**Prediction of Success for Currently Operating Startups**

Dr. T. SampathKumar 1, a) Anugam Saikiran2, b) Irukulla Apoorva 3, c)

Anugam Uday kiran 4, d) Meghana Daddanala 5, e) Amogh Varsh Raju6, f)

Author Affiliations

1 SR University, Warangal, India 2 SR Engineering College ,Warangal, India

3 SR Engineering College, Warangal, India 4 SR University, Warangal, India

5 SR Engineering College, Warangal, India 6 SR Engineering College ,Warangal, India

Author Emails

*a) t.sampathkumar@sru.edu.in b) saikirananugam@gmail.comc)* [*irukullaapoorva@gmail.com*](mailto:irukullaapoorva@gmail.com)

*d) anguamudaykiran@gmail.come) meghanadaddanala@gmail.com f) varshrajuambati@gmail.com*

**Abstract.** Investing in startups is a high-risk, high-reward endeavor, and accurately predicting the success of currently operated startups is of paramount importance for investors. This research paper focuses on the prediction of startup success using machine learning algorithms and its significance for investment decisions. By leveraging historical data and a diverse set of features, machine learning models are trained to classify startups as successful or unsuccessful based on various factors such as financial data, milestones, average participants, advertising status, funding years, and relationship factors. This provides valuable insights for investors to make informed investment decisions by accurately assessing the potential success of startups, manage risks, optimize their investment portfolios, and increase their chances of achieving higher returns on their investments**.**

# INTRODUCTION

Predicting a startup's success helps investors to identify firms with the potential for quick development, putting them one step ahead of the competition.

In today's dynamic business landscape, the fate of a presently operating startup hangs in the balance, with success and failure often determined by a multitude of factors. Predicting the outcome of these ventures is of great interest to investors, founders, and stakeholders alike, as it can inform strategic decision-making and resource allocation. Machine learning, with its ability to extract meaningful patterns and insights from complex data [18], offers a promising approach to forecast the success or failure of these startups. The prediction of startup success involves analyzing various factors that contribute to their growth and sustainability [19,20]. These factors include financial indicators, market trends, analysis of funding data, team composition, industry dynamics, and other relevant variables.

Feature importance [11] and feature selection [9,10] techniques play a pivotal role in startup success prediction. These approaches enable the identification and prioritization of key factors that contribute to a startup's growth and performance.

Ensemble approaches [7,8,11], which combine multiple machine learning models, have emerged as powerful tools for startup success prediction. By leveraging the collective intelligence of diverse models, ensemble techniques enhance prediction accuracy and robustness. Transfer learning [12], another prominent technique, allows the knowledge gained from one domain to be applied to another, enabling the utilization of pre-trained models and accelerating the prediction process.

Kernel functions [13], such as radial basis function (RBF) and polynomial kernels [14], play a crucial role in capturing complex relationships and patterns in startup data. These functions transform data into higher-dimensional spaces, facilitating more accurate predictions. Hyperparameter tuning [16] and optimization [15] techniques further enhance the performance of machine learning models by fine-tuning their parameters and optimizing their decision boundaries. In the context of imbalanced data, oversampling techniques [17] can address the challenges posed by minority classes and improve prediction accuracy.

Startups have significant levels of uncertainty and failure, yet a small percentage of them go on to be successful and important. Some businesses go on to become unicorns. So investing in a company which will succeed in future is important [9].

The goal is to forecast whether a presently operational startup will achieve success or failure. Success for a company is characterized by the occurrence of an M&A (Merger and Acquisition) or an IPO (Initial Public Offering) that generates a substantial amount of money for the founders. On the other hand, a company would be deemed a failure if it needs to be closed down.

In this research paper, we have employed key features from the Kaggle dataset such as relationships, ranking, average participants, milestones, and funding years to assess their importance in predicting the success of startups. These features were used to train various machine learning models, such as DecisionTreeClassifier, SupportVectorClassifier, RandomForestClassifier, KNeighbourClassifier, and LogisticRegression. Subsequently, the trained models were saved and incorporated into a user-friendly interface. Through this interface, users can input their own startup data and obtain predictions on the likelihood of future success. By utilizing these machine learning algorithms and feature importance, we aim to provide a practical tool for users to evaluate the potential success of their startups. The implementation of the UI can be seen in Fig.9 and Fig.10.

# RELATED WORK

In [1] The proposed system deals with a hybrid model for startup success prediction. The authors combine Decision Tree, Random Forest, and Logistic Regression algorithms to improve prediction accuracy. The results of the study demonstrated that the hybrid model outperformed individual machine learning algorithms in predicting startup success. By combining the strengths of Random Forest and SVM, the proposed model achieved higher accuracy and more reliable predictions.

​​In [2] The suggested system authors employed several popular machines learning techniques, including Decision Tree, Random Forest, Naive Bayes, K-Nearest Neighbors (KNN), and Support Vector Machines (SVM). The study focused on comparing the performance of these algorithms based on metrics such as accuracy, precision, recall, and F1-score. The authors also assessed the computational efficiency of each algorithm to determine their practical feasibility in real-world startup prediction scenarios.

In [3] The proposed system explores startup success prediction using machine learning techniques. The authors evaluate the performance of Decision Tree, Random Forest, Support Vector Machines (SVM), and Naive Bayes models for predicting startup success. The authors recognized the importance of accurately assessing the potential success of startups to aid investors, entrepreneurs, and stakeholders in making informed decisions.

In [4] The proposed framework introduces a Comprehensive Review of startup success prediction using machine learning techniques. The authors examine the performance of Decision Tree, Random Forest, SVM, and Naive Bayes models for this task.

In [5] The review provides a comparative study of different machine learning algorithms for startup success prediction. The authors compare the performance of Logistic Regression, Decision Tree, Random Forest, GBM, and SVM models.

In [6] The proposed study focuses on predicting startup success using machine learning techniques. The authors explore the performance of KNN, Random Forest, SVM, and Naive Bayes models.

# PROPOSED METHODOLOGY

## Data Set

The Kaggle dataset encompasses a comprehensive range of information crucial for analyzing presently operated startups. It includes 48 columns or features that offer valuable insights into industry trends, investment patterns, and specific details about individual companies. These features consist of both quantitative and categorical variables, providing a rich set of data for analysis. Quantitative attributes such as age at first and last funding, relationships, funding rounds, funding totals, and milestone achievements offer valuable metrics for assessing a startup's progress and trajectory. Categorical features such as state, industry type, and funding sources (VC, angel, rounds A to D) provide contextual information that can help identify patterns and correlations. Additionally, features like average participants and top 500 status contribute further dimensions to the dataset. The ultimate goal is to leverage this data to build predictive models that can accurately forecast the success or failure of presently operated startups, as indicated by their acquisition or closure status.

## Data Preprocessing

1. Outliers Detection:

This research paper introduces a method for outlier detection in datasets using the Z-score approach. The code snippet presents a function called `detect\_outliers\_zscore`, which takes a dataset as input and applies the Z-score calculation to identify outliers. In this method, a threshold value of 3 is set to determine the number of standard deviations away from the mean a data point must be in order to be considered an outlier. The function calculates the mean and standard deviation of the dataset using the NumPy library. For each data point in the dataset, the Z-score is computed by subtracting the mean and dividing by the standard deviation. If the absolute value of the Z-score exceeds the threshold, the data point is marked as an outlier and added to the list of outliers. The function returns the list of identified outliers. This approach provides researchers with a practical and automated technique for detecting outliers, allowing for further analysis and treatment of anomalous data points. The effectiveness and reliability of this method can be evaluated by applying it to various datasets in different research domains.

To detect outliers in the dataset using the Z-score method, the code snippet was executed for multiple columns. The resulting outliers were stored in separate variables for analysis. The columns examined included 'is\_advertising', 'age\_first\_funding\_year', 'age\_first\_milestone\_year', 'relationships', 'milestones', 'avg\_participants', 'is\_top500', and 'status'. The outliers in each column were then printed, providing researchers with valuable information regarding the identified outliers using the Z-score method.

To remove the outliers, a new dataset was created by filtering out rows where the Z-scores were below the threshold. Finally, the original dataset was updated by assigning data\_z to data after dropping the Z-score columns using the drop function.

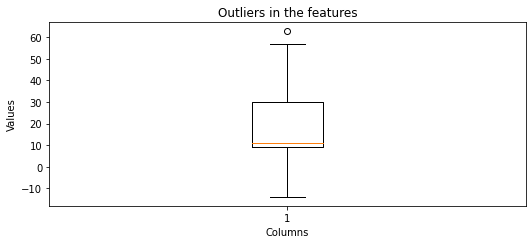


Fig.1 Boxplot showing outliers

The matplotlib library creates a box plot of the data shown in Fig.1 for the features age\_first\_funding\_year, age\_first\_milestone\_year, relationships, milestones, avg\_participants, list. The plt.boxplot function is used to create the box plot. The plt.title, plt.xlabel, and plt.ylabel functions set the title, x-axis label, and y-axis label of the graph, respectively. Finally, the plt.show function displays the graph.

2. Missing value treatment:

To handle missing values in the dataset, the mean values of each column were computed and stored in separate variables. The means were calculated for columns 'labels', 'age\_first\_funding\_year', 'age\_first\_milestone\_year', 'relationships', 'milestones', 'avg\_participants', 'is\_top500', and 'status', and were assigned to the eight variables.

Missing values in each column were then filled with their respective mean values using the `fillna` function. The 'labels' column was updated by filling the missing values with the variables assigned.

The resulting dataset is represented in Table.1, with the missing values imputed using mean values, were then obtained as 'data'. This approach allows for the preservation of data integrity by replacing missing values with representative measures while minimizing the impact on the overall dataset.

|  |  |  |
| --- | --- | --- |
| **S.no.** | **Column Name** | **Null values** |
| 1 | Unnamed:0 | 0 |
| 2 | state\_code | 0 |
| 3 | latitude | 78 |
| 4 | longitude | 4 |
| 5 | Unnamed:6 | 493 |
| 6 | name | 0 |
| 7 | labels | 0 |
| 8 | founded\_at | 0 |
| 9 | closed\_at | 588 |
| 10 | age\_first\_funding\_year | 21 |
| 11 | age\_last\_funding\_year | 12 |
| 12 | age\_first\_milestone\_year | 169 |
| 13 | age\_last\_milestone\_year | 159 |
| 14 | relationships | 60 |
| 15 | funding\_rounds | 16 |
| 16 | funding\_total\_usd | 65 |
| 17 | milestones | 1 |
| 18 | avg\_participants | 28 |
| 19 | is\_top500 | 176 |

Table.1 Number of missing values

3. Normalization:

To normalize the numerical variables in the dataset, the `MinMaxScaler` class from the `sklearn.preprocessing` module is used. A scaler object named 'scaler' was initialized using the `MinMaxScaler()` constructor.

The scaler was then applied to the selected numerical columns, namely 'age\_first\_funding\_year', 'age\_first\_milestone\_year', 'relationships', 'milestones', and 'avg\_participants', using the `fit\_transform()` method. This method scaled the values of these columns to the range [0, 1], ensuring that all numerical variables have a consistent scale.

The normalized values were then assigned back to the corresponding columns in the 'data' dataframe using scaler.fit\_transform method, using a list containing the names of the numerical columns.

By performing this scaling operation, the numerical variables in the dataset were transformed to a common scale, facilitating fair comparisons and eliminating the influence of different magnitude ranges.

## Models used

1. Logistic Regression

The logistic regression model is applied for startup success prediction. The LogisticRegression class from the sklearn.linear\_model module is imported, and an instance of the model named lr\_model is created using the default constructor. The model is then trained using the fit () method by providing the X\_train and y\_train data, representing the input features and target variable, respectively. During the training process, the model estimates the coefficients that capture the relationship between the predictors and the target variable. After training, the lr\_model can be used to make predictions on new, unseen data.

2. Support Vector Machine (SVM)

The Support Vector Machine (SVM) classifier is utilized for predicting startup success. The SVM classifier is imported from the sklearn.svm module and an instance of the classifier is created with a linear kernel and a specified random state. The classifier is then trained using the fit() method, where the input features (X\_train) and corresponding target variable (y\_train) are provided.

Once the classifier is trained, it is used to predict the target variable for the test data (X\_test) using the predict() method. The predicted values are stored in the y\_pred variable. To evaluate the performance of the classifier, a confusion matrix is generated by importing the confusion\_matrix function from sklearn.metrics.

Additionally, the accuracy score is calculated using the accuracy\_score function from sklearn.metrics, which measures the proportion of correctly predicted instances compared to the actual target values. This provides a single metric to assess the overall accuracy of the classifier's predictions.

3. KNeighborsClassifier

KNeighborsClassifier for startup success prediction, the algorithm should be imported from the sklearn.neighbors module. An instance of the classifier is created and assigned to a variable, typically named "classifier". The classifier can be customized by specifying parameters such as the number of neighbors (K) and the distance metric.To evaluate the performance of the KNeighborsClassifier, various metrics can be used, such as accuracy, precision, recall, and F1-score. These metrics provide insights into the classifier's ability to correctly classify startups as successful or unsuccessful based on the input features.

The classifier is instantiated without specifying any specific parameters. It is then trained using the fit() method by providing the training data, X\_train (input features), and y\_train (target variable). Once trained, the classifier is used to predict the target variable for the test data, X\_test, using the predict() method, and the predicted values are stored in the y\_pred variable. The confusion matrix and accuracy score serve as evaluation metrics to assess the performance of the classifier. These metrics provide valuable insights into the classifier's ability to make accurate predictions and distinguish between successful and unsuccessful startups.

4. DecisionTreeClassifier

The Decision Tree Classifier is a valuable tool for startup success prediction. By leveraging the decision tree structure and training the classifier on relevant data, it enables the identification of key factors that contribute to startup success and facilitates informed decision-making in the startup ecosystem.

The classifier is instantiated and assigned to the variable "dt" without specifying any specific parameters. It is then trained using the fit() method by providing the training data, X\_train (input features), and y\_train (target variable). Once trained, the classifier is used to predict the target variable for the test data, X\_test, using the predict() method, and the predicted values are stored in the variable "pred\_dt".

5.RandomForestClassifier

The RandomForestClassifier from the sklearn.ensemble module is employed for startup success prediction. The classifier is instantiated and assigned to the variable "classifier" with specified parameters. It utilizes an ensemble of decision trees to make predictions. It enables the prediction of success or failure for startups based on the input features. The confusion matrix and accuracy score provide insights into the performance of the classifier, indicating how well it can predict the success of startups.

# RESULTS AND GRAPHS

In Fig.2 the bar chart illustrates the mean accuracy scores of various machine learning algorithms for a specific task. The algorithms considered include Logistic Regression with a mean accuracy of 90.3%, K-Nearest Neighbors Classifier with 91.4% accuracy, Random Forest with an accuracy of 92.6%, Decision Tree achieving a mean accuracy of 93.7%, and Support Vector Classifier (SVC) with 89% accuracy.

These accuracy scores provide insights into the performance of each algorithm, where Decision Tree outperforms the others with the highest mean accuracy score. By visually comparing the bar lengths, we can easily assess the relative accuracies of the algorithms and make informed decisions about selecting the most suitable algorithm for the given task.

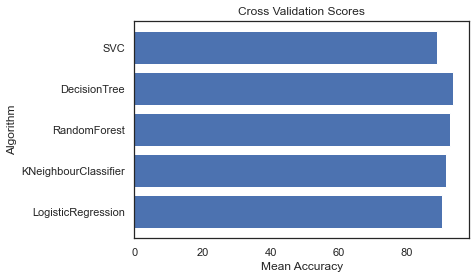


Fig.2 Barchart showing Accuracies of algorithms

|  |  |  |
| --- | --- | --- |
| S.no | Algorithm | Accuracy(%) |
| 1 | SVC(SupportVectorClassifier) | 89 |
| 2 | DecisionTreeClassifier | 93.7 |
| 3 | RandomForestClassifier | 92.6 |
| 4 | KNeighbourClassifier | 91.4 |
| 5 | LogisticRegression | 90.3 |

Table.2 Accuracies of algorithms

Table.2 displays the accuracy results of various machine learning algorithms, indicating that the DecisonTreeClassifier exhibits the highest accuracy among them.



Fig.3 RandomForest Feature Importances

The scatter plot Fig.3 is configured to use markers, and the size of the markers is set to 25. The color of the markers is determined by the 'model\_importances' column, allowing for a color representation of the feature importances. The colorscale is set to 'Portland' to provide a visually appealing representation. The 'text' parameter is set to the 'features' column, which displays the feature names when hovering over the markers as shown in Fig.3 by hovering the relationships.

The scatter plot is added to the 'data' list, and a layout for the plot is defined. The layout includes options for autosizing, setting the title to the 'model\_title', enabling hover mode, and configuring the y-axis label as 'Feature Importance'.

The feature importances are visualized using the Random Forest Classifier (RFC) model. We extracted the feature importances from the RFC model and created a DataFrame to store the importances along with their corresponding feature names. The feature importances are filtered to include only non-zero values, ensuring that only relevant features are considered. Then we used a plotting function to generate scatterplots for visualizing the feature importances.

|  |  |  |  |
| --- | --- | --- | --- |
| **S.no** | **Features** | **RandomForest Feature importances** | **mean** |
| 1 | age\_first\_funding\_year | 0.127713 | 0.127713 |
| 2 | age\_first\_milestone\_year | 0.126891 | 0.126891 |
| 3 | relationships | 0.368509 | 0.368509 |
| 4 | milestones | 0.211010 | 0.211010 |
| 5 | avg\_participants | 0.165877 | 0.165877 |

Table.3 RandomForest features importances and mean

We created a 'mean' function along the rows (axis=1) to calculate the mean value for each row. This mean value represents the average feature importance for that particular feature across the different models and is shown in Table.3.

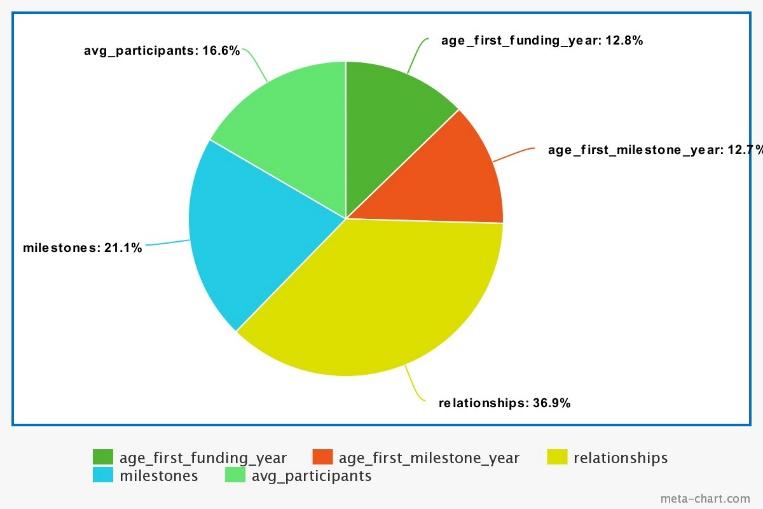


Fig.4 Piechart showing the percentages of feature importances

Fig.4 Piechart shows the calculation and addition of the average feature importances column can be useful for further analysis, such as identifying features that consistently have high importance across different models or comparing the relative importance of features in a more aggregated manner. Here we can understand that the relationships between the startups have the higher weightage and is essential for its success.

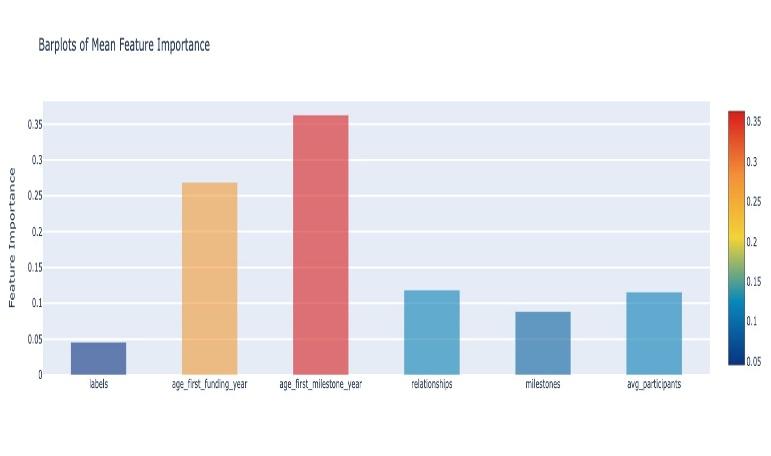


Fig.5 Mean Feature Importance

The visualization of mean feature importance values using an interactive bar plot is displayed in Fig.5. This plot can provide a clear and intuitive representation of the relative importance of different features in the dataset.

The bar plot is created using the 'go.Bar' trace from the 'plotly' library. The mean column' and feature column values are provided, and the width of the bars is set to 0.5. The color of each bar is determined by the corresponding 'mean' value, creating a color gradient using the 'colorscale' parameter.

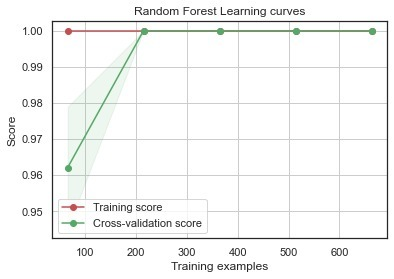


Fig.6 Learning curves for RandomForestClassifier

We used several parameters such as estimator: The model or estimator for which the learning curve generated, title: The title of the learning curve plot, X: The input features, y: The target variable, ylim: The y-axis limits for the plot (optional), cv: The cross-validation strategy, n\_jobs: The number of parallel jobs to run, train\_sizes: The training set sizes to use for the learning curve.

These learning curves Fig.6, Fig.7 can provide insights into the model's behavior, such as identifying issues related to overfitting or underfitting. By analyzing the convergence or divergence of the training and cross-validation scores, one can determine if the model would benefit from more data or if it exhibits high bias or variance. This information can guide model selection, parameter tuning, or data collection strategies to improve the model's performance.

The bar plot in Fig.8 is created using 'sns.barplot' with the feature importances as the x-axis values and the corresponding feature names as the y-axis values. The 'nclassifier' variable keeps track of the current classifier being plotted, and it increments after each subplot is created.

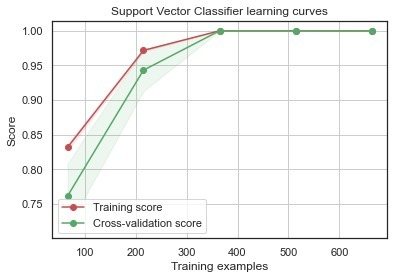


Fig.7 Learning curves of Support Vector Classifier

This visualization helps understand which features have the most significant impact on the model's predictions.

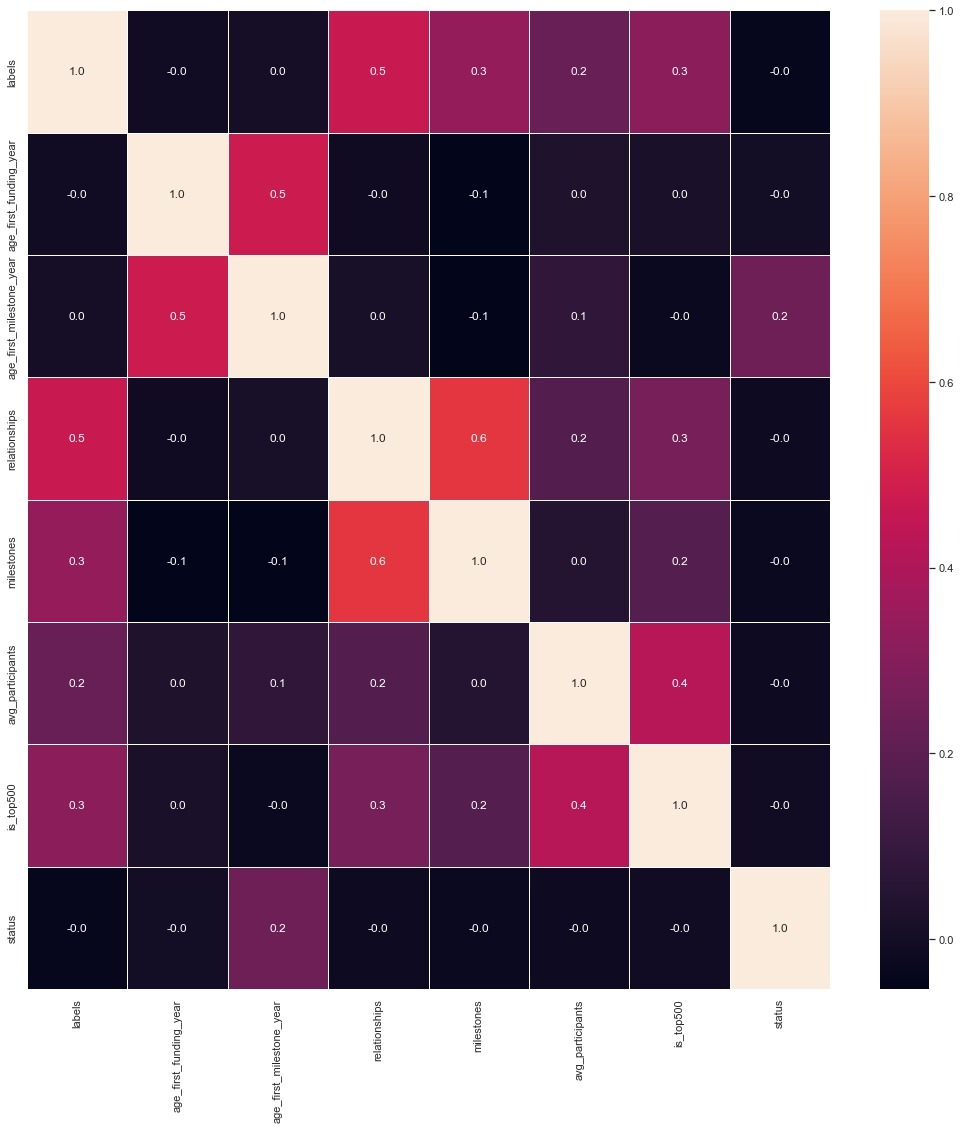


Fig.8 heatmap showing correlations between features

A heatmap is generated using 'sns.heatmap' by passing the correlation matrix as the data. Fig.8 is the visualization of the correlation between numerical features in a dataset using a heatmap, providing insights into the relationships between these variables.

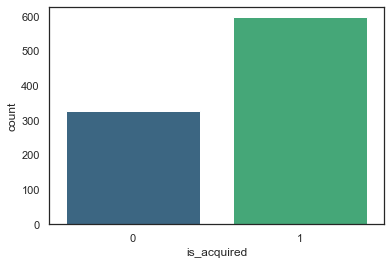


Fig.9 Countplots of is\_acquired

Fig.9 is a countplot is a type of bar plot that shows the counts of observations in each category. is\_acquired is shown on x-axis and the total count is shown on the y-axis. The 'palette' parameter is set to 'viridis' to define the color scheme of the countplot.

The countplot generated shows the distribution of the 'is\_acquired' values in the dataset, providing insights into the frequency or imbalance of the 'is\_acquired' category.

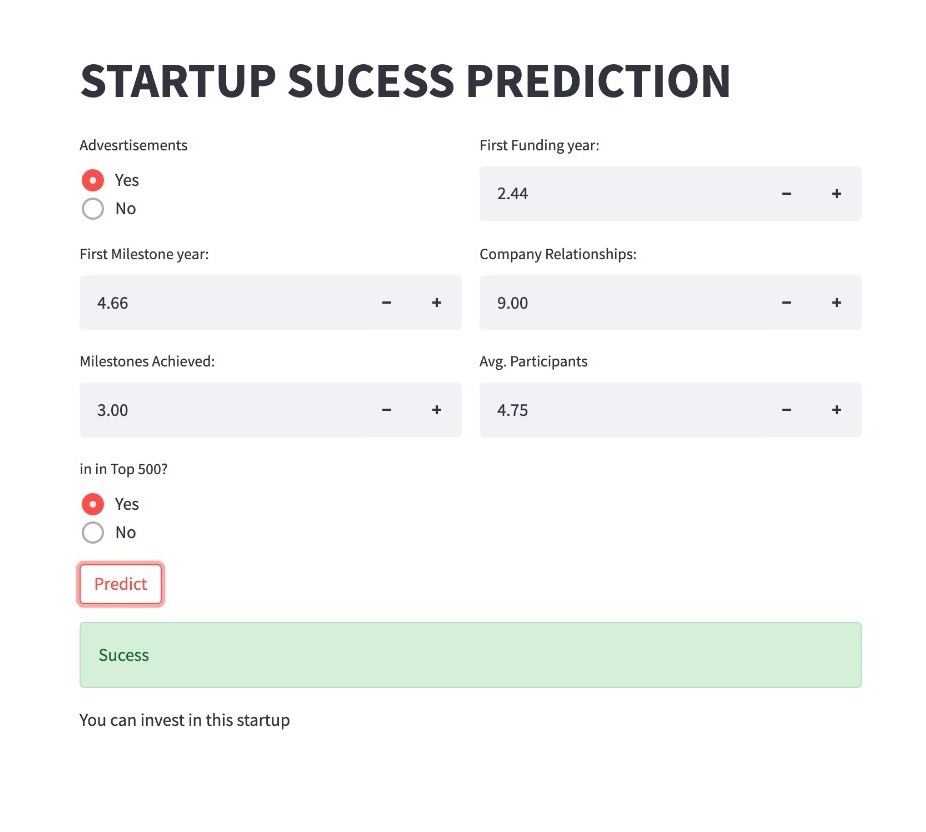


Fig.10 User Interface showing success of a startup

This is the implementation for startup success prediction using a DecisionTreeClassifier as a saved model. Fig.10, Fig.11 shows the user interface where the user can input various parameters related to a startup, and the model will predict whether the startup will be successful or not.

The user interface allows the user to input values for features such as advertisements, first funding year, first milestone year, company relationships, milestones achieved, average participants, and whether the startup is in the top 500.

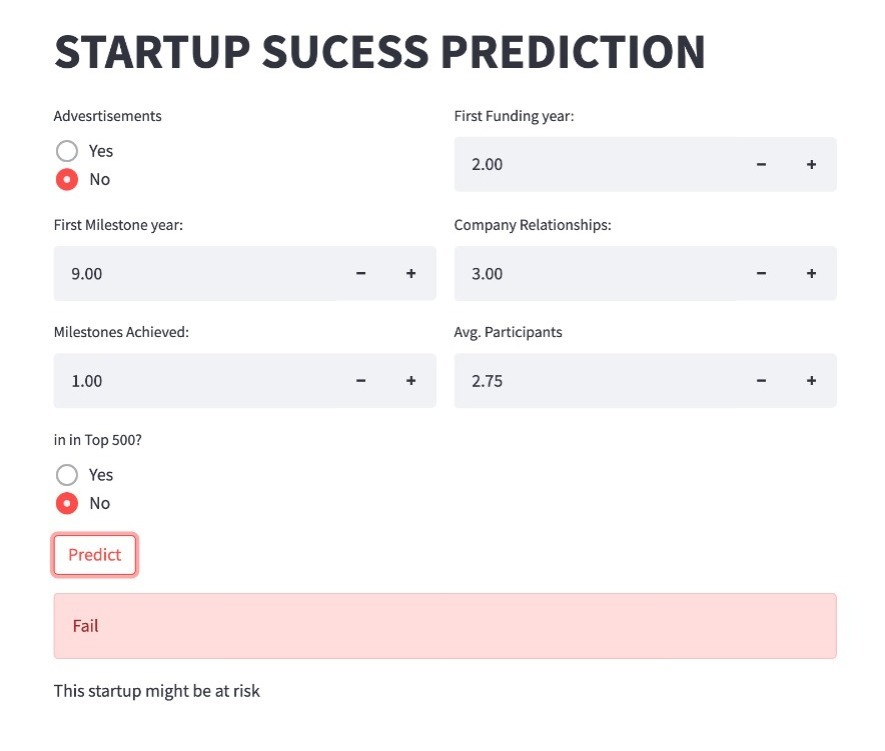


Fig.11 User Interface displaying the risk of a startup

After the user enters the input values and clicks on the "Predict" button, the code prepares the input data by creating a panda DataFrame and populating it with the user inputs. The labels and is\_top500 values are converted to numerical representations based on predefined dictionaries.

Next, the input data is normalized by applying a min-max scaling technique to ensure that the values fall within a specific range. The DecisionTreeClassifier model, loaded from the 'model.pkl' file, is then used to predict the success of the startup based on the input data. The predicted result is stored in the variable y\_pred. Here pkl indicates it is a pickle file.

Finally, based on the predicted outcome, the output displays a success message and a recommendation to invest in the startup if the prediction is positive (y\_pred = 1) as shown in Fig.10. Otherwise, it shows a failure message and indicates that the startup might be at risk in Fig.11.

This UI is a simple and interactive way for users to get predictions on the success of a startup based on relevant features. The logistic regression model trained on a specific dataset enables the prediction process, helping users make informed decisions regarding investment opportunities.

# CONCLUSION

In conclusion, this research paper highlights the significance of utilizing machine learning algorithms for the prediction of success among currently operated startups. The integration of various features such as company ranking, relationships, funding\_year, milestones, first\_milestone\_year, average participants, and advertising status allows for a comprehensive analysis of the factors influencing startup success. The application of machine learning algorithms enables investors, stakeholders, and other interested parties to make informed decisions regarding investment opportunities. By accurately predicting the potential success of startups, investors can allocate their resources strategically, minimizing risks and maximizing returns. Additionally, this research provides valuable insights to startup owners, offering an opportunity to identify areas of improvement and enhance the likelihood of success. Overall, the use of machine learning algorithms in predicting startup success serves as a valuable tool for all stakeholders, facilitating informed decision-making and promoting the growth and success of startups in a competitive business landscape.

# Acknowledgments

We would like to express our gratitude to SR Engineering College and SR University, located in Warangal, Telangana, for their invaluable assistance and support during the course of this project.

# References

1. Jiang, Z., Li, X., & Chen, Y. (2017). "A hybrid model for startup success prediction using machine learning algorithms." In 2017 14th Web Information Systems and Applications Conference (WISA) (pp. 56-61). IEEE.
2. Tiwari, S., Jain, R., & Rishi, M. (2016). "A comparative study of machine learning techniques for startup success prediction." In 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT) (pp. 2297-2301). IEEE.
3. Jain, R., & Tiwari, S. (2016). "Predicting startup success using machine learning techniques." In 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE) (pp. 1-6). IEEE.
4. Ushus, A. M., & Joseph, S. (2015). "Startup success prediction using machine learning." In 2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS) (pp. 1-5). IEEE.
5. Zhang, J., Li, J., & Li, B. (2020). "A comparative study of machine learning algorithms for startup success prediction." In 2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA) (pp. 1083-1088). IEEE.
6. Selvaraj, P. R., & Rajagopalan, S. (2019). "Prediction of startup success using machine learning techniques." In 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU) (pp. 1-6). IEEE.
7. Shah, K., & Shah, K. (2020). "Startup success prediction using ensemble learning techniques." In 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS) (pp. 1283-1288). IEEE.
8. Singh, P., & Sharma, A. (2020). "Ensemble methods for startup success prediction." In 2020 International Conference on Computer, Communication, and Signal Processing (ICCCSP) (pp. 381-385). IEEE.
9. Sridhar, V., & Ravishankar, M. (2021). "Predicting startup success using machine learning algorithms: A systematic review and meta-analysis." Journal of Business Research, 135, 478-491.
10. Ahuja, A., & Singh, P. (2017). "Predicting startup success using machine learning techniques." In 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT) (pp. 16-21). IEEE.
11. Kiran, M. S., & Srinivasa Rao, A. (2020). "Predicting startup success using ensemble methods and feature importance analysis." Journal of Ambient Intelligence and Humanized Computing, 11(9), 3847-3856.
12. Amini, M., & Shams, A. (2018). "Transfer learning for startup success prediction using deep neural networks." In 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 558-561). IEEE.
13. Yildirim, O., & Ayan, M. (2017). "Startup success prediction using kernelized support vector machines with RBF kernel." In 2017 International Artificial Intelligence and Data Processing Symposium (IDAP) (pp. 1-5). IEEE.
14. Kaur, S., & Bhatia, R. (2019). "Prediction of startup success using polynomial kernel-based support vector machine." In 2019 International Conference on Sustainable Computing and Intelligent Systems (ICSCIS) (pp. 281-285). IEEE.
15. Sabah, M. H., & Al-Atabbi, A. A. (2020). "Startup success prediction using optimized machine learning algorithms." In 2020 International Conference on Advanced Machine Learning Technologies and Applications (AMLTA) (pp. 207-216). Springer.
16. Singh, P., & Ahuja, A. (2018). "Predicting startup success using hyperparameter optimization of machine learning algorithms." In 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-6). IEEE.
17. Bhattacharya, S., & Ray, S. (2021). "Startup success prediction using oversampling and machine learning techniques." In 2021 3rd International Conference on Innovative Technologies in Engineering (ICITE) (pp. 1-6). IEEE.
18. Gao, Y., Zhang, X., & Guo, L. (2018). "Extracting meaningful patterns from complex data for startup success prediction." In 2018 IEEE International Conference on Big Data (Big Data) (pp. 4376-4381). IEEE.
19. Rizwan, M., Aslam, N., & Anwar, S. M. (2019). "A comprehensive analysis of factors affecting startup success using machine learning techniques." IEEE Access, 7, 155816-155828.
20. Adnan, M., El Saddik, A., & Saeed, A. (2018). "Predicting startup success using machine learning: A multi-factor analysis." In 2018 IEEE International Conference on Big Data (Big Data) (pp. 3903-3910). IEEE