Shark Tank US Data Analysis and Prediction

Subhrajit Dey (sd5963@nyu.edu)
Raunak Choudhary (rc5553@nyu.edu)
Saniya Gapchup (syg2021@nyu.edu)

Department of Computer Science and Engineering
New York University, Tandon School of Engineering

Abstract

This study explores the investment patterns of Shark Tank US investors through comprehensive data analysis and machine learning techniques. Using a dataset spanning 16 seasons, models such as Random Forest, Logistic Regression, SVM, and XGBoost were employed to predict both investment likelihood (classification) and investment amounts (regression). SMOTE was applied to address class imbalances, improving classification accuracy. Random Forest emerged as the most consistent classifier, while XGBoost excelled in predicting investment amounts for specific sharks. Challenges, such as variability in investment behaviors and outliers, were mitigated by using advanced feature engineering like the usage of embeddings. We evaluated the models using various metrics, including RMSE, R^2, and confusion matrices, to analyze results and compare the performances of different models. The study highlights the effectiveness of predictive modeling in understanding funding dynamics and offers insights for entrepreneurs and investors to optimize decision-making processes.

1. Introduction

The startup ecosystem has become a cornerstone of innovation and economic growth, with entrepreneurial ventures significantly contributing to job creation and technological advancement [1]. However, securing funding remains a formidable challenge for startups, especially in their nascent stages [1]. Television programs like Shark Tank US have provided a unique lens into the investment decision-making processes of prominent venture capitalists and angel investors, offering valuable insights into the dynamics of startup funding [3].

This study aims to analyze the factors that influence funding decisions on Shark Tank US, focusing on the investment patterns of notable "sharks" such as Mark Cuban and Barbara Corcoran. By examining historical data from the show, the research seeks to identify key determinants that affect the likelihood of a startup securing investment [3]. Understanding these factors is crucial for aspiring entrepreneurs aiming to navigate the complex landscape of startup financing [4].

The motivation for this project stems from the authors' entrepreneurial aspirations and a desire to contribute to the broader understanding of the U.S. funding environment [1] [4]. By leveraging machine learning techniques, this study explores the feasibility of predicting funding outcomes based on various startup characteristics, founder profiles, and pitch details. Such predictive models are valuable tools for entrepreneurs and investors, bridging the gap between qualitative assessments and data-driven decision-making [1].

The primary objectives of this research are to:

a) Examine the investment patterns of individual sharks, including the amounts they invest and the industries they prefer.
b) Evaluate the performance of different predictive models, identifying the most effective ones and analyzing the underlying factors contributing to their success.
c) Formulate strategies to address the class imbalance between investments and non-investments in the dataset while effectively incorporating non-numerical fields into the analysis.

To achieve these goals, the study employs a multi-output binary classification approach to predict whether each shark has invested or not, comparing the performance of support vector classification (SVC), logistic regression, XGBoost, and random forest models. For predicting investment amounts, random forest regressors and XGBoost regressors were utilized, with their performances analyzed to identify the most effective methods.

Through meticulous data preprocessing and evaluation, this research aspires to build upon existing literature and offer new insights into the entrepreneurial funding process [2] [4]. Beyond academic contributions, the findings open opportunities to develop applications simulating the entire Shark Tank experience and creating interactive online AI games for users to play with their friends, further demystifying and engaging with the funding process [3] [4].

2. Literature Review

The study of factors influencing startup funding and investment decisions has long been of interest to researchers, particularly with the rise of entrepreneurship as a critical driver of economic growth. The Shark Tank US dataset provides a unique opportunity to explore these dynamics in a real-world context. While prior studies on startup investment have largely focused on venture capital and crowdfunding platforms, the structured environment of Shark Tank US offers a rich dataset for analysis, encompassing founder pitches, shark decisions, and investment outcomes.

2.1 Studies on Investment Decision-Making

Investment decision-making is a multifaceted process that involves evaluating financial metrics, assessing market potential, and understanding the psychological biases of investors. Research by Gompers, etc. (2021) highlights the role of venture capitalists' experience, network effects, and risk tolerance in funding decisions [1]. Similarly, studies like those by Malmström M., etc. (2020) investigate gender biases and their impact on funding outcomes, demonstrating the need for equitable evaluation criteria [5]. These insights are foundational for understanding the dynamics within Shark Tank US.

2.2 Machine Learning Applications in Startup Funding

Machine learning has emerged as a powerful tool for analyzing investment trends and predicting funding outcomes. Techniques such as logistic regression, random forests, and XGBoost have been widely employed in predicting the success of crowdfunding campaigns and venture-backed startups. The integration of machine learning in this study allows for a granular analysis of factors influencing shark decisions, bridging the gap between qualitative insights and data-driven predictions.

2.3 Insights from the Shark Tank US Dataset

The dataset published by Thirumani on Kaggle [6] [7] serves as a valuable resource for understanding investment trends within the context of Shark Tank US. The initial analysis sheds light on key metrics such as deal rates, valuation trends, and industry preferences [6]. This study builds on that foundation, utilizing machine learning models to uncover deeper insights and enhance predictive capabilities. The structured nature of the dataset encompassing fields like pitch characteristics, deal outcomes, and industry classifications offers a robust platform for exploratory and predictive analysis.

2.4 Challenges and Opportunities

The primary challenge in leveraging the Shark Tank US dataset lies in its inherent class imbalance, with significantly more "no deal" outcomes compared to successful deals. Prior research has demonstrated the efficacy of resampling techniques and algorithmic adjustments in addressing class imbalance, particularly in domains with similar skewed distributions [8]. Additionally, the integration of non-numerical fields, such as textual descriptions and categorical variables, presents an opportunity to derive richer insights using natural language processing (NLP) and feature engineering techniques [9].

2.5 Contribution to the Literature

This study contributes to the existing body of knowledge by applying machine learning to a novel dataset, highlighting the nuanced factors that drive investment decisions within Shark Tank US. In classification testing, all four models—support vector classification (SVC), logistic regression, random forests, and XGBoost—were evaluated to compare their predictive accuracy for investment likelihood, while in regression testing, only the top two models, Random Forest and XGBoost, were analyzed to predict investment amounts. By comparing these models' performance, this research not only evaluates predictive accuracy but also provides actionable insights for entrepreneurs seeking funding.

In conclusion, the literature on investment decision-making and machine learning provides a strong foundation for this study. By extending these methodologies to the Shark Tank US dataset, this research aims to advance the understanding of startup funding dynamics and inspire future innovations in predictive analytics.

3. Models

3.1 Support Vector Classification (SVC)

Support Vector Machines (SVM) are supervised learning models widely used for classification and regression tasks. Support Vector Classification (SVC) focuses specifically on classification problems. It identifies an optimal hyperplane that separates data points of different classes while maximizing the margin between the hyperplane and the nearest data points (support vectors). This margin maximization enhances the model's robustness and generalization.

For non-linear data, SVC uses kernel functions to transform the input data into a higher-dimensional space, enabling it to find non-linear decision boundaries. Popular kernel types include linear, polynomial, radial basis function (RBF), and sigmoid.

The curve shown in the diagram, as illustrated in Figure 1 for SVC, shows a feature space where data points are plotted. A hyperplane divides the classes, with the support vectors (closest points to the hyperplane) explicitly marked. The margin, which is maximized during training, is the distance between the hyperplane and the support vectors. Nonlinear SVC diagrams may also show the transformation of data into higher-dimensional spaces for classification.

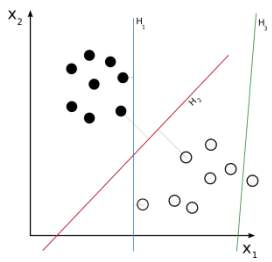

Figure 1: SVM Separating Hyperplanes

3.1.1 Formulae

a) Decision Function:

For a linear SVM:

$$f(x) = w^T x + b$$

- w: weight vector
- x: input feature vector
- b: bias term

b) Optimization Objective:

$$\min \frac{1}{2}||w||^2$$

Subject to:

$$y_i(w^T x_i + b) \geq 1 \quad \forall i$$

- y_i : class label of the i-th data point

c) Kernelized SVM:

For non-linear decision boundaries:

$$K(x, x') = \phi(x)^T \phi(x')$$

- $\phi(x)$: mapping to a higher dimensional space
- K(x,x'): kernel function

3.1.2 Pros and Cons

Pros:

- Effective in High Dimensions: SVC handles high-dimensional datasets efficiently.
- Versatility: It can solve linear and non-linear problems using appropriate kernel functions.
- Robustness: Focusing on support vectors reduces sensitivity to outliers.

Cons:

- Computationally Intensive: Training can be slow for large datasets, especially with non-linear kernels.
- Memory Requirements: High memory usage with large numbers of support vectors.
- Hyperparameter Selection: Requires careful tuning of the kernel type and regularization parameter (CCC).

3.2 Logistic Regression

Logistic regression is a widely used statistical model designed for binary classification problems. Unlike linear regression, which predicts continuous values, logistic regression predicts the probability of a binary outcome by applying a logistic function to a linear combination of input features. This probability is then mapped to class labels using a threshold, typically 0.5.

Logistic regression assumes a linear relationship between the input features and the log-odds of the dependent variable. Despite its simplicity, it is a powerful algorithm for classification tasks, particularly when interpretability is important.

The curve shown in the diagram, as illustrated in Figure 2 for logistic regression typically represents the flow from input features to the output probability. It starts with the input layer representing the feature vector x. The linear transformation $w^T x+b$ is applied, followed by the logistic function (sigmoid), which maps the result to a probability value between 0 and 1. Finally, a decision threshold (commonly 0.5) is applied to classify the output as 0 or 1. The curve shown below includes the input features, linear transformation, sigmoid function, and binary classification decision.

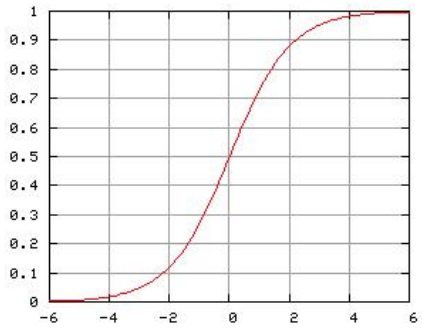

Figure 2: Logistic Regression Curve

3.2.1 Formulae

a) Logistic Function:

The logistic function maps any real-valued number to the range [0, 1]:

$$P(y = 1|x) = \frac{1}{1 + e^{-(w^T x+b)}}$$

- P(y = 1|x): Probability of the positive class given the input x
- w: Weight vector
- x: Input feature vector
- b: Bias term
-

b) Log-Odds:

The relationship between the logistic function and log-odds:

$$\ln\left(\frac{P(y = 1|x)}{1 - P(y = 1|x)}\right) = w^T x + b$$

c) Loss Function:

The loss function for logistic regression is based on cross-entropy:

$$L(w,b) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(P(y_i)) + (1-y_i)\log(1-P(y_i))]$$

- N: Number of samples
- y_i: Actual class label for sample i
- $P(y_i)$: Predicted probability for sample i

3.2.2 Pros and Cons

Pros:

- Simplicity: Easy to implement and computationally efficient for small to medium-sized datasets.
- Interpretability: Coefficients provide insights into the relationship between features and the target variable.
- Probability Outputs: Logistic regression provides probabilities, enabling a nuanced understanding of predictions.
- Works Well on Linearly Separable Data: It performs well when the classes are linearly separable.

Cons:

- Assumes Linearity: Relies on the assumption of a linear relationship between features and the log-odds, which may not hold for complex datasets.
- Not Suitable for Non-linear Problems: Limited by its inability to model non-linear decision boundaries without feature transformation.
- Sensitive to Outliers: Outliers can significantly influence the model's performance.

3.3 XGBoost

XGBoost (eXtreme Gradient Boostin), is an advanced machine learning algorithm based on the gradient boosting framework. It is designed for speed and performance, offering scalability and efficiency for supervised learning tasks such as regression and classification. XGBoost creates an ensemble of weak learners, typically decision trees, where each subsequent tree corrects the errors of the previous ones by optimizing a differentiable loss function. It is widely used due to its ability to handle structured and tabular data effectively.

The distinguishing features of XGBoost include regularization techniques (L1 and L2) to prevent overfitting, sparse-aware learning to handle missing values, and parallelized tree construction for faster computation. These innovations make it highly versatile and competitive for machine learning tasks.

The schematic diagram of the XGBoost, as depicted in the Figure 3, demonstrates its iterative and additive approach to learning. The process begins with the input training data, where the initial predictions are made using a base model (commonly set to a constant). Each subsequent tree in the sequence focuses on minimizing the residual errors from the previous trees by computing the gradients of the loss function.

The diagram highlights the following key components:

- Tree Construction: Each tree is built sequentially to correct the mistakes of its predecessor. The decision trees are shown in individual blocks, with nodes representing splits based on feature values and leaves containing the predictions for subsets of the data.

- Gradient and Hessian Computation: Gradients and second-order gradients (Hessians) are calculated for each instance to measure the direction and magnitude of optimization required. This enables precise updates to the model at each step.
- Model Aggregation: The outputs of all trees are aggregated to form the final prediction. This additive model ensures that errors are iteratively reduced, improving overall performance.
- Validation and Testing: A separate validation set is used to monitor performance and prevent overfitting. The diagram includes a testing phase where the final model is applied to unseen data.

This iterative process, combined with regularization, makes XGBoost robust against overfitting and highly effective for complex machine learning tasks.

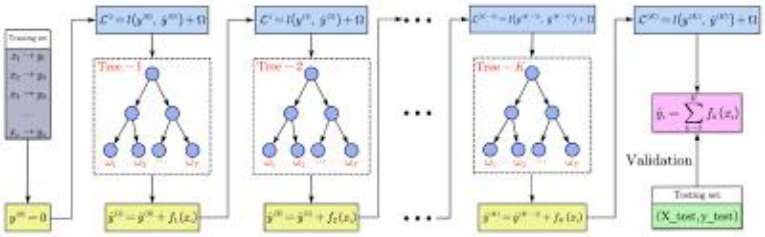

Figure 3: Schematic Diagram of XGBoost

3.3.1 Formulae

a) Objective Function:

The objective function in XGBoost combines a loss function and a regularization term:

$$\text{Objective} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

- $l(y_i, \hat{y}_i)$: Loss function measuring the difference between actual (y_i) and predicted ($\hat{y}_i$) values.
- $\Omega(f_k)$: Regularization term to control model complexity.
  - $\Omega(f_k) = \gamma T + (1/2)\ \lambda\ \|w\|^2$
  - T: Number of leaves in the tree.
  - w: Leaf weights.
  - γ , λ : Regularization parameters.

b) Additive Tree Model:

XGBoost makes predictions by summing the outputs of all trees:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)$$

- f_k: Output of the k-th tree.
- x_i: Input feature vector for the i-th instance.

c) Weight Update Rule:

During optimization, the weights for tree leaves are updated using gradients:

$$w_j = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$$

- g_i: First-order gradient of the loss function.
- h_i: Second-order gradient (Hessian).
- I_j: Set of instances in leaf j.
- λ : Regularization parameter.

3.3.2 Pros and Cons

Pros:

- High Performance: Efficient handling of large datasets and fast training through parallelization.
- Regularization: L1 and L2 regularization techniques reduce overfitting.
- Customizability: Supports custom loss functions and evaluation metrics.
- Robustness: Handles missing values and sparse data effectively.

Cons:

- Complexity: Requires careful tuning of multiple hyperparameters.
- Computationally Intensive: Can be resource-intensive for very large datasets.
- Interpretability: Ensemble methods like XGBoost can be harder to interpret compared to simpler models like logistic regression.

3.4 Random Forest

Random Forest is a versatile ensemble learning algorithm designed for both classification and regression tasks. It operates by constructing a collection of decision trees during training and aggregating their outputs to make a final prediction. For classification, Random Forest uses majority voting, and for regression, it averages the outputs of the individual trees.

The core idea behind Random Forest is to reduce overfitting and improve generalization by combining multiple trees trained on different subsets of data. Additionally, it introduces randomness by selecting a random subset of features for splitting at each node. This feature randomness decorrelates the individual trees, leading to a more robust model.

Random Forest is widely appreciated for its simplicity, effectiveness, and ability to handle high-dimensional data. It also automatically handles missing values and captures non-linear relationships between features and the target variable.

The architecture diagram of Random forest, shown in Figure 4, illustrates how Random Forest operates as an ensemble model. It begins with the original dataset, which is divided into multiple subsets through bootstrap sampling. Each of these subsets is used to train an individual decision tree.

The trees () produce independent predictions based on their training data. These predictions are then aggregated using majority voting for classification tasks or averaging for regression tasks. This ensemble approach ensures robust predictions by reducing overfitting and improving generalization, leveraging the diversity of trees trained on different data subsets and feature combinations.

This structure, as depicted, showcases the strength of Random Forest in combining multiple weak learners to achieve high accuracy and reliability.

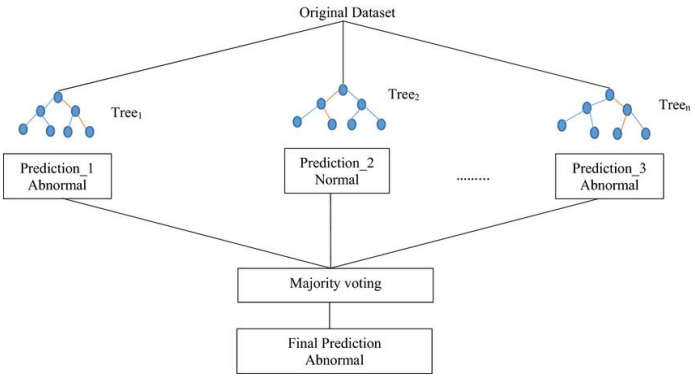

Figure 4: Architecture Diagram of Random Forest

3.4.1 Formulae

a) Prediction for Classification:

For a classification task, the output of Random Forest is the mode of the predictions from individual trees:

$$\hat{y} = \text{mode}(T_1(x), T_2(x), \ldots, T_B(x))$$

- $T_b(x)$: Prediction from the b-th tree.
- B: Total number of trees.

b) Prediction for Regression:

For regression tasks, the prediction is the mean of the outputs from all trees:

$$\hat{y} = \frac{1}{B} \sum_{b=1}^{B} T_b(x)$$

c) Feature Importance:

Feature importance is computed based on the decrease in Gini impurity or information gain across all trees:

$$\text{Feature Importance} = \frac{1}{B} \sum_{b=1}^{B} \Delta\text{Impurity}_b$$

- $\Delta\text{Impurity}_b$: Decrease in impurity due to splits involving the feature in tree b.

3.4.2 Pros and Cons

Pros:

- Reduced Overfitting: By averaging multiple trees, Random Forest minimizes the risk of overfitting.
- Versatility: Handles both classification and regression tasks effectively.
- Feature Importance: Provides insights into feature importance, aiding interpretability.
- Robustness: Performs well with missing data and outliers.
- Scalability: Suitable for high-dimensional data.

Cons:

- Computationally Intensive: Training and prediction can be slow with a large number of trees.
- Less Interpretability: The ensemble nature makes it less interpretable compared to a single decision tree.
- Bias in Small Datasets: May exhibit bias if the dataset is too small.

4. Experiment

4.1 Data

The dataset used in this project is sourced from Kaggle [6] [7] and provides comprehensive information on all 16 seasons of the American reality television series *Shark Tank US*. It contains over 1,360 records, each corresponding to a pitch, and spans 53 features that capture detailed aspects of entrepreneurial pitches, investor preferences, and outcomes. The dataset is a rich resource for understanding trends in funding and investment behavior, offering both categorical and numerical variables for analysis.

Key features of the dataset include "Season Number," "Startup Name," and "Industry," which offer temporal and categorical insights into the pitches. Financial details such as "Original Ask Amount," "Total Deal Amount," and "Got Deal" provide critical indicators of investment dynamics. Entrepreneurial demographics are captured through features like "Pitchers Gender," "Pitchers Average Age," and "Pitchers City," which enable nuanced analyses of representation and geographic trends. Additionally, the dataset tracks shark-specific contributions through fields like "Mark Cuban Investment Amount" and "Barbara Corcoran Investment Equity" allowing for detailed evaluations of individual investor behavior.

The dataset reveals several noteworthy trends. For instance, industries like "Food and Beverage," "Lifestyle/Home," and "Fashion/Beauty" dominate the investments, with substantial activity observed across these categories. Entrepreneurial teams in the 30-50 age range are most prevalent, reflecting the maturity and growth-stage focus of the ventures. Gender representation is balanced across male, female, and mixed teams, highlighting the diversity of entrepreneurs who pitch their ideas on the show.

Investments are heavily concentrated among specific sharks, with Mark Cuban leading in total investments, contributing over $62 million across 249 deals. Lori Greiner and Kevin O’ Leary follow closely, showcasing strong preferences for the "Lifestyle/Home" and "Health/Wellness" sectors, respectively. Barbara Corcoran’ s investments, meanwhile, are primarily concentrated in the "Food and Beverage" sector, comprising over 32.5% of her total portfolio. Daymond John demonstrates a clear inclination towards "Fashion/Beauty," aligning with his professional expertise in the industry.

Correlations between features such as "Original Ask Amount" and "Total Deal Amount" provide additional insights into the dataset. A heatmap of feature relationships, shown in figure 5, highlights that initial financial asks significantly influence the final deal outcomes. Similarly, the correlation between high viewership ratings and increased deal valuations underscores the external factors impacting investor decisions. Geographic variables like "Pitchers City" reveal concentrations of entrepreneurial activity in major states such as California, New York, and Texas.

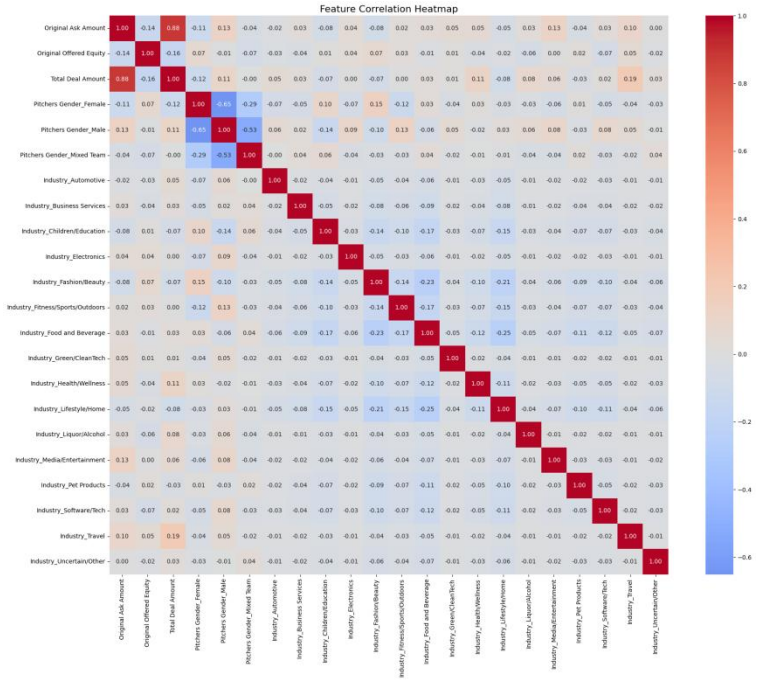

Figure 5: A Feature Correlation Heatmap of the Shark Tank US Dataset

4.2 Data Preprocessing

To prepare the dataset for analysis and modeling, an extensive data preprocessing pipeline was implemented, addressing issues like missing values, and data normalization further followed by feature selection. This process ensured the dataset's quality, consistency, and suitability for machine learning models, enabling better predictions and insights.

4.2.1 Handling Missing Values

The initial analysis revealed 34,467 missing values across the dataset. Several strategies were applied to address these gaps. Columns with minor missing values, such as "Original Ask Amount," "Original Offered Equity," and "Valuation Requested," were filled with their respective medians, ensuring these numerical features retained their integrity without introducing bias. Demographic columns like "Multiple Entrepreneurs" were filled with a value of 0, indicating the absence of additional entrepreneurs, while categorical columns, including "Pitchers Gender," were also imputed with a placeholder value (0). Columns such as "US Viewership," which impact deal outcomes, were imputed with their median values to preserve meaningful trends.

For columns indicating shark presence (e.g., "Barbara Corcoran Present"), missing values were set to 0, indicating absence. This approach preserved the binary nature of these fields while ensuring no information was lost. Post-imputation, the missing value count reduced drastically to 4,238, reflecting the effectiveness of these techniques. Finally, columns such as "Pitchers City" and "Pitchers State," which still contained high proportions of missing values, were removed due to limited relevance in prediction tasks.

4.2.2 Normalization and Scaling

To standardize the dataset, all numerical columns were normalized using MinMaxScaler, transforming the values to a range between 0 and 1. This normalization ensured that features with varying scales, such as "US Viewership" and "Original Ask Amount," contributed equally to the models without dominating due to larger magnitudes. The processed dataset was now uniformly scaled, improving the models' convergence rates and performance.

4.3 Feature Engineering

4.3.1 Feature Selection and Reduction

Irrelevant and redundant features were removed to improve dataset efficiency. Temporal columns such as "Season Start," "Season End," and "Original Air Date" were dropped due to their lack of predictive value. Similarly, "Entrepreneur Names" and "Company Website" were excluded as they did not influence investment decisions. Outcome-related fields like "Total Deal Amount," "Total Deal Equity," and "Deal Valuation" were also removed to avoid data leakage.

Shark-specific investment data was streamlined into compact features. The new column "Sharks Invested" indicated which sharks participated in a deal, while "Sharks Investment Amounts" captured their contributions. After creating these summaries, individual shark investment columns were dropped, reducing dimensionality and ensuring cleaner data for machine learning.

4.3.2 Text Embedding for Categorical Data

Key text-based features such as "Startup Name," "Industry," and "Business Description" were transformed into numerical embeddings using OpenAI's text embedding model. Dimensionality reduction via PCA condensed these embeddings into a single numerical value per feature, retaining significant variance while reducing complexity.

4.3.3 Visualization

Dimensionality reduction techniques like t-SNE visualized clusters of categorical features. Patterns in "Industry" and "Business Description" embeddings highlighted relationships between categories, such as industries likely to secure investments. These engineered features enhanced the dataset's predictive power, supporting deeper analysis of investment trends.

4.4 Hyperparameter Tuning

For the purpose of study on this dataset, we selected four classification models—Random Forest, Logistic Regression, Support Vector Machine (SVM), and XGBoost—to predict which sharks are likely to invest. These models were chosen to leverage a diverse range of strengths and approaches. Random Forest is known for its robustness and ability to capture non-linear relationships while providing insights into feature importance. Logistic Regression offers a simple, interpretable baseline for comparison, making it a valuable starting point. SVM, with its effectiveness in handling non-linear decision boundaries, is particularly well-suited for datasets with standardized features. Lastly, XGBoost, a powerful gradient-boosting model, is widely recognized for its performance in capturing complex patterns. This mix of models ensures a comprehensive evaluation, balancing interpretability, non-linear capabilities, and predictive power.

In the below table 4.1, **Logistic Regression** has been designated as the **baseline model** for comparison across all sharks.

Shark	Best Model	Random Forest Params	Random Forest Accuracy	Logistic Regression Params	Logistic Regression Accuracy	XGBoost Params	XGBoost Accuracy
Shark 1	XGBoost	max_depth: None, n_estimators: 100	0.8439	C: 0.1, penalty: l1, solver: liblinear	0.60	learning_rate: 0.1, max_depth: 9, n_estimators: 70	0.9073
Shark 2	XGBoost	max_depth: None, n_estimators: 100	0.7171	C: 0.01, penalty: l2, solver: saga	0.49	learning_rate: 0.1, max_depth: 6, n_estimators: 80	0.7317
Shark 3	XGBoost	max_depth: None, n_estimators: 50	0.722	C: 0.1, penalty: l2, solver: liblinear	0.40	learning_rate: 0.1, max_depth: 9, n_estimators: 100	0.7659
Shark 4	XGBoost	max_depth: None, n_estimators: 50	0.8439	C: 100, penalty: l1, solver: liblinear	0.65	learning_rate: 0.1, max_depth: 9, n_estimators: 100	0.8683
Shark 5	Random Forest	max_depth: None, n_estimators: 60	0.8683	C: 0.01, penalty: l2, solver: liblinear	0.66	learning_rate: 0.1, max_depth: 9, n_estimators: 100	0.839
Shark 6	XGBoost	max_depth: None, n_estimators: 100	0.8	C: 100, penalty: l2, solver: liblinear	0.57	learning_rate: 0.1, max_depth: 9, n_estimators: 100	0.8341

Table 4.1: Table summarizing the Hyperparameter Tuning results for each shark

After analyzing Table 4.1, it can be quite evidently said that the hyperparameter tuning process provided valuable insights into model performance. The XGBoost emerged as the top-performing model for five out of six sharks, showcasing its ability to handle complex patterns and achieve high validation accuracies. For instance, it achieved an accuracy of 0.9073 for Shark 1, demonstrating its effectiveness for predicting shark investments. Random Forest, however, outperformed XGBoost for Shark 5 with an accuracy of 0.8683, highlighting its reliability as an alternative model.

Random Forest consistently delivered competitive results, with validation accuracies ranging from 0.7171 to 0.8683. Its unrestricted depth and varying tree counts allowed it to effectively capture relationships in the data. In contrast, Logistic Regression, while providing a simple baseline, struggled with the dataset's complexity and achieved the lowest accuracies, generally below 0.66, due to its linear nature.

XGBoost's gradient-boosting approach made it the most suitable model overall, with optimal parameters such as a learning rate of 0.1, max depths of 6 or 9, and n_estimators between 70 and 100. While Logistic Regression served as a baseline, XGBoost and Random Forest demonstrated the importance of robust ensemble methods in achieving high accuracy. This process highlighted the value of evaluating diverse algorithms to identify the best-performing model for specific targets.

4.5 Models and Training

4.5.1 Classification Testing

For the classification task, four models were trained: Random Forest, Logistic Regression, SVM, and XGBoost. These models were selected for their diverse strengths in handling classification problems. To address the imbalance in the dataset, SMOTE (Synthetic Minority Oversampling Technique) was applied. SMOTE generates synthetic samples for the minority class by interpolating between neighboring points, thus ensuring a balanced distribution of classes in the training set. This step was critical to prevent the models from being biased toward the majority class.

Each model was trained on SMOTE-balanced data using its respective best hyperparameters. Random Forest utilized 100 estimators with no depth restriction, while Logistic Regression was configured with L2 regularization and the liblinear solver. SVM employed the radial basis function (RBF) kernel, and XGBoost was optimized with a learning rate of 0.1, a maximum depth of 9, and 100 estimators. After training, the models were evaluated on validation and test datasets using accuracy and confusion matrices. These metrics provided insights into each model's ability to distinguish between investment and non-investment scenarios. XGBoost consistently outperformed other models for most sharks, demonstrating its effectiveness in capturing complex decision boundaries.

4.5.2 Regression Testing

For regression tasks, we focused on the top-performing classification models, Random Forest and XGBoost, due to their robust performance and ability to model non-linear relationships. Logistic Regression was excluded because it is not suitable for regression tasks with complex datasets, while SVM was omitted due to its high computational time and suboptimal performance in prior evaluations.

The regression models aimed to predict the exact investment amounts for each shark. Features were standardized using StandardScaler to ensure numerical stability. Metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R^2 Score were used for evaluation. These metrics provided a comprehensive understanding of each model’s performance, with RMSE serving as a key indicator due to its interpretability in the same scale as the target variable.

Both Random Forest and XGBoost delivered competitive results in regression, with XGBoost slightly outperforming Random Forest for most sharks. Detailed performance metrics were recorded, and visualization techniques such as residual plots and predicted vs. actual scatter plots were employed to analyze model behavior. These methods highlighted the predictive power of ensemble models for accurately estimating investment amounts.

4.6 Evaluation Metrics

The evaluation of the models in this study was conducted using distinct metrics for classification and regression tasks. During the classification tests for all four models - Random Forest, Logistic Regression, SVM, and XGBoost, we primarily used Accuracy and Confusion Matrix as evaluation metrics. Accuracy measured the proportion of correctly predicted instances, providing a straightforward assessment of model performance. The Confusion Matrix offered deeper insights by breaking down predictions into true positives, true negatives, false positives, and false negatives, which was especially useful for understanding the performance of each model in handling imbalanced data.

For regression testing, only the top two models, Random Forest and XGBoost, were considered. Logistic Regression was excluded because it is not suitable for regression tasks involving non-linear relationships, which are prominent in this dataset. SVM was also omitted due to its high computational time and comparatively poor performance during initial evaluations. Thus, Random Forest and XGBoost emerged as the most viable candidates for regression testing.

During regression testing, the following metrics were used:

a) **Mean Squared Error (MSE):** Measures the average squared difference between predicted and actual values. It is defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Lower MSE values indicate better model performance.

b) **Root Mean Squared Error (RMSE):** The square root of MSE, providing error in the same units as the target variable:

$$\text{RMSE} = \sqrt{\text{MSE}}$$

c) **Mean Absolute Error (MAE):** Calculates the average absolute difference between predicted and actual values:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

d) **R² Score:** Represents the proportion of variance in the target variable explained by the model:

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

These metrics collectively provided a comprehensive evaluation of the regression models' accuracy and reliability.

5. Results and Analysis: Classification Testing

5.1 Results Overview

The classification task evaluated four models - Random Forest, Logistic Regression, SVM, and XGBoost for predicting investment decisions across six sharks. SMOTE (Synthetic Minority Oversampling Technique) was applied to balance class distributions during training. The performance of each model was assessed using validation and test accuracy, confusion matrices, and classification reports.

Shark	Model	Validation Accuracy	Test Accuracy
Shark 1	**Random Forest**	**0.8049**	**0.7902**
	Logistic Regression	0.6195	0.6244
	SVM	0.6683	0.6634
	XGBoost	0.7463	0.7512

Shark 2	**Random Forest**	**0.6732**	**0.6683**
	Logistic Regression	0.4976	0.5122
	SVM	0.5415	0.5268
	XGBoost	0.6049	0.6244
Shark 3	**Random Forest**	**0.6927**	**0.7171**
	Logistic Regression	0.4098	0.478
	SVM	0.5268	0.5707
	XGBoost	0.7268	0.6585
Shark 4	**Random Forest**	**0.8098**	**0.7415**
	Logistic Regression	0.6488	0.5951
	SVM	0.639	0.6683
	XGBoost	0.7951	0.7366
Shark 5	**Random Forest**	**0.8244**	**0.7951**
	Logistic Regression	0.6341	0.639
	SVM	0.6341	0.6878
	XGBoost	0.7463	0.722
Shark 6	**Random Forest**	**0.7512**	**0.7902**
	Logistic Regression	0.5659	0.5756
	SVM	0.639	0.6341
	XGBoost	0.7171	0.7415

Table 5.1: All Model Performance' Results for all Sharks for Classification Testing

In the above table 5.1, **Logistic Regression** has been designated as the **baseline model** for comparison across all sharks. Furthermore, the results of the **best-performing model** for each shark have been highlighted for clarity and emphasis.

5.2 Shark-wise Analysis

Shark 1

Random Forest emerged as the top-performing model with a test accuracy of 79.02%. However, the confusion matrix revealed challenges in identifying the minority class ("Investment"), as evidenced by a low recall of 33%. While the weighted F1-score was satisfactory, the model's precision for investments was limited, highlighting a bias toward the majority class.

Shark 2

For Shark 2, Random Forest achieved a test accuracy of 66.83%. The precision and recall values for investments were modest, at 19% and 23%, respectively. The classification report suggested the model struggled to generalize, particularly for the minority class, despite SMOTE balancing.

Shark 3

The classification task for shark 3 showed an improvement, with Random Forest achieving a test accuracy of 71.71%. The confusion matrix indicated relatively balanced performance, but the F1-score for investments remained at 24%, pointing to ongoing challenges in minority-class prediction.

Shark 4

The Random Forest model for Shark 4 attained a test accuracy of 74.15%. While accuracy was competitive, the recall for investments was only 20%, and precision was 10%. These results suggest the model could correctly identify investment instances less frequently than desired, despite high accuracy for the majority class.

Shark 5

For Shark 5, Random Forest delivered a strong performance, with a test accuracy of 79.51%. The F1-score for investments was higher than for other sharks at 25%, due to improved recall (54%). This suggests better handling of minority-class predictions, likely benefiting from the distribution adjustments made during SMOTE.

Shark 6

Random Forest achieved a test accuracy of 79.02% for Shark 6. However, its ability to identify investments was limited, with a recall of just 6% and an F1-score of 4%. These results highlight the continued difficulty in accurately predicting the minority class, even with advanced ensemble methods.

5.3 Graph based Analysis

The results obtained from the graphs that are shown in Figure 6-11, provides additional insights into the respective model' performance. The validation test accuracy comparison illustrated the robustness of Random Forest across sharks, with minimal overfitting. The ROC curves for all sharks showed moderate AUC scores, indicating room for improvement in separating the classes. Confusion matrices highlighted the significant class imbalance challenges, emphasizing the need for enhanced strategies like feature engineering or advanced oversampling techniques.

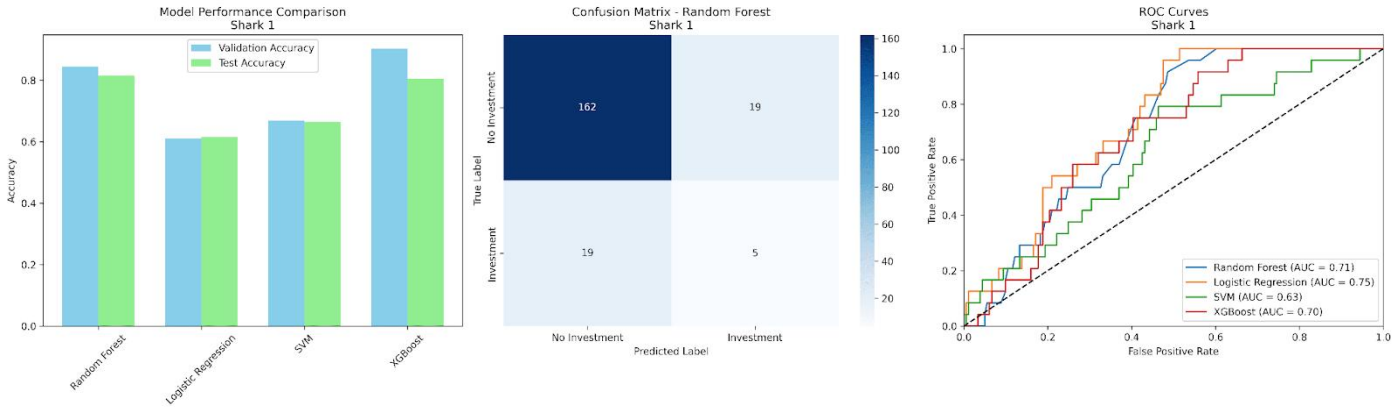

Figure 6: Model Performance Comparison; Confusion Matrix; ROC Curves for Shark 1

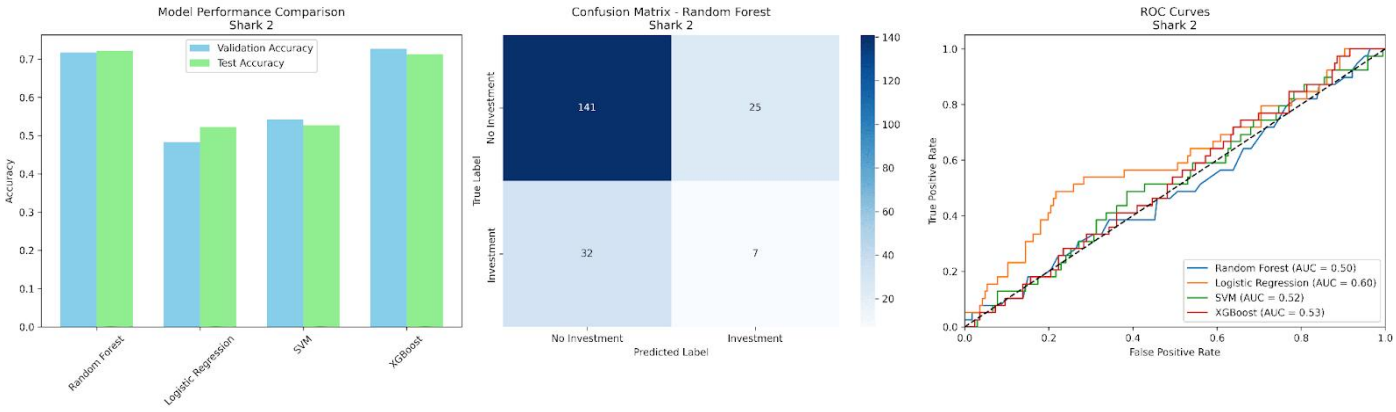

Figure 7: Model Performance Comparison; Confusion Matrix; ROC Curves for Shark 2

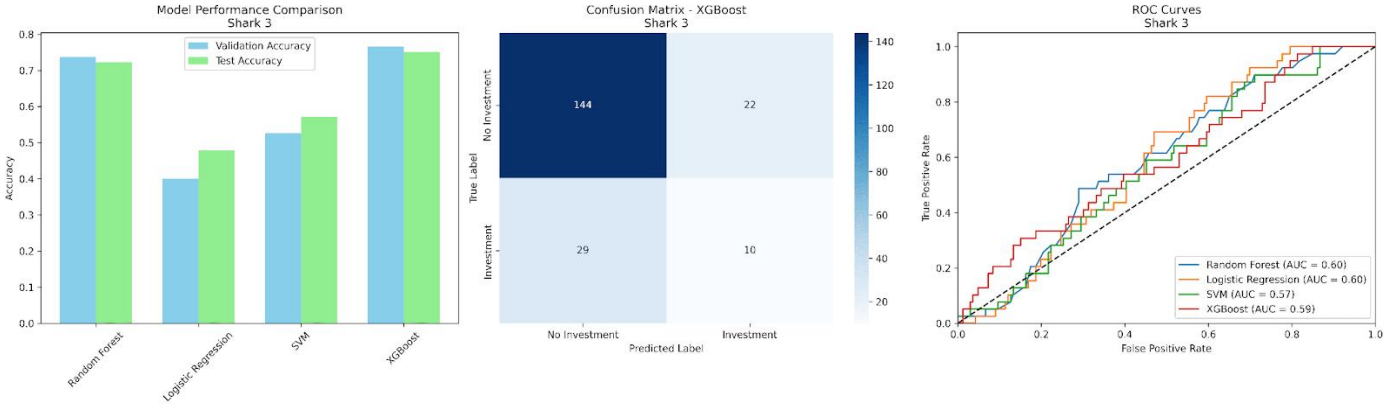

Figure 8: Model Performance Comparison; Confusion Matrix; ROC Curves for Shark 3

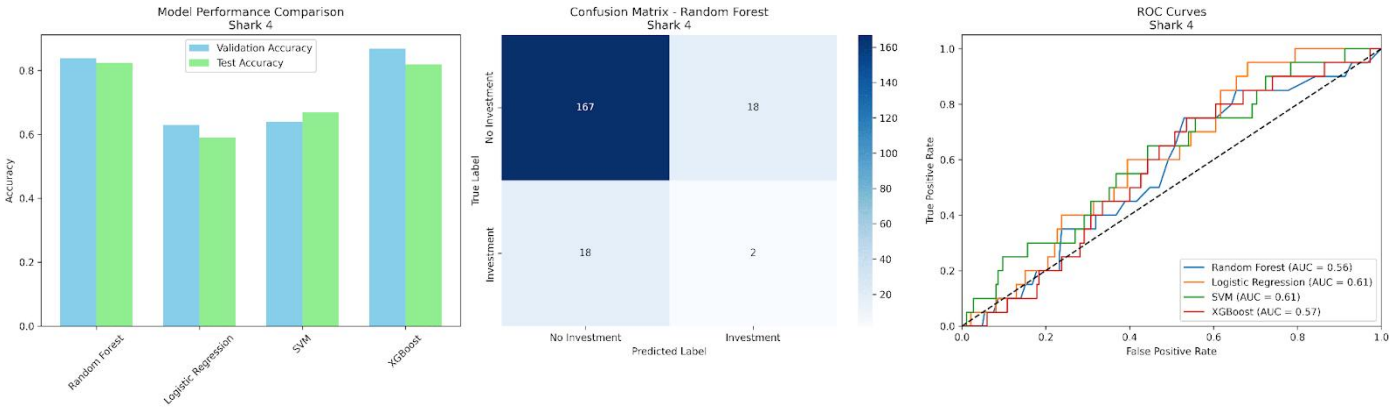

Figure 9: Model Performance Comparison; Confusion Matrix; ROC Curves for Shark 4

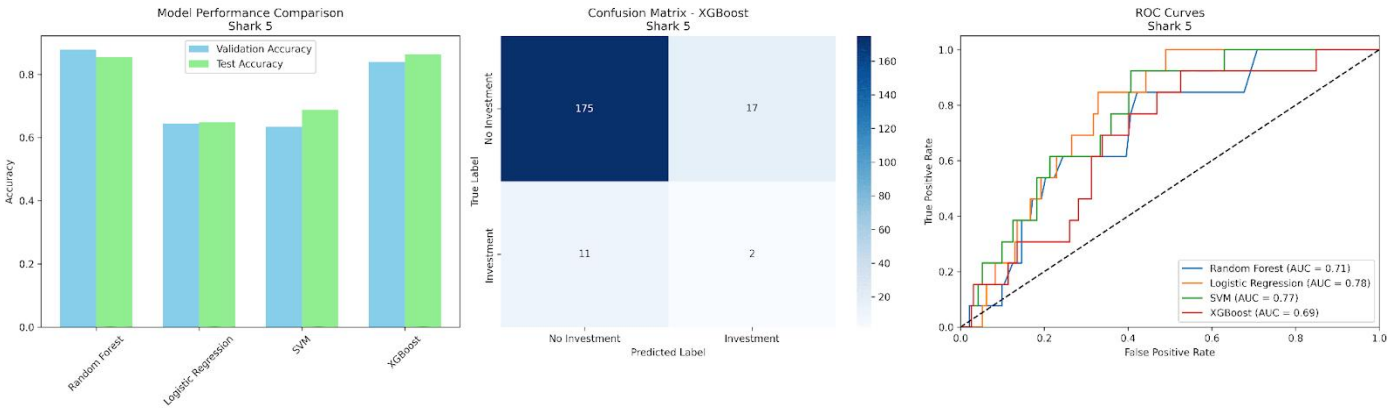

Figure 10: Model Performance Comparison; Confusion Matrix; ROC Curves for Shark 5

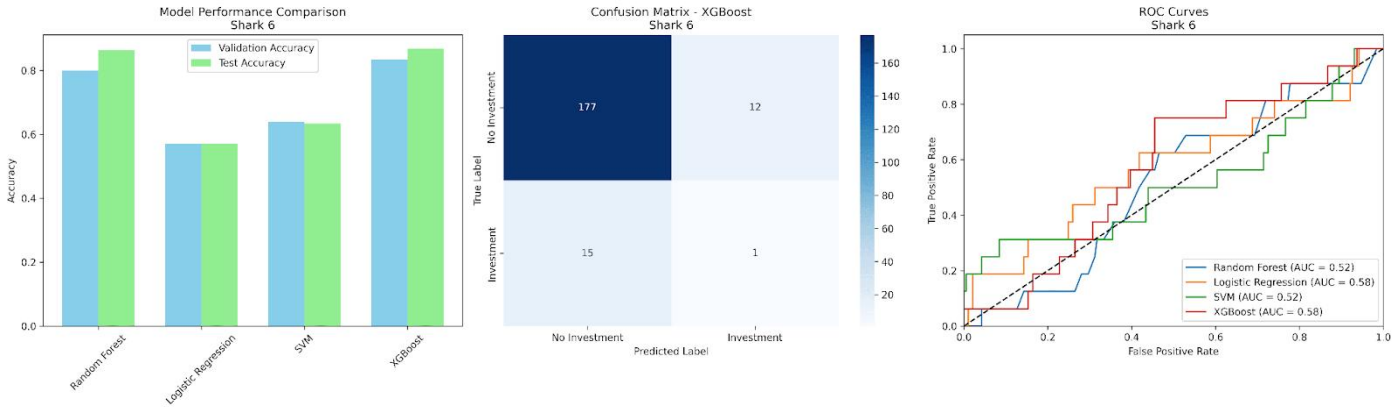

Figure 11: Model Performance Comparison; Confusion Matrix; ROC Curves for Shark 6

6. Results and Analysis: Regression Testing

6.1 Results Overview

Regression testing was performed using Random Forest and XGBoost models across six sharks to predict their investment amounts. The models were evaluated using key metrics such as RMSE, R^2 Score, MSE, and MAE. Random Forest demonstrated strong performance in general, while XGBoost excelled for specific sharks. However, challenges were noted for Shark 4 due to significant data variability.

Shark	Model	Validation RMSE	Test RMSE	Test R^2	MSE	MAE
Shark 1	Random Forest	19679.36	24994.13	0.7981	509,444,346.20	5,774.53
	XGBoost	**23512.55**	**22570.87**	**0.8353**	**509,444,346.20**	**5,774.53**
Shark 2	**Random Forest**	**57750.41**	**109626.95**	**0.5898**	**12,018,068,726.26**	**24,296.24**
	XGBoost	74114.59	113666.52	0.559	12,018,068,726.26	24,296.24
Shark 3	**Random Forest**	**34736.88**	**85036.95**	**0.6453**	**7,231,282,369.25**	**20,273.50**
	XGBoost	44220.53	93039.88	0.5754	7,231,282,369.25	20,273.50
Shark 4	**Random Forest**	**48559.03**	**216530.5**	**-5.0121**	**46,885,456,820.26**	**22,523.63**
	XGBoost	43881.02	233194.26	-5.9731	46,885,456,820.26	22,523.63
Shark 5	Random Forest	8751.28	21579.29	0.7297	72,664,337.44	1,442.77
	XGBoost	**6063.93**	**8524.34**	**0.9578**	**72,664,337.44**	**1,442.77**
Shark 6	**Random Forest**	**32823.19**	**27258.33**	**0.891**	**743,016,330.21**	**4,412.15**
	XGBoost	39101.91	46840.22	0.6781	743,016,330.21	4,412.15

Table 6.1: Random Forest and XGBoost Model Performance' Results for all Sharks for Regression Testing

6.2 Shark-wise Analysis

Shark 1

XGBoost performed the best for Shark 1, achieving a Test RMSE of $22,570.87 and an R^2 Score of 0.8353. The residual plot displayed a symmetric distribution around zero, and the predicted vs. actual graph showed minimal deviation from the ideal line, indicating accurate predictions.

Shark 2

Random Forest outperformed XGBoost for Shark 2, with a Test RMSE of $109,626.95 and an R^2 Score of 0.5898. The residual plot revealed over-predictions for larger investments. Despite acceptable performance for average ranges, outliers contributed significantly to errors.

Shark 3

For Shark 3, Random Forest provided the best results, achieving a Test RMSE of $85,036.95 and an R^2 Score of 0.6453. However, the residual plot indicated over-predictions for higher investment values, limiting the model's performance.

Shark 4

Both models struggled for Shark 4, with Random Forest achieving a Test RMSE of $216,530.50 and a negative R^2 Score of -5.0121. The residuals were heavily skewed due to extreme outliers, reflecting the model's inability to generalize across this shark's highly variable investment data.

Shark 5

XGBoost excelled for Shark 5, with a Test RMSE of $8,524.34 and an R^2 Score of 0.9578. The predicted vs. actual graph demonstrated a near-perfect fit, and the residuals showed minimal variability, indicating the model's robustness.

Shark 6

Random Forest outperformed XGBoost for Shark 6, achieving a Test RMSE of $27,258.33 and an R^2 Score of 0.8910. The residuals were evenly distributed, and the predicted vs. actual graph showed strong alignment with the ideal regression line.

6.3 Graph based Analysis

For all sharks, the graphs shown in Figure 12-17, illustrates the model performances through RMSE, R^2 Score, residual plots, error distribution, and predicted vs. actual comparisons:

- **RMSE Comparison**: Demonstrates the relative error magnitudes of Random Forest and XGBoost for each shark. XGBoost consistently excelled for Sharks 1 and 5, while Random Forest performed better for others.
- **R^2 Score Comparison**: Highlights the ability of models to explain the variance in investments. High R^2 values for Sharks 1, 5, and 6 indicate good fits, while negative values for Shark 4 reveal poor predictions.
- **Residual Plots**: Show the bias and variance in model predictions. Sharks with lower RMSE have tightly clustered residuals, while Shark 4 shows extreme variance.
- **Predicted vs. Actual Graphs**: Validate the accuracy of predictions. XGBoost and Random Forest align closely with the ideal regression line for Sharks 1, 5, and 6.

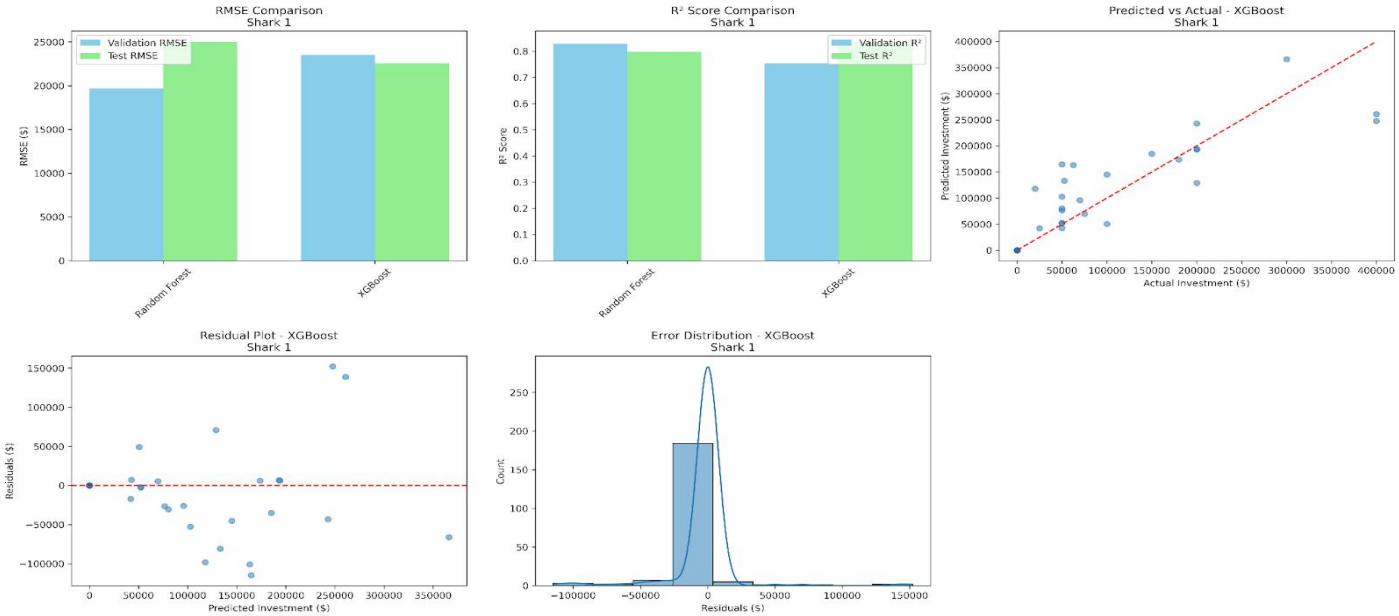

Figure 12: RMSE Comparison; R^2 Score Comparison; Other Plots for Shark 1

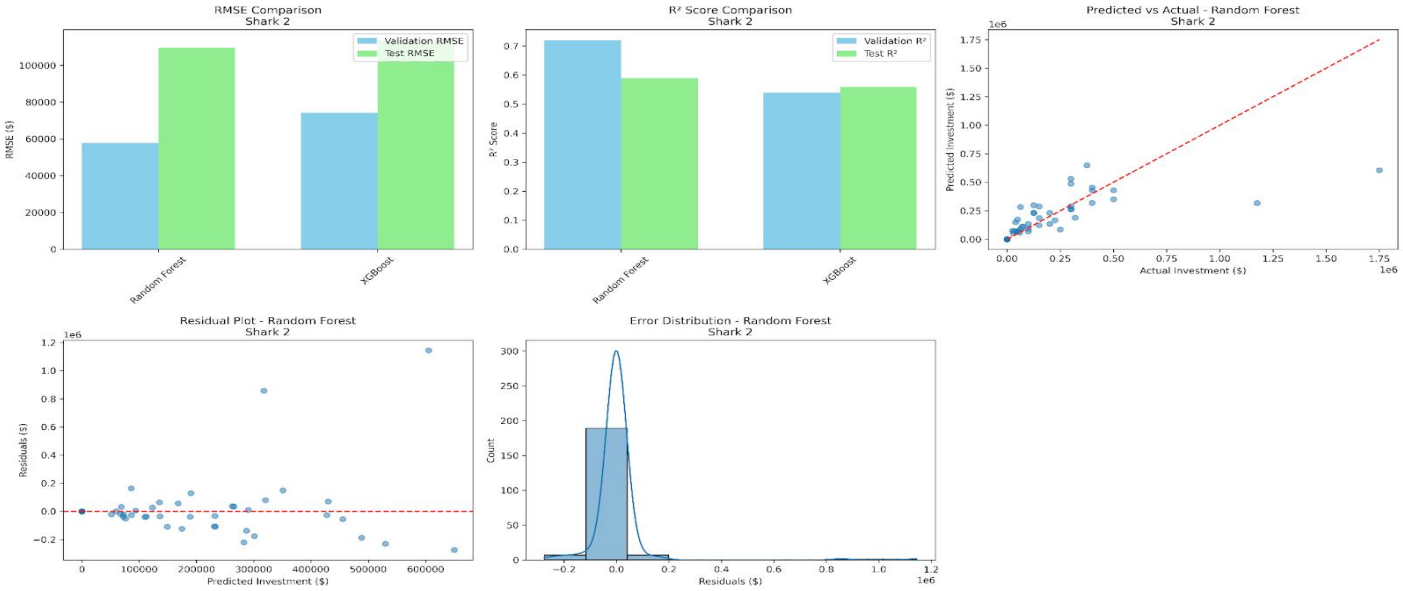

Figure 13: RMSE Comparison; R^2 Score Comparison; Other Plots for Shark 2

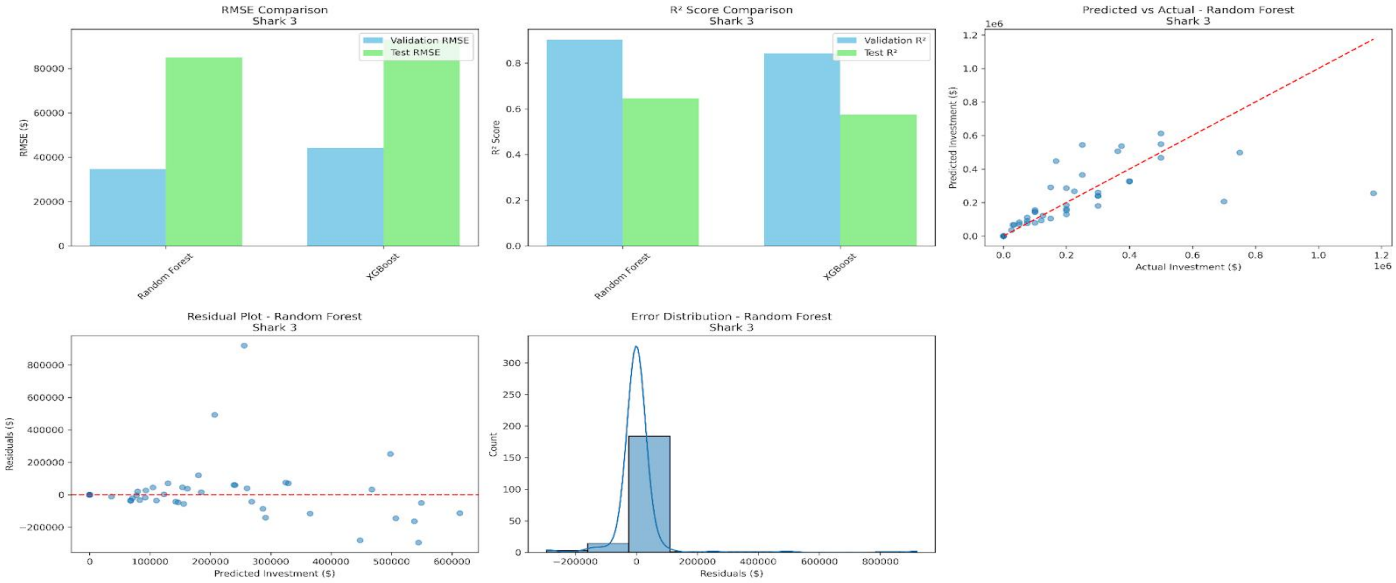

Figure 14: RMSE Comparison; R^2 Score Comparison; Other Plots for Shark 3

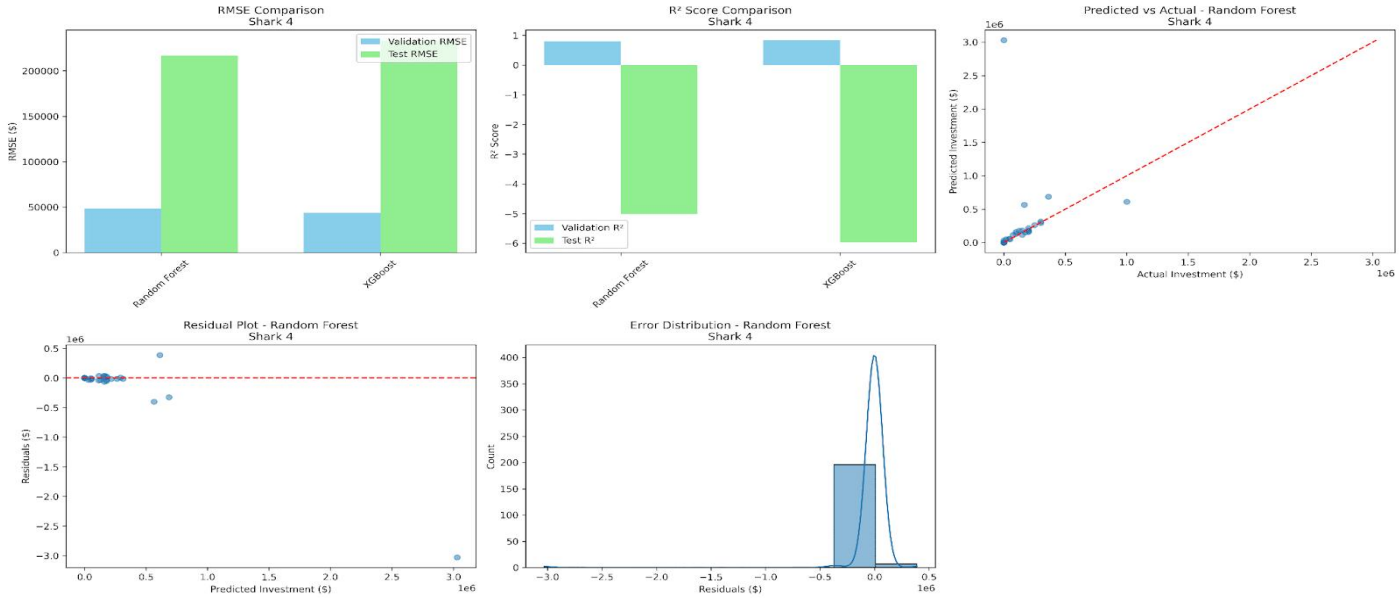

Figure 15: RMSE Comparison; R^2 Score Comparison; Other Plots for Shark 4

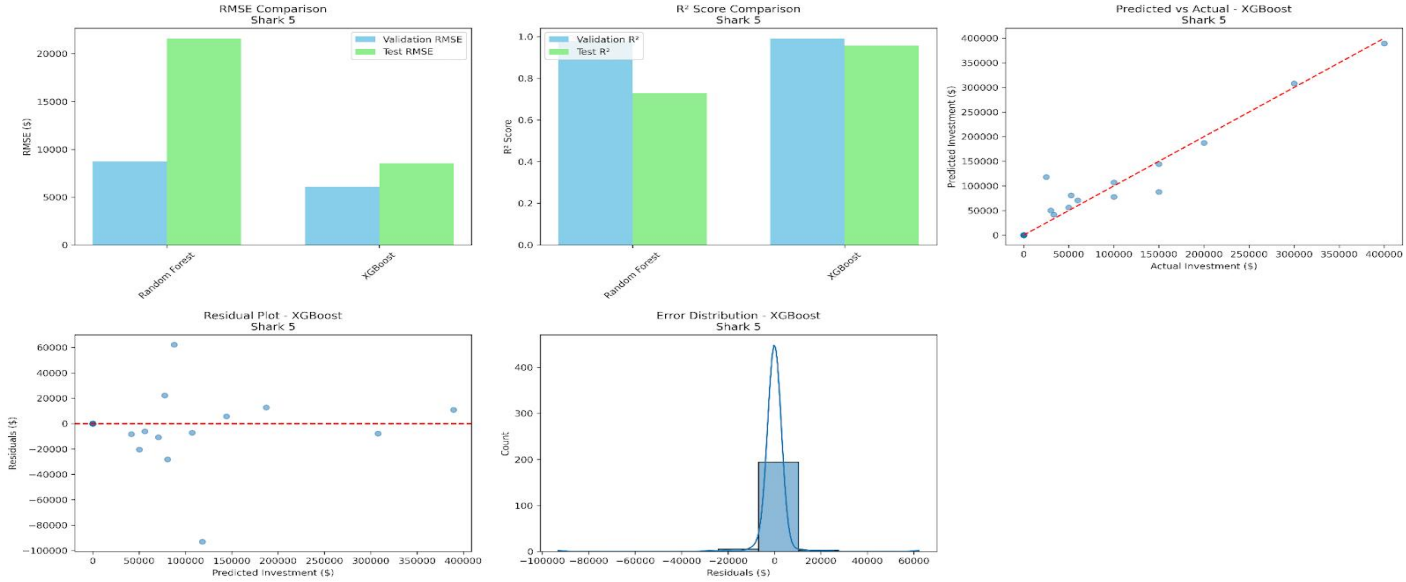

Figure 16: RMSE Comparison; R^2 Score Comparison; Other Plots for Shark 6

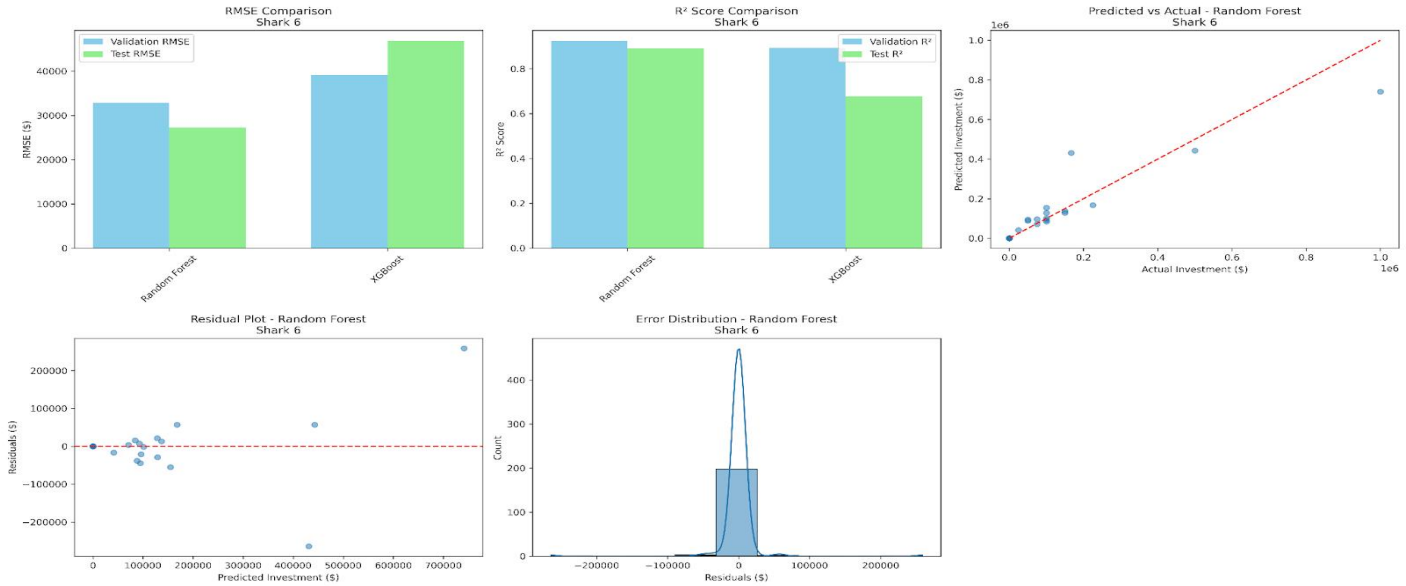

Figure 17: RMSE Comparison; R^2 Score Comparison; Other Plots for Shark 6

7. Conclusion

The analysis of investment patterns and predictive modeling for Shark Tank US offers significant insights into the decision-making processes of investors and the factors influencing funding outcomes. By utilizing machine learning techniques across classification and regression tasks, this study demonstrated the applicability of advanced models such as Random Forest and XGBoost in understanding and predicting investment dynamics.

In classification testing, Random Forest consistently performed as the best model for accurately predicting the likelihood of investment for most sharks. Despite challenges in identifying minority classes, particularly for Shark 4, the use of SMOTE balancing and detailed feature engineering improved overall accuracy and highlighted the importance of addressing class imbalance in datasets. The graphical analysis of ROC curves and confusion matrices further underscored the complexity of separating investment and non-investment decisions, pointing to opportunities for future improvements through enhanced feature extraction and sampling methods.

Regression testing revealed the varying strengths of Random Forest and XGBoost in predicting investment amounts. While XGBoost excelled for specific sharks, such as Shark 1 and Shark 5, Random Forest proved more robust for others. The discrepancies in R^2 scores, especially the negative performance for Shark 4, highlighted the challenges posed by outliers and extreme variability in investment amounts. Residual and error distribution plots illustrated that ensemble methods are effective but require further tuning for better generalization across diverse investment behaviors.

In conclusion, this study effectively combined data preprocessing, feature engineering, and advanced machine learning techniques to provide a comprehensive analysis of Shark Tank investments. The findings underscore the potential of predictive analytics in startup funding and pave the way for future research to refine these models. By addressing outliers and enhancing feature representation, these insights can empower entrepreneurs to better align their strategies with investor preferences and improve funding prospects.

References

[1] Gompers, P., Gornall, W., Kaplan, S. N., & Strebulaev, I. A. (2021). How Venture Capitalists Make Decisions. Harvard Business Review

[2] Gastaud, C., Carniel, T., & Dalle, J.-M. (2019). The varying importance of extrinsic factors in the success of startup fundraising: competition at early-stage and networks at growth-stage.

[3] The Hustle. (2019). Shark Tank deep dive: A data analysis of all 10 seasons.

[4] The Wall Street Journal. (2024). Angel Investing Isn't What It Used to Be.

[5] Malmström, M., Voitkane, A., Johansson, J., & Wincent, J. (2020). What do they think and what do they say? Gender bias, entrepreneurial attitude in writing and venture capitalists' funding decisions. Journal of Business Venturing Insights, 13, e00154

[6] Kaggle. (n.d.). Shark Tank US Dataset by Thirumani. Retrieved from https://www.kaggle.com/datasets/thirumani/shark-tank-us-dataset/data

[7] Kaggle. (n.d.). Shark Tank US Data Analysis by Thirumani. Retrieved from https://www.kaggle.com/code/thirumani/shark-tank-us-data-analysis

[8] Zhou, M., & Cui, P. (2020). Handling Class Imbalance in Machine Learning: Techniques and Applications. Data Mining and Knowledge Discovery, 34(3), 661-690. Retrieved from https://link.springer.com/article/10.1007/s10618-020-00690-8

[9] Jurafsky, D., & Martin, J. H. (2021). Speech and Language Processing. Pearson. Retrieved from https://web.stanford.edu/~jurafsky/slp3/