STARTUP ASSESSMENT PREDICTION

DETAILED PROJECT PROPOSAL

**Team**:

Pratibha Iyengar  
Siddhant Sapte  
Jigar Panchal  
Nikhil Kamat  
Karishma Padale

**Date**:

03/27/2020

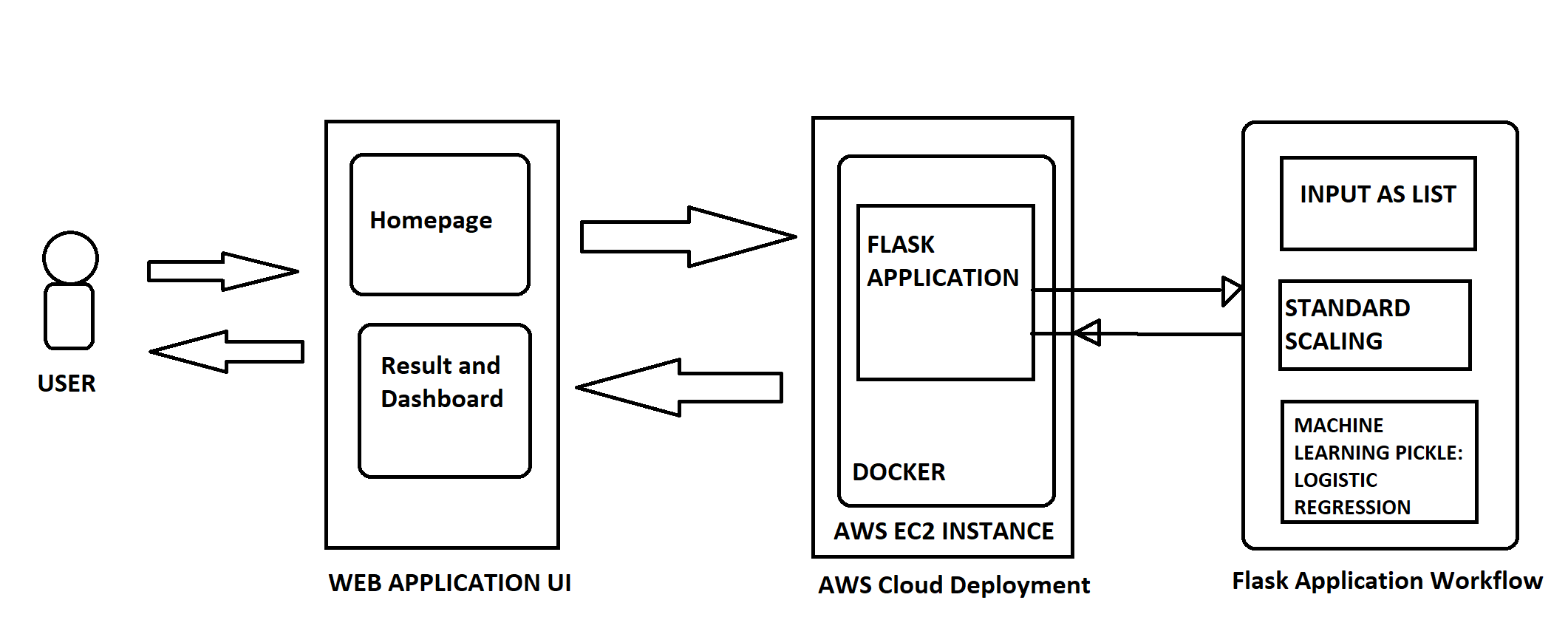
**Under the Guidance of**:

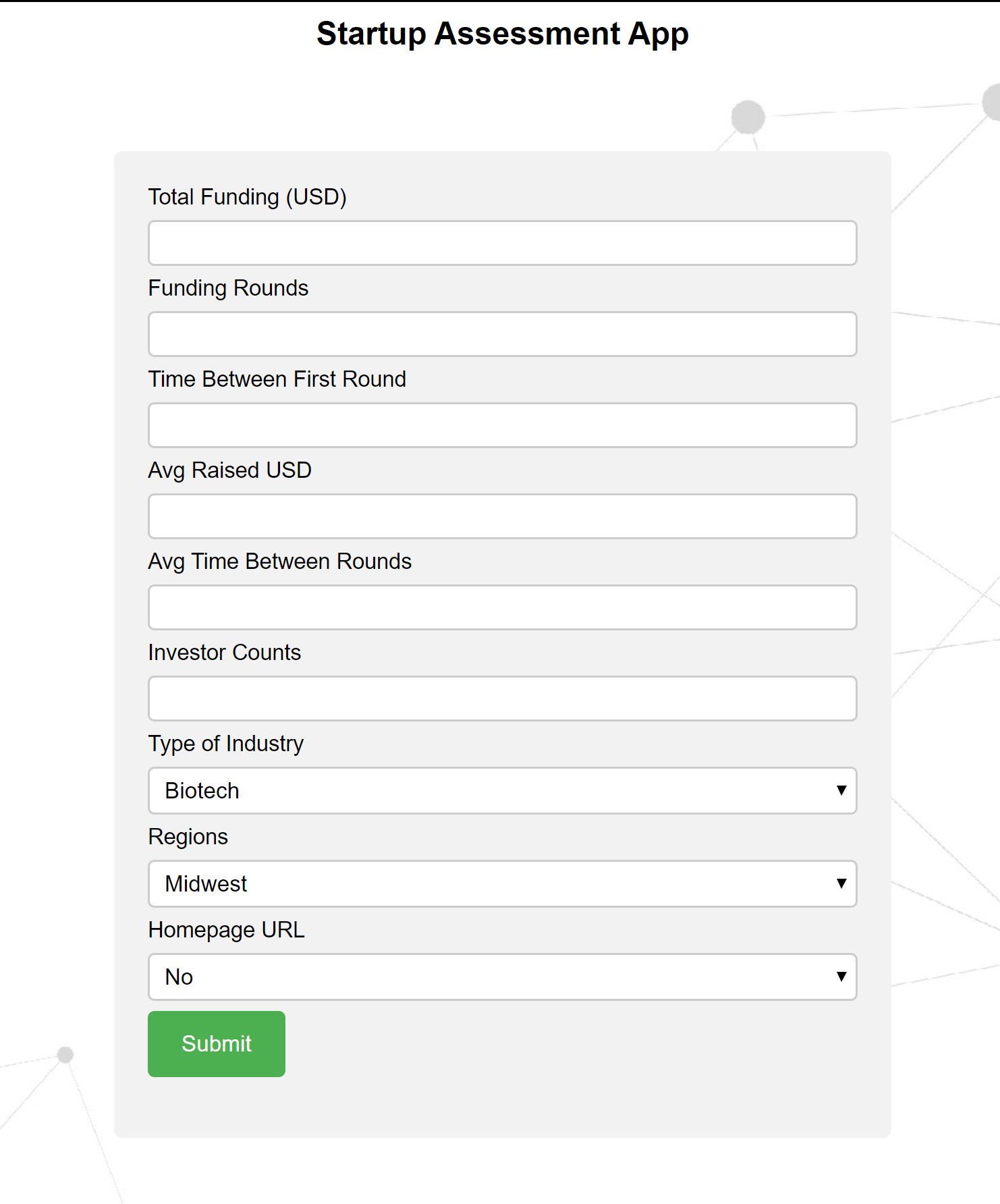
Professor Sharad Shandilya

**Project Objective:**

Our objective is to help Redcrow utilize predictive models to assess the success or failure of startup.  
We have built a Minimum Viable Project which is a web application built in Flask and deployed with docker on AWS EC2 instance.

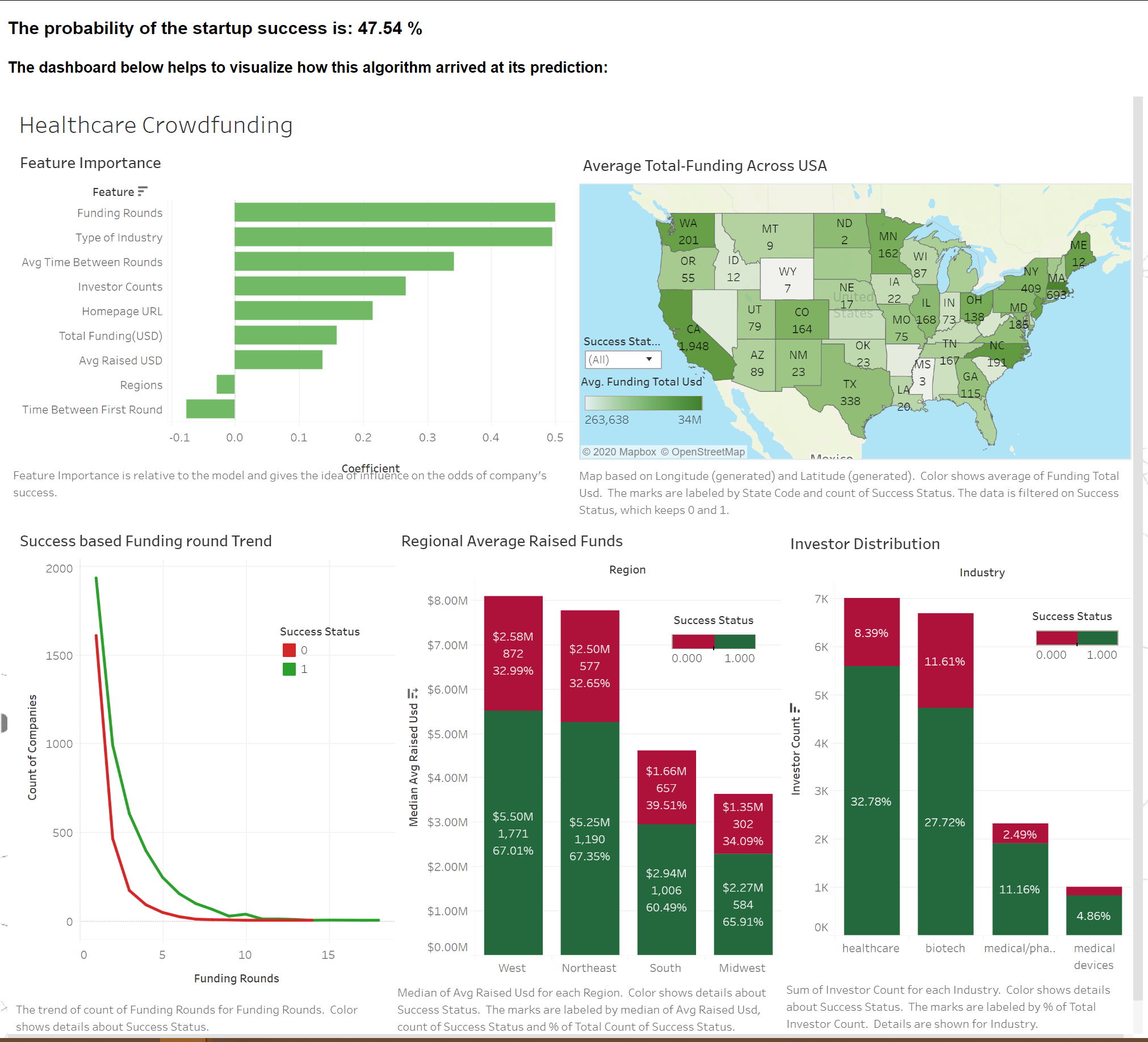
The idea is to take as input the feature values for the startup company under assessment and predict the success probability on logistic regression model in the background. We also provide a dashboard that visualizes the data, business insights and feature importance on top of which the model was derived.

**Project Workflow:**

The web application contains the startup feature input page and result page.   
The input page layout is as follows:  
 

The dockerized web application on AWS EC2 instance contains the pickle files that are used to indicate selected features, standard scaling and the trained machine learning model.

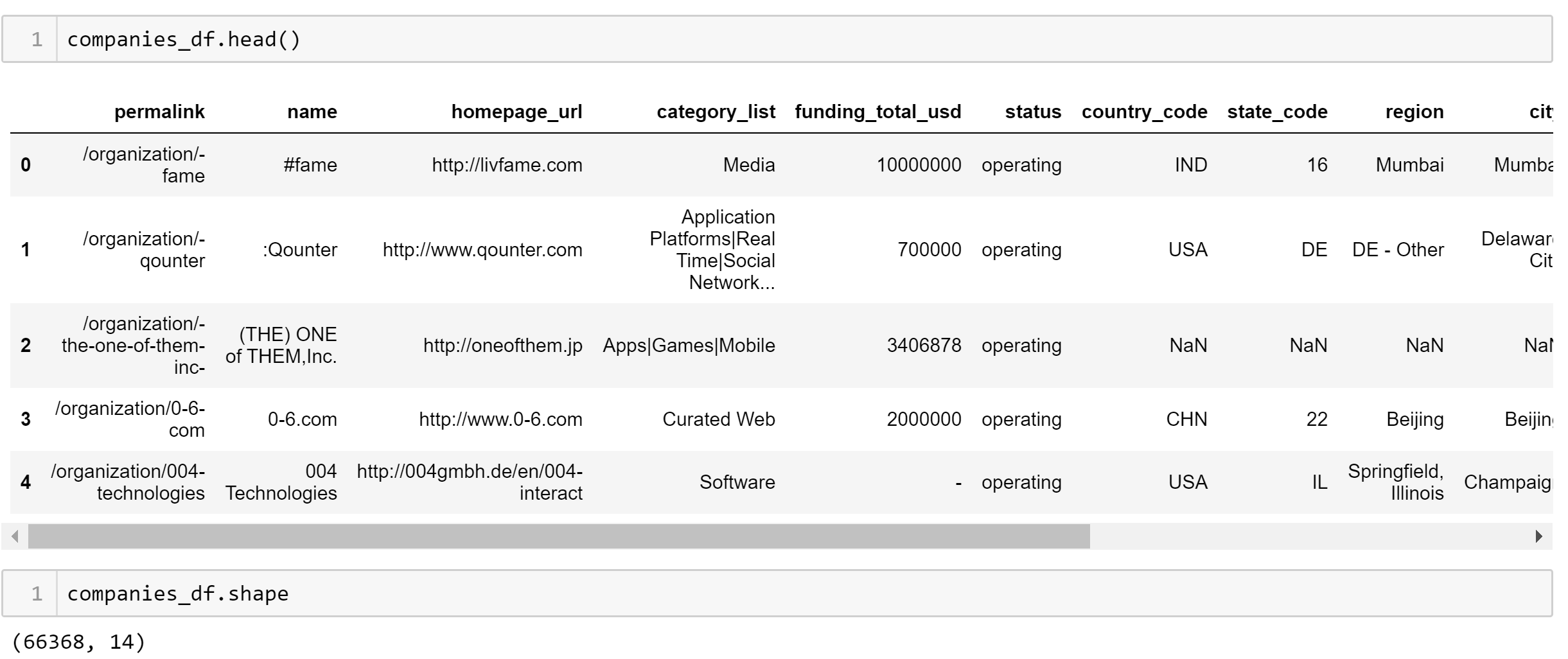
The features are passed as list to the scaler and then the model is run to predict the probability of success and renders a Tableau dashboard for descriptive analysis of the Healthcare crowdfunding trends, investor distribution and funding trends across USA. The Dashboard is self-explanatory with captions and Feature importance that indicate the basis of the model prediction.

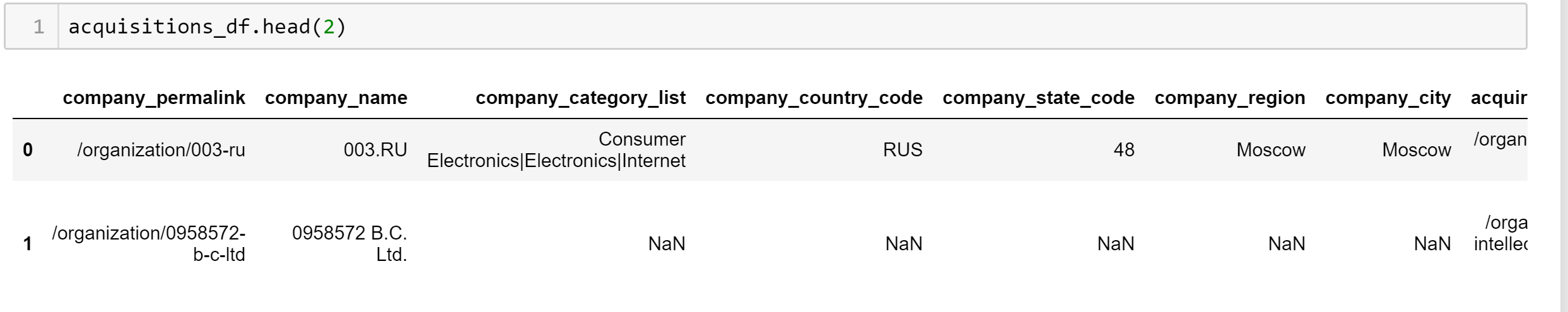
The result page displayed is as follows:  


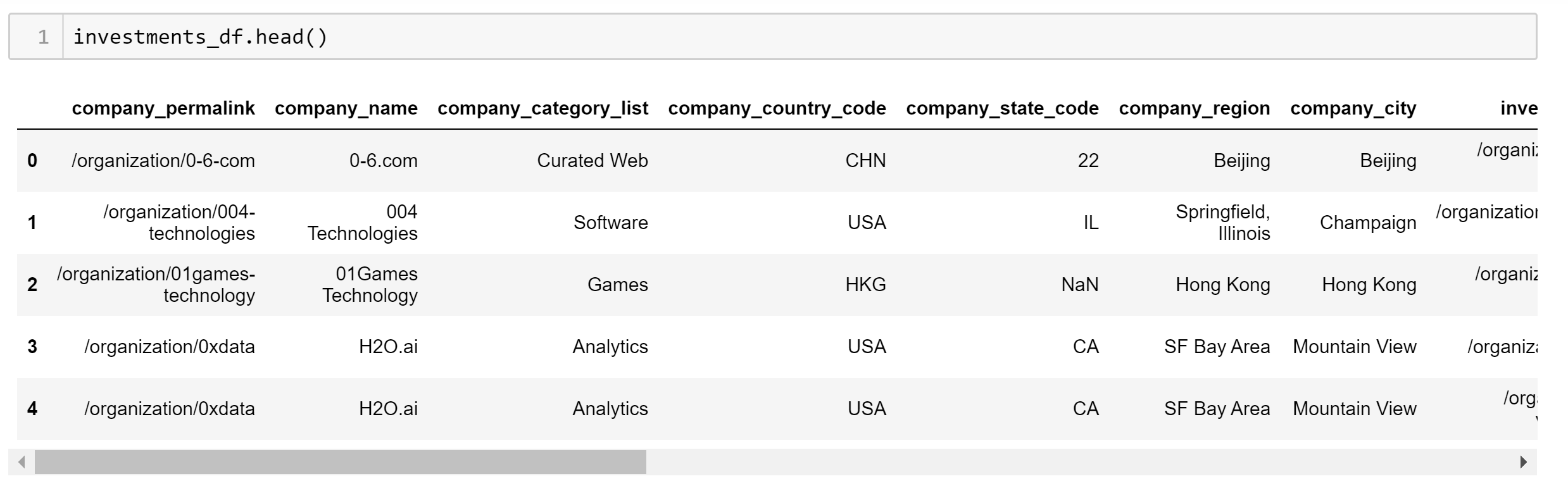
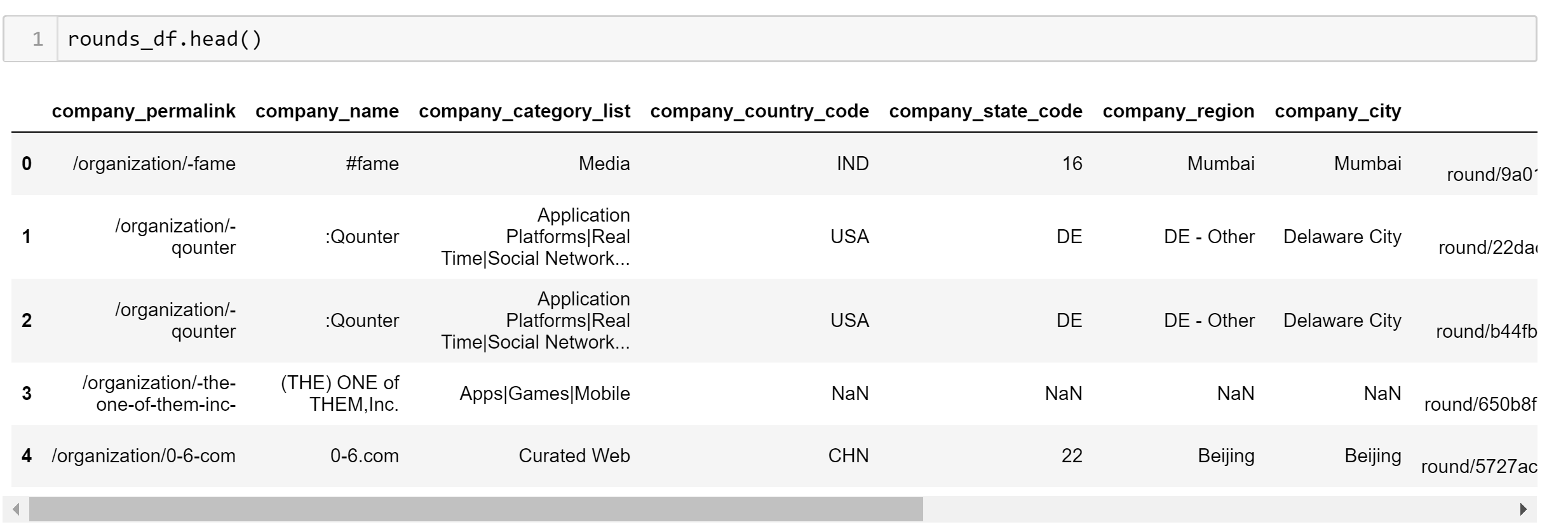
**Project Data:**

<https://www.kaggle.com/mauriciocap/crunchbase2013>

Data files:  
1. Company: Details about startups, fund total, first and last funding dates, status of company, hompage URL, category, funding rounds, location. One entry per company



2. Acquisition: All the acquisitions details , acquired and acquirer company details. One entry per acquisition  


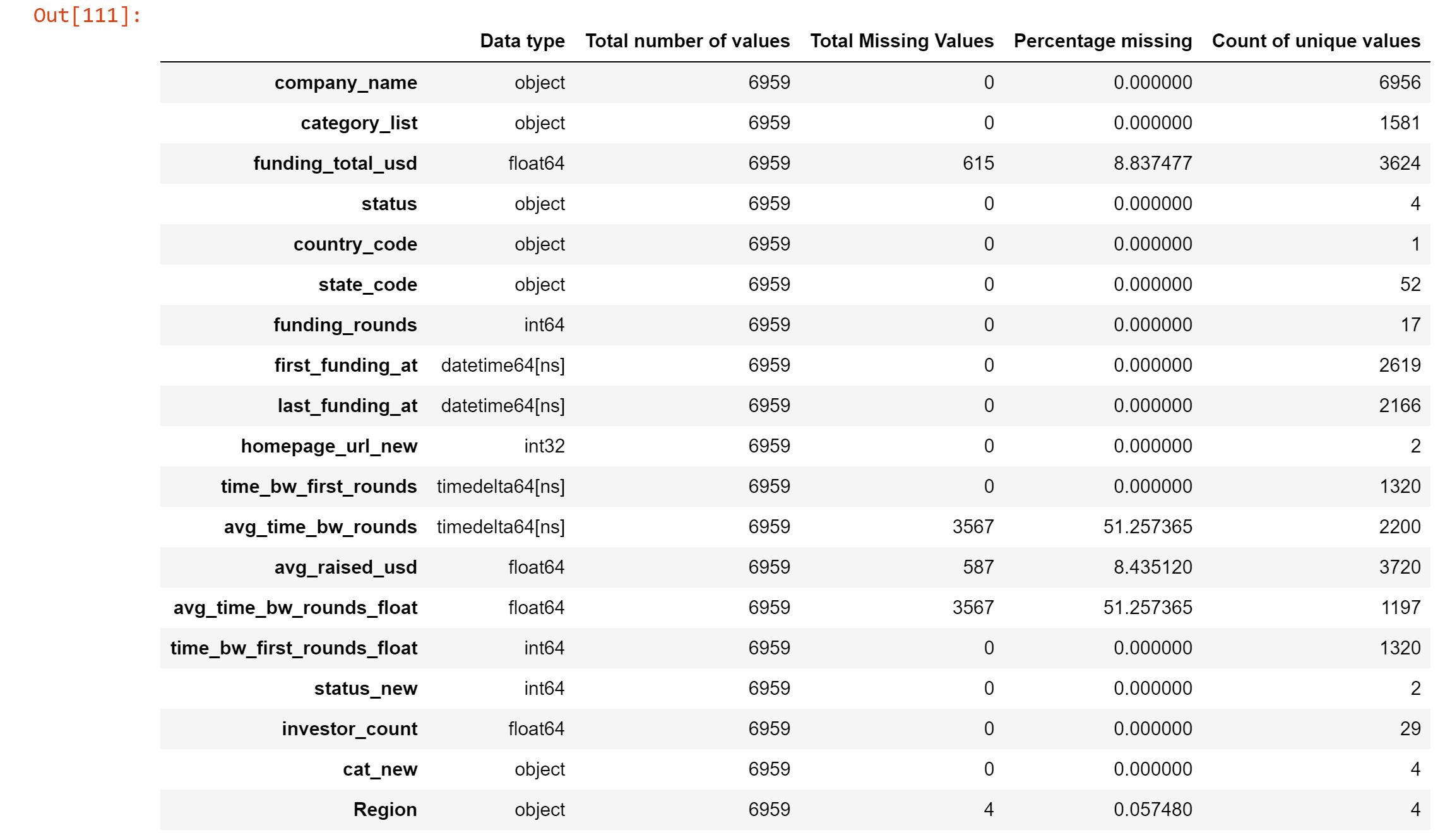
3. Investments: Investor details, amount raised, funding round type. One entry per investment   
  
4. Rounds: Rounds details, amount raised, funding round code, funding round type. One entry per round per company  


5. Additions.

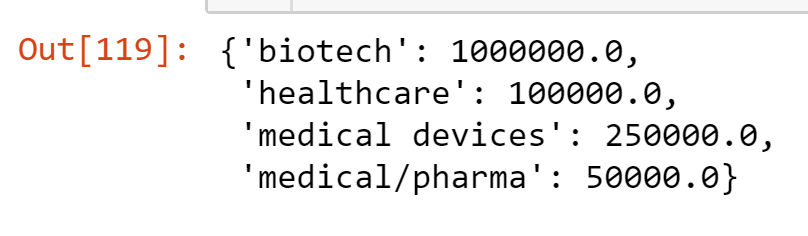
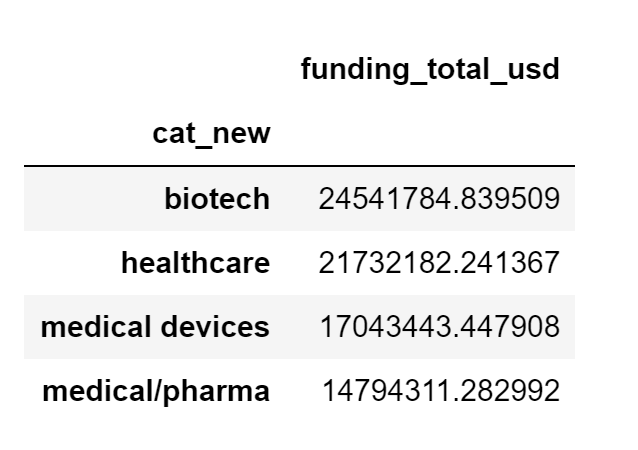
Our final dataset revolves around the company data and all healthcare related startups.

**Data Wrangling**

* Handled Null values in Company data and filtered healthcare related startups located in USA
* Performed date formatting in first and last funding dates to consider startups funded in 1900s and 2000s as they would be more relevant
* Converted categorical variable like homepage\_url and status to binary values 0 and 1
* While homepage\_url conversion is straightforward, we have made some assumptions for status variable.
* Status values: ‘operating’, ‘closed’, ‘acquired’, ’ipo’
* ‘acquired’ or ‘ipo’ are successful companies and ‘closed’ are failures
* Now companies in ‘operating’ status for a long time or which were funded a long time ago and do not have any recent campaigns, they are failures. To be precise, the latest funding year is 2015 and if companies are in operating state with first and last funding dates ranging between (with an offset of 2.5 years) 2013 and 10 years since before 2015, that is 2005, then it has practically seen no movement and can be labelled a failure.
* The rounds dataset has multiple records for the same company giving details about time between rounds, amount raised in each round. So, we aggregate the average time between two rounds and average amount raised in each round for that company. This information is merged on company to the main dataframe.
* Apart from this, we add the time between two rounds to the main dataset and this is calculated from the difference between two funding dates from the rounds dataset
* Next, we add the number of investors for each company from the investments dataset.
* Investor count for all companies is not available but we make an assumption for these companies that at least 1 customer will exist.
* The multi-valued categories feature is classified as biotech, healthcare, medical/pharma, medical devices.





* The above missing values were handled as follows:
  + **Region**: Region (or zone) is based on State, since state\_code is unknown, we fill it with mode value
  + **avg\_time\_bw\_rounds\_float**: Ideally,missing values in avg\_time\_bw\_rounds\_float with avg time between first and last round-  
    (Difference in first and last funding dates)/(funding\_rounds)= missing\_avg\_time\_bw\_rounds   
    but since first and last funding dates were same dates for all missing values, diff = 0 or time\_bw\_first\_rounds = 0 and division would be throw exception. Also logically, we could just say avg\_time\_bw\_rounds = 0
  + **funding\_total\_usd**: Fill missing values with mode in each category defined in category as that could be taken as a trend for those categories of startups  
    
  + **avg\_raised\_usd**: Fill null values with mean funding\_total raised in each category  
    
* Create X with independent variables (numeric and label encoded variables)and y = status\_new (0/1)

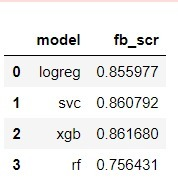
**Machine Learning Model**

* After Train-test split and Standard scaling, the training set was trained for 4 models- Logistic Regression, SVM, XGBoost, Random Forest
* We compared the train-test accuracy scores for logistic regression, SVM, XGBoost, random forest with default parameters
* A close up of a map

  Description automatically generatedBased on the scores and ROC curve analysis, we decided to go ahead and use Logistic Regression

A screenshot of a cell phone

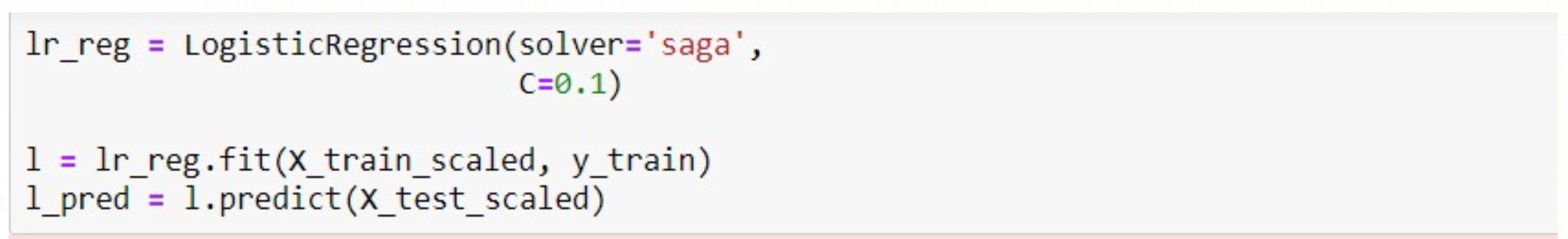
Description automatically generated

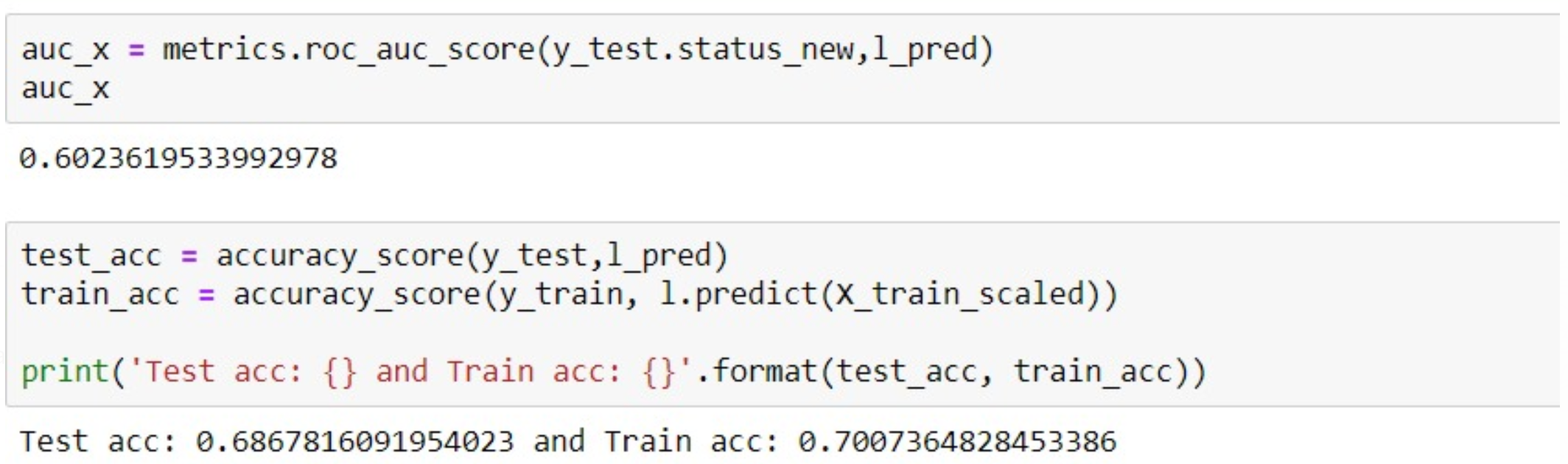
* Fbeta score was calculated to understand how well recall and precision of the models are  
  
* Logistic Regression and Xgboost consistently showed good scores and hence we decided to tune them

**Model Tuning**

**Logistic Regression**A simple model like logistic regression can sometimes be a poor generalization of data and end up in overfitting, with regularization we can avoid this by adding a penalty to the objective function and control model complexity.

For logistic regression, “c” parameter is used for regularization and is the go-to hyperparameter. So we set the set the c value to 0.1. The default value is 1.0 and smaller values of c render stronger regularization.

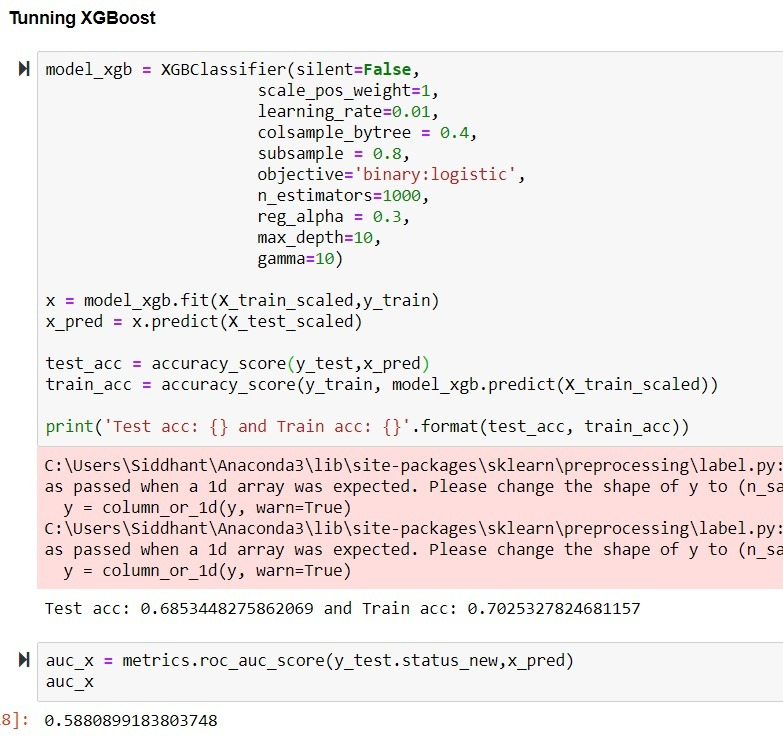


The hyperparameter tuning, in our case, did not influence the accuracy of the model greatly as the AUC value and train-test accuracy were quite close to the previous.   


**XGBoost**

The XGBoost model was tuned to for learning rate, estimators, max\_depth and gamma.

Here is a definition of each of these parameters as documented in <https://xgboost.readthedocs.io/en/latest/parameter.html>   
The estimators were increased to 1000 trees to accommodate and objective was switched to ‘binary:logistic’



On comparison the AUC score is slightly better for logistic regression and so our finalized model is Logistic Regression.

**Pickle Files**So for further usage, we use pickle files of the selected features, standard scaler and, logistic regression model.

Pickle file helps in serialization and deserialization of Python objects into a byte stream so that it can be stored in a file or transported over a network etc. The pickle library creates the instance of the original python object and populates it with the appropriate data.

The Pickle files listed above will be used to list the important features, scale it according to standard deviation of the train population and predict using the unpickled logistic regression model.   


**Web Application**We built the web application in Python Flask with two routes- Input and Result. The input is a set of Text and Drop-down fields which take inputs about the Startup being assessed.

The Feature values are :  
1. Total Funding(USD)  
2. Funding Rounds  
3. Time between first rounds  
4. Average Raised USD  
5. Average Time between Rounds  
6. Investor Counts  
7. Type of Industry (with Healthcare domain)  
8. Regions (In USA)  
9. Homepage URL (existing or not)

Then these inputs are passed as list using the deserialized feature selection pickle file and scaled using the standard scaler. Post which the prediction probability is obtained from the unpickled logistic regression model.

The prediction result is routed to the result.html page which also renders the Tableau dashboard based on the final dataframe used for modeling. The dashboard is explained in further detail in the Tableau Data Visualization section.

**Dockerization and AWS Services**Docker is an open-sourced tool made available by DotCloud and helps to create, run and deploy applications as a whole package including the libraries, dependencies and system configurations with help of containers. So, the application can run on any other Linux machine with a completely different set of settings. In other words, Docker brings virtualization with it but with the biggest convenience of not having to create an entire virtual OS as done with a Virtual Machine and instead it utilizes the Linux kernel of the host but with ability to run with the configurations it was originally created with.

We decided to use Docker containerization for this project due to the flexibility it provides and the convenience it will bring with it for future development and expansion of this platform.

So first we moved the application to test docker container locally. For this we leveraged the Oracle Virtual Machine to install, start and stop docker as required and to create container, move application files and pickle files to the docker container.

The Dockerfile has the following commands to use Python 3.6 container, run requirements.txt for installing Flask, Scikit-learn and numpy

Then built and ran the image to test usability of the application on localhost.

For moving the application to cloud, we decided the most optimal service that would also provide scalability along with dockerization would be an Amazon EC2 instance. We used the default EC2 instance for this project.

1. Remotely moved the application files to be deployed to EC2 server using scp command

2. Remotely logged into EC2 server using the ec2-instance PEM file which is the authentication key file from Oracle VirtualBox

3. Install docker

4. Started Docker service and build docker image with Dockerfile

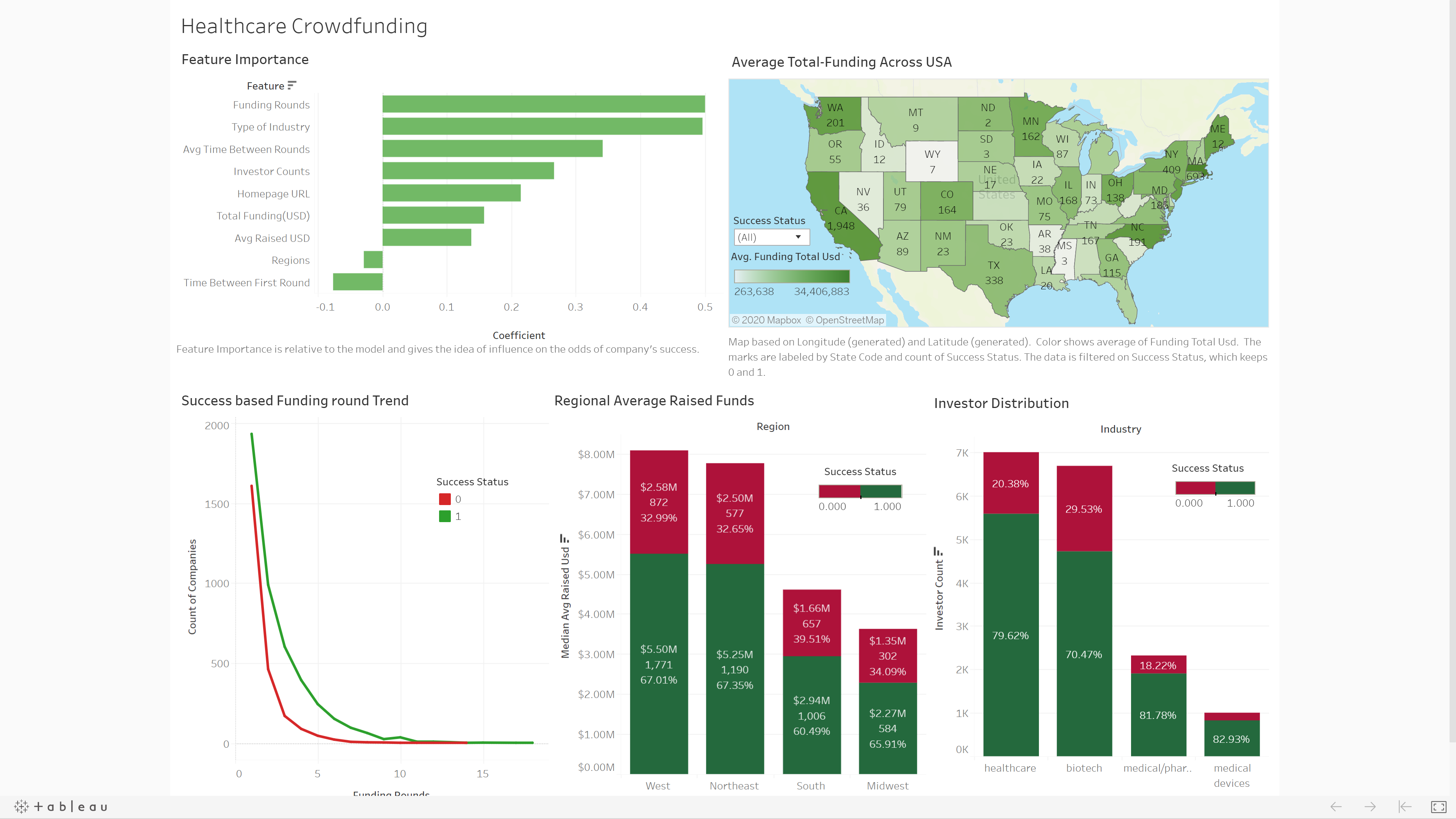
5. Run docker image and the application is up and ready

**Tableau Data Visualization**

The result is the predicted probability of startup success and a Tableau dashboard with indicators that helped us derive the model especially with the feature importance chart and will also help the user in deriving insights regarding crowdfunding trends, the funding round trends for success and failure startups, investor distribution. The dashboard is mainly a descriptive-analytical report on top of the predicted probability of success, in short.

Our project revolves around the success and failure of a startup and on those binary terms, we chose red and green to clearly indicate the divide in success status. The dashboard is not only to indicate the modeling basis, which is primarily focused on the Feature importance chart but also to give the assessor a sense of how the funds are distributed across the USA and its region.

All our charts are indicators of how investments have happened on the underlying data for successful and unsuccessful startups as indicators or in other words, helpful in adding perspective to the probability predicted with the machine learning model.

[](https://public.tableau.com/views/HealthcareCrowdfunding/Dashboard2?:display_count=y&:origin=viz_share_link)

**Insights**

Here are some insights we derived from the Charts:

* The number of funding rounds and the type of industry have the highest impact in predicting the success of a company followed by other critical features like average time between rounds, number of investors and existence of a Company Website;
* Average Total Funding in USA is concentrated on the West coast and East Coast of the country with high number of companies situated there too;
* While around 7 million USD was raised in the West, 2.58 million invested were on companies that eventually failed. The percentage of Average Raised Funds by failed companies is 32-39% across the regions. But this means the 39% of failed companies in South is a high number considering that the total is about 1660 companies and the raised amount is around 4 million USD which is far behind West and Northeast. So, we can say that companies in South and Midwest have lesser chance of becoming successful;
* Higher number of investors readily raise funds for Healthcare and Biotech companies;
* 70-82% of investors invested in startups that became a success while it is interesting to note that 82% of investors who invested in something uncommon like medical devices saw the highest success rate among the four categories;

**Conclusion**

* The current project model follows a static workflow but our scope extends to make it dynamic with taking inputs,( can also input feature details about many startups together), retraining the machine learning model and then predicting the probabilities
* Over a period, such a model would also be essential in gaining new insights and trends with their inclusion in dashboard data
* Redcrow can implement this idea in their own setting by utilizing equity crowdfunding data for analysis along with their assessment process with rating metrics and build a machine learning model that extracts important features for prediction in a healthcare equity crowdfunding setting
* The investor behavior or opinion can additionally be a separate set of analysis which would probably take a different line and scope while the Investment Assessment Committee ratings, investment transactions, rounds data, and comments can be used for prediction. The latter aligns best with the structure and project flow we have taken while the prior can be provided to the investors to bring their insights and perspectives with data rather than just intuitions.
* A combination of the Investment Committee report and a customized analytical reporting tool focused on investor interests can really up the game in Equity Crowdfunding platforms like RedCrow.
* We understand that Redcrow is a fast-growing platform and to accommodate scalability and flexibility with dockerization and leveraging AWS Cloud services which open ample possibilities to expansion and build creative solutions with the exponential growth of Redcrow

**REFRENCES**

* Xgboost.readthedocs.io. 2020. Xgboost Parameters — Xgboost 1.1.0-SNAPSHOT Documentation. [online] Available at: <https://xgboost.readthedocs.io/en/latest/parameter.html> [Accessed 28 March 2020].
* What is Docker? (n.d.). Retrieved from <https://opensource.com/resources/what-docker>
* Daly, D. J., & Daly, D. J. (1987). Economics 2: EC2. Retrieved from <https://aws.amazon.com/ec2/>
* Marks, H. (2018, December 19). What Is Equity Crowdfunding? Retrieved from [https://www.forbes.com/sites/howardmarks/2018/12/19/what-is-equity-crowdfunding/#57a55d6b3b5dhttps://www.forbes.com/sites/howardmarks/2018/12/19/what-is-equity-crowdfunding/#57a55d6b3b5d](https://www.forbes.com/sites/howardmarks/2018/12/19/what-is-equity-crowdfunding/%2357a55d6b3b5dhttps:/www.forbes.com/sites/howardmarks/2018/12/19/what-is-equity-crowdfunding/%2357a55d6b3b5d)