Predicting Startup Success

Data Science for Business - Project

by

Rugved Mhatre

Sakshee Sawant

Yash Jain

New York University

Stern School of Business

May, 2025

Contents

1 Introduction

Startups are inherently high-risk ventures, with the majority failing to achieve long-term success. For venture capital firms like Sequoia Capital, which operate in an environment of uncertainty and fierce competition, the ability to identify high-potential startups early is both a strategic advantage and a critical need.

Traditionally, venture investing relies heavily on qualitative assessments—such as founder pedigree, market narratives, and product differentiation—supplemented by a limited set of operational and financial metrics. While these heuristics have historically driven many successful investments, they are often subjective, time-intensive, and susceptible to cognitive bias.

This project, Project RiskLens, aims to augment Sequoia's early-stage investment process through data science and machine learning. By leveraging historical startup data and applying modern predictive techniques, our goal is to develop a model that estimates the likelihood of a startup's long-term success. This tool is not intended to replace human intuition, but to act as a data-driven filter that enhances speed, accuracy, and consistency in evaluating early-stage opportunities.

In this report, we detail the end-to-end data science pipeline—from data acquisition and feature engineering to model development, evaluation, and deployment considerations. The result is a practical tool that can assist Sequoia Capital in prioritizing high-potential startups and optimizing their investment portfolio with greater confidence.

1.1 Business Understanding

Venture capital (VC) operates on a high-risk, high-reward paradigm. A small number of investments—often fewer than 10%—generate the majority of a fund's returns, while the vast majority either break even or fail entirely. Despite rigorous diligence processes, the early-stage investment landscape remains fundamentally uncertain. The stakes are high: a single unicorn can return 100x, while a failed investment ties up capital and opportunity.

To illustrate the scale: U.S. VC firms invest over $150 billion annually, with $50 billion dedicated to early-stage startups. Yet over 75% of VC-backed startups fail to deliver returns. For a $1 billion fund, that could mean $750 million committed to ventures that never return profit.

Small improvements in predictive capability can have outsized effects on fund performance. Consider a VC firm with a 10% success rate—where "success" is defined as a 5x return or greater. A 2% absolute improvement—raising the success rate to 12%—translates to 2 additional successful investments per 100. If each winning startup generates $25–50 million in return, this improvement could yield an additional $100–200 million in profits, all without increasing capital deployed.

Such impact compounds over a fund lifecycle. Scaled across hundreds of investments, a data-informed model that enhances early filtering can significantly improve return on investment, reduce missed opportunities, and provide a competitive edge in sourcing high-potential ventures earlier.

1.2 Project Objective

The objective of Project RiskLens is to develop a supervised machine learning model that:

- Estimates the probability of a startup's success based on structured input features;
- Identifies key drivers of success from historical data (e.g., industry sector, funding history, location);

- Supports investor decision-making by offering an interpretable, transparent scoring mechanism.

This system will help Sequoia prioritize due diligence, uncover hidden opportunities, and allocate attention more efficiently during early-stage deal evaluation.

1.3 Stakeholders and Impact

The primary stakeholders for this project include:

- Sequoia Capital's investment partners and analysts, who will use the model to enhance screening processes;
- Portfolio managers, who may use the model to benchmark startups over time;
- Data and platform teams, who can integrate the model into internal dashboards and workflows.

The long-term impact lies in enabling Sequoia to:

- Identify "sleepers" that may otherwise be overlooked,
- Reduce the noise from biased or overly subjective filters,
- Increase overall fund performance through better portfolio selection.

2 | Data Understanding

2.1 Data Source

This project utilizes the Crunchbase startup dataset[1], a comprehensive and widely-recognized resource for startup data across the globe. The dataset contains 54,294 entries, each representing a distinct startup, with 39 attributes capturing key operational, financial, and organizational details.

The data spans a wide historical range—from 1902 to 2014—offering a rich temporal view of startup evolution, funding trends, and exit outcomes. This historical breadth allows for the identification of long-term patterns and shifts in startup success factors over time.

Each startup entry includes variables such as the founding year, location, market sector, funding rounds, amounts, and the status indicators (operating, acquired and closed).

The dataset also offers geographic diversity, covering startups from multiple countries and regions. However, there is a noticeable concentration in well-established startup ecosystems such as the United States, United Kingdom, Canada, and parts of Western Europe, reflecting the source's primary coverage areas.

Overall, the Crunchbase dataset provides a robust foundation for predictive modeling, allowing us to capture key signals that may correlate with a startup's likelihood of achieving a successful exit via Merger & Acquisition (M&A) or Initial Public Offering (IPO).

[1] Kaggle Dataset Link

2.2 Data Features

Numerical Features

Feature	**Description**
funding_rounds	Number of funding rounds the startup has received
funding_total_usd	Total funding received (USD)
seed	Amount of seed funding raised (USD)
venture	Amount of venture funding raised (USD)
equity_crowdfunding	Amount raised through equity crowdfunding (USD)
undisclosed	Amount raised in undisclosed funding rounds (USD)
convertible_note	Funding through convertible notes (USD)
debt_financing	Funding through debt financing (USD)
angel	Amount raised through angel investments (USD)
grant	Amount received via grants (USD)
private_equity	Private equity raised (USD)
post_ipo_equity	Equity funding received after IPO (USD)
post_ipo_debt	Debt funding received after IPO (USD)
secondary_market	Funding received through secondary market (USD)
product_crowdfunding	Funding through product crowdfunding (USD)
round_A - round_H	Amount raised in each funding round Series A to H (USD)

Text Features

Feature	Description
permalink	URL identifying the startup on Crunchbase
name	Name of the startup
homepage_url	Homepage of the startup
category_list	Categories describing the startup
market	Market targeted by the startup
country_code	Country where the startup is based
state_code	State code (for U.S. startups)
region	Geographical region of the startup
city	City where the startup is based

Date-Time Features

Feature	Description
founded_year	Year the startup was founded
founded_at	Exact date the startup was founded
founded_month	Year and month the startup was founded
founded_quarter	Quarter the startup was founded (e.g., 2012-Q1)
first_funding_at	Date of first funding round
last_funding_at	Date of most recent funding round

Target Features

Feature	Description
status	Current status of the startup (e.g., operating, acquired, closed)

2.3 Data Imbalance

Although it is well-established that a large majority of startups fail—studies estimate that over 90% do not yield substantial returns—the dataset used in this analysis reflects the opposite trend. In our Crunchbase-derived dataset, the number of startups labeled as *closed* (failures) is significantly lower compared to those labeled as *operating* or *acquired* (successes). This imbalance introduces a critical modeling challenge.

Upon inspection, we found that only around 5% of the startups in the dataset are marked as *closed*, while approximately 87% are still *operating* and 8% have been *acquired* (Figure 2.1). This contradicts real-world statistics and distorts the learning process for most classification models, which are often biased toward the majority class unless corrected.

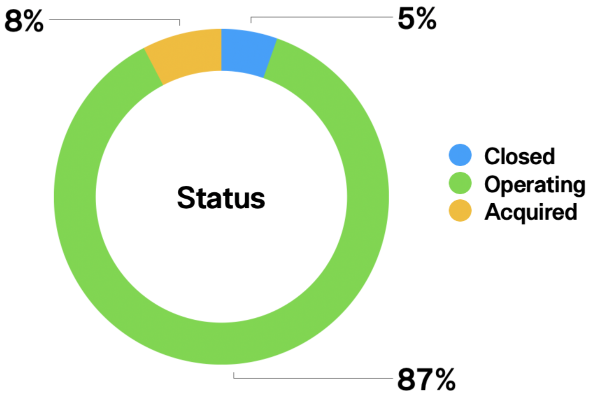


Figure 2.1: Startup Status Distribution

This skewed distribution arises largely due to the nature of data collection on platforms like Crunchbase [1]. Since Crunchbase relies heavily on public disclosures and user-submitted information, startups that are active, funded, or acquired are more likely to be featured and updated. Failed or shuttered startups are often underreported or removed from the platform, creating a strong **survivor bias**.

Survivor bias in this context means that the dataset disproportionately includes startups that have survived long enough to attract funding or media attention, thereby making it seem as though failure is rare. As a result, models trained on this data without correction may generalize poorly to real-world investment decisions where startup failure is far more common.

To address this, we employed feature engineering and data balancing techniques to balance the classes. These techniques help mitigate the bias during training and allow the model to better distinguish between truly promising startups and those more likely to fail.

2.4 Preliminary Observations

Before modeling could begin, a thorough review of the dataset revealed several challenges that required significant data preparation and engineering effort.

First, the dataset contains a substantial amount of missing values across multiple features. Fields such as *founded_year*, *funding_total_usd*, and several funding round-specific columns have numerous null entries, which needed to be addressed through imputation or exclusion. Without this cleanup, most models would return unreliable predictions.

In addition to cleaning, the dataset also lacked several key features that are typically influential in predicting startup success—such as team background, financial KPIs, and clear industry groups. As a result, we had to engineer new features from existing data to capture signals like funding velocity, funding stage progression, and industry groups. These transformations helped enhance the model's predictive power but could not fully compensate for the missing business-critical variables.

Another major limitation is that the dataset only includes data up to the year 2014. This historical restriction means the model does not incorporate recent shifts in startup behavior, funding trends, or macroeconomic factors—reducing its ability to generalize to today's startup ecosystem.

Furthermore, the dataset is primarily composed of publicly available fields such as funding amounts and market categories. Attempts to acquire richer data—such as founder experience, revenue growth, user metrics, or number of patents—were largely unsuccessful due to high commercial licensing costs or lack of public disclosure. Much of this information resides behind proprietary paywalls, limiting the scope of open-source modeling efforts.

Given these constraints, we proceeded with developing a predictive model using the available data, applying feature engineering, balancing techniques, and imputation strategies to extract as much signal as possible from the limited inputs. While the resulting model offers valuable insights, it must be interpreted with an understanding of these foundational limitations.

3 | Data Preparation

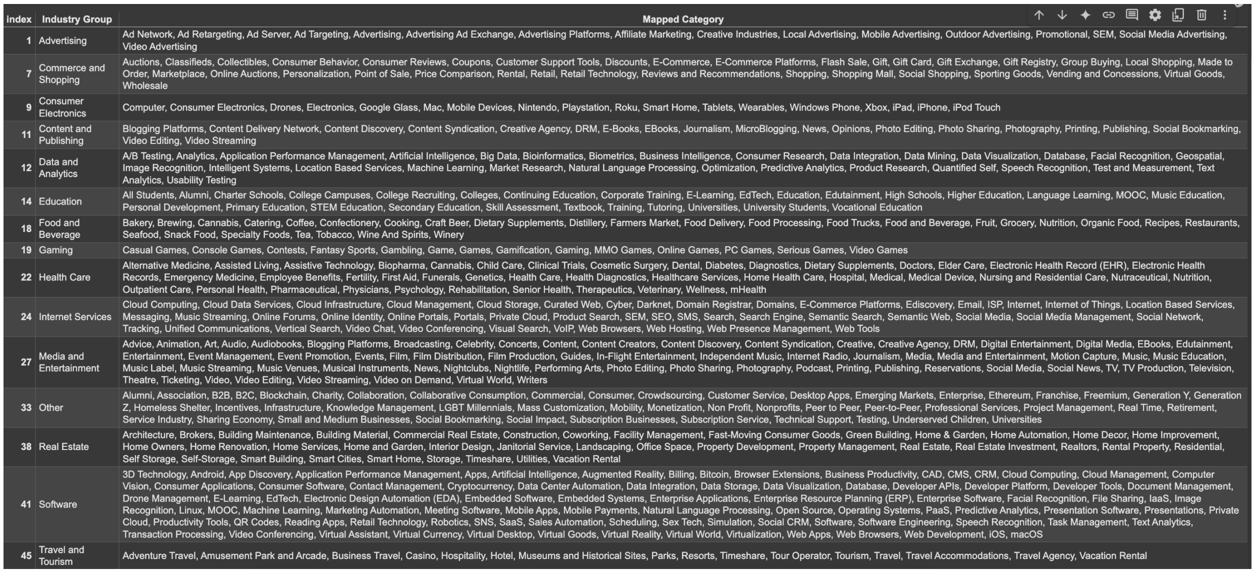
The dataset, while rich in startup metadata, was initially very messy and required extensive pre-processing before it could be used for modeling. Many columns contained missing values, inconsistent formats, erroneous entries, and outliers that would have significantly impacted model performance if left unaddressed.

In this section, we outline the complete set of data cleaning and preparation steps we performed. This includes handling missing values, normalizing formats, converting categorical variables, engineering new features, and addressing the data imbalance discussed earlier. Each of these steps played a critical role in ensuring the dataset was reliable, consistent, and suitable for training predictive models.

Without this rigorous data preparation phase, the model would have learned from noisy, biased, and incomplete inputs—leading to poor generalization and misleading results. Our goal was to maximize the signal-to-noise ratio and create a foundation that reflects real-world investment scenarios as accurately as possible, given the available data.

3.1 Creation of Industry Groups

One of the first issues that stood out in the raw dataset was the inconsistency in how startup sectors and markets were labeled. The *category_list* column contained multiple values separated by the '|' character, with no enforced structure or standard taxonomy. Similarly, the *market* col-



index	Industry Group	Mapped Category
1	Advertising	Ad Network, Ad Retargeting, Ad Server, Ad Targeting, Advertising, Advertising Ad Exchange, Advertising Platforms, Affiliate Marketing, Creative Industries, Local Advertising, Mobile Advertising, Outdoor Advertising, Promotional, SEM, Social Media Advertising, Video Advertising
7	Commerce and Shopping	Auctions, Classifieds, Collectibles, Consumer Behavior, Consumer Reviews, Coupons, Customer Support Tools, Discounts, E-Commerce, E-Commerce Platforms, Flash Sale, Gift, Gift Card, Gift Exchange, Gift Registry, Group Buying, Local Shopping, Made to Order, Marketplace, Online Auctions, Personalization, Point of Sale, Price Comparison, Rental, Retail, Retail Technology, Reviews and Recommendations, Shopping, Shopping Mall, Social Shopping, Sporting Goods, Vending and Concessions, Virtual Goods, Wholesale
9	Consumer Electronics	Computer, Consumer Electronics, Drones, Electronics, Google Glass, Mac, Mobile Devices, Nintendo, Playstation, Roku, Smart Home, Tablets, Wearables, Windows Phone, Xbox, iPad, iPhone, iPod Touch
11	Content and Publishing	Blogging Platforms, Content Delivery Network, Content Discovery, Content Syndication, Creative Agency, DRM, E-Books, EBooks, Journalism, MicroBlogging, News, Opinions, Photo Editing, Photo Sharing, Photography, Printing, Publishing, Social Bookmarking, Video Editing, Video Streaming
12	Data and Analytics	A/B Testing, Analytics, Application Performance Management, Artificial Intelligence, Big Data, Bioinformatics, Biometrics, Business Intelligence, Consumer Research, Data Integration, Data Mining, Data Visualization, Database, Facial Recognition, Geospatial, Image Recognition, Intelligent Systems, Location Based Services, Machine Learning, Market Research, Natural Language Processing, Optimization, Predictive Analytics, Product Research, Quantified Self, Speech Recognition, Test and Measurement, Text Analytics, Usability Testing
14	Education	All Students, Alumni, Charter Schools, College Campuses, College Recruiting, Colleges, Continuing Education, Corporate Training, E-Learning, EdTech, Education, Edutainment, High Schools, Higher Education, Language Learning, MOOC, Music Education, Personal Development, Primary Education, STEM Education, Secondary Education, Skill Assessment, Textbook, Training, Tutoring, Universities, University Students, Vocational Education
18	Food and Beverage	Bakery, Brewing, Cannabis, Catering, Coffee, Confectionery, Cooking, Craft Beer, Dietary Supplements, Distillery, Farmers Market, Food Delivery, Food Processing, Food Trucks, Food and Beverage, Fruit, Grocery, Nutrition, Organic Food, Recipes, Restaurants, Seafood, Snack Food, Specialty Foods, Tea, Tobacco, Wine And Spirits, Winery
19	Gaming	Casual Games, Console Games, Contests, Fantasy Sports, Gambling, Game, Games, Gamification, Gaming, MMO Games, Online Games, PC Games, Serious Games, Video Games
22	Health Care	Alternative Medicine, Assisted Living, Assistive Technology, Biopharma, Cannabis, Child Care, Clinical Trials, Cosmetic Surgery, Dental, Diabetes, Diagnostics, Dietary Supplements, Doctors, Elder Care, Electronic Health Record (EHR), Electronic Health Records, Emergency Medicine, Employee Benefits, Fertility, First Aid, Funerals, Genetics, Health Care, Health Diagnostics, Healthcare Services, Home Health Care, Hospital, Medical, Medical Device, Nursing and Residential Care, Nutraceutical, Nutrition, Outpatient Care, Personal Health, Pharmaceutical, Physicians, Psychology, Rehabilitation, Senior Health, Therapeutics, Veterinary, Wellness, mHealth
24	Internet Services	Cloud Computing, Cloud Data Services, Cloud Infrastructure, Cloud Management, Cloud Storage, Curated Web, Cyber, Darknet, Domain Registrar, Domains, E-Commerce Platforms, Ediscovery, Email, ISP, Internet, Internet of Things, Location Based Services, Messaging, Music Streaming, Online Forums, Online Identity, Online Portals, Portals, Private Cloud, Product Search, SEM, SEO, SMS, Search, Search Engine, Semantic Search, Semantic Web, Social Media, Social Media Management, Social Network, Tracking, Unified Communications, Vertical Search, Video Chat, Video Conferencing, Visual Search, VoIP, Web Browsers, Web Hosting, Web Presence Management, Web Tools
27	Media and Entertainment	Advice, Animation, Art, Audio, Audiobooks, Blogging Platforms, Broadcasting, Celebrity, Concerts, Content, Content Creators, Content Discovery, Content Syndication, Creative, Creative Agency, DRM, Digital Entertainment, Digital Media, EBooks, Edutainment, Entertainment, Event Management, Event Promotion, Events, Film, Film Distribution, Film Production, Guides, In-Flight Entertainment, Independent Music, Internet Radio, Journalism, Media, Media and Entertainment, Motion Capture, Music, Music Education, Music Label, Music Streaming, Music Venues, Musical Instruments, News, Nightclubs, Nightlife, Performing Arts, Photo Editing, Photo Sharing, Photography, Podcast, Printing, Publishing, Reservations, Social Media, Social News, TV, TV Production, Television, Theatre, Ticketing, Video, Video Editing, Video Streaming, Video on Demand, Virtual World, Writers
33	Other	Alumni, Association, B2B, B2C, Blockchain, Charity, Collaboration, Collaborative Consumption, Commercial, Consumer, Crowdsourcing, Customer Service, Desktop Apps, Emerging Markets, Enterprise, Ethereum, Franchise, Freemium, Generation Y, Generation Z, Homeless Shelter, Incentives, Infrastructure, Knowledge Management, LGBT Millennials, Mass Customization, Mobility, Monetization, Non Profit, Nonprofits, Peer to Peer, Peer-to-Peer, Professional Services, Project Management, Real Time, Retirement, Service Industry, Sharing Economy, Small and Medium Businesses, Social Bookmarking, Social Impact, Subscription Businesses, Subscription Service, Technical Support, Testing, Underserved Children, Universities
38	Real Estate	Architecture, Brokers, Building Maintenance, Building Material, Commercial Real Estate, Construction, Coworking, Facility Management, Fast-Moving Consumer Goods, Green Building, Home & Garden, Home Automation, Home Decor, Home Improvement, Home Owners, Home Renovation, Home Services, Home and Garden, Interior Design, Janitorial Service, Landscaping, Office Space, Property Development, Property Management, Real Estate, Real Estate Investment, Realtors, Rental Property, Residential, Self Storage, Self-Storage, Smart Building, Smart Cities, Smart Home, Storage, Timeshare, Utilities, Vacation Rental
41	Software	3D Technology, Android, App Discovery, Application Performance Management, Apps, Artificial Intelligence, Augmented Reality, Billing, Bitcoin, Browser Extensions, Business Productivity, CAD, CMS, CRM, Cloud Computing, Cloud Management, Computer Vision, Consumer Applications, Consumer Software, Contact Management, Cryptocurrency, Data Center Automation, Data Integration, Data Storage, Data Visualization, Database, Developer APIs, Developer Platform, Developer Tools, Document Management, Drone Management, E-Learning, EdTech, Electronic Design Automation (EDA), Embedded Software, Embedded Systems, Enterprise Applications, Enterprise Resource Planning (ERP), Enterprise Software, Facial Recognition, File Sharing, IaaS, Image Recognition, Linux, MOOC, Machine Learning, Marketing Automation, Meeting Software, Mobile Apps, Mobile Payments, Natural Language Processing, Open Source, Operating Systems, PaaS, Predictive Analytics, Presentation Software, Presentations, Private Cloud, Productivity Tools, QR Codes, Reading Apps, Retail Technology, Robotics, SNS, SaaS, Sales Automation, Scheduling, Sex Tech, Simulation, Social CRM, Software, Software Engineering, Speech Recognition, Task Management, Text Analytics, Transaction Processing, Video Conferencing, Virtual Assistant, Virtual Currency, Virtual Desktop, Virtual Goods, Virtual Reality, Virtual World, Virtualization, Web Apps, Web Browsers, Web Development, iOS, macOS
45	Travel and Tourism	Adventure Travel, Amusement Park and Arcade, Business Travel, Casino, Hospitality, Hotel, Museums and Historical Sites, Parks, Resorts, Timeshare, Tour Operator, Tourism, Travel, Travel Accommodations, Travel Agency, Vacation Rental

Figure 3.1: Mapping from Raw ‘category list‘ Entries to Cleaned ‘Industry Group‘. Shows how noisy, inconsistent labels were grouped into standardized categories.

umn often contained noisy or redundant entries.

Since these values were self-reported by startups or collected from varying sources, we found substantial duplication and semantic overlap across category and market labels. For example, we encountered distinct entries such as *Software*, *iOS App*, *Web Application*, and *Application*, all of which conceptually belong to the same industry group. This lack of normalization would severely dilute the predictive power of any model attempting to learn sector-specific patterns.

To address this, we created a new standardized column named *Industry_Group*. This field represents a generalized mapping of all category and market labels into higher-level industry groupings. The classification was done manually and iteratively, requiring careful judgment to collapse redundant terms into broader, meaningful segments.

Given the time-consuming nature of this preprocessing step, it was prioritized early in our pipeline. The cleaned and enriched dataset, with the new *Industry_Group* column, was saved as a separate file named *StartupInvestments_IndustryGroup.csv*. This file also includes corrections to improperly formatted date fields that were present in the original *StartupInvestments.csv* file.

In addition to the industry grouping, we added a new feature column, *diff_funding_weeks*,

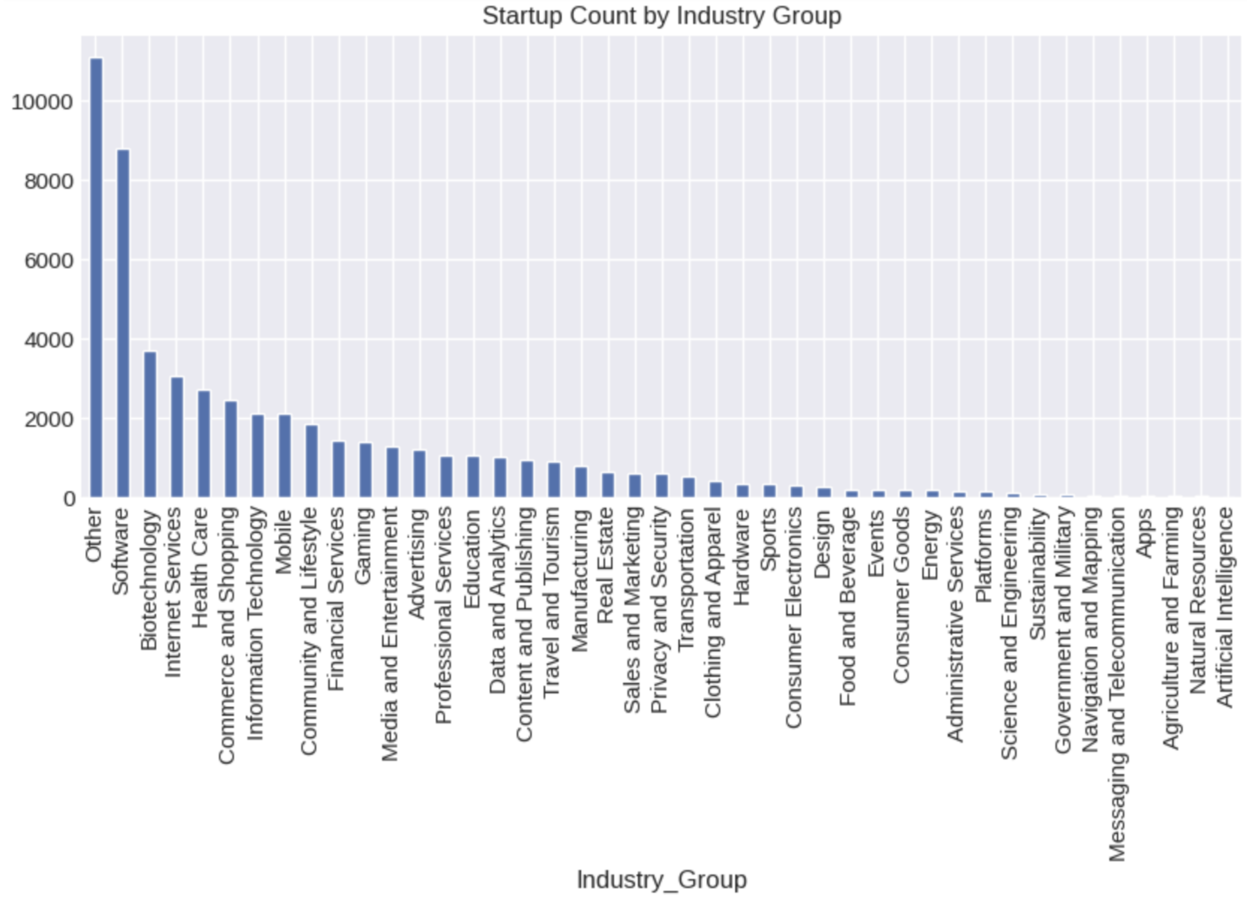


Figure 3.2: Distribution of startups across standardized industry groups. Most entries fall under broad or ambiguous categories, hence the high count in "Other".

which computes the number of weeks between a startup's first and last funding events. This temporal metric captures funding momentum and is included in the updated dataset for downstream modeling.

This transformation was crucial for reducing noise in the input features and enabling the model to better differentiate patterns across major sectors without being distracted by inconsistent category granularity.

3.2 Balancing the Dataset

As discussed earlier, the dataset exhibits a significant class imbalance. The majority of startups in the original data were labeled as *operating* (approximately 87%), while only about 8% had been *acquired*. This imbalance poses a major challenge for supervised learning, as models tend to overfit to the dominant class, reducing their ability to correctly identify rare but meaningful events.

From a venture capital perspective, the ideal investment exit strategies are either through a successful acquisition or an IPO. However, the *operating* label alone does not provide a clear signal of success, as many such startups may stagnate or operate at a loss without formally closing or exiting. To more accurately reflect true positive outcomes from the perspective of VC returns, we refined our success criteria.

To identify IPOs, we examined the *post_ipo_equity* and *post_ipo_debt* columns. We assumed that any startup with either of these values greater than zero had likely filed for an IPO. Based on this logic, we created a new binary column, *filed_ipo*, to flag such companies.

Using this refined definition of success, we filtered the dataset to retain only startups that were either *acquired*, had *filed_ipo*, or had *closed*. This transformation allows us to frame the prediction problem more meaningfully—classifying startups as either successful exits (via acquisition or IPO) or failures (closed).

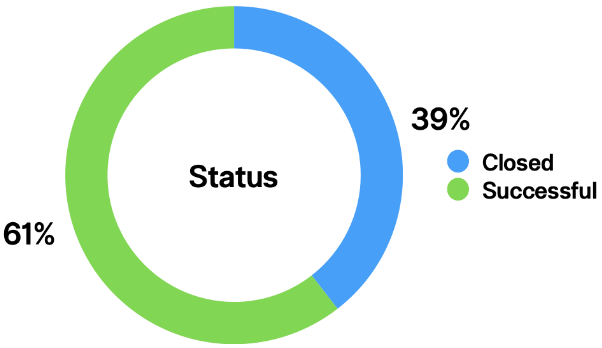


Figure 3.3: Startup Status Distribution after Feature Engineering

As a tradeoff, this filtering significantly reduced the dataset size—from 54,294 records to just 6,586 records. However, it greatly improved the quality and relevance of the data for modeling investment outcomes. The resulting class distribution was also more balanced, with approximately 60.5% successful outcomes and 39.5% closures, which provides a more stable foundation for training classification models. (Figure 3.3)

3.3 Handling Missing Values

Even after balancing the dataset and reducing it to include only relevant startup outcomes, we observed that several important features continued to contain missing values. Left unaddressed, these null entries would pose significant challenges during model training and reduce the robustness of our predictions.

The most prominent features with missing data were Founded Year, Country Code, and Funding Weeks Difference. Each of these variables is crucial for capturing a startup's timeline, geographical context, and funding dynamics—making their imputation a necessary pre-processing step.

After analyzing the distribution of the available data, we adopted the following strategies:

- **Founded Year:** This variable indicates the year in which a startup was established and plays an important role in understanding market timing and funding behavior. We filled missing values in *founded_year* with the *median* year across the dataset to avoid skewing the temporal distribution.

- **Country Code:** Country information is critical for identifying regional funding trends and exit patterns. Since the majority of startups in the dataset are based in the United States, we replaced missing *country_code* entries with the most frequent value, *USA*.
- **Funding Weeks Difference:** This feature represents the number of weeks between a startup's first and last funding rounds. It is an important proxy for funding momentum. Only a single instance had a missing value, which we conservatively filled with *0*, assuming that the startup had a single funding event.

These imputation strategies allowed us to retain all rows in the dataset without introducing significant distortions, preserving both data volume and model fidelity.

3.4 Feature Engineering

To enhance the predictive power of the dataset, we engineered several new features derived from existing fields. These transformations allowed the model to capture underlying patterns that are not directly observable in the raw data.

In addition to the previously discussed Industry Group, Funding Weeks Difference, and Filed IPO columns, we created the following features:

- **Years Since Founding:** This feature captures the age of the startup as of 2014, which is the latest year present in our dataset. It is computed as the difference between 2014 and the *founded_year*. This proxy for operational maturity can be a strong indicator of a startup's lifecycle stage.

- **Has Raised Funding (Binary):** A binary variable indicating whether a startup has raised any funding. It is set to 1 if the *funding_total_usd* is greater than 0, and 0 otherwise.

- **Has Seed / Venture / Angel Funding (Binary):** We created three separate binary variables to indicate the presence of specific funding types. These are:

  - *has_seed_funding*
  - *has_venture_funding*
  - *has_angel_funding*

  Each variable is assigned a value of 1 if the respective funding type amount is greater than 0.

- **Funding Bin (Categorical):** Since funding amounts are highly skewed with many outliers, we binned *funding_total_usd* into categorical ranges using the following bins:

  - **Very Low:** $0 to $1,000,000
  - **Low:** $1,000,001 to $10,000,000
  - **Medium:** $10,000,001 to $50,000,000
  - **High:** $50,000,001 to $100,000,000
  - **Very High:** Above $100,000,000

  This transformation helps the model handle extreme funding values more robustly.

- **Categorical Encoding:** To prepare the dataset for machine learning algorithms, we encoded the categorical variables *country_code*, *industry_group*, and *funding_bin* using the `LabelEncoder` class from the `sklearn.preprocessing` module. This ensured that all features were in a numerical format suitable for model consumption.

These engineered features provide a more structured and informative representation of startup characteristics, enabling the model to better distinguish between successful and unsuccessful companies.

4 | Exploratory Data Analysis

With the cleaned and engineered dataset in place, we conducted exploratory data analysis to uncover meaningful patterns, trends, and anomalies within the data. The primary objectives of this phase were to better understand the distribution of key features, identify relationships between predictors and the target variable, and validate the effectiveness of our feature engineering efforts.

Exploratory data analysis also helped highlight important domain insights—such as funding behavior across industries, differences in startup trajectories by geography, and the prevalence of success among different funding stages. These observations informed our model selection and evaluation strategies in the subsequent phases.

4.1 Startup Age and Success Rate

One of the most insightful patterns we observed during the analysis was the relationship between a startup's founding year and its likelihood of success. To investigate this, we plotted the success rate as a function of the startup's founding year.

The results revealed a clear trend: older startups tend to have higher success rates compared to more recently founded companies. For instance, startups founded around 1990 exhibited success rates close to 80%, whereas those founded in 2014 showed success rates closer to 40%. (Figure **??**)

This pattern aligns with expectations—startups founded earlier have had more time to mature,

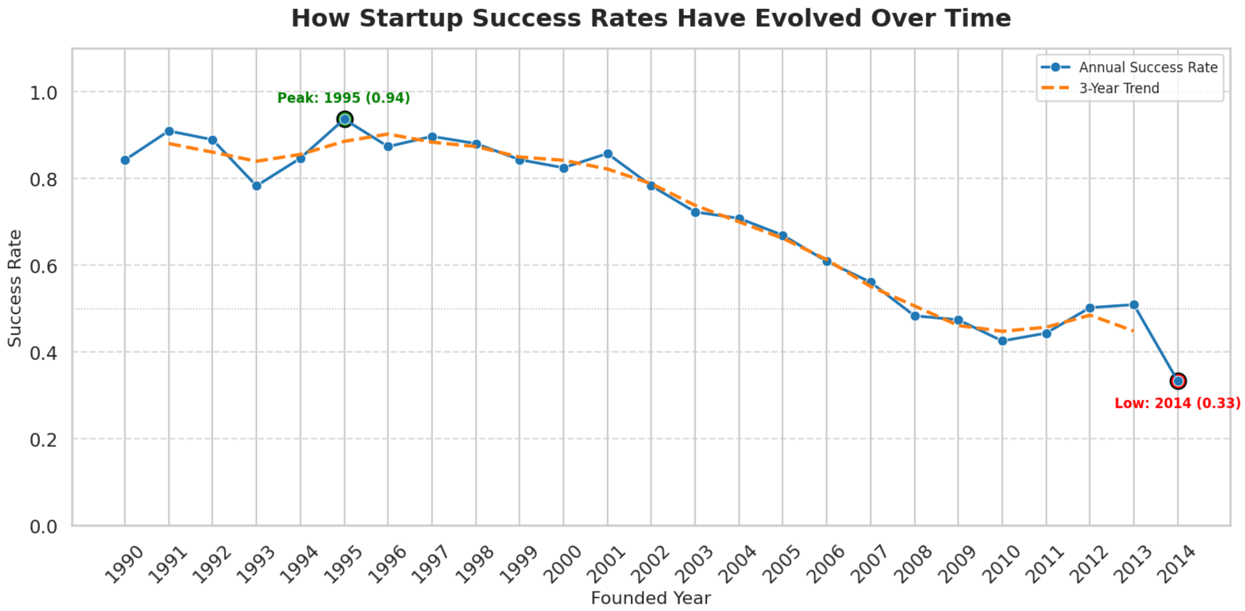


Figure 4.1: Annual startup success rates by founding year (1990–2014), with a peak observed in 1995 (94%) and a decline to a low in 2014 (33%). The trend suggests that older startups are more likely to succeed, potentially due to maturity and survivorship over time.

raise capital, and reach an exit event (e.g., acquisition or IPO). Meanwhile, startups founded in more recent years may still be in early growth stages or have not yet reached maturity to realize a definitive outcome.

This temporal insight emphasizes the importance of considering startup age and lifecycle timing when modeling success, as it captures a strong underlying signal of maturity and market evolution.

4.2 Location and Success Rate

Another important factor influencing startup success is geographic location. To explore this, we analyzed the relationship between a startup's country and its success rate. The plot of success rates by country revealed a strong geographical pattern.

We found that the top 10 countries with the highest number of successful startups are: Canada, USA, Germany, Israel, France, Ireland, Great Britain, China, Spain, and India (Figure 4.2). These

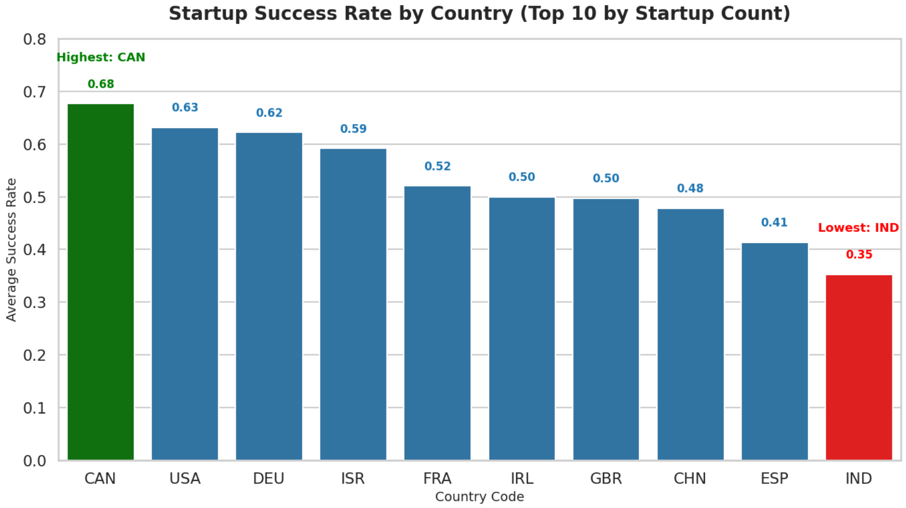


Figure 4.2: Startup success rates by country for the top 10 countries by startup count. Canada leads with an average success rate of 68%, followed by the USA and Germany. India has the lowest average success rate among the top 10 at 35%, highlighting the impact of regional ecosystem differences on startup outcomes.

countries consistently show higher rates of startup success, indicating that location plays a critical role in entrepreneurial outcomes.

This pattern reflects broader socioeconomic factors. Locations with strong investment in education, technical skill development, and access to early-stage capital tend to foster more successful startups. Additionally, stable political environments and robust legal frameworks further enable business growth and investor confidence.

These findings underscore that startup success is not purely a function of innovation or product-market fit—it is also heavily influenced by the support ecosystems and institutional conditions in which the startups operate.

4.3 Industry and Success Rate

To examine how success varies across sectors, we analyzed the relationship between the Industry Group feature and startup success rate. The results revealed clear patterns that align well with real-world trends in venture outcomes.

Startups in the Healthcare, Software-related industries, and Biotechnology sectors demonstrated the highest success rates in our dataset. These were followed by startups in Commerce and Shopping, Gaming, and Community and Lifestyle categories. (Figure 4.3)

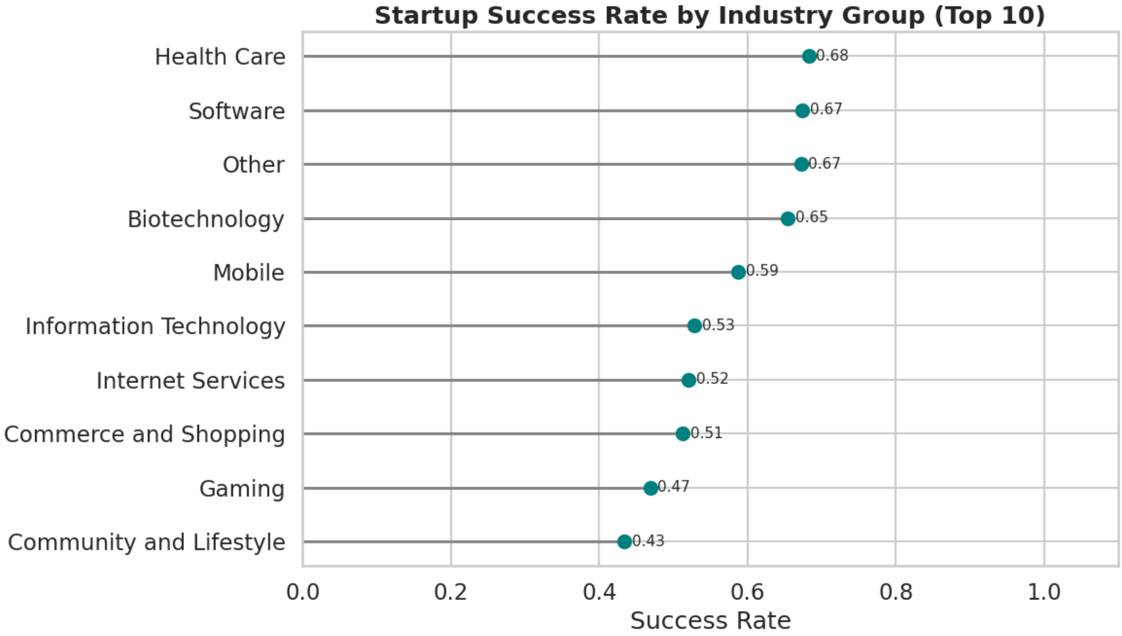


Figure 4.3: Startup success rates by industry group for the top 10 sectors by representation. Health Care, Software, and Biotechnology lead with success rates above 65%, while sectors like Gaming and Community and Lifestyle show lower performance. This highlights the importance of industry context in predicting startup outcomes.

This finding reflects broader market realities—many of the most high-profile and well-funded startups over the past two decades have emerged from these exact sectors. Industries such as healthcare and biotechnology benefit from long-term demand, innovation-driven R&D, and high-value exit potential. Meanwhile, software and commerce startups are often able to scale rapidly, supported by lower capital expenditure and global reach.

These insights reinforce the notion that industry sector plays a pivotal role in a startup's trajectory and should be treated as a key input feature in any predictive modeling pipeline.

4.4 Feature Correlation

To better understand the relationships among features and their association with startup success, we generated a correlation heatmap comparing all numerical variables in the dataset. This analysis helps identify linear dependencies and potential predictive signals.

As expected, it is challenging to find features that strongly correlate with the binary success outcome (Figure 4.4). However, we did observe some modest positive correlations between success and a few meaningful features—specifically: years since founding, number of funding rounds, funding velocity (using *diff_funding_weeks)*, and the amount raised in venture funding rounds.

Beyond the target variable, we also identified strong correlations between specific funding features. Notably, there is a high positive correlation between *venture* funding and funding raised in rounds *A* through *H.* This suggests that once a startup secures venture capital, it is significantly more likely to continue raising funds through subsequent series rounds—a well-known pattern in startup growth trajectories.

Furthermore, features such as funding velocity and number of funding rounds are positively correlated with higher funding totals across advanced rounds. These relationships highlight the compounding nature of venture investment—early momentum and access to capital greatly increase a startup's likelihood of scaling.

While correlation does not imply causation, these insights reinforce the importance of early funding access and operational maturity in predicting future success.

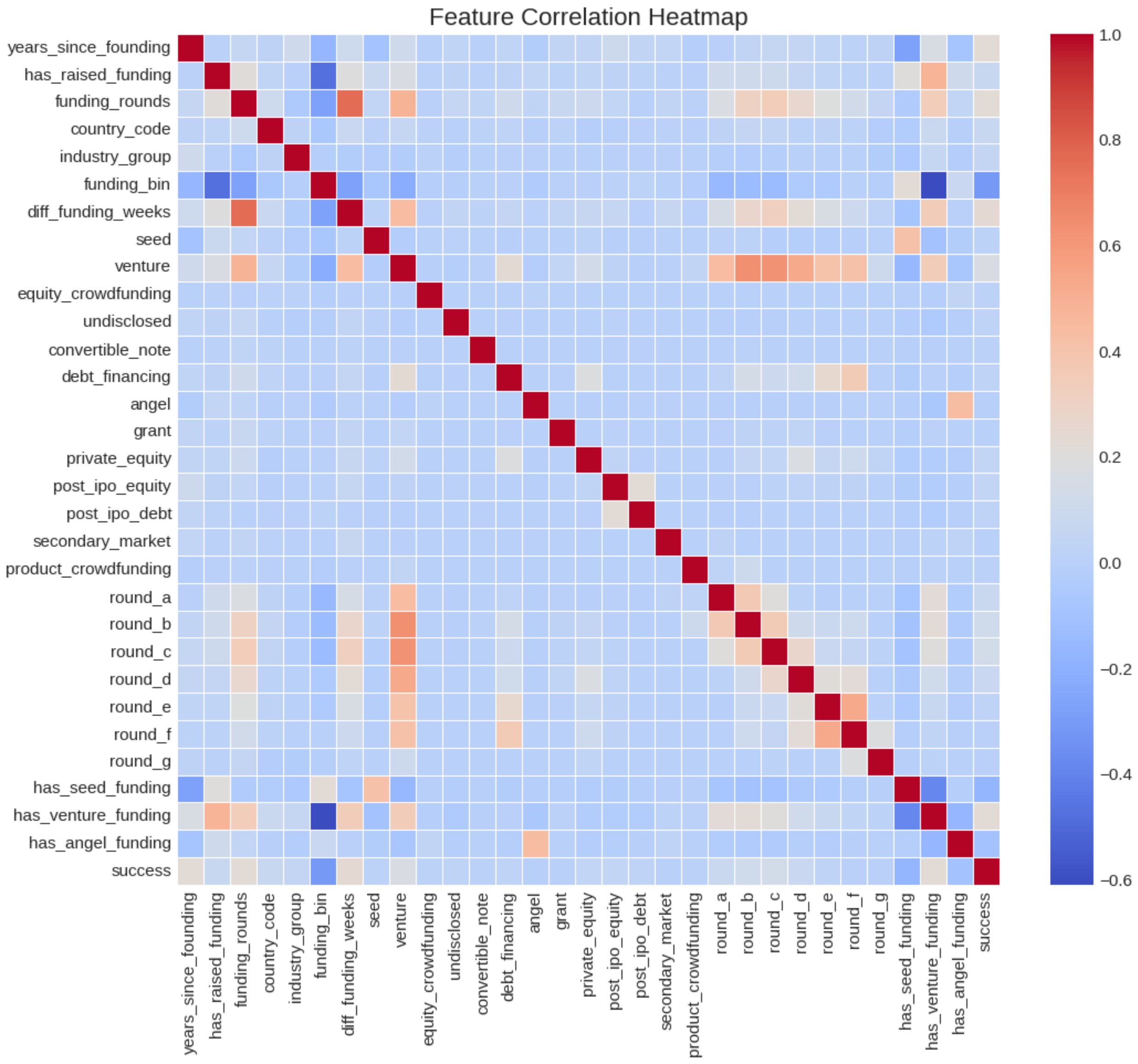


Figure 4.4: Feature Correlation with Startup Success

5 | Modeling

After extensive data cleaning, feature engineering, and exploratory analysis, we were ready to apply machine learning models to predict startup success. Our objective was to classify startups as either successful (acquired or filed IPO) or unsuccessful (closed), based on the engineered features.

To begin, we split the dataset into training and testing sets using an 80-20 split. This allowed us to train models on the majority of the data while reserving a portion for out-of-sample evaluation to assess generalization performance.

We adopted an iterative modeling approach—starting with a simple baseline and progressively moving to more complex and expressive models. This stepwise process allowed us to evaluate the value added by increased model complexity and helped in selecting the most suitable algorithm for our prediction task.

- **Logistic Regression:** We began with Logistic Regression as a simple linear baseline model. It is interpretable and serves as a useful benchmark.
- **Decision Tree Classifier:** We then introduced a non-linear model that can capture interactions between features without requiring feature scaling.
- **Random Forest:** A more robust ensemble method that reduces overfitting by averaging across multiple decision trees.
- **XGBoost and LightGBM:** These gradient boosting frameworks were used for their efficiency and strong performance on structured data.

- **CatBoost:** This model handles categorical variables natively and performs well without extensive preprocessing.
- **Stacking Ensemble:** Finally, we experimented with a meta-model that combines the predictions of multiple base learners to improve generalization.

Each model was evaluated on the same training and test splits using consistent performance metrics, which we discuss in the following section. Our goal was to identify the model that best fits our use case.

5.1 Logistic Regression

We began our modeling process with Logistic Regression, which serves as a strong and interpretable baseline for binary classification problems. Logistic Regression is particularly effective for estimating the probability of class membership and provides a good benchmark to compare more complex models against.

Prior to training, we applied feature scaling using a standard scaler to normalize the input features. This ensures that the model treats all features equally and avoids bias toward variables with larger magnitudes.

The model was trained using the following configuration:

```
LogisticRegression(max_iter=5000, class_weight='balanced', solver='lbfgs')
```

The `class_weight='balanced'` parameter was especially important given our dataset's class imbalance, as it penalizes misclassification of minority classes more heavily.

Evaluation

The model achieved an overall accuracy of 69%. For the negative class (0 — unsuccessful startups), the precision was 0.59 and recall was 0.69, meaning the model captures a reasonable

Table 5.1: Logistic Regression Classification Report

Class	Precision	Recall	F1-Score
0 (Unsuccessful)	0.59	0.69	0.64
1 (Successful)	0.77	0.69	0.73
Accuracy			**0.69**

portion of actual failures but includes some false positives. For the positive class (1 — successful startups), precision was higher at 0.77, indicating fewer false positives, though recall was slightly lower at 0.69. (Table 5.1)

The F1-score, which balances precision and recall, was 0.64 for class 0 and 0.73 for class 1, suggesting the model performs better at identifying successful startups than failures. This aligns with real-world expectations where success patterns may be more distinguishable than failure patterns in limited data.

Finally, we evaluated the model using the AUC metric and achieved an AUC score of **0.7596**, indicating that the model has reasonable discriminative power and performs better than random guessing.

This model serves as our baseline to assess the value added by more complex classifiers.

5.2 Decision Tree Classifier

After establishing a baseline using Logistic Regression, we implemented a Decision Tree Classifier to explore a more flexible, non-linear model that can automatically capture feature interactions and decision boundaries.

The model was trained using the following parameters:

```
DecisionTreeClassifier(
    criterion='gini', max_depth=None,
    min_samples_split=10, min_samples_leaf=5,
    class_weight='balanced')
```

The parameters were chosen to balance model flexibility and generalization. We used `'gini'` to evaluate split quality using Gini impurity, although entropy could also be used for information gain. The `max_depth` was left as `None`, allowing the tree to grow until all leaves are pure or until other stopping conditions are met. To control overfitting, we set `min_samples_split=10`, requiring at least 10 samples to split a node, and `min_samples_leaf=5`, ensuring each leaf contains a minimum of 5 samples. Finally, we used `class_weight='balanced'` to automatically adjust for the class imbalance by weighting classes inversely proportional to their frequencies.

Evaluation

Table 5.2: Decision Tree Classification Report

Class	Precision	Recall	F1-Score
0 (Unsuccessful)	0.55	0.68	0.61
1 (Successful)	0.75	0.64	0.69
Accuracy			**0.65**

The Decision Tree model achieved an accuracy of 65% and an AUC score of **0.7207**, which is lower than the performance of our Logistic Regression baseline (AUC = 0.7596) (Table 5.2). While the model showed stronger recall for the negative class, its overall performance was weaker—likely due to overfitting or insufficient generalization from deeper branches in the tree.

Despite its flexibility, the Decision Tree did not offer an improvement over the simpler logistic model.

5.3 Random Forest Classifier

To improve model performance over the Decision Tree classifier, we implemented a Random Forest Classifier. Random Forest is an ensemble learning method that constructs multiple decision trees and averages their predictions to produce a more robust and generalizable model.

We trained the Random Forest using the following configuration:

```
RandomForestClassifier(
    n_estimators=200, max_depth=None,
    min_samples_split=10, min_samples_leaf=5)
```

We set n_estimators=200 to build a sufficiently large number of trees, allowing the ensemble to better capture complex patterns and reduce variance. The max_depth=None setting allows each tree to grow fully, although this can be tuned further if needed. To control overfitting, we used min_samples_split=10 and min_samples_leaf=5, which prevent overly fine splits and ensure that each decision point in the tree is based on a reasonable sample size.

Evaluation

Table 5.3: Random Forest Classification Report

Class	Precision	Recall	F1-Score
0 (Unsuccessful)	0.72	0.59	0.65
1 (Successful)	0.76	0.85	0.80
Accuracy			**0.75**

The Random Forest model significantly outperformed both Logistic Regression and the single Decision Tree classifier. It achieved an overall accuracy of 75% and a high AUC score of **0.8011**, indicating strong discriminative capability between successful and unsuccessful startups.

The classifier showed particularly strong performance in identifying successful startups (class 1), with a recall of 0.85 and an F1-score of 0.80 (Table 5.3). This means the model was able to correctly capture a large proportion of truly successful companies while maintaining a good balance between precision and recall.

Although the recall for the unsuccessful class (class 0) was lower at 0.59, the increase in overall accuracy and AUC suggests that the Random Forest was more effective at capturing the patterns associated with positive startup outcomes. This performance gain, along with its robustness and stability, makes Random Forest a strong candidate for real-world deployment.

5.4 XGBoost Classifier

To leverage gradient boosting techniques, we implemented an XGBoost classifier. XGBoost is a powerful and efficient algorithm that uses gradient-boosted decision trees and has become a leading choice for structured data problems. It is particularly effective in handling class imbalance and overfitting when properly tuned.

We trained the XGBoost model with the following configuration:

```
XGBClassifier(
    objective='binary:logistic', eval_metric='auc',
    scale_pos_weight=(negatives / positives), n_estimators=100,
    learning_rate=0.1, max_depth=5)
```

We set `objective=’binary:logistic’` to perform binary classification, and `eval_metric` to directly optimize for the area under the ROC curve. The `scale_pos_weight` parameter was computed as the ratio of negative to positive samples in the training data, helping XGBoost address the class imbalance by penalizing errors on the minority class more heavily.

The model uses 100 boosting rounds (`n_estimators=100`) with a `learning_rate=0.1`, which controls how much each tree contributes to the final prediction. We also set `max_depth=5` to limit tree complexity and reduce overfitting.

Evaluation

Table 5.4: XGBoost Classification Report

Class	Precision	Recall	F1-Score
0 (Unsuccessful)	0.63	0.70	0.66
1 (Successful)	0.79	0.73	0.76
Accuracy			**0.72**

The XGBoost model achieved an overall accuracy of 72% and a strong AUC score of **0.8014**, indicating excellent discriminative capability between successful and unsuccessful startups. This AUC is slightly higher than that of the Random Forest model, suggesting XGBoost performs marginally better in ranking predictions.

The model shows a solid balance across both classes. It achieved a recall of 0.70 for the unsuccessful class and 0.73 for the successful class, indicating a well-rounded performance that avoids heavily favoring one class over the other. The F1-score of 0.76 for the successful class further supports the model's ability to identify high-potential startups with consistency.

With its competitive accuracy, strong AUC, and efficient handling of class imbalance, XGBoost stands out as one of the top-performing models in our evaluation. Its support for fine-tuned control and deployment readiness makes it highly suitable for real-world predictive decision-making in the venture investment space.

5.5 LightGBM Classifier

To further explore gradient boosting approaches, we implemented a LightGBM (Light Gradient Boosting Machine) classifier. LightGBM is known for its efficiency, scalability, and speed, particularly on large tabular datasets. It uses histogram-based algorithms to speed up training without significantly compromising performance.

The model was configured as follows:

```
LGBMClassifier(
    objective='binary', boosting_type='gbdt',
    is_unbalance=True, n_estimators=100,
    learning_rate=0.1, max_depth=5)
```

We set the `objective='binary'` for binary classification, and `boosting_type='gbdt'` to use traditional gradient-boosted decision trees. The parameter `is_unbalance=True` automatically adjusts class weights internally to handle the class imbalance, as an alternative to manually setting scale_pos_weight. We used 100 trees (n_estimators=100) with a learning_rate=0.1, balancing model learning speed and generalization. The max_depth=5 limits tree complexity, helping prevent overfitting.

Evaluation

Table 5.5: LightGBM Classification Report

Class	Precision	Recall	F1-Score
0 (Unsuccessful)	0.63	0.70	0.67
1 (Successful)	0.79	0.73	0.76
Accuracy			**0.72**

The LightGBM model achieved an accuracy of 72% and an ROC AUC score of **0.8001**, putting it in the same high-performing category as XGBoost and Random Forest. The classifier was particularly effective in predicting successful startups (class 1), with an F1-score of 0.76 and precision of 0.79.

The recall for the unsuccessful class (class 0) reached 0.70, indicating that the model is able to correctly identify many of the startups that eventually fail, while still maintaining high precision on the successful class.

Overall, LightGBM delivered competitive performance while training quickly and efficiently. Its performance confirms that gradient boosting models are particularly well-suited for this type of structured, imbalanced classification problem. LightGBM is a strong candidate for production deployment due to its speed, predictive power, and ease of tuning.

5.6 CatBoost Classifier

CatBoost is a gradient boosting framework developed by Yandex that is particularly well-suited for datasets with categorical variables. It handles categorical features internally without requiring manual encoding, which simplifies the modeling pipeline and often improves performance. It also mitigates overfitting through its use of ordered boosting and oblivious decision trees.

We trained the CatBoost model with the following configuration:

```
CatBoostClassifier(
    iterations=100, learning_rate=0.1,
    depth=6, eval_metric='AUC',
    class_weights=class_weights)
```

`iterations=100` controls the number of boosting rounds, while `learning_rate=0.1` dictates the contribution of each tree to the final model—balancing learning speed and generalization. The `depth=6` defines the maximum depth of each tree, offering enough complexity to learn non-linear interactions. The model is optimized using the `AUC` metric, aligning with our objective of distinguishing between successful and unsuccessful startups. To handle class imbalance, we used the `class_weights` parameter to assign higher weight to the minority class (unsuccessful startups).

Evaluation

Table 5.6: CatBoost Classification Report

Class	**Precision**	**Recall**	**F1-Score**
0 (Unsuccessful)	0.63	0.71	0.67
1 (Successful)	0.79	0.73	0.76
Accuracy			**0.72**

The CatBoost classifier achieved an accuracy of 72% and the highest ROC AUC score among all models evaluated: **0.8019**. This indicates excellent capability in distinguishing between successful and unsuccessful startups.

The model achieved a strong F1-score of 0.76 for the successful class, and a recall of 0.71 for the unsuccessful class, suggesting that it maintains a good balance in identifying both outcomes accurately. These results are consistent with those from other boosting models like XGBoost and LightGBM but show a slight performance edge in terms of AUC.

CatBoost's internal handling of categorical features and its robust regularization mechanisms make it a highly effective choice for structured datasets like this one. Its combination of performance, simplicity in feature handling, and resistance to overfitting make it an excellent candidate for deployment.

5.7 Fine-Tuning Random Forest

To attempt further improvements on the base Random Forest model, we applied hyperparameter optimization using `RandomizedSearchCV`. This method allows for efficient exploration of the parameter space by randomly sampling combinations rather than performing an exhaustive grid search, making it suitable for complex models with many tunable parameters.

We ran `RandomizedSearchCV` on the Random Forest Classifier and identified the following best parameter combination:

```
{'n_estimators': 100, 'min_samples_split': 2,
'min_samples_leaf': 1, 'max_features': 'sqrt',
'max_depth': 20}
```

The tuned model uses `n_estimators=100`, which specifies the number of trees in the ensemble—a slightly lower count than our earlier configuration. Setting `min_samples_split=2` and `min_samples_leaf=1` allows the tree to split as deeply as possible, increasing model flexibility but also the risk of overfitting. The `max_features='sqrt'` parameter ensures that only a random subset of features is considered at each split, which promotes diversity among trees and reduces overfitting. Finally, `max_depth=20` limits the depth of the trees, helping to constrain complexity while allowing sufficient expressiveness to learn meaningful patterns.

Evaluation

Table 5.7: Fine-Tuned Random Forest Classification Report

Class	Precision	Recall	F1-Score
0 (Unsuccessful)	0.69	0.58	0.63
1 (Successful)	0.75	0.83	0.79
Accuracy			**0.73**

Despite the use of hyperparameter tuning, the fine-tuned Random Forest did not outperform the original untuned model. While the recall for the successful class improved (0.83 vs. 0.85 previously), performance on the unsuccessful class declined, with lower recall (0.58 vs. 0.59) and F1-score (0.63 vs. 0.65).

Additionally, the overall accuracy (0.73) and ROC AUC score (0.7889) were slightly lower than the original Random Forest, which achieved an AUC of 0.8011. This suggests that the untuned model was already well-balanced for our data, and the tuning may have led to overfitting or less generalizable decision boundaries.

These results highlight an important lesson: hyperparameter tuning does not always guarantee performance improvement. In this case, the simpler and more regularized base Random Forest yielded more reliable and robust results.

5.8 Stacked Ensemble Model

To leverage the strengths of multiple classifiers and improve overall predictive performance, we implemented a stacked ensemble model. Stacking combines several base learners and uses a meta-learner to aggregate their predictions. This approach is often more powerful than any individual model, as it allows the ensemble to learn how to correct the biases and errors of its components.

The stacking model was composed of the following base learners - Logistic Regression, Decision Tree, Random Forest, LightGBM, XGBoost, and CatBoost. The **meta-learner** was a Logistic Regression model, trained on the predictions of the base learners. This second-layer model learns how to optimally combine the base model outputs to improve final classification accuracy. The stacking model was implemented using `StackingClassifier` with `passthrough=False` to prevent raw feature leakage into the meta-model.

Evaluation

Table 5.8: Stacked Ensemble Classification Report

Class	**Precision**	**Recall**	**F1-Score**
0 (Unsuccessful)	0.64	0.70	0.67
1 (Successful)	0.79	0.74	0.77
Accuracy			**0.73**

The stacked ensemble model achieved an overall accuracy of 73% and an ROC AUC score of **0.8021**, making it the best-performing model in our study. It demonstrates robust and balanced performance across both classes. For the unsuccessful class, it attained a recall of 0.70, while for the successful class it maintained a strong F1-score of 0.77.

The ensemble benefits from the complementary strengths of its constituent models—capturing both linear and non-linear decision boundaries, handling imbalanced data effectively, and maintaining high generalization ability. These results validate the effectiveness of model stacking for structured, imbalanced classification tasks like predicting startup success.

With its superior AUC and balanced class-wise performance, the stacked model represents the most promising candidate for real-world deployment scenarios where both precision and recall are critical.

5.9 Model Comparison

To summarize the performance of all evaluated models, we present a comparison table highlighting the key classification metrics: accuracy, precision, recall, F1-score, and ROC AUC score. The models are ranked in descending order of AUC, as it is the most relevant metric for our imbalanced classification task.

Table 5.9: Performance Comparison of All Models (Sorted by AUC)

Model	Accuracy	Precision	Recall	F1-Score	AUC
Stacking Ensemble (All Models)	0.726	0.792	0.742	0.766	**0.802**
CatBoost	0.720	0.792	0.729	0.759	**0.802**
XGBoost	0.718	0.787	0.731	0.758	0.801
Random Forest	**0.747**	0.759	**0.851**	**0.802**	0.801
LightGBM	0.720	0.791	0.730	0.759	0.800
Fine-Tuned Random Forest	0.732	0.753	0.829	0.789	0.789
Logistic Regression	0.687	0.772	0.686	0.726	0.760
Decision Tree Classifier	0.653	0.751	0.637	0.690	0.721

From the table, we observe that the Stacking Ensemble model achieves the highest overall AUC and offers the best balance across all evaluation metrics. Interestingly, while the Random Forest model achieved the highest recall and accuracy, its AUC was slightly lower, suggesting it may be more prone to overfitting.

Gradient boosting models such as CatBoost, XGBoost, and LightGBM consistently deliver strong results, validating their effectiveness for structured and imbalanced datasets. The fine-tuned Random Forest showed improvements in some areas but did not surpass the untuned version in AUC.

Overall, ensemble approaches—both bagging and boosting—proved to be the most effective strategies for predicting startup success.

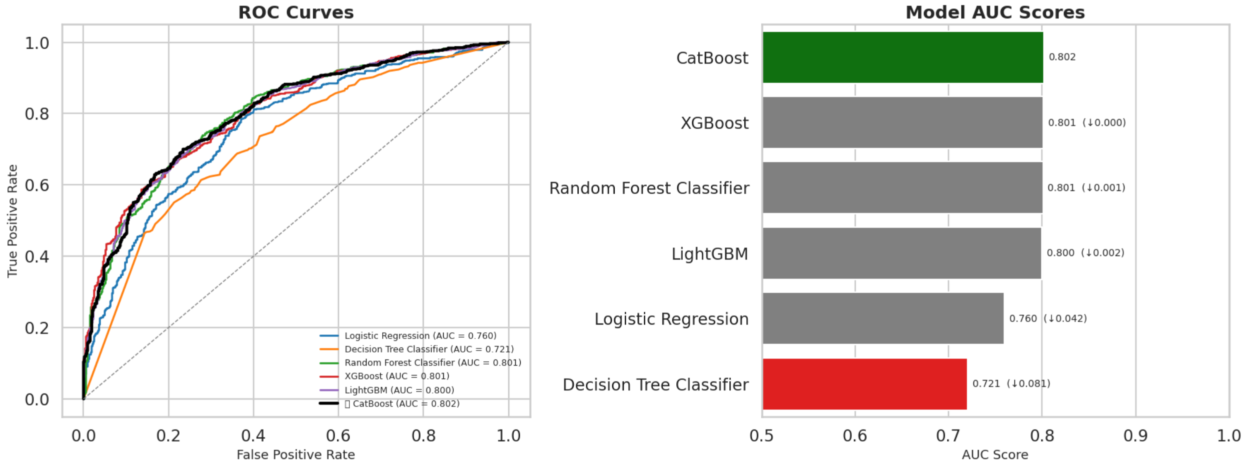


Figure 5.1: ROC curves (left) and AUC score comparison (right) for all models evaluated. CatBoost achieved the highest AUC score at 0.802, followed closely by XGBoost and Random Forest (both 0.801). Logistic Regression and Decision Tree underperformed relative to ensemble methods, with AUC scores of 0.760 and 0.721, respectively. These results highlight the advantage of gradient boosting and ensemble techniques for modeling startup success.

6 Evaluation

After training and evaluating a diverse set of models—including linear classifiers, decision trees, ensemble methods, and gradient-boosted learners—we now assess their suitability in the context of our business objective.

In our use case, the cost of missing a high-potential startup (false negative) is significantly higher than mistakenly investing in one that fails (false positive). For venture capital firms, identifying startups that are likely to succeed is paramount—even at the expense of occasionally selecting some that do not. This makes **recall** for the positive class (i.e., identifying successful startups) the most critical evaluation metric.

While models like the Stacked Ensemble and CatBoost delivered strong overall AUC scores, we prioritize models that excel in recall. Among all evaluated models, the **Random Forest Classifier** achieved the highest recall of **0.85** for the successful startup class, while also maintaining a solid F1-score and accuracy. This demonstrates the model's ability to capture a wide range of potential successes with a reasonable trade-off in precision.

Based on this analysis, we selected the **Random Forest** as our final model. It provides the best balance of performance and interpretability, and most importantly, aligns with the business objective of maximizing successful investment opportunities.

6.1 Feature Importance

Throughout our modeling process, we consistently observed that a few features emerged as the most influential predictors of startup success. Across tree-based models such as Random Forest, XGBoost, LightGBM, and CatBoost, the following features were repeatedly ranked among the top in importance - Years Since Founding, Industry Group, Funding Velocity (measured using *diff_funding_weeks*), and Funding Bin (categorical representation of total funding).

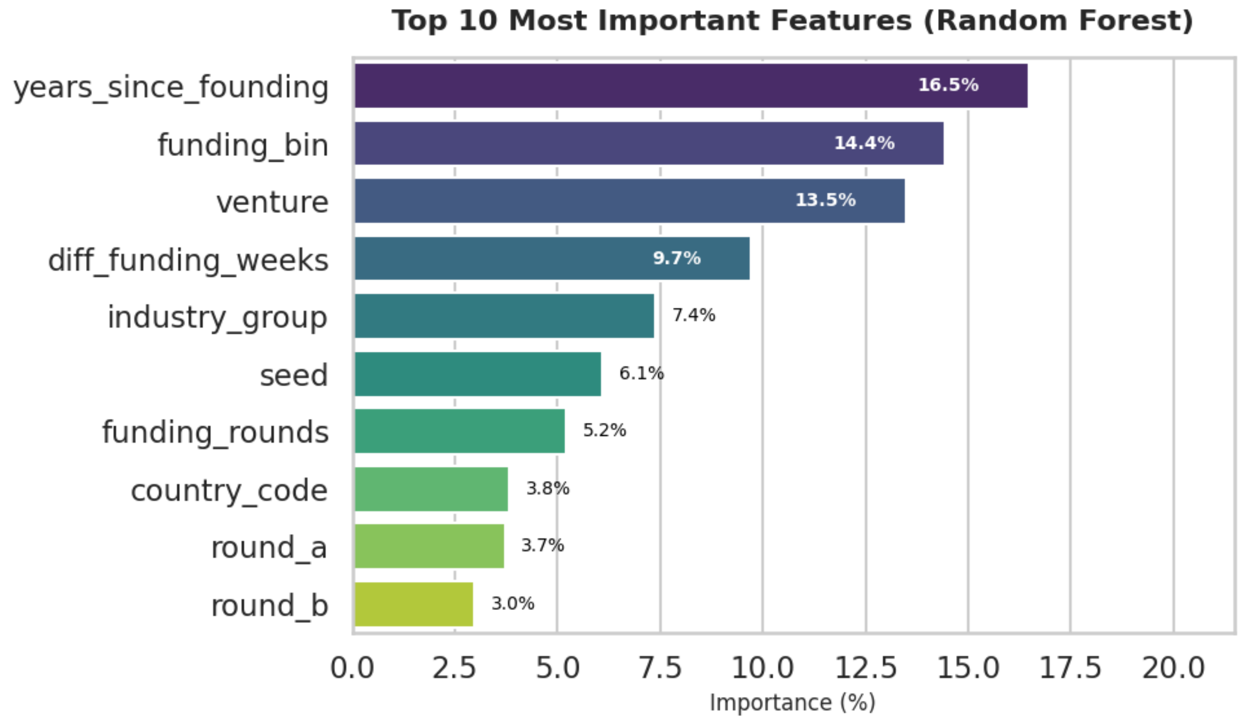


Figure 6.1: Top 10 most important features based on Random Forest feature importance. `years_since_founding`, `funding_bin`, and venture funding emerge as the strongest predictors of startup success, followed by funding velocity and industry group. These features reflect both temporal maturity and capital traction as key drivers in predicting outcomes.

These findings are both data-driven and intuitively valid.

Years Since Founding reflects the maturity of the startup. Older startups have had more time to grow, stabilize, and potentially reach an acquisition or IPO. As observed in our EDA, success rates were higher for startups founded in earlier years.

Industry Group captures the sector in which a startup operates. Certain industries—such as Healthcare, Biotechnology, and Software—historically yield higher success rates due to strong investor interest, market demand, and exit opportunities. Grouping noisy raw market data into structured industry categories proved highly effective.

Funding Velocity, calculated as the number of weeks between the first and last funding rounds, serves as a proxy for investment momentum. Startups that raise capital quickly tend to have stronger market signals and investor confidence, which correlates with future success.

Funding Bin captures the scale of total funding raised. By grouping funding amounts into discrete levels, we reduced the impact of outliers and enabled models to learn meaningful thresholds of capital that separate high-growth startups from underfunded ones.

Together, these features provide a strong representation of startup growth trajectory, investor behavior, and market positioning—key dimensions in predicting long-term success. Their prominence across multiple models reinforces their predictive power and practical relevance for venture decision-making.

6.2 Profit Analysis

To evaluate the real-world applicability of our models, we conducted a profit simulation under a hypothetical venture capital investment framework. Our goal was to understand how different prediction thresholds and market conditions affect investment profitability, and to illustrate how a model like Random Forest could support data-driven funding decisions.

We simulated four investment scenarios that vary in risk tolerance and return potential:

1. Bad Market: VC invests $3M in each startup and receives $2M upon success (net loss per investment).

2. Conservative: VC invests $2M and earns $5M per successful startup.

3. Moderate: VC invests $5M and receives $10M return on success.

4. Aggressive: VC invests $10M per startup and earns $25M in return.

For each of these scenarios, we plotted the profit curves for our top four models—Random Forest, CatBoost, XGBoost, and Stacked Ensemble—to identify the optimal investment threshold (i.e., percentage of test set startups to fund) that maximizes profit. Our main focus, however, was on the Random Forest model, given its strong recall and business alignment.

Random Forest Profit Analysis

The optimal percentage of startups to invest in — and the corresponding maximum profit — varied depending on the investment scenario:

- Bad Market: Invest in top 55% of startups to maximize profit of $740 million

- Conservative: Invest in top 79% of startups for a maximum profit of $3 billion

- Moderate: Same 79% threshold yields a profit of $5.8 billion

- Aggressive: Again, investing in top 79% yields the highest profit of $15.28 billion

Model Comparison and Strategic Use

These profit curves can also be used to compare different models based on a fixed investment budget or startup selection quota. For example, if a VC firm plans to fund only the top 60% of startups based on predicted probability, we can use the profit curves to determine which model produces the highest expected return under that constraint.

While these scenarios are simplifications, they demonstrate how machine learning predictions can be integrated with business context to inform capital allocation strategies. They also highlight the flexibility of our models in adapting to different levels of risk tolerance and expected return.

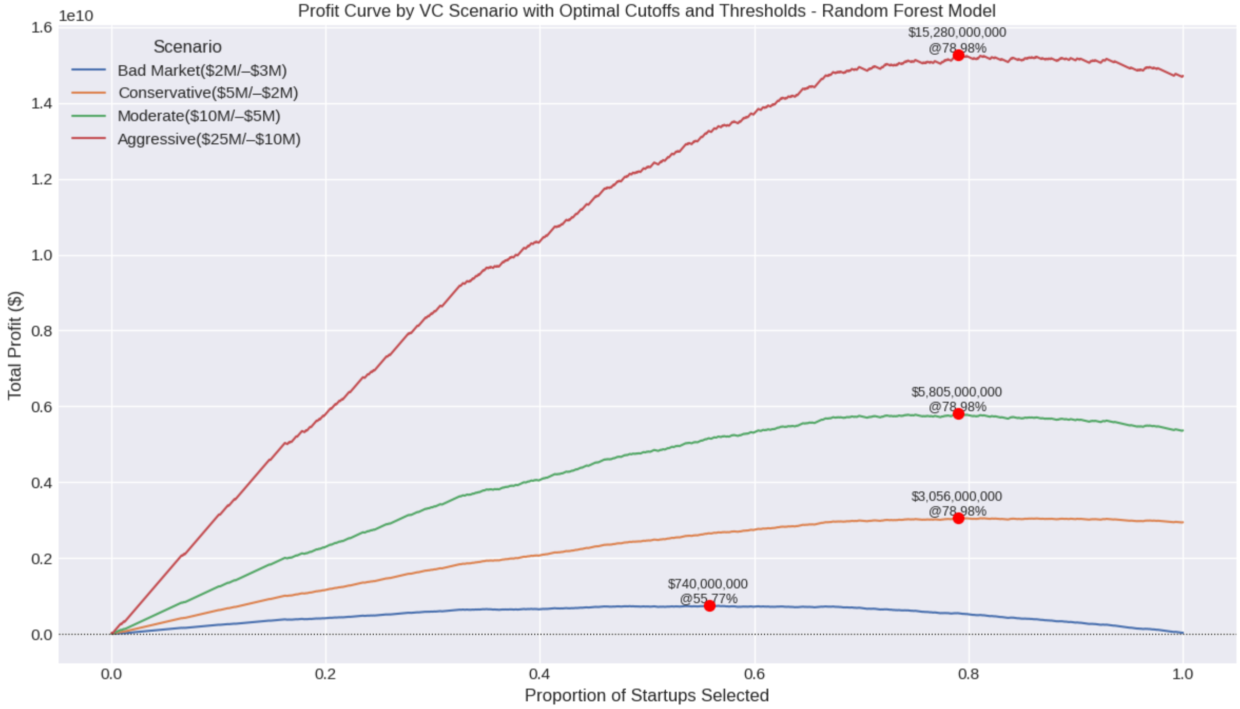


Figure 6.2: Profit Curve - Random Forest

Limitations

It is important to note that these simulations are hypothetical and do not capture the full complexity of startup investing. Real-world decisions involve nuanced due diligence, varying deal sizes, market dynamics, follow-on funding rounds, and many non-quantifiable factors. Still, our goal here was to show the potential practical use case of our predictive model—serving as a decision support tool to complement, not replace, human judgment in venture capital investment.

7 Future Work

While RiskLens demonstrates promising predictive power using public startup funding data, several important directions remain for improving both predictive performance and real-world utility. This section outlines future extensions spanning data enrichment and deployment.

7.1 External Data Integration

To improve feature richness and model accuracy, future iterations of this project should incorporate additional external data sources. Specifically, integrating founder-level information—such as educational background, prior entrepreneurial experience, or network centrality—could offer strong signals of early-stage success potential. Likewise, data on patent filings, intellectual property, and technological assets may enhance predictions, especially for science-driven sectors like hardware and biotech. In addition, macroeconomic indicators and market trends (e.g., hiring rates, sectoral investment flows) can help contextualize a startup's position relative to its environment.

7.2 Survival Analysis

Beyond binary classification, time-to-event modeling techniques such as Cox proportional hazards models or survival trees could be used to predict time-to-exit. Estimating how long a startup will take to reach acquisition, failure, or IPO would provide more nuanced risk estimates and support portfolio-level planning for capital allocation and liquidity timing.

7.3 Real-Time Monitoring and Drift Handling

Given the dynamic nature of the startup ecosystem, it is crucial that the model remains up to date. We propose building quarterly or real-time data ingestion pipelines capable of detecting and responding to:

- **Data Drift:** Changes in the input feature distributions.
- **Prediction Drift:** Shifts in the output class distributions.
- **Model Staleness:** Degradation in model performance due to evolving business patterns.

Integrating automated retraining based on these signals will help maintain model relevance and robustness in production environments.

7.4 Proposed Model Deployment & Operations

Once a final model—such as the Random Forest classifier—is selected and validated, we recommend deploying it via a batch inference pipeline, supported by monitoring and retraining infrastructure.

Batch Inference Pipeline

The pipeline can be triggered manually by analysts or scheduled (e.g., weekly). The process involves:

- Loading the latest startup data from internal or external sources.
- Generating success probability predictions using the trained model.

- Saving predictions in a structured format (e.g., CSV or BigQuery).
- Serving outputs to dashboards, internal tools, or reports.

Monitoring and Retraining

As described above, the system would monitor for drift or staleness and automatically retrain on new labeled data when appropriate.

Towards a Continuous Learning System

Over time, this architecture can evolve into a continuous learning loop—where predictions drive decisions, decisions generate new feedback data, monitoring identifies shifts, and retraining keeps the model aligned with reality. This feedback loop ensures RiskLens remains accurate, adaptive, and valuable in a rapidly changing venture ecosystem.

8 | Conclusion

In this project, we developed RiskLens, a machine learning-based system designed to predict startup success using publicly available funding and categorical data. Through rigorous data cleaning, thoughtful feature engineering, and extensive model evaluation, we demonstrated that it is possible to extract meaningful insights about a startup's likelihood of success—even with limited, noisy, and imbalanced data.

Our modeling experiments showed that ensemble methods like Random Forest and XGBoost outperform simpler baselines such as Logistic Regression, achieving AUC scores above 0.79. Features related to funding momentum, industry sector, and startup maturity were among the most predictive. However, we also recognized the structural limitations of the dataset, including survivor bias and missing founder or operational data, which may impact generalizability.

Despite these limitations, RiskLens provides a scalable and interpretable framework for early-stage venture risk analysis. It can support venture capital firms in prioritizing diligence pipelines, validating hypotheses, and complementing human intuition with data-driven evidence.

Looking ahead, incorporating external data sources, adopting time-aware modeling techniques, and establishing real-time monitoring pipelines will help extend the system's reliability and practical value in live investment settings.

9 | Appendices

Appendix A: Project Code and Datasets

All project code, data pre-processing, model training, and analysis were implemented in Python using the Google Colab platform. The following links provide access to the full code notebook and datasets used in this project:

- **Google Colab Notebook:** Click here to view the notebook
- **Original Dataset (Crunchbase Export):** Google Drive Link
- **Cleaned and Preprocessed Dataset:** Google Drive Link

These resources are provided for full transparency and reproducibility of the results presented in this report.

Appendix B: Group Contribution

We would like to emphasize that the success of this project is attributed to the equal and significant contributions of all group members. Each phase, including coding, data analysis, result generation, and the drafting of the presentation and final report, was collaboratively completed.

Bibliography

[1] Crunchbase Staff. *Where does Crunchbase get their data?* https://support.crunchbase.com/hc/en-us/articles/360009616013-Where-does-Crunchbase-get-their-data. 2025.