LUISS

Degree Program in Management and Computer Science BSc

Course of Data Analysis for Business

Predictive modeling for identifying Success Factors in Startups: An empirical Analysis.

Prof. Iafrate Francesco

Supervisor

Marc Violides
260221

Candidate

Academic Year 2022/2023

Acknowledgements

Allow me to start by expressing my utmost gratitude to professor Iafrate for his unwavering availability and patience, but most importantly for the passion he instilled in me with regards to the field of Data Analysis, and most certainly when it comes to applying it to real-world scenarios in order to help shape the world in a more positive light.

I would also like to extend my gratitude to my parents for providing me with the opportunity to be a part of this esteemed university through their unconditional love and support. Even though my late father is not here to witness my achievements, I am confident he is watching from above and supporting me all the way. I hope to someday make them proud through the impact I will have in my future endeavors, using this thesis as my launchpad to a world of greatness and limitless opportunities.

A special thanks also goes to Crunchbase and especially Tim Li from the support team for providing me access to their startup data. Without this data, I would not be able to write this thesis with the most up-to-date information, and my conclusions would not be as impactful as they currently are.

Abstract

This empirical study investigates the potential of machine learning in forecasting company success in order to assist investors in making informed decisions. We identify startup hurdles and characteristics that contribute to their success using data from Crunchbase, a community-based platform that gives credible information on companies. After careful feature engineering and variable selection, we train a variety of classification models, including non-linear and tree-based models like random forests and decision trees, linear models like AIC, and penalized approaches like Lasso. The original random forest was found to have the best accuracy at 85%, which was later increased to 86% through random search and grid search. This is superior to the majority of papers done on similar topics using Crunchbase data. It also has extremely elevated precision, recall and F1 Score with each scoring 88%, 97%, and 92%, respectively. Our findings imply that machine learning might be a beneficial tool to eliminate the guesswork in investing in startups. This thesis includes descriptive statistics, data visualizations, bivariate analysis, correlation and regression analysis, and model comparison.

Table of Contents

01. Introduction

a. Background and Motivation

The world of startups has seen a surge of activity in recent years, with a large number of entrepreneurs trying to shake up the status quo and come up with new and innovative solutions to problems. On the surface, startups are a breath of fresh air for many investors that are seeking to diversify their portfolios, because more times than not they are led by entrepreneurs which have great aspirations. However, these aspirations can sometimes lead to a disillusionment of reality and can come at a steep cost for both the entrepreneur and the investor. As we will repeat time and time again throughout this thesis, according to Forbes, up to 90% of startups fail. This number might seem fictional, but it is far from it and very much a reality. Having said that, this high failure rate does not discourage new entrepreneurs from continuing to flock to the startup scene, and it certainly does not stop venture capitalists and other investors to believe in the ideas that are being presented to them. Having this bulk of startups burst onto the scene might seem optimal for the country's overall economy, but it gives a headache for investors who believe in the ideas the entrepreneurs are selling them. One reason that startups are so different to traditional matured companies is that they do not have sales, profits and other financial statements instruments that can back their claims, or if they do, these do not span over a long enough time period to observe reliable patterns. This leads to a lot of uncertainty in the startup scene and consequently a lot of bad investments in the long-term. Though no silver bullet guarantees a startup's triumph, data-driven insights and predictive modeling can aid investors in making informed judgments.

Existing research usually relies on survey data that lacks generalizability and focuses solely on the US. This is why in this paper we take a wider approach by covering several continents with the hope of helping as many investors and entrepreneurs as possible.

b. Research Questions and Objectives

By leveraging historical data and identifying patterns in prosperous and unsuccessful firms, investors can more accurately evaluate a company's potential for success. We have the capacity to reduce guessing and give a dependable approach to navigate the uncertain world of startups by harnessing the power of data.

Many entrepreneurs are attempting to disrupt existing corporate structures and solve problems in novel ways, creating a crowded startup ecosystem. The high rate of initial failure, on the other hand, implies that creating a profitable corporation is a difficult task. The goal of this study is to investigate the potential of

machine learning in forecasting startup success. More specifically, my goal is to provide a solution for these investors by predicting whether or not to invest in a startup based on previous data from organizations with comparable characteristics.. Of course, keep in mind that this is merely advice and should not be taken as a general fact.

As a result, we might ask ourselves whether machine learning algorithms can properly and reliably predict startup success. This sentence will serve as the central topic for our thesis. We have established three study objectives to address this issue. To begin, I plan to employ machine learning approaches to identify the essential characteristics that contribute to company success. Second, I will investigate how well different machine learning techniques predict startup success. Lastly, I plan to evaluate and improve on the best model to give out the best advice for investors looking to put their funds in a startup.

c. Contribution and Significance of the Study

Through this work, I hope to provide assistance to prospective investors who are willing to invest in a startup but may have reservations to do so because of the high perceived risk that comes with this investment. Hopefully, this study will alleviate the conception of startups being inherently risky ventures. Of course, risk can only be mitigated and not completely eliminated, as risk is present in every investment opportunity. Ultimately, my hope is that this study will reduce the gap in risk perception between investing in established companies and startups who just burst onto the scene.

d. Thesis Organization

The thesis is structured in such a way that it guides the reader through the research process. The introduction sets the stage for the research by defining the study's scope and constraints. What is more, it also provides context, research questions, and objectives.

The literature review part provides an overview of current research on startups on one hand, and states the role of machine learning in the success of startups on the other hand. In the first section, we will start by defining startups and see their characteristics, before delving into the challenges of the startup scene. Lastly, we will also be interested in the factors that are perceived to make startups successful by the literature, and the models that have proven to work over time. While in the second section, we will be

interested in the emergence of machine learning onto the startup scene and look at some previous studies that have been made on predictive modeling for startup success.

The methodology section starts by outlining the research design, as well as the data collection and processing techniques to go from a multitude of crunchbase tables to a single one through feature engineering and variable selection. After that, I describe the variables and measures that are present in our merged dataset by checking for data inconsistencies, addressing class imbalance issues present in the target variable, and taking a look at how correlation matrices and regression analysis can help with feature selection. Then, we will be interested in the predictive modeling techniques that will be used in this work. Finally, we will take a look at some evaluation metrics that will be extremely useful in understanding how our model is performing.

The results and analysis part goes on to provide univariate and bivariate analyses to grasp the structure of each variable individually, and to understand the relationship between the independent variables and the target feature, as well as the one between the independent features themselves. We will then go on and perform a correlation and regression analysis in order to select the variables which we deem useful and impactful for the modeling phase which follows. We later compare the models, take the best performing ones and seek to optimize them in order to achieve optimal results.

In the discussion section, I will interpret the findings and see what their practical implications are. I also plan on comparing my findings with those of similar studies to see whether my model outperformed theirs. In this part, I also discuss the limitations of the study.

Lastly, I conclude by stating the key takeaways from the project and the main results. I also include a reference section where I include all of the references that were useful throughout this thesis, as well as an appendix part which includes the python scripts that I deem as essential for this project.

02. Literature review

a) Overview of Startups: Definition, Challenges and Success Factors

i) Definition and characteristics of startups

Startups have become an important engine for innovation and economic expansion. Although most people believe that describing what a startup is is a simple undertaking, there is no generally applicable description that explains exactly what a startup is. Different definitions place varied emphasis on elements like the company's age, its potential for expansion, or the inventiveness of its goods or services. For instance, the European Commission defines startups as young, innovative businesses with high-growth potential, in contrast to the US Small Business Administration, which defines them as enterprises that are younger than five years old and have fewer than 500 employees (Henrekson & Sanandaji, 2018). But despite the variety of definitions, startups share a few key traits that are agreed upon in common literature.

First and foremost, startups are renowned for being dynamic and creative. They are frequently started by entrepreneurial visionaries who have a burning desire to solve specific issues and are not afraid to take calculated risks in order to succeed (Shane, 2000). Startups typically work in an atmosphere that is fast-paced and constantly changing and are able to make quick decisions (Blank, 2013). What is more, they often have a small, agile staff that can change direction and adjust to quickly shifting market conditions (Ries, 2011).

Another important characteristic of startups that is widely recognized is their high potential for growth. This is largely thanks to the fact that they are not constrained by the conventions of legacy systems, procedures, and cultural norms. Instead, they are free to test out novel theories and methods for tackling issues. Due to their flexibility, startups may expand effectively and at rates that are frequently significantly higher than those of more established companies. Therefore, entrepreneurs have the ability to establish new markets while upending those that already exist (Henrekson & Sanandaji, 2018).

What is often neglected by the general public is the significant acceleration in job creation and economic growth that the startups lead to. Indeed, a fundamental role is played by startups in promoting technological development and producing novel goods and services, both of which ultimately benefit the entire community by creating opportunities that did not previously exist. Consequently, the impact of startups may be direct through the generation of jobs within their own companies, but it also has an indirect

effect through the knock-on impact that strengthens regional economies.

Nowadays, startups are more frequently associated with technology use since they make use of online tools and platforms to develop new goods and services. For instance, fintech businesses employ technology to disrupt traditional banking and financial services (Fuglie, Heisey, King, Day-Rubenstein, & Schimmelpfennig, 2020), while Healthtech startups use artificial intelligence and machine learning to enhance healthcare outcomes (Chen & Asch, 2017).

To sum up this first section, we can say that startups stand out for being inventive, dynamic, and always evolving. Although there is no agreed-upon definition of what makes up a startup, they are universally acknowledged as being important sources of innovation and as being major forces in driving economic growth and the creation of jobs (DeTienne & Chandler, 2007).

ii) Challenges and reality of the startup scene

The startup culture is frequently portrayed as glamorous and thrilling, with tales of young entrepreneurs becoming billionaires overnight. However, this is only the tip of the iceberg, as funding a startup is similar to the lottery in that what you see on TV is only the winners: you do not see the huge amount of people who failed. Indeed 90 percent of startups fail according to Forbes.. This statistic puts in evidence the difficulty of the environment in which startups operate. But the question is: what is the reason for this astonishingly elevated failure rate?

To start with, the first challenge that faces startups is a lack of resources. Effectively, more often than not, startups are handed a limited amount of funding and until they demonstrate their market viability, they rarely receive considerable investment. This lack of funds and trust leads to startups not fulfilling their potential, and consequently competing with larger, more established organizations becomes extremely challenging (DeTienne et al., 2019). Furthermore, from a Research and Development perspective, the limited resources make it a struggle for the startups to acquire top people that know the ins and outs of the industry, making it even more difficult to compete (Lepak et al., 2019).

Startups encounter a challenge with the dynamic and evolving business environment. Adapting to shifting market conditions and consumer demands can be especially difficult for startups that develop innovative technologies or products, as there may be limited demand or understanding of the potential benefits. To maintain long-term survival, companies must strike a balance between innovation and revenue

(McGrath, 2010).

In addition, startups may encounter legal and regulatory hurdles, particularly if they operate in heavily regulated sectors like healthcare or finance. Complying with regulations can be expensive and time-consuming, and non-compliance may result in significant penalties and legal consequences

Eventually, startups could encounter obstacles in expanding their operations as they develop. Growth comes with a unique set of challenges, such as managing larger teams, guaranteeing quality control, and maintaining financial management. Anticipating and tackling these challenges is necessary for startups to accomplish long-term growth sustainably (Peng et al., 2019).

So it can be concluded that the world of startups presents a formidable and competitive landscape with various obstacles on the path to success. These hurdles include limited resources, constantly evolving market dynamics, regulatory compliance, and difficulties in scaling operations (Naldi et al., 2017). Having said that, startups have the potential to make enormous contributions to the economy and society if they successfully overcome these difficulties.

iii) Success factors and models for startups

Even though the startup scene is filled with challenges, there have been many unicorns over the course of time. By unicorn we are referring to the term coined by venture capitalist Aileen Lee back in 2013, describing a startup that has attained a valuation of $1 billion or more. A huge part of the startup literature has tried to explain the reasons for these large-scale successes, but only three models have emerged as benchmarks to explain this phenomenon. The models which gained the most traction are the lean startup model, the business model canvas, and the effectuation model, all of which we will study in more detail in this section. Through these models, we are able to obtain key insights into the factors that influence startup success and help entrepreneurs in making informed decisions in regards to launching and expanding their businesses. Of course, we must remember that these models are simplistic by nature and are far from reality.. In order to succeed, entrepreneurs are encouraged to take into account these models, but there is no trivial solution.

The lean startup model is a methodology developed by Eric Ries in 2011. It focuses on continuous innovation by creating a minimum viable product (MVP) and using customer feedback to improve upon it.

This model aims to reduce the risk of failure by emphasizing rapid experimentation and constant learning. Moreover, it stresses the importance of flexibility and adaptability, meaning startups must be open to new ideas and willing to pivot when necessary to achieve their objectives.

The business model canvas, created by Alexander Osterwalder and Yves Pigneur (2010), is another model that has gained significant popularity in recent years. It offers a visual framework to help startups understand their business models' crucial components, such as value propositions, customer segments, revenue streams, and cost structures. By mapping out these elements, entrepreneurs can gain a clearer perspective of their business and make informed decisions on resource allocation, making them better equipped to succeed in the competitive startup ecosystem.

Lastly, the effectuation model, developed by Saras Sarasvathy in 2001, highlights how entrepreneurs think and act differently from traditional managers when establishing new ventures. Effectuation focuses on the utilization of existing resources and the formation of partnerships to create new opportunities. This model is particularly relevant for startups with limited capital, as it offers a framework for creating value despite resource constraints.

As alluded to earlier, merely following one of these models is not a recipe for success. Effectively, there would be far more unicorns on the scene if this were the case. The reason entrepreneurs get the credit they get is because they not only take into account, but also execute upon some critical factors that can make or break a company. First, having a well-rounded team that is able to complement each other’s skills and share a clear vision of the company’s future can go a long way in the success of the company (DeTienne & Chandler, 2007). This factor seems trivial, but anyone who has managed people in any way knows that people management skills are the most valuable to have when making something out of nothing. Second, having thorough awareness of their target market and knowing what distinguishes them from their competitors is what every entrepreneur should strive to accomplish (Shane, 2000). That being said, knowing your strengths is great, but acting upon them takes the company to a whole new level. Furthermore, having the ability to quickly pivot and react to changing market conditions is crucial in the company’s long-term success (Blank, 2013). Indeed, when things are going well, it is easy to operate, but the real value of the entrepreneur comes out when being faced with an unusual turn of events, like the Covid outbreak for example. By prioritizing these characteristics, startups are expected to boost their chances of success and thrive in today's competitive business world.

b) The role of Machine Learning in the success of startups

i) The emergence of Machine Learning in the startup scene

The emergence and rapid growth of Machine Learning (ML) has seen many advocates for it to be used in startups. Most notably, the entrepreneurs themselves are starting to acknowledge the importance of Machine Learning as a key factor for their business to strive. This has led to a constant growth in the amount of startups opting for ML. In Layman terms, ML refers to a subset of artificial intelligence that enables computers to learn from data and improve their performance on a given task over time (Mitchell, 1997). What is more, the use of ML in the startup ecosystem is not only important to facilitate business operations, streamline their operations and improve their products and services, but it also allows startups to better understand their customers through the insights gained from the aforementioned tools.

To begin with, a leading use of ML in the field of startups has been the adoption of predictive modeling. Predictive modeling is a statistical technique that consists in forecasting future results by relying on historical data. Through the extensive use of ML algorithms, startups are able to analyze vast amounts of data in order to detect patterns and make predictions about future trends. If done correctly, these predictions have the potential to save companies hundreds and even millions of dollars, depending on the company size. Let's take the example of a startup that operates in the e-commerce industry. The adoption of predictive modeling by that startup will allow it to forecast future sales based on historical customer behavior. This approach has been used successfully in several studies, including the work of Perifanis et al. (2021), which showed that through the use of ML algorithms in sales forecasting, business decisions and the accuracy of forecasts drastically improved. Additionally, another crucial work in this field was that of Singh et al. (2020), which showed the effectiveness of using ML techniques for sales forecasting in the Indian e-commerce market.

Another ML application that has taken the startup world by storm is the advent of natural language processing (NLP). To explain it in simple terms, NLP is a branch of ML that focuses on the interaction between computers and human languages. The most notable industries which saw a boom in the use of NLP are customer service and healthcare where the aforementioned technology eliminates repetitive tasks, ranging from answering some trivial questions individuals may have to diagnosing the condition of some people based on certain information. Studies have shown that NLP can significantly improve the efficiency and accuracy of customer service tasks, as demonstrated by the work of Mah et al. (2021) and Murali et al. (2021). In this study, Mah et al. (2021), tested out how effective an NLP-based customer service platform would be for an online education platform. To train it, Mah et al. used an immense dataset of inquiries and

responses that they get from customers. The study found that customer satisfaction grew at unprecedented levels mainly due to reduced response times, and allowed employees to focus on other tasks.

Furthermore, another application that goes under the radar in relation to the application of ML to the field of startups is the use of ML for fraud detection, image and speech recognition, as well as recommendation systems. The use of ML by the startups in these areas are drastically improving the efficiency and reliability of their systems, making them more competitive in their respective markets. For instance, the study by Bhattacharyya et al. (2011) showed that ML algorithms could detect fraudulent transactions with high accuracy. In this work, Bhattacharyya et al. trained and tested their ML models based on a large dataset of credit card transactions. Some ML algorithms, like decision trees and neural networks, proved to be highly accurate in detecting fraudulent transactions. Another study that serves as a landmark in this domain is that of Himeur et al. (2019), which demonstrated the effectiveness of using ML for personalized recommendation systems. In their work, Himeur et al. trained their model based on users' interactions with items in order to develop a recommendation system tailored to their needs. Through this study, it was demonstrated that ML-based recommendation systems were able to outperform collaborative filtering and content based filtering, which are still in use in most companies nowadays.

As we know, it is not all sunshine, because with potential benefits of ML for startups, comes a wave of challenges that could disrupt these entities if not dealt with correctly. These include the elevated cost of implementing ML systems, the need for specialized expertise who know how to deal with this kind of technology, and most importantly the ethical implications of using ML in decision-making processes. These three challenges are not mutually exclusive, because in order for a startup to reach great heights, it has to invest heavily in technology, and especially in ML systems. However, having the technology is not enough, but it has to be complemented by the presence of experts who know how to operate them. What is more, being ethical goes a long way in reaching the ultimate goal of success in that it brings a huge amount of traction to the business by painting it in a positive light. Several studies have evidenced the importance of addressing these challenges, including the work of Jordan and Mitchell (2015) on the ethical implications of using ML in decision-making and the study by Géron (2017) on the importance of having a solid understanding of ML concepts before implementing them in a business context. In the former study, Jordan and Mitchell argue that biased data can lead to biases and discriminatory behavior of the ML model, as we have seen with many chatbots before the advent of Chatgpt, notably Microsoft's Tay chatbot. Consequently not taking appropriate safeguards and oversight to prevent bias and discriminatory behavior can have dangerous consequences. In the latter work, Géron states that it is one thing to understand that ML is able to provide significant benefits for the business, and another thing to implement it without any issues. Géron's recommendation is that of investing in the brain power of the company through intensive training and courses to ensure that the employees have a good grasp of how these technologies work and use it to their

full potential.

ii) Previous Studies on Predictive Modeling for Startup Success

In this section we will focus more on the studies that covered predictive modeling for startup success, which is one of the applications of ML that is invading the startup ecosystem and which we saw earlier on. We focus particularly on this aspect since our goal is to provide investors and entrepreneurs with a recommendation on which startups will succeed and which are more likely to fail. In other words, we are building a predictive model that is able to determine startup success.

We witnessed the influence of predictive modeling on the e-commerce industry in the previous section, but this is only one facet of its broad variety of applications. Indeed, predictive modeling has become an essential component of the startup ecosystem in a wide range of businesses.

To start with, the health care sector saw an early adoption in predictive modeling which led to being crucial in improving patient outcomes, as well as reducing healthcare costs in general. The most notorious study was that of Park et al. (2020) which found that ML-based predictive models had the capability to predict the risk of hospital readmission for heart failure problems. What is more, the work of Rajkomar et al. (2018) showed that these models allowed healthcare providers to identify patients that are at risk through a high accuracy in predicting patient mortality rates, allowing the medical staff to intervene early.

Another sector in which predictive modeling was largely adopted is that of banking and finance. Effectively, predictive modeling was able to drastically improve financial forecasting and risk management, allowing for an overall more functioning economy. Some studies related to this subject were done by Cano et al. (2014), and Himeur et al. (2020). In the former, Cano and the other researchers discovered that a large part of predictive modeling improving the banking sector was due to its ability to accurately predict credit risk. Consequently, this allowed banks to make better lending decisions and reduce losses that are caused by lending money to people who are not able to repay their debt.

The last field that predictive modeling has significantly impacted is the field of marketing and advertising. Customer segmentation and the personalizations of marketing campaigns were optimized to a huge extent by the inheritance of predictive modeling. The study done by Park et al. (2019) is a perfect advocate of that. Indeed, this study demonstrated how customer behavior and preferences were accurately predicted by ML-based models, so that marketers were able to tailor their messages and offers to individual

customers. This singularity allowed for a more intimate relationship between the customer and the marketer, hence improving customer satisfaction, logically followed by an increase in sales. Other studies that were undertaken added another layer on top of that by demonstrating the effectiveness of ML-based models in predicting customer churn and developing targeted retention strategies.

Overall, these studies highlight the importance of predictive modeling in a variety of industries, as well as the potential benefits for firms who employ it. By carefully studying user behavior, projecting future trends, and identifying areas of risk and opportunity, startups may make data-driven decisions that lead to success and growth.

03. Methodology

a) Research design

In this section, we outline the overall approach and plan of the study. As I seek to do in the entirety of the paper, I will try to make the approach as understandable as possible for the general public. The research question was formulated in the introduction and was basically centered around whether we could exploit machine learning tools to predict startup success in a more accurate way than a human would. The focus of the study is restricted to startups aged 10 years or less. This age is very specific because even though we acknowledge that the common consensus among the startup literature is that a company is in its startup phase in its first five years, it is also true that when a company exits this phase, it faces the real adversity of the real world outside its startup bubble. Hence, this is why we chose to extend the selected companies' age by 5 years in order to guarantee that the company was not subject to bankruptcy or failure in the short to medium term. Among the study's flaws include potential data biases due to the public nature and community-based nature of the platform from which the data is collected from. Data cleaning, exploratory data analysis, feature engineering, and predictive modeling with machine learning algorithms are some of the approaches and techniques used for data collecting, analysis, and interpretation.

b) Data Collection and Processing from Crunchbase corpus

In this section I will take you through my thought process to collect and process the data which are located across many tables in order to have a unique dataset. The data used in this work is collected from Crunchbase through the education license they granted me with for the purpose of my thesis. Crunchbase is one of the largest and most comprehensive startup ecosystem databases, with information on firms, investors, significant people, and events. Despite its community-based nature, the credibility it has gained throughout the years has made it to be the reliable source it is today.

In the spirit of transparency and scientific honesty, it is essential in my opinion to include the main tables and variables that were present in the Crunchbase corpus and explain the reason for their inclusion or exclusion from my work. This is exactly what I will be doing in the next subsections. What is more, this makes my work more reproducible in case someone else is getting their data from Crunchbase, and is interested in predicting startup success. As a result, the study's breadth and rigor is ensured, while also

allowing readers to evaluate the validity and application of my technique.

i) Table selection

As of today, the CSV export tool of Crunchbase consists of 17 tables, but only 4 of them have been used in my work. The selected tables are organizations, people, degrees and funding_rounds and the reason for their inclusion is presented in table 1 below . Also, another main reason for wanting to select these tables is that they are all connected in some way through their primary and foreign keys, as can be seen in the ERD diagram of figure 3.1. For completeness purposes, I also show the tables that I did not select to be part of my work in table 2.

Table (csv files)	**Description (From Crunchbase)**	**Why include it in our project?**
Organizations	Organization profile that is available on the Crunchbase platform.	This will be considered as the main dataframe, to which we will merge the others because it contains the most useful variables for the purpose of our project.
People	People profile available on the Crunchbase platform.	As we saw in the literature review part, human capital plays a crucial role in the success of a startup, and understanding the background of the founders, as well as the company's team composition can go a long way in grasping how functional the company is and its potential.
Degrees	Detail for people's educational background.	It is no secret that a large number of startup entrepreneurs are college dropouts, having great aspirations. This table will help us to see whether this fact is true, and to see whether the number of degrees a founder has is able to shape the success of a startup.

Table (csv files)	Description (From Crunchbase)	Why include it in our project?
Funding_rounds	Details for each funding round in the dataset	This table is crucial to understand the number of funding rounds startups undergo on average and whether a lot of funding goes into early stages of the business (pre-seed, seed) or rather into the later stages (round D and later)

Table 1: Crunchbase Tables included in thesis

These are the selected tables that I believe bring an added value to the project. The ones I excluded do not bring any additional information for the purpose of our work: let us go through them one by one because it is extremely important to understand the exclusion of tables that could be seen as useful, but in reality can be hurtful in our case. Indeed, having a lot of variables can lead to overfitting, and this is worsened when the variables are not meaningful.

Table (csv files)	Description (per Crunchbase)	Why exclude from our project?
Organization_description	Long descriptions for organization profiles.	This provides too much detail that is not interpretable in a Machine Learning model, unless we use NLP tools. This is outside the scope of this thesis but could be looked up to build up on it.
people_description	Long description for people profiles.	The same reason as the one stated above.
acquisitions	List of all acquisitions available on Crunchbase platform.	Only a limited number of companies have been acquired compared to the overall number of startups present on Crunchbase, so having this table would lead to lots of missing values when merged with the ones we chose.

Table (csv files)	Description (per Crunchbase)	Why exclude from our project?
org_parents	Mapping between parent organizations and subsidiaries.	This would provide too much unnecessary granularity that is outside of the scope of this project.
ipos	Detail for each IPO in the dataset.	Only a limited number of the startups present in the dataset have gone public, so a lot of sparsity.
cateogry_groups	Mappings between organization categories and category groups.	We already have a column in the organizations data frame that takes care of this variable, leading this table to become completely irrelevant and redundant.
jobs	List of all jobs and advisory roles.	Job titles are included in the people table which we already selected, so the selection of the jobs table would lead to redundant information once again.
investors	Active investors, including both organizations and people.	What we care about is the company itself, not the people investing in it. Our ultimate goal is To help people investing in the company, but we have enough information in the organizations table like the total number of investors and total amount of investments.
investments	All investments made by investors.	Same reason as to why we excluded investors.

Table (csv files)	Description (per Crunchbase)	Why exclude from our project?
investment_partners	Partners who are responsible for their firm's investments.	This is extremely further away from the purpose of this work, and it is definitely not an aspect that we are interested in.
funds	Detail for investors' investment funds.	This is also out of the scope of our project. funding_rounds provides much more information in that regard.
events	Event details.	This table is of no use to us for what we are trying to accomplish.
event_appearances	Event participation details.	Since events are not relevant, this is even more irrelevant since it is a detail of the former.

Table 2: Crunchbase Tables excluded from thesis

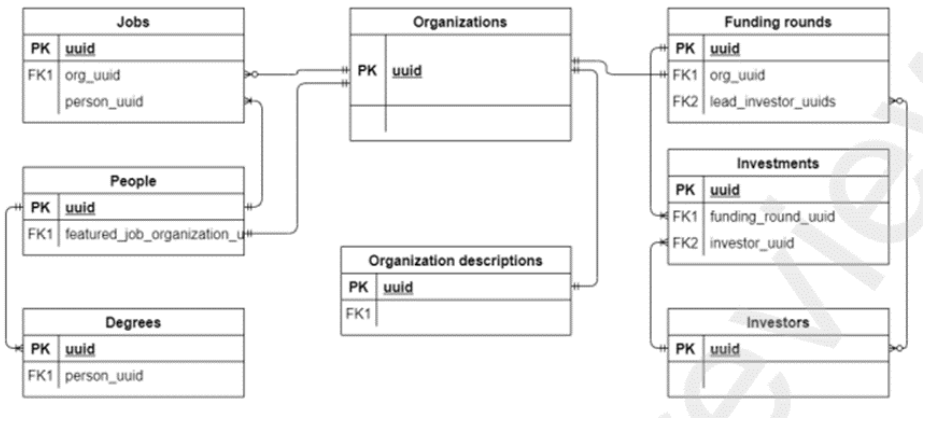

Figure 3.1: Simplified ERD diagram of Crunchbase data

Source: Predicting the Success of Startups using Crunchbase and LinkedIn Data. Yiea-Funk Te, Michèle Wieland, Martin Frey, Asya Pyatigorskaya, Penny Schiffer, Helmut Grabner

We can observe that the organization table is the main dataframe in the sense that most foreign keys in the other tables are referring to its primary key. Let us now better understand how each of the selected tables are

linked, before starting to analyze them individually:

Organizations and people: The people table has a variable called "featured_job_organization_uuid" which directly references the company in which these people are working from the organizations table.

People and degrees: degrees is not directly linked to the main data frame organizations, but rather goes through the people table first. It is linked to the people dataframe by the fact that "person_uuid" references the person that has obtained that degree from the people table.

Organizations and funding_rounds: Funding rounds is directly linked to the organization table by the fact that org_uuid in the former references the id of the organization present in the latter table.

ii) Individual Table analysis and feature engineering

Let us now take a closer look at each table and understand which variables we selected from each table, while also taking a peek at why we excluded the others. You might notice that I have selected a few of the present variables, but my slogan for this thesis is: **"data quality over data quantity"**.

In this subsection, we also take a look at the feature engineering that the original data went through in order to arrive at a satisfactory merged table.

Starting with the "organizations" table, it is composed of 39 variables. This is by far the largest table of them all and we will focus on it the most since it will serve as the main table in our work, to which we will add the other tables. As can be seen in table 3, I went through its main variables and decided which ones to keep and which ones to exclude, stating the reason for their inclusion or exclusion. I follow the same pattern for the other selected tables, as can be observed for the People, degrees and funding_rounds tables in table 4, table 5 and table 6, respectively. After presenting which variables I selected from each table, a feature engineering section follows each table in order to tailor the variables for the type of problem I am solving. In many of these cases, I am looking to aggregate information from tables where one organization can be present more than once, in order to limit it to one observation in the main dataframe.

Variable(s)	Description	Selected?	Reason for inclusion / exclusion
uuid	This is a unique identifier for each organization.	Yes	This will be useful to merge tables, but not for ML problems
status	Is the company operating, acquired or closed?	Yes	This will serve as our target variable
primary_role	Is it a company or a school?	Yes	We are interested in filtering only the rows with companies
country_code	3 letter code that refers to a country	Yes	This will be useful for a further transformation we will do later on when diving countries by continents
facebook_url linkedin_url twitter_url	Social media links of the companies	Yes	Might seem irrelevant, but thanks to feature engineering will be of great use
employee_count	Range of employees present in the company	Yes	This will be useful for us to understand size of company and will be great to serve as input in ML model
founded_on	When was the company founded	Yes	This is useful for us to understand how old the company is. Is it still in the startup phase or has it matured?
last_funding_on	When was the last funding contributed to the company?	Yes	This is useful to understand how dependent the company is on external funds and other uses
total_funding_usd	What is the amount of funding the company received?	Yes	This variable is useful because we get to know the amount of funding that the company has received.

Variable(s)	Description	Selected?	Reason for inclusion / exclusion
num_funding_rounds	How many rounds of funding did the company undertake?	Yes	Useful to be fed in the ML model, as it could explain whether the amount of funding rounds can lead to more success.
category_groups_list	Which category group does the firm belong to?	Yes	The sector in which a company operates is crucial for startup success in my opinion, as the environment plays a huge part.
category_list	Which category does the organization belong to?	No	This is less general than category groups and thus we have much more values, inconvenient for ML models.
closed_on	When did the company close?	No	This is of no use since most companies in our dataset did not close, so a lot of nulls.
state_code	Refers to which US state is the company located on	No	This refers to states in the US, but we are not only taking the US as a country, so this variable is irrelevant
region city	Refers to which region and which city the company is located in	No	This is too wide for the purpose of our work, and country_code will be the only useful location variable for us
address postal_code	Organization's address and postal code	No	Provides too much granularity location-wise
Name Legal Name	Organization's name and legal name	No	The uuid is enough to identify the company
roles	All the roles undertook by the organization	No	We already considered primary_role, no need for this variable
type	Similar to primary_role	No	We already considered primary_role, no need for this variable

Variable(s)	Description	Selected?	Reason for inclusion / exclusion
permalink cb_url domain homepage_url logo_url	All these are useful for the company URL	No	We have no interest in them
rank	Useful for crunchbase to locate companies.	No	We have no interest in it
created_at updated_at	When the company profile was created/updated on crunchbase	No	We have no interest in them
alias 1 alias 2 alias 3	Alternative names for the company	No	This is hugely irrelevant for us
num_exits	Number of exits from the company	No	Might be useful, but I found that consists mostly of blanks
email phone	Ways of reaching out to the company	No	It is of no use to us
short_description	A short description of the company	No	It complicates things too much and requires NLP

Table 3: Organizations Table from Crunchbase

Feature engineering In the Organisations table:

country_code ⇒ Continent:

Contrary to the strategy by Xiang et al., (2012) and other research papers where it is stated that only US companies were selected because Crunchbase is an American website, I believe that nowadays it

is more than a reliable source for startups that operate outside the US. To make the transformation, I used the following kaggle dataset; https://www.kaggle.com/datasets/statchaitya/country-to-continent. This allowed me to go from the 3-letter country code to the continent. As far as I am concerned, this transformation to study startups by continent is one of its kind in the literature world, and I hope it will provide a breakthrough for investors looking to invest in companies outside the US.

facebook_url, linkedin_url, twitter_url ⇒ social_media_presence

Who would have thought that a bunch of links would ever be useful? In this case, instead of getting rid of them, we transform all of them into a binary variable that is equal to 1 if the company has any presence on social media, and 0 if it doesn't. According to the literature review, we saw what an important role the company's image plays in the view of the general public. Hence, having a social media presence will be crucial to grow the brand image across the world.

founded_on ⇒ company_age

This might seem like the most trivial of transformations, since we want to know how old the company is to be able to differentiate startups from matured companies. Indeed, we must not forget that Crunchbase provides data not only on startups but also on older companies since these started as startups at some point in time.

last_funding_on => years_since_last_funding

last_funding_on by itself is of no use to us since it is a date. Transforming it as today - lat_funding_on will give us the number of years since the startup last received funding from investors. This will allow us to see the level of dependence the company is currently operating at.

last_funding_on - founded_on ⇒ years_of_funding_dependence

These two columns are too valuable not to be exploited once more. Subtracting the founding date from the date the startup last received funding_at gives us the number of years the company was relying on funding rather than on its own resources. On face value, this seems to be like a solid variable to have in our final dataset.

employee_count ⇒ employee_count_cat

Since employee count consists of ranges of the number of employees, let's group them into bins where each bins corresponds to a specific range.

This is not the end of our feature engineering process, but it is the end of the new variables created. Later on, we will see that some numerical columns will have to be transformed into categorical purposes mainly for

sparsity purposes.

Moving on, let us now look at the people table in table 4. For this table, I will not include some variables we excluded before for the same reason in order to avoid unnecessary congestion. Mainly, these are variables such as created_at, updated_at, permalink, cb_url and logo_url.

Variable(s)	**Description**	**Selected?**	**Reason for inclusion / exclusion**
uuid	unique identifier of a person	Yes	This will be useful because it is linked to the degrees table (see ERD diagram above)
featured_job_organisation_uuid	Id of the organization in which the person works at	Yes	This serves as a foreign key for the organizations table
gender	Gender of the person	Yes	This might seem irrelevant in 2023, but it is extremely useful in my opinion to see if gender disparities in senior positions still persist nowadays
featured_job_title	Job title of person	Yes	This will be useful for us in order to get the most influential person at the company
featured_job_organization_name	Organization name	No	We do not need it as we already have the link with the organizations table
first_name last_name	First and last name of the person of interest	No	They don't bring any added value to the ML problem at stake
country_code state_code region city	Locations useful to identify where the person is located	No	For our study, we only care about where the company HQ is located.
Name	Name of the person	No	The uuid is enough to identify a person

Variable(s)	Description	Selected?	Reason for inclusion / exclusion
permalink cb_url	All these are useful for the company URL	No	These are of no use to us for the purpose of our work
facebook_url twitter_url linkedin_url	Social media links of person of interest	No	These are of no use to us for the purpose of our work

Table 4: People Table from Crunchbase

Other than the gender variable being immediately useful to us, this table serves more as an intermediary between the organizations table and the degrees table since these two are not directly linked. Before taking a look at the aforementioned table, let us see some transformations that the current People table has undertook:

Feature engineering In the People table:

uuid, featured_job_organizations_uuid ⇒ num_seniors:

This might seem like useless variables when taking them alone, but by grouping people of the same organization together, we can get the number of seniors working at the company. Indeed, Crunchbase holds data on people with some kind of influential role, which can be considered seniors or experts in their domain, and this is confirmed by the job titles described. This means that simply counting the number of employees in each organization from this table gives us the number of seniors.

num_seniors, gender ⇒ male_senior_ratio ⇒ all_male:

We immediately make use of the variable we created in the previous point and calculate the ratio of males that occupy senior positions in a company. As mentioned before, this should not be a measure to rely upon in 2023, but I used it to see if gender disparity is still an issue in our day and age. We would hopefully expect things to be balanced. Unfortunately, the vast majority of seniors were found to be males, thus I decided to have a more balanced variable by transforming it into a binary variable called “all_male”. In this binary variable, if all the seniors at that company are males, then it has a value of 1, else it has a value of 0. In other words, all_male checks if a company has a hierarchical structure of only males.

We proceed by shifting our interest to table 5 where the Degrees table is portrayed and the most relevant variables are selected.

Variable(s)	**Description**	**Selected?**	**Reason for inclusion / exclusion**
person_uuid	Id of person who has obtained a certain degree	Yes	This is the link between the degrees and the people table
uuid	Unique identifier of each degree	Yes	I will not use it in our final dataset, only to count the number of degrees each person of interest has
is_completed	Did the person complete the degree in question or not?	No	Since there is a lot of sparsity in this column, I consider that if a person took part in a degree, it is enough to be counted as having a degree. This is partly true because there is still some experience gained from a degree even if not completed
name	Degree name	No	This is too specific, making irrelevant for our ML problem
person_name	Person name	No	This is too specific, making irrelevant for our ML problem
institution_uuid institution_name	Id and name of institution where the person got the degree	No	This is of no use to us and does not align with the ultimate goal of our project
degree_type	Type of degree obtained by the person of interest	No	Although it might seem plausible for many that this variable should be selected, it has over 100 values, so I decided to rather be interested in the number of degrees.

Variable(s)	Description	Selected?	Reason for inclusion / exclusion
started_on completed_on	When did the person complete the degree	No	In our dataset, I found that in these variable 30% of values were missing on average, so we would have to sacrifice a lot of data

Table 5: Degrees Table from Crunchbase

The variables selected from the Degrees table will be particularly useful for the feature engineering part that I will present later on.

Feature engineering In the Degrees table:

person_uuid, uuid ⇒ num_degrees:

This transformation will allow us to get the number of degrees that each person in our data frame has. This will be useful to understand if the more degrees the founder has means the more chance of success for the company, or whether it is the complete opposite.

Lastly, figure 3.7 is a much more concise table with few variables that are present, but feature engineering will allow us to render these variables extremely valuable.

Variable(s)	Description	Selected?	Reason for inclusion / exclusion
org_uuid	Id of the organization the funding round is attributed to	Yes	This is the foreign key that links the organizations and funding_rounds tables
investor_count	Number of investors in that particular round	Yes	This will be useful to know the total number of investors in a specific organization
investment_type	Type of investment that has been made in a particular round	Yes	This will be a crucial variable in our feature engineering process that will allow us to separate at which funding stage a certain startup received some funding

Variable(s)	Description	Selected?	Reason for inclusion / exclusion
uuid	Unique identifier of the funding round	No	This will not be useful to count the number of funding rounds since we already have this information in the organizations table
name	Name of the funding round	No	This is too specific, making irrelevant for our ML problem
raised_amount_usd	Total amount of money raised in a specific funding round	No	We already have the total amount of funding a company received, so this would not be of great use to us

Table 6: funding_rounds Table from Crunchbase

Feature engineering In the funding_rounds table:

org_uuid, investor_count ⇒ investor_count:

To get the new variable we sum the number of investors that each organization got in total. So we sum the investors that took part in one of the company's investment rounds to get the total number of investors.

investment_type ⇒ pre_seed, seed, series_a, later_investment

From a single variable, we create four new variables, this is the beauty of data analysis! Each of the new variables describe certain funding stages in the company. The order of these phases in increasing chronological order is the one mentioned in the description. What I decided to do is to group everything after series_a and series_b as a single variable in order to avoid having a lot of missing values in our final dataset.

investment type ⇒ debt_financing

Another variable that I created from the investment type is the debt financing one which combines all possible sources of financing that come from debt, mainly debt financing, convertible notes and post ipo debt.

c) Variables and Measures

Now that we have all the pieces of the puzzle, we can take a look at what our merged table looks like below in table 7.

Allow me to do a quick summary of the variables that made the cut. You will notice that there is a large amount of binary features present in the final dataset. This was made for reasons explained in the various feature engineering parts, but generally it is because there is so much sparsity in the original dataset that it was not feasible to feed it into a ML model. Here are the variables that resulted from the merge of the Crunchbase tables, and which will be the starting point for our EDA and modeling:

- **status:** This will serve as our target variable. It is a binary categorical feature with values 0 and 1. 0 means that the company is exited (either acquired or out of business), while 1 means that the company is still in operation.
- **category_groups_list:** This is a list of the group or industry to which the company belongs. There are 44 unique values, so it is concise. This variable is categorical.
- **num_funding_rounds:** This is the number of funding rounds that a company undertakes. There are 24 unique values. This variable is numerical discrete.
- **total_funding_usd:** This is a standardized value for the funding received by a company. Although this may not be the best practice, we standardized this variable early on in order to be able to compare it to all the other variables which are already pretty much on the same scale. We will later standardize all variables. This variable is numerical and continuous.
- **num_seniors:** This feature represents the number of seniors that a company has overall. It is a discrete numerical variable since a person cannot be split in half. It consists of 30 unique values.
- **male_senior_ratio:** This is the ratio of men occupying senior positions in the startup. 1 means that this company is composed of an all-male hierarchy, 0 means that it is composed of an all-female hierarchy, and everything in between is possible. However, as mentioned earlier, this variable will be transformed to a binary feature called **"all_male"** because of the vast majority of companies having all-male hierarchies.
- **num_degrees:** This variable represents the number of degrees that the founder of a company is in possession of to see the impact of education on startup success. This is a discrete numerical feature with 10 unique values
- **early_investment:** This is a binary feature where it is 1 if the company has had an early investment, 0 otherwise. By early investment, we refer to angel investment, pre-seed investment or grant.
- **seed:** This is a binary feature where it is 1 if the company has a seed round, 0 otherwise.
- **any_investment_funding:** This is also a binary feature that takes value 1 if the company has taken part in any funding round (series a onwards). Otherwise, it takes value 0.

- **debt_or_other:** Same as the previous three defined variables, this is also a binary feature. It takes value 1 if the company has undertaken debt or any other form of funding that is not part of an investment round. Otherwise, it takes value 0.
- **investor_count:** Number of investors that have invested in the startup. This is a discrete numerical variable with 55 unique values.
- **employee_count_cat:** This variable divides the range of employees that work in a certain company in intervals of 50. The lowest value, 1, is in range 1-10; while the highest value, 9, consists of a company that has 10000+ employees. 0, on the other hand, corresponds to a startup with an unknown amount of employees.
- **social_media_presence:** This variable is a binary feature with value 1 if the company has a social media presence (Linkedin, Facebook and Twitter), and 0 if it does not have a social media presence, or only one or two of those three.
- **continent:** This feature is self-explanatory, as it indicates one of the six continents the startup's HQ is located in. It is a categorical variable with 6 unique values.
- **company_age:** It tells us how long ago the company was established. It could have been established up to 10 years from now. This feature is numerical discrete with 11 unique values.
- **years_since_last_funding:** Indicates how long ago did the company receive any kind of external funding that has been accounted for in the data. It is numerical discrete with 11 unique values.
- **years_of_funding_dependence:** This is the difference between the last year the company received external funding and the year in which the company was founded. It is a numerical and discrete variable having 11 unique values.

All the variables have 61860 non-null values, except for category_groups_list which has 1355 null values. The difference between the number of null values and the overall size of the dataset is too small and insignificant, so I have opted to remove the observations with the null values. However, it must be noted that 60505 is not the final number of observations, since we did not deal with outliers yet, which we will study in the next section.

A key decision I have decided to include startups that have been in operation for more than 10 years since Jan 2022. The choice of the lower bound date is because even though there is no specific definition of what constitutes a startup, the general consensus is that startups have a lifetime of 5 years before exiting the startup phase. What is more, exiting the startup phase successfully does not mean that the startup will go on to be successful, because in the real world it has much less of the benefits it had in its startup phase, hence why we decided to include 5 extra years to see if the company made it through even after exiting the startup phase.

Column	Non-null count	Unique values
status	61860	2
category_groups_list	60505	44
num_funding_rounds	61860	24
total_funding_usd	61860	24126
num_seniors	61860	30
male_senior_ratio	61860	61
num_degrees	61860	10
seed	61860	2
early_investment	61860	2
any_investment_funding	61860	2
debt_or_other	61860	2
investor_count	61860	55
employee_count_cat	61860	10
social_media_presence	61860	2
continent	61860	6
company_age	61860	11
years_since_last_funding	61860	11
years_of_funding_dependence	61860	11

Table 7: Merged table with featured engineered variables

i) Checking for data inconsistencies

Working with data can be extremely tricky if not tackled properly. One of the most important things to check for when starting any project in ML is that we limit data inconsistencies as much as possible. By data inconsistencies, we mainly refer to duplicates, missing values and outliers. Of course, data

inconsistencies can take other forms such as typos, formatting errors, inconsistent units of measures, and many more. However, these are of no interest to us, since they are not causing any issue as far as the scope of the study is concerned. Next, I will briefly explain each of the main inconsistencies that we may have in our dataset and the approach we took to resolve them.

Outliers

To make it simple, outliers refer to data points that deviate largely from the mean. Biasing the results and reducing the predictive model's accuracy are just some of the problems that are incurred by leaving outliers in the dataset. There are several models developed by the literature to exclude outliers, but the one that I chose is known as the Tukey method. The aforementioned method sets lower and upper bounds to 1.5 times the interquartile range (IQR) below the first quartile (Q1) and above the third quartile (Q3), respectively. Any data points that are located outside that range will be excluded from our final dataset. It must be noted that there are a multitude of ways to deal with outliers that don't consist in removing them from the dataset. Indeed, some of these techniques include modifying the data or utilizing imputed values like imputation by mean or median. The problem with these types of methods is that they introduce bias that we do not necessarily need to incur in our project, given the huge amount of observations that are in our possession. Therefore, eliminating blatant outliers was considered to be the best strategy in this circumstance.

Duplicates

Having duplicate values in our dataset can lead to biased prediction and skewed results, so it is of the utmost importance to check for them and make sure that they are dealt with properly. A common misconception is that all duplicates are bad and that they should be removed, but this should not be the case. Indeed, sometimes duplicates are present because it simply makes sense for many observations to have the value. For example, say company A and company B both have 4 seniors in their company: does this mean that we should remove one of them? Absolutely not, it just makes sense for things to be that way. However, duplicates are a problem when the feature is considered to be a primary key for example. In our case, the company id is the primary key, so it would be flawed to have more than a single observation with the same company id. This is the only column that should be thoroughly checked for this kind of inconsistency, as the others could easily overlap and lead to many observations having the same variable value.

Missing values

Lastly, the most important data inconsistency in my opinion is the missing values. Effectively, there are such a wide range of ways to deal with this kind of issue that it could lead to a wide variety of possible

outcomes later on. Some of the decisions we can take with missing values is to remove them entirely, but we would be losing valuable information depending on the percentage of the total observations we are removing. Other ways to deal with missing values are similar to what we saw with outliers: we could very much impute by mean or median, or by other measures in order to avoid any loss of information. However, this would not eclipse the fact that with this method comes some bias.

In my project, the percentage of missing values compared to the whole set of observations present was insignificant, so I decided to go ahead and remove these observations.

ii) Addressing Class imbalance in Target Variable

As we alluded to earlier, a huge issue with our dataset is the large class imbalance of the target variable "status". In this subsection, we first look at the extent of this issue, before understanding why this could be a problem, and lastly we will understand the solutions we can take to solve it. Among these possible solutions, I will select one to proceed with.

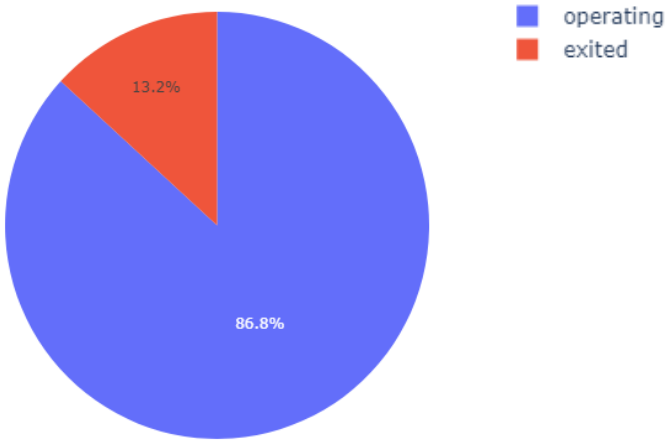

Figure 3.2: Pie Chart of Target Variable "status"

Looking at the pie chart in figure 3.2 above, we notice a huge class imbalance in the target variable "status", where 86.8% of observations are operating and 13.2% are exited. The issue with this class imbalance is that if it goes unnoticed, it can lead to a lot of invalid conclusions, such as having extremely

high accuracy just by predicting all the companies to be operating. Indeed, what happens is that in this case there will be an over-representation of the majority class in the model's prediction.

However, I must note that some models like decision trees and random forests are robust to class imbalances. The aforementioned models are in possession of mechanisms like weighting the classes or using ensemble techniques, allowing them to not be affected by this imbalance. But by relying on such a few sets of models, we would be missing out on the potential of other algorithms. This is why we look at immediate solutions to resolve the issue of class imbalance.

Upsampling, which includes creating more samples from the minority class in order to balance the dataset, is one possible solution to this problem. Another strategy is downsampling, which involves randomly eliminating samples from the dominant class in order to balance the dataset. Downsampling, on the other hand, may not be the best option in our case due to the possibility of data loss. Upsampling, on the other hand, allows us to keep all of the original dataset's information while balancing the classes.

The Synthetic Minority Over-sampling Technique (SMOTE) is the most commonly used method of upsampling. SMOTE is a method for generating new minority class samples by interpolating between existing samples. SMOTE, in particular, takes a random sample from the minority class and determines its k nearest neighbors. Then, by interpolating between the original sample and its k nearest neighbors, it generates new samples. SMOTE generates a synthetic dataset with an equal balance of classes by repeating this technique for each minority class example.example.

SMOTE outperforms traditional upsampling techniques in a variety of ways. To begin, it can produce synthetic samples that are more typical of the minority class, boosting the generalization of the model. Second, because it gives new samples rather than duplicates of existing data, it can reduce overfitting. It does, however, have certain restrictions. For example, if the dataset only contains a few minority class examples, SMOTE may generate samples that are quite similar to one another, limiting the variety of the dataset. Furthermore, SMOTE can introduce noise into the dataset if the nearest neighbors are not representative of the underlying data distribution.

We chose SMOTE to upsample the minority class in our example since it allows us to keep all of the information from the original dataset while balancing the classes. We will run multiple models on the original dataset to see how resistant they are to class imbalances. This is due to the fact that some models, such as tree-based models, are less sensitive to class imbalances and can operate without upsampling.

It is crucial to note that the SMOTE approach should only be used on the training set, not the validation or test sets, which would cause data leakage issues. This is due to the increased danger of overfitting when using SMOTE on validation or test sets. Overfitting occurs when the model learns from synthetic examples provided by SMOTE, resulting in a biased assessment of the model's performance. When

SMOTE is used simply on the training set, the model is able to learn from synthetic samples and generalize to fresh data effectively. To ensure an unbiased evaluation of the model's performance on the validation and test sets, I will divide the dataset into training, validation, and test sets and use SMOTE primarily on the training set.

iii) Correlation and techniques for feature selection

It is critical to select the most important features from the dataset in order to develop an effective predictive model. I employed correlation analysis, a popular technique for variable selection, to accomplish this. I would like to mention that an association matrix could also come into play for categorical variables, but since we have only 2 in our dataset, we have encoded them and included them in the correlation plot. A correlation plot allows us to see which attributes are most related to the target variable and python allows us to visualize these relationships in an intuitive and interpretable way through the seaborn library. What this correlation matrix will allow us to do is to understand two crucial things: the first is the relationship between the independent features and the target variable; while the second is the relationship between the independent variables themselves. This way, on one hand we can hypothesize which variables are presumed to have the largest impact on the status of a company; and on the other hand we can understand if multicollinearity exists between the independent variables and to what extent. By multicollinearity, it is intended that independent variables happen to be highly correlated with each other. Multicollinearity could be an issue where it becomes difficult to separate between the effect of independent variables on the target feature. This is why, it is not enough to only have a correlation plot, and it should be complemented by another technique. This is where studying a reduced logistic model comes into play.

A reduced logistic regression model will be composed of the target variable and of two independent variables that are deemed to have high multicollinearity. We will include an interaction term in this model in order not only to see how strongly each of the independent features impact the target feature, but also to see how reliant one’s effect is on the other. Although this technique is not the fastest and there are faster ones out there like PCA, stepwise selection and LASSO, it is definitely more interpretable than those just mentioned. Since this is a thesis where I am explaining my empirical research, interpretability is an essential factor that comes into play, thus making reduced logistic regression a very attractive approach to deal with issues such as multicollinearity.

Once we are satisfied with the independent features, whether it is the strength of their relationship

with the target variable, or the weakness of their relationship with the other independent variables, we can proceed to the modeling phase. It must be noted that some models are robust to multicollinearity and that all of this would be dealt with automatically by them. However, it does not hurt to emphasize on the importance of these techniques for variable selection in order to have a complete picture of how a ML problem should be looked at.

d) Predictive Modeling techniques

The problem at stake is a predictive modeling problem. According to Gartner's definition, predictive modeling is a commonly used statistical technique to predict future behavior. In our case, we want to predict whether future companies will be successful or not, in order to help the investors make more informed decisions and reduce losses.

i) Cross-validation and train-validation-test split:

For predictive modeling to work, it is necessary to evaluate the model's performance on previously unseen data so that we can guarantee generalizability. For this reason, we have opted for a train-test split, which separates the dataset into three parts: a training set, a validation set and a testing set. The training set will be used to train the model, the validation set will be used to tune the hyperparameters, and lastly the test set will be used to evaluate the model's performance on unseen data. However, literature argues that even though the train-validation-test sets split has proven to be successful, it still relies on a random selection of data points. To eliminate this element of randomness that creeps in, we use cross-validation (CV). CV, and we will particularly use k-fold CV, consists in splitting the data into k subsets, or folds. Each fold will serve as a validation set at some point, while the remaining k-1 folds are used for training. In this way, we are able to assess the performance of the model and draw conclusions in a more reliable way, without relying on randomness as a factor. What is more, this allows us to reduce the risk of overfitting.

Both of these techniques will be useful in our work. First, we will use the train-validation-test split in order to evaluate the VANILLA models and optimize the best model by finding the most optimal hyperparameters through the validation set. We will then evaluate the results on the test set in order to avoid any sort of bias that the tuning could have led to on the existing split. Second, we use cross validation on this best model in order to get a more generalizable result which is more reliable and which does not depend on an element of randomness in its assessment.

ii) Vanilla models for classification

As previously determined, we are in presence of an imbalanced target feature. I intend to apply the SMOTE technique that we already explained, but I also explained my willingness to keep the original dataset. Thus, we need robust models that can deal with class imbalances without prioritizing one class over the other. In this subsection, we will see which models will be optimal for our case study.

Finally, now that we are satisfied with our dataset and that we have set the scene from what to expect, we may proceed to fit some models.

Since we are dealing with a classification problem, we must employ models that deal with classification. By classification problem, we are referring to a ML task that attempts to predict the class or category of an observation based on a set of input variables or features. In this scenario, we would like to anticipate a company's performance based on factors like funding, location, and industry. Classification is a type of supervised learning, in that the output categories are predefined. Now that I have set the context of the type of problem we are dealing with, we can engage in the intricacies of the modeling phase.

My thought process for the modeling phase is to go through a large variety of classification models in their VANILLA form, then optimize those with the best performance according to the validation set. By VANILLA models, we are referring to algorithms in their most basic state without any additional regularization or feature engineering. This approach has the benefit of being both efficient and informative, giving us a starting point to work from when we do start optimizing our models later on. Whereas by best performance, I intend that I will be using a variety of classification metrics such as Accuracy, Precision, Recall and F1 score and see among which models these are the highest. In addition to these metrics, I will use an AUC curve to ensure that we have the best model. The details of each of these metrics will be discussed in the next section.

As models, to start, I would like to compare plain logistic regression to a technique belonging to each of a linear model (AIC or BIC), a penalized approach (Lasso, ridge or elastic net), and lastly a non-linear model (one of KNN, splines, gam, tree-based algorithms). Additionally, I implement Kernel SVM at the end. This helps draw a complete picture of the best possible model tailored for our problem.

By using a linear model, we are able to quantify the relative importance of each independent variable compared to the target one. Particularly, the model that I have opted to choose in this project is the AIC stepwise selection applied to the logistic regression model since it will help us obtain the most relevant features tailored to our model. I chose it over BIC because it is less strict, allowing for a more favorable balance between model complexity and goodness of fit.

A penalized approach, on the other hand, will help us prevent overfitting in high-dimensional datasets. We do not expect too much penalization in our case because we do not have too many variables. Indeed, we have 18 variables in total, one of them being the target feature. The penalized approach that I favored is Lasso, since it not only reduces the importance of less relevant variables, but it also performs variable selection, hence removing irrelevant features completely.

In simple words, with non-linear models we are able to get good predictive performance even when the relationship between the input and output variables is non-linear. Indeed, as opposed to linear models where a linear relationship is assumed between these sets of variables, it is not the case for non-linear models. This leads to non-linear approaches being able to detect more complex interactions and patterns in the data that would otherwise go unnoticed. Among these models, I will use tree-based approaches like

decision trees and random forests. The former corresponds to a binary tree structure; whereas the latter is an ensemble method that combines multiple decision trees, which allows it to reduce overfitting and increases the model's stability. I will also employ KNN which is another non-linear model that functions by predicting the class of a new data point according to the k closest data points according to some distance metric (we will use euclidean). An advantage of KNN over tree-based methods is that it does not assume a decision boundary, hence it is able to capture more complex patterns when data points are difficult to separate.

I used random forest as a non-linear model, and later on I used KNN and Decision Tree as well). I also implemented Kernel SVM at the end. Logistic regression is a well-known classification technique that can deal with both binary and multi-class situations. It is a straightforward yet successful method that can be quickly taught on large datasets. Kernel SVM, on the other hand, is a non-linear model that maps input data into a high-dimensional space using kernel functions. This is handy for locating data points that cannot be separated linearly. The KNN method is a lazy learning technique that classifies new instances based on their similarity to neighbors in the training set.

iii) Optimizing models: grid search, random search

Having vanilla models is great to compare baseline models and eliminate the number of models being studied right from the start. This allows us to save computation power and be more efficient overall. However, this should only serve as a starting point and we should seek to provide a model that is tuned for the particular problem that we are studying. This is done through optimizing the algorithms that made the cut.

When it comes to optimizing models, there are a few extensively utilized strategies. Grid search and random search are two of the most common strategies. Grid search is a strategy that includes trying out all potential hyperparameter combinations within a specific range in order to find the best combination. This method is time-consuming, but it ensures that the ideal combination is found. Random search, on the other hand, samples hyperparameters at random within a set range and is therefore faster than grid search. The disadvantage of random search is that it does not guarantee the discovery of the optimal combination. Finally, the choice between grid search and random search will be determined by the individual problem and project restrictions.

In an ideal world, only grid search would be used to test all plausible hyperparameter combinations, and this would make sure we get the best possible model. However, since i have limited computational resources, I decided to proceed in 2 steps: first, I perform a randomized search among some parameters and

include those which were in the original random forest; then I perform grid search by including the values of the hyperparameters that I got from the random search random forest model, and some neighboring values. This guarantees that I get the best possible model considering the limited resources that are at my disposition. We then compare the best model with the original random forest to see if there has been much improvement, or whether the vanilla model performs in a similar fashion.

e) Evaluation metrics

i) Confusion Matrix

Confusion Matrix will be extremely useful for us to see that models that seem to have a high accuracy like logistic regression portray only a part of the image, while the bigger image is that this high accuracy is a result of the class imbalance that we have mentioned time and time again throughout this thesis. In other words, the aforementioned evaluation metric is elevated because the model mostly predicts positive, or operating, status. Since the vast majority of the values relating to the target variable are positive, we will get a very good accuracy, even though we are missing out on most, if not all, the negative, or exited, companies. Hence, our model would be worse than guessing the outcome in a sense.

ii) Accuracy, Precision, Recall, F1 Score

Let us go more into detail into each of these metrics to understand why they were particularly chosen. Starting with accuracy, it is a popular metric that simply measures the proportion of events that are accurately identified. While accuracy is critical in many cases, it can be deceptive when dealing with imbalanced datasets like it is the case in our problem. In such cases, precision and recall may be more helpful metrics. Precision is the fraction of correct positive forecasts among all correct positive predictions, whereas recall is the proportion of true positive cases among all correct positive predictions. A high precision score indicates that the model effectively avoids false positives, and a high recall value indicates that the model effectively detects all positive situations. Lastly, the F1 score is a balanced evaluation of the model's performance because it is a harmonic mean of precision and recall.

iii) Area under the curve (AUC)

AUC, or Area Under the Curve, is another often employed metric for evaluating binary classification models. The accuracy of the model in identifying positive and negative classes is assessed by computing the area under the Receiver Operating Characteristic (ROC) curve. The ROC curve is produced by plotting the true positive rate versus the false positive rate at various threshold levels. If a model has a higher AUC score, it is assumed that it will be able to predict the true class labels more accurately. AUC is a useful metric to assess a model's overall performance, especially when the dataset is unbalanced like it is the case for our problem. It is also useful in order to consider several threshold values and get the best results according to what we are searching for. However, even after stating the importance of AUC, we will not use it in our thesis, since the metrics already in place are more than enough to cover this problem and adding this metric would only generate further confusion. Indeed, the confusion matrix will tell us whether the partition between positive and negative labels is satisfactory in a more visual way, and the other set of metrics will provide us with the details we mentioned in the previous paragraph.

04. Results and Analysis

a) Descriptive Statistics and Data Visualizations: Univariate Analysis

This section will be split into 2 parts, one in which we analyze the data containing some outliers, and one where we analyze the data after having removed these outliers.

i) Numerical variables analysis: pre/post outliers

Before removing outliers

We have to separate numeric from categorical columns and analyze each of them to get a sense of what our data looks like. Once we have done that, we have to differentiate discrete from continuous variables inside the numeric variables themselves.

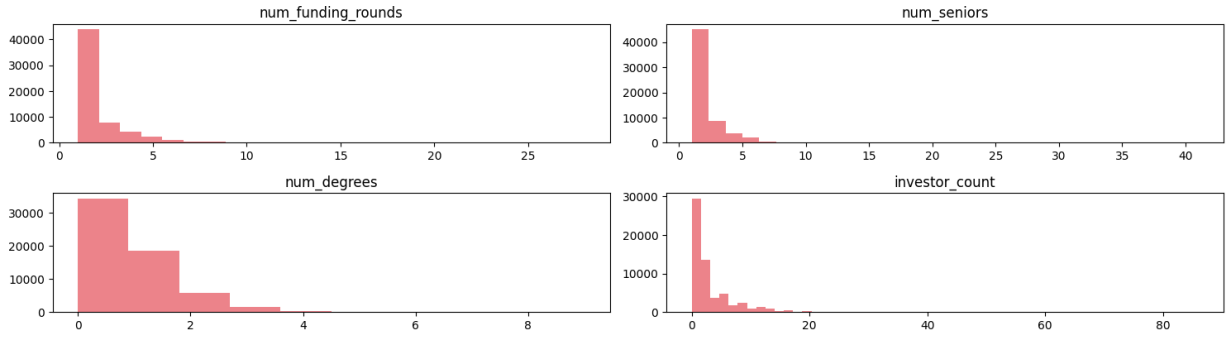

Figure 4.1: Histograms of continuous numerical columns

By taking a look at the histograms of the continuous numerical columns displayed in figure 4.1, we notice left skewness but most importantly we notice that the scale is too big and extends over far away values. Let us see if this is the case for the discrete variables as well.

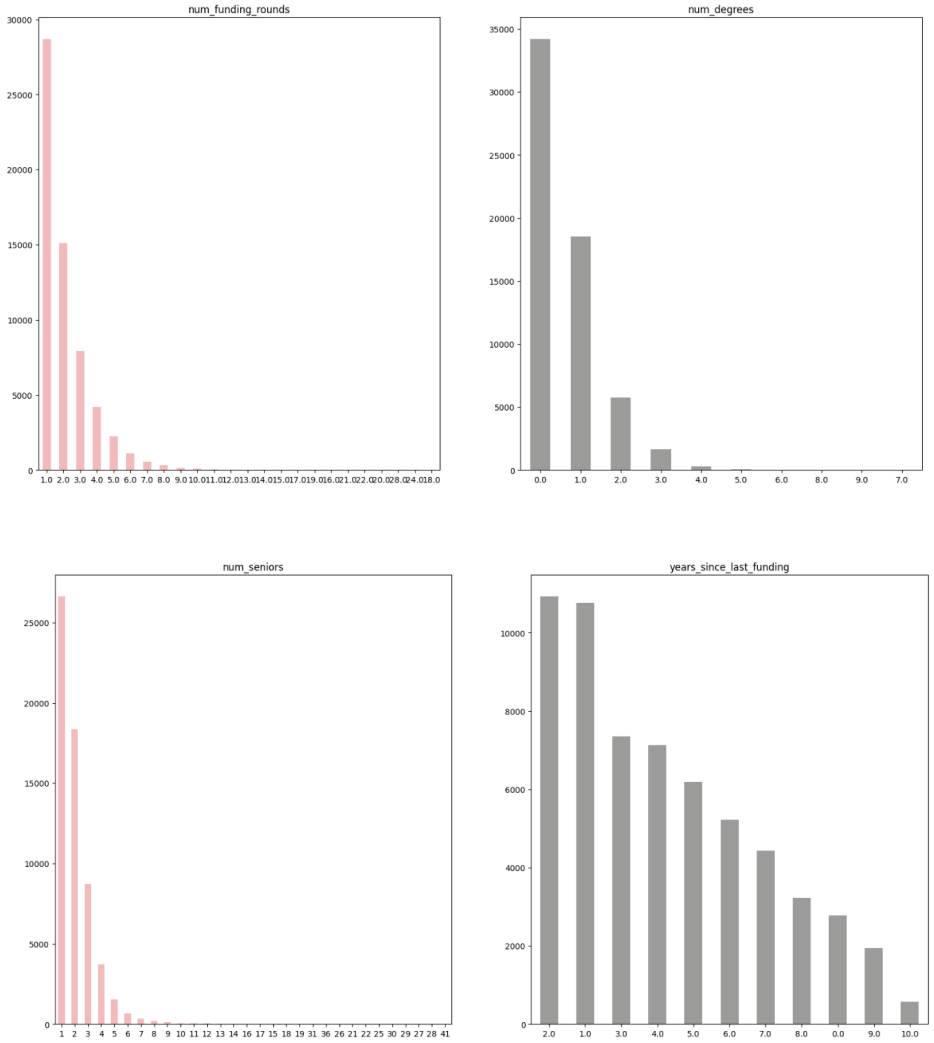

Figure 4.2: Bar plots of discrete numerical columns

We take a look at the box plots of the discrete features in figure 4.2. Again, we notice the same for the discrete variables. This hypothesis of outliers being present is confirmed by the boxplots of numerical columns in figure 4.3 where we see that there are some very highly skewed variables and others that have far values from the quantiles. For the purpose of my work and for completeness issues, I decided to get rid of these outliers, even though some algorithms are robust to this type of inconsistency. There are several ways to get rid of outliers, but we will choose the interquartile range (IQR) method. This is what we get by removing the outliers.

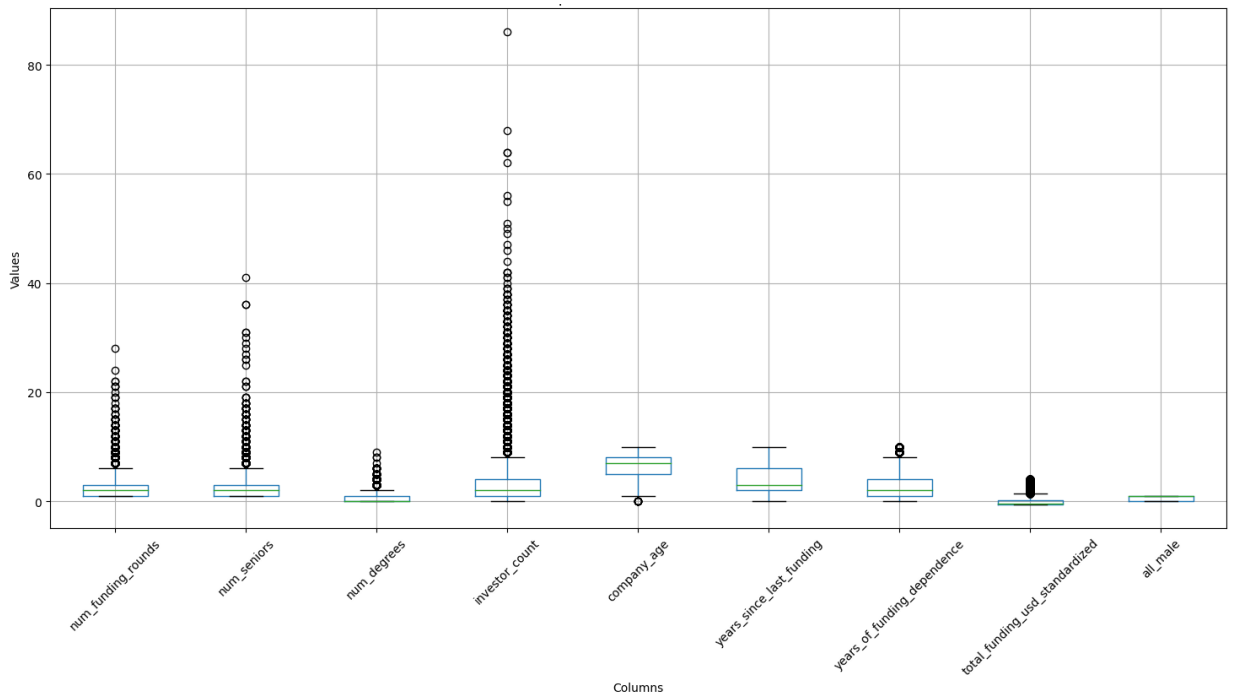

Figure 4.3: Boxplot of numerical columns pre-outlier-removal

After removing outliers

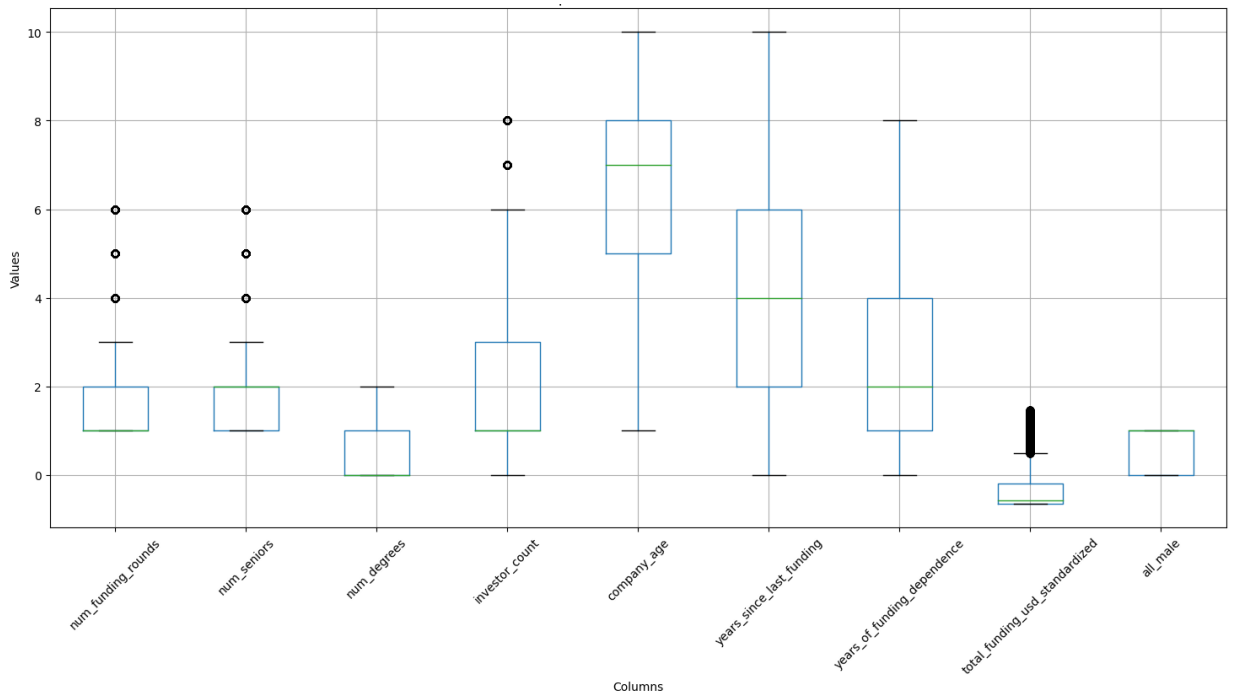

Figure 4.4: Boxplots of numerical columns post-outlier-removal

Let's analyze the cleaned boxplots in figure 4.4 in order to get insights on each numerical column and better

understand what we might expect from them:

- At first glance, num_funding_rounds and num_seniors seem to be the same variable, but far from it. They have similar characteristics in that their Q1 and Q3 are 1 and 2, and they have the same min and max.
- The median value of the variable 'num_funding_rounds' is one, indicating that half of the companies have only received one funding round. The average is now 1.77, indicating that some companies with many rounds of funding are driving up the average. The first quartile (Q1) is also one, indicating that the great majority of companies have received only one round of funding. The number of 2 in the third quartile (Q3) indicates that one-quarter of firms have received two or more rounds of funding. The variance is 1.17 and the standard deviation is 1.08, indicating that the financing round distribution is not particularly varied.
- With a median of 2 and a mean of 1.86, the variable num_seniors has a skewed distribution toward lower values. Q1 and Q3 are one and two, respectively. This indicates that most organizations have one or two senior individuals, which is astonishingly low and this may impact the company's capacity to make strategic decisions and attract investments. Furthermore, the variance is 1.13 and the standard deviation is 1.06. This suggests that the distribution is not skewed.
- The variable num_degrees has a median of 0 and a mean of 0.50. This shows that founders of startups in our dataset mostly have zero degrees. According to the interquartile range (Q1=0, Q3=1), 75% of startup owners have zero or one degree team members. The low variance of 0.44 and standard deviation of 0.66 for this variable imply that its values are close to the mean.
- The variable "investor_count" has a median of 1 and a mean of 1.92. This proves that the majority of businesses have only one investor, although some have several ones. This interquartile range (IQR) (Q1 = 1, Q3 = 3) is broad, which shows the great variation in the number of investors. The significant variance of 3.87 and the standard deviation of 1.97 corroborate this claim. In the context of the study, this variable may be an essential predictor of a startup's success because having more investors may indicate more financial stability and the chance for expansion.
- The variable "company_age" has a median of 7 years and a mean of 6.4 years. I mentioned that during the selection process, I chose startups that were founded 10 years ago. These results indicate that the vast majority of enterprises in the dataset are still in their early stages. The 5 to 8 year IQR indicates that the majority of startups are in this range. With a variance of 5.56 and a standard deviation of 2.36, the age distribution of the companies in the sample is reasonably distributed. Thus, there are some substantially older or younger companies than the mean.
- The column "years_since_last_funding" has an average of 4.03 and a median of 4.0, with a Q1 of 2.0 and a Q3 of 6.0. This implies that the majority of companies obtained investment within the last six years, with a four-year average between fundraising rounds. The variance is 6.22 and the standard deviation is 2.49, indicating that the duration between investment rounds varies significantly across

the companies in the sample.

- years_of_funding_dependence has a median of 2.0, a mean of 2.56, a first quartile of 1.0, and a third quartile of 4.0. 3.58 times the standard deviation is the variance. According to these estimations, the majority of the businesses in the dataset rely on capital for less than four years on average, with a dependency rate of 2.56 years. The years of funding dependence across the enterprises in the dataset vary substantially due to the relatively high variance and standard deviation.
- Total_funding_usd_standardized has a -0.56 median and a -0.33 mean, with a first quartile of -0.65 and a third quartile of -0.19. The variance is 0.23 and the standard deviation is 0.47. According to these numbers, the majority of the companies in the sample received less investment than the median of the funding distribution, with an average of less finance than the median of the funding distribution. Negative statistics indicate that money has been adjusted using the mean. This results in funding being less than the mean. The much lower variance and standard deviation imply that the standardized investment levels in the sample are less variable among enterprises.

ii) Categorical variables analysis

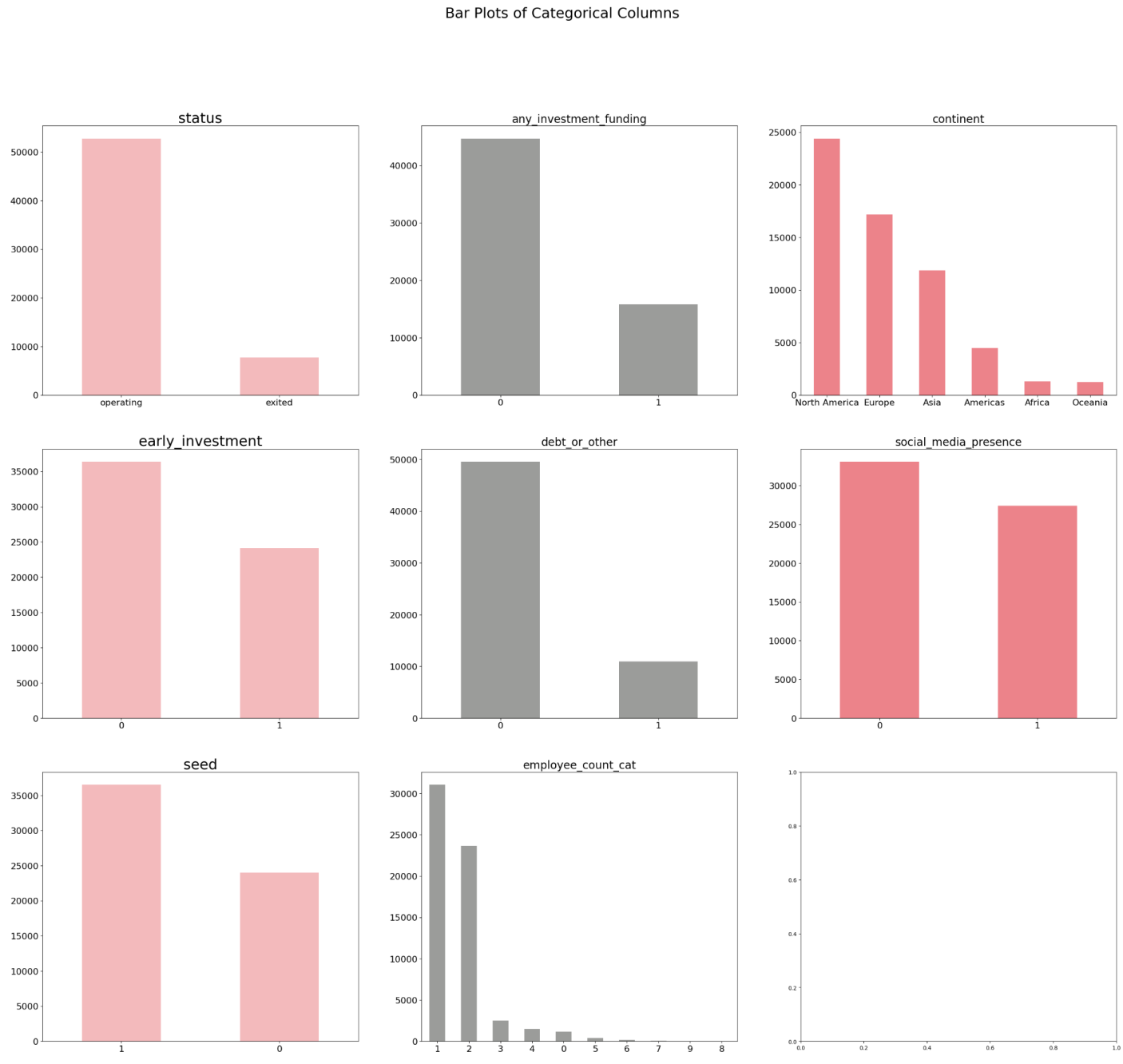

Figure 4.5: Bar plots of categorical columns

For this subsection, we will try to get insights from the categorical variables present in our dataset. Figure 4.5 shows bar plots of these features. We take a closer look at some of these variables to understand how they could be useful for our thesis:

- Immediately, we notice that there is a big class imbalance in the target variable "status", but this is worth studying separately later on
- The same can be said for the following two variables: any_investment_funding and debt_or_other.

This might lead us to exclude these variables from the mode, but I do not believe that the difference is significant enough for these features to be considered irrelevant. What is more, we will use some algorithms like random forests and decision trees that are resistant to class imbalance. Therefore, I have opted to keep them. Still, I will perform an association test to see whether they are strongly associated with the target variable or not. Regarding the variables per se, we see that most companies in our dataset do not undergo any kind of investment funding round; and most do not incur any form of debt

- Regarding the continent, it does not surprise us that North America is the continent with the most observations, since as mentioned previously, Crunchbase is an american-based company. However, Europe and Asia are still largely present in our dataset. Whereas the Americas (all the American continent excluding the USA) and Oceania naturally have smaller observations because of their small sizes. Africa is another continent with a low amount of observations, probably because of the few amount of startups that emanate from this continent, or because of the limited amount of data that Crunchbase is able to collect in this region.
- Early_investment and the seed round are composed of approximately the same proportion between the startups that apply for an early investment (pre-seed, grant or angel investment) or for a seed round and those that do not apply for any of both. Keeping in mind what I mentioned earlier regarding more firms than not taking part in any investment round, this suggests that most of them seek to get their funds early on and become financially independent as soon as they can. Still, more companies than not do not participate in any round of investment.
- Another aspect that is worth taking a look at is that the amount of startups present on social media is approximately equal to that of startups that are not present on social media. Keep in mind that we took a strict approach by assuming that a company is on social media if it has Facebook, twitter and linkedin.
- The most dominant range of employees in our dataset is between 1 and 10 employees, so small companies, and at a close second come those with an employee range of 11 to 50 employees.
- Lastly, the top 3 startup sectors are commerce and shopping (where we saw e-commerce as an important application), artificial intelligence and Financial services (largely due to the Fintech revolution that is currently taking place). I did not include this categorical column in my graph because it contains too many categories and things get messy.

Also an important overall observation is that these variables are mostly on the same scale, except for total funding which was already dealt with for convenience purposes. Thus, there is no need for standardization to be applied in our problem.

b) Descriptive Statistics and Data Visualizations: Bivariate Analysis

This section is extremely important to get an initial grasp of how different variables present in our dataset affect the target variable. We will take a look and comment on some of the most notable features that are worth mentioning. Later on, we will also interpret a smaller model of a logistic regression in order to better analyze some variables that we find to have an intriguing relationship with the "status" feature.

Here we will take a look at the boxplots of numerical variables against the status column in the original dataset. Even though there is a class imbalance, the boxplots still yield the same results after using the SMOTE technique, so it does not make a real difference in that regard.

i) Numerical variables vs Target variable

In this subsection, we compare the impact of different numerical variables on the target variable at stake. This allows us to understand the distribution of the status of our companies depending on different factors such as the number of funding rounds, the number of seniors, the number of investors, and many more variables.

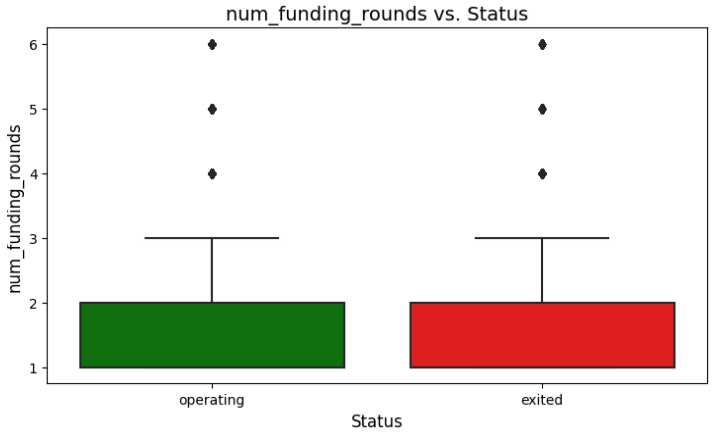

Figure 4.6.1: Boxplot analysis: number of funding rounds vs Status

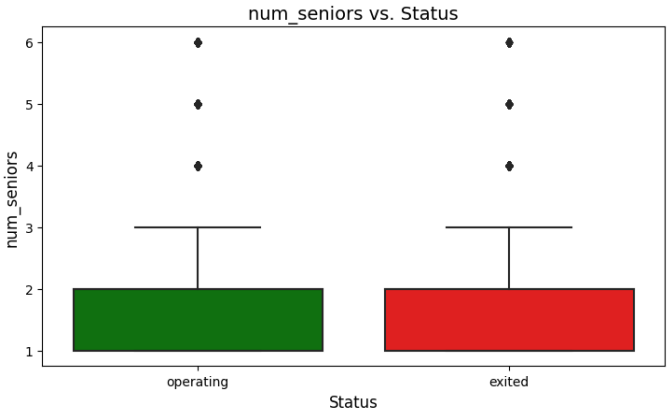

Figure 4.6.2: Boxplot analysis: number of seniors vs Status

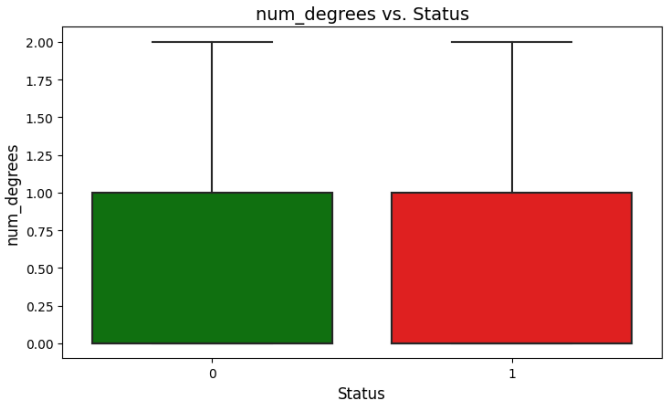

Figure 4.6.3: Boxplot analysis: number of degrees vs Status

We first take a look at how the number of funding rounds, seniors and degrees differs across companies that have different status. These are accounted for in figure 4.6.1, figure 4.6.2, and figure 4.6.3, respectively. Even though the number of funding rounds has an extremely close distribution between operating and exited companies, there may be some underlying differences in the dataset that are not portrayed in these visualizations. Thus, it is crucial to do some additional statistical analyses to determine whether that is the case and we should remove it. What is more, its interaction with other variables could affect the outcome variable.

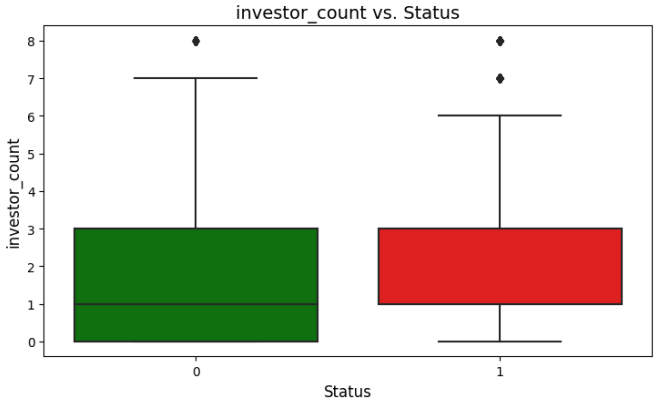

Figure 4.6.4: Boxplot analysis: investor count vs Status

With investor_count the scenario is a bit different as can be seen in figure 4.6.4, where the median is still the same, but the mean for operating companies is slightly higher. This means that operating startups have higher amounts of funding rounds on average. This is due in large part to the fact that at most 25% of firms that have exited have not had any funding rounds, whereas at most 25% of operating companies have had a funding round. This tells us that even though firms prefer not to rely on funding, data shows that at least a funding round is necessary.

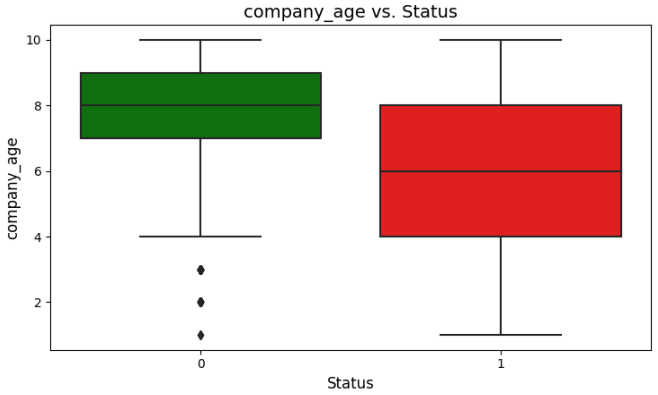

Figure 4.6.5: Boxplot analysis: company age vs Status

The contrast between operating and exited startups when considering company age is much clearer, as can be seen in figure 4.6.5: at least half of companies that have exited have an age of 8 years, whereas at least half of companies that are operating have an age of 6 years. This shows that companies do not necessarily fail within their startup phase (first 5 years), hence why we took this approach to go until 10 years.

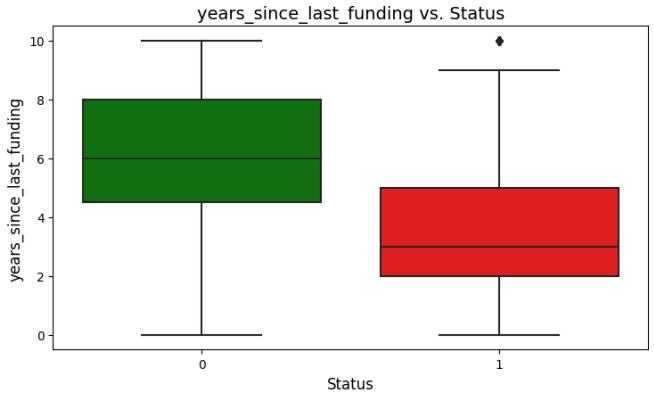

Figure 4.6.6: Boxplot analysis: years since last funding vs Status

Proceeding in the same way, figure 4.6.6 exhibits the relationship between the years since a company was last funded and its status. At least half of the companies that have exited have received their last funding 6 years ago, whereas at least half of those operating have received it 3 years ago. This may be due to the fact that exited companies tend to be older than operating ones, as we have seen in the previous graph related to company age.

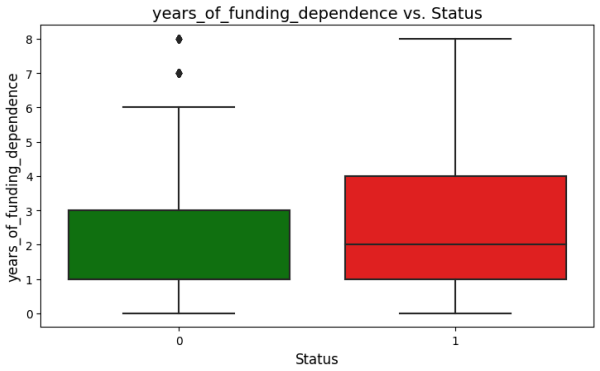

Figure 4.6.7: Boxplot analysis: years of funding dependence vs Status

There is not much to be added regarding funding dependence, as can be illustrated in figure 4.6.7, if not that operating companies tend to be more dependent on funding compared to exited companies. This may seem counterintuitive at first, but we have seen that some dependence at the beginning can be a positive thing.

ii) Categorical variables vs target variable

In order to compare categorical features with the target variable, we seek to compare the proportion of exited

and operating companies between each of the values that the categorical variable takes. In other words, we try to understand in which category exited startups are more present, while understanding those in which they are less present, and the same goes for operating firms. This is useful for us so that we can understand what to expect from the data we have at stake. Let's start by drawing the box plots, and analyze each one separately:

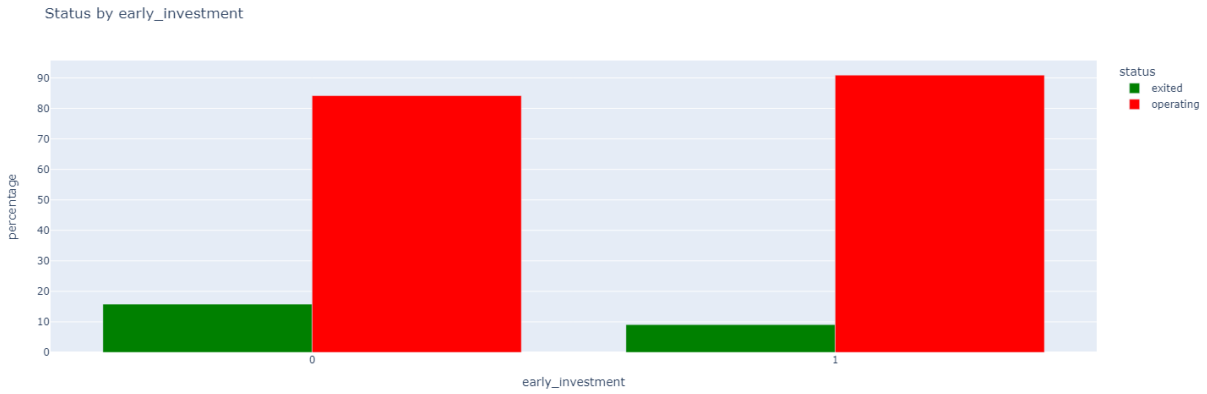

Figure 4.7.1: Barplot bivariate analysis: early investment vs Status

From the bar charts exhibited in figure 4.7.1, we see that exited companies tend to not incur an early investment, whereas operating companies do. It can be hypothesized that a reason for startup success might be taking on early investments.

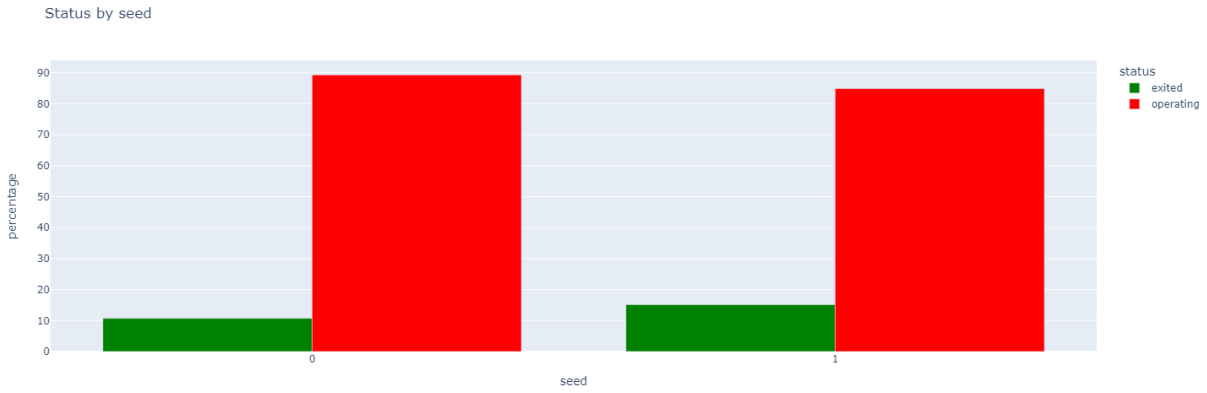

Figure 4.7.2: Barplot bivariate analysis: seed vs Status

The opposite trend is seen for seed investments, as illustrated in figure 4.7.2: exited companies tend to have seed funding rounds, whereas operating companies are split, with the majority of them not having one. This suggests that companies should take early investments and then become more reluctant to take on external

funding.

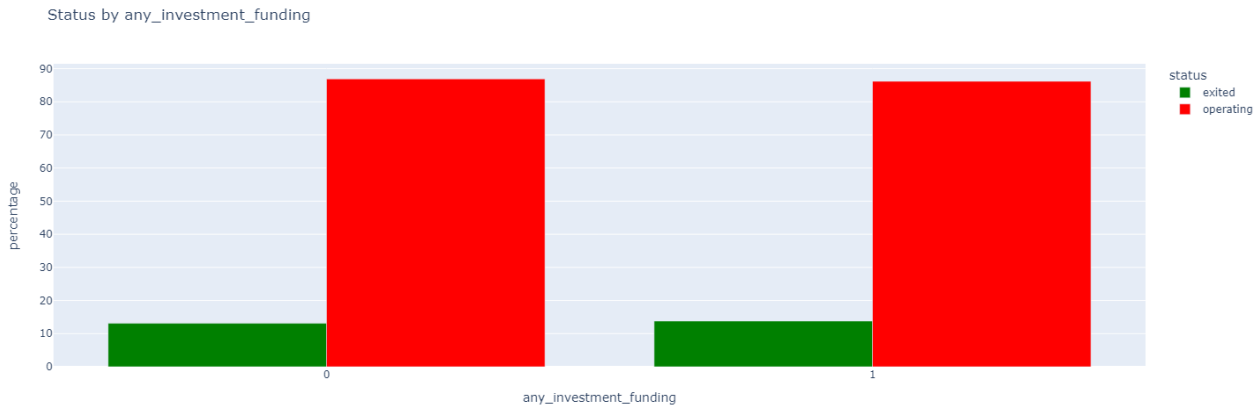

Figure 4.7.3: Barplot bivariate analysis: any investment funding vs Status

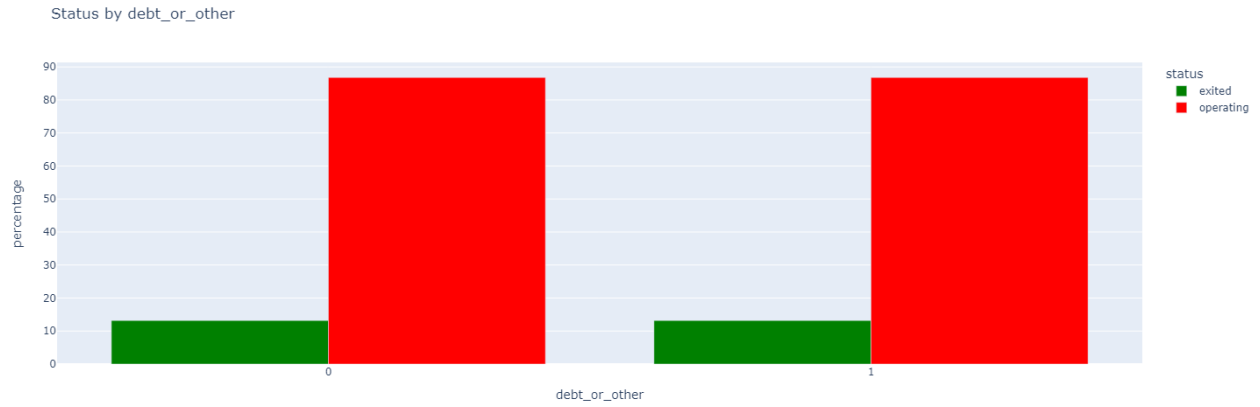

Figure 4.7.4: Barplot bivariate analysis: debt or other vs Status

In both the cases of undertaking any sort of later investment or incurring debt that are depicted in figure 4.7.3 and figure 4.7.4, respectively, operating and exited companies have similar patterns, so not much can be said in that regard.

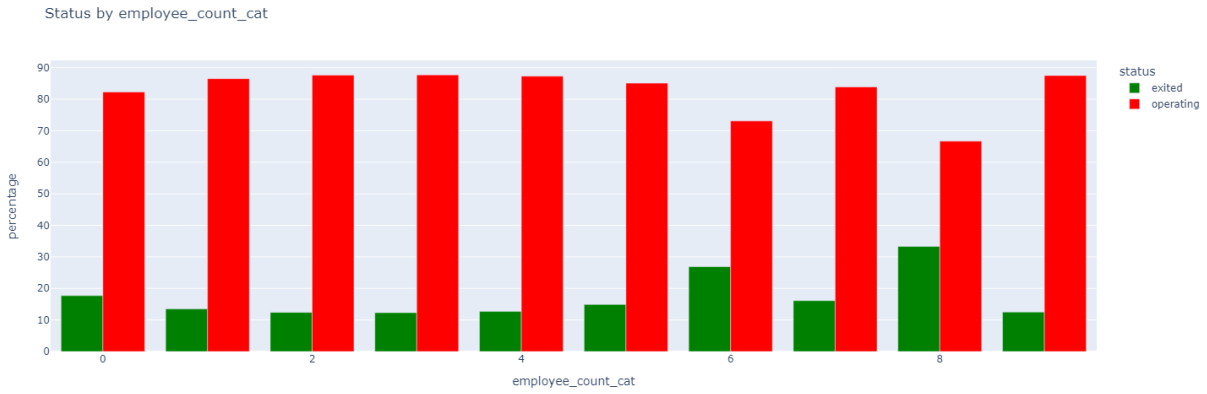

Figure 4.7.5: Barplot bivariate analysis: seed vs Status

Regarding the number of employees, figure 4.7.5 shows that the higher it gets, the more likely for the company to be exited. However, we notice that if it becomes too high, it can actually be beneficial for the company and it is able to stay in operation.

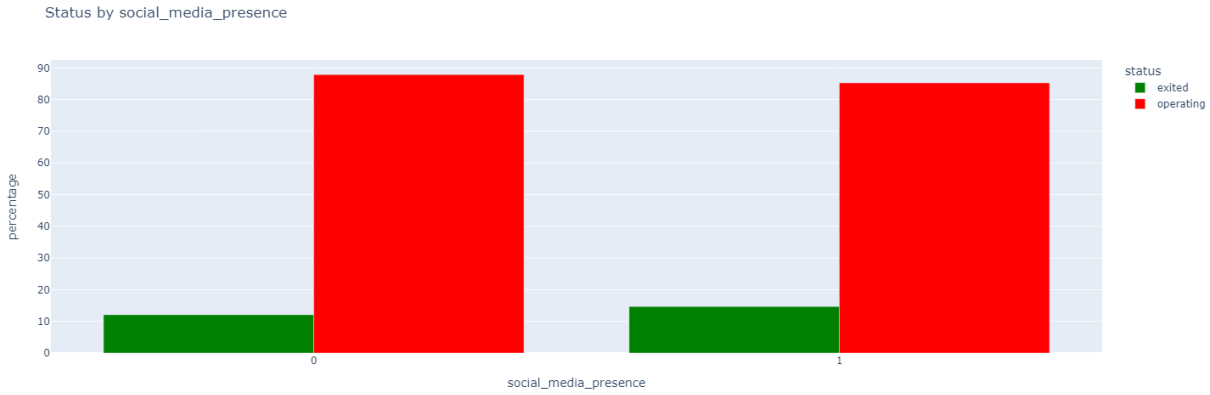

Figure 4.7.6: Barplot bivariate analysis: social media presence vs Status

Also concerning social media presence, not much difference can be spotted in figure 4.7.6, so I will not comment on it any further.

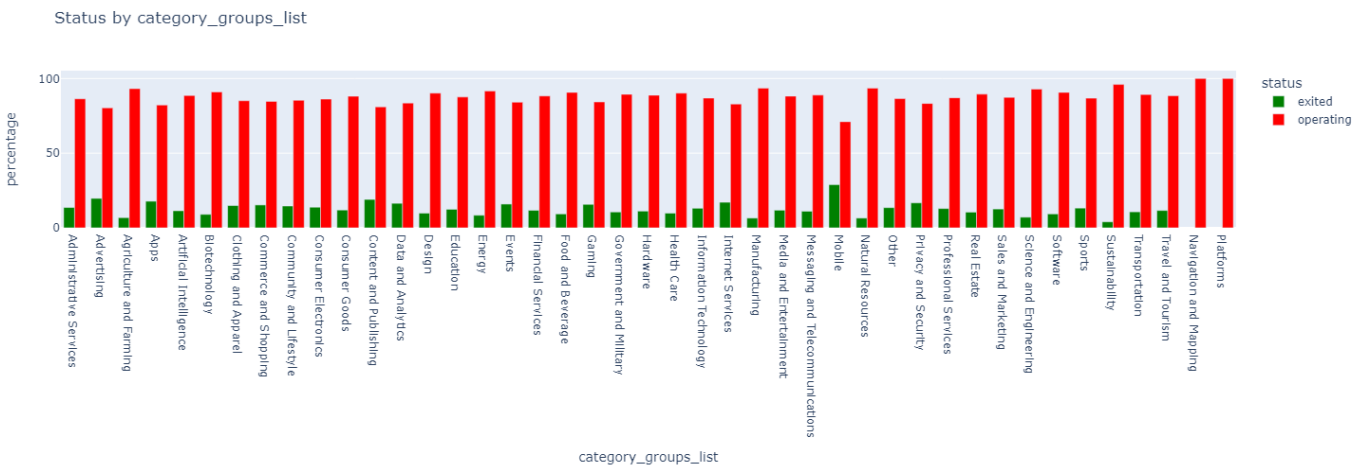

Figure 4.7.7: Barplot bivariate analysis: sector vs Status

Lastly, concerning the sector, we observe in figure 4.7.7 that the Mobile sector has the higher exit rate by far, whereas operating companies are spreaded pretty equally among the other sectors so that there is no clear winner.

c) *Correlation analysis and* Variable selection through statistical tests

i) Correlation Analysis

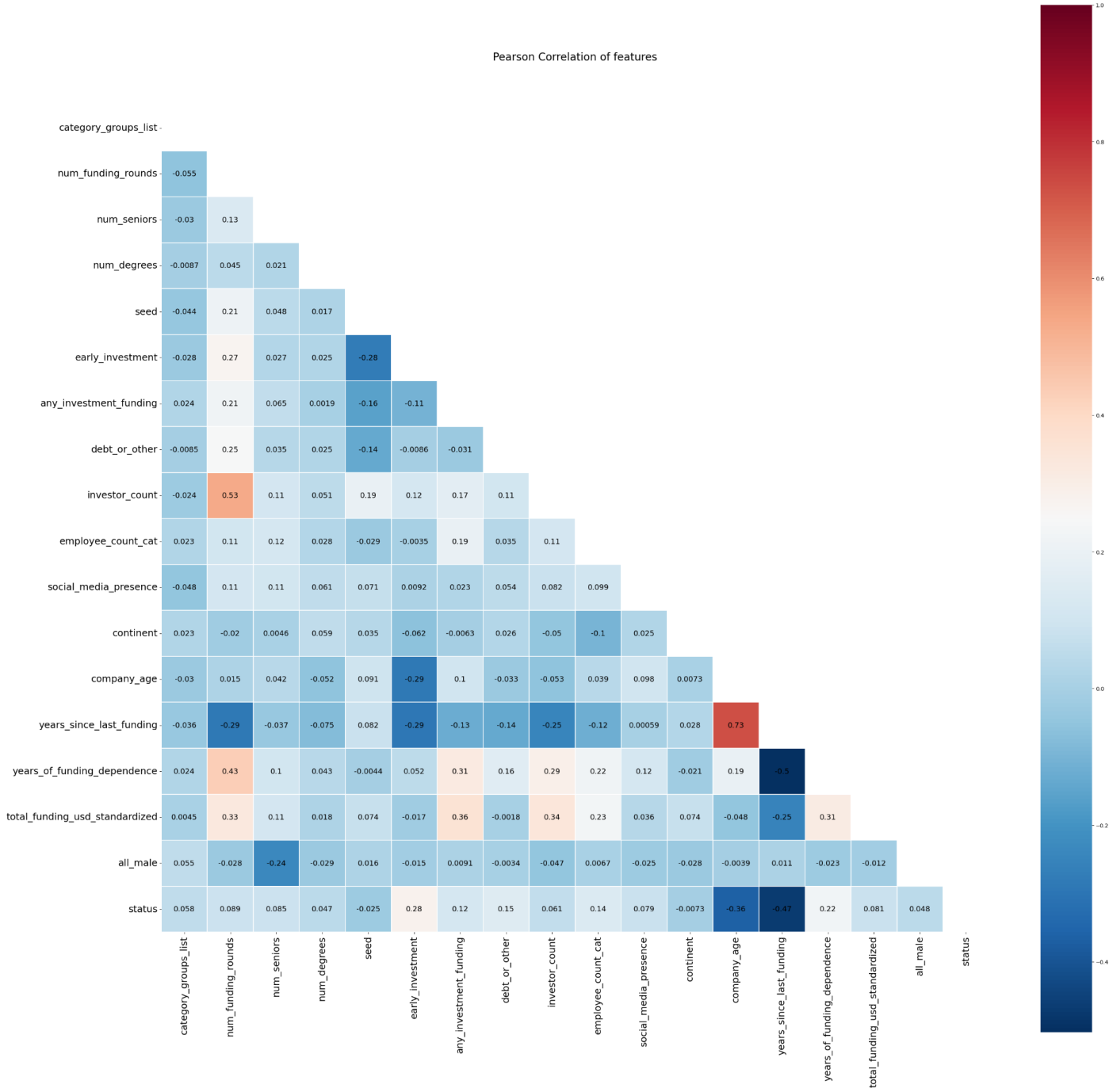

Figure 4.8: Pearson Correlation plot of features

Using a correlation plot between all numerical variables present in our dataset through figure 4.8, we detect some variables like company_age and years_since_last_funding that have a high negative correlation with the target variable, and others like years_of_funding_dependence which have a moderately high correlation with the target variable. However, something that jumps out is the high correlation between years_since_last_funding and company_age, which is bad for our model,

since it can lead to redundancy and overfitting, as well as because of the multicollinearity effect where it becomes difficult to separate between the effect of independent variables on the target feature.

Based on these results, we would like to take a closer look at some variables and inspect their relationship between each other and with the target variable. However, it appears that there aren't many highly linked independent variables, therefore variable selection approaches may not be required for our modeling. Still, in order to ensure that we have the best model feasible, we intend to use LASSO and Ridge regression techniques on the data to analyze the influence of any multicollinearity that may exist between the independent variables. By performing these extra steps, we will ensure that our model is resilient and capable of producing the most accurate forecasts.

ii) Reduced logistic regression model

As alluded to, there are two independent variables that are highly correlated. Let's look at a reduced logistic regression model by using status as the dependent variable; while taking company_age and years_since_last_funding. We will also include an interaction term to see whether the individual effect of these independent variables depends on one another.

	coefficient	**std err**	**z**	**P > |z|**
Intercept	4.21	0.07	59	0.0
years_since_last_funding	-0.97	0.02	-52.99	0.0
company_age	-0.2	0.01	-29.38	0.0
years_since_last_funding: company_age	0.07	0.002	32.74	0.0

Table 8: Reduced logistic regression model analysis with status as dependent variable

The results of the reduced logistic regression model can be seen in table 8. Keep in mind, this model was used after scaling the data so the interpretation will be a bit more technical. What is more, as we have seen from the boxplots, there is not much difference between the scales of the variables among themselves, so the scaling should not have shifted the results drastically. Having said that, allow me to proceed with the model interpretation:

To test the null hypothesis that each predictor variable has no effect on the log-odds of success, the Wald chi-square test was utilized. All predictor parameters are significantly related to the

log-odds of success (p 0.001). Let us interpret these results in terms of odds and probabilities, rather than log-odds as they are seen in the picture.

To draw a clearer picture, the odds of the startup being successful when company age and years since last funding are set to their means are equal to $e^{b0} = e^{4.2} = 66.6$. This is abnormally high with around 98% probability of success, but we will see why later when dealing with class imbalance.

Then, the odds of the startup being successful when company age increases by 1 standard deviation and years since last funding is set to its mean is equal to $e^{b0+b1} = 25.2$. So the probability of this type of firm being successful in the case is 96%, showing that an increase in company age can cause a decrease in success rate.

What is more, the odds of a startup being a success when the company age is set to its mean and years since last funding increases by 1 standard deviation is $e^{b2} = 0.74$. In other words, this case leads to a probability of success of around 43%, showing the importance of these variables. Overall, our findings indicate that years_since_last_funding and company_age are both important predictors of company success, and that the effect of years_since_last_funding on success may be modified by the value of company_age.

Finally, the odds ratio of the company being successful when both company age and years since last funding increase by 1 standard deviation is $e^{b2+b3} = 0.8$. So the probability of a startup being successful in this case is 44%. Considering the inflated values that we are observing in this model because of class imbalance, this means that the interaction between both variables is not as significant as we might have presumed originally. Consequently, we do not necessarily need to remove any of them. Later on, we will see some model selection methods like Lasso and AIC that will help us see which variables are really useful to predict the dependent variable and which are not.

As stated earlier, we do not need to do an association matrix, since all the categorical variables were transformed to numerical ones using label encoding and were plotted in the previous correlation plot

d) Model comparison

i) Vanilla model results

Let us now observe whether our efforts of feature engineering and variable selection will allow us to accurately predict startup success. Again, as I mentioned in the methodology section, we are first interested in comparing the plain logistic regression to one with AIC stepwise selection applied to it, and another with Lasso regularization applied to it. We are also interested in fitting a non-linear model and in this case I start by fitting a random forest model. We perform these comparisons both on the original dataset and on the upsampled dataset to see if something changes. Figure 4.10 displays the accuracy results, coupled with the confusion matrix that we got by comparing the first set of models.

Values in this table are rounded to the nearest 1000th

Model	Accuracy	True Negatives	False Positives	False Negatives	True Positives
Logistic Regression (Original)	0.869	46	1087	43	7439
Random Forest (Original)	0.864	91	1042	129	7353
AIC step (Original)	0.867	45	1088	54	7428
Lasso (Original)	0.869	46	1087	43	7439
Logistic Regression (Upsampled)	0.713	760	373	2098	5384
Random Forest (Upsampled)	0.855	177	956	297	7185
AIC Step (Upsampled)	0.714	768	365	2102	5380
Lasso (Upsampled)	0.713	760	373	2098	5384

Table 9: Vanilla models results on original vs upsampled datasets

At a first glance of table 9, when fitting the models with the normal dataset, it seems that logistic regression has the upper hand with an accuracy of 0.869. Upon further inspection, we notice that random

forest is the clear winner, since the accuracy of the logistic regression model is so high because it detects mostly positive labels (or operating status in the case of our dataset). This can be detected by the fact that only 89 (TN + FN) variables out of 8615 (TN + FN + TP + FP) total features in the validation set were negative predictions by the logistic regression in the original dataset, whereas the rest are positive predictions. This ratio is improved in the upsampled dataset since class imbalance is no longer an issue, but we observe a huge dropoff in the accuracy from 0.87 to 0.71. The same can be said about logistic regression with AIC stepwise selection and Lasso regularization. However, we are interested in seeing whether some variables were excluded using these two techniques.

Model	Features selected
Lasso	All the 17 original variables are present.
AIC selection	Only 8 out of 17 were selected by this model: category_groups_list, seed, any_investment_funding, debt_or_other, continent, company_age, years_since_last_funding, years_of_funding_dependence

Table 10: Models for variable selection

Unsurprisingly, table 10 shows that with Lasso the features are still the same whereas with AIC, only 8 of the 17 independent features are still present. However, because of the fact that random forest is the winner among the current selection of algorithms, and since it is robust to overfitting and is not affected by a large number of features, we will keep things as they are for now.

Let us proceed by seeing the rest of the fitted models and compare them to the random forest algorithm:

values in the following two tables are rounded to the nearest 1000th

Model	Accuracy	Precision	Recall	F1 Score
Classification tree	0.787	0.887	0.866	0.876
KNN	0.853	0.877	0.965	0.919

Random forest	0.862	0.875	0.981	0.925
Logistic regression	0.867	0.873	0.992	0.929
Kernel SVM	0.867	0.868	0.999	0.929

Table 11: Comparison of vanilla models with original dataset

Model	Accuracy	Precision	Recall	F1 Score
Classification tree	0.79	0.887	0.869	0.878
KNN	0.687	0.912	0.708	0.797
Random forest	0.855	0.882	0.962	0.92
Logistic regression	0.71	0.942	0.709	0.809
Kernel SVM	0.701	0.94	0.7	0.802

Table 12: Comparison of vanilla models with upsampled dataset

Once again, table 11 and table 12 go hand in hand with our initial set of results, where random forest seems to be the most robust to class imbalance, maintaining relatively elevated accuracy, precision, recall and F1 Score across the board, so we will stick to it as our main model to optimize.

Consequently, we can go ahead and optimize the random forest model using cross-validation to get the best possible results for our specific problem.

ii) Optimizing best vanilla models

We start by hypothesizing what could be the best number of trees that our random forest model contains. To do that, we iterate a random forest model over the dataset by varying the number of trees from 0 to 100 and check for the best performing model. We are using accuracy here, because it is not an issue for random forest which deals well with class imbalance, as opposed to logistic regression, where solely relying on accuracy would have been a bad metric.

To understand how we can optimize the model, let us understand what are the hyperparameters of a random forest algorithm:

- **n_estimators:** Refers to the number of decision trees included in the random forest model
- **max_features:** Refers to the maximum amount of features which are considered when splitting a node in each decision tree. We use sqrt in order to avoid overfitting.
- **max_depth:** Refers to the depth of the individual decision trees.
- **min_samples_split:** Refers to the minimum number of samples that are needed in order to split a node in a decision tree.
- **min_samples_leaf:** Refers to the minimum number of samples that are required for a node to be a leaf in a decision tree. By leaf node, it means the end of a tree which does not have children.
- **bootstrap:** Can take either true or false as values. By setting it to true, we are resampling the dataset to create a multitude of datasets for training and hence avoid overfitting.

Starting with what is considered the most essential parameter, we see how many decision trees we should include in our random forest model. To do that, we iterate over 100 different values for n_estimators:

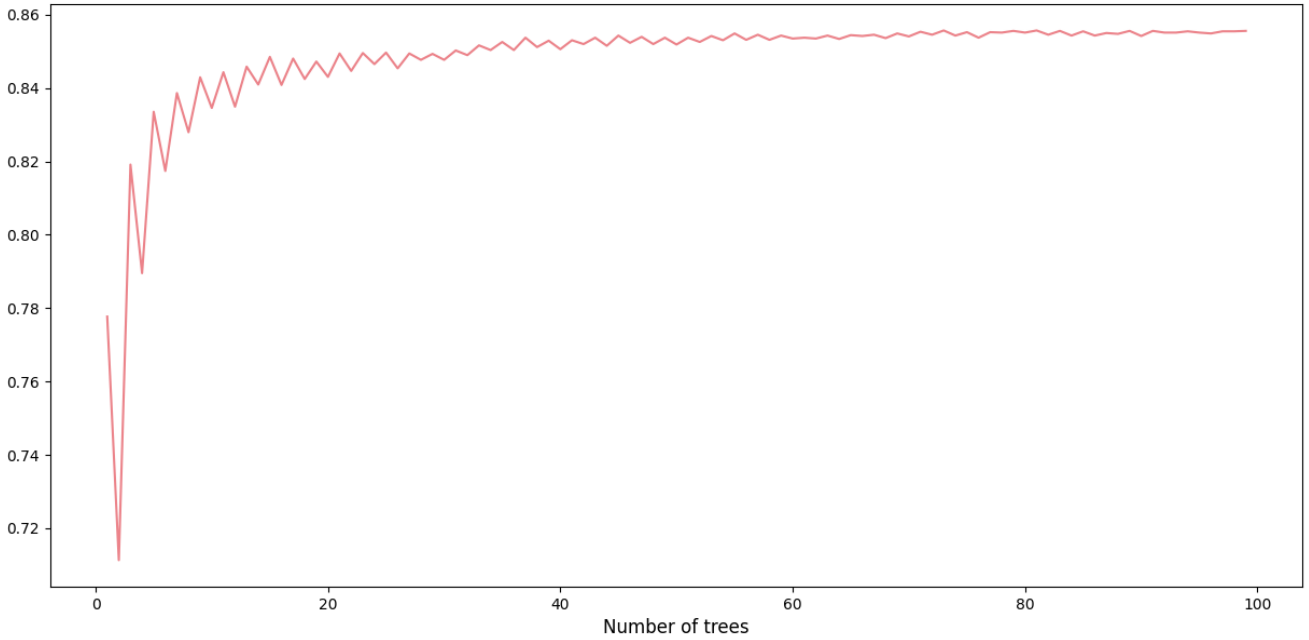

Figure 4.9: Optimal number of trees for the random forest model

We see in figure 4.9 that after a certain number of trees the accuracy stabilizes and having more trees would thus lead to some overfitting.

Note that we will not do a similar graph for all hyperparameters since it would be too time-consuming, and

it is even impossible to do so in some cases. What we do instead is make use of random search and grid search which can do the job of optimization automatically for us.

As explained in the methodology search, we start by using the random search because it is less time-consuming, then go for the grid search to get the best model that is centered around the values of the parameters found in the random search.

Values in the following table are rounded to the nearest 100th

Model	Accuracy	Precision	Recall	F1 Score
RF Vanilla	0.85	0.88	0.96	0.92
RF Random Search	0.86	0.88	0.97	0.92
RF Grid Search	0.86	0.88	0.97	0.92

Table 13: Results on optimized random forest models

Table 13 demonstrates that there is an ever-so-slight improvement in accuracy, recall and F1 score when comparing the newly generated models with the original random forest. If we compare the random forest with random search to the random forest with grid search we observe almost identical results that are quite indistinguishable. However, this slight improvement to go to 86% accuracy was expected, since having around 85% accuracy in the original model is already high, especially that not only positive labels are being predicted. This can be seen with the even more elevated scores of the other metrics, whether it is precision, recall or F1 Score. We will talk more about these results in the interpretation section.

iii) Applying best model on test set

We must remember that up until now, we have been looking at results on the validation set in order to tune the parameters as needed. Table 14 exhibits how our best model would perform on new and unseen data, and by that we are referring to the true test data that we defined at the very start. By best model, we can either take the random forest with the hyperparameters that we got either from the random search or the grid search since we did not see much improvement. Here are the results we get:

Model	Accuracy	Precision	Recall	F1 Score
RF Grid Search	0.85	0.88	0.95	0.92

Table 14: Results on original test set

The resulting metrics are extremely satisfactory and similar to those we got on the validation set. Indeed, we get an accuracy of about 85%, a precision of about 88%, a recall of around 95% and a F1 Score of approximately 92%. This shows that our model can indeed be applied on new and unseen data, generalizing well to this type of information.

05. Discussion

a. Interpretation of Findings and Implications for Practice

We can draw various findings about startup success based on our model comparison and optimisation outcomes. First, we discovered that, even with AIC stepwise selection and Lasso regularization, logistic regression did not outperform the random forest technique. This is because logistic regression is more sensitive to class imbalance, which was present in our sample. The random forest method, on the other hand, is more resistant to class imbalance and can accommodate a large number of features without overfitting. As a result, we chose to optimize the random forest algorithm for our particular case.

We discovered that the optimal model has an accuracy of roughly 85% after optimizing the random forest algorithm, which is a high accuracy for forecasting company success. This implies that the variables we included in our model, such as funding amount, company age, and industry type, are important predictors of startup success. Furthermore, the excellent precision, recall, and F1 score values indicate that our approach is effective at both finding successful businesses and avoiding false positives.

Although it is not immediately apparent, our findings can prove to have vastly important practical implications. On the one hand, investors and venture capitalists can use our technique to identify firms with high potential for success. This allows investors to make informed decisions about which companies they should invest in and which not, without going through the burden of having to go through insufficient financial statements and mostly blindly trusting the entrepreneur. On the other hand, this indirectly benefits startup owners because more investors will be willing to put their money in these kinds of companies, because these investments will be seen as less risky than they currently are. But also, entrepreneurs will benefit directly, as they will be able to know whether everything is going well in their current business model, or whether they should prioritize an element over another. Furthermore, policymakers can utilize my model to discover factors associated with startup success. This will allow them to put in place policies that promote the growth of startups in various industries.

b. Comparison with Previous Studies and Models

My results seem to be optimal, but let us find out how they compare to other papers who have dealt with this exact same topic. It would only be fair for me to compare papers who have used Crunchbase data, since those who did not were faced with the disadvantage of not having a reliable source of information.

Following a thorough examination of the literature, we discovered a study conducted by Bangdiwala et al. (2022) that employed a Crunchbase dataset to predict startup success. They employed logistic regression and random forest algorithms to forecast company performance based on criteria including funding amount, employee count, and industry. Their random forest model attained an accuracy of 92.43%, which is higher than our optimized random forest model's accuracy, but with a shocking 67% AUC score.

In another study by Yiea-Funk Te et al. (2020), a wide variety of machine learning algorithms were used. Notably, they employed logistic regression, decision trees, and gradient boosting to predict startup success based on variables similar to those we used but with different feature engineering. Their best model attained an accuracy which is still lower than our optimized random forest model's accuracy. A difference of around 3% might not seem significant, but in the grand scheme of it, it could be millions of dollars of investments saved by the investors. This is why accuracy is such an important metric to look at, but beware of the model used because we have seen that accuracy would have not been a reliable model with logistic regression in this case.

Also, I would like to mention another thesis paper done by Francisco Ramadas da Silva Ribeiro Bento at the Nova University. In his study, the author of the work where he achieved 83% accuracy and 92% precision. This is the highest precision obtained by anyone using the crunchbase data. However, in my opinion the accuracy is not as high because we relied too much on binary features.

I am proud to state that from an accuracy point of view, I outperformed most of the previous research papers that I could find regarding the topic of predicting startup success based on Crunchbase data. Through the use of a random forest, which we optimized by using grid search and random search with cross-validation, I was able to earn an accuracy of 86%, which is 3% higher than the results of the aforementioned papers. Regarding the other metrics, I was a bit behind on the precision metric when comparing it to Francisco Ramadas da Silva Ribeiro Bento’s work, but I was very well above average compared to the other works, with a precision of around 87%. optimized for our specific purpose, our study outperformed earlier studies in forecasting company success. This implies that the variables we included in our model, such as investment amount, firm age, and business type, are strong predictors of startup success. However, it is important to note that the datasets used in earlier studies and our work may differ in terms of the variables included and the time period covered, which may alter the model's accuracy.

c. Limitations and Future Research Directions

Despite these promising results, some limitations are naturally present. For a start, I have collected my data from Crunchbase. Crunchbase being an open-source platform, anyone can edit and publish new content. However, it must be said that it is very well moderated and is seen by the most reliable sources as an extremely powerful tool to get insights from.

Also, despite the extensive feature engineering that I performed and the vast amount of features provided to me by Crunchbase to help me in my startup prediction task, there are still a lot of intricacies that go into founding a startup and running it that are not accounted for in my model. Indeed, I only used 18 variables because I favored data quality over data quantity, and because I do not possess a lot of computing power and resources to be able to do more than that. This is an area I would like to see other people improve on.

All in all, I would say that the implications of my project far outweigh the limitations that occur in these kinds of projects, and I have to say that I am thoroughly pleased with the results I have got.

06. Conclusion

Through this empirical study, my goal was to provide investors with a powerful tool for them to navigate the world of startups without going through the burdens which come with investing in startups. The research question stated at the very beginning was whether machine learning models could predict startup success in an accurate and reliable way in order to eliminate the guesswork done by investors. To solve this issue, I first set out an extensive literature review section that provided a better understanding of the ambiguity surrounding the startup scene and the challenges faced by startups, as well as the potential of ML in addressing these challenges. After having set the scene, I moved on to the methodology part in which I explained the key decisions I took. Most notably, I highlighted the substantial feature engineering and variable selection that went into developing the merged table, as well as the difficulties encountered in coping with the imbalanced target class. The univariate and bivariate analysis allowed us to hypothesize on the importance of some of the variables and stake and helped us understand their relationships with each other, which was also useful for the correlation and regression analysis part. SMOTE was implemented in order to deal with this class imbalance, but the random forest being a robust algorithm for dealing with class imbalance proved to have extremely high accuracy for both the original and upsampled dataset at around 85%. I then proceeded to improve on that model through the implementation of random search and grid search, only to get a 1% improvement on the original model. Overall, the results obtained were very satisfactory and were higher than the averages of papers written on a similar topic using Crunchbase data. Some limitations like the strict amount of computing power at my disposal and the few amount of variables used might be a roadblock for achieving even better results and I believe that future work can improve hugely in that regard. This is not to say that 86% when put in perspective can save investors millions of dollars.

07. References

Henrekson, M., & Sanandaji, T. (2018). Small business activity does not measure entrepreneurship. Proceedings of the National Academy of Sciences, 115(30), 7468-7473.

Shane, S. (2000). Prior knowledge and the discovery of entrepreneurial opportunities. Organization Science, 11(4), 448-469.

Blank, S. (2013). Why the lean start-up changes everything. Harvard Business Review, 91(5), 63-72.

Ries, E. (2011). The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses. Crown Business.

Fuglie, K. O., Heisey, P. W., King, J. L., Day-Rubenstein, K. A., & Schimmelpfennig, D. E. (2020). Agricultural research investment and policy reform in high-income countries. USDA Economic Research Service.

Chen, J. H., & Asch, S. M. (2017). Machine learning and prediction in medicine-beyond the peak of inflated expectations. The New England Journal of Medicine, 376(26), 2507-2509.

DeTienne, D. R., & Chandler, G. N. (2007). Opportunity identification and its role in the entrepreneurial classroom: A pedagogical approach and empirical test. Academy of Management Learning & Education, 6(3), 420-431.

DeTienne, D. R., McKelvie, A., & Chandler, G. N. (2019). Making sense of entrepreneurial exit strategies: A typology and test.

Lepak, D. P., Smith, K. G., & Taylor, M. S. (2019). Value creation and value capture: A multilevel perspective. Academy of Management Annals, 13(1), 52-86.

McGrath, R. G. (2010). Business models: A discovery driven approach. Long Range Planning, 43(2-3), 247-261.

Peng, M. W., Ahlstrom, D., Carraher, S. M., & Shi, W. S. (2019). An institution-based view of global IPR history. Journal of International Business Studies, 50(9), 1465-1487.

Mitchell, T. M. (1997). Machine learning. McGraw Hill.

Bangdiwala, M., Mehta, Y., Agrawal, S., & Ghane, S. (2022). Predicting Success Rate of Startups using Machine Learning Algorithms. In 2022 2nd Asian Conference on Innovation in Technology (ASIANCON) (pp. 1-6). IEEE. doi: 10.1109/ASIANCON55314.2022.9908921

Perifanis, N.-A., & Kitsios, F. (2021). Investigating the Influence of Artificial Intelligence on Business Value in the Digital Era of Strategy: A Literature Review. Journal of Open Innovation: Technology, Market, and Complexity, 7(2), 90.

Singh, K., Booma, P. M., & Eaganathan, U. (2020). E-commerce system for sale prediction using machine learning technique. Journal of Physics Conference Series, 1712(1), 012042. doi: 10.1088/1742-6596/1712/1/012042

Mah, P.M., Skalna, I. and Muzam, J. (2021). Natural Language Processing and Artificial Intelligence for Enterprise Management in the Era of Industry 4.0. Symmetry, 13(5), 746.

Murali, P. K., Wang, C., Lee, D., Dahiya, R., & Kaboli, M. (2021). Deep Active Cross-Modal Visuo-Tactile Transfer Learning for Robotic Object Recognition. IEEE Robotics and Automation Letters (RA-L), 6(3), 4541-4548. doi: 10.1109/LRA.2021.3067464

Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data mining for credit card fraud: A comparative study. Decision Support Systems, 50(3), 602-613. doi: 10.1016/j.dss.2010.10.003

Himeur, Y., Sayed, A., Alsalem, A., Bensaali, F., Amar, A., Varlamis, I., ... & Dimitrakopoulos, G. (2021). Blockchain-based Recommender Systems: Applications, Challenges and Future Opportunities. Journal of Systems and Software, 110951. DOI: 10.1016/j.jss.2021.110951

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255-260.

Géron, A. (2017). Hands-on machine learning with Scikit-Learn and TensorFlow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media, Inc.

Yiea-Funk Te, Wieland, M., Frey, M., Pyatigorskaya, A., Schiffer, P., & Grabner, H. (2022). Predicting the Success of Startups Using Crunchbase and Linkedin Data. KeAi: The Journal of Finance & Data Science.

08. Appendices

a. Data processing and visualization scripts

Overview of dataframe to deal with data inconsistencies

```
def dataoveriew(df):
    print('Overview of dataset\n')
    print("What are the types of features?\n")
    print(df.dtypes)
    print("\nHow many missing values?", df.isnull().sum().values.sum())
    print("\nHow many unique values?")
    print(df.nunique())
```

Univariate analysis

Histogram for numerical variables

```
def histogram_numerical(df_continous, df_num = df_num):
  fig = plt.figure(figsize=(15, 12))
  plt.suptitle('Histograms of Continuous Numerical Columns\n',
horizontalalignment="center",
                fontsize=26)
  for i in range(df_continuous.shape[1]):
      plt.subplot(5, 2, i+1)
      f = plt.gca()
      f.set_title(df_num.columns.values[i])
      vals = np.size(df_num.iloc[:, i].unique())
      plt.hist(df_num.iloc[:, i], bins=vals, color='#ec838a')
      plt.tight_layout(rect=[0, 0.03, 1, 0.95])
      if vals >= 100:
          vals = 100
  plt.show()
```

Barplot for categorical variables (similar one for discrete)

```
def barplot_cat(df_cat, df=df):
  fig, axes = plt.subplots(nrows=3, ncols=3, figsize=(35, 30))
  plt.suptitle('Bar Plots of Categorical Columns\n', horizontalalignment="center",
               fontsize=26)
  for i, item in enumerate(df_cat):
      if i < 3:
          ax = df[item].value_counts().plot(kind='bar', ax=axes[i, 0], rot=0,
color='#f3babc')
          ax.set_title(item, fontdict={'fontsize': 26})

      elif i >= 3 and i < 6:
          ax = df[item].value_counts().plot(kind='bar', ax=axes[i - 3, 1], rot=0,
color='#9b9c9a')
          ax.set_title(item, fontdict={'fontsize': 20})

      elif i < 8:
          ax = df[item].value_counts().plot(kind='bar', ax=axes[i - 6, 2], rot=0,
color='#ec838a')
          ax.set_title(item, fontdict={'fontsize': 20})

      elif i == 8:
          ax = df[item].value_counts().plot(kind='bar', ax=axes[i - 6, 2], rot=45,
color='#ec838a')
          ax.set_title(item, fontdict={'fontsize': 20})
          ax.tick_params(axis='x', labelsize=8)

  plt.show()
```

Bivariate Analysis

Numerical vs Target: Histogram + boxplot

```
# Defining histogram function for bivariate analysis
def hist(feature):
    group_df = df.groupby([feature, 'status']).size().reset_index()
    group_df = group_df.rename(columns={0: 'Count'})
    figure = px.histogram(group_df, x=feature, y='Count', color='status',
marginal='box', title=f'Status frequency to {feature} distribution',
color_discrete_sequence=["green", "red"])
    figure.show()
```

```
#defining boxplot for numerical vars in bivariate analysis
def hist_box(feature, df):
```

```
    fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(15, 5))
    fig.suptitle(f"{feature} vs. Status", fontsize=16, y=1.1)

    sns.histplot(x=feature, hue='status', data=df, ax=axes[0], kde=True,
multiple="stack", palette=["green", "red"])
    axes[0].set_xlabel(feature, fontsize=12)
    axes[0].set_ylabel("Frequency", fontsize=12)
    axes[0].set_title(f"{feature} Distribution", fontsize=14)

    sns.boxplot(x='status', y=feature, data=df, ax=axes[1], palette=["green",
"red"])
    axes[1].set_xlabel("Status", fontsize=12)
    axes[1].set_ylabel(feature, fontsize=12)
    axes[1].set_title(f"{feature} vs. Status", fontsize=14)

    plt.tight_layout()
    plt.show()
```

Categorical vs Target: Bar plots

```
#Define bar plots for bivariate analysis
def bar(feature, df=df):
    # Groupby the categorical feature
    temp_df = df.groupby([feature, 'status']).size().reset_index()
    temp_df = temp_df.rename(columns={0: 'Count'})

    # Calculate the total count for each category of the categorical feature
    category_counts = temp_df.groupby(feature)['Count'].sum().reset_index()
    category_counts = category_counts.rename(columns={'Count': 'total_count'})

    # Merge the category counts with the temp_df
    temp_df = temp_df.merge(category_counts, on=feature)

    # Calculate the percentage of each status within each category
    temp_df['percentage'] = round(temp_df['Count'] / temp_df['total_count'] * 100,
1)

    # Setting graph framework
    fig = px.bar(temp_df, x=feature, y='percentage', color='status', title=f'Status
by {feature}', barmode="group",
                 color_discrete_sequence=["green", "red"])

    return fig.show()
```

b. Data preparation code for modeling phase

Categorical variables to numeric transformation

```
def to_numeric(s):
  if s == "operating":
    return 1
  elif s == "exited":
    return 0

df_inter = df.copy()
df_inter["status"] = df["status"].apply(to_numeric)

# creating instance of labelencoder
labelencoder = LabelEncoder()
#using label encoder for these two columns as there is a lot of variables
df['continent'] = labelencoder.fit_transform(df['continent']) # using label encoder
on continent
df['category_groups_list'] = labelencoder.fit_transform(df['category_groups_list'])

#Converting target variable to numerical
df["status"] = df["status"].apply(to_numeric)
```

Data preparation for model: train/validation/test split

```
X = df.drop('status', axis=1) # drop the target variable from X
y = df['status']

# Split your data into training and testing sets
X_train_tot, X_test_tot, y_train_tot, y_test_tot = train_test_split(X, y,
test_size=0.1, random_state=42)

# Split training data into training and validation sets
X_train, X_test, y_train, y_test = train_test_split(X_train_tot, y_train_tot,
test_size=0.2, random_state=42)
```

SMOTE implementation to deal with class imbalance

```
# Instantiate SMOTE
smote = SMOTE(random_state=42)

# Resample using SMOTE
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

# Convert resampled data to a DataFrame
df_upsampled_train = pd.DataFrame(X_train_resampled, columns=X_train.columns)
df_upsampled_train['status'] = y_train_resampled

# Separate X and y in the upsampled training set
X_upsampled_train = df_upsampled_train.drop('status', axis=1)
y_upsampled_train = df_upsampled_train['status']
```

Correlation plot for variable selection

```
#creating correlation matrix
colormap = plt.cm.viridis
plt.figure(figsize = (35, 35))
plt.title('Pearson Correlation of features', y = 1.05, fontsize = 15)
matrix = np.triu(df.corr())
sns.heatmap(df_upsampled_train.corr(), linewidth = 0.1, vmax = 1.0,
            square =True,
            cmap=colormap, linecolor = 'white', annot=True, mask = matrix)
plt.xticks(fontsize = 18)
plt.yticks(fontsize = 18)

# save plot as PNG
plt.savefig("my_plot.png", dpi=300, bbox_inches='tight')

#Show plot
plt.show()
```

Reduced logistic regression for model interpretation: slight change in that I used normal df, not the upsampled one

```
# Specify the formula for logistic regression
formula = 'status ~ years_since_last_funding * company_age'

# Fit the model using training data
model = sm.formula.glm(formula=formula, family=sm.families.Binomial(),
                       data=df_upsampled_train).fit()

print(model.summary())
```

c. Predictive Modeling Code and Results

First set of models fit on original dataset to view confusion matrix

```
#Original
# Fit logistic regression on original dataset
lr = LogisticRegression(random_state=42)
lr.fit(X_train, y_train)
y_pred = lr.predict(X_test)
original_lr_accuracy = accuracy_score(y_test, y_pred)
original_lr_cm = confusion_matrix(y_test, y_pred)

# Fit random forest on original dataset
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)
original_rf_accuracy = accuracy_score(y_test, y_pred)
original_rf_cm = confusion_matrix(y_test, y_pred)

#Fit logistic regression model on original dataset with AIC step selection
lr_aic = LogisticRegression(solver='liblinear', random_state=42)
my_selector = RFE(lr_aic, step=1)
my_selector.fit(X_train, y_train)
y_pred_aic = my_selector.predict(X_test)
original_lr_aic_accuracy = accuracy_score(y_test, y_pred_aic)
original_lr_aic_cm = confusion_matrix(y_test, y_pred_aic)


#Fit Lasso on original dataset
lasso = LogisticRegression(solver='liblinear', penalty='l1', random_state=42)
lasso.fit(X_train, y_train)
y_pred_lasso = lasso.predict(X_test)
original_lasso_accuracy = accuracy_score(y_test, y_pred_lasso)
original_lasso_cm = confusion_matrix(y_test, y_pred_lasso)
```

First set of models fit on upsampled dataset to view confusion matrix

```
#Upsampled
# Fit logistic regression on upsampled dataset
lr_up = LogisticRegression(random_state=42)
lr_up.fit(X_upsampled_train, y_upsampled_train)
y_pred_up = lr_up.predict(X_test)
upsampled_lr_accuracy = accuracy_score(y_test, y_pred_up)
```

```
upsampled_lr_cm = confusion_matrix(y_test, y_pred_up)

# Fit random forest on upsampled dataset
rf_up = RandomForestClassifier(random_state=42)
rf_up.fit(X_upsampled_train, y_upsampled_train)
y_pred_up = rf_up.predict(X_test)
upsampled_rf_accuracy = accuracy_score(y_test, y_pred_up)
upsampled_rf_cm = confusion_matrix(y_test, y_pred_up)

#Fit logistic regression model on upsampled dataset with AIC step selection
lr_aic_up = LogisticRegression(solver='liblinear', random_state=42)
selector_up = RFE(lr_aic_up, step=1)
selector_up.fit(X_upsampled_train, y_upsampled_train)
y_pred_aic_up = selector_up.predict(X_test)
original_lr_aic_accuracy_up = accuracy_score(y_test, y_pred_aic_up)
original_lr_aic_cm_up = confusion_matrix(y_test, y_pred_aic_up)


#Fit Lasso on upsampled dataset
lasso_up = LogisticRegression(solver='liblinear', penalty='l1', random_state=42)
lasso_up.fit(X_upsampled_train, y_upsampled_train)
y_pred_lasso_up = lasso_up.predict(X_test)
original_lasso_accuracy_up = accuracy_score(y_test, y_pred_lasso_up)
original_lasso_cm_up = confusion_matrix(y_test, y_pred_lasso_up)
```

Second set of models fit on original dataset

```
#Original dataset
# Fit each model to the training set and evaluate its performance on test set
models = [('LR', LogisticRegression(solver='liblinear')),
          ('KSVM', SVC(kernel='rbf')),
          ('KNN', KNeighborsClassifier()), ('CT', DecisionTreeClassifier()),
          ('RF', RandomForestClassifier(n_estimators=100, random_state=42))]

results = []
for name, model in models:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    prec = precision_score(y_test, y_pred)
    rec = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    results.append([name, acc, prec, rec, f1])
```

```
# Create a DataFrame to display the results
results_df = pd.DataFrame(results, columns=['Model', 'Accuracy', 'Precision',
                                            'Recall', 'F1 Score'])
results_df = results_df.sort_values(['Precision', 'Recall'], ascending=False)
print(results_df)
```

Second set of models fit on upsampled dataset

```
#Upsampled
# Fit each model to the training set and evaluate its performance on test set
models = [('LR', LogisticRegression(solver='liblinear')),
          ('KSVM', SVC(kernel='rbf')),
          ('KNN', KNeighborsClassifier()), ('CT', DecisionTreeClassifier()),
          ('RF', RandomForestClassifier(n_estimators=100, random_state=42))]

results = []
for name, model in models:
    model.fit(X_upsampled_train, y_upsampled_train)
    y_pred_up = model.predict(X_test)
    acc = accuracy_score(y_test, y_pred_up)
    prec = precision_score(y_test, y_pred_up)
    rec = recall_score(y_test, y_pred_up)
    f1 = f1_score(y_test, y_pred_up)
    results.append([name, acc, prec, rec, f1])

# Create a DataFrame to display the results
results_df = pd.DataFrame(results, columns=['Model', 'Accuracy', 'Precision',
                                            'Recall', 'F1 Score'])
results_df = results_df.sort_values(['Precision', 'Recall'], ascending=False)
print(results_df)
```

d. Script for model optimization

Optimizing random forest using random search:

```
# improve random forest
print('Parameters currently in use:\n')
pprint(rf_up.get_params())
results = pd.DataFrame(columns=['Model', 'Accuracy', 'Precision', 'Recall',
                                'F1 Score'])

# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start=200, stop=600, num=10)]
# Number of features to consider at every split
max_features = ['sqrt']
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(10, 40, num=11)]
max_depth.append(None)
# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 4]
# Method of selecting samples for training each tree
bootstrap = [True, False]
# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf,
               'bootstrap': bootstrap}
pprint(random_grid)

# Use the random grid to search for best hyperparameters
# First create the base model to tune
rf = RandomForestClassifier()

# Perform RandomizedSearchCV
rf_random = RandomizedSearchCV(estimator=rf,
                               param_distributions=random_grid,
                               n_iter=20, cv=2,
                               verbose=4, random_state=42,
                               n_jobs=-1)

# Fit the random search model
rf_random.fit(X_upsampled_train, y_upsampled_train)
```

```
# Print the best hyperparameters found by RandomizedSearchCV
print(rf_random.best_params_)

# Create a new random forest model with the best hyperparameters
better_rf = RandomForestClassifier(**rf_random.best_params_)

# Train the model on the upsampled training data
better_rf.fit(X_upsampled_train, y_upsampled_train)

# Use the trained model to make predictions on the test data
y_pred = better_rf.predict(X_test)

# Calculate evaluation metrics for the model
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

# Append the evaluation metrics to the results dataframe
model_results = pd.DataFrame([['RF Random Search',
                               acc, prec, rec, f1]],
                             columns=['Model', 'Accuracy', 'Precision',
                                      'Recall', 'F1 Score'])

results = results.append(model_results)


# Sort the results dataframe by accuracy and precision in descending order
results = results.sort_values(["Accuracy", "Precision"],
                              ascending=[False, False])

print(results)
```

Optimizing latest model using grid search:

```
# Create the parameter grid based on the results of random search
param_grid = {
    'bootstrap': [False],
    'max_depth': [22, 28],
    'max_features': ['sqrt'],
    'min_samples_leaf': [1, 2],
    'min_samples_split':  [2.0, 5],
```

```
    'n_estimators': [555, 600]
}
# Create a based model
rf = RandomForestClassifier()
# Instantiate the grid search model
# n_jobs = -1 uses 100% of the cpu of one of the cores: makes the process faster
grid_search = GridSearchCV(estimator = rf, param_grid = param_grid,
                           cv = 2, n_jobs = -1, verbose = 2)


grid_search.fit(X_upsampled_train, y_upsampled_train)
pprint(grid_search.best_params_)
best_grid = grid_search.best_estimator_
pprint(best_grid)
# Predicting the Test set results
y_pred = best_grid.predict(X_test)
# Evaluate results
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
model_results = pd.DataFrame([['RF Grid Search',
                               acc, prec, rec, f1]],
                             columns=['Model', 'Accuracy', 'Precision',
                                      'Recall', 'F1 Score'])
results = results.append(model_results, ignore_index=True)
results = results.sort_values(["Precision",
                               "Recall"], ascending=False)
print(results)
```