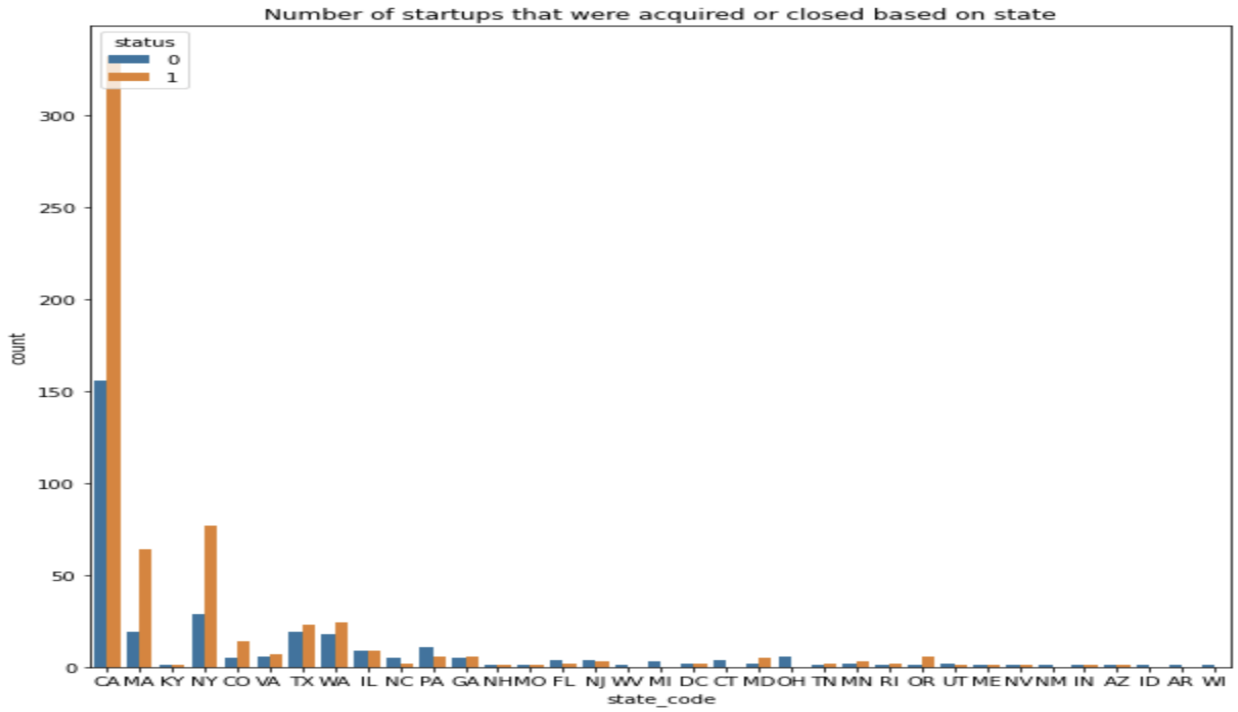
```
Percentage of missing values in 'closed_at' :  64%
Percentage of missing values in 'Unnamed: 6': 53%
```

2.1 The percentage of null values in both columns is more than 50 %.In the closed_at column, the null values are almost 63%, and using that column will not anyway help us to find the status so we are removing the column. Unnamed 6 is just about the state and zip code combined together which we already have as a separate column. So we are removing both these columns.

2.2 Age_last_milestone_year, relationships, funding_rounds, funding_total_usd, milestones, and is_top500 are the columns that have positive relationship with the status. So they play a major role in determining the target.

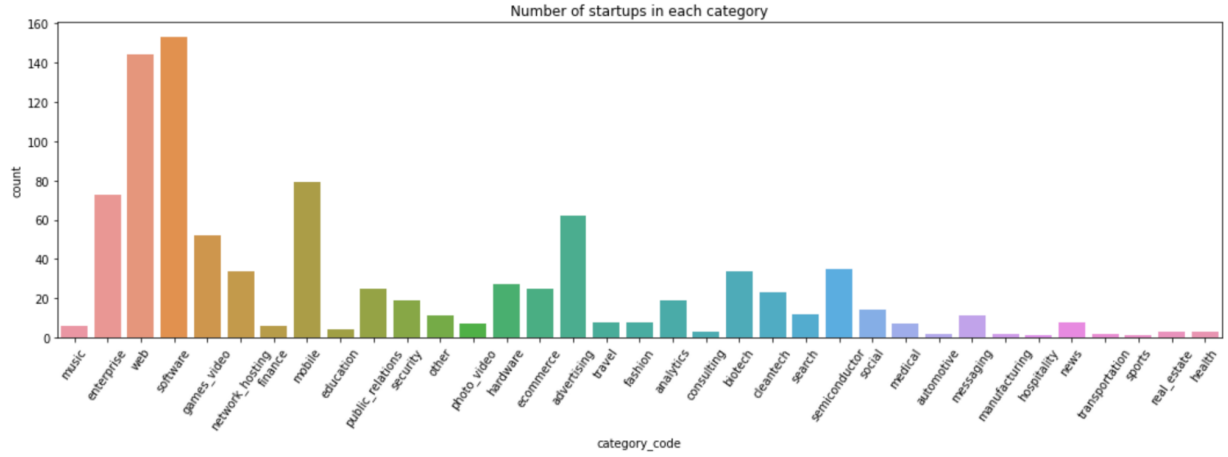
2.3

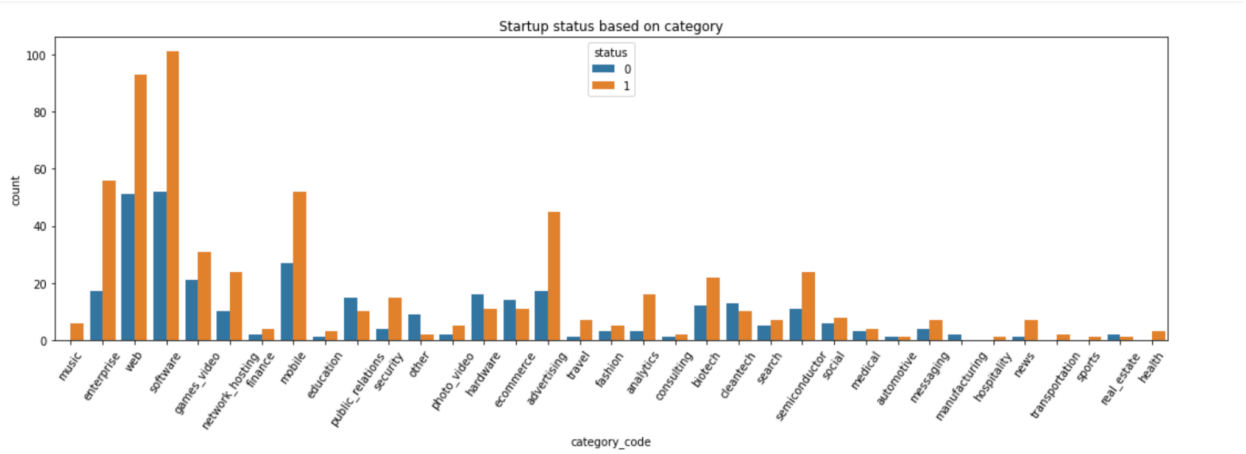




The number of startups in California is higher than others and also the number of startups that achieved success is more than 50% in California. So the state is also a major factor to decide on the status of the startups.

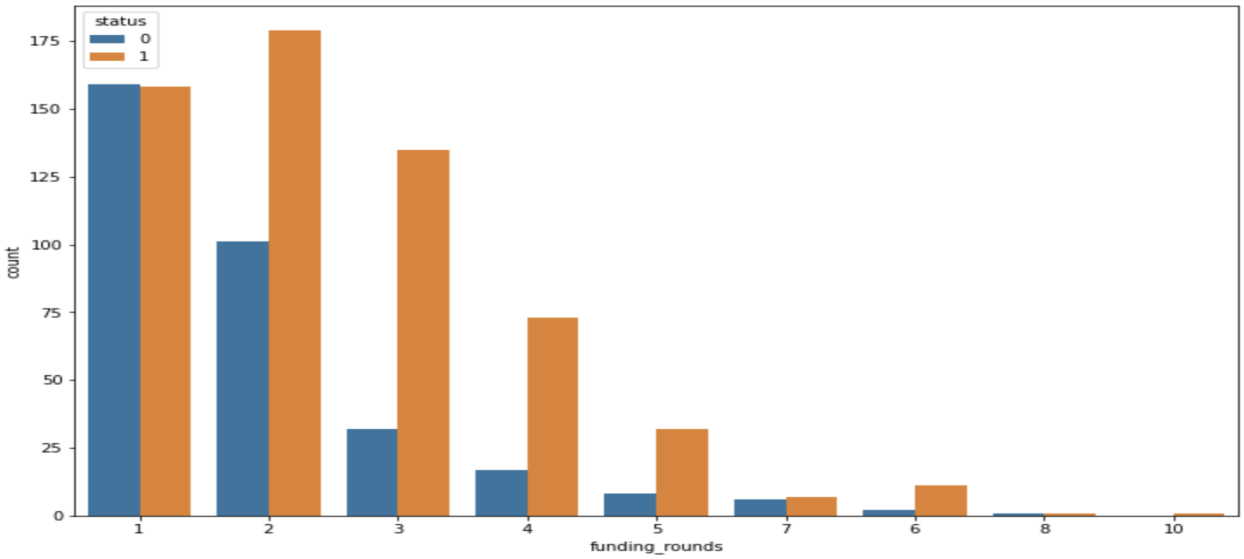
2.4





Many startups are open for software and the software startups are the most successful in the second the web has a good success rate after software.

2.5



3. Difficulties being encountered and how you plan to resolve them

3.1 Undersatnding the domain: To build a model, a developer should have a proper understanding the domain to build a model and choose relevant features. Understanding the domain helps build an intuition and verifies our process of feature engineering and guage the columns required. We took 2 days to read the various factors important for the success of the start up from multiple sources.

3.2 Finding datasets: We identified the use case, however finding the right dataset was a challenge. We referred to the previous research on the related problem statement and finalised

the structure of the dataset. It helped us to find the dataset as we had a definite structure of the ideal dataset.

3.3 Quality of datasets: There are various websites and other platforms available to find the relevant dataset. However, the quality of the dataset meaning the amount of null values, extra columns, duplicate rows and garbage values in the datasets are present in the dataset. This does not guarantee good accuracy. It will involve various steps for data cleaning. We defined rules and set some parameters for our dataset. Using these, we planned steps for data cleaning and pre processing.

3.4 Data cleaning: For some of the columns, we encountered garbage values like negative values for fundings, year etc, which cannot be provided as input for the model development. We did not anticipate those negative values would be present. Hence the further steps were not giving us an ideal output.
We chalked down all the possibilities to encounter garbage values in every columns and replaced it with null values or zero.

4. Remaining tasks

As the exploratory data analysis is done now the remaining tasks are:
4.1 Split the dataset
We need to split the dataset to target and input features. Before applying the machine learning models, dataset will be splitted to Train and test dataset. This is necessary to measure the model. We are planning to take status column as our target and all other relevant columns after dropping as the input features.

4.2 Modelling
In this step, we are planning to apply various classification and regression models for evaluating the success of the startup. We haven't finalised the machine learning models to be used in our project. We will be selecting models according to the analysis of our feature engineering. Here we will be creating the model to train the dataset and then test the model to calculate the accuracy of the model.

4.3 Ensemble modeling
In this step we are going to combine our machine learning models and then calculate the score for improving the accuracy of analysis. This will help to achieve a better accuracy and performance compared to any single model.

4.4 Parameter Tuning
In this step, we are planning to do the hyper parameter tuning. This is done for improving the performance of the model. We need to tune the parameters for a better accuracy.

4.5 Prediction

In this step we will be testing the model with the test dataset. To calculate the performance of the model

5. Any others that you think is relevant

Exploring test case designs to test our model end to end.