**STARTUP ALCHEMY:**

Turning Data into Success Spells

**PROJECT REPORT**

***Submitted by***

**AYUSH BANSAL (2210993778)**

**MANYA SAINI (2210993814)**

**VIRENDER GARG (2210993861)**

**Logo

Description automatically generated**

**BE-CSE (Artificial Intelligence)**

***Guided by***

**Dr. KANIKA**

**Dr. TANVI SOOD**

**CHITKARA UNIVERSITY INSTITUTE OF ENGINEERING & TECHNOLOGY**

**CHITKARA UNIVERSITY, RAJPURA**

**NOVEMBER, 2023**

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **Topics** | **Page No.** |
| 1. Acknowledgements | 2 |
| 2. Abstract | 3 |
| 1. List of Figures | 4 |
| 1. Chapters:    1. 4.1 Introduction | 5 |
| 4.2. Problem Formulation | 6 |
| 4.3. Proposed Solution | 7 |
| 4.4. Flowchart | 12 |
| 4.5. Software and Hardware Requirements | 13 |
| 4.6. Dataset | 16 |
| 4.7. Code | 19 |
| 4.8. Results  4.9. Conclusion | 39  48 |
| 5. References | 49 |

1. **ACKNOWLEDGEMENTS**

With immense pleasure, we, **Ayush Bansal, Manya Saini** and **Virender Garg** are presenting the “**STARTUP ALCHEMY: Turning Data into Success Spells**” project report as a part of the curriculum of ‘BE-CSE(AI)’.

We would like to express our sincere thanks to “**Dr. Kanika**” and “**Dr. Tanvi Sood**”, for their valuable guidance and support in completing our project.

We would also like to express our gratitude towards our Dean “**Dr. Sushil Kumar Narang**”, for giving us the opportunity to do a project on Startup Success Prediction: STARTUP ALCHEMY. Without their support and suggestions, this project would not have been completed.

Signature :………….. Signature :………….. Signature :…………..

Name: Ayush Bansal Name: Manya Saini Name: Virender Garg

Roll No.: 2210993778 Roll No.: 2210993814 Roll No.: 2210993861

1. **ABSTRACT**

In response to the dynamic and ever-evolving landscape of startups, there is an increasing need for innovative approaches that can accurately predict success and provide strategic guidance. Our project, titled "Startup Alchemy: Turning Data into Success Spells," represents a concerted effort to harness the synergies between data and artificial intelligence (AI) to forecast the success of startups. With a special emphasis on the vibrant domain of AI startups in healthcare, we embark on a multifaceted journey. Through the concurrent training of three distinct models and the adept use of advanced visualization techniques, our project aspires to furnish stakeholders with a comprehensive and nuanced understanding of the myriad factors that influence startup success.

At its core, our project revolves around the meticulous curation of a tailored dataset, meticulously capturing pivotal features that wield significant influence over the trajectory of startups. Employing sophisticated machine learning models for predictive analytics, we delve into the dataset to unravel intricate patterns and discern the hidden determinants of startup success. The outcomes of these models undergo not only robust analytical scrutiny but also find expression through compelling visualizations crafted using the powerful Matplotlib library in Python. This synergistic approach ensures a holistic interpretation of results, effectively demystifying the complexities inherent in the landscape of startup success.

As we navigate through the intricacies of the startup ecosystem, our project seeks to empower entrepreneurs, investors, and decision-makers with actionable insights. These insights, derived from cutting-edge AI and data analytics, are poised to act as compasses, guiding stakeholders in shaping the future trajectory of AI startups in healthcare. The subsequent sections of this report will delve into the methodologies, results, and implications of our project, providing a comprehensive roadmap for stakeholders seeking to navigate the dynamic and often unpredictable terrain of startup success.

1. **LIST OF FIGURES**

**Figure Roll No. Page no.**

1. Basic Flow Chart of the Project 4.4.1 12

2. Dataset.xlsx(1) 4.6.1 16

3. Dataset.xlsx(2) 4.6.2 16

4. Dataset.xlsx(3) 4.6.3 17

5. funding-data.csv 4.6.4 17

6. states.csv 4.6.5 18

7. Correlation between success and 4.8.1 39 other factors affecting the startup

8. Sankey Plot 1 4.8.2 40

9. Sankey Plot 2 4.8.3 40

10. Sankey Plot 3 4.8.4 41

11. Scatter Plot 4.8.5 41

12. Web Page 4.8.6 42

13. Model 1 4.8.7 42

14. Model 1 4.8.8 43

15. Model 1 4.8.9 43

16. ROC Curve 1 4.8.10 44

17. ROC Curve 2 4.8.11 45

18. ROC Curve 3 4.8.12 45

19. Geospatial Visualization 1 4.8.13 46

20. Geospatial Visualization 2 4.8.14 47

21. Geospatial Visualization 3 4.8.15 47

**4.1 INTRODUCTION**

In a landscape characterized by innovation, risk, and transformative potential, the success of startups remains a pursuit that intrigues and challenges stakeholders across industries. "Startup Alchemy" represents our endeavor to unravel the complexities of startup success prediction, a task made even more intricate when applied to the specialized domain of AI startups in healthcare.

Dataset Creation and Model Training: At the core of our project is the creation of a bespoke dataset, meticulously curated to encapsulate crucial features that influence startup success. We leverage machine learning, employing three simultaneous models – Gradient Boosting, Random Forest, and Decision Tree – to unravel patterns and correlations within this data. The distinctiveness of our approach lies in its applicability to AI startups in healthcare, a sector that demands a specialized lens.

Visualization for Comprehensive Interpretation: To ensure the accessibility and interpretability of our predictive models, we turn to advanced visualization techniques. Platforms like Matplotlib in Python serve as our canvas, enabling stakeholders to comprehend and explore the outcomes comprehensively. The visual representation of our findings is instrumental in translating complex machine learning outcomes into actionable insights.

Focus on AI Startups in Healthcare: Recognizing the dynamic nature of the healthcare industry and the transformative potential of AI, our project narrows its focus to startups operating at this intersection. By doing so, we acknowledge the unique challenges and opportunities that define success in this niche, tailoring our predictive models to the specific demands of the healthcare landscape.

Through "Startup Alchemy," we aim to empower decision-makers with a tool that transcends traditional predictive models, offering a nuanced understanding of success factors in the realm of AI startups in healthcare. The subsequent sections delve into the methodologies, results, and implications of our project, outlining a roadmap for stakeholders to navigate the dynamic and often unpredictable terrain of startup success.

**4.2 PROBLEM FORMULATION**

In the ever-evolving landscape of startups, predicting success remains an elusive pursuit. The absence of a reliable and comprehensive predictive model hampers the ability of entrepreneurs, investors, and stakeholders to make informed decisions. Traditional methods often fall short in capturing the intricate dynamics that influence a startup's trajectory, particularly within the specialized domain of artificial intelligence (AI) startups in healthcare.

This project addresses the critical gap in predictive tools tailored for the unique challenges faced by AI startups in healthcare. The absence of a nuanced, multi-model approach limits the ability to discern patterns, correlations, and key factors that contribute to success in this specific niche. Without a comprehensive understanding of these factors, stakeholders risk making decisions based on incomplete information, leading to suboptimal outcomes.

The overarching problem, therefore, is the lack of a robust predictive framework that integrates advanced machine learning models and data visualization techniques to offer actionable insights into the success potential of AI startups in healthcare. "Startup Alchemy" seeks to bridge this gap, providing a solution that empowers stakeholders with a predictive tool tailored to the intricacies of the healthcare-focused AI startup ecosystem.

By formulating and implementing this innovative project, we aim to empower decision-makers with a reliable and interpretable solution, facilitating better strategic planning, investment decisions, and ultimately contributing to the success and sustainability of AI startups in the healthcare domain.

**4.3 PROPOSED SOLUTION**

The project encompasses a range of features designed to address the challenges faced by startup entrepreneurs. It provides a comprehensive solution tailored to overcome the complexities inherent in predicting the success of startups:

### 1. Research and Project Selection:

### The inception of "Startup Alchemy: Turning Data into Success Spells" was grounded in a comprehensive exploration of the contemporary startup landscape. Extensive research identified the need for innovative approaches to predict startup success, particularly within the burgeoning field of AI startups in healthcare. The project aims to bridge the gap between data-driven insights and strategic decision-making.

### Benefits:

* *Informed Decision-Making*: Enables stakeholders to make strategic decisions based on predictive analytics.
* *Targeted Focus*: Specifically addresses the challenges and opportunities within the AI startup ecosystem in healthcare.

**2. Creating Bespoke Dataset**

A critical foundation of our project lies in the creation of a bespoke dataset, meticulously curated to encapsulate the essential features influencing startup success. This process involves the extraction and compilation of relevant data points that are crucial for training robust machine learning models.

**Benefits:**

* *Tailored Insights*: The dataset is customized to capture nuances relevant to the success of startups.
* *Model Training*: Provides a robust foundation for training machine learning models.

### 3. Visualization for Selecting Factors

### Leveraging the capabilities of Python's Matplotlib, our project employs a repertoire of sophisticated visualization techniques to discern and select key factors influencing startup success. The following visualization methods play a pivotal role in this exploratory process:

### Correlation Visualization

### Utilizing correlation matrices, we visually represent the relationships between different variables in the dataset. This technique enables a nuanced understanding of how factors coalesce or diverge, guiding the identification of correlated features.

### Benefits:

* **In-Depth Correlation Analysis:** Provides a comprehensive view of inter-variable relationships.
* **Multivariate Insight:** Identifies patterns and dependencies among multiple factors.

1. **Sankey Plot**

The Sankey plot offers an intuitive and interactive way to visualize the flow of resources or, in our case, the impact of different factors on startup success. It helps unveil the intricate pathways that contribute to favorable or unfavorable outcomes.

**Benefits:**

* **Flow Visualization:** Illustrates the contribution and distribution of various factors.
* **Pathway Identification:** Highlights the predominant pathways influencing startup success

.

1. **Scatter Plot**

Scatter plots serve as a powerful tool for visualizing the relationship between two continuous variables. In our context, scatter plots aid in uncovering trends, clusters, or outliers, providing valuable insights into the distribution of influential features.

**Benefits:**

* **Variable Interaction:** Depicts how pairs of variables interact with each other.
* **Outlier Detection:** Identifies anomalies that might impact startup success predictions.

#### Benefits of Visualization Techniques:

* ***Intuitive Exploration*:**
  + The combination of correlation matrices, Sankey plots, and scatter plots offers an intuitive exploration of complex data relationships.
  + Stakeholders can interactively navigate visual representations for a deeper understanding.
* ***Informed Feature Selection*:**
  + Visualization serves as a guide in selecting influential features for model training.
  + Clear visualizations empower stakeholders to make informed decisions on the inclusion or exclusion of specific variables.

Incorporating these diverse visualization techniques ensures a robust and nuanced exploration of the dataset, enhancing the clarity and efficacy of feature selection for subsequent model training stages.

1. **Dividing Dataset into Train and Test Data**

The dataset is systematically divided into training and testing subsets, ensuring the models are trained on a representative sample and evaluated on an independent dataset. This step is crucial for assessing the generalization capabilities of the models.

**Benefits:**

* *Model Evaluation*: Enables the assessment of model performance on unseen data.
* *Generalization*: Enhances the ability of models to make accurate predictions on new startup data.

**5. Training Three Models**

Our project employs three distinct machine learning models—Gradient Boosting, Random Forest, and Decision Tree Classifier. Each model brings unique strengths to the predictive analytics framework, enhancing the robustness and reliability of the overall system.

* + 1. **Gradient Boosting:**
* **Explanation:** Gradient Boosting is an ensemble learning method that builds a series of weak learners (typically decision trees) sequentially. Each subsequent model corrects the errors of the previous ones, leading to a strong predictive model.
* **Benefits:** It is highly effective in capturing complex relationships in the data and minimizing prediction errors.
  + 1. **Random Forest:**
* **Explanation:** Random Forest is an ensemble learning method that constructs a multitude of decision trees during training and outputs the mode of the classes (classification) or the mean prediction (regression) of the individual trees.
* **Benefits:** It excels in handling a large number of features and provides robust predictions by reducing overfitting.

**3.** **Decision Tree Classifier:**

* **Explanation:** A Decision Tree Classifier is a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.
* **Benefits:** Decision trees are intuitive, easy to interpret, and can capture both linear and non-linear relationships in the data

**Benefits of Model Training:**

* *Model Diversity*: Incorporates diverse models for a comprehensive understanding of startup prediction.
* *Ensemble Learning*: Enhances predictive accuracy through the combination of multiple models.

**6. Results Presentation through Web Page**

A user-friendly web page is designed to present the results, allowing stakeholders to interactively explore and select the desired model for predictions. This interactive platform enhances accessibility and usability.

**Benefits:**

* *User Interaction*: Enables stakeholders to interactively explore and choose the desired predictive model.
* *Accessibility*: Enhances the accessibility of predictive analytics outcomes.

**7. Displaying ROC Curve after Model Training**

The ROC (Receiver Operating Characteristic) curve is a powerful tool for evaluating and comparing the performance of classification models. After training each model, the ROC curve is displayed to provide a visual representation of its discriminative capabilities.

**Benefits:**

* *Performance Evaluation*: Offers a visual assessment of the model's ability to distinguish between classes.
* *Comparative Analysis*: Facilitates the comparison of performance across different models.

**8. Training Model and Visualizing Results on USA Map Projection**

The project extends beyond traditional visualizations by integrating geographical insights. The results of the trained models are visualized on a map of the USA, where each state is shaded based on the number of successful and unsuccessful startups. This spatial representation provides a unique perspective on regional startup dynamics.

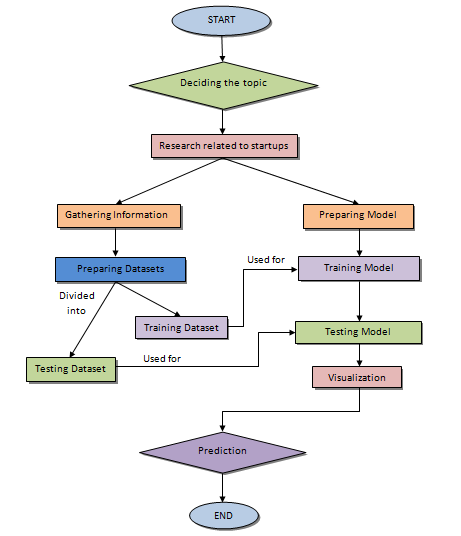
**Benefits:**

* Geographical Context: Incorporates geographical insights into startup success prediction.
* Regional Analysis: Enables a nuanced understanding of startup outcomes across different states.

Each of these features contributes a vital component to the overarching goal of predicting startup success, offering a holistic and innovative approach to leveraging data and AI in the dynamic realm of startups, particularly in the context of AI startups in healthcare.

**4.4 FLOW CHART**

The Flow Chart of the project is given below:

****

*Figure 4.4.1 Basic Flow Chart of the project*

**4.5 SOFTWARE AND HARDWARE REQUIREMENTS**

**Software Requirements**

**1. Programming Language:**

* **Python (Version 3.11.1):** Python serves as the core programming language for the project. It's widely used for data analysis, machine learning, and web development.

### 2. JavaScript:

Javascript was mainly used along with other web development tools for making our website user friendly. It helped in administering the responsiveness of our website and also as the main tool for creating visualizations and interactive charts and maps, which are the key features of our projects.

**3. Database Management:**

* **Excel (Microsoft Excel):** Microsoft Excel serves as the primary platform for the creation and storage of the project's database. It provides a familiar interface for data manipulation, analysis, and easy collaboration.

**4. Libraries and Frameworks:**

* **scikit-learn :** Scikit-learn is a machine learning library in Python. It provides simple tools for data mining and data analysis, making it a go-to choice for predictive modeling.
* **pandas :** Pandas is a data manipulation and analysis library. It's used for cleaning, transforming, and analyzing the dataset.
* **numpy :** NumPy is a fundamental package for scientific computing with Python. It provides support for large, multi-dimensional arrays and matrices, along with mathematical functions.
* **Flask :** Flask is a lightweight web framework in Python. It's used for developing the web application, especially for creating the API endpoints.
* **d3.js :** D3.js is a JavaScript library for producing dynamic, interactive data visualizations in web browsers. It's often used for creating interactive charts and maps.
* **Matplotlib :** Matplotlib is a 2D plotting library for Python. It's used for creating static, animated, and interactive visualizations in Python.

**5. Web Development Tools :**

* **HTML, CSS, JavaScript:** These are the standard technologies for building web pages and adding interactivity.
* **Flask (Web Framework):** Flask serves as the web framework for developing the backend of the web application. It was used mainly for the representation of the ROC Curve and the Map displayed on our website.

**6. Integrated Development Environment (IDE):**

* **Jupyter Notebook :** Jupyter Notebook is utilized specifically for visualization and factor selection tasks, providing an interactive computing environment suitable for data exploration and visualization.
* **Visual Studio Code (Version 1.84):** Visual Studio Code (VS Code) serves as the primary integrated development environment for coding, debugging, and version control. It provides a lightweight yet powerful environment for web development. Python visualization and many other tasks.

**7. Cross-Browser Compatibility:**

* **Ensure cross-browser compatibility (e.g., Chrome, Firefox, Safari, Edge):** The project works seamlessly across various web browsers.

## Hardware Requirements

### The hardware requirements for the project are relatively modest. A standard computer or laptop with sufficient processing power, memory, and storage capacity is suitable for development purposes.

### 1. Processor:

* **Multi-core processor (Recommended: Quad-core or higher):** A multi-core processor enhances the speed and efficiency of computations, crucial for machine learning tasks.

### 2. RAM:

* **Minimum of 8 GB RAM (Recommended: 12 GB or higher for large datasets):** Sufficient RAM is essential for handling large datasets and running machine learning models efficiently.

### 3. Storage:

* **Minimum of 50 GB free disk space for datasets and model storage:** Adequate storage space is required for storing datasets, models, and other project-related files.

### 4. Operating System:

* **Compatible with Windows, macOS, or Linux:** The project should be adaptable to various operating systems commonly used by developers.

### 5. Internet Connectivity:

* **Required for downloading libraries, datasets, and potential updates:** A stable internet connection is necessary for downloading dependencies, datasets, and any updates needed during the development process.

These requirements collectively form the foundation for the successful development and execution of your startup prediction project, empowering the decision-makers with a tool that transcends traditional predictive models, offering a nuanced understanding of success factors.

**4.6 DATASET**

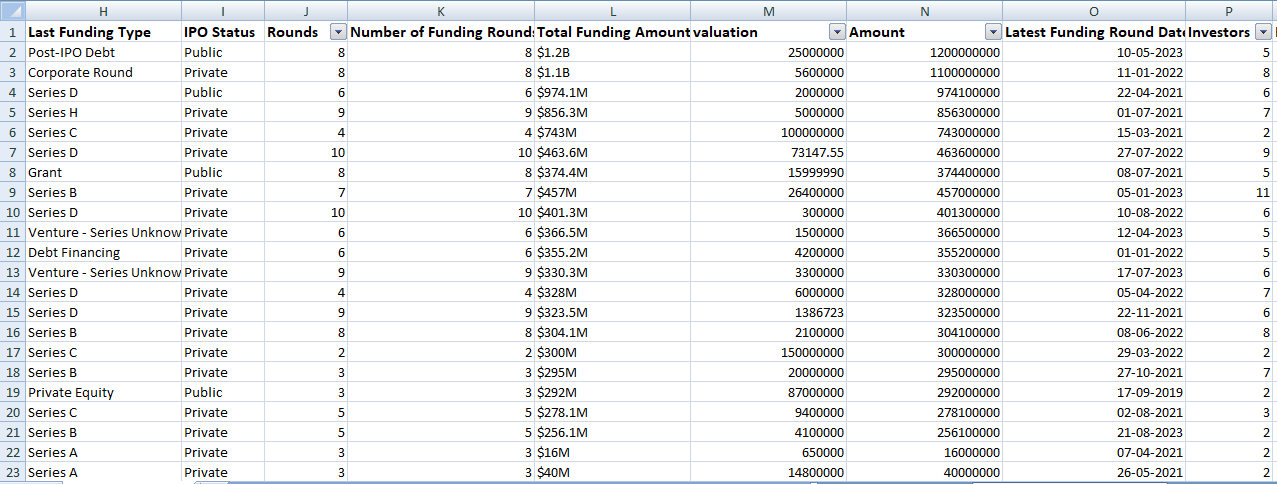
In crafting our dataset, we meticulously curated information from reputable sources such as Crunchbase, Kaggle, and Wikipedia, amalgamating a rich and diverse collection of data to form the backbone of our predictive analytics framework.

**- Dataset.xlsx**

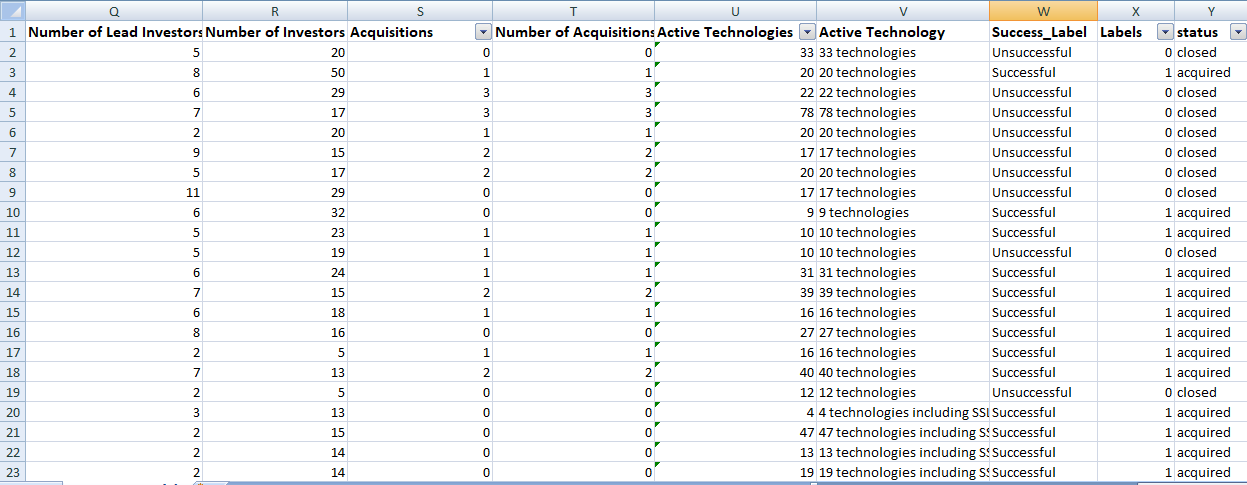
The Dataset Excel sheet is the main data sheet including all the different details of different AI healthcare startups, such as their, Startup Name, Location, Founding Year, Description, Number of Employees, Funding Type, IPO Status, Number of Funding Rounds, Total Funding, Valuation, Last Funding Round Date, Number of Investors, Number of Lead Investors, Number of Acquisitions, Number of Active Technologies and their Success Label.

****

*Figure 4.6.1 Dataset.xlsx (1)*

****

*Figure 4.6.2 Dataset.xlsx (2)*

****

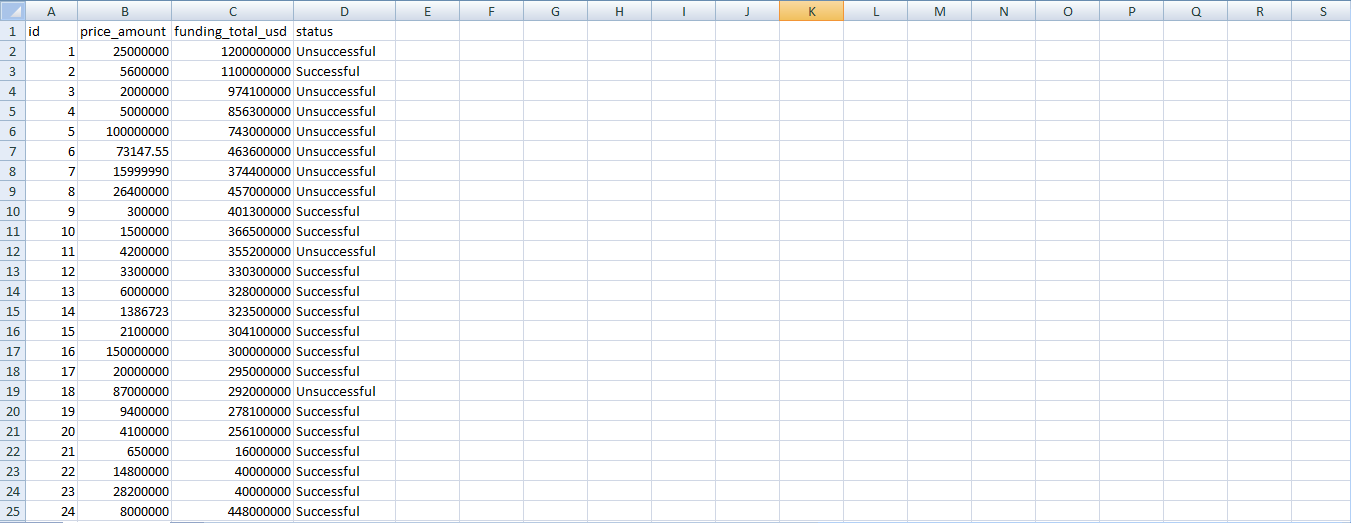
*Figure 4.6.3 Dataset.xlsx (3)*

The main data was broken into two halves one for training the models and the other for testing the models:

**- train-data.csv**

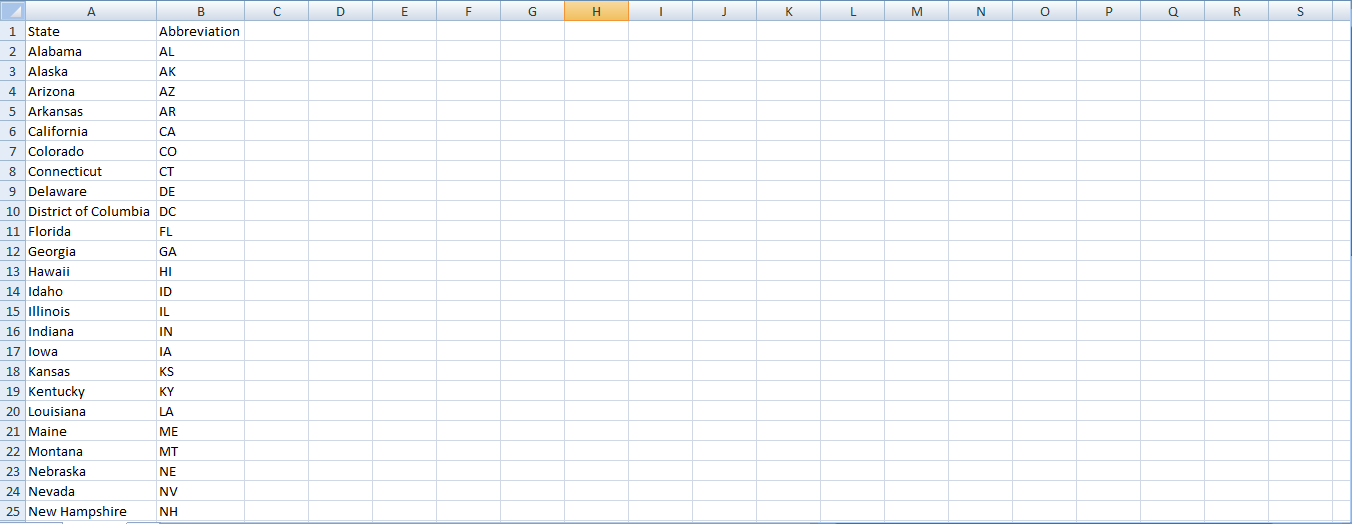
**- test-data.xlsx**

Another Data sheet comprises of the funding data, involving the valuation amount and the total funding amount for finding a relation between these two:



*Figure 4.6.4 funding-data.csv*

Another csv file contained all the US States along with their Abbreviations so that the model can be easily trained and we could predict success of the startups based on geographical area.



*Figure 4.6.5 states.csv*

Lastly there was a JSON file that contained the coordinates of all the states of US and was essentially used by javascript to project the map of US during prediction visualization.

This meticulously self-curated dataset serves as the foundation for our project, reflecting a comprehensive amalgamation of information meticulously gathered from reputable sources. Its creation underscores our commitment to crafting a robust and reliable predictive analytics framework tailored to the intricacies of startup success.

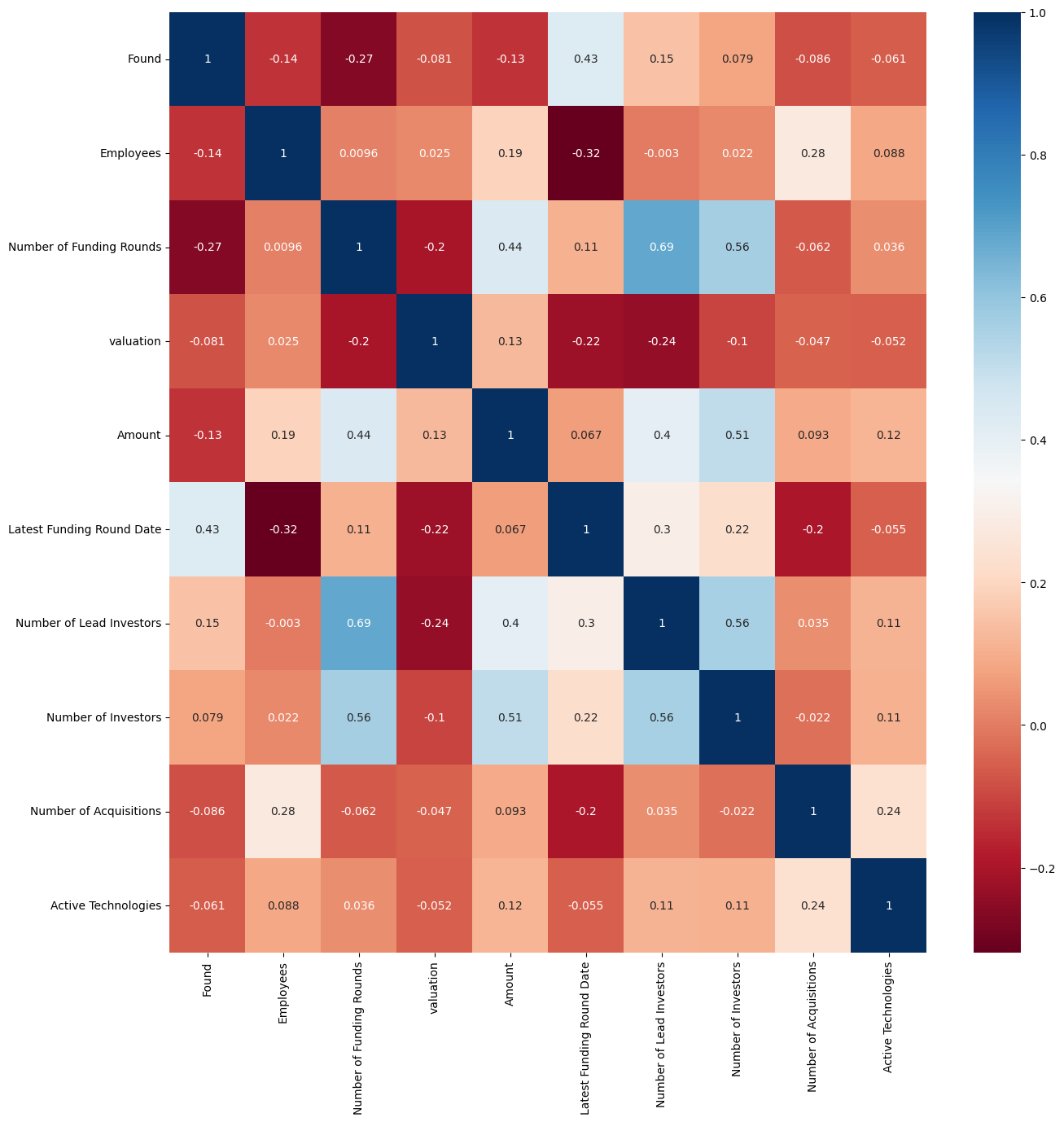
* 1. **CODE**

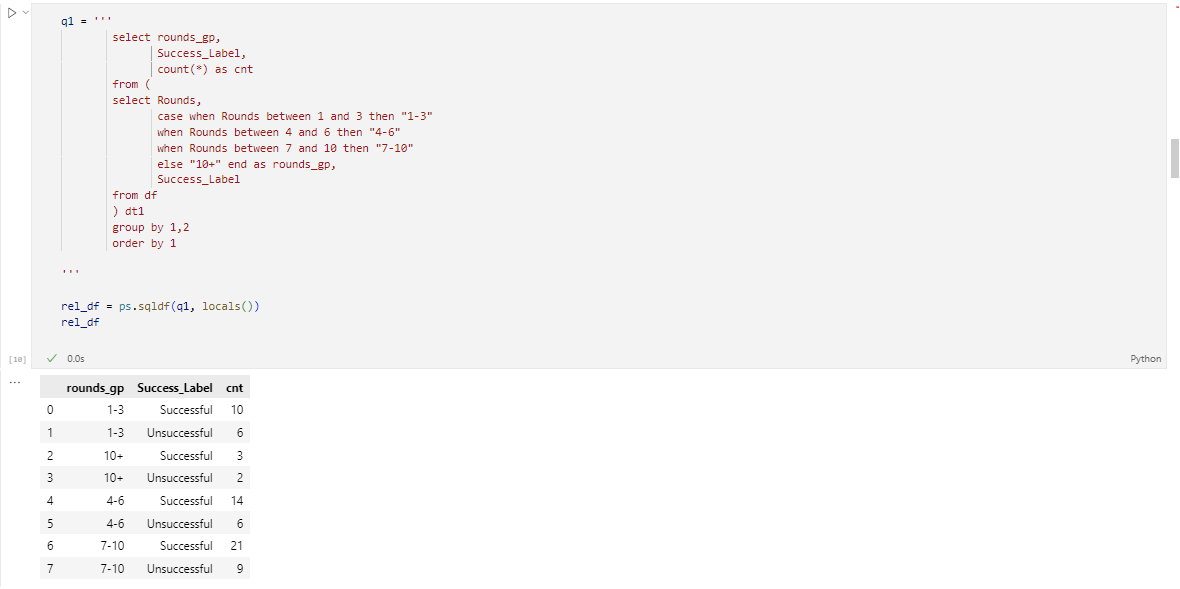
Following are the key snippets of code that form the backbone of our startup success prediction system. Each code snippet is carefully annotated to elucidate its role and significance in the overall framework, offering insights into the intricacies of model training, data visualization, and predictive analytics.

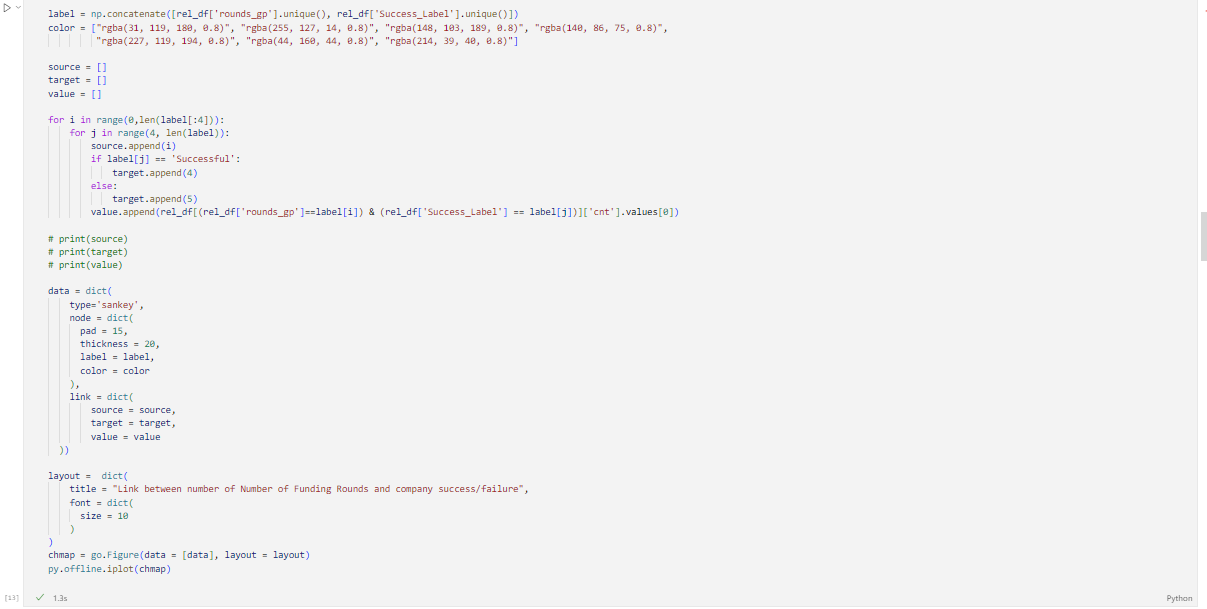
**1. Visualization for selecting factors.**

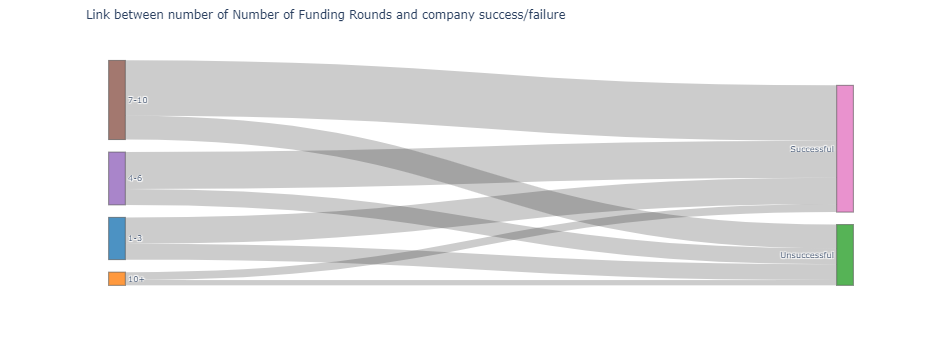
The following is code from python Notebook that showcase various visualizations for selecting various key factors that affect the success of a startup.





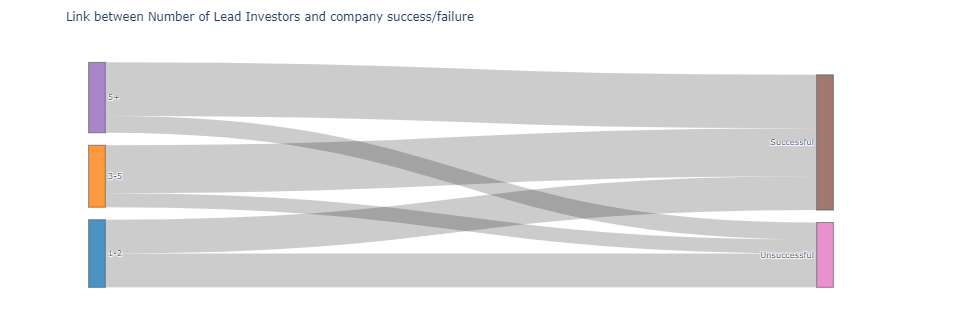






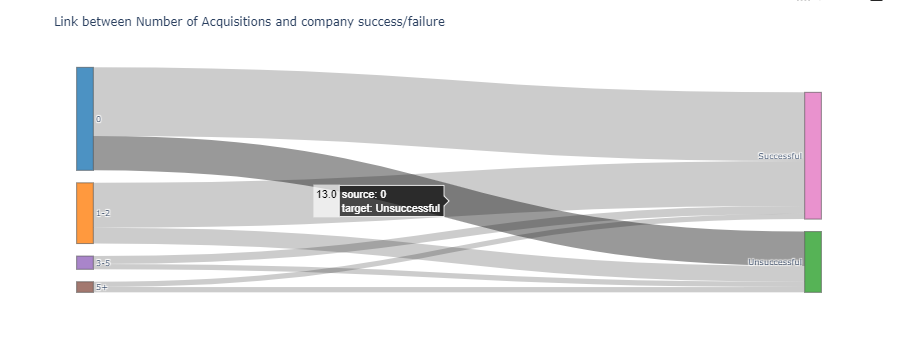




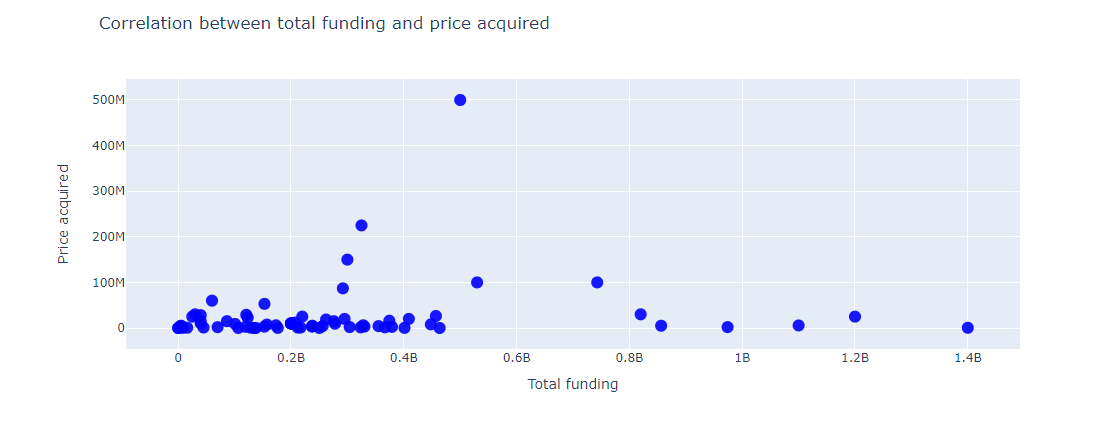












The code meticulously selects key features, including the number of funding rounds, the number of lead investors, and the number of acquisitions, exemplifying their impact on startup outcomes. The visualizations include insightful Sankey plots that unravel the intricate relationships between these variables. Additionally, a correlation scatter plot between the price acquired and the total funding amount provides a nuanced understanding of the financial dynamics.

**2. Training of Models**

The following python code trains 3 different models- Gradient Boosting, Decision Tree and Random Forest, simultaneously, upon the data being loaded using Flask Cors that fulfils the server domain request from the designed web page.

import **pandas** as **pd**

import **numpy** as **np**

from **sklearn**.**metrics** import **roc\_curve**

from **sklearn** import **metrics**

from **flask\_cors** import **CORS**

import **simplejson** as **json**

from **flask** import **Flask**, **stream\_with\_context**

from **sklearn**.**model\_selection** import **train\_test\_split**

from **sklearn**.**ensemble** import **RandomForestClassifier**, **GradientBoostingClassifier**

from **io** import **StringIO**

from **werkzeug**.**datastructures** import **Headers**

from **werkzeug**.**wrappers** import **Response**

import **csv**

from **sklearn**.**tree** import **DecisionTreeClassifier**

app = **Flask**(\_\_name\_\_)

**CORS**(app)

**@app.route**('/roc\_curve/m=<m>/d=<d>/l=<l>/n=<n>/c=<c>', methods=['GET'])

def **roc\_curve**(m,d,l,n,c):

    if m == "model1":

        model = **GradientBoostingClassifier**(criterion = 'friedman\_mse', learning\_rate = **float**(c),

                                      loss = 'log\_loss', max\_depth = **int**(d),

                                      max\_features = 'log2', max\_leaf\_nodes = **int**(l),

                                      n\_estimators = **int**(n))

    elif m == "model2":

        model = **RandomForestClassifier**(criterion = 'entropy', max\_depth = **int**(d),

                                       max\_features = 'sqrt', max\_leaf\_nodes = **int**(l),

                                       n\_estimators = **int**(n))

    else:

        model = **DecisionTreeClassifier**(max\_depth = **int**(d),

                                       max\_leaf\_nodes = **int**(l))

**print**("Roc",m,d,l,n,c)

    classifier = model.**fit**(X\_train, y\_train)

**print**(model)

    y\_pred = classifier.**predict\_proba**(X\_test)

    fpr, tpr, thresholds = **metrics**.**roc\_curve**(y\_test,y\_pred[:,1],pos\_label=1)

    result=[]

    for i in **range**(**len**(fpr)):

        result.**append**({"fpr":fpr[i],"tpr":tpr[i]})

    return **json**.**dumps**(result)

**@app.route**('/viz/m=<m>/d=<d>/l=<l>/n=<n>/c=<c>', methods=['GET'])

def **viz**(m,d,l,n,c):

    if m == "model1":

        model = **GradientBoostingClassifier**(criterion = 'friedman\_mse', learning\_rate = **float**(c),

                                      loss = 'log\_loss', max\_depth = **int**(d),

                                      max\_features = 'log2', max\_leaf\_nodes = **int**(l),

                                      n\_estimators = **int**(n))

    elif m == "model2":

        model = **RandomForestClassifier**(criterion = 'entropy', max\_depth = **int**(d),

                                       max\_features = None, max\_leaf\_nodes = **int**(l),

                                       n\_estimators = **int**(n))

    else:

        model = **DecisionTreeClassifier**(max\_depth = **int**(d),

                                       max\_leaf\_nodes = **int**(l))

**print**("Viz",m,d,l,n,c)

    classifier = model.**fit**(X\_train, y\_train)

**print**(model)

    X\_new = test\_df[features].values

    test\_df['prob\_acquired'] = classifier.**predict\_proba**(X\_new)[:,1]

    test\_df['acquired'] = **np**.**where**(test\_df['prob\_acquired']**>=**0.5, 1, 0)

    test\_df['closed'] = **np**.**where**(test\_df['prob\_acquired']**<**0.5, 1, 0)

    q1 = **pd**.**merge**(states\_df, test\_df, left\_on = 'State', right\_on = 'state', how = 'left')

    ren\_col = {'prob\_acquired': 'mean\_prob\_companies\_acquired\_by\_state',

               'acquired': 'count\_prob\_companies\_acquired\_by\_state',

               'closed': 'count\_prob\_companies\_closed\_by\_state'}

    output\_df = q1.**groupby**('State').agg({'prob\_acquired': 'mean',

                             'acquired': 'sum',

                             'closed': 'sum'}).**rename**(columns = ren\_col)

    output\_df = output\_df.**reset\_index**()

    output\_df.loc[output\_df['mean\_prob\_companies\_acquired\_by\_state'].**isnull**(), 'mean\_prob\_companies\_acquired\_by\_state'] = 0

    output\_df.loc[output\_df['count\_prob\_companies\_acquired\_by\_state'].**isnull**(), 'count\_prob\_companies\_acquired\_by\_state'] = 0

    output\_df.loc[output\_df['count\_prob\_companies\_closed\_by\_state'].**isnull**(), 'count\_prob\_companies\_closed\_by\_state'] = 0

    def **generate**(output\_df):

        d = **StringIO**()

        w = **csv**.**writer**(d)

*#write header*

        w.**writerow**(**tuple**(output\_df.columns))

        yield d.**getvalue**()

        d.**seek**(0)

        d.**truncate**(0)

        for i in **range**(output\_df.shape[0]):

            w.**writerow**(**tuple**(output\_df.iloc[i].values))

            yield d.**getvalue**()

            d.**seek**(0)

            d.**truncate**(0)

*# add a filename*

    headers = **Headers**()

    headers.**set**('Content-Disposition', 'attachment', filename='log.csv')

*# stream the response as the data is generated*

    return **Response**(

**stream\_with\_context**(**generate**(output\_df)),

        mimetype='text/csv', headers=headers

    )

train\_df = **pd**.**read\_csv**('data/train-data.csv', encoding = 'utf-8')

test\_df = **pd**.**read\_excel**('data/test-data.xlsx')

states\_df = **pd**.**read\_csv**('data/states.csv', encoding = 'utf-8')

features = ['Founding Year', 'Employees', 'Rounds',

             'valuation', 'Amount', 'Investors', 'Number of Investors',

             'Acquisitions', 'Active Technologies',

             ]

data = train\_df[features + ['Labels']].values

X = data[:,:-1]

y = data[:,9]

X\_train, X\_test, y\_train, y\_test = **train\_test\_split**(X, y, test\_size = 0.38775, random\_state = 0)

app.**run**(debug = True)

After training the models, an API endpoint is exposed, that allows other applications or services to interact with the trained model. This can be done using Flask to create routes that handle incoming requests, process the input data, and return the model's predictions.



**3. Web Application**

**- HTML (index.html)**

<!DOCTYPE *html*>

<html *lang*="en">

<head>

  <meta *charset*="UTF-8">

  <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

  <title>Startup Success Prediction: Model Development and Analysis</title>

  <link *rel*="stylesheet" *href*="styles.css">

*<!-- External scripts -->*

  <script *src*="https://d3js.org/d3.v5.min.js"></script>

  <script *src*="https://cdnjs.cloudflare.com/ajax/libs/d3-legend/2.13.0/d3-legend.js"></script>

  <script *type*="text/javascript" *src*="./main.js"></script>

</head>

<body>

  <div *id*="tooltip" *class*="hidden">

    <p><span *id*="state"></span></p>

    <p><span *id*="acquired"></span></p>

    <p><span *id*="closed"></span></p>

    <p><span *id*="prob"></span></p>

  </div>

  <h1>STARTUP ALCHEMY: Turning Data into Success Spells</h1>

  <section>

*<!-- Model selection -->*

    <label *for*="modelname">Choose a model:</label>

    <select *id*="modelname" *onchange*="**chooseModel**()">

      <option *value*="model1">Gradient Boosting</option>

      <option *value*="model3">Decision Trees</option>

      <option *value*="model2">Random Forests</option>

    </select>

    <br><br>

*<!-- Parameter input fields -->*

    <table>

      <tr>

        <td>

          <div>

            <label *for*="depth">Max depth</label>

            <br><input *type*="number" *id*="depth"><br>

          </div>

        </td>

        <td>

          <div>

            <label *for*="leaf">Max leaf nodes</label>

            <br><input *type*="number" *id*="leaf"><br>

          </div>

        </td>

        <td>

          <div *id* = "estimators">

            <label *for*="estimators" >Number of estimators</label>

            <br><input *type*="number" *id*="estimator"><br>

          </div>

        </td>

        <td>

          <div *id*="lr">

            <label *for*="lr" >Learning rate</label>

            <br><input *type*="number" *id*="l"><br>

          </div>

        </td>

      </tr>

    </table>

    <br>

*<!-- Train and visualize buttons -->*

    <input *name*="updateButton" *type*="button" *id*="train" *value*="Train model and draw ROC Curve" *onclick*="**connectFlask**()">

    &nbsp;&nbsp;&nbsp;&nbsp;

    <input *name*="visualize" *type*="button" *id*="viz" *value*="Train model, predict and visualize on new data" *onclick*="**visualize**()">

    <hr>

*<!-- Chart container -->*

    <div *id*="chart1" *class*="chart-container"></div>

  </section>

</body>

</html>

**- CSS (styles.css)**

body {

  font-family: 'Helvetica Neue', sans-serif;

  margin: 0;

  padding: 0;

  background-image: **linear-gradient**(to right, #3e4095, **rgb**(53, 174, 211));

  background-size: cover;

  color: #fff;

}

h1 {

  text-align: center;

  font-size: 3rem;

  font-weight: 700;

  color: #fff;

  margin-top: 2rem;

  padding: 1rem;

  background-color: **rgba**(0, 0, 0, 0.5);

  border-radius: 10px;

}

section {

  padding: 2rem;

  box-shadow: 0px 2px 10px **rgba**(0, 0, 0, 0.3);

  background-color: **rgba**(255, 255, 255, 0.9);

  border-radius: 10px;

  color:#1a1818;

}

*/\* Model selection \*/*

#modelname {

  width: 200px;

  margin-right: 1rem;

  padding: 0.5rem;

  border: 1px solid #333;

  background-color: **rgba**(255, 255, 255, 0.9);

  border-radius: 5px;

}

*/\* Parameter input fields \*/*

input[type="number"] {

  width: 120px;

  margin-bottom: 1rem;

  padding: 0.5rem;

  border: 1px solid #333;

  background-color: **rgba**(255, 255, 255, 0.9);

  border-radius: 5px;

}

*/\* Train and visualize buttons \*/*

input[type="button"] {

  padding: 0.5rem 1rem;

  border: none;

  cursor: pointer;

  background-color: #333;

  color: #fff;

  border-radius: 5px;

}

input[type="button"]:hover {

  background-color: #555;

}

*/\* Charts \*/*

.chart-container {

  width: 90%;

  margin: 0 auto;

  background-color: **rgba**(255, 255, 255, 0.9);

  overflow: hidden;

  padding: 20px;

  border-radius: 10px;

  color: #333;

}

*/\* Tooltip \*/*

#tooltip {

  position: absolute;

  display: none;

  background-color: #333;

  color: #fff;

  padding: 10px;

  font-size: 14px;

  pointer-events: none;

  border-radius: 5px;

}

    th, td {

      padding: 10px;

      text-align: center;

      color:#333;

    }

    th {

      background-color: #555;

      color: #1a1818;

    }

**- JAVASCRIPT (main.js)**

function **chooseModel**(){

    var x = document.**getElementById**("modelname").value;

    if (x === "model2"){

        document.**getElementById**("lr").style.visibility="hidden";

        document.**getElementById**("estimators").style.visibility="visible";

    }

    if (x === "model3") {

        document.**getElementById**("lr").style.visibility="hidden";

        document.**getElementById**("estimators").style.visibility="hidden";

    }

    if (x === "model1") {

        document.**getElementById**("lr").style.visibility="visible";

        document.**getElementById**("estimators").style.visibility="visible";

    }

}

 function **connectFlask**() {

  var url = "http://127.0.0.1:5000/roc\_curve/"

  var modelname = document.**getElementById**("modelname").value;

  var max\_depth = document.**getElementById**("depth").value;

  var max\_leaf\_nodes = document.**getElementById**("leaf").value;

  var n\_estimators = document.**getElementById**("estimator").value;

  var learning\_rate = document.**getElementById**("l").value;

  if (learning\_rate==="") {

    learning\_rate = .1;

  }

  if (n\_estimators==="") {

    n\_estimators = 100;

  }

  if (max\_depth==="") {

    max\_depth = 5;

  }

  if (max\_leaf\_nodes==="") {

    max\_leaf\_nodes = 10;

  }

  url1 = url.**concat**("m=").**concat**(modelname).**concat**("/d=").**concat**(max\_depth).**concat**("/l=").**concat**(max\_leaf\_nodes).**concat**("/n=").**concat**(n\_estimators).**concat**("/c=").**concat**(learning\_rate);

  d3.**json**(url1).**then**(function(data) {

**drawRoc**(data);

  });

}

function **drawRoc**(data){

var w = 800;

var h = 450;

var padding = 50;

var xScale = d3.**scaleLinear**()

    .**domain**([0,1])

    .**range**([0, w/2])

var yScale = d3.**scaleLinear**()

    .**domain**([0,1])

    .**range**([h-h/10, 0]);

d3.**select**("#chart1")

    .**selectAll**("\*")

    .**remove**();

var svg = d3.**select**("#chart1")

    .**append**("svg")

    .**attr**("width", w + padding)

    .**attr**("height", h + padding)

    .**attr**("align","center-right")

    .**append**("g")

    .**attr**("transform", "translate(" + 275 + "," + padding + ")");

svg.**append**("g")

    .**attr**("class", "x axis")

    .**attr**("transform", "translate(0," + (h-h/10) + ")")

    .**call**(d3.**axisBottom**(xScale));

svg.**append**("g")

    .**attr**("class", "y axis")

    .**call**(d3.**axisLeft**(yScale));

var rocline = d3.**line**()

   .**x**(function(d) { return **xScale**(d.fpr)})

   .**y**(function(d) { return **yScale**(d.tpr)});

svg.**append**("path")

    .**datum**(data)

    .**attr**("class", "line")

    .**attr**("d", rocline);

svg.**append**("line")

    .**attr**("x1", **xScale**(0))

    .**attr**("y1", **yScale**(0))

    .**attr**("x2", **xScale**(1))

    .**attr**("y2", **yScale**(1))

    .**attr**("stroke-width", 2)

    .**attr**("stroke", "black")

    .**attr**("stroke-dasharray", "8,8");

svg.**append**("text")

    .**attr**("transform", "rotate(0)")

    .**attr**("y", (padding + 380))

    .**attr**("x",180)

    .**attr**("dy", "1em")

    .**text**("False Positive Rate ");

svg.**append**("text")

    .**attr**("transform", "rotate(-90)")

    .**attr**("y", 0 - (padding/.9))

    .**attr**("x",0 - (h / 2))

    .**attr**("dy", "1em")

    .**text**("True Positive Rate ");

}

function **visualize**(){

*//Width and height*

    var w = 1200;

    var h = 600;

*//Define map projection*

    var projection = d3.**geoAlbersUsa**()

                           .**translate**([w/2-w/12, h/2])

                           .**scale**([1200]);

*//Define path generator*

    var path = d3.**geoPath**()

                     .**projection**(projection);

*//Define quantize scale to sort data values into buckets of color*

    var color = d3.**scaleQuantize**()

                        .**range**(['rgb(170, 237, 240)','rgb(253,174,97)','rgb(255,255,191)','rgb(166,217,106)','rgb(26,150,65)']);

*//Remove previous svg elements*

    d3.**select**("#chart1")

    .**selectAll**("\*")

    .**remove**();

*//Create SVG element*

    var svg = d3.**select**("#chart1")

                .**append**("svg")

                .**attr**("width", w)

                .**attr**("height", h);

    var url = "http://127.0.0.1:5000/viz/"

    var modelname = document.**getElementById**("modelname").value;

    var learning\_rate = document.**getElementById**("l").value;

    var max\_depth = document.**getElementById**("depth").value;

    var max\_leaf\_nodes = document.**getElementById**("leaf").value;

    var n\_estimators = document.**getElementById**("estimator").value;

    if (learning\_rate==="") {

    learning\_rate = .1;

  }

  if (n\_estimators==="") {

    n\_estimators = 100;

  }

  if (max\_depth==="") {

    max\_depth = 5;

  }

  if (max\_leaf\_nodes==="") {

    max\_leaf\_nodes = 10;

  }

  url1 = url.**concat**("m=").**concat**(modelname).**concat**("/d=").**concat**(max\_depth).**concat**("/l=").**concat**(max\_leaf\_nodes).**concat**("/n=").**concat**(n\_estimators).**concat**("/c=").**concat**(learning\_rate);

            d3.**csv**(url1).**then**(function(data) {

*//Set input domain for color scale*

                color.**domain**([

                    d3.**min**(data, function(d) { return d.mean\_prob\_companies\_acquired\_by\_state; }),

                    d3.**max**(data, function(d) { return d.mean\_prob\_companies\_acquired\_by\_state; })

                ]);

*//Load in GeoJSON data*

                d3.**json**("\\data\\us-states.json").**then**(function(json) {

*//Merge the ag. data and GeoJSON*

*//Loop through once for each ag. data value*

                    for (var i = 0; i < data.length; i++) {

*//Grab state name*

                        var dataState = data[i].State;

*//Grab data value, and convert from string to float*

                        var dataValue = **parseFloat**(data[i].mean\_prob\_companies\_acquired\_by\_state);

*//Grab data value, and convert from string to float*

                        var dataAcquired = **parseFloat**(data[i].count\_prob\_companies\_acquired\_by\_state);

*//Grab data value, and convert from string to float*

                        var dataClosed = **parseFloat**(data[i].count\_prob\_companies\_closed\_by\_state);

*//Find the corresponding state inside the GeoJSON*

                        for (var j = 0; j < json.features.length; j++) {

                            var jsonState = json.features[j].properties.name;

                            if (dataState == jsonState) {

*//Copy the data value into the JSON*

                                json.features[j].properties.value = dataValue;

                                json.features[j].properties.acquired = dataAcquired;

                                json.features[j].properties.closed = dataClosed;

*//Stop looking through the JSON*

                                break;

                            }

                        }

                    }

*//Bind data and create one path per GeoJSON feature*

                    svg.**selectAll**("path")

                       .**data**(json.features)

                       .**enter**()

                       .**append**("path")

                       .**attr**("d", path)

                       .**style**("fill", function(d) {

*//Get data value*

                            var value = d.properties.value;

                            if (value>=0) {

*//If value exists…*

                                return **color**(value);

                            } else {

*//If value is undefined…*

                                return "#ccc";

                            }

                       })

                       .**style**("fill-opacity", 1)

                       .**style**("stroke", "#a9a9a9")

                       .**style**("stroke-width", 1.5)

                       .**on**("mouseover", function(d) {

*//Get this bar's x/y values, then augment for the tooltip*

*//Update the tooltip position and value*

                    var coordinates = d3.**mouse**(this);

                    var xPosition = coordinates[0] + w/11;

                    var yPosition = coordinates[1] + h/2.25;

                    var format = d3.**format**(".4f");

                    d3.**select**("#tooltip")

                        .**style**("left", xPosition + "px")

                        .**style**("top", yPosition + "px")

                        .**select**("#state")

                        .**text**(d.properties.name);

                    d3.**select**("#tooltip")

                        .**style**("left", xPosition + "px")

                        .**style**("top", yPosition + "px")

                        .**select**("#acquired")

                        .**text**("Startups predicted to be successful: " + d.properties.acquired);

                    d3.**select**("#tooltip")

                        .**style**("left", xPosition + "px")

                        .**style**("top", yPosition + "px")

                        .**select**("#closed")

                        .**text**("Startups predicted to fail: " + d.properties.closed);

                    d3.**select**("#tooltip")

                        .**style**("left", xPosition + "px")

                        .**style**("top", yPosition + "px")

                        .**select**("#prob")

                        .**text**("Mean probability of startups to be successful: " + **format**(d.properties.value));

*//Show the tooltip*

                    d3.**select**("#tooltip").**classed**("hidden", false);

          d3.**select**("#tooltip").**style**("display", "block");

               })

               .**on**("mouseout", function() {

*//Hide the tooltip*

                    d3.**select**("#tooltip").**classed**("hidden", true);

                    d3.**select**("#tooltip").**style**("display", "none");

               });

                });

            var linear = d3.**scaleQuantize**()

              .**domain**([

                    d3.**min**(data, function(d) { return d.mean\_prob\_companies\_acquired\_by\_state; }),

                    d3.**max**(data, function(d) { return d.mean\_prob\_companies\_acquired\_by\_state; })

                ])

              .**range**(['rgb(170, 237, 240)','rgb(253,174,97)','rgb(255,255,191)','rgb(166,217,106)','rgb(26,150,65)']);

            var svg = d3.**select**("svg");

            svg.**append**("g")

              .**attr**("class", "legendLinear")

              .**attr**("transform", "translate(1000,200)");

            var legendLinear = d3.**legendColor**()

              .**shape**('rect')

              .**shapeWidth**(50)

              .**shapePadding**(5)

              .**orient**('vertical')

              .**ascending**(true)

              .**scale**(linear)

              ;

            svg.**select**(".legendLinear")

              .**call**(legendLinear);

            });

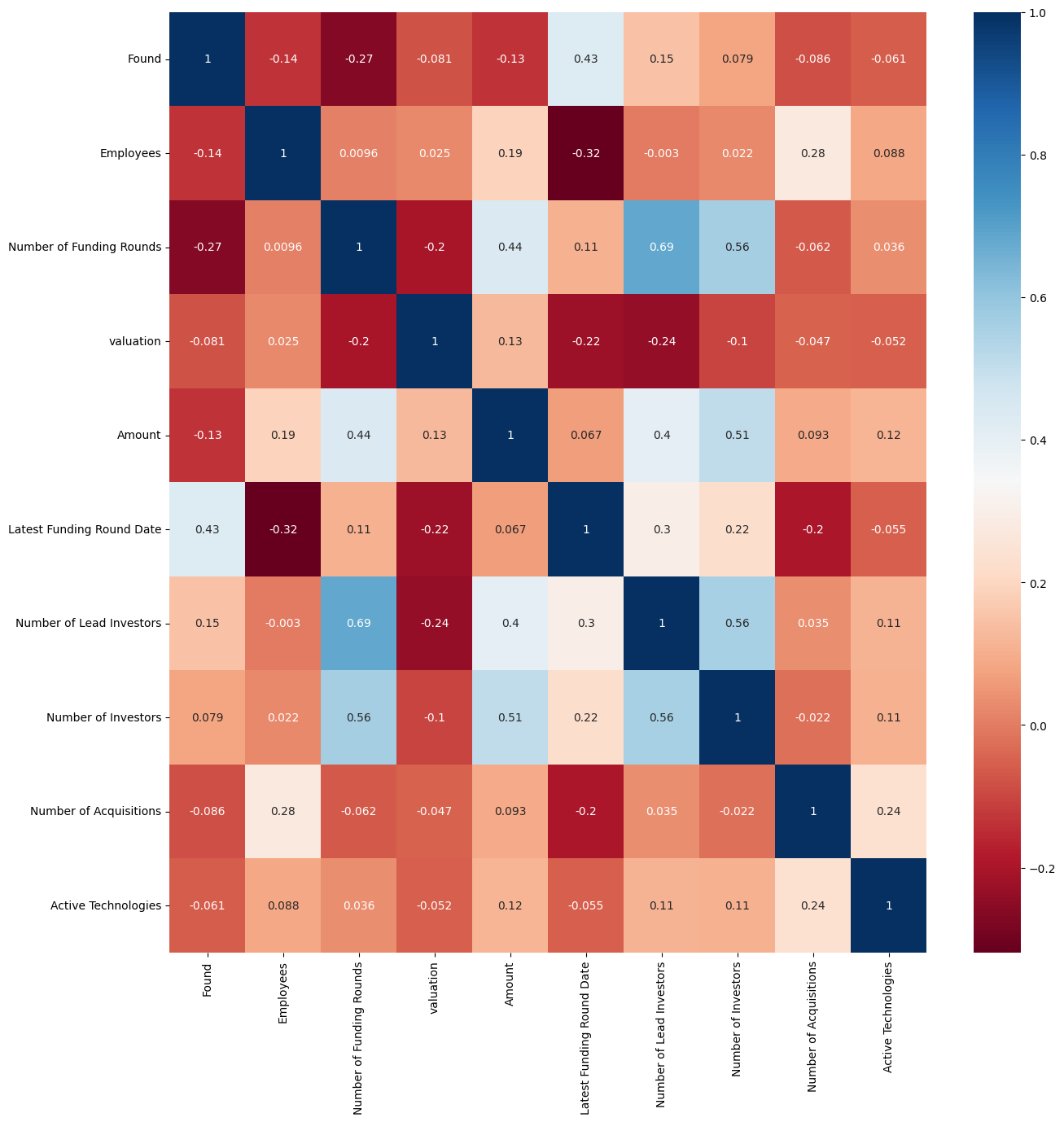
}

The integration of diverse data visualization techniques, machine learning model training, and the development of an interactive web application, through the above code, culminates in a comprehensive and powerful solution, laying the foundation for informed decision-making in the dynamic landscape of startup success prediction.

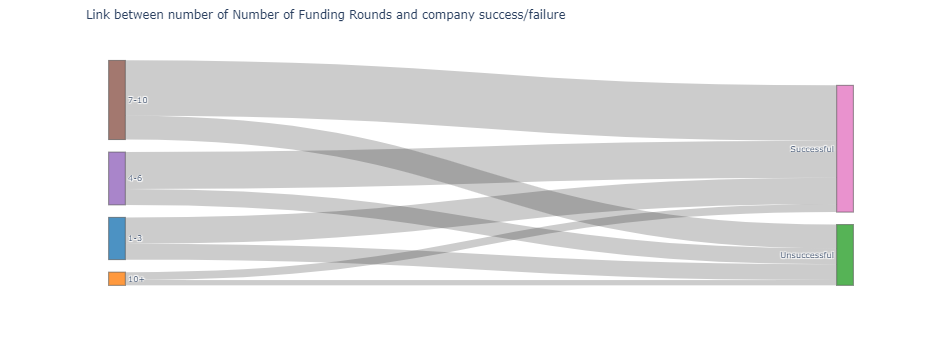
**4.8 RESULTS**

**Results of Visualizations:**

Our visualization efforts yielded insightful outcomes, illuminating key factors that influence startup success. Notably, sankey plots were employed to showcase relationships between the number of funding rounds, lead investors, and acquisitions. Additionally, a correlation scatter plot illustrated the nuanced connection between the price acquired and total funding amount. These visualizations, embedded within our exploratory data analysis, provide an intuitive understanding of the intricate interplay of variables in the startup ecosystem.

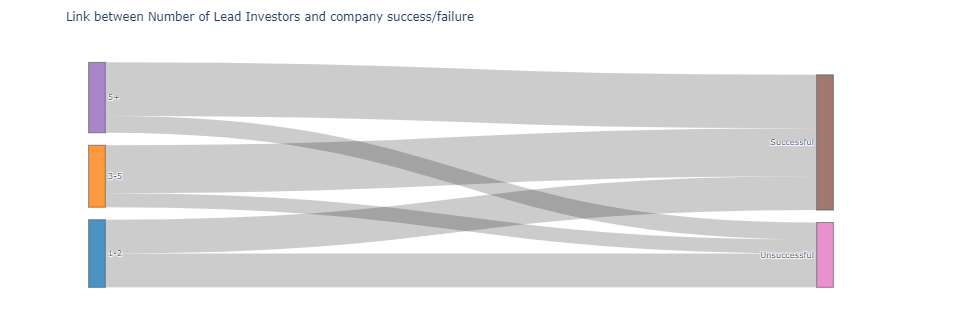


*Figure 4.8.1 Correlation between success and other factors affecting the startup.*



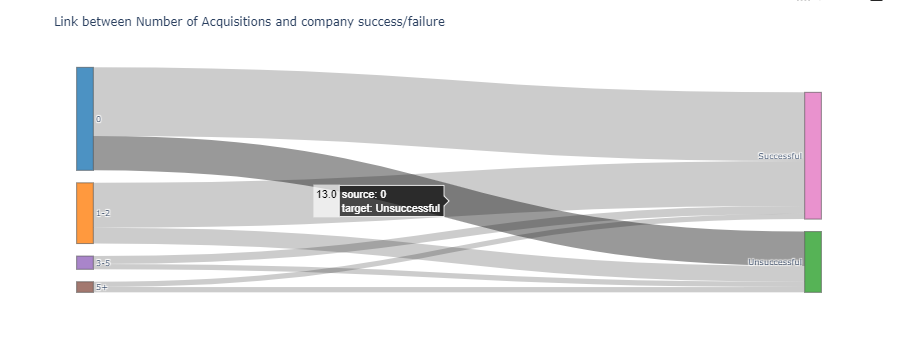
*Figure 4.8.2 Sankey Plot 1*

*(between Funding Round and Success/Failure)*

**

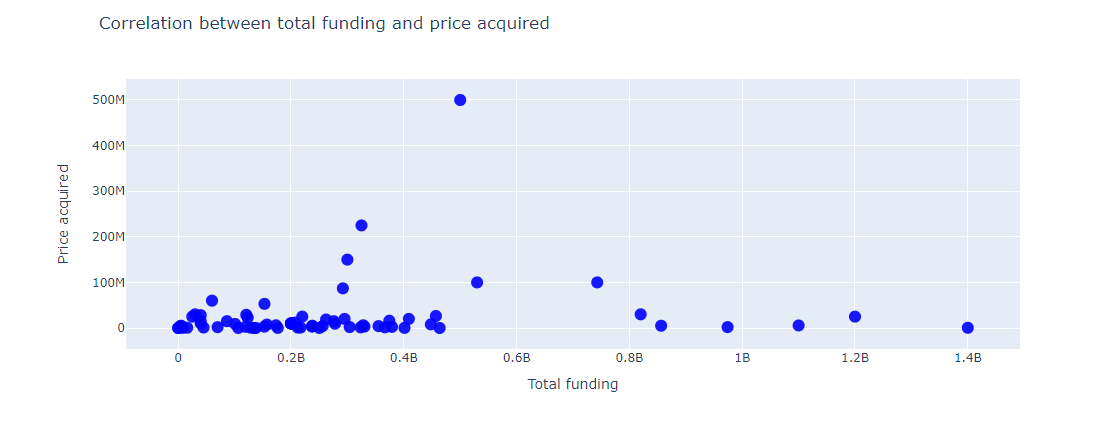
*Figure 4.8.3 Sankey Plot 2*

*(between Number of Lead Investors and Success/Failure)*

**

*Figure 4.8.4 Sankey Plot 3*

*(between Number of Acquisitions and Success/Failure)*

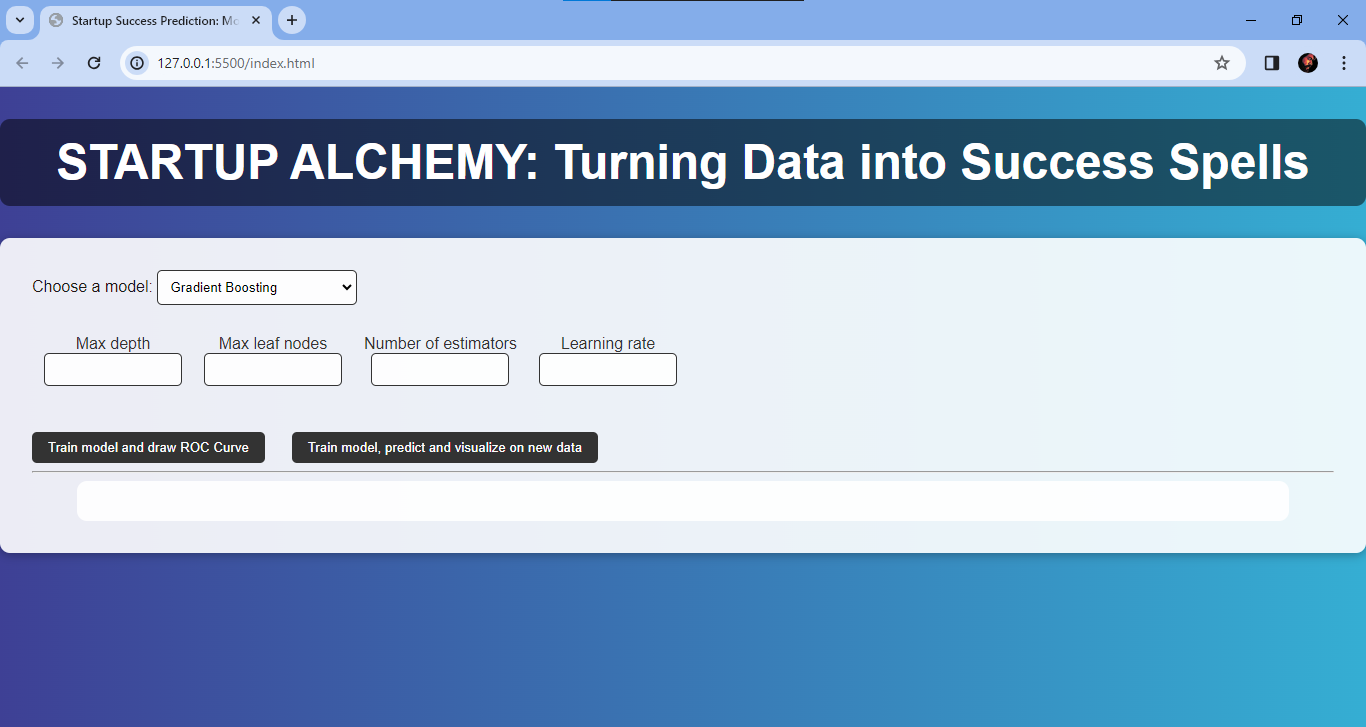
**

*Figure 4.8.5 Scatter Plot*

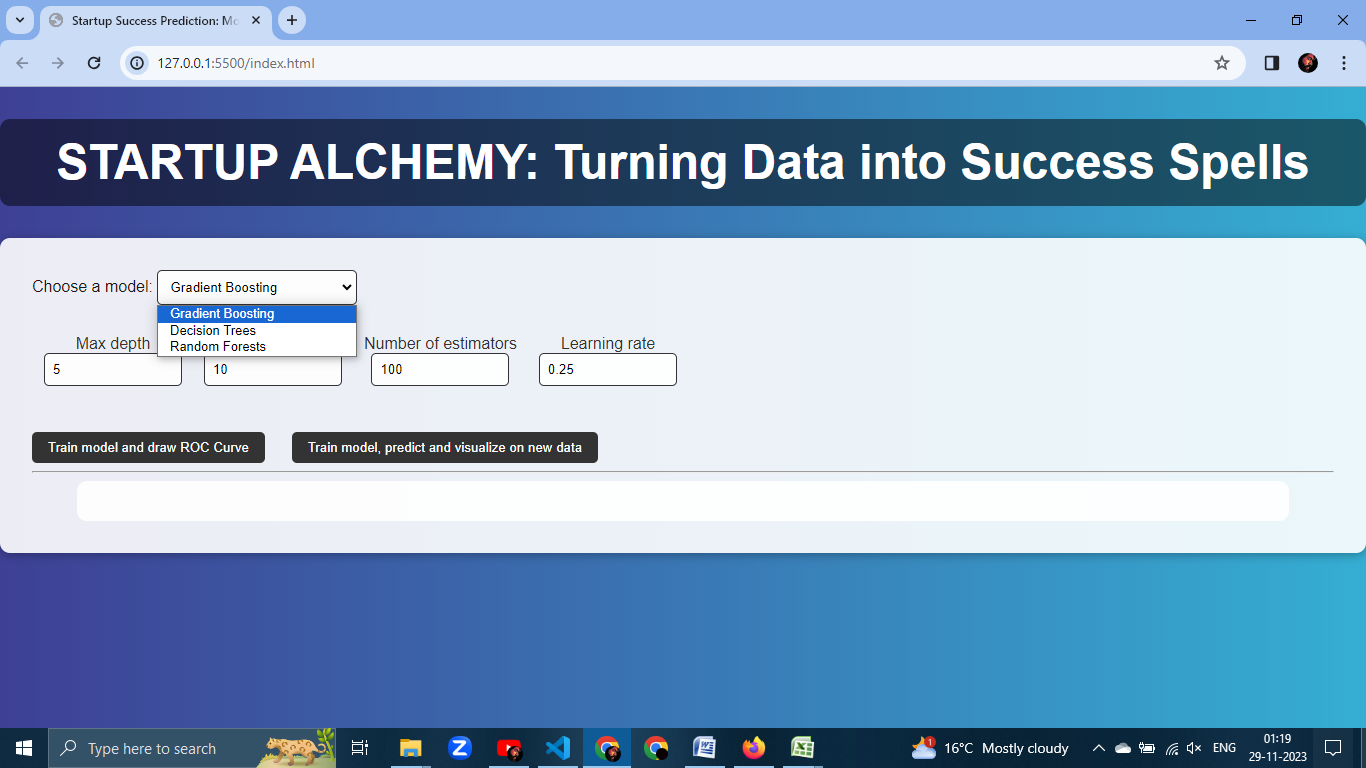
*(showing Correlation between Price Acquired and Total Funding received)*

**Web Page:**

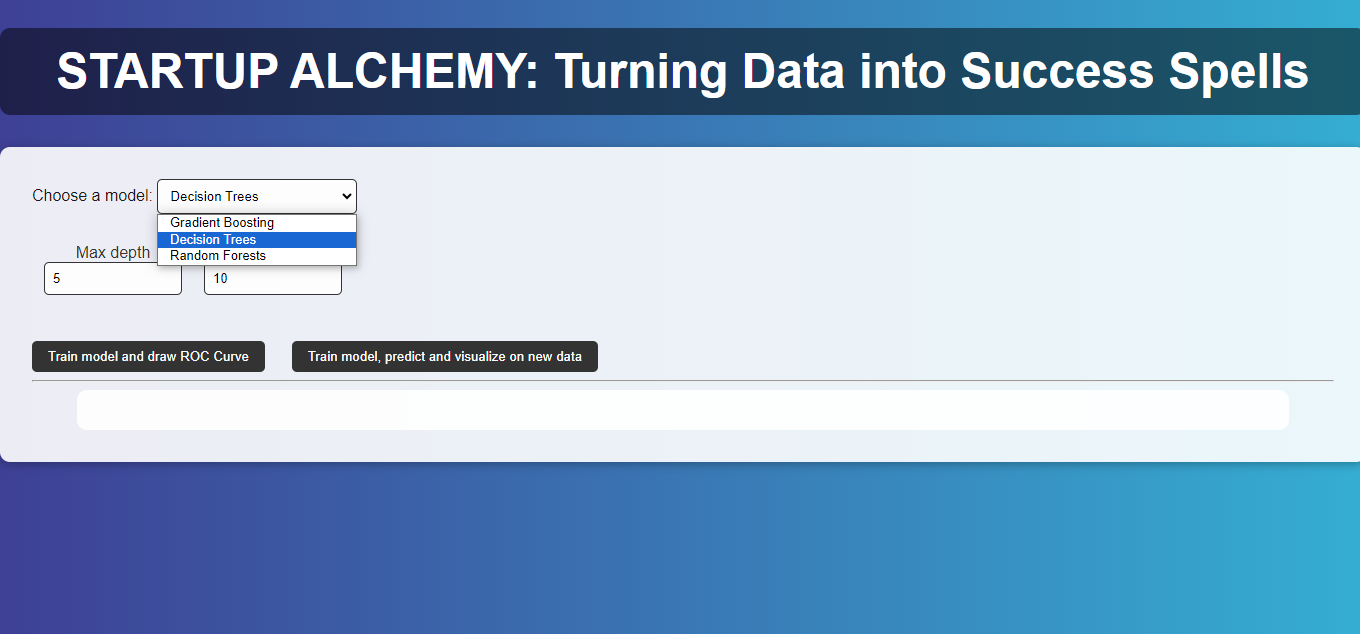
The web page designed for showcasing prediction is user interactive and provides an option between 3 prediction models- Gradient Boosting, Decision Trees, Random Forests, and 2 kind of visualization for each model- ROC Curve and state wise Prediction on the map of US.



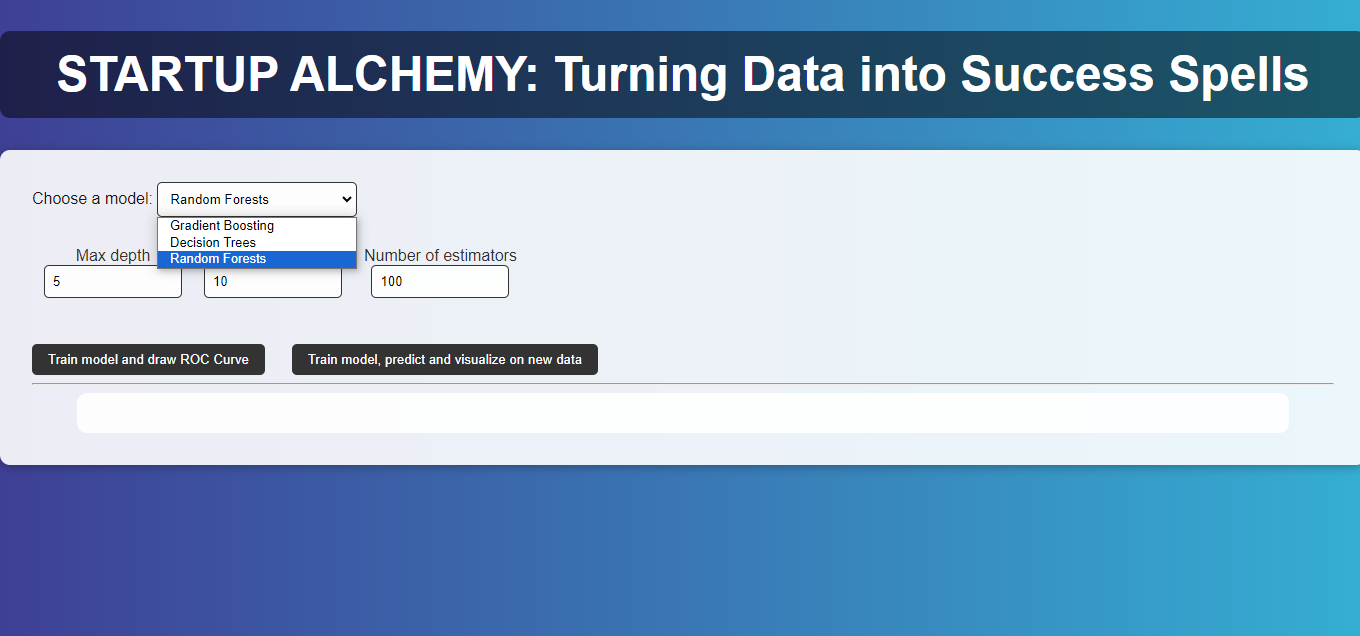
*Figure 4.8.6 Web Page*



*Figure 4.8.7 Model 1*

**

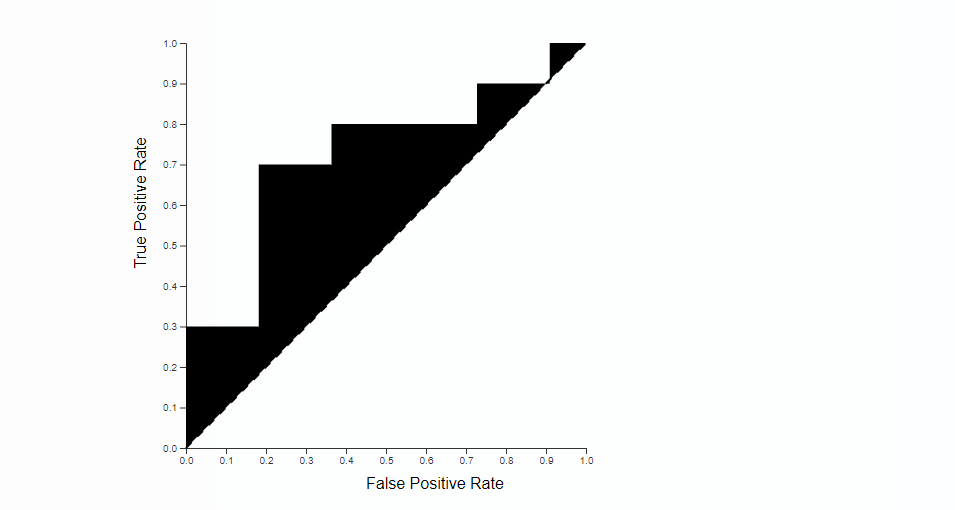
*Figure 4.8.8 Model 2*

**

*Figure 4.8.9 Model 3*

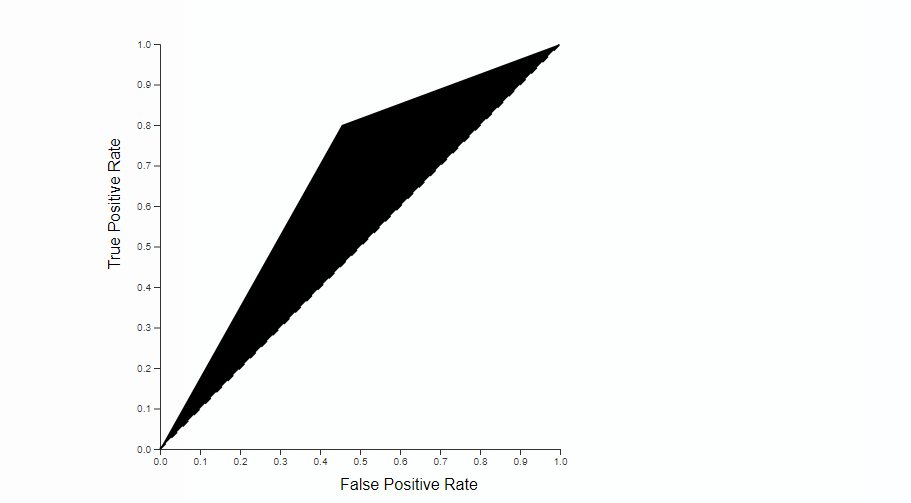
**ROC Curve Analysis:**

The ROC curve analysis showcased the robust performance of our machine learning models. Each model—Gradient Boosting, Random Forest, and Decision Trees—exhibited commendable predictive accuracy. The curves visually convey the trade-offs between true positive rates and false positive rates, allowing for a nuanced assessment of model performance. The area under the ROC curve (AUC-ROC) metrics further underscore the efficacy of our predictive models.



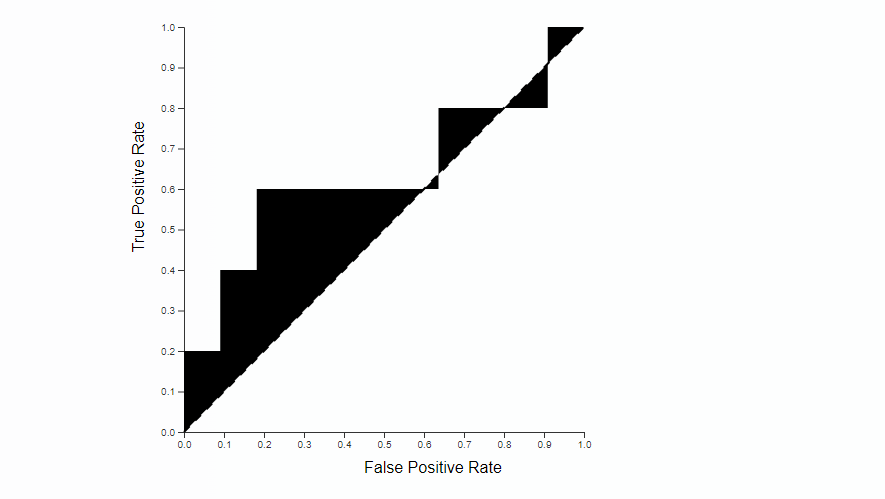
*Figure 4.8.10 ROC Curve 1*

*(Gradient Boosting Model)*

**

*Figure 4.8.11 ROC Curve 2*

*(Decision Tree Model)*

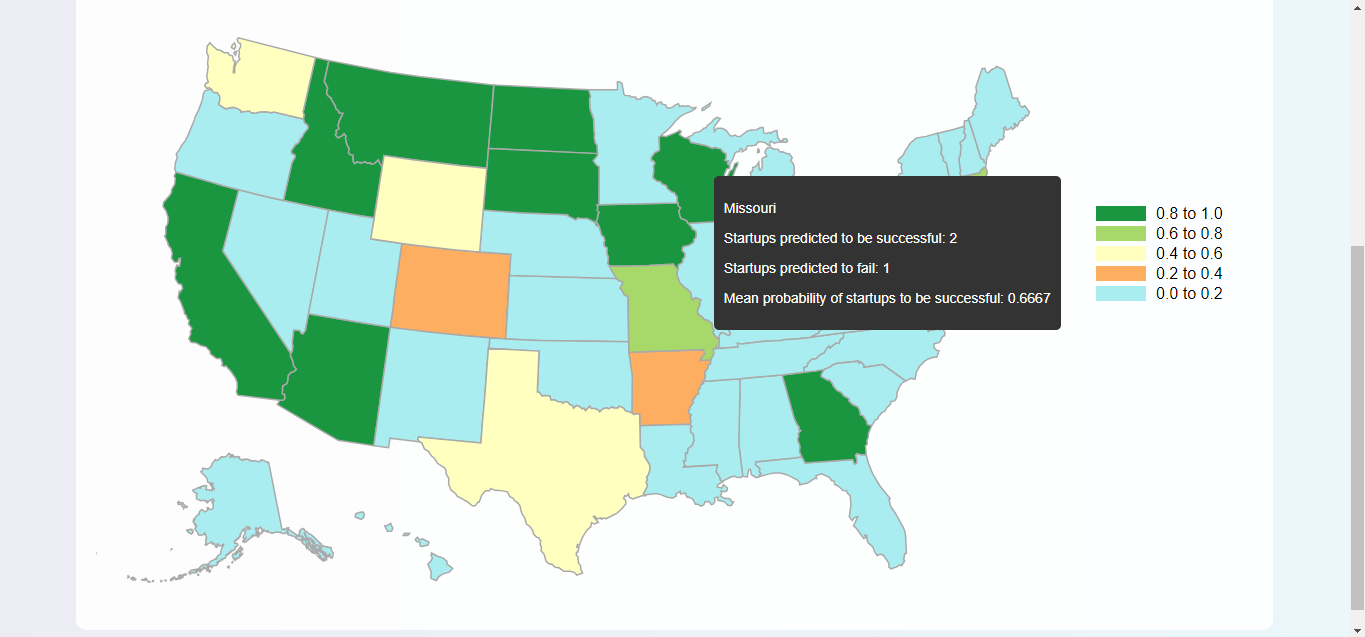
**

*Figure 4.8.12 ROC Curve 3*

*(Random Forests Model)*

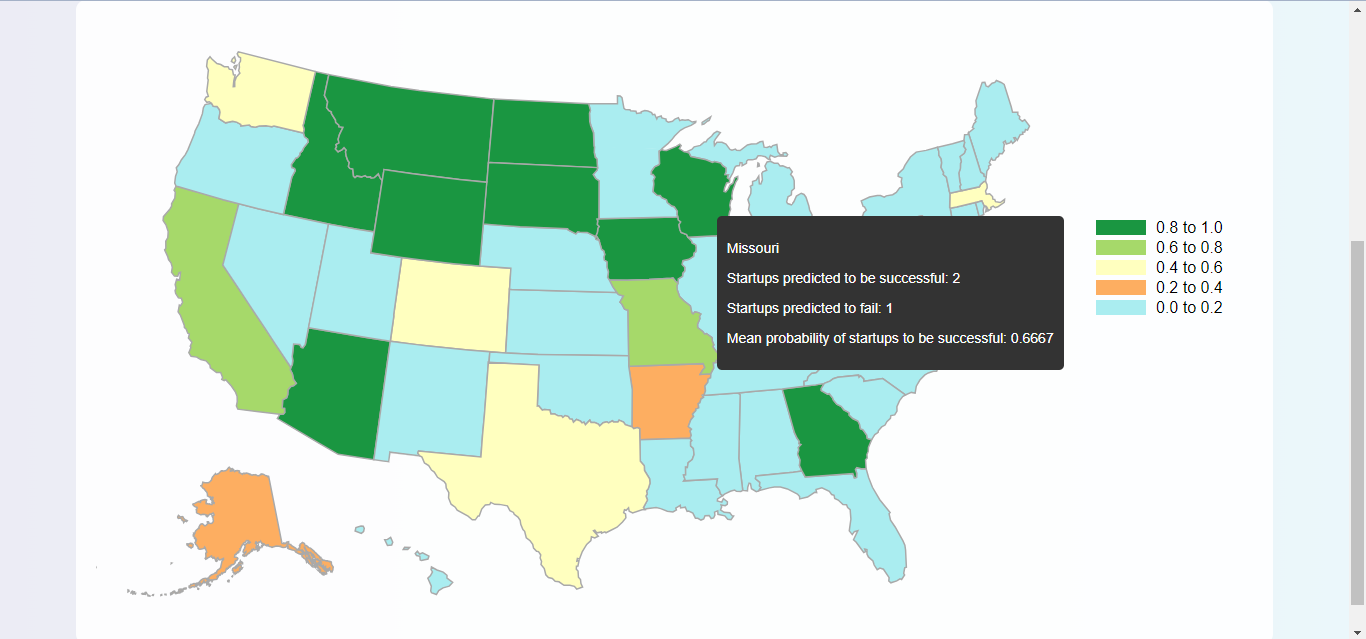
**Geospatial Visualization on US Map:**

Our project's geospatial visualization component presents a captivating portrayal of startup success across different U.S. states. The map vividly displays the distribution of successful and unsuccessful startups, with varying colors indicating the magnitude of each category. This visual representation enables stakeholders to glean valuable insights into regional trends, aiding strategic decision-making in the dynamic landscape of startup entrepreneurship.



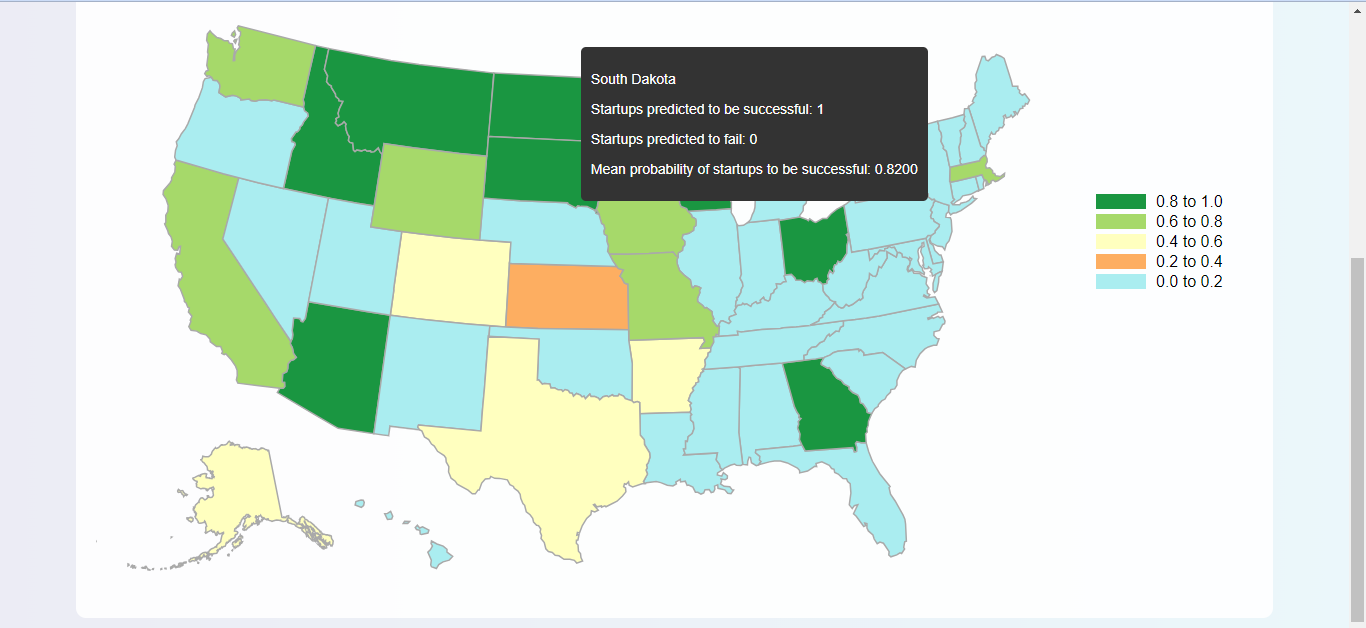
*Figure 4.8.13 Geospatial Visualization 1*

*(Gradient Boosting Algorithm)*

**

*Figure 4.8.14 Geospatial Visualization 2*

*(Decision Trees Algorithm)*

**

*Figure 4.8.15 Geospatial Visualization 3*

*(Random Forests Algorithm)*

**4.9 CONCLUSION**

In conclusion, "Startup Alchemy: Turning Data into Success Spells" represents a significant leap forward in leveraging data and artificial intelligence for predicting the success of startups, with a specialized focus on AI startups in healthcare. Through the simultaneous training of Gradient Boosting, Random Forest, and Decision Tree Classifier models, our project introduces a robust and diverse approach to predictive analytics in the dynamic landscape of startup ecosystems. The benefits of model diversity and ensemble learning enhance the overall predictive accuracy, offering stakeholders nuanced insights into the factors influencing startup success.

The creation of a bespoke dataset and its meticulous curation, alongside the application of advanced visualization techniques, provides an intuitive exploration of data relationships. The use of Python's Matplotlib library for visualization, along with other JavaScript libraries for frontend interactivity, ensures a comprehensive understanding of influential features for model training. The project's outcomes are not only confined to predictive accuracy but also extend to an informed feature selection process that can guide strategic decision-making.

Looking ahead, the project's future scope includes further refinement of models, incorporating more sophisticated algorithms and expanding the dataset to encompass a broader spectrum of startup attributes. The deployment of machine learning models on real-time data and continuous monitoring of their performance can contribute to an adaptive and evolving predictive system. Additionally, integrating user feedback and refining the user interface can enhance the project's usability and accessibility for a wider audience.

In conclusion, "Startup Alchemy" lays the foundation for informed decision-making in the startup landscape, providing a transformative tool for entrepreneurs, investors, and decision-makers. The amalgamation of data science, artificial intelligence, and advanced visualization techniques creates a powerful synergy, unlocking insights that can potentially shape the future trajectory of startups, particularly in the burgeoning field of AI in healthcare. As we reflect on the journey of transforming raw data into predictive spells, the project stands as a testament to the potential of data-driven innovation in the entrepreneurial realm.

1. **REFERENCES**

Following published works and sites were referred that helped us complete this project :

1. Research Papers and Thesis:

* + **“Predicting Startup Success Using Publicly Available Data”**  by Emily Gavrilenko, A Thesis presented to the Faculty of California Polytechnic State University, San Luis Obispo.
  + **“Predicting the outcome of startups: less failure, more success”** by Krishna, A., Agrawal, A., & Choudhary, A. (2016, December). In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)* (pp. 798-805). IEEE.
  + **“Next-generation business models for artificial intelligence start-ups in the healthcare industry.”** by Kulkov, Ignat. *International Journal of Entrepreneurial Behavior & Research* 29.4 (2023): 860-885.

2. Data Sources:

- Crunchbase: www.crunchbase.com

- Kaggle: www.kaggle.com

- Medical Startups: https://www.medicalstartups.org/top/ai/

3. Learning Sources:

- GitHub: www.github.com

- YouTube: www.youtube.com

- W3Schools: www.w3schools.com

- Scikit Learn: www.scikit-learn.org