Imperial College London

Department of Mathematics

Increasing Venture Capital Investment Success Rates Through Machine Learning

Author: Thomas Hengstberger (CID: 01822754)

A thesis submitted for the degree of

MSc in Mathematics and Finance, 2019-2020

Declaration

The work contained in this thesis is my own work unless otherwise stated.

Acknowledgements

First and foremost, I would like to thank my amazing wife for all her support and encouragement in this academic pursuit - without her this surely would not have been possible. She is truly the most amazing person I am fortunate enough to have in my life.

I would also like to thank my supervisor Dr Mikko Pakkanen for his support in taking on this research topic and for providing guidance as the research progressed. His feedback was invaluable in determining which results to pursue further and how to improve the structure of the research, particularly as new results emerged and the research goals evolved.

I would also like to express my gratitude to all of the lecturers in the MSc Mathematics and Finance course, for making the course content genuinely interesting and relevant. A lot of the structure and ideas that went into this research were inspired by the topics covered throughout the year.

Finally, I would also like to thank the Crunchbase team for providing academic access to their database, particularly as this was processed at the height of the COVID-19 lockdown periods in the UK and US. Without their generosity and willingness to share their data, this paper would not have been possible.

Abstract

This thesis explores the use of machine learning models to identify successful startup investments for the purposes of Venture Capital (VC) investing. As this research topic is relatively new and not typically a focus for quantitative model development, the process of how VC-investments generate returns is first formalised and then translated into a binary target variable. The forward looking target variables developed from this analysis create a point of divergence from existing research, as they require both a future investment entry point (the *initial* event) and a subsequent investment *exit event*, as opposed to only a future funding event.

A time series of data covering 64,197 startups founded between 2000-2017 in the UK and US is developed using static data snapshots sourced from Crunchbase. This time series is then utilised to encode the forward looking target variables. The time series is split into historical training sets and future test sets. To imitate how predictive models might be used in a commercial setting, four machine learning models (logistic regression, random forest, support vectors machines and extreme gradient boosting) are developed using historical training sets, while their performance is evaluated based on the predictive accuracy achieved on future test sets. The resulting models developed on this principle are able to correctly identify successful startup investments in 46-52% of cases within 4 years (average result over 10 randomised train-test iterations), depending on the investment target identification requirements, which compares favourably to the average VC success rate of *up to* ca. 30%.

The results presented in this paper are unique as this appears to be the first research which strictly segregates historical and future data for training and evaluation purposes using the time series data developed from Crunchbase. It also appears to be the only research which aims to predict outcomes that directly influence the returns of VC portfolios and in doing so is much more reflective of how these models would be used in a commercial setting.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Financial markets can be broadly split into *public markets* and *private markets*. Public markets consist of assets, such as stocks and bonds, that can be traded on publicly accessible platforms (e.g. a stock exchange), whereas private markets encompass all other non-publicly traded assets, such as privately held companies (startup or mature companies), real estate, infrastructure and private debt. Whereas the general public can trade assets in public markets or have their investments managed by fund managers (professional investors), private market investments can typically only be accessed through fund managers with the necessary skills and experience to understand the market dynamics and investment risks.

Public market fund managers are estimated to oversee around \$3.05 trillion in assets [1], while private market fund managers are estimated to oversee more than twice this amount at an estimated \$6.50 trillion [2]. Within either market, fund managers are typically classified according to the types of opportunities they pursue. With a focus on private markets, the typical, high level fund manager categories (based on [2]) are listed in Table 1.1.

Fund Type	**Description**
Buyout	Acquire full ownership or majority stakes in any mature and typically profitable company (public or private), with the aim to increase profitability.
Growth	Acquire minority or controlling stakes in any semi-mature company to finance expansion or restructuring.
Venture Capital	Acquire minority stakes in early- or mid-stage, high growth-potential technology and life sciences companies (startups) to provide financing for continued growth.
Real Estate	Acquire land or properties and develop, operate or improve them to increase their value.
Private Debt	Direct lending or acquisition of loans extended to, typically, privately held companies.
I&NR[1]	Acquire, build or develop infrastructure and/ or natural resources projects.
Other	All other private market investments that do not fit into the above categories.

Table 1.1: Overview of Private Market Strategy Classifications

One of the key attractions of public markets is the availability of pricing and liquidity, meaning fund managers can replace or rebalance portfolios as required, by buying, selling or even short-selling assets. In comparison, private market investments are difficult to value and highly illiquid, so that assets are bought with the understanding that these have a long holding period typically ranging anywhere from 5-15 years. By implication, this means that private market fund strategies are long-only and so investors have little or no ability to replace investments or rebalance portfolios once transactions have been executed. Although investment selection is fundamental for both public and private market assets, the inability to sell assets (without significant transaction costs/

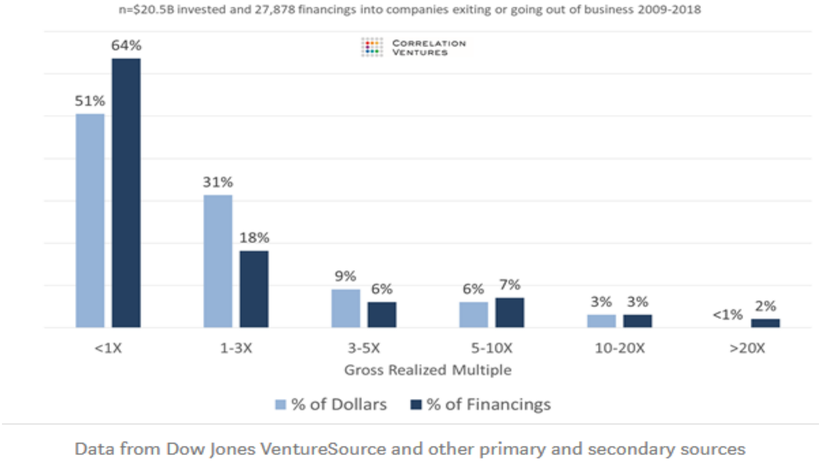
[1] Infrastructure & Natural Resources

losses) or rebalance portfolios makes the selection of investments in private market portfolios even more important. In this paper, we will focus on Venture Capital investment selection and whether this can be improved with machine learning models built using data sets that cover a large volume of global startups.

1.1 Venture Capital Portfolios

To understand *Venture Capital* portfolios, we first note that ca. 75% of fund managers manage less than $250 million and the median investment value for early and mid stage investments ranges from $7.5-18.3 million (see Figures A.1 and A.3). This means that the median Venture Capital fund will select between 10 to 30 investments from which returns must be generated.

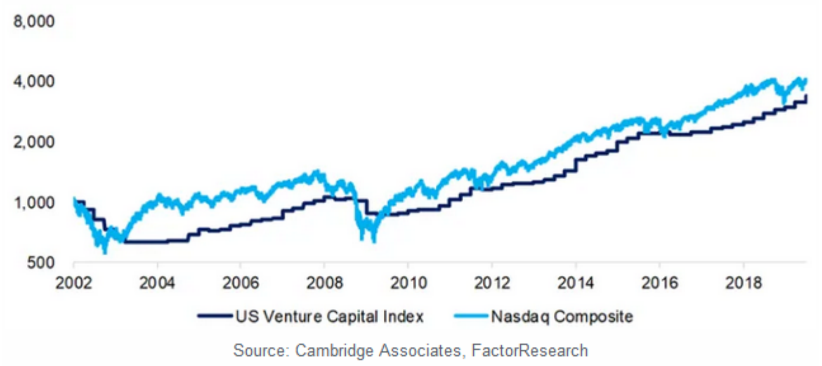
Given the risky nature of early stage companies, these investors expect some of their investments to fail. In fact, estimates for the success rate on investments by Venture Capital funds vary depending on the study. For example a review of data from Dow Jones VentureSource, as in Figure 1.1 taken from [3] estimates this at 20-30% (looking at returns above 3x gross multiples, which corresponds to annualised gross returns of 12% over 10 years and excludes management fees and other costs), while the *Harvard Business Review* [4] estimates this at 28.8%. This means that these funds take into account a large number of investments failing and because of this, the majority of returns must be generated from a small number of very successful investments, as is also evident from Figure 1.1.



Data from Dow Jones VentureSource and other primary and secondary sources

Figure 1.1: Distribution of Realized U.S. Venture Outcomes 2009-2018

A comparison of the NASDAQ Composite[2] and US Venture Capital Index[3] shows that returns generated by the average Venture Capital investor have not outperformed this public market benchmark, as shown in Figure 1.2 taken from [5].



Source: Cambridge Associates, FactorResearch

Figure 1.2: US Venture Capital Index vs. NASDAQ Composite, 2002-2019

[2] An index heavily weighted towards companies in the information technology sector

[3] An index calculated by Cambridge Associates

Research by [6] compared the annualised returns achieved in the private equity markets vs. returns on broad equity indices from 2006-2015 and concluded that both have returned ca. 11% p.a.. This raises the question of why investors should allocate funds to the private market segment, particularly Venture Capital fund managers, and serves as a source of motivation for improving the success rate on Venture Capital investments in order to deliver increased returns. In particular, Figure 1.2 shows that although startup investing is quite risky, with up to 64% of financings losing money, it also shows that gross returns above 1,000% are achieved on around 5% of financings (in the US). This highlights the large upside available to investors who are better able to identify and invest in successful startups. The increased availability of data on startups, as detailed in Table 2.1, serves as further motivation to pursue the development of quantitative models to better identify successful startups and in doing so enhance returns of Venture Capital portfolios.

1.2 Structure of Paper

We start by reviewing the existing research on the development of machine learning models for Private Market strategies and Venture Capital investments in Chapter 2. Although research on this topic can be found as early as 2001, more recent research focuses on using large-scale databases that have become available over the last 13 years. We cast a critical eye on the results of this research to understand whether these findings could be used in practice, as claims regarding the level of predictive ability that the various machine learning models have achieved appear impressive.

To understand how best to implement machine learning models for Venture Capital investing, in Chapter 3 we explore how Venture Capital investing works in practice and how returns are generated. To link this process to the intended models, we attempt to formalise this investment microstructure and from there derive the target variable we intend to predict. We then explore some of the mathematical details behind four machine learning models (logistic regression, random forest, support vector machines and extreme gradient boosting) to understand how best to calibrate them for our purposes, as well as to understand their strengths and weaknesses.

In Chapter 4 we explore the startup database (Crunchbase) to better understand the data that will be used in the machine learning models. We start by contextualising what this data represents and then aim to understand its reliability and any biases that might be present. We then provide a brief overview of the data and describe how it will be processed into a time series format.

In Chapter 5 we then describe how this time series format will be used to produce tests for how well machine learning models, that are trained on historical data, perform at predicting successful investment targets on future (unseen) data. The machine learning models are then implemented on a historical calibration/ future forecasting basis and evaluated using measures that take into account the desire of increasing Venture Capital portfolio returns by increasing the number of successful companies selected, while also taking into account the return generating process of these investments.

Chapter 2

Literature Review

2.1 Evolution of Data Availability for Venture Capital

The research of quantitative methods to predict company events, up until ca. 2015, relied mostly on self-created survey-data or bespoke data sets that were limited in size and scope. Large scale databases for researching this topic, particularly in a Venture Capital setting, have only been established in the last 13 years. Furthermore, these databases have only matured in scope and scale over the last 5 years, leading to a rise in research of more advanced quantitative methods on this topic. Before reviewing the literature, it is therefore helpful to first review some of these recently developed databases, as this will provide some context on how research has evolved over the last 20 odd years.

Company	**Started**	**Description**
Crunchbase	2007	Covers around 1 million companies globally (2020/06), with information on public and private companies, such as funding rounds, founders and employees, mergers and acquisitions and industry trends. Data is sourced from the venture program[1], machine learning (processing publicly available sources), in-house data teams and the Crunchbase community (crowdsourcing). Only limited data is publicly accessible, with *Pro* and *Enterprise* subscriptions required to access the data (apart from academic access).
PitchBook	2007	Covers around 3 million companies globally (2020/06), with information on public and private companies, such as funding rounds, founders and employees, mergers and acquisitions and industry trends. Data is sourced from machine learning (processing publicly available sources) and a large in-house data research team. Only limited data is publicly accessible, with various subscriptions available to access the data.
Dealroom	2013	Covers ca. 550,000 startups, scale ups and corporates (2020/06), with a more European focus. Information spans public and private companies, such as funding rounds, founders and employees, mergers and acquisitions and industry trends. Data is sourced from machine learning (processing publicly available sources), government sources and through crowdsourcing. Only limited data is publicly accessible, with various subscriptions available to access the data.

Table 2.1: Overview of StartUp Databases

Crunchbase is the most commonly used data source in recent research and is also the source used in this paper, so we will focus on understanding its content and structure - although from a review of sample data from Dealroom and PitchBook, these insights should also hold for those

[1] The venture program incentivises investors to keep their portfolio companies' Crunchbase profile information up-to-date in return for discounted product costs (source = CB and Techcrunch)

data sources. Crunchbase tailors their product for use in marketing, deal sourcing, fund raising and business development, with different company details updated at different times and stored in separate databases. This means the data is primarily set up in a way to present the user with a current snapshot of a company, rather than a time-series view. Fortunately, some time series information is stored across various databases so that with some effort, the data can be transformed into a comprehensive time-series view.

Given this context, we now review the literature covering the use of quantitative methods to predict company events. We group the research by the scale of data used, as this provides a natural chronlogical segmentation as well as some much needed context for the outcomes of the research.

2.2 Research Using Small Data Sets

Overview

2001, Lussier's paper [7] is one of the earliest and most cited research papers in this field. It looks at developing a logistic regression model to predict the success or failure of Croatian companies over a 3 year period. Success is defined as making at least the industry-average profits over the previous 3 years, while failure is defined as not making a profit over the same period. Data to develop the model was drawn from a manual survey by the author, resulting in 120 data points with 84 success and 36 failed outcomes. The logistic regression model is able to correctly classify 72% of the data points.

2003, Ragothaman et al [8] looks at predicting corporate acquisitions using Artificial Intelligence-based rule induction techniques. The authors use a total of 194 data points from COMPUSTAT, covering 97 companies that were acquired between 1994 to 1996 and 97 companies that were not acquired. Commercial software is then used to train a rule-based system labelled ACQTARGET, which is compared to standard statistical techniques such as multiple discriminant analysis and Logit analysis. They concluded that the ACQTARGET system performs in line with the standard statistical techniques in predicting the acquisition of companies.

2009, the paper [9] by Wei et al is one of the most cited subsequent research papers on this topic and looks at whether mergers and acquisitions, using data enriched with patent information, can be predicted using machine learning techniques. It is based on 61 mergers and acquisitions in Japan between 1997-2008, using data from Thomson Reuters Platinum database. 523 non-merger and acquisition data points are generated from the same 61 acquisition examples, as explained in [9, p 197]. After enriching the data from the United States Patent and Trademark Office (USPTO) they develop an ensemble learning algorithm to predict merger and acquisition outcomes, achieving an accuracy of 88% and precision of up to 42%.

2014, Yankov et al [10] evaluate the use of machine learning models to predict whether technology startups in Bulgaria will be successful, where success is defined by a startup surviving and increasing in size over a 5 year period. Data to develop the model was drawn from a manual survey by the authors, resulting in 142 data points on Bulgarian technology startups. Their models are able to correctly identify 83.76% of successful companies.

2019, Martinez [11] investigates the use of logistic regression models to predict 3 startup outcomes: 1) achieving funding above €1,000,000, 2) Employing more than 10 staff and 3) Achieving an annual investment IRR of at least 20%. The research uses 91 Dutch startup and scale ups founded between 2009-2017 identified from finleap.nl. Data on these 91 companies from the early stages of the company is sourced via questionnaire, while the outcome variables are taken from finleap.nl (implied to be 2019 values). Logistic models are developed on this data, achieving a predictive accuracy of 1) 71%, 2) 71% and 3) 76%.

Discussion

The above research provides insights into the drivers of success/ failure on small samples of startups and develops some quantitative methods to enhance the identification of merger and acquisition targets, as well as trying to classify some startup outcomes. The limited and specialised data used to derive results make broader applications of their findings in industries such as Venture Capital challenging, but the insights some of them provide form the basis of future research on this topic.

2.3 Research Using Large, Static Data

Overview

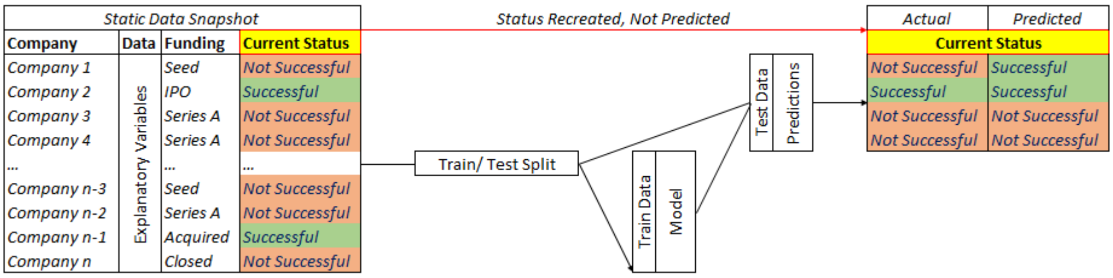
2012, the paper [12] by Xiang et al is the first published research to use data from Crunchbase to try and predict merger and acquisition activity using machine learning models. They use Crunchbase data, as at 2011/12, on 59,631 companies founded between 1970-2007 and enrich this with news coverage features for 10% of the companies. Machine learning models on this static data set are then developed to predict whether a company should be classified as acquired, achieving a True Positive and False Positive rate of between 60.0-79.8% and 0.0-8.3% respectively.

2017, Dellerman et al [13] develop a hybrid machine learning and collective intelligence prediction method. They source data on 1,500 technology startups with angel investments from Crunchbase, Mattermark[2] and Dealroom. A combination of machine learning and human classification models are developed and combined to predict whether these companies will reach Series A funding. No results on the success of this method are included in the paper.

2017, Bento [14] studies the use of machine learning models to classify 'successful' and 'not successful' outcomes for about 87,000 US companies present on Crunchbase on 2017/01/23, where 'successful' is defined as a company with an IPO or acquisition event and 'not-successful' is assigned to all other companies. The machine learning models are developed on the snapshot view as at 2017/01/23 and achieve a True Positive and False Positive rate of around 94% and 8% respectively.

Discussion

The above papers take the approach of developing models to recreate positive and negative company statuses (based on funding) using a static data snapshot, as per Figure 2.1.



Static Data Snapshot

Company	Data	Funding	Current Status
Company 1	Explanatory Variables	*Seed*	*Not Successful*
Company 2		*IPO*	*Successful*
Company 3		*Series A*	*Not Successful*
Company 4		*Series A*	*Not Successful*
...		...	...
Company n-3		*Seed*	*Not Successful*
Company n-2		*Series A*	*Not Successful*
Company n-1		*Acquired*	*Successful*
Company n		*Closed*	*Not Successful*

Actual	*Predicted*
Current Status	
Not Successful	*Successful*
Successful	*Successful*
Not Successful	*Not Successful*
Not Successful	*Not Successful*

Figure 2.1: Classification Models using a Static Data Approach

These results validate the ability of machine learning models to accurately categorise data, however they are backward looking and not aligned to the use-case that would be required in a commercial setting. In practice, these models need to be forward looking and give the user the ability to (A) act on the prediction (enter an investment) and (B) benefit from the predicted event (crystallise a return on an investment), as per Figure 2.2.

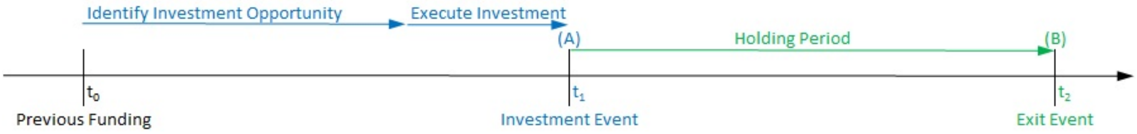


Figure 2.2: Return Generating Investment Process

As elaborated in Section 3.2, a quantitative model therefore needs to predict the 2 events (A) and (B) into the future.

[2] Mattermark is a similar offering to Crunchbase, Pitchbook and Dealroom, but was acquired and closed down in 2017. The company relaunched in 2019 and claims to cover 80 data points on over 4 million companies (2020/06)

2.4 Research Using Large, Time Series Data

Overview

2018, Hunter et al [15] develop a framework to model the behaviour of company funding events. They then create a Bayesian approach to building investment portfolios which maximise the probability of containing winners (company reaching either IPO or acquisition status). Data are drawn from Crunchbase up to 2016/06 and covers 24,000 US companies founded between 2000-2016. PitchBook and LinkedIn data are used to clean and enrich this data. Only companies which have seed or series A funding as the first funding event (baseline date) are selected to be eligible for the investment portfolio. The companies' data at the baseline date are then used to develop a model which maximises the probability of including winners in the portfolio.

2018, Sharchilev et al [16] develop machine learning models that try to predict if a company will have another funding round within a 1 year period. Data is drawn from Crunchbase and combined with additional features derived from crawling news sources. The data is structured as a rolling 30 day snapshot of all companies and news features up until 2017/05. Only companies which have angel or seed funding (trigger rounds) are included in the model development. Models are trained on rolling snapshots up until 2014/05 and tested on rolling snapshots from 2015/05 until 2016/05. They report precision levels, for the top-100 and top-200 companies their models identify, of 0.626 and 0.535 respectively. A visualisation of this is provided in Figure A.4.

2019, Arroyo et al [17] develop models to predict the future funding event of pre series C funded companies. Data on 120,507 global companies, founded between 2011/08 and 2015/08, are drawn from Crunchbase as at 2018/08 conditional on not being closed, reached IPO, been acquired or having achieved series C+ funding as at 2015/08. Only data that can be estimated for these companies as at 2015/08 are included in the development of models which aim to predict whether these companies will be closed, reach IPO, be acquired, achieve another funding round or have no event over the next 3 years (i.e. up until 2018/08). They report precision levels between 0.84-0.86 when predicting a binary outcome indicator consisting of *good* (funding round, acquired or IPO) and *bad* (no event and closed).

Discussion

It's assumed in [15] that investors will invest after the seed or series A funding round and then generate returns when these companies are acquired or have an IPO. Reviewing the Crunchbase data [18] for US companies with seed or series A funding in 2011 and 2012, we can see that of these 5,590 companies, 739 will be acquired or have an IPO, *but* 291 of these have the acquisition or IPO events immediately after seed or series A funding (283 acquisitions, 8 IPOs). As such, a Venture Capital investor wouldn't be able to invest in about 38% of the companies before the return event occurs, which isn't factored into the analysis and so overstates the degree to which the method is able to construct portfolios containing winners. In fact, 8 out of 18 winners from the portfolios in [15, p. 36-39] are acquired immediately after seed or series A funding (see Table A.2).

The stated aim in [16, p 7, Section 3] is: "for a given startup that has received seed or angel funding, predict whether it will secure a further Series A or larger round of funding during the next year". A particular shortcoming with this approach, in context of Figure 2.2, is that in order for investors to benefit from this prediction they would already need to be investors in the seed or angel rounds (potentially crystallising a return when the next funding event occurs). Alternatively, investors could benefit from this approach if a company with angel or seed investing has more than one subsequent round of funding, which isn't factored into their analysis. Analogous to the analysis of winners in the portfolios of [15], predicting a next round of funding isn't a sufficient condition for investors to profit from the model's predictions. The impact of these points is difficult to assess and so we refrain from drawing any further conclusions.

The aim of [17] is to predict a broader set of events to assist Venture Capital investors in deciding which companies to analyse in more detail. One shortcoming in their approach is that the model doesn't provide insights to the user whether there is an entry and exit point available for the Venture Capital investor (as per Figure 2.2) to generate a return. As we've seen in the discussion of other research, being able to predict future outcomes for companies such as IPO, acquisition or achieving another funding does not mean the investor has a chance to enter before the return generating event occurs. The impact of these points is difficult to assess and so we refrain from drawing any further conclusions.

Chapter 3

Quantitative Models for Venture Capital

We'll start by looking at existing Venture Capital investment practices and then provide an overview of the funding terminology before moving on to discussing the processes that create returns for investors. This will then be linked to what we want to predict with models, how it will be reflected in the data and how it will be encoded for prediction. Finally an overview of the models that will be used to make predictions will be provided along with a summary of their theoretical background.

3.1 Venture Capital Investing

A review of the available literature did not yield any definitive studies on how Venture Capital investors currently make their decisions or whether there is any trend towards using more quantitative methods to improve these processes. A handful of blogs[1] and interviews with Venture Capital investors[2] suggest that quantitative methods methods have not yet been widely adopted and remain a niche operating model within the industry.

A 2016 survey of 885 Venture Capital investors, covering 681 firms [19] looked at what the most important factors are for Venture Capital investors when assessing an investment target. Perhaps somewhat surprisingly, 47% ranked the management/ founding team as *the most important factor* when deciding whether to invest compared to 10%, 13% and 8% for the business model, product and market, respectively. Similarly, 95% of firms identified the management/ founding team as an *important factor* when deciding whether to invest compared to 83%, 74% and 68% for the business model, product and market, respectively [19, Table 5, p. 178]. Furthermore, when comparing the importance of deal sourcing, investment selection and value-add, 49% of Venture Capital investors ranked investment selection as *the most important item*, while only 27% and 23% rank value-add and deal flow as *the most important item* [19, Table 13, p. 187]. These tables are reproduced in Figures A.5 and A.6 respectively.

Hence, given the availability of people-related data on the databases listed in Table 2.1, this bodes well for quantitative models being able to predict successful startups (even without company specific data) and further motivates their use to improve the deal selection process.

3.2 Venture Capital Market Microstructure

Chapter 2 showed how some quantitative models, using larger startup databases, have been developed over the last few years. Most of this research has focused on developing models that process these new data sources into making some sort of prediction on companies' funding statuses. However, one of the key elements missing from this research is the consideration of how investments are executed and how returns are generated. In other words, the market microstructure of Venture

[1] https://medium.com/iveyfintechclub/machine-learning-and-big-data-in-private-equity-is-networking-still-needed-9f8912a61ee9

[2] https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/a-machine-learning-approach-to-venture-capital#

Capital investing doesn't seem to have been factored into the development of quantitative models so far.

Market microstructure plays an important role in the development and implementation of trading strategies. For example, in public markets, if the market is order-driven or quote-driven changes the way trading strategies are created to take into account the way that investments are entered and exited, as these microstructures impact the price and execution timing of investments. Another consideration is that market microstructures are constantly evolving in response to regulation, participant demands and competition, which also needs to be taken into account.

Private market microstructures in particular also evolve over time. For example, within the Venture Capital segment, the introduction of the Jumpstart Our Business Startups (JOBS) Act 2012 in the US gave rise to electronic marketing and investing venues for early stage companies for retail investors - also known as crowdfunding sites. Equity crowdfunding[3] is a small proportion of total funding round activity (making up 3% of volume and 0.3% of value across 2017-2019 in the USA and UK, based on Crunchbase data), but evolving and growing over time. It also presents new ways for companies to raise funds, but also presents new ways for investors to enter and exit investments.

Funding Rounds

Before discussing the Venture Capital investment process, we look at some of the terminology associated with the funding events. Investing is typically categorised into a number of stages, with each category further subdivided into a funding series. These reflect the funding requirements given the growth stage of the company and can also be used as a proxy for the riskiness of the investment. An overview of these funding rounds, compiled from [20] and [21], is provided in Table 3.1 and provides further insights into the microstructure of Venture Capital investing.

Funding Stage	Round	Investors	Description
Pre-seed	*Pre-seed*	Early Stage Angels Angels Early Stage VC Accelerators Friends and Family Startup Accelerators	Pre-institutional, pre-product funding that is a very low amount, often below $150k.
Seed	*Seed*	Angels Early Stage VC Accelerators	Among the first funding rounds, ranging from $10k-2m, used to develop company's market traction.
Early Stage	*Series A*	VCs Super Angels	Funding usually ranging from $1-15m that targets revenue growth and increased marketing. The *series* rounds naming is aligned to the type of preferred stock that investors receive.
Early-Mid Stage	*Series B*	VCs Late Stage VCs	Funding to help the company scale, usually ranging from $5-35m.
Mid-Late Stage	*Series C+*	Late Stage VCs Buyout Funds (PE) Hedge Funds Banks	Later stage funding to help company scale operations further, usually ranging from $20-300m

Table 3.1: Overview of Startup Funding by Investment Stage, Funding Rounds and Investor Type

Table 3.1 shows how investors overlap in funding stages, but also that Venture Capital funds tend to specialise in particular funding rounds. As startup companies grow and progress to subsequent funding stages, the ultimate goal for Venture Capital investors is to realise a return from their investment, which is referred to as the *exit strategy* (or *exit event*).

[3] Crowdfunding can also be structured as debt. Crunchbase data covers only equity crowdfunding

Venture Capital Investment Process

We will now try to formalise the microstructure for Venture Capital investments, which will form the basis on how various outcomes are represented in data and finally use this to formulate a target variable for quantitative models. The microstructure we assume to hold is:

1. Investments into early stage companies (funding rounds) happen at discrete time intervals.
2. When an early stage company receives a funding round, the investors commit capital and so enter into the *initial* investment.
3. There are (generally) no liquid markets investors can use to exit the *initial* investment, hence they must wait for a secondary event, i.e. the *exit event* in order to crystallize returns. *Exit events* include[4]:
   (a) Initial Public Offering (IPO): The startup's shares start trading on a public exchange and so provide liquidity for investors to sell all or part of their shares acquired in the pre-IPO funding rounds. The assumption is that the value of the initial investment has increased.
   (b) Strategic Acquisition: The startup's shares are acquired by a company, which provides an opportunity for investors to sell all or part of the shares they acquired prior to the acquisition. The assumption is that the value of the initial investment has increased.
   (c) Sale of Shares (Other): While IPOs and strategic acquisitions are the most common exit routes, investors can exit investments by selling their shares directly to another investor (direct placement) or by selling their shares to the company's management (management buyout). The change in value in these instances is unclear, however for simplicity we assume that the value of the initial investment has increased.
   (d) Additional Funding Round: Although not typically listed as an *exit event*, most startups only receive additional funding if they are successful. [22, p. 8] shows that 70-80% of startups receive a higher valuation in the next funding round (known as an *up round*), while data from Fenwick & West[5] shows that valuations increase between funding rounds on average by 61% (median 32%). Thus we can reasonably assume that the value of the initial investment has increased and investors have made a (theoretical) gain.
   (e) Company Failure: The company fails and it's assumed that the value of the initial investments drops to 0. We will refer to this as a negative *exit event*, as at this point investors typically write-off the value of their shares and the investment is closed out.

Targeted Outcome (Variable)

Under this microstructure framework quantitative models should predict whether a company will have an *initial* investment event in order for an investor to acquire shares in the startup (Point 2) and a positive *exit event* (Points 3a-3d). The positive *exit event* requirement can be interpreted in the stricter sense of looking only at *exit events* 3a-3c or in a more lenient sense by looking for *exit events* 3a-3d. While the stricter *exit event* options are more in line with the traditional guidelines for successful Venture Capital investments, we will also look also look at the somewhat more lenient definition. Doing this will not only provide insights into the differences in predictive ability for each definition, but will also allow us to better place the results from this research into the context of existing research as discussed in Section 5.4.

This 2 step prediction approach contrasts to the general approach taken in quantitative models in public markets, which, in oversimplified terms, aim to predict a single future event for an asset (e.g. price increase, flat, decrease). This means public market strategies generally aim to predict what is analogous to the *initial* investment (Point 2), as the assumption is that the same asset can be invested in prior to this future event via a market order at the current price. Both of these processes are represented in Figure 3.1 for comparison.

At this point, one could make the argument that after the *initial* event, active management by Venture Capital investors is a contributing factor to a positive *exit event* and so influences the data that is used for predictive purposes. In other words, one can question how a model should

[4] We exclude instances where companies continue operating without additional funding, as this is not an opportunity for investors to exit and the change in value for the investor is unclear

[5] https://www.fenwick.com/insights/publications/silicon-valley-venture-capital-survey-second-quarter-2020

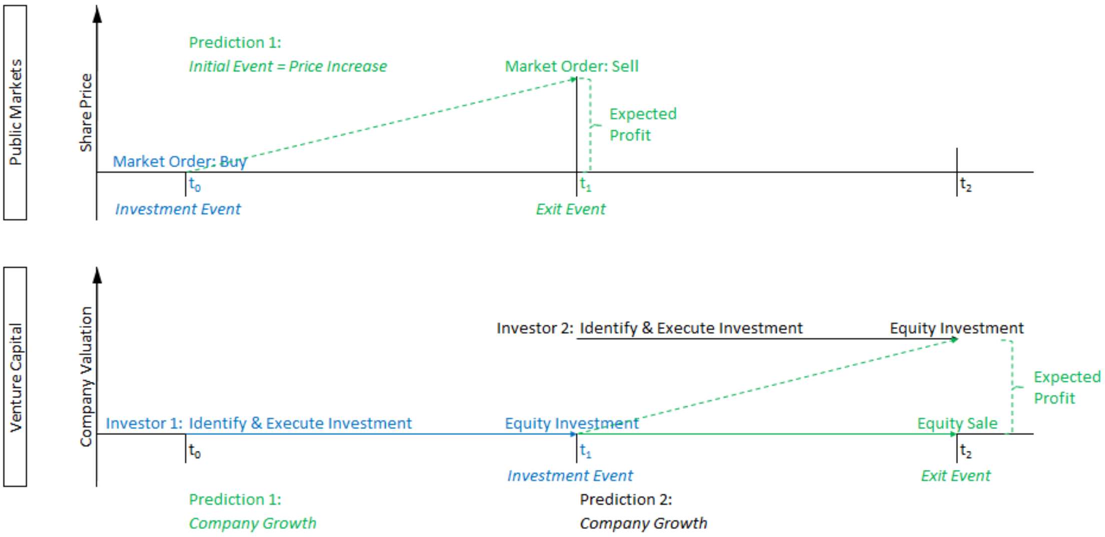


Figure 3.1: Public Market and Venture Capital Investment Microstructure

predict 2 company events into the future, when the outcome of the *initial* investment could alter the outcome of the *exit event*. This means one would ideally first model how an *initial* investment changes the data of the company under consideration and then use these predicted data to forecast whether this will result in a positive *exit event*. Taking this approach would potentially produce models with better predictive ability and model insights for investors.

However, given the limitations of the data available we will take a more simplistic approach. We will focus on developing models to predict companies which have an *initial* investment opportunity and a positive *exit event* using the company data as it was before both of these events occurred, assuming that all Venture Capital investors are equally able to guide a company from an *initial* investment to a successful *exit event*. Given the discussion in 3.1, which showed that the majority of Venture Capital investors rank the importance of the management/ founding team and the investment selection as the most important factor when assessing an investment target, using the people related data from Crunchbase should provide a reasonable basis for the development of quantitative models for this purpose. Given this aim, the target variable will take on the stricter form in Table 3.2 or the more lenient form in Table 3.3.

Current Funding Status	**Next Event**	**Final Event**	**Target Variable**
Seed or Series A+ (excl. exit events)	Series A+ (excl. exit events)	Events listed in 3a-3c	1
Seed or Series A+ (excl. exit events)	Any (e.g. funding, exit, etc.) or no event	None	0

Table 3.2: Strict Target Variable Formulation

Current Funding Status	**Next Event**	**Subsequent Event**	**Target Variable**
Seed or Series A+ (excl. exit events)	Series A+ (excl. exit events)	Events listed in 3a-3d	1
Seed or Series A+ (excl. exit events)	Any (e.g. funding, exit, etc.) or no event	None	0

Table 3.3: Lenient Target Variable Formulation

Further details on the use of the target variable (dependent variable) is provided in Section 3.3.

3.3 Venture Capital Model Development

With the market microstructure for Venture Capital firms in mind, we now turn to the models that can be used to predict whether a company will have both an *initial* investment and a positive *exit event.* This sequence of events can be encoded in many different ways, however we will look at a binary variable indicating whether both events have happened (1) or not (0). The four machine learning models developed to predict the target variable are: logistic regression (LR), random forest (RF), support vector machines (SVM) and extreme gradient boosting (XGB). LR results provide a baseline for performance, given its relative simplicity compared to the other models.

3.3.1 Logistic Regression

Logistic regression is used to model the probability of the outcome of a categorical response variable (i.e. target or dependent variable). In contrast to linear regression, it is a nonparametric technique and so doesn't make any assumptions about the distribution of the data or the residuals (errors) of the fitted model. Focusing on the simpler case, where Y is a binary response variable (assuming outcomes of 0 or 1) and writing p = probability($Y = 1$) given X, we then have the *odds* of the event $Y = 1$ given by:

$$\text{odds}(Y = 1) = \frac{p}{1 - p}, \text{where } 0 < p < 1$$

Taking the logarithm of the odds (typically using base e), yields the **logi**stic un**it** measurement (aka logit). Logistic regression assumes a linear relationship between the logit of the dependent response variable Y and the K independent variables $\boldsymbol{X}$. The equation describing this relationship, as per [23, p. 14], is given by (3.3.1).

$$\begin{aligned} \text{logit}(Y) &= \alpha + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_K X_K \\ &= \alpha + \boldsymbol{\beta}^T \boldsymbol{X} \end{aligned} \tag{3.3.1}$$

The relationship described in (3.3.1) is fit by maximising the log likelihood of the observations. Assuming observations x_i and responses y_i where $i = 1, ..., N$, the log-likelihood is given by:

$$l(\alpha, \boldsymbol{\beta}) = \sum_{i=1}^{N} \{y_i \log(p) + (1 - y_i) \log(1 - p)\} \tag{3.3.2}$$

where p is a function of $\alpha, \boldsymbol{\beta}$ and $\boldsymbol{X}$. Once the maximum log likelihood parameters have been found, the estimated probability $\hat{p}$ can be calculated by reversing the logit-unit transformation, yielding:

$$\begin{aligned} \hat{p} &= \frac{\mathrm{e}^{\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{X}}}{1 + \mathrm{e}^{\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{X}}} \\ &= \frac{1}{1 + \mathrm{e}^{-(\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{X}})} \end{aligned} \tag{3.3.3}$$

The logistic function has a characteristic S-shape, such that probabilities close to 0/ 1 are associated with their corresponding 0/ 1 response. The exact point, or *decision point*, to use for classifying the predicted probabilities from (3.3.3) as either 0 or 1 depends on the aim of the exercise, however a default value of 0.5 is generally used.

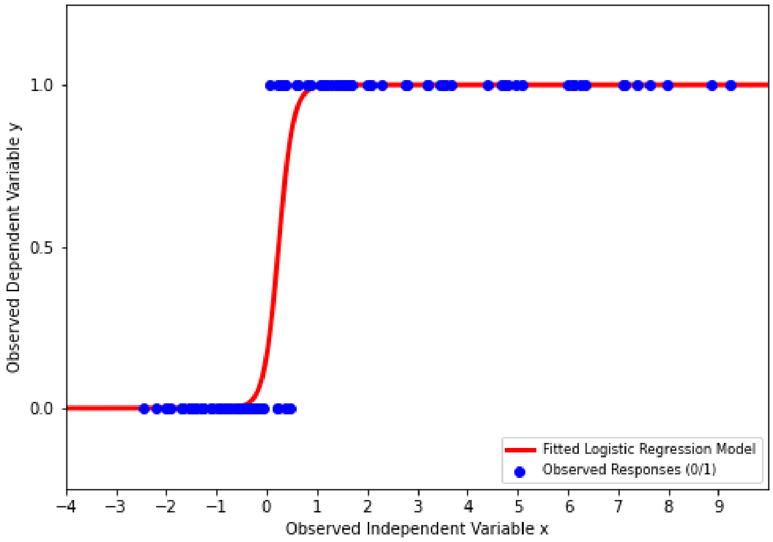


Figure 3.2: Example of Logistic Function with Toy Data

The relationship between the logit(Y) and the independent variables can be extended to cover curvilinear interactions between the variables $X_1, X_2, ..., X_K$, increasing the complexity of both the model and the relationships that might exist in the data (see [24, p. 275]). However, for the purpose of establishing a baseline value for the various machine learning models in this paper, we will consider the linear relationship given by (3.3.1).

3.3.2 Random Forest

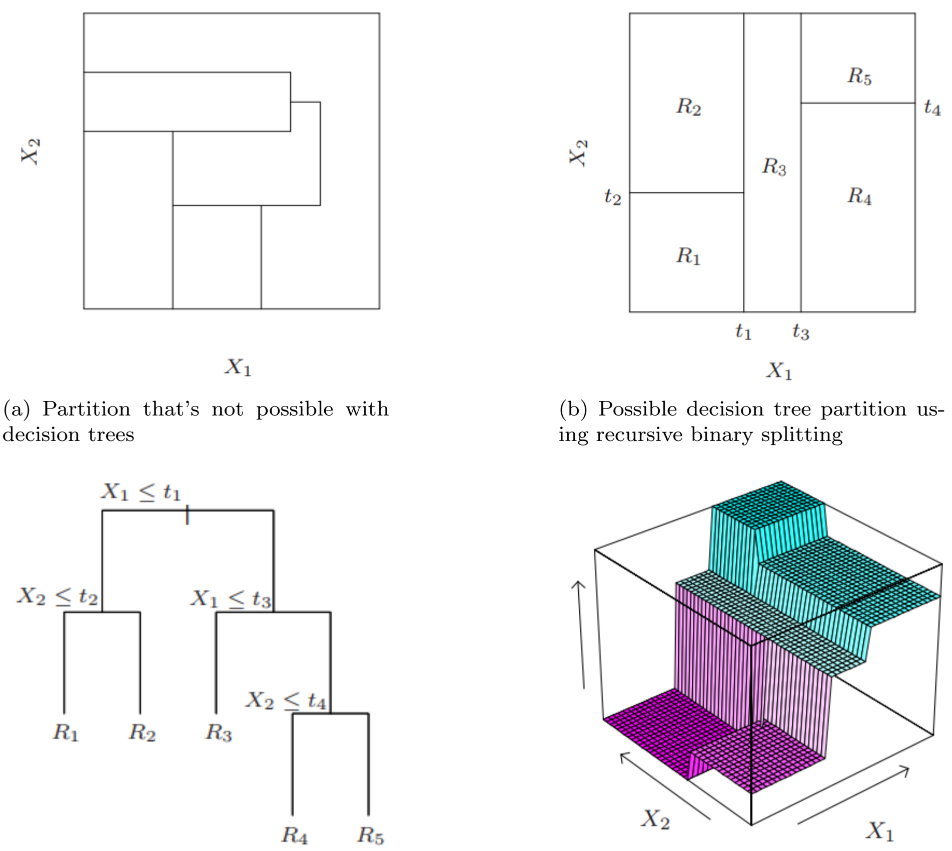
The basis of the random forest (RF) algorithm is decision trees, which take on the form of either regression or classification trees. Given the aim of this paper and the format of the dependent variable, we will present the categorical classification tree algorithms, although the same algorithm can handle both categorical and numerical data both as inputs and outputs. We will start with a brief background on tree-based methods before analysing the classification tree algorithm. Subsequently, we will discuss the advantages and disadvantages of this approach before looking at the random forest algorithm as a way to improve on these drawbacks.

Tree-Based Methods

”Tree-based methods partition the feature space into a set of rectangles, and then fit a simple model (like a constant) in each one.” [25, p. 305]. These rectangular regions however, can only be created through recursive binary partitions which places some restrictions on the way the regions are defined. This concept is best illustrated visually, using a hypothetical example with 2 independent variables X_1 and X_2, with partitions at $t_1 - t_4$ as per the example [25, p. 306, Figure 9.2], reproduced in Figure 3.3.

What this means is that we look for groupings (’rectangles’ or ’regions’) of data points for which the best response is their most frequent response[6]. These regions are created by taking each independent variable and sequentially introducing a binary split into their values to see which split best separates the responses contained within the dependent variables. The formulation of this sequence of splits of the independent variable values and how to define what the ’best’ separation of responses in the dependent variable is, is essence of the Classification Tree algorithm. The terminology used for Classification Trees is provided in Table 3.4.

[6] This is not the only definition of the ’best’ response for categorical and numerical dependent variables



(a) Partition that's not possible with decision trees

(b) Possible decision tree partition using recursive binary splitting

(c) Corresponding decision tree for Figure 3.3 (b), creating 5 rectangular regions $R_1 - R_5$

(d) 3 dimensional representation of the regions in Figure 3.3 (b)

Figure 3.3: Illustrative Example of a Decision Tree and the Regions

Term	Meaning
Branch	A binary partition of the data (e.g. $X_1 \leq t_1$ and $X_1 > t_1$), indicated by two lines leading away from a root node or decision node
Root Node	Initial partition of the tree, corresponding to $X_1 \leq t_1$ in Figure 3.3 (c). If a root node does not have any branches (i.e. does not partition the data), it is referred to simply as a leaf.
Decision Node	Subsequent partition points within the tree, corresponding to $X_2 \leq t_2, X_1 \leq t_3$ and $X_2 \leq t_4$ in Figure 3.3 (c). A decision node must have branches leading away from it, otherwise it is a leaf.
Split Point	The values used by the root node and decision nodes to partition the data, corresponding to the values $t_1 - t_4$ in Figure 3.3 (c)
Leaf	A terminal point where a node does not split, indicated by no branches leading away from it. E.g. $R_1 - R_5$ in Figure 3.3 (c)
Stump	A tree with only a root node and 2 leaves is a stump
Pruning	Pruning reduces overfitting of decision trees by removing leaves and decision nodes starting from the bottom of the tree (e.g. R_4 and R_5 in Figure 3.3 (c)) and moving upwards. Pruning removes a decision node if the corresponding leaves don't meet a certain criteria, such as achieving a minimum decrease in gini impurity or other penalty functions that take the number of leaves into account (cost complexity pruning). In an extreme scenario, pruning can remove all leaves in a tree leaving only the root node.

Table 3.4: Tree-based Methods Terminology

Classification Trees

[25, Chapter 9.2], [26, Chapter 8] and [27] cover this topic in various degrees of detail, so we will borrow from their approaches and notation to cover how the Classification Tree algorithm quantifies and selects the root node, decision node, split points and leaves.

Let $\boldsymbol{x_i} = (x_{i1}, x_{i2}, ..., x_{ip})$ represent the i^{th} observation vector of p independent variables and y_i represent the observation of the i^{th} dependent variable. For $i = 1, ..., N$ observations we end up with a matrix of observed independent and dependent variables. While the independent variables x_i can be either continuous or categorical data, the dependent variable y_i in this case is categorical. The aim is now to find $M \geq 1$ regions $R_1, ..., R_M$, with corresponding number of observations $N_1, ..., N_M$, into which to partition the data, so that for each of these regions the response variable is a constant value $c_m, m \in [1, ..., M]$. The resulting model $f(\boldsymbol{x})$ for responses from each of these M regions can then be written as:

$$f(\boldsymbol{x}) = \sum_{m=1}^{M} c_m \mathbb{1}_{\boldsymbol{x} \in R_m}$$

Given that most data sets contain numerous independent variables, which can take on many more values on which to partition the data, it is often computationally infeasible to consider all possible data partitions to compute the global optimal partitioning. Therefore, this algorithm is implemented using a greedy algorithm, which selects the node split point by only considering what the 'best' possible data partition is at current node under consideration. With a categorical dependent variable y, the quantitative measure for the 'best' possible split is often based on the Gini impurity, although other measures such as the misclassification error and cross-entropy loss are also considered. The Gini impurity is a measure of the proportion $\hat{p}_{m,k}$ of correctly and incorrectly classified dependent variables for each partition under consideration, where k represents the number of categories the dependent variable y can take. The formal definitions are:

$$\hat{p}_{m,k} = \frac{1}{N_m} \sum_{\boldsymbol{x_i} \in \boldsymbol{R_m}} \mathbb{1}_{y_i = k}$$

$$\text{Gini Impurity} = \sum_{k=1}^{K} \hat{p}_{m,k}(1 - \hat{p}_{m,k})$$

The greedy algorithm to find the best binary partition of the data to produce the regions R_m starts by finding the initial partition of the data (i.e. the root of the tree). To do this, each of the independent variables $x_{i,j}, j \in [1, ..., p]$ (subsequently denoted x_j) are considered sequentially and for each independent variable, the best splitting point s (categorical or numerical) to determine the half-planes is determined.

The half-planes at the splitting point s, producing leaves a and b, are written as:

$$\begin{aligned} R_a(j, s) &= \{x | x_j \leq s\} \\ R_b(j, s) &= \{x | x_j > s\} \end{aligned} \tag{3.3.4}$$

Since each potential half-plane in (3.3.4) will have its own impurity, the overall Gini impurity for any proposed splitting value s is calculated as the weighted average Gini impurity of the half-planes (weighted on the proportion of observations per leaf). To determine the splitting value s, the overall Gini impurity is calculated for each splitting value s within x_j and the minimum is selected. Finally, the lowest overall Gini impurity for independent variables x_j is selected as the splitting point for the root of the tree. The same procedure is repeated to identify subsequent decision nodes until a stopping criteria is reached - such as a minimum number of observations within a region. This will produce M leaves, each of which is a decision node where a stopping criteria was reached, or where no further split of the data is possible. Each of the regions (3.3.4) defined by the M leaves will then be labelled as the region $R_m, m \in [1, ..., M]$. This procedure is summarised in Algorithm 1.

After a classification tree has been constructed, it generally contains a large number of decision nodes and leaves, which often leads to overfitting but can be managed through pruning.

Algorithm 1 Classification Tree (using Gini Impurity)

```
Set stopping criteria (e.g. node from binary splitting has less than specified number of observations)
Create Classification Tree():
    while stopping criteria not reached do
        Perform Binary Data Split on lowest Decision Nodes
        for pre-binary split region $R(j)$ and post-binary split regions $R_a(j,s)$ and $R_b(j,s)$ do
            Calculate Gini impurities
        end
        if Gini impurity pre-binary split $\leq$ Weighted Gini impurity post-binary split then
            Make pre-binary split region a leaf
        else
            Split region by independent variable $j$ and splitting point $s$
        end
    end
End Create

Function Binary Data Split():
    for each independent variable $j$ do
        for each splitting point $s$ do
            Calculate Gini impurity
        end
        Select splitting point $s$ yielding lowest Gini impurity
    end
Return independent variable $x_j$ and splitting point $s$ with lowest Gini impurity
```

Pros and Cons of Classification Trees

We summarise the discussion on these models from [26, Chapter 8.1.4]:

+ Classification trees are considered 'white box' models, meaning the model can be interpreted and explained in meaningful ways.

+ The models are well suited for both quantitative and qualitative variables (e.g. categorical and numerical measures)

+ Inputs for tree methods are not affected by the scale of input variables, reducing errors that can occur due to normalisation or a lack thereof.

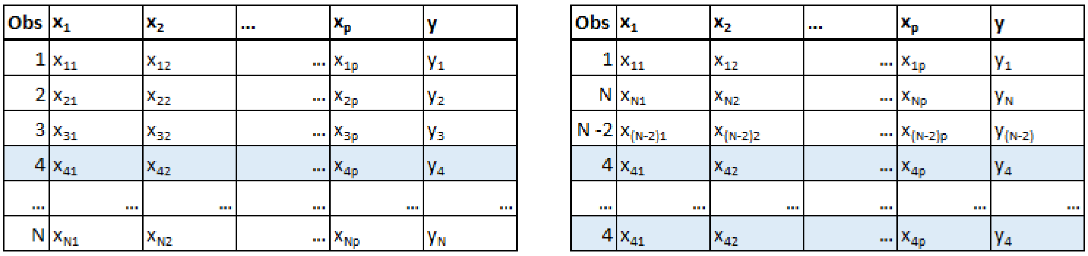
– Classification trees are prone to overfitting, which generally leads to poor performance on validation or other unseen data. Consequently, these tree models are often unstable and exhibit high variance, so that a small change in the data used to construct the tree can cause large changes in the tree model it produces.

– In a number of instances, trees have been found to have less predictive accuracy than other models, such as simple regression.

Random Forest (RF) Algorithm

The random forest algorithm addresses most of the drawbacks of classification trees while only sacrificing some of the interpretability. The algorithm starts by creating what is called a *bootstrapped* dataset. If the original data set contains N observations for p dependent variables $x_{i,j}$, where $i \in [1, N]$ and $j \in [1, p]$, the bootstrapped data set is created by randomly selecting, with replacement, N rows from the original data set. The matrix representations of the original and bootstrapped data can be represented as Figure 3.4:

Next, a classification tree T, as per Algorithm 1 is constructed with the bootstrapped data set, with the modification that instead of using all independent variables j, only a random selection of these j independent variables is used to determine the variable and splitting point for a decision node, leading to a `Modified Binary Data Split`. As per the recommendations of [25, Chapter 15.3], the default number of independent variables to use in the random forest algorithm for the `Modified Binary Data Split` is $m = \lfloor \sqrt{p} \rfloor$, although m is often treated as a tuning parameter that should be optimised.



Obs	x_1	x_2	...	x_p	y
1	x_{11}	x_{12}	...	x_{1p}	y_1
2	x_{21}	x_{22}	...	x_{2p}	y_2
3	x_{31}	x_{32}	...	x_{3p}	y_3
4	x_{41}	x_{42}	...	x_{4p}	y_4
...	...	...	...	...	...
N	x_{N1}	x_{N2}	...	x_{Np}	y_N

(a) Original Data Set

Obs	x_1	x_2	...	x_p	y
1	x_{11}	x_{12}	...	x_{1p}	y_1
N	x_{N1}	x_{N2}	...	x_{Np}	y_N
N -2	$x_{(N-2)1}$	$x_{(N-2)2}$	...	$x_{(N-2)p}$	$y_{(N-2)}$
4	x_{41}	x_{42}	...	x_{4p}	y_4
...	...	...	...	...	...
4	x_{41}	x_{42}	...	x_{4p}	y_4

(b) Bootstrapped Data Set

Figure 3.4: Illustration of Data Bootstrapping

This process is repeated B times to produce an *ensemble* of trees $\{T_b\}_1^B$, each of which produces a prediction of the dependent (target) variable y using the independent variables x_i. The predicted values from this *ensemble* of trees are then tallied and the predicted value that occurs most often is used as the random forest output - this is called the *majority voting rule*. The summary of this algorithm, adapted from [25, Algorithm 15.1, p. 588] is provided in Algorithm 2.

Algorithm 2 Random Forest

```
Set stopping criteria
Create Ensemble $\{T_b\}_1^B$():
    for $b = 1$ to $B$ do
        Create bootstrap data set $\mathbf{Z}$
        Build Classification Tree $T_b$ using Algorithm 1 with data set $\mathbf{Z}$ and Modified Binary
        Data Split
    end
End Create

Prediction Using Ensemble $\{T_b\}_1^B$():
    for $b = 1$ to $B$ do
        Let $\hat{C}_b(\boldsymbol{x_i})$ be predicted class of tree $T_b$
    end
    Final prediction $\hat{C}_{RF}^B(\boldsymbol{x_i}) = majority\ vote\ \{\hat{C}_b(\boldsymbol{x_i})\}_1^B$
Return $\hat{C}_{RF}^B(\boldsymbol{x_i})$

Function Modified Binary Data Split():
    Set $m = \lfloor \sqrt{p} \rfloor$
    for $m$ randomly selected independent variables do
        for each splitting point $s$ do
            Calculate Gini impurity
        end
        Select splitting point $s$ yielding lowest Gini impurity
    end
Return independent variable $x_j$ and splitting point $s$ with lowest Gini impurity
```

An analysis on the performance of the random forest algorithm vs. classification trees (and other variants) is contained in [25, Chapter 15, p. 587]. It highlights that the advantages of this algorithm lie in the way that the random selection of independent variables to grow the trees reduces the variance of the resulting prediction. Intuitively, this means that allowing trees to grow with different roots and decision nodes in each iteration builds an ensemble of trees, which collectively contains better approximations of what the optimal decision tree should be.

3.3.3 Support Vector Machines (SVM)

SVM is a classification method that develops non-linear classification boundaries. It works best with categorical dependent variables, in particular dependent variables with a binary response ($K = 2$), although this can be extended to $K > 2$ outcomes.

Similar to the random forest algorithm, SVM is based on simpler concepts known as *maximal margin classifiers* and subsequently *support vector classifiers*. The weaknesses of these simpler methods is their inflexibility to deal with complex separation boundaries, which the SVM approach overcomes by including higher-dimensional representations of the original data set in its estimation of these boundaries. We will therefore start by looking at *maximal margin classifiers* and *support vector classifiers* before discussing their advantages and disadvantages. Based on this understanding, we will then look at the SVM approach and how this overcomes these disadvantages.

Maximal Margin Classifiers (MMC) and Support Vector Classifiers (SVC)

Starting with the formal definition of a hyperplane, [26, p. 338] states that "[i]n a p-dimensional space, a hyperplane is a flat affine subspace of dimension $p-1$" and has the mathematical definition:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p = 0 \tag{3.3.5}$$

In other words, any point $(X_1, X_2, ..., X_p)^T$ in the p-dimensional space which satisfies (3.3.5) is a point **on** the hyperplane. This definition means that any point X^* **not** on the hyperplane falls into one of two categories:

$$\begin{cases} \beta_0 + \beta_1 X_1^* + \beta_2 X_2^* + ... + \beta_p X_p^* & > 0 \qquad \text{or} \\ \beta_0 + \beta_1 X_1^* + \beta_2 X_2^* + ... + \beta_p X_p^* & < 0 \end{cases} \tag{3.3.6}$$

If we consider trying to classify a categorical dependent variable $y \in \{-1, 1\}$, then this feature of hyperplanes can be exploited to establish boundaries within the independent variables $\boldsymbol{x_i} = (x_{i1}, x_{i2}, ..., x_{ip})$ for $i = 1, ..., N$. How to establish this boundary is first addressed by maximal margin classifiers (MMC).

Starting with any hyperplane that separates the data, we can calculate the perpendicular distances between the observed data points and the hyperplane. MMC then sets the hyperplane parameters $\boldsymbol{\beta}$ so that the perpendicular distances between the hyperplane and the observed data is maximised. Hence using this MMC-hyperplane, any observation x^* can be classified using (3.3.6) by taking the sign $y^* \in \{-1, 1\}$ from (3.3.7).

$$y^* = \text{sign}(f(x^*) = \text{sign}(\beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + ... + \beta_p x_p^*) \tag{3.3.7}$$

Another approach to establishing the MMC is by looking at the *margin*, which is the smallest distance between a separating hyperplane and the observed data. The MMC is constructed to maximise the *margin* between the hyperplane and the observed data. This leads to the conclusion that the only data points which matter in MMC are those which are involved in maximising the *margin*. These data points are referred to as *support vectors*. These concepts are best illustrated using an example, such as the one provided in [26, Figure 9.3, p. 342], which is reproduced in Figure 3.5 and illustrates the MMC separating hyperplane, the *margin* and the *support vectors* in a 2-dimensional space (X_1, X_2).

MMC is useful when data can be perfectly separated by a hyperplane, however in most instances this will not be the case. Support vector classifiers (SVC) build on the ideas of MMC and add flexibility to the linear separating hyperplane construction by allowing for some misclassifications. To understand how this is achieved in SVC, we first look at the formal definition on how to construct the MMC, given in (3.3.8)-(3.3.10) as per [26, p. 343].

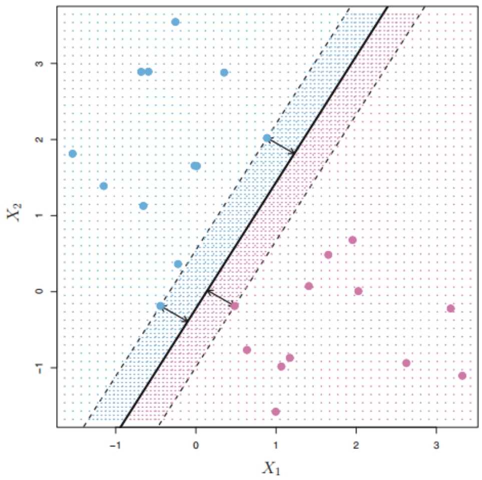


Figure 3.5: Example illustrating the MMC hyperplane as the solid black line, the *margin* as distance between the MMC hyperplane and dashed lines and the *support vectors* as the 1 purple and 2 blue dots that lie on the dashed lines

$$\max_{(\beta_0,\beta_1,\ldots\beta_p,M)} M \tag{3.3.8}$$

$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 = 1 \tag{3.3.9}$$

$$y_i^*(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) \geq M \quad \forall i = 1, ..., N \tag{3.3.10}$$

Since $y_i^*(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) > 0$, which is evident from (3.3.6) and (3.3.7), condition (3.3.10) ensures data is correctly classified under MMC. In contrast, SVC introduces *slack variables* ϵ_i into (3.3.10) that allow some observations to be misclassified (if needed), so that the formal definition for SVC is given in (3.3.11)-(3.3.14) as per [26, p. 346].

$$\max_{(\beta_0,\beta_1,\ldots\beta_p,\epsilon_0,\ldots,\epsilon_N,M)} M \tag{3.3.11}$$

$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 = 1 \tag{3.3.12}$$

$$y_i^*(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i = 1, ..., N \tag{3.3.13}$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{N} \epsilon_i \leq C, \quad \text{where } C \geq 0 \tag{3.3.14}$$

C in (3.3.14) is a tuning parameter and SVC again aims to find a linear separating hyperplane, while allowing for some misclassification of data (via the slack variables in (3.3.14)). To better understand the role of the slack variables, their interpretation is provided in (3.3.15).

$$\epsilon_i \begin{cases} > 1: & \text{observation is on the wrong side of the hyperplane} \\ > 0: & \text{observation is on the wrong side of the margin} \\ = 0: & \text{observations is on the right side of the margin} \end{cases} \tag{3.3.15}$$

The resulting model is subsequently better able to deal with potentially ambiguous data (classification overlaps) and so is less sensitive to any changes in the training data. [26, Figure 9.6, p. 346], reproduced in Figure 3.6, provides further details on this interpretation.

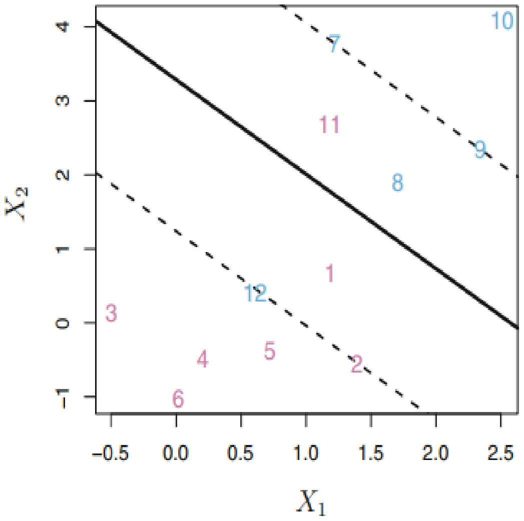


Figure 3.6: The black solid line represents the separating hyperplane. The dashed lines represent the margins. Points 2-6, 7, 9 and 10 are on the *right* side of (or on) their respective margins. 1 and 8 are on the *wrong* side of their respective margins. 11 and 12 are on the wrong side of the hyperplane *and* their respective margin

From [25, Chapter 12.2.1, p. 420], we also have that the solution of the SVC equations (3.3.11) - (3.3.14) depends only on the inner product of the observations:

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j} \tag{3.3.16}$$

so that the solution of the linear SVC decision boundary can be represented as [26, p. 351]:

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_{i'} \rangle \tag{3.3.17}$$

where $\mathcal{S}$ is the collection of indices from the solution's support vectors. Thus after finding the linear decision boundary of the SVC, one can again classify any observations x^* based on the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + ... + \beta_p x_p^*$ as per (3.3.7). Additional technical details on MMC and SVC can be found in [26, Chapter 9.1.4, p. 342 and Chapter 9.2.2, p. 345] and [25, Chapter 12.2, page 417].

Support Vector Machines (SVM)

Support vector machines (SVM) build on the SVC linear classification method and generalise the approach to cases where where linear decision boundaries are not appropriate, such as the case illustrated by Figure 3.7 taken from [26, Figure 9.8, p. 349].

One of the ways to create non-linear decision boundaries is by enlarging the input feature space of the data. For example, for data of the format $X_1, X_2, ..., Xp$, we can derive the additional data points $X_1^2, X_2^2, ..., Xp^2$. Then, using both sets of points $X_1, X_1^2, X_2, X_2^2, ..., Xp, Xp^2$ we can fit an SVC and create a non-linear decision boundary. In more general terms, SVM enlarges the feature space of the data in a systematic way by using what is known as *kernels*. A *kernel* is simply a function K used in the generalised form of the inner product (3.3.16), which we denote by $K(x_i, x_{i'})$.

Thus, as in [25, Chapter 12.3, p. 423] and [26, Chapter 9.3.2, p. 351], the solution of the SVM using the generalization of the inner product is represented by:

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i) \tag{3.3.18}$$

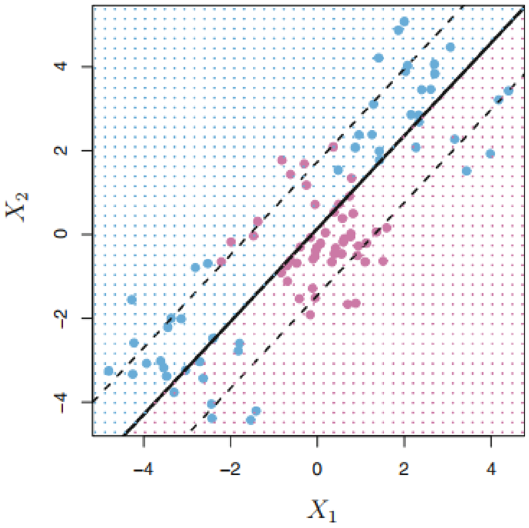


Figure 3.7: Example of Data with a Non-linear Decision Boundary and the Resulting Poor Fit from the SVC Approach

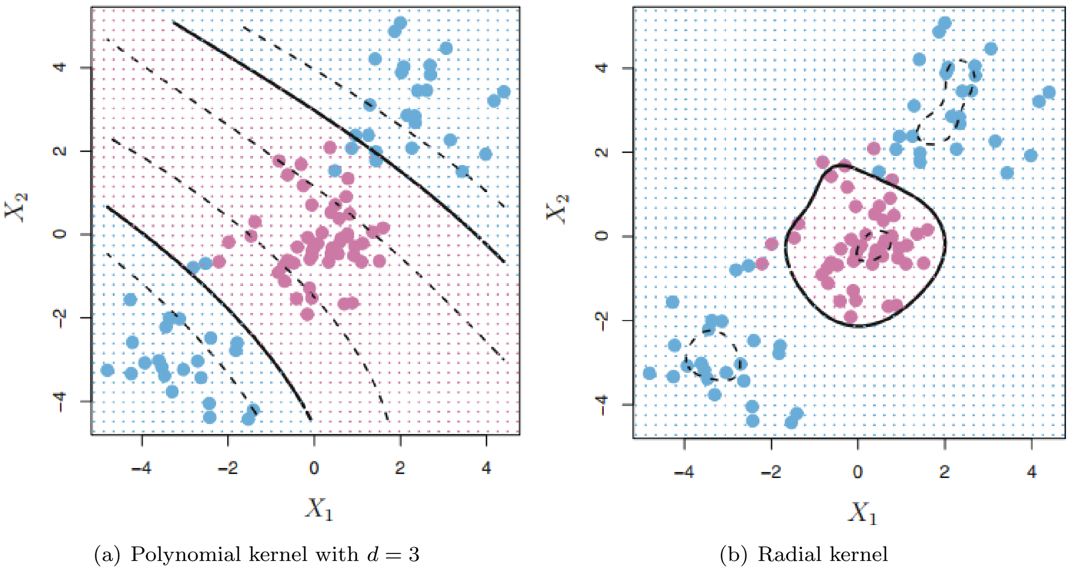
The 2 most common functions used in the *kernel* of the SVM are the *polynomial* functions:

$$K(x_i, x_{i'}) = (r + \gamma \sum_{j=1}^{p} x_{ij} x_{i'j})^d \tag{3.3.19}$$

and *radial* function:

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2) \tag{3.3.20}$$

Thus when a SVC is fit to data represented in a higher-dimensional space using non-linear kernels, the resulting classifier is referred to as a SVM. Figure 3.8(a) shows an example of an SVM classifier using a *polynomial* basis with $d = 3$ and Figure 3.8(b) shows the resulting SVM classifier using a *radial* basis.



(a) Polynomial kernel with $d = 3$

(b) Radial kernel

Figure 3.8: SVM Decision Boundaries Using Different Kernels

3.3.4 Extreme Gradient Boosting

Extreme gradient boosting (XGB) is quite an intricate algorithm that builds on the concepts used in a number of other algorithms. XGB could be seen as a highly modified version of the Gradient Boosting (GB) algorithm, which in turn is based on the concept of classification trees already discussed in the RF algorithm Section 3.3.2. The *extreme* part of XGB is due to the lengths the algorithm goes to in order to optimise and speed up calculations when compared to the GB algorithm. We will therefore start this section by analysing the GB algorithm. Subsequently, we will discuss the advantages and disadvantages of this approach, before looking at how the XGB algorithm improves on GB's drawbacks. As with the RF algorithm, given the aim of this paper and the format of the dependent variable, we will present the categorical classification versions of the GB and XGB algorithms. The same algorithms however, can handle both categorical and numerical data as both inputs and outputs.

Gradient Boosting (GB) Algorithm

The Gradient Boosting (GB) algorithm is another *ensemble* tree algorithm, similar to the RF algorithm discussed in Section 3.3.2. However the algorithms have fundamental differences particularly around how the trees are constructed, which will become evident as the details of the algorithm are explained.

Starting with a 2-class dependent (target) variable y (e.g. 'No' and 'Yes') encoded with values 0 and 1 and using the notation as per Section 3.3.2, where $\boldsymbol{x_i} = (x_{i1}, x_{i2}, ..., x_{ip})$ represents the i^{th} observation vector of p independent variables and y_i represents the observation of the i^{th} dependent variable, then for $i = 1, ..., N$ observations we end up with a matrix of observed independent and dependent variables.

Similar to logistic regression in Section 3.3.1, this algorithm focuses on predicting the log-odds (and hence the probability) of the dependent variable y's outcome. One could then interpret a predicted probability above 0.5 as one of the outcomes (e.g. 'Yes') and any predicted probability below 0.5 as the other outcome (e.g. 'No'), although any other boundary for this classification is also possible. The outline of the GB algorithm is given in Algorithm 3 and each step will be explained using [25, Algorithm 10.3, p. 361] and [28] as a basis.

Algorithm 3 Gradient Boost, Based on [25, Algorithm 10.3, p. 361] and [28]

```
Inputs:
    * Data $\{(x_i, y_i)\}_{i=1}^{N}$
    * A differentiable Loss Function $L(y_i, F(x))$
Initialise: $F_0(x) = \underset{\gamma}{\text{argmin}} \sum_{i=1}^{N} L(y_i, \gamma)$
for $m$ = 1 to $M$ do
    Compute $r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$ for $i = 1, ..., N$
    Fit a regression tree to the $r_{im}$ values and create terminal regions $R_{jm}$ for $j = 1, ..., J_m$
    for $j = 1, ..., J_m$ do
        Compute $\gamma_{jm} = \underset{\gamma}{\text{argmin}} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$
    end
    Update $F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_m \mathbb{1}\{x \in R_{jm}\}$
end
Output: $F_M(x)$
```

The differentiable **Loss Function** $L(y_i, F(x))$ most commonly used in the classification version of the GB algorithm is based on the log-likelihood function also used to fit the logistic regression parameters, given in (3.3.2). However, in contrast to the logistic regression setting where parameters are fit based on maximising the log-likelihood function, the GB classification algorithm is initialised by minimising the *negative* log-likelihood function:

$$F_0(x) = \underset{\gamma}{\text{argmin}} \sum_{i=1}^{N} L(y_i, \gamma) = \underset{\gamma}{\text{argmin}} \left(-\sum_{i=1}^{N} \{y_i \log(p) + (1 - y_i) \log(1 - p)\} \right) \tag{3.3.21}$$

In (3.3.21), γ is the log(odds), so that the loss function is given by (3.3.22)

$$\begin{aligned} L(y_i, \gamma) &= -y_i \log(p) + (1 - y_i) \log(1 - p) \\ &= -y_i \left(\log(p) - \log(1 - p)\right) - \log(1 - p) \\ &= -y_i \log(\text{odds}) + \log(1 + \mathrm{e}^{\log(\text{odds})}) \end{aligned} \tag{3.3.22}$$

$$\big(\text{since } \log(1 - p) = \log(1 - \frac{\mathrm{e}^{\log(\text{odds})}}{1 + \mathrm{e}^{\log(\text{odds})}}) = -\log(1 + \mathrm{e}^{\log(\text{odds})})\big)$$

This function is differentiable, since:

$$\frac{dL(y_i, \gamma)}{d(\log(\text{odds}))} = -y_i + \frac{\mathrm{e}^{\log(\text{odds})}}{1 + \mathrm{e}^{\log(\text{odds})}} \tag{3.3.23}$$

$$= -y_i + p \tag{3.3.24}$$

To initialise the algorithm, we need to find the γ that minimises $F_0(x) = \underset{\gamma}{\text{argmin}} \sum_{i=1}^{N} L(y_i, \gamma)$. To find this γ we therefore take sum of the derivatives for each observation $i = 1, ..., N$ and set them equal to 0, to yield (3.3.25).

$$\sum_{i=1}^{N} \frac{dL(y_i, \gamma)}{d(\log(\text{odds}))} = \sum_{i=1}^{N} \left(-y_i + \frac{\mathrm{e}^{\log(\text{odds})}}{1 + \mathrm{e}^{\log(\text{odds})}} \right) = \sum_{i=1}^{N} (-y_i + p) = 0$$

Thus

$$p = \frac{1}{N} \sum_{i=1}^{N} y_i$$

which is the empirical probability of *Y=1.* Hence, in terms of log-odds

$$F_0(x) = \log\left(\frac{p}{1 - p}\right) = \log\left(\frac{\sum_{i=1}^{N} y_i}{N - \sum_{i=1}^{N} y_i}\right) \tag{3.3.25}$$

Hence using this loss function, the algorithm is initialised with the empirical log(odds) of the dependent variable y, encoded with values 0 and 1. The loop then starts by computing r_{im}. The dependent variable y has been encoded with values 0 and 1, so that a low predicted probability (e.g. below 0.5) corresponds to 0 and a high predicted probability (e.g. above 0.5) corresponds to 1. Therefore, since the *negative* derivative of the loss function is nothing but the difference between the observed value y and the predicted probability p, r_{im} is a *pseudo* residual:

$$\begin{aligned} r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} &= \left[y_i - \frac{\mathrm{e}^{\log(\text{odds})}}{1 + \mathrm{e}^{\log(\text{odds})}}\right]_{F(x)=F_{m-1}(x)} \\ &= [y_i - p]_{F(x)=F_{m-1}(x)} \end{aligned}$$

The $F(x) = F_{m-1}(x)$ specification means that *pseudo* residuals are calculated using the estimated value of y from the previous iteration. In case of $m = 1$, the initialised value $F_0(x)$ (3.3.25) is used and for $m > 1$, the value derived from the previous loop-iteration is used.

The next step builds regression trees for the *pseudo* residuals in each of the terminal regions. The regression tree roots, decision nodes and leaves usually differ in each iteration m as they are fit to predict new *pseudo* residuals to try and correct any misclassifications from the previous iteration. Also, the trees are generally restricted to have between 8 to 32 leaves through the input parameter J_m, although in the simplest setting $J_m = J$ for all m. Regression trees are similar to classification trees, but use different loss functions, predict numerical rather than categorical outcomes for the dependent variable y and use an *averaging rule* rather than the *majority voting rule*. More details on regression trees can be found in [26, Chapter 8.1.1, p. 304], [25, Chapter 9.2.2, p. 307] and [28].

As the algorithm progresses, new γ's are calculated for each new leaf created by the regression trees for the terminal regions. This is often achieved by using a Taylor expansion of the loss function $L(y_i, F_{m-1}(x_i) + \gamma)$, details of which can be found in [28]. Finally, the predicted log(odds) of an observation are updated using the learning rate ν (typically set to 0.1) multiplied by the γ that belongs to each applicable terminal region.

Pros and Cons of GB Algorithm

The RF algorithm constructs trees using random selection of independent variables in each iteration, which leads to a reduction in the variance of its predictions. In contrast, the GB algorithm constructs trees sequentially using outcomes (and their *pseudo* residuals) from the previous tree to build new trees that aim to improve upon any misclassifications. Thus, while the GB algorithm shares a number of strengths of classification trees and the RF algorithm, it also has some weaknesses that we will briefly outline:

+ Each iteration of the GB algorithm improves its predictive ability

+ Due to the restricted size of each GB tree via the tuning parameters J_m, the trees can be quicker to construct than the larger RF trees

− Due to the sequential improvement of each GB tree, the algorithm can lead to overfitting

− GB models can be harder to tune successfully as the algorithm relies on more tuning parameters than the RF algorithm

− In each iteration, the GB algorithm needs to minimise the loss function (e.g. based on the log(odds)) before updating the predicted value, which can slow down its implementation and (depending on the data) require more resources than the RF algorithm.

Extreme Gradient Boosting (XGB) Algorithm

Extreme gradient boosting (XGB) is a highly modified version of the GB algorithm with a number of differences and additional steps that aim to improve its speed and predictive accuracy. Due to the number of features and optimisations in XGB, we present a simplified version of the XGB classification algorithm in Algorithm 4. This algorithm is based on using a 0/ 1 encoded version of the dependent variable y. As with the GB algorithm, XGB updates the log-odds of y in each iteration, with the corresponding probability being interpreted as the probability of the categorical outcome (e.g. probability below 0.5 means the categorical outcome encoded with 0 and probability above 0.5 means the categorical outcome encoded with 1).

In order to understand the algorithm conceptually, we will present a simplified regression tree objective function (3.3.27) and then work through some of the details before relating it to the steps in Algorithm 4. We use the conceptual overview presented in [29] and start with the objective function used in the tree construction given in (3.3.26):

$$\underset{\mathcal{O}_{value}^{m}}{\text{argmin}} \left[\sum_{i=1}^{N} L(y_i, l_i^m) + \gamma\, T + \frac{1}{2}\lambda(\mathcal{O}_{value}^{m})^2 \right] \tag{3.3.26}$$

where:

$$\begin{aligned}
l_i &= \log(\text{odds}) \text{ of } y_i \\
L(y_i, l_i) &= -y_i l_i + \log(1 + \mathrm{e}^{l_i}) \\
\gamma &= \text{penalty parameter to encourage tree pruning} \\
\lambda &= \text{regularisation term} \\
\mathcal{O}_{value} &= \log(\text{odds}) \text{ output value of regression tree}
\end{aligned}$$

The ($\gamma\, T$)-term in (3.3.26) is not used in the tree construction process, so in order to focus on providing conceptual insights this term will be ignored. As with GB in Algorithm 3, each iteration updates the predicted log(odds) using $l_i^m = l_i^{m-1} + (\mathcal{O}_{value})_i^m$, so that the objective function can be written as (3.3.27).

$$\underset{\mathcal{O}_{value}^m}{\operatorname{argmin}} \left[\sum_{i=1}^{N} L(y_i, l_i^{m-1} + \mathcal{O}_{value}^m) + \frac{1}{2}\lambda(\mathcal{O}_{value}^m)^2 \right] \tag{3.3.27}$$

To make finding the optimal $\mathcal{O}_{value}^m$ efficient, the Taylor expansion of $L(y_i, p_i^{m-1} + \mathcal{O}_{value}^m)$ is utilised, as per (3.3.28).

$$\begin{aligned}
L(y_i, l_i^{m-1} + \mathcal{O}_{value}^m) &\approx L(y_i, l_i^{m-1}) + \frac{d}{dl_i} L(y_i, l_i^{m-1}) \mathcal{O}_{value}^m + \frac{1}{2}\frac{d^2}{dl_i^2} L(y_i, l_i^{m-1})(\mathcal{O}_{value}^m)^2 \\
&= L(y_i, l_i^{m-1}) + (g)\mathcal{O}_{value}^m + \frac{1}{2}(h)(\mathcal{O}_{value}^m)^2 \qquad (3.3.28)
\end{aligned}$$

Hence, in each XGB-iteration m, the regression tree should be built to minimise (3.3.29)

$$\begin{aligned}
\sum_{i=1}^{N} L(y_i, l_i^{m-1} + \mathcal{O}_{value}^m) + \frac{1}{2}\lambda(\mathcal{O}_{value}^m)^2 &\approx \sum_{i=1}^{N} L(y_i, l_i^{m-1}) + \frac{1}{2}\lambda(\mathcal{O}_{value}^m)^2 \\
&\left[\sum_{i=1}^{N}(g)\right] \mathcal{O}_{value}^m + \frac{1}{2}\left[\sum_{i=1}^{N}(h)\right](\mathcal{O}_{value}^m)^2 \qquad (3.3.29)
\end{aligned}$$

The derivative for $g = \frac{d}{dl_i} L(y_i, l_i^{m-1})$ is given in (3.3.23)-(3.3.24), while the derivative in terms of log(odds) and p for $h = \frac{d^2}{dl_i^2} L(y_i, l_i^{m-1})$ can also be derived as per (3.3.30)-(3.3.31) [28].

$$\begin{aligned}
\frac{d^2}{dl_i^2} L(y_i, l_i^{m-1}) &= \frac{-\mathrm{e}^{2\, l_i^{m-1}}}{(1 + \mathrm{e}^{l_i^{m-1}})^2} + \frac{\mathrm{e}^{l_i^{m-1}}}{1 + \mathrm{e}^{l_i^{m-1}}} \\
&= \frac{\mathrm{e}^{l_i^{m-1}}}{1 + \mathrm{e}^{l_i^{m-1}}} \frac{1}{1 + \mathrm{e}^{l_i^{m-1}}} \qquad (3.3.30) \\
&= p_i^{m-1}(1 - p_i^{m-1}) \qquad (3.3.31)
\end{aligned}$$

Hence, taking the derivative in (3.3.29) with respect to $\mathcal{O}_{value}^m$ and setting this equal to 0 yields the optimal value $\mathcal{O}^{*m}_{value}$ as per (3.3.32).

$$\begin{aligned}
&\frac{d^2}{d(\mathcal{O}_{value}^m)^2} \left\{ \sum_{i=1}^{N} L(y_i, l_i^{m-1}) + \left[\sum_{i=1}^{N}(g)\right] \mathcal{O}_{value}^m + \frac{1}{2}\left[\sum_{i=1}^{N}(h) + \lambda\right](\mathcal{O}_{value}^m)^2 \right\} \\
&= \sum_{i=1}^{N}(g) + \left[\sum_{i=1}^{N}(h) + \lambda\right](\mathcal{O}_{value}^m) = 0
\end{aligned}$$

Hence

$$\begin{aligned}
\mathcal{O}^{*m}_{value} &= \frac{-\sum_{i=1}^{N}(g)}{\sum_{i=1}^{N}(h)+\lambda} \\
&= \frac{\sum_{i=1}^{N}\left[y_i - p_i^m\right]}{\left[\sum_{i=1}^{N} p_i^{m-1}(1-p_i^{m-1})\right]+\lambda} \\
&= \frac{\sum_{i=1}^{N}\text{residual}_i^m}{\left[\sum_{i=1}^{N} p_i^{m-1}(1-p_i^{m-1})\right]+\lambda}
\end{aligned} \tag{3.3.32}$$

By first looking at some of the mathematical details on how the XGB-regression tree finds the optimal output value for each leaf, we can now analyse XGB presented in Algorithm 4 and make more sense of the steps and criteria used to construct the regression trees.

Algorithm 4 Simplified eXtreme Gradient Boost (Classification), Based on [29]

```
Inputs:
    * Data $\{(x_i, y_i)\}_{i=1}^N$
    * A differentiable Loss Function $L(y_i, F(x))$
    * $\gamma$ for pruning
    * $\lambda$ for regularisation
Initialise: m = 1 and $p_i^0 = 0.5$ for all $i = 1, ..., N$
while (residuals > stopping criteria) OR (# trees built $m \leq M$) do
    Compute $r_i^m = y_i - p_i^{m-1}$ = observed - predicted classification
    Fit a regression tree to the $r_i^m$ values and create terminal regions $R_j^m$ for $j = 1, ..., J$ using similarity score instead of Gini impurity
    Calculate gain for each level of the regression tree
    Prune regression tree using $\gamma$ as minimum gain threshold
    Calculate output at each leaf
    Update log(odds)$_i$ for observation $y_i$: $l_i^m = l_i^{m-1} + \varepsilon\, \mathcal{O}^{*m}_{value}$
    Calculate leaf probabilities: $p_i^m = \frac{e^{\log(\text{odds})_i^m}}{1+e^{\log(\text{odds})_i^m}}$
    Set m = m + 1
end
Output: $p_i^m$ for each leaf to be used for predicted classification
```

The data format and **Loss Function** are the same as used in GB and so we will focus on the new concepts introduced in XGB, starting with the initialisation of p_i. By default, XGB initialises the probability associated with each y_i in the observed data to 0.5, although this default value can be changed to something more meaningful such as the empirical probability derived from the data (i.e. $\frac{\sum_{i=1}^{N} y_i}{N}$). γ and λ are input parameters that can be tuned to optimise the in- and out-of-sample performance of the algorithm.

Next, XGB calculates the residuals of the dependent variable using the previous probability from the algorithm (for $m = 1$, it uses the default p_i^0 and for $m > 1$ it uses the probabilities estimated in the previous iteration). A regression tree is fit, using the criteria called *similarity score*, rather than the *gini impurity* as for the previous algorithms. The *similarity score* is related to the last 2 terms in (3.3.29) and aims to set the split in the decision node at the point which maximises the *negative* sum of g and h at the optimal value $\mathcal{O}^{*m}_{value}$ (i.e. set at the minimum value of their positive sum). The *similarity score* is given by (3.3.33)

$$S = -\left\{\left[\sum_{i=1}^{N}(g)\right]\mathcal{O}^{*m}_{value} + \frac{1}{2}\left[\sum_{i=1}^{N}(h)\right](\mathcal{O}^{*m}_{value})^2\right\}$$

substituting (3.3.32)

$$
\begin{aligned}
&= \frac{1}{2}\frac{\left[\sum_{i=1}^{N}(g)\right]^2}{\sum_{i=1}^{N}(h)+\lambda} \\
&= \frac{1}{2}\frac{\left[\sum_{i=1}^{N}\text{residual}_i^m\right]^2}{\left[\sum_{i=1}^{N}p_i^{m-1}(1-p_i^{m-1})\right]+\lambda} \qquad (3.3.33)
\end{aligned}
$$

In practice, to reduce computation times, the $\frac{1}{2}$ is ignored, as the algorithm is only looking for the largest similarity score S. The similarity score aims to create splits in the tree so that each new sub-group of observations is clustered together. This is achieved through the $\left[\sum_{i=1}^{N}\text{residual}_i^m\right]^2$ term which increases if observations are 'on the same side' of the probability being used and decreases (due to the offset in residuals) if the observations are 'on different sides' of the probability being used. The trees created by splitting the decision nodes along the highest similarity score are by default (in Python) limited to no more than 6 levels.

Next, the *gain* for each level is computed as the difference between the sum of the similarity scores in the new leaves and the similarity score of the original decision node:

$$\text{Gain} = S_{\text{left leaf}} + S_{\text{right leaf}} - S_{\text{decision node}} \qquad (3.3.34)$$

Using the *gain*, the regression trees are pruned from the bottom (lowest level) upwards. If the *gain* of a level is less than γ, the split (leaves) is removed (pruned). The gain of the new, higher regression tree level after pruning (i.e. the previous decision node) is again compared to γ and pruning continues unless *gain* $> \gamma$. If the gain in a level exceeds γ pruning stops along the branch. However, in an extreme scenario, the pruning process could reduce the regression tree to a single leaf (i.e. no splits).

After establishing the regression tree, XGB calculates the *output*, which is just the value $\mathcal{O}{*}_{value}^{m}$ calculated in (3.3.32). Using this *output* value, the log(odds) estimate of each y_i is updated as $l_i^m = l_i^{m-1} + \varepsilon\, \mathcal{O}{*}_{value}^{m}$, where ε is the learning rate and has a default value of 0.3.

Finally, the probability for each dependent variable observation y_i is calculated from the log(odds), from which the predicted outcome (category) can be inferred. This process is repeated until either the residuals fall below a certain level (stopping criteria) or a maximum number of trees M has been built.

XGB is considered one of the best performing machine learning algorithms, both in terms of predictive ability (in- and out-of-sample) as well as speed of calibration. By 2016 XGB had been used in 17 of the 29 winning submissions for Kaggle Machine Learning competitions with 8 of these 17 winning submissions using only XGB and no other approach (i.e. ensemble of multiple algorithms) [30]. Furthermore, tests on large data sets such as those by [31] have also shown XGB can run up to 17 times faster than the RF algorithm and up to 86 times faster than the GB algorithm and still achieve a higher predictive accuracy. Figure 3.9 taken from [31] shows this comparison.

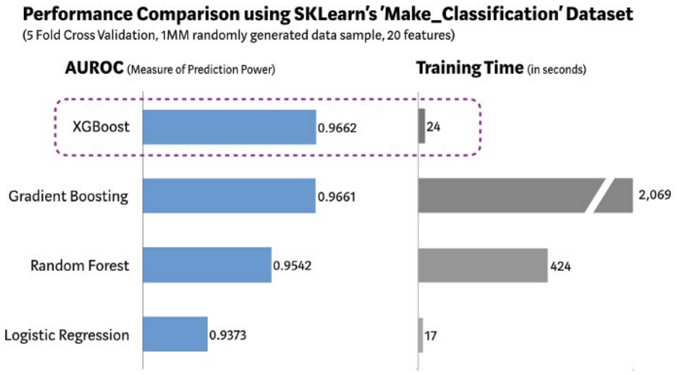


Figure 3.9: Comparison of Theoretical Model Speeds and Accuracies

Chapter 4

Data and Analysis

4.1 Data Overview

We used academic access to Crunchbase data [18] to access 17 data sets covering various company features. In its raw format, the data contains a snapshot of all companies as at the extract date as well as some historical event information. After an analysis of the features, the 7 data sets listed in Table 4.1 were selected to construct a time series view of the startups before the external data listed in Table 4.2 was appended in an attempt to enhance the predictive ability of the models.

Data set	**Description of data used**
Acquisitions	Company level data on acquisition activity including: acquiree (company) unique ID and acquisition date.
Degrees	Per person data including: person unique ID, degree held, degree completion date and degree issuing institution.
Funding rounds	Company level data on funding activity including: company unique ID, funding round and funding round date.
IPOs	Company level data on IPOs including: company unique ID and IPO date.
Jobs	Per person data including: person unique ID, company unique ID, job title, employment start date, employment end date and current job indicator.
Organizations	Company level data including: company unique ID, country, current status, company industry, founded date, closed date, company legal name, facebook url, twitter url and linkedin url.
People	Per person data including: person unique ID, gender and nationality.

Table 4.1: Overview of Data Sets Used from Crunchbase

Data Set	**Description**
Economic data	Consumer price index (CPI), average 10 year government bond yield, GDP growth rates, M1 money supply, M3 money supply, unemployment rates, government asset purchase levels (quantitative easing), private household consumption and expenditure. These macro-ecnomic indicators are added per country on a 3-month lagged basis to ensure they would be available both historically and for future calibration purposes. They aim to capture broad macro-economic developments which could impact the success or failure of startups.

Table 4.2: Overview of External Data Added

4.2 Analysis

4.2.1 Context

To understand the data that we will be working with, it is important to first understand what the Crunchbase data represents and its context within the population of all startup companies.

As listed in Table 1.1, the broad aim of venture capital funds is to "acquire minority stakes in early-stage, high-growth potential technology and life sciences companies (startups) to provide financing for continued growth". Crunchbase in turn describes itself as "a directory of technology-related companies and people"[1] and is a "platform for professionals to discover innovative companies, connect with the people behind them, and pursue new opportunities"[2]. Crunchbase therefore represents a key segment of companies that Venture Capital investors would typically be interested investing in.

To understand what proportion startups Crunchbase represents, we can look at the total number of new startups each year and compare this to the number of startups listed on Crunchbase by founding year. Starting with the United States, the Census Bureau [32], [33], [34] reports 3 key measures:

Measure	Definition
Business Application	Application for an Employer Identification Number (EIN), which is a company tax identification number for the IRS.
Business Formation	A company that reports wage payment (after receipt of EIN).
Nonemployer Company	Businesses that have no paid employment or payroll, are subject to federal income taxes, and have receipts of \$1,000 or more (\$1 or more for the Construction sector).

Table 4.3: Key Startup Measures Reported by the US Census Bureau

Using figures reported up to 2019Q1, the number of business applications has risen from ca. 2.5 million in 2005 to ca. 3.5 million in 2019. However, over the same period, the number of businesses formed over the next 12 months (4 quarters) has decreased significantly, so that the number of new businesses formed over the same period has fallen from ca. 0.8 million to ca. 0.6 million. Another phenomenon in the US, is the number of nonemployer companies which make up over 80% of all companies. From the definition in Table 4.3, these companies would clearly not be of interest to Venture Capital investors. Applying this 80% haircut to the number of business formations within 4 quarters each year, means there are only around 60,000 new startups being formed each year (not accounting for any other exclusions Venture Capital investors might apply). Comparing this to the number of companies listed on Crunchbase (by founding year), implies Crunchbase captures around 12% of relevant companies within 1 year of formation and around 25% within 5 years after formation. Data supporting these statements are presented in Tables A.3, A.4 and A.5.

For the UK, data is only only available for company incorporations rather than company formations. Given that the proportion of nonemployer companies in the UK[3] (76%, see Table A.8) is similar to that of the US at (80%), we can compare the proportion of all new businesses (applications/ incorporations) covered by the Crunchbase data set as per Tables A.6 and A.7. This shows that the proportion of *all* US and UK companies covered by Crunchbase is similar, albeit slightly higher in the UK, from which we could infer that Crunchbase covers at least the same proportion (or more) of relevant UK companies within 1 and 5 years of founding.

4.2.2 Data Reliability

Before starting with data cleaning, processing and predictive modelling, some checks for reliability were performed on the Crunchbase data. Of the 989,122 companies covered by Crunchbase globally on 2020/06/06 only 33,778 (3.4%) are listed as being closed, while the remaining 955,344 (96.6%) are either still operating (84.7%), acquired (9.6%) or had an IPO (2.3%). While the obvious

[1] https://support.crunchbase.com/hc/en-us/articles/360001464407-Hi-

[2] https://about.crunchbase.com/

[3] UK definition: sole proprietorships and partnerships with only a self-employed owner-manager(s) and companies with one employee, assumed to be an employee director

interpretation of this is one of survivorship bias, we first address the question also raised in [17, Section III-B, p 124237] on whether the Crunchbase data is potentially not being updated for closed companies. In other words, there is a question of whether or not the Crunchbase company status is accurate. To address this question, we first focus on the 70,659 UK-based companies listed on Crunchbase. The reason for this choice is that Companies House[4] releases a quarterly snapshot of all UK-based companies [35] that includes their current status. We can therefore compare this to the status listed on Crunchbase and so assess the accuracy of the data. We limit this assessment to UK-based companies founded between 2000-2017 and companies which have at least one entry in the 'funding rounds' data base, as these criteria will be the basis for developing predictive models. Applying these filters leaves **10,168** companies, of which 9,913 (97.5%) are active and 255 (2.5%) are closed.

For each company, Crunchbase has both a company name and a legal (registered) name. For the UK-based companies under consideration, Crunchbase lists the company name for all companies, but the legal name for only 2,677 (26.3%) companies. In order to merge the 2 data sets, a combination of either a company's cleaned[5] legal name (where available) or cleaned company name together with the founded year and postcode was used. Where data couldn't be matched on this basis, further attempts to use only cleaned legal/ company name and founded year and only cleaned legal/ company name were made. A final attempt included matching the previous 1, 2 and 3 cleaned company names held by Companies House against the cleaned legal/ company name on Crunchbase. The results of this matching process are shown in Table 4.4.

Match Method	**Matched**	**Match Rate**
Cleaned legal/ company name, founded year and postcode	1,144	11.3%
Cleaned legal/ company name and founded year	735	7.2%
Cleaned legal/ company name	3,801	37.4%
Previous 1 cleaned legal/ company name	374	3.7%
Previous 2 cleaned legal/ company name	42	0.4%
Previous 3 cleaned legal/ company name	4	0.0%
Total	**6,100**	**60.0%**

Table 4.4: Overview of Crunchbase and Companies House Matches

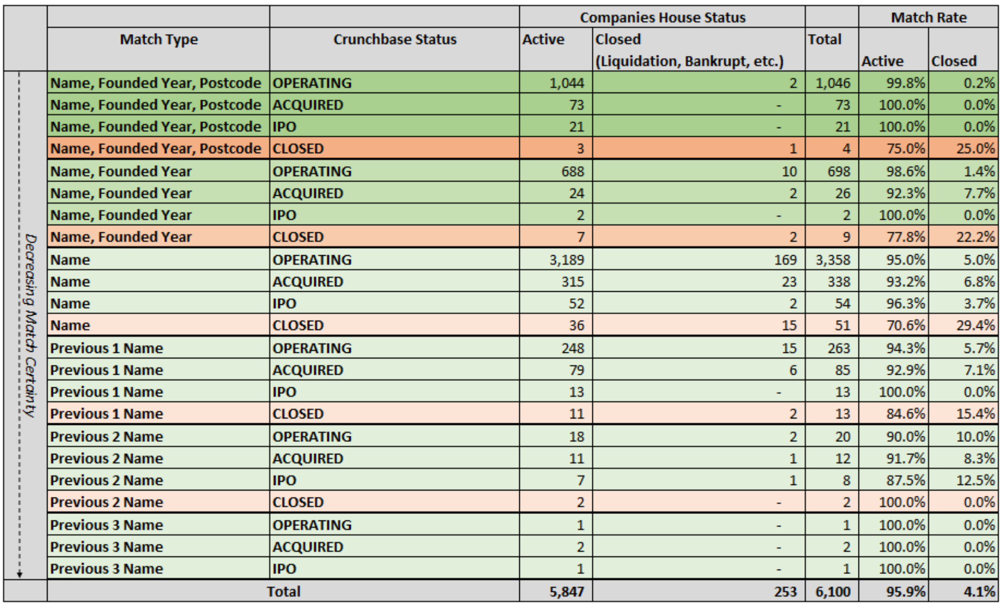
The statuses for the 6,100 (60.0%) companies were subsequently compared, showing that for cases that were matched using cleaned legal/ company name, founded year and postcode close to 100% of the companies with a status of operating, acquired or IPO were listed as being active on Companies House. This alignment however, decreases as the certainty of the match decreases although the statuses still match in at least 90% of cases. In the sample of 6,100 companies, only 79 (1.3%) had a closed status on Crunchbase and accordingly the match between the Crunchbase and Companies House status is relatively volatile. Interestingly, the majority of closed cases on Crunchbase are indicated as being active on Companies House. A manual review of a sample of these cases revealed that some acquired companies had a closed status on Crunchbase while in fact still being operational - meaning the owners considered the old company closed as they were operating under the acquirer company's name.

Another interesting observation is the increased number of companies classified as being closed on Companies House when data was matched only on the company name. Although the match in these cases is less certain, it does suggest there should be up to ca. 200% more companies (1.3% vs. 4.1%) listed as being closed. However, if we increased the number of closed UK companies (2.5%) from the **10,168** sample by 200%, this would still imply only 7.5% are closed while 92.5% are active. A summary of the status comparisons is provided in Figure 4.1.

The reliability of most of the other data listed in Table 4.1 is difficult if not impossible to verify independently. A description of potential validation sources and procedures is provided in Table A.9. A summary of the results from checking a random sample of US companies' acquisition, IPO and organizations data is also provided below this table.

[4] Companies House is an executive agency, sponsored by the Department for Business, Energy & Industrial Strategy responsible for incorporating and dissolving limited companies.

[5] Company names were cleaned by removing: non-word characters, non-Unicode whitespaces, leading spaces, trailing spaces and any instances of 'LTD' and 'LIMITED'

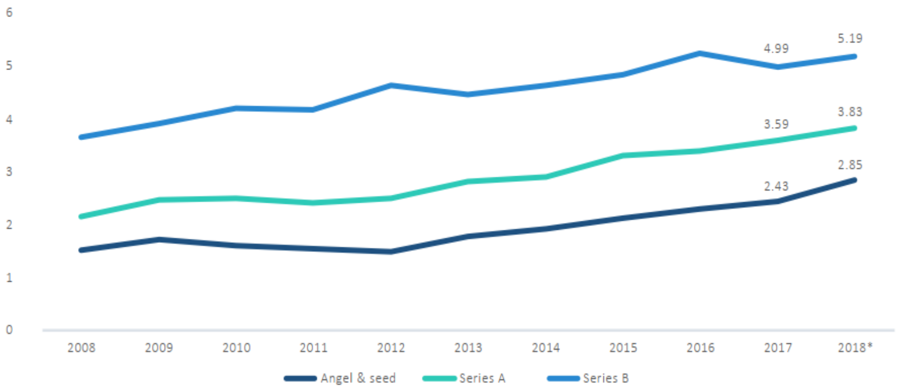


			Companies House Status			Match Rate	
	Match Type	Crunchbase Status	Active	Closed (Liquidation, Bankrupt, etc.)	Total	Active	Closed
Decreasing Match Certainty	Name, Founded Year, Postcode	OPERATING	1,044	2	1,046	99.8%	0.2%
	Name, Founded Year, Postcode	ACQUIRED	73	-	73	100.0%	0.0%
	Name, Founded Year, Postcode	IPO	21	-	21	100.0%	0.0%
	Name, Founded Year, Postcode	CLOSED	3	1	4	75.0%	25.0%
	Name, Founded Year	OPERATING	688	10	698	98.6%	1.4%
	Name, Founded Year	ACQUIRED	24	2	26	92.3%	7.7%
	Name, Founded Year	IPO	2	-	2	100.0%	0.0%
	Name, Founded Year	CLOSED	7	2	9	77.8%	22.2%
	Name	OPERATING	3,189	169	3,358	95.0%	5.0%
	Name	ACQUIRED	315	23	338	93.2%	6.8%
	Name	IPO	52	2	54	96.3%	3.7%
	Name	CLOSED	36	15	51	70.6%	29.4%
	Previous 1 Name	OPERATING	248	15	263	94.3%	5.7%
	Previous 1 Name	ACQUIRED	79	6	85	92.9%	7.1%
	Previous 1 Name	IPO	13	-	13	100.0%	0.0%
	Previous 1 Name	CLOSED	11	2	13	84.6%	15.4%
	Previous 2 Name	OPERATING	18	2	20	90.0%	10.0%
	Previous 2 Name	ACQUIRED	11	1	12	91.7%	8.3%
	Previous 2 Name	IPO	7	1	8	87.5%	12.5%
	Previous 2 Name	CLOSED	2	-	2	100.0%	0.0%
	Previous 3 Name	OPERATING	1	-	1	100.0%	0.0%
	Previous 3 Name	ACQUIRED	2	-	2	100.0%	0.0%
	Previous 3 Name	IPO	1	-	1	100.0%	0.0%
	Total		5,847	253	6,100	95.9%	4.1%

Figure 4.1: Comparison of Crunchbase and Companies House Statuses, By Match Type

4.2.3 Survivorship Bias

The implication that the number of closed, UK-based companies should potentially be increased by 200% still leaves ca. 92.5% of companies being classified as operational, which shows a definite survivorship bias in the Crunchbase data set. Models built on data with survivorship bias can overstate the positive outcomes that can be achieved (due to the data overstating positive outcomes/ understating negative outcomes). While there are many ways one can deal with this type of data, in the Venture Capital context this data feature may actually be beneficial. A 2019 review of funding activity by PitchBook reveals that the age at which startup companies receive early stage funding is increasing, as illustrated in Figure 4.2 [36, p. 10]. This shows that the median age of companies receiving angel & seed, series A and series B funding has increased to 2.85 years, 3.83 years and 5.19 years respectively and means Venture Capital firms require startup companies to prove their ability to survive before providing them with funding. The survivorship bias in the Crunchbase data could therefore be a consequence of companies only listing on Crunchbase after they have demonstrated their ability to survive.



Source: PitchBook
*As of November 30,2018

Figure 4.2: Median Years Since Founding by Funding Series, 2008-2018.

Also, when looking at this data more closely, even though the majority of companies are still active, the number of companies receiving more than 1 funding round is quite limited, as illustrated in Figure 4.3.

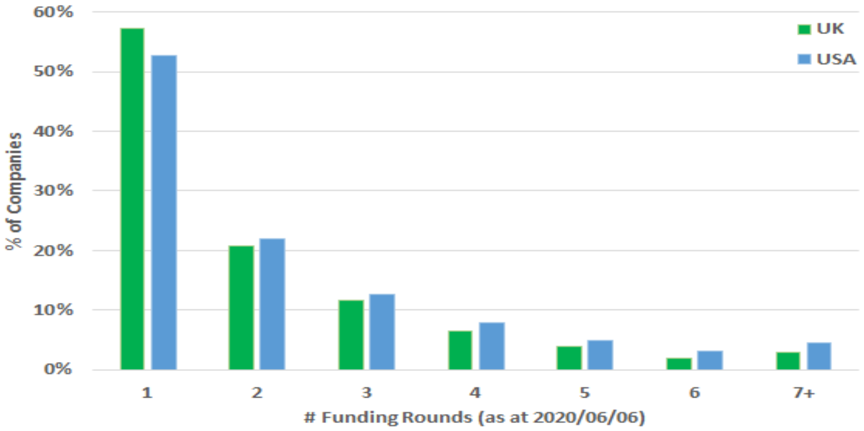


Figure 4.3: Number of Funding Rounds for UK and US-based Companies, Founded Between 2000-2017 with at least 1 Funding Round.

We can interpret this to mean:

1. The Crunchbase platform acts as a filtering mechanism for startups. Mostly startups that have had some previous success are included and this seems to be a strong indicator of being able to remain operational.
2. A startup company remaining operational in most cases is not the mark of a successful investment for a Venture Capital firm. As illustrated in Figure 3.1, these firms need an exit point in order to crystallise their returns. Hence firms using the Crunchbase platform to identify investments are still left with a significant challenge of finding the right investments, as the majority of startups do not offer profitable investment entry and exit points.

These insights would suggest that companies captured by Crunchbase are filtered towards companies that are both better able to survive and are more aligned to the profile of companies that Venture Capital investors would want to invest in. Therefore, as long as this data generating process for Crunchbase continues, we can see the survivorship bias as a positive feature for modelling, rather than a negative one that needs to be corrected for.

4.2.4 Data Overview

Overview

The 17 databases available from Crunchbase were split by topic and connected by various unique ID keys. Of these 17 data sets, the 7 listed in Table 4.1 were deemed most relevant for the purposes of this paper and so will be the focus of this section. Since most of the 989,122 startups are either US-based (327,401 or 33%) or UK-based (70,659 or 7%), we will focus on companies from these countries. Using the same criteria as in Section 4.2.2 for including companies in the model data (companies founded between 2000-2017 with at least one entry in 'funding rounds' data base), we end up with 58,260 US-based startups and 10,168 UK-based startups, for a total of 68,428 startups. We choose only companies with 'funding rounds' entries, as the premise for the *target variable* is that a company must already have either seed or series A+ funding in order to be considered by the Venture Capital investor.

An overview of the latest funding status is shown in Figures 4.4 and 4.5. The number of data points matched from 4 of the data sets to the 68,428 companies is shown in Table 4.5. We exclude 'organizations' as this is the base file, while the 'degrees' and 'people' data bases need to be matched to 'jobs' first to extract the employment period, which is done in the time series processing and is therefore out of scope for this section.

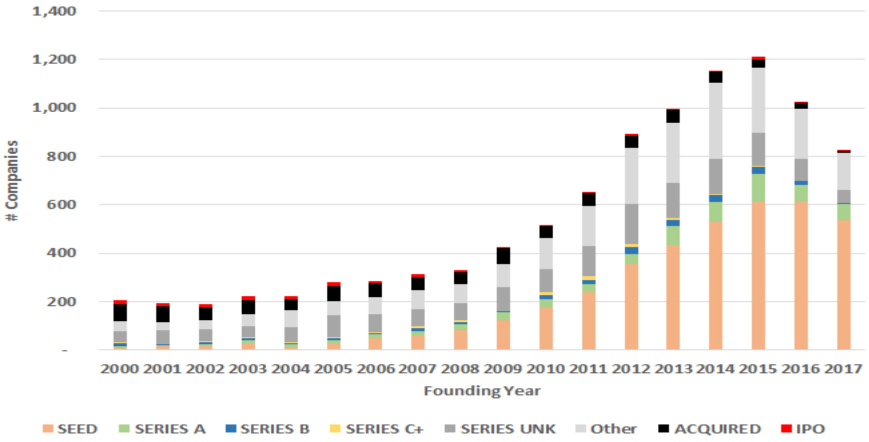


Figure 4.4: Overview of the Latest Funding Round for UK Startups, by Founding Year

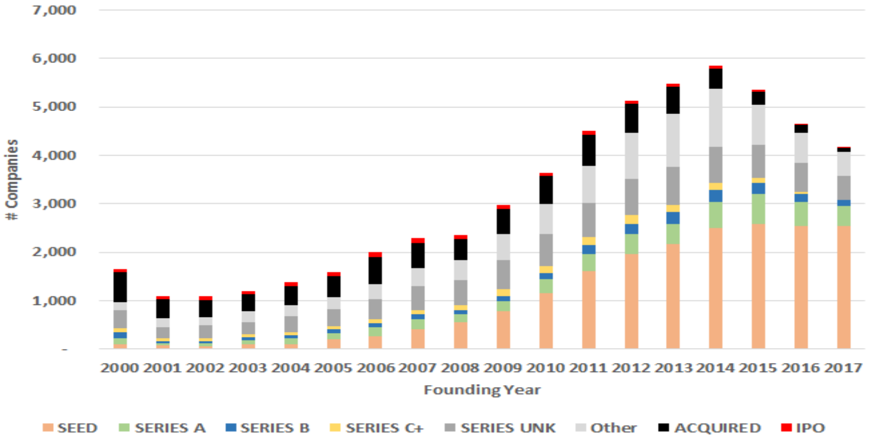


Figure 4.5: Overview of the Latest Funding Round for US Startups, by Founding Year

Country	Data Point Distrib.	Acquisitions	IPOs	Funding Rounds	Jobs
UK	1	865	184	5,802	1,949
	2	56	6	1,927	1,680
	3+	8	0	2,439	4,446
	Total Data Points	1,003	196	20,289	36,043
	Unique Companies	929	190	10,168	8,075
USA	1	8,039	1,426	29,889	10,280
	2	271	50	11,482	9,443
	3+	25	4	16,889	31,718
	Total Data Points	8,658	1,540	130,147	318,797
	Unique Companies	8,335	1,480	58,260	51,441
Total	1	8,904	1,610	35,691	12,229
	2	327	65	13,409	11,123
	3+	33	4	19,328	36,164
	Total Data Points	9,661	1,736	150,436	354,840
	Unique Companies	9,264	1,670	68,428	59,516

Table 4.5: Overview of the UK and US-based Startups Matched Across Different Databases

Companies with multiple 'Acquisitions' are those companies which have been acquired by multiple acquirers at different points in time. Similarly, companies with multiple 'IPOs' reflects a company moving its listing from one stock exchange to another (e.g. NASDAQ to NYSE). In the construction of the time series data, only the first acquisition or IPO event will be considered.

Missing Data

When data is not populated for some or all of independent and dependent variables, the results derived from this data can be skewed or misleading. Missing data can be an extensive topic in itself, covering areas such as identifying the type(s) of missing data and how to best deal with these. However, because of the discussion on context and survivorship bias above, we can already conclude that Crunchbase has various forms of missing data (e.g. not all successful startups are included and companies that have survived are more likely to be included than ones that fail). Assuming that Crunchbase data will continue to be generated in the same way going forward, we can then instead interpret these features as a filtration of the opportunity set for Venture Capital investors.

To simplify the time series construction, the variables listed in Table 4.1 and A.13 were chosen on the basis of having no missing data. While other variables with missing data could be included after applying imputation methods, the 124 independent variables used in the time series already provide a good baseline to assess the predictive ability of the various models.

4.2.5 Data Processing

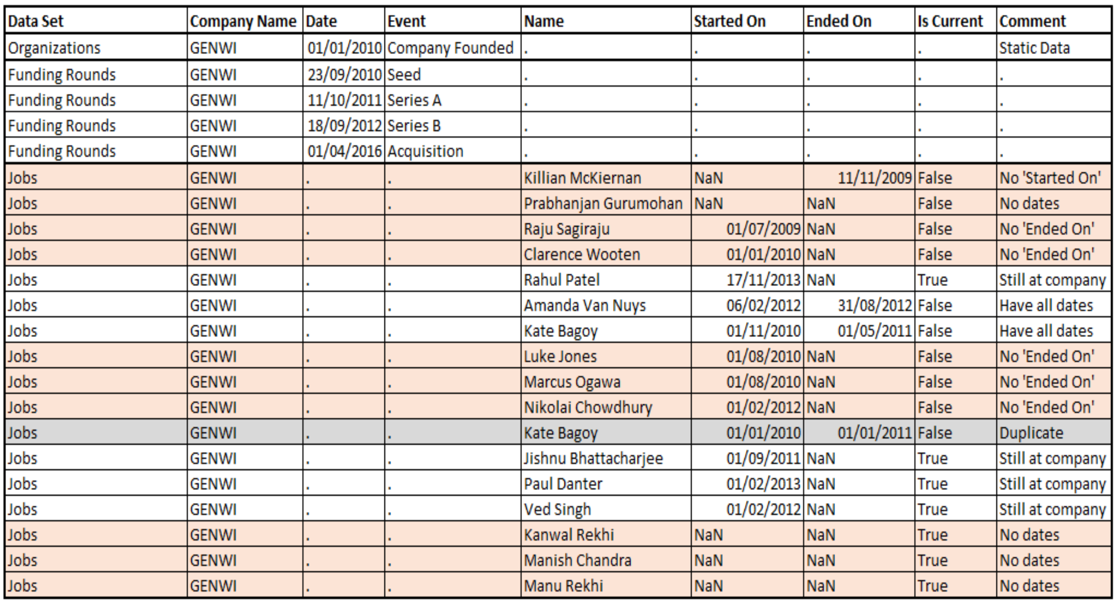
Before converting the data for the 68,428 UK and US companies' data into a time series, the data are cleaned to remove any anomalies such as startups which only have debt financing, post-IPO funding or Initial Coin Offering (ICO) funding round entries, startups which have a funding round, acquisition, IPO or closure event listed before the founding date and startups without a name. Removing these anomalies leaves 64,197 startups (9,541 UK and 54,656 US-based).

Next, the various disjointed Crunchbase data sets for these remaining startups were transformed into a single, coherent time series that aims to capture the dynamic state of startups over time and relates these to whether or not a Venture Capital investment would be successful or not (via the *target variable*). The time series is created on the principals that:

1. All dates for all events are pushed to the end of the quarter (in each year), essentially creating a uniform, *coarse* grid of dates that can be used across all companies.
2. The time series has data in each quarter, for all relevant companies, ranging from the founding date to the most recent event (IPO, acquisition, funding, none, etc.) ending on 2020/06/30.
3. The data captured at each quarter end summarises the dynamic state of the company (e.g. number of employees, gender balance, net number of employees with PhD's, funding round, etc.), as well as reflecting a handful of static variables (e.g. industry, country and social media presence).
4. The data at each quarter end has an accompanying *target* variable which indicates whether a company has both an investment entry point and exit point (*target=1*) or not (*target=0*), as per Figure 3.1.

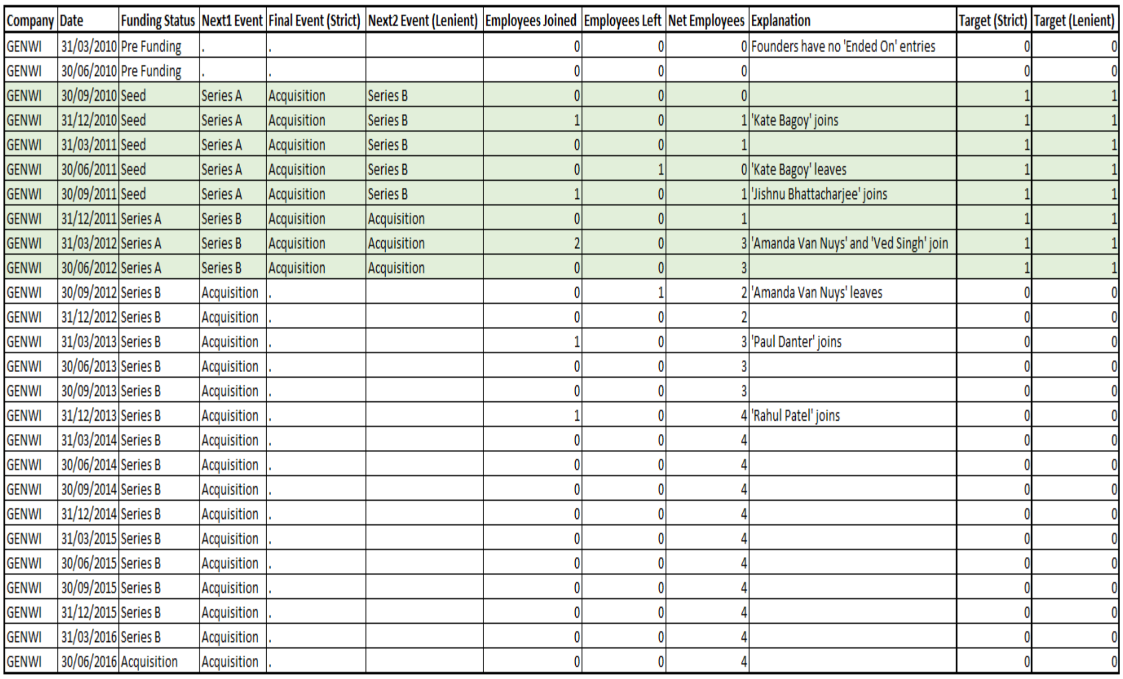
The use of a *coarse* grid of event dates (i.e. the quarter ends) can be justified by the operational realities: company changes (e.g. personnel) don't happen instantly but often take months, while Venture Capital investments take time to evaluate and execute. Therefore, pushing all time series data entries to the end of a quarter gives enough time for any company changes to be reflected in the data. Furthermore, making predictions at each quarter-end for a *target* variable that reflects events happening over the next 3 or more months, leaves enough time for Venture Capital investors to evaluate and invest in companies identified from these predictive models. A simplified example of this data processing for a single company is illustrated in Figures 4.6-4.7.

Figure 4.6 shows some of the typical features of the raw Crunchbase data: a sequence of events over time (funding/ hires) and some data flaws, such as missing values for job start and/ or end dates (highlighted in orange) and duplicate entries (highlighted in grey). For the time series data, data flaws such as missing job start and/ or end dates mean that this data has to be excluded as it cannot be correctly assigned to the corresponding timeline values.



Data Set	Company Name	Date	Event	Name	Started On	Ended On	Is Current	Comment
Organizations	GENWI	01/01/2010	Company Founded	.	.	.	.	Static Data
Funding Rounds	GENWI	23/09/2010	Seed	.	.	.	.	.
Funding Rounds	GENWI	11/10/2011	Series A	.	.	.	.	.
Funding Rounds	GENWI	18/09/2012	Series B	.	.	.	.	.
Funding Rounds	GENWI	01/04/2016	Acquisition	.	.	.	.	.
Jobs	GENWI	.	.	Killian McKiernan	NaN	11/11/2009	False	No 'Started On'
Jobs	GENWI	.	.	Prabhanjan Gurumohan	NaN	NaN	False	No dates
Jobs	GENWI	.	.	Raju Sagiraju	01/07/2009	NaN	False	No 'Ended On'
Jobs	GENWI	.	.	Clarence Wooten	01/01/2010	NaN	False	No 'Ended On'
Jobs	GENWI	.	.	Rahul Patel	17/11/2013	NaN	True	Still at company
Jobs	GENWI	.	.	Amanda Van Nuys	06/02/2012	31/08/2012	False	Have all dates
Jobs	GENWI	.	.	Kate Bagoy	01/11/2010	01/05/2011	False	Have all dates
Jobs	GENWI	.	.	Luke Jones	01/08/2010	NaN	False	No 'Ended On'
Jobs	GENWI	.	.	Marcus Ogawa	01/08/2010	NaN	False	No 'Ended On'
Jobs	GENWI	.	.	Nikolai Chowdhury	01/02/2012	NaN	False	No 'Ended On'
Jobs	GENWI	.	.	Kate Bagoy	01/01/2010	01/01/2011	False	Duplicate
Jobs	GENWI	.	.	Jishnu Bhattacharjee	01/09/2011	NaN	True	Still at company
Jobs	GENWI	.	.	Paul Danter	01/02/2013	NaN	True	Still at company
Jobs	GENWI	.	.	Ved Singh	01/02/2012	NaN	True	Still at company
Jobs	GENWI	.	.	Kanwal Rekhi	NaN	NaN	True	No dates
Jobs	GENWI	.	.	Manish Chandra	NaN	NaN	True	No dates
Jobs	GENWI	.	.	Manu Rekhi	NaN	NaN	True	No dates

Figure 4.6: Simplified Example of Raw Data from Crunchbase for GENWI



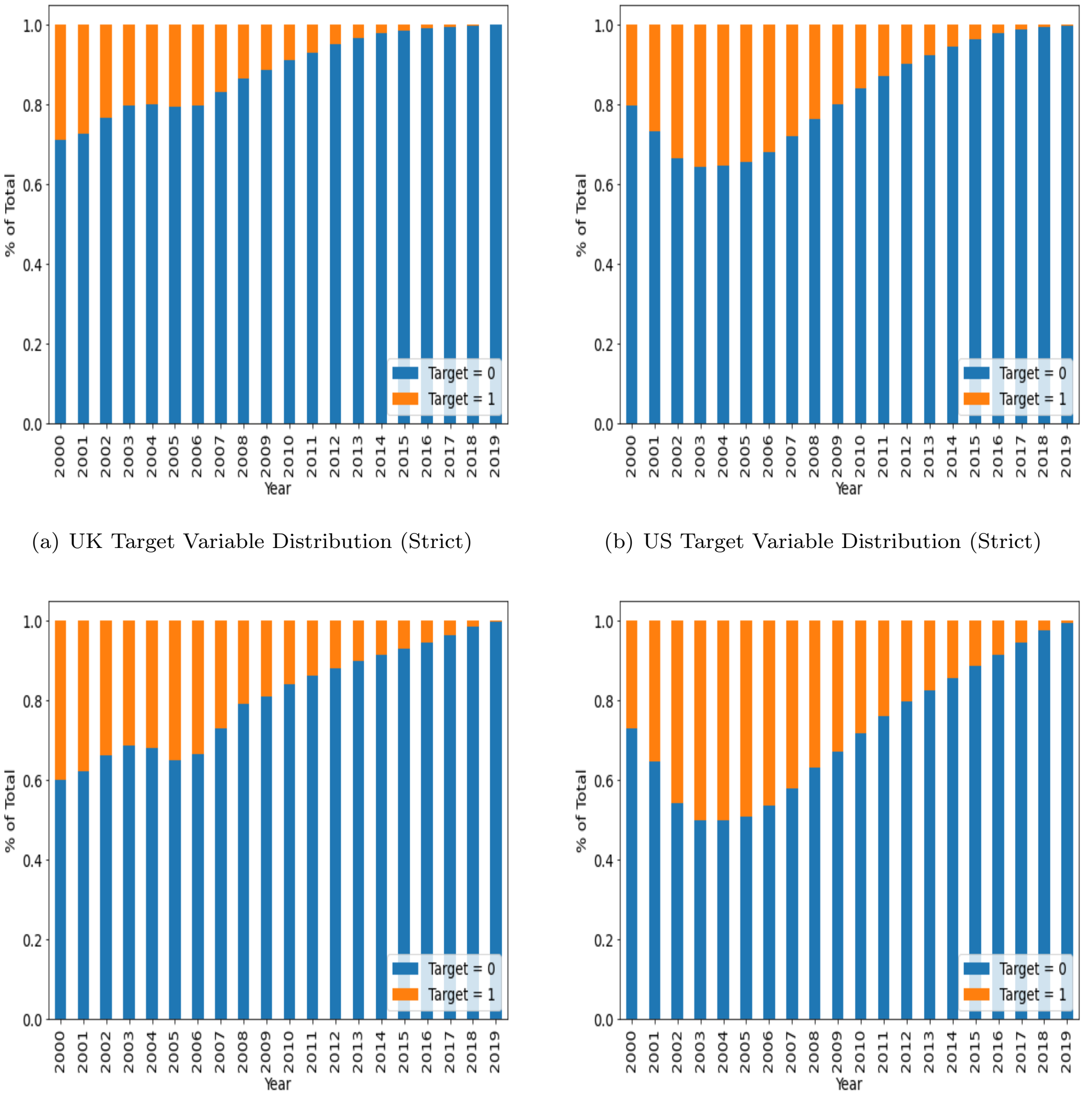
Company	Date	Funding Status	Next1 Event	Final Event (Strict)	Next2 Event (Lenient)	Employees Joined	Employees Left	Net Employees	Explanation	Target (Strict)	Target (Lenient)
GENWI	31/03/2010	Pre Funding	.	.		0	0	0	Founders have no 'Ended On' entries	0	0
GENWI	30/06/2010	Pre Funding	.	.		0	0	0		0	0
GENWI	30/09/2010	Seed	Series A	Acquisition	Series B	0	0	0		1	1
GENWI	31/12/2010	Seed	Series A	Acquisition	Series B	1	0	1	'Kate Bagoy' joins	1	1
GENWI	31/03/2011	Seed	Series A	Acquisition	Series B	0	0	1		1	1
GENWI	30/06/2011	Seed	Series A	Acquisition	Series B	0	1	0	'Kate Bagoy' leaves	1	1
GENWI	30/09/2011	Seed	Series A	Acquisition	Series B	1	0	1	'Jishnu Bhattacharjee' joins	1	1
GENWI	31/12/2011	Series A	Series B	Acquisition	Acquisition	0	0	1		1	1
GENWI	31/03/2012	Series A	Series B	Acquisition	Acquisition	2	0	3	'Amanda Van Nuys' and 'Ved Singh' join	1	1
GENWI	30/06/2012	Series A	Series B	Acquisition	Acquisition	0	0	3		1	1
GENWI	30/09/2012	Series B	Acquisition	.		0	1	2	'Amanda Van Nuys' leaves	0	0
GENWI	31/12/2012	Series B	Acquisition	.		0	0	2		0	0
GENWI	31/03/2013	Series B	Acquisition	.		1	0	3	'Paul Danter' joins	0	0
GENWI	30/06/2013	Series B	Acquisition	.		0	0	3		0	0
GENWI	30/09/2013	Series B	Acquisition	.		0	0	3		0	0
GENWI	31/12/2013	Series B	Acquisition	.		1	0	4	'Rahul Patel' joins	0	0
GENWI	31/03/2014	Series B	Acquisition	.		0	0	4		0	0
GENWI	30/06/2014	Series B	Acquisition	.		0	0	4		0	0
GENWI	30/09/2014	Series B	Acquisition	.		0	0	4		0	0
GENWI	31/12/2014	Series B	Acquisition	.		0	0	4		0	0
GENWI	31/03/2015	Series B	Acquisition	.		0	0	4		0	0
GENWI	30/06/2015	Series B	Acquisition	.		0	0	4		0	0
GENWI	30/09/2015	Series B	Acquisition	.		0	0	4		0	0
GENWI	31/12/2015	Series B	Acquisition	.		0	0	4		0	0
GENWI	31/03/2016	Series B	Acquisition	.		0	0	4		0	0
GENWI	30/06/2016	Acquisition	Acquisition	.		0	0	4		0	0

Figure 4.7: Simplified Example of Processed Time Series Data from Crunchbase for GENWI, Showing Both the Strict and Lenient Target Variable Derivation (in this case they overlap).

Figure 4.7 shows a typical example of how the Crunchbase data were converted into a time series, as well as how the target variable was assigned. The *coarse* quarterly grid for the time series captures the dynamic states of the company GENWI, while the target variable's design gives Venture Capital investors sufficient time to act on a prediction of a successful investment opportunity, as well as ensuring both an investment entry and exit point. Because we take the view that Venture Capital investments are made from series A onward and given that the data were selected on the basis of companies having at least 1 funding round, the *target* variable is set to 0 for all data points before the first funding event.

Data Imbalance

The *target variable* is skewed towards the value 0, as fewer startups present an investment target with both an initial investment and exit event. Figure 4.8 illustrates that even for the oldest startups, the *target variable* = 1 only makes up around 30% of the data using the strict definition and 40% with the more lenient definition. The cost-sensitive learning approach, discussed in Section 5.1, will be used in the models to compensate for this.



(a) UK Target Variable Distribution (Strict)

(b) US Target Variable Distribution (Strict)

(c) UK Target Variable Distribution (Lenient)

(d) US Target Variable Distribution (Lenient)

Figure 4.8: UK and US Target Variable Distributions

The differences in model performance predicting these different definitions of the target variable will be illustrated in the subsequent sections.

Chapter 5

Model Implementation and Results

We now turn our attention to the implementation of models discussed in Chapter 3 that aim to predict successful Venture Capital investments based on the time series data discussed in Chapter 4. We will first discuss the metrics used to evaluate the models, then present and interpret the numerical results and end by comparing the performance of the various models, highlighting the strengths and drawbacks in each case.

5.1 Model Implementation

All models were implemented in Python using the relevant Scikit-learn[1] libraries, taking into account the class imbalance in the *target variable* where necessary. For logistic regression, random forest, SVM and XGB models, weighted penalties for the classes were used to correct for the classification class imbalance, as this could otherwise bias the model predictions. While alternatives to overcome the class imbalance exist, such as *Synthetic Minority Oversampling Technique* (SMOTE) and *Adaptive synthetic sampling approach* (ADASYN), research by [37] shows that in almost all cases a cost-sensitive learning approach (i.e. weighted penalties) outperforms over- and undersampling techniques.

Before implementing the models, categorical variables were transformed using one-hot encoding and where appropriate, one of the dummy-variables was dropped to avoid multicollinearity in the data set. For example, categorical variables such as the industry group could be one-hot encoded without dropping any dummy variables, as a company can belong to multiple industries and each company has its own combination of industries that it belongs to (i.e. membership of a particular industry is not implied by the absence of membership in another industry). On the other hand, the funding status (seed, series A, etc.) and country (USA or UK) were encoded by dropping one of the encoded dummy variables.

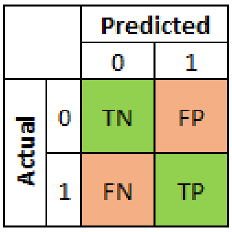
The list of variables in Table A.13 shows which variables were one-hot encoded with and without dropping of one of the dummy variables.

5.2 Evaluation Metrics and Testing

Evaluation Metrics

Each of the models from Chapter 3 was calibrated to predict the *target* variable outlined in Section 3.2. Given the imbalance of the target variable and use-case of these models, a discussion around how to evaluate the predictions made by each model is necessary. The binary nature of the *target* variable means that model predictions can be classified into 4 categories. Using terminology which labels the *target* variable value of 0 as 'negative' and a value of 1 as 'positive', translates into the definitions given in Table 5.1. These 4 categories make up what is called a confusion matrix, as represented in Figure 5.1.

[1] https://scikit-learn.org/



		Predicted	
		0	1
Actual	0	TN	FP
	1	FN	TP

Figure 5.1: Confusion Matrix

Category	Description
True Negative (TN)	Correctly predict negative outcome. (Actual = 0, Predicted = 0)
False Negative (FN)	Incorrectly predict negative outcome. (Actual = 1, Predicted = 0)
True Positive (TP)	Correctly predict positive outcome. (Actual = 1, Predicted = 1)
False Positive (FP)	Incorrectly predict positive outcome. (Actual = 0, Predicted = 1)

Table 5.1: Confusion Matrix Categories

Given the discussion around the success rate on investments for Venture Capital funds in Section 1.1, we use an optimistic estimate of a 30% success rate as a benchmark for the models being reviewed. Given this relatively low success rate, it is clear that returns in these portfolios can be improved significantly by either increasing the number of successful companies identified or (to some extent) decreasing the number of unsuccessful companies identified. Connecting these aims with the confusion matrix categories means that we want to either increase the number of true positives, decrease the number of false positives or decrease the number of false negatives identified. The 2 measures that focus on these outcomes are *precision* as defined in (5.2.1) and *recall* as defined in (5.2.2).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5.2.1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5.2.2}$$

Borrowing the descriptions of these measures from the Scikit-learn documentation, precision can be intuitively understood to measure "the ability of the classifier not to label as positive a sample that is negative"[2], while the recall can intuitively be understood to measure "the ability of the classifier to find all the positive samples"[3]. Essentially, the higher the precision and recall are, the better the classifier will be at correctly identifying and correctly finding successful Venture Capital investments.

Similar considerations in [17, Section IV-A, p. 124239] and [16, Section 5.4, p. 7], also highlight that there is no single measure which can be used to separate a good from a bad classifier. Instead if we examine the operational meaning of each of these classifiers, we can see that a high precision translates into an increased ability of the classifier to correctly identify successful investments. Since the average number of investments that the median Venture Capital fund will make is relatively small (10-30), the precision rate needs to be high as one cannot rely on the proportion of correct decisions 'averaging out' across a large sample. On the other hand, recall is a measure of how many successful investments the classifier flags to investors. Again considering that average number of Venture Capital portfolio investments is relatively small, the recall in this case is of lesser importance, as long as *enough* successful investments are flagged by the classifier for evaluation by the investor.

In summary, we prefer classifiers that have a **high precision** and a **high enough recall**, so that a sufficient number of successful investments are flagged and correctly identified.

Test Set Up

An important practical consideration for these models is that they are calibrated using historical data but must produce actionable predictions on unseen, future data. To assess the performance of the models on this basis, we split the data set into a number of mutually exclusive groups:

Group 1: Training data. All companies which have data up to a calibration date (e.g. 2010/12/31) are split into 2 groups: a training and test set. For all companies in the training set, the historical time series data up to the calibration date (e.g. 2001/01/01 - 2010/12/31) is extracted and used to calibrate the models. We will also consider varying the time

[2] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html

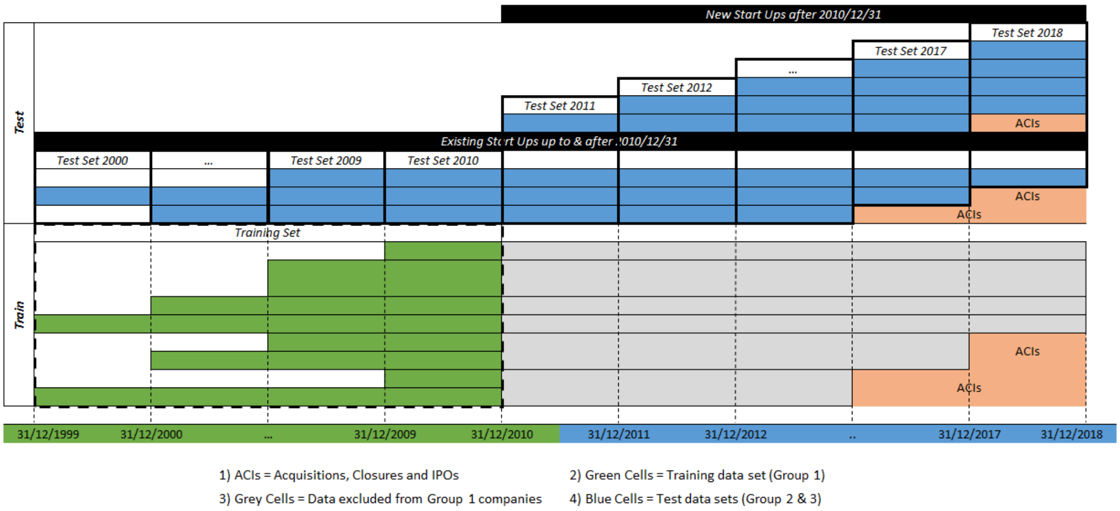
[3] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html

window used for calibration (e.g. 5 years: 2006/01/01 - 2010/12/31), to assess whether more recent data produces better predictive models based on out-of-time tests.

Group 2: Test data (historical). The time series data up to the calibration date (e.g. 2010/12/31) is extracted for all companies excluded from the training data. This data is then grouped into annual test sets (e.g. 2000, 2001, ..., 2009, 2010).

Group 3: Test data (future). For all companies from Group 2 and for all companies founded after the model calibrations date (which by definition are not in Group 1), the time series data is extracted and grouped into annual test sets (e.g. 2011, 2012, ..., 2017, 2018). The models calibrated on the training data (Group 1) are then applied to this test data (future) (Group 3) and the target variable predictions are compared to the actual target variables. It must be noted that for each future annual test set, there is less time for the model predictions to be realised (since the data was extracted on 2020/06/06), so that we can expect the precision and recall of these test data sets (future) to decline. For example, predictions of the target variable for the annual test set 2015 have ca. 5 years' time to be realised, while the predictions for the annual test set 2017 only have 3 years' time.

This grouping of training data, test data (historical) and test data (future) is represented in Figure 5.2.



1) ACIs = Acquisitions, Closures and IPOs
2) Green Cells = Training data set (Group 1)
3) Grey Cells = Data excluded from Group 1 companies
4) Blue Cells = Test data sets (Group 2 & 3)

Figure 5.2: Overview of training set, test set (historical) and test set (future) splits, for the example calibration date 2010/12/31.

All companies which become acquired, closed or have an IPO ('ACIs') are removed from the data set following the event. The roll-forward of this calibration/ test procedure to (e.g.) 2011/12/31 is constructed so that newly founded companies from 2011 are also included in the new training data (Group 1), while test sets (Groups 2 and 3) are constructed in the same way as described above, taking care not to include any companies from the training set in historical or future test sets. This roll-forward is represented in Figure 5.3.

The models will therefore aim to classify out-of-time test data, covering US and UK companies across all funding stages. The number of companies used for the training and testing are provided in Section A.5, while the list of data fields available for all the models is presented in Table A.13.

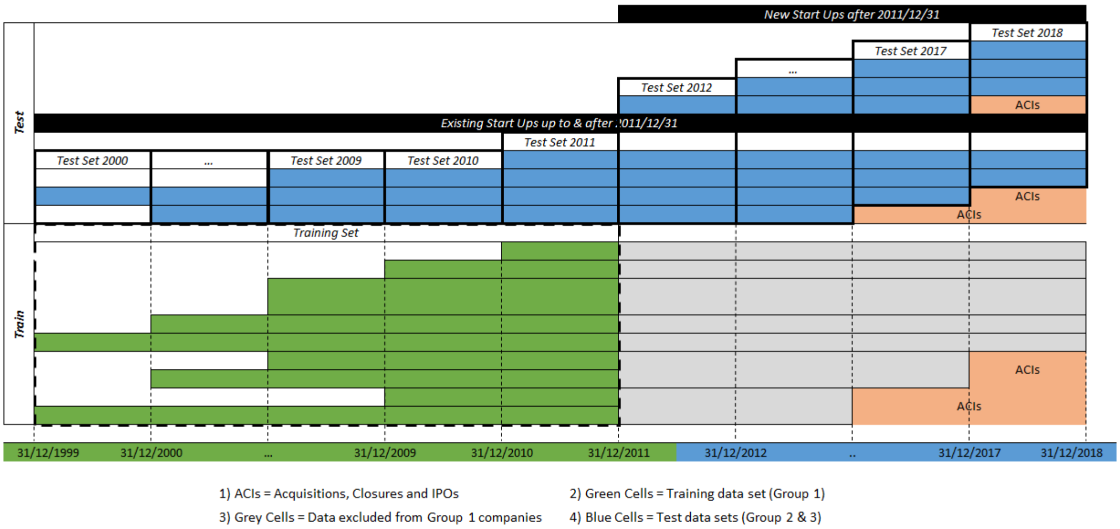

New Start Ups after 2011/12/31
Test Set 2018
Test Set 2017
...
Test Set 2012
ACIs
Test
Existing Start Ups up to & after :011/12/31
Test Set 2011
Test Set 2000
...
Test Set 2009
Test Set 2010
ACIs
ACIs
Training Set
Train
ACIs
ACIs
31/12/1999
31/12/2000
...
31/12/2009
31/12/2010
31/12/2011
31/12/2012
..
31/12/2017
31/12/2018
1) ACIs = Acquisitions, Closures and IPOs
2) Green Cells = Training data set (Group 1)
3) Grey Cells = Data excluded from Group 1 companies
4) Blue Cells = Test data sets (Group 2 & 3)

Figure 5.3: Overview of training set, test set (historical) and test set (future) roll-forward to the example calibration date 2011/12/31.

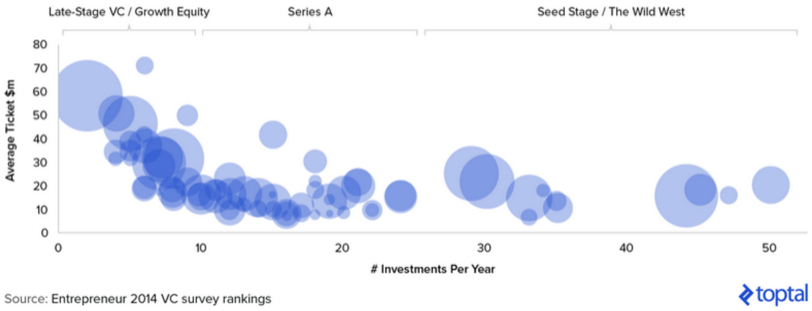
5.3 Numerical Results

Test Procedure Overview

We start by setting up all model calibrations to use a cost-sensitive learning approach. Fortunately, all Sci-kit learn libraries for the models under consideration have built-in functions that enable this approach. Even though the cost-sensitive approach compensates for the class imbalance and consequently the predicted probabilities of the target variable, we still aim to improve the precision-recall trade-off of these models by adjusting the decision point of the predicted classification probabilities. We therefore adjust the decision point to achieve a precision that is as high as possible, while the recall is still high enough for the model to identify enough companies as investment targets. Most models set the decision point for the predicted probability to 0.5 by default, so that probabilities below 0.5 are classified as a $\textit{target variable} = 0$ and vice versa. However, by increasing this decision point one increases the number of true positives and decreases the number false positives, thereby increasing the precision (since only predictions with higher probabilities of being a $\textit{target variable} = 1$ are classified as such). This increased precision comes at the cost of a lower recall, since using a higher decision point means that the model classifies fewer cases as $\textit{target variable} = 1$, which by implication creates more false negatives (actual $\textit{target variable} = 1$ cases classified as $\textit{target variable} = 0$).

Given the size of data and custom evaluation metric, only a limited number of hyperparameters are tested to see if they improve the precision-recall trade off on future test data sets. Also, the number of true positive cases identified is evaluated to see whether the model identifies enough investment opportunities for the Venture Capital investor to assess and act on, which should be at least 5-30 per year, based on Figure 5.4 from [38] (within the Series A band). Noting that the quarterly time grid in the data could mean that the same company is identified up to 4 times as an investment target within a given year, we take the conservative approach and require that the model identify at least 20-120 investment targets for each yearly test set.

To align the calibrations and assessments to how the models would be used in practice, we focus on the precision achieved on the Group 3 data, particularly for the calibration dates 2015/12/31 and 2016/12/31 and forecast years 2016 and 2017, respectively. These calibration and forecast pairs will be labelled (C/F)2015/16 and (C/F)2016/17, respectively and provide ca. 4 years' and 3 years' time for predictions to be realised. These calibration and forecast pairs should give the best indication of the model performance, given the difficulties present in evaluating the most recent company predictions (as outlined in the description of Group 3) and so will be used to select the best parameters and compare models. In the event that performance between these calibration and forecast pairs is too similar, the tie-breaker consisting of the calibration date 2017/12/31 and forecast year 2018 ((C/F)2017/18) will be used. These metrics are compared for models using different calibration windows (e.g. 2, 5, 10 or all years before the calibration date) to assess which



Source: Entrepreneur 2014 VC survey rankings

Figure 5.4: Number of Investments per Year (x-axis), by Assets Under Management (bubble size, max = $1bn, min = $0.1bn) and Average Deal Value (y-axis)

calibration window performs best in predicting future investment targets.

Finally, using these hyperparameters, the (C/F)2015/16 and (C/F)2016/17 results from 10 train-test iterations are analysed to assess whether the median precision levels surpass the 30% benchmark success rate and whether at least 20-120 investment targets are identified. To make results comparable across models and time frames, the 10 train-test iterations are performed using 10 fixed (but randomly selected) *random_state* values, which are used for the test-train data split and models (where applicable).

5.3.1 Logistic Regression (LR)

Apart from using a 'balanced' *class_weight* to implement a weighted penalty function (cost-sensitive learning approach), no other hyperparameters were adjusted. To aid with the convergence of the logistic regression calibration, the data used for training the model were scaled using Sci-kit learn StandardScaler's *fit_transform* method. This method ensures all data points are normalised by subtracting the mean and dividing by the standard deviation. Data for all test samples were normalised separately using only the test sample data to avoid any information leakage. The parameters used in this model, as well as their results, are listed in Table 5.2.

Parameter & Description	**Selected Values**	
Decision point	*Strict: 0.9875*	*Lenient: 0.9875*
Cut-off point for the classification probability. All observations with a predicted classification probability below this point were classified as *target variable* = 0, while all above this point were classified as *target variable* = 1.	These values provided the highest precision while still identifying sufficient investment targets.	
Calibration window	*Strict: 10 years*	*Lenient: 10 years*
The number of years' of data, prior to the calibration date, to use for the model calibration.	Values in the range [8,12]/ [8,12] years also performed well for the strict/ lenient target definition.	

Table 5.2: Logistic Regression Parameters and Selected Values

The median precision and recall achieved with these hyperparameters is shown in Table 5.3. Additional statistics are provided in Table A.10.

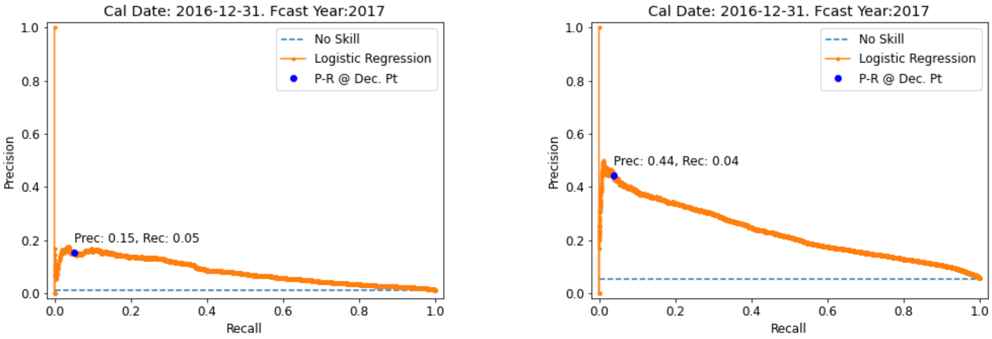
	(C/F) 2015/16		**(C/F) 2016/17**		**(C/F) 2017/18**	
	S	*L*	*S*	*L*	*S*	*L*
Median Precision	0.16	0.43	0.11	0.40	0.07	0.22
Median Recall	0.03	0.02	0.03	0.02	0.03	0.03
Median Targets	201	233	184	232	155	184

Table 5.3: Median LR Statistics for Strict (*S*) and Lenient (*L*) Target Definitions

For these hyperparameters, Figure 5.5 shows the precision-recall curves for (C/F)2016/17 and highlights how the decision point influences this trade-off, with the chosen decision point labelled 'P-R @ Dec. Pt.'. Increasing (decreasing) the decision point would increase (decrease) precision while decreasing (increasing) the recall and so move this point to the left (right). The line labelled 'No Skill' indicates the precision that would be achieved by a classifier that "predict[s] a random class or a constant class in all cases"[4], in this instance a classifier that always predicts the positive case. Hence the 'No Skill' precision is the proportion of positive cases in the dataset, as per (5.3.1).

$$\text{No Skill} = \frac{\text{Positive Cases}}{\text{Positive Cases} + \text{Negative Cases}} \tag{5.3.1}$$

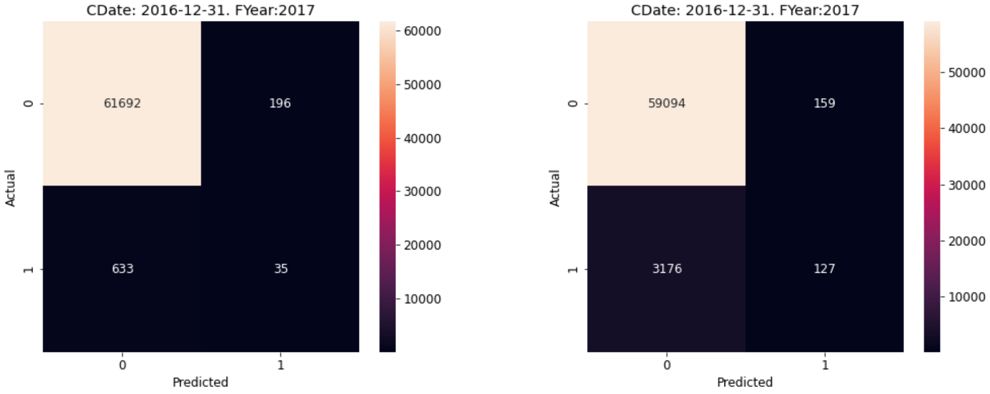
The corresponding confusion matrices in Figure 5.6 show the number of investment targets identified (predicted = 1). Figures 5.7 and 5.8 show the precision and recall levels for each year after calibration for the calibration dates 2015-2017/12/31. Outputs for different calibration dates are provided in Section A.5.



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.5: Precision-Recall Curve Examples for (C/F)2016/17



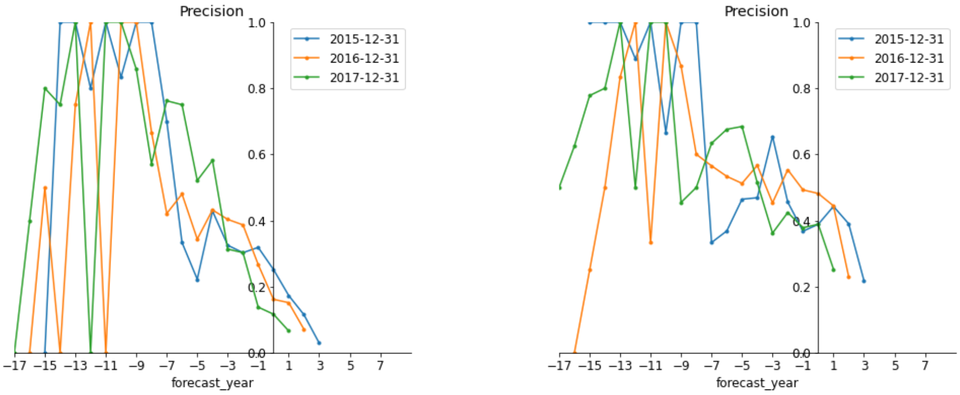
(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.6: Confusion Matrix Examples for (C/F)2016/17

The total number of outcomes reported in these confusion matrices (62,556) represents the outcomes for startups with up to 4 entries in the forecast year. In comparison, the figures reported in Table A.10 represent the unique number of startups in the forecast year. One could infer from this that startups have on average ca. 3.5 entries in the forecast year, however this figure is understated as startups founded later in the forecast year naturally contribute fewer entries.

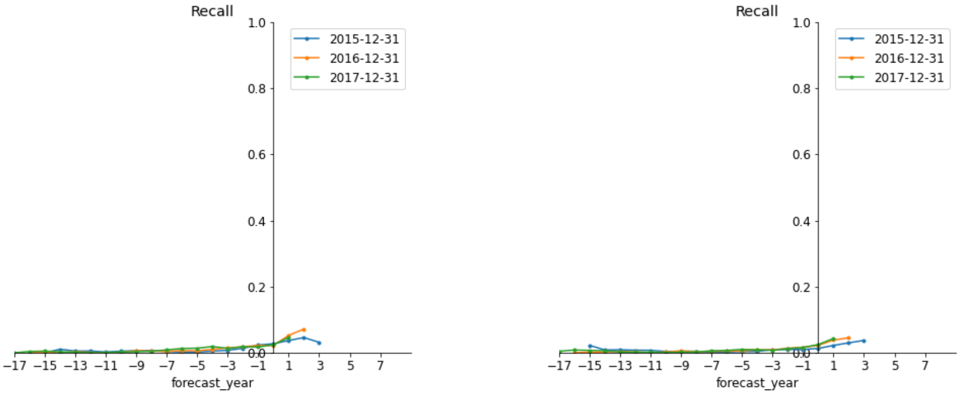
[4] https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.7: Example Precision per Year After Calibration Date. Group 2 (3) = − (+) x-axis



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.8: Example Recall per Year After Calibration Date. Group 2 (3) = − (+) x-axis

5.3.2 Random Forest (RF)

The random forest algorithm can use either a 'balanced' or 'balanced_subsample' *class_weight* to implement a weighted penalty function. The difference between these methods is that 'balanced' uses the entire training set's imbalance value as a weighting, while 'balanced_subsample' uses the imbalance weighting for the bootstrapped sample used to grow each tree. The 'balanced_subsample' was used as it performed slightly better in most cases. The default number of features $m = \lfloor\sqrt{p}\rfloor$ is used to construct the trees. The remaining hyperparameters tested and their results are provided in Table 5.4

Parameter & Description	**Selected Values**	
n_estimators	*Strict: 75*	*Lenient: 20*
Number of trees to use in the random forest, corresponding to B in $\{T_b\}_1^B$ from Algorithm 2. The default value for this is set at 100	Values in the range [50,100]/ [15,25] also performed well for the strict/ lenient target definition. Before and after these ranges, the precision and recall both decrease.	

Parameter & Description	**Selected Values**	
Max_depth	*Strict: 20*	*Lenient: 15*
The longest path between the root node and the leaf node. The default value for this is set to *None*, which means the tree is expanded until all leaves are pure or until all leaves contain less than a specified number of samples	Values in the range [16,20]/ [10,20] also performed well for the strict/ lenient target definition.	
Decision point	*Strict: 0.50*	*Lenient: 0.70*
As per Table 5.2.	These values provided the highest precision while still identifying sufficient investment targets.	
Calibration window	*Strict: 10 years*	*Lenient: 10 years*
As per Table 5.2.	10 years was consistently the best performing calibration window for the strict definition. Although no single value performed consistently well for the lenient definition, 10 years also produced the best compromise across the (C/F)2015/16, (C/F)2016/17 and (C/F)2017/18 tests.	

Table 5.4: Random Forest Hyperparameters and Selected Values

The median precision and recall achieved with these hyperparameters is shown in Table 5.5. Additional statistics are provided in Table A.11.

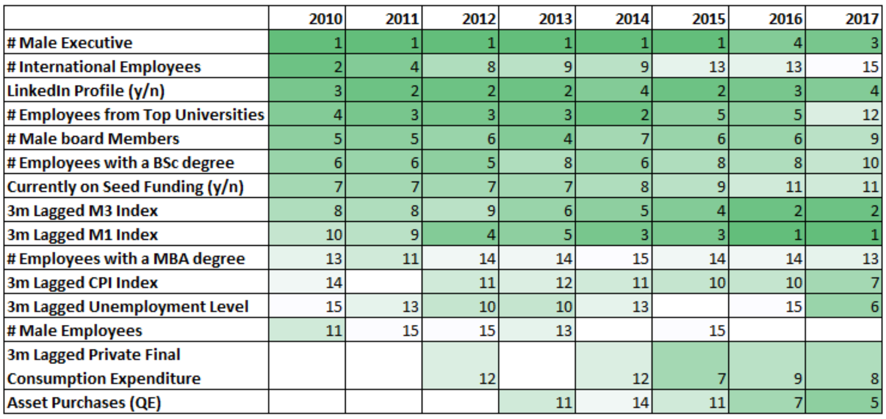
	(C/F) 2015/16		**(C/F) 2016/17**		**(C/F) 2017/18**	
	S	*L*	*S*	*L*	*S*	*L*
Median Precision	0.24	0.45	0.23	0.37	0.00	0.15
Median Recall	0.05	0.12	0.03	0.09	0.00	0.02
Median Targets	220	1,223	74	791	3	196

Table 5.5: Median RF Statistics for Strict (S) and Lenient (L) Target Definitions

Sci-kit learn allows for an analysis of feature importance in the calibrated RF-models, with Figures 5.9-5.10 showing the top 15 features over time (by calibration year). We use the standard feature-importance classifier based on mean decrease in Gini impurity even though it is indicated to be ”biased; i.e. it tends to inflate the importance of continuous or high-cardinality categorical variables”[5]. More advanced analysis of feature importance is possible, such as permutation and drop-column importance however, we omit this as the paper focuses on comparing the performance of machine learning models to predict successful investment targets.

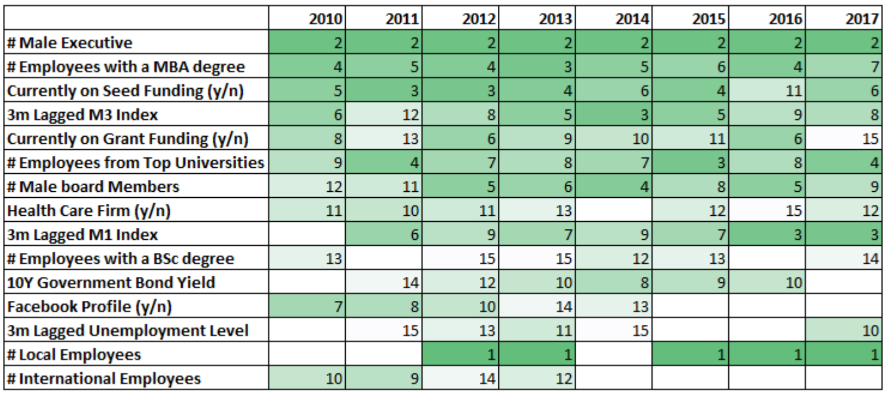
We present the same outputs as for the logistic regressions, showing precision-recall (Figure 5.11) and confusion matrices (Figure 5.12) for (C/F)2016/17 as well as the precision and recall levels for each year after the calibration dates 2015-2017/12/31 (Figures 5.13 and 5.14). Additional outputs are again listed in Section A.5.

[5] https://explained.ai/rf-importance/index.html



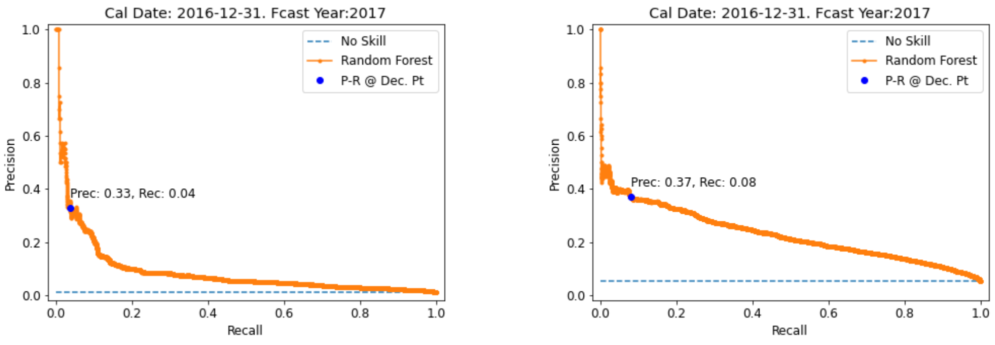
	2010	2011	2012	2013	2014	2015	2016	2017
# Male Executive	1	1	1	1	1	1	4	3
# International Employees	2	4	8	9	9	13	13	15
LinkedIn Profile (y/n)	3	2	2	2	4	2	3	4
# Employees from Top Universities	4	3	3	3	2	5	5	12
# Male board Members	5	5	6	4	7	6	6	9
# Employees with a BSc degree	6	6	5	8	6	8	8	10
Currently on Seed Funding (y/n)	7	7	7	7	8	9	11	11
3m Lagged M3 Index	8	8	9	6	5	4	2	2
3m Lagged M1 Index	10	9	4	5	3	3	1	1
# Employees with a MBA degree	13	11	14	14	15	14	14	13
3m Lagged CPI Index	14		11	12	11	10	10	7
3m Lagged Unemployment Level	15	13	10	10	13		15	6
# Male Employees	11	15	15	13		15		
3m Lagged Private Final Consumption Expenditure			12		12	7	9	8
Asset Purchases (QE)				11	14	11	7	5

Figure 5.9: Top 15 RF Model Features over time. Strict Target Variable Definition



	2010	2011	2012	2013	2014	2015	2016	2017
# Male Executive	2	2	2	2	2	2	2	2
# Employees with a MBA degree	4	5	4	3	5	6	4	7
Currently on Seed Funding (y/n)	5	3	3	4	6	4	11	6
3m Lagged M3 Index	6	12	8	5	3	5	9	8
Currently on Grant Funding (y/n)	8	13	6	9	10	11	6	15
# Employees from Top Universities	9	4	7	8	7	3	8	4
# Male board Members	12	11	5	6	4	8	5	9
Health Care Firm (y/n)	11	10	11	13		12	15	12
3m Lagged M1 Index		6	9	7	9	7	3	3
# Employees with a BSc degree	13		15	15	12	13		14
10Y Government Bond Yield		14	12	10	8	9	10	
Facebook Profile (y/n)	7	8	10	14	13			
3m Lagged Unemployment Level		15	13	11	15			10
# Local Employees			1	1		1	1	1
# International Employees	10	9	14	12				

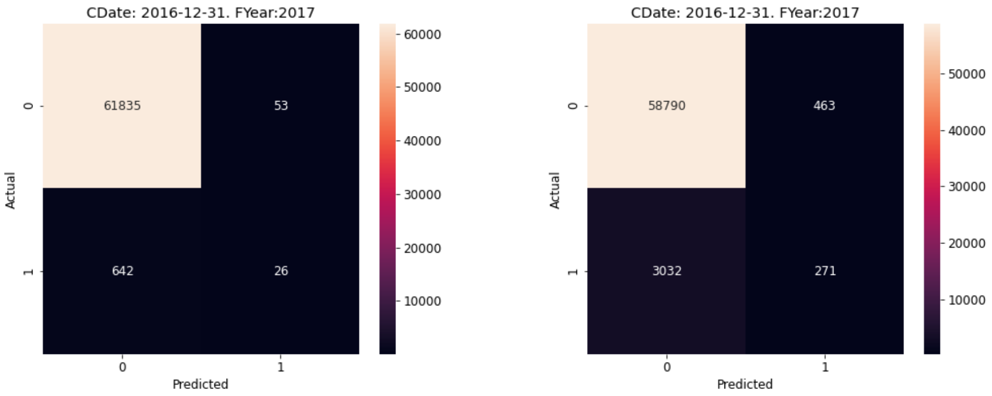
Figure 5.10: Top 15 RF Model Features over time. Lenient Target Variable Definition



(a) Strict Target Variable

(b) Lenient Target Variable

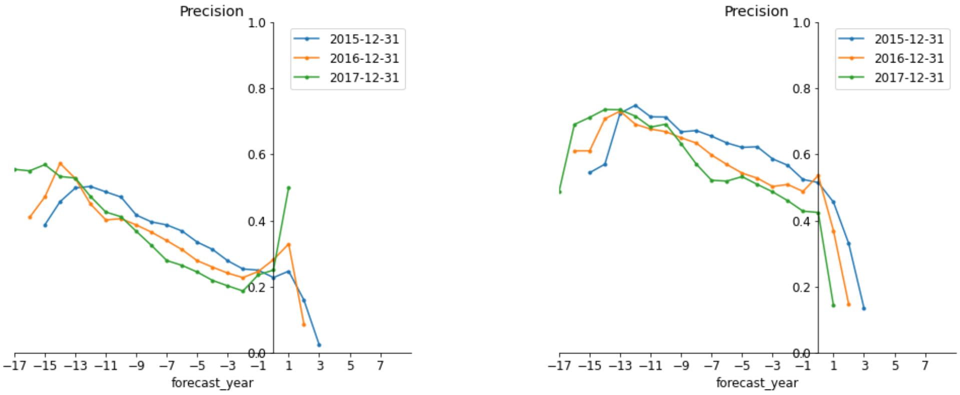
Figure 5.11: Precision-Recall Curve Examples for (C/F)2016/17



(a) Strict Target Variable

(b) Lenient Target Variable

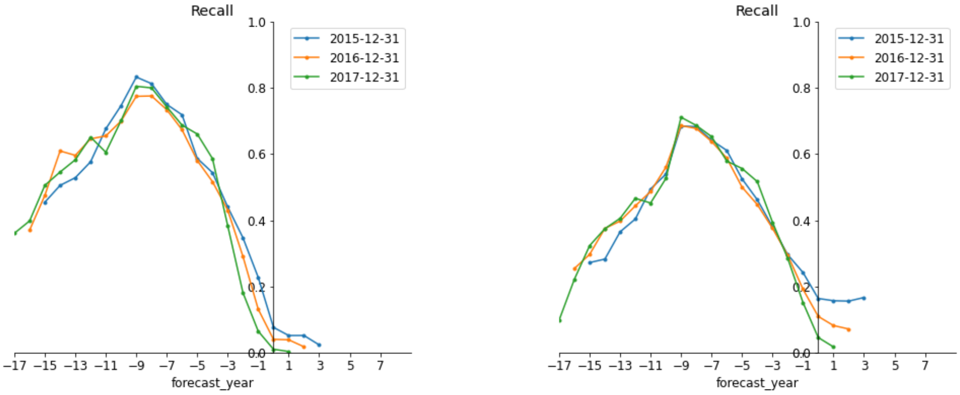
Figure 5.12: Confusion Matrix Examples for (C/F)2016/17



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.13: Example Precision per Year After Calibration Date. Group 2 (3) = − (+) x-axis



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.14: Example Recall per Year After Calibration Date. Group 2 (3) = − (+) x-axis

5.3.3 Support Vector Machines (SVM)

A 'balanced' *class_weight* is used to implement a weighted penalty function. Given that SVM tries to maximise the distance between the separating hyperplane and the observations, it is important to scale the input data to avoid features with larger values dominating the SVM calibration. Therefore, as with logistic regression, the data used for training the SVM model were scaled using Sci-kit learn StandardScaler's *fit_transform* method. This method ensures all data points are normalised by subtracting the mean and dividing by the standard deviation. Data for all test samples were normalised separately using only the test sample data to avoid information leakage.

While SVM has a rich theoretical background and can produce high accuracy results, it also has a number of drawbacks. For example, SVM is not well-suited to large data sets, such as the one we are using in this paper, as the standard SVM algorithm has $O(n^3)$ time and $O(n^3)$ space complexities [39]. Given the 124 independent variables and large data set size, the training time for the SVM was unfeasible and so this algorithm was implemented using a sub-sampling technique, which relies on choosing a stratified random sample of all companies in the calibration window. However, even when limiting the data to a stratified sample of 1,000 companies, the SVM training time was far longer than any of the other models under consideration and so was prohibitively time-consuming to calibrate and tune to produce satisfactory results. Nevertheless, the hyperparameters were tuned using a stratified sample of 1,000 companies, and then validated using a larger stratified sample of 5,000 companies. Only a single train-test iteration was performed. This approach produced unsatisfactory results as the hyperparameters were sensitive to the 1,000 companies chosen for the tuning sample and subsequently, the validation using the larger sample of 5,000 companies.

Another consideration is the fact that SVM's separating hyperplane produces a binary classification output for the *target variable*, which in this case may not be well-suited to the task at hand. This means that tuning the model to increase precision can only be achieved by tuning the hyperparameters (in particular C and γ). However given the difficulty and time-consuming nature of this endeavour, no satisfactory precision-recall values beyond those in Table 5.6 could be found within a practical amount of time. The hyperparameters tested to find the optimal precision-recall trade off on future test data sets and their results are provided in Table 5.6.

Parameter & Description	**Selected Values**	
C	*Strict: 0.9*	*Lenient: 1.0*
C is listed as a regularisation parameter, where the strength of the regularisation is inversely proportional to C ($C > 0$). This means that, as per (3.3.14), the parameter controls the degree to which misclassifications are allowed in the model calibration and so small values of C create wider margins (which translates to smaller slack variables) and vice versa. It's default value is set to 1.0	Values in the range [0.9,1.0]/ [0.9,1.0] performed best for the strict/ lenient target definition, although only marginal precision gains could be made.	
γ	*Strict: 0.00006*	*Lenient: 0.00006*
This parameter applies to the radial basis function (rbf), polynomial and sigmoid kernels, whose influence can be seen in the equations (3.3.19) and (3.3.20). Focusing on the rbf, γ defines how far the influence of a single training example reaches. Hence γ can be seen as the inverse of the radius of influence of samples selected by the model as support vectors. The default value is set to 'scale', which sets $\gamma = \frac{1}{(n\_features * X.var())}$	Values in the range [0.00004,0.00007]/ [0.00004,0.00007] performed best for the strict/ lenient target definition. The resulting precision was very sensitive to this parameter and only marginal precision gains could be made.	

Parameter & Description	**Selected Values**	
Kernel	*Strict: rbf*	*Lenient: rbf*
The kernels used for the generalised inner products, as described in Section 3.3.3, with examples as per (3.3.19) and (3.3.20). Options available are linear, rbf, polynomial and sigmoid kernels, with the default value being the rbf	Rbf kernel performed best for the strict/ lenient target definition.	
Calibration window	*Strict: 5 years*	*Lenient: 10 years*
As per Table 5.2.	Although no single value performed consistently well, these values produced the best compromise across the (C/F)2015/16, (C/F)2016/17 and (C/F)2017/18 tests.	

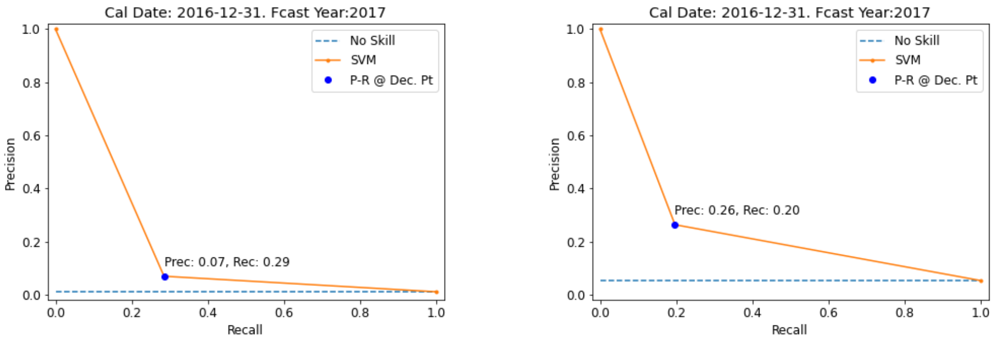
Table 5.6: SVM Hyperparameters and Selected Values

The precision and recall achieved with these hyperparameters is shown in Table 5.7. Due to the excessive run-times for this model, only a single train-test iteration was performed.

	(C/F) 2015/16		**(C/F) 2016/17**		**(C/F) 2017/18**	
	S	*L*	*S*	*L*	*S*	*L*
Precision	0.12	0.36	0.07	0.26	0.03	0.11
Recall	0.14	0.09	0.29	0.20	0.20	0.28
Targets	3,514	2,946	7,111	6,594	4,236	9,973

Table 5.7: Example SVM Output for Strict (S) and Lenient (L) Target Definitions

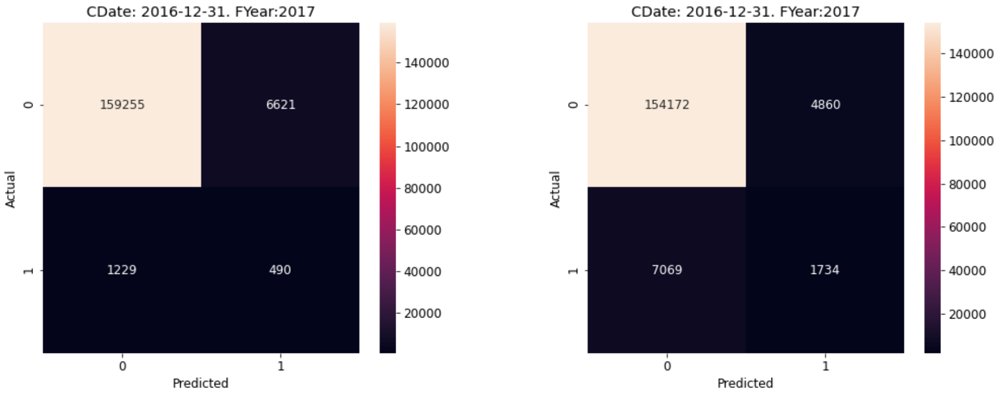
We present the same outputs as in previous sections for the 1,000 company stratified tuning sample (5,000 company stratified validation sample in Section A.5), showing precision-recall (Figure 5.15) and confusion matrices (Figure 5.16) for (C/F)2016/17 as well as the precision and recall levels for each year after the calibration dates 2015-2017/12/31 (Figures 5.17 and 5.18). Additional outputs are again listed in Section A.5.



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.15: Precision-Recall Curve Examples for (C/F)2016/17

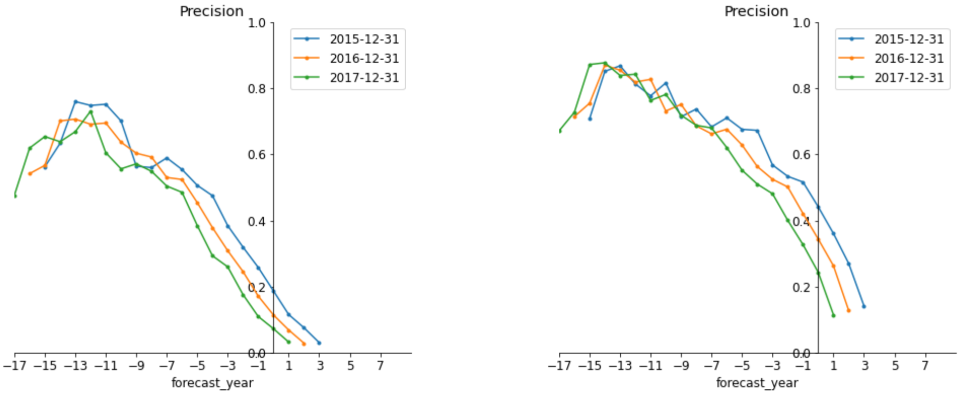


(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.16: Confusion Matrix Examples for (C/F)2016/17

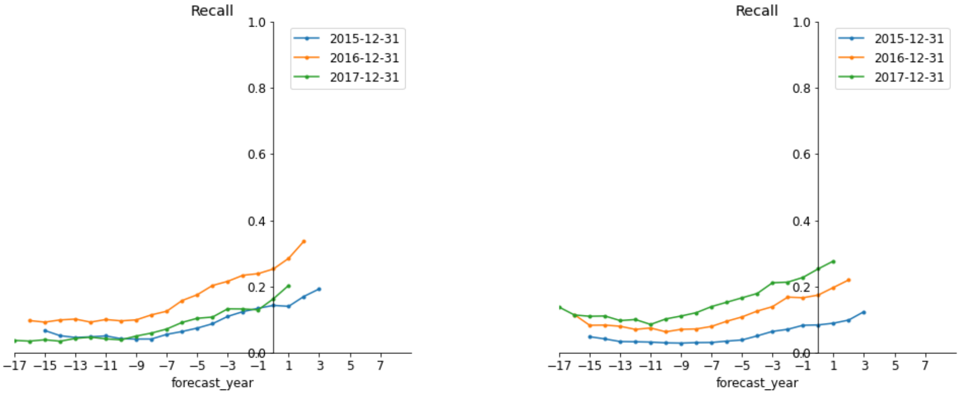
Note: The total number of outcomes in the confusion matrices in Figure 5.16 is higher, as we include all startups not contained in the downsampled training set in the subsequent year's test set (subject to not being ACIs).



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.17: Example Precision per Year After Calibration Date. Group 2 (3) = − (+) x-axis



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.18: Example Recall per Year After Calibration Date. Group 2 (3) = − (+) x-axis

5.3.4 Extreme Gradient Boosting (XGB)

A cost-sensitive learning implementation of the XGB algorithm is achieved by first calculating the ratio between the number of *target variable* = 0 and *target variable* = 1 data points and then using this ratio for the *scale_pos_weight* parameter. The hyperparameters tested and their results are provided in Table 5.8

Parameter & Description	Selected Value	
Max_depth	*Strict: 6*	*Lenient: 9*
As per Table 5.4, except that the default value is set to 6.	Values in the range [6,9]/ [6,9] also performed well for the strict/ lenient target definition.	
Min_child_weight	*Strict: 20*	*Lenient: 20*
This is the minimum sum of the hessians in a leaf node during the tree construction. This corresponds to the minimum value that (e.g.) $\sum_{i=1}^{N} p_i^{m-1}(1-p_i^{m-1})$ needs to have in (3.3.33). The default value is set to 1.	Values in the range [20,50]/ [20,50] also performed well for the strict/ lenient target definition.	
γ	*Strict: 1.00*	*Lenient: 3.33*
Gain threshold each node-split is required to achieve in order to avoid pruning, where the gain is given as in (3.3.34). Any node-split that does not achieve this minimum gain is removed from the tree. The default value is set to 0.	Values in the range [0,10]/ [0,10] also performed well for the strict/ lenient target definition.	
Colsample_bytree	*Strict: 0.25*	*Lenient: 0.25*
The proportion of features (columns) to use in constructing each XGB-tree, similar to the RF-algorithm. The default value is set to 1.	Values in the range [0.25,0.50]/ [0.25,0.50] also performed well for the strict/ lenient target definition.	
Decision point	*Strict: 0.86*	*Lenient: 0.85*
As per Table 5.2.	These values provided the highest precision while still identifying sufficient investment targets.	
Calibration window	*Strict: 7.5 years*	*Lenient: 7.5 years*
As per Table 5.2.	Values in the range [5,10]/ [5,10] years also performed well for the strict/ lenient target definition.	

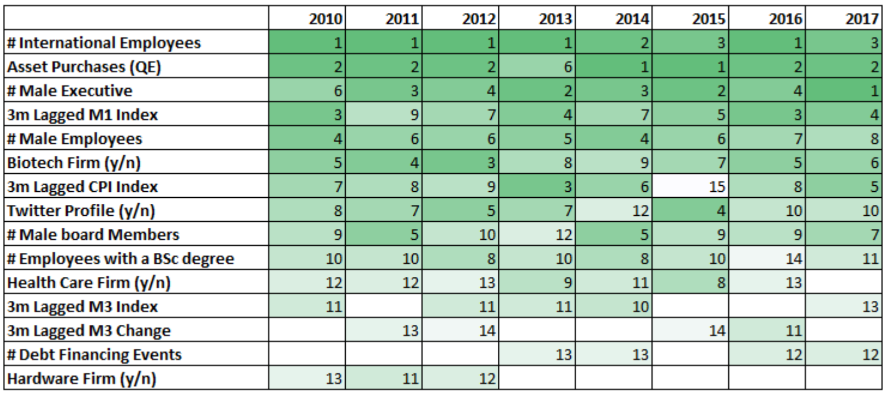
Table 5.8: XGB Hyperparameters and Selected Values

The median precision and recall achieved with these hyperparameters is shown in Table 5.9. Additional statistics are provided in Table A.12.

	(C/F) 2015/16		(C/F) 2016/17		(C/F) 2017/18	
	S	L	S	L	S	L
Median Precision	0.49	0.52	0.23	0.41	0.09	0.16
Median Recall	0.01	0.04	0.01	0.03	0.01	0.03
Median Targets	30	368	16	261	31	267

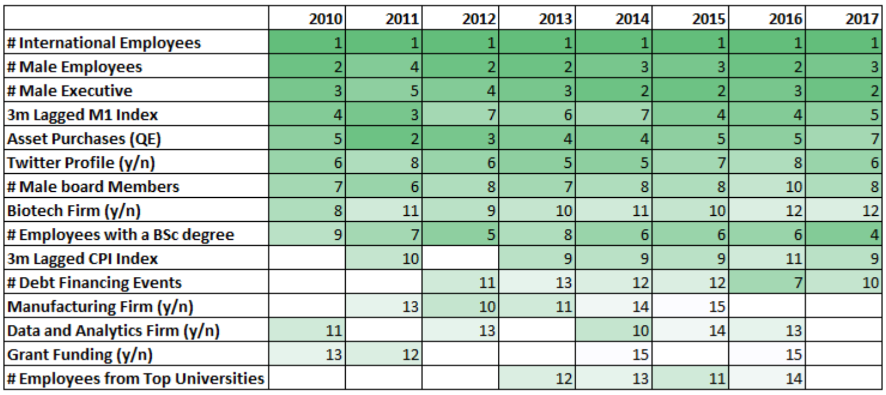
Table 5.9: Median XGB Statistics for Strict (S) and Lenient (L) Target Definitions

XGB has built-in functionality to extract the feature importance of the calibrated XGB-models, with Figures 5.19-5.20 showing the top 15 features over time (by calibration year). We use the standard feature-importance classifier based on the number of times a feature appears in a tree. We again omit more detailed feature analysis for the same reasons provided in Section 5.3.2.



	2010	2011	2012	2013	2014	2015	2016	2017
# International Employees	1	1	1	1	2	3	1	3
Asset Purchases (QE)	2	2	2	6	1	1	2	2
# Male Executive	6	3	4	2	3	2	4	1
3m Lagged M1 Index	3	9	7	4	7	5	3	4
# Male Employees	4	6	6	5	4	6	7	8
Biotech Firm (y/n)	5	4	3	8	9	7	5	6
3m Lagged CPI Index	7	8	9	3	6	15	8	5
Twitter Profile (y/n)	8	7	5	7	12	4	10	10
# Male board Members	9	5	10	12	5	9	9	7
# Employees with a BSc degree	10	10	8	10	8	10	14	11
Health Care Firm (y/n)	12	12	13	9	11	8	13	
3m Lagged M3 Index	11		11	11	10			13
3m Lagged M3 Change		13	14			14	11	
# Debt Financing Events				13	13		12	12
Hardware Firm (y/n)	13	11	12					

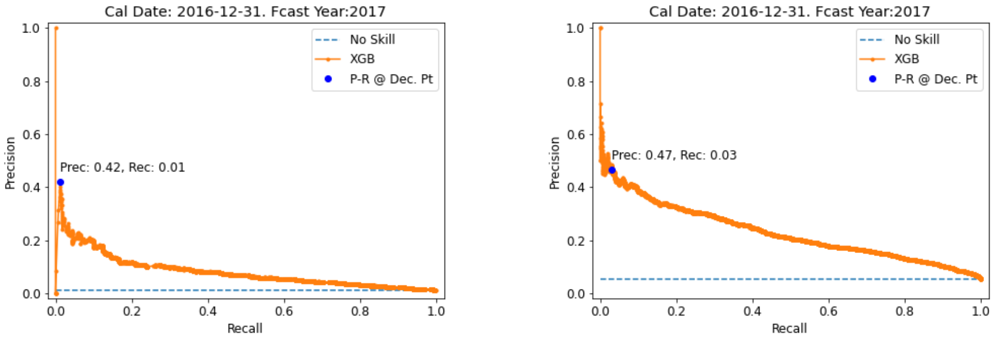
Figure 5.19: Example Top 15 XGB Model Features Over Time. Strict Target Variable Definition



	2010	2011	2012	2013	2014	2015	2016	2017
# International Employees	1	1	1	1	1	1	1	1
# Male Employees	2	4	2	2	3	3	2	3
# Male Executive	3	5	4	3	2	2	3	2
3m Lagged M1 Index	4	3	7	6	7	4	4	5
Asset Purchases (QE)	5	2	3	4	4	5	5	7
Twitter Profile (y/n)	6	8	6	5	5	7	8	6
# Male board Members	7	6	8	7	8	8	10	8
Biotech Firm (y/n)	8	11	9	10	11	10	12	12
# Employees with a BSc degree	9	7	5	8	6	6	6	4
3m Lagged CPI Index		10		9	9	9	11	9
# Debt Financing Events			11	13	12	12	7	10
Manufacturing Firm (y/n)		13	10	11	14	15		
Data and Analytics Firm (y/n)	11		13		10	14	13	
Grant Funding (y/n)	13	12			15		15	
# Employees from Top Universities				12	13	11	14	

Figure 5.20: Example Top 15 XGB Model Features Over Time. Lenient Target Variable Definition

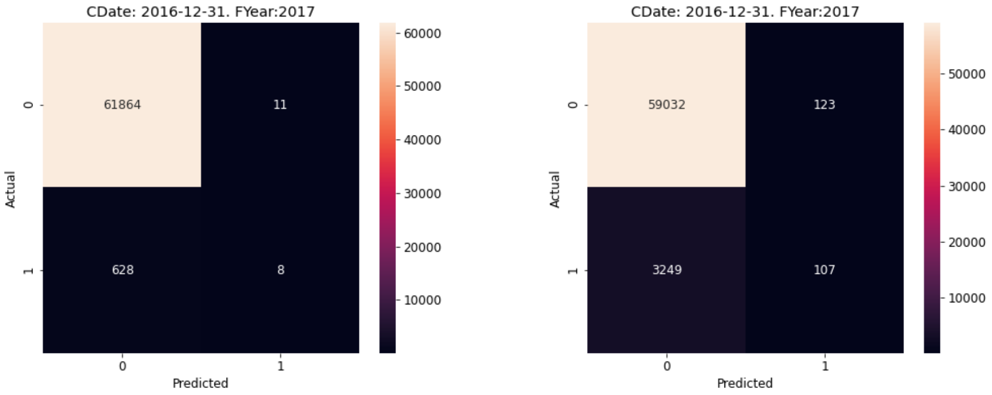
We present the same outputs as in previous sections, showing precision-recall (Figure 5.21), confusion matrices (Figure 5.22) for (C/F)2016/17 as well as the precision and recall levels for each year after the calibration dates 2015-2017/12/31 (Figures 5.23 and 5.24). Additional outputs are again listed in Section A.5.



(a) Strict Target Variable

(b) Lenient Target Variable

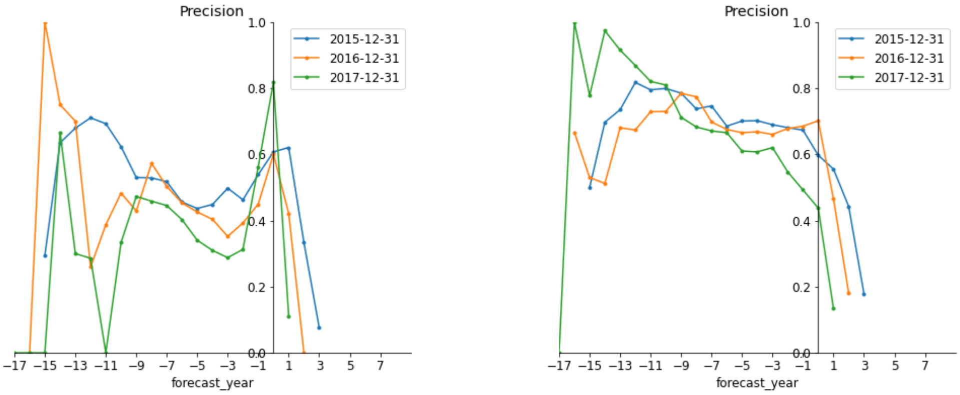
Figure 5.21: Precision-Recall Curve Examples for (C/F)2016/17



(a) Strict Target Variable

(b) Lenient Target Variable

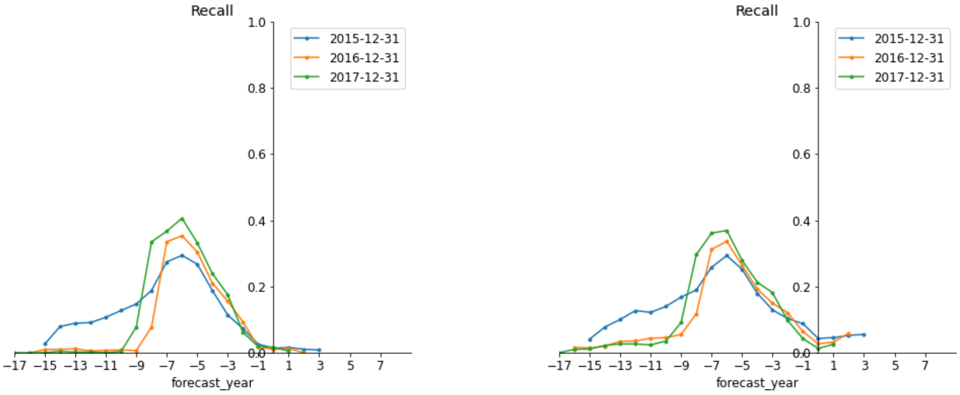
Figure 5.22: Confusion Matrix Examples for (C/F)2016/17



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.23: Example Precision per Year After Calibration Date. Group 2 (3) = − (+) x-axis



(a) Strict Target Variable

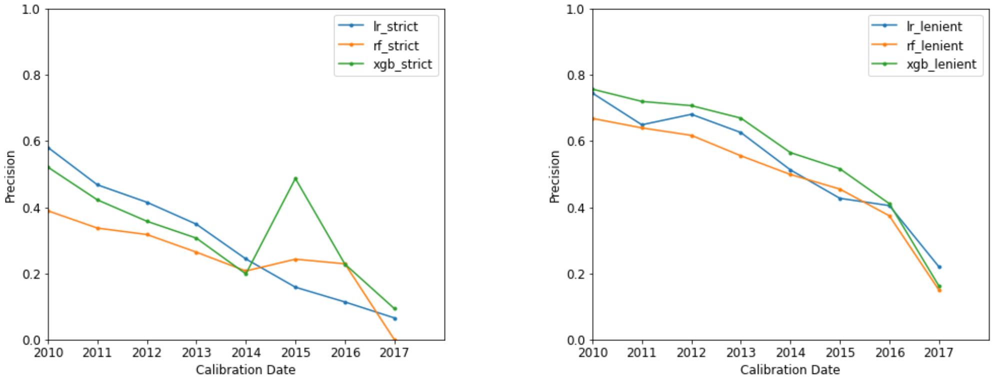
(b) Lenient Target Variable

Figure 5.24: Example Recall per Year After Calibration Date. Group 2 (3) = − (+) x-axis

5.4 Summary of Results

Model Forecasts

We focus on the results obtained for the calibration-forecast pairs (C/F)2015/16 and (C/F)2016/17 (over 10 train-test iterations), as these are recent data points and the predictions have 4 and 3 years, respectively, to materialise. We exclude results for SVM, as only 1 train-test iteration could be run due to the prohibitive model training time requirement. To compare the predictive ability of various models, we plot their precision for the first year after the calibration date. Figure 5.25 shows these results for all calibration dates from 2010-2017/12/31. For example, the point *2016* on the x-axis corresponds to the (C/F)2016/17 pair, *2015* to the (C/F)2015/16 pair, etc.. The last 3 points on the x-axis correspond to the precision values from Tables 5.3, 5.5 and 5.9. Full statistics for the 10 train-test iterations and a box-plot of the precision rates are given in Tables A.10, A.11 and A.12, as well as Figures A.7, A.9 and A.13



(a) Strict Target Variable

(b) Lenient Target Variable

Figure 5.25: Median Precision per Calibration-Forecast Pair: 2010-2017

While this graph represents the precision each model is able to achieve in the first year after calibration, each precision value is determined for an increasing number of years over which the predictions are allowed to materialise (right = lowest (2 years), left = highest (9 years)). Reading this graph from right to left could then be indicative of the proportion of the machine learning models' year 1 predictions that would materialise as correctly identified investment targets, given a certain amount of time. The same graphs could be constructed for the various models' year 2, 3, etc. forecasts, however this will always decrease the amount of time available for the predictions to materialise. For this reason (and others discussed below) we will analyse the precision levels of the models' year 1 forecasts and discuss the results and their interpretation for each target variable separately.

Strict Target Variable

We can see that the XGB and random forest (RF) models achieved the highest precision levels in the first 3 calibration-forecast pairs. However, the precision declines for the (C/F)2014/15 pair, which appears to be an outlier as the precision levels rebound and increase to around 50% for XGB and 40% for RF. Looking at the precision values furthest to the left (towards 2010) in Figure 5.25(a), we see that logistic regression (LR) actually achieves a higher precision value. Using the interpretation discussed above, when reading the graph from right to left, this would suggest that, given enough time, the LR model's predictions actually produce the highest proportion of correctly identified investment targets (although closely matched by XGB).

However, what this interpretation ignores is the fact that there could be (and likely are) fundamental changes in the underlying data that generate these results. Thus we must decide what is more important: a model that has a higher precision over a shorter outcome period or one that, given enough time, has the highest precision level eventually? We take the view that a higher precision over a shorter period of time is more important, since:

1. The precision rates achieved on older calibration-forecast pairs may not be representative of the precision rates these models can achieve on more recent calibration-forecast pairs as the volume of data and breadth of companies covered by Crunchbase is much greater in the last few years than (e.g.) 10 years ago.
2. Assuming the exit event (acquisition or IPO) generates the same value for investors regardless of when it happens, then a shorter holding period increases the internal rate or return (IRR) for investors and so improves the Venture Capital portfolio returns.
3. Regime changes in the underlying data (such as macro-economic, regulatory or political policy changes that can impact the success rate of startups) are likely to have a smaller influence on the performance of the model in the short term. Hence the precision rate over longer time periods (older calibration dates) can be heavily influenced by historical factors that may not apply going forward.

Therefore, focusing on the (C/F)2015/16 and (C/F)2016/17 pairs suggests that XGB and RF-models are the best suited for predicting successful investment targets. Furthermore, using the 30% success benchmark discussed in Section 1.1 we can also say that the average/ median models trained using the XGB algorithm actually outperform the average Venture Capital firm even over a relatively short 4 year outcome period (excluding any further upside from additional model predictions materialising as correct) by over 50% (30% vs. average of 46%/ median of 49%).

Table A.12 also shows that the same model would identify a median of 30 investment targets for (C/F)2015/16 and a median of 16 for (C/F)2016/17. As discussed, we would conservatively divide this number by 4 to compensate for the possibility of the same targets being identified in each of the 4 quarters in a year, which would still leave 7 and 4 investment targets, which a suitably high volume given the discussion in Section 5.3. Given the precision levels achieved, the decision-point could be reduced to produce more targets.

It is perhaps somewhat surprising that these models are able to identify a relatively high proportion of successful investment targets using a relatively simple data set. However, as we've seen in Section 3.1, evaluating the management/ founding team in a company is considered to be (one of) the most important factors in Venture Capital investing. Thus it is possible that these models, which include Crunchbase data covering people, degrees and job movements, are able to establish causal links between these factors and a company's success. It is therefore interesting to see that of the 5 most important features for the XGB model, 3 consistently relate to the company employee profile (e.g. number of international employees, number of male executives and number of male employees) while 2 are macro-economic features relating to the level of monetary stimulus from the government (asset purchases (quantitative easing) and the total M1 money supply) - as shown in Figure 5.19. The RF-model shows a slightly different picture, with the number of international employees becoming less important over time, while the level of monetary stimulus from the government becoming more important over time. The top 5 features still relate to the employee profile (number of male executives and employees from top universities (see Table A.14)), monetary stimulus (total M1 and M3 money supplies) while also including the presence on LinkedIn as an important feature - as shown in in Figure 5.9.

Lenient Target Variable

For this target variable definition, XGB again appears to be the best performing model, although the LR model is not very far off, suggesting either one could be used. We reach most of the same conclusions here as we did for the *strict* target variable definition, however the interpretation using the *lenient* target variable definition is slightly different. As discussed previously, a company reaching another funding round is not necessarily an indicator of success for Venture Capital investors. Nevertheless, identifying companies with the potential to reach a next funding round (or exit) allows Venture Capital investors to widen the net on sourcing investment targets. In particular, investors with a very active or superior investment management strategy might prefer to identify targets of this nature and apply their skills in order to generate successful investments. As their is no benchmark available for this approach, we highlight only the 50-60% precision rate achievable from these models. We touch briefly on the model feature selection, which shows that the top 5 features of the XGB-model, as per Figure 5.20, remain consistent between the different target variable definitions, which is encouraging given the consistent performance of the XGB-model. In contrast, the top RF-model features shown in Figure 5.10 are more erratic over time, which makes the model hard to interpret.

Results in the Context of Previous Research

Even though this paper diverges from existing research in the derivation of the target variables, we can provide some context for the results and discuss how these could be interpreted. Focusing on the potentially most comparable research from [16] and [17], discussed in Section 2.4, the target variables in these papers are defined by looking only at whether startups secure another funding round, are acquired or have an IPO within a set period of time (1 year and 3 years). If we consider the derivation of the lenient and strict target variables, then we can see that they are subsets of the outcomes that these previous papers aim to predict.

This is because both the lenient and strict target variables only allow for a funding round as the *initial* investment event, which is a subset of events allowed for in [16] and [17]. Furthermore, the secondary *exit event* requirement is an **AND** condition, which makes the lenient and strict target variables a further subset of outcomes (ignoring the differences in time horizons). If we denote the target variables in these 2 papers by $y_{[16]}$ and $y_{[17]}$ and the lenient and strict target variables by y_L and y_S respectively, then we can say that:

$$y_S \subseteq y_L \subseteq y_{[16]}, y_{[17]} \tag{5.4.1}$$

This means that the models developed in this paper are trying to identify a much more specific set of investment targets. [16] reports precision levels (for $y_{[16]}$ within 1 year) of 0.626 and 0.535 for the top-100 and top-200 companies identified by the models (ranked by predicted probabilities), while [17] reports precision levels (for $y_{[17]}$ within 3 years) of 0.84-0.86. For the lenient target variable definition we can achieve (median) precision levels up to 0.23 within 2 years (LR model for (C/F)2017/18, Table A.10), 0.41 within 3 years (XGB model for (C/F)2016/17, Table A.12) and 0.52 within 4 years (XGB model for (C/F)2015/16, Table A.12). The even smaller set of targets available under the strict definition subsequently achieve (median) precision levels up to 0.09 within 2 years (XGB model for (C/F)2017/18, Table A.12), 0.23 within 3 years (RF/ XGB model for (C/F)2016/17, Table A.11/ Table A.12) and 0.49 within 4 years (XGB model for (C/F)2015/16, Table A.12).

Summary

The approach taken in this paper is believed to be much more reflective of the commercial requirements of Venture Capital investors so that the decrease in precision of the models is justifiable, particularly in the context of (5.4.1). However, as already discussed, the precision levels achieved are still significantly better than the benchmark success rate of current Venture Capital investors (over a shorter time frame), so that the approach outlined in this paper could be implemented in practice to construct startup investment portfolios that should generate much higher returns.

Furthermore, the results presented in this paper are based on targeting a **high precision**, while providing a **high enough recall**, so that a sufficient number of successful investments are flagged and correctly identified. However, these models can be easily adapted to Venture Capital investors' specific use-cases and risk-aversion preferences. For example:

1. Investors who choose to rely on the models to provide investment target recommendations with a high risk-aversion would likely tune them to maximise precision and so minimise the risk of an investment not reaching an *exit event* (strict or lenient).
2. Investors who aim to invest in a larger number of startup companies (e.g. larger funds) could set a minimum target identification (per year) hurdle and optimise the precision levels for this criteria.
3. Investors who believe that they have an edge in selecting investments could instead utilise the raw model outputs ($\hat{p}$ = probability ($y = 1$) given x). One approach would be to sort the predicted probabilities $\hat{p}$ to rank the companies from the highest to the lowest investment potential and then overlay the investor's own investment selection and evaluation criteria in order to make investment decisions.

Appendix A

Additional Tables and Figures

A.1 Introduction

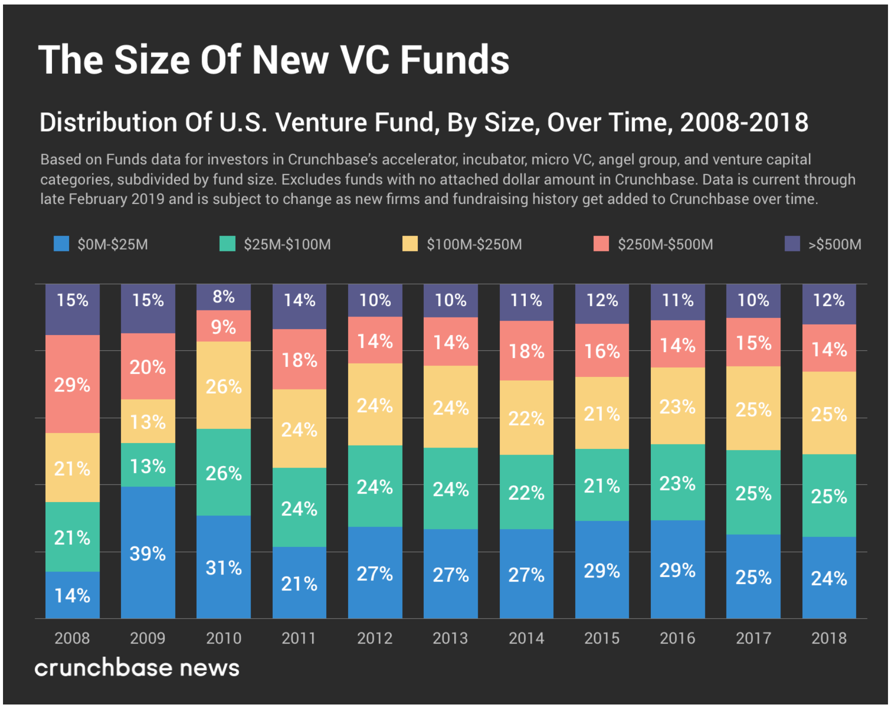
		Strategy						
		Buyout	**Growth**	**Venture Capital**	**Real Estate**	**Private Debt**	**Infrastructure & Natural Resources**	**Other**
AUM ($ bn)	**Est 2019 H1**	**$ 2,067**	**$ 691**	**$ 988**	**$ 992**	**$ 813**	**$ 813**	**$ 107**
Investment Ownership	**Full Ownership**	x			x		x	x
	Majority Stake	x	x		x		x	x
	Minority Stake		x	x				x
Financing	**Debt**	x	x		x		x	x
	Equity	x	x	x	x	x	x	x
Active Management of Investments	**Yes**	x			x		x	x
	Varying Degrees		x	x				x
	No			x		x		x
Exit Strategies	**IPO**	x	x	x				x
	Stock Relisting	x						x
	Sale to other investor(s)	x	x	x	x	x	x	x
	Amortisation					x		x

Table A.1: Overview of private capital investment strategies

This table summarises the information on the various private market investment strategies, where the categories and AUM ($bn) figures are taken from [2], while the remaining information is summarised from various sources including: [40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50] and [51].

From this overview, it is clear that investment strategies are by no means mutually exclusive as private market investments are not always clearly delineated and because fund managers adapt their investment strategies if they believe this can generate additional returns.

Figure A.1 is taken from [52], while Figures A.2 and A.3 are taken from [22, p. 8-9].



The Size Of New VC Funds

Distribution Of U.S. Venture Fund, By Size, Over Time, 2008-2018

Based on Funds data for investors in Crunchbase's accelerator, incubator, micro VC, angel group, and venture capital categories, subdivided by fund size. Excludes funds with no attached dollar amount in Crunchbase. Data is current through late February 2019 and is subject to change as new firms and fundraising history get added to Crunchbase over time.

crunchbase news

Figure A.1: Distribution of U.S. Venture Fund, by Size, 2008-2018

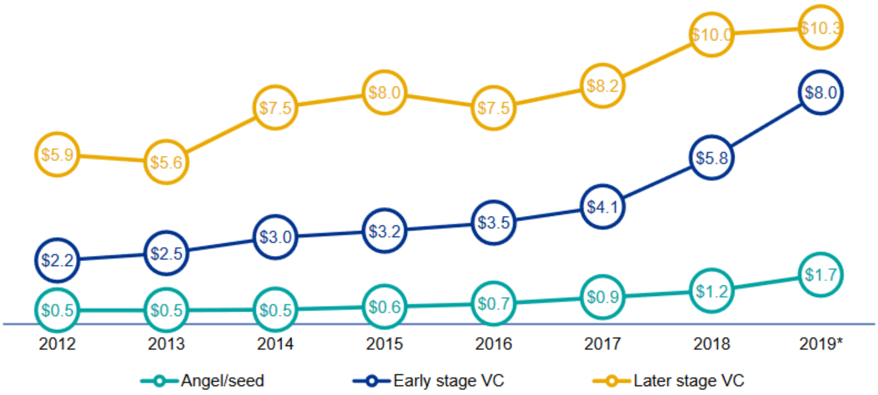


Figure A.2: Global Median Deal Size ($M) by Stage, 2012–2019 (Data as at 12/31/19)

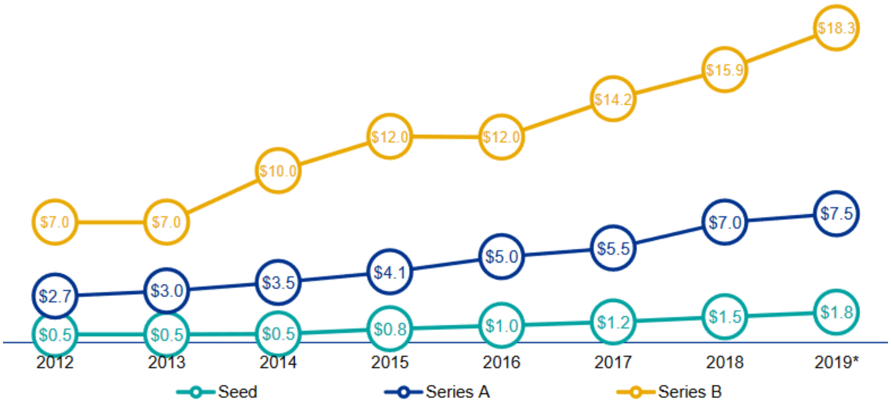


Figure A.3: Global Median Deal Size ($M) for Seed-Series B, 2012–2019 (Data as at 12/31/19)

A.2 Literature Review

Using the time series data created from [18], the 8 (out of the 18) winners in Table A.2 selected for the portfolios in [15, p. 36-39] were identified as not providing an investor with the opportunity to invest prior to the return generating event (acquisition).

Misidentified Winner	All Funding Events	Funding Date
Flutter	Seed Round Acquired	2012/06/07 2013/10/02
Funzio	Series A Acquired	2011/05/05 2012/05/01
Jybe	Seed Round Acquired	2011/02/22 2013/03/20
Leaky	Seed Round Acquired	2011/08/02 2013/08/31
Longboard Media	Series A Acquired	2011/09/15 2012/11/06
Metaresolver	Seed Round Acquired	2012/03/19 2013/02/21
Struq	Series A Acquired	2012/04/19 2014/10/01
ViewFinder	Seed Round Acquired	2012/08/08 2013/12/03

Table A.2: Misidentified Winners

Figure A.4 is a visual representation of the train and test procedure outlined in [16, p. 6].

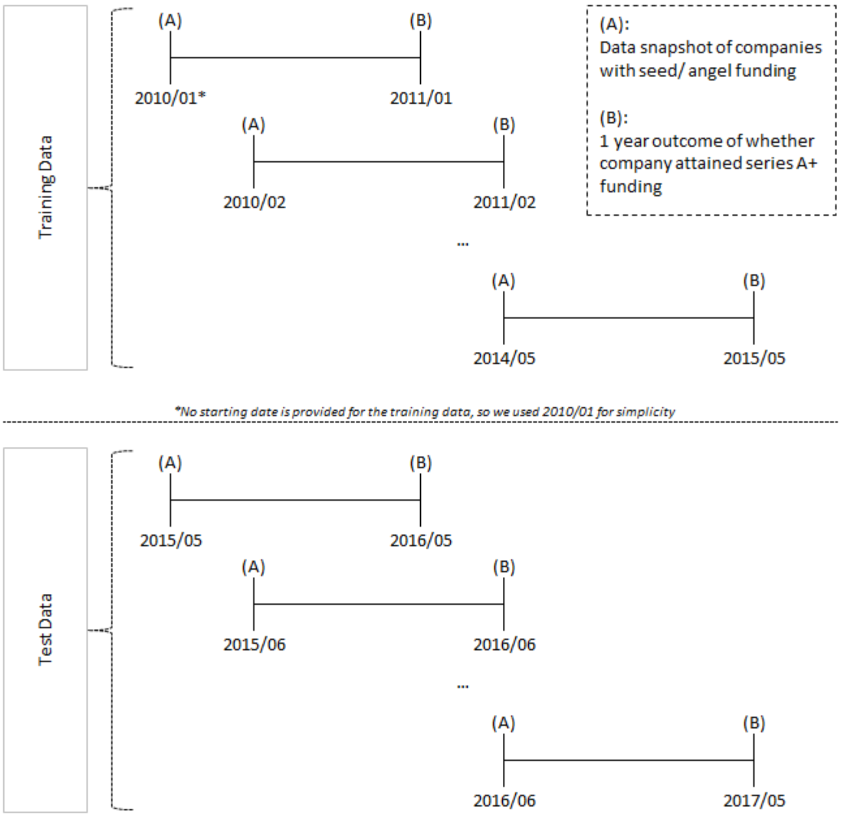
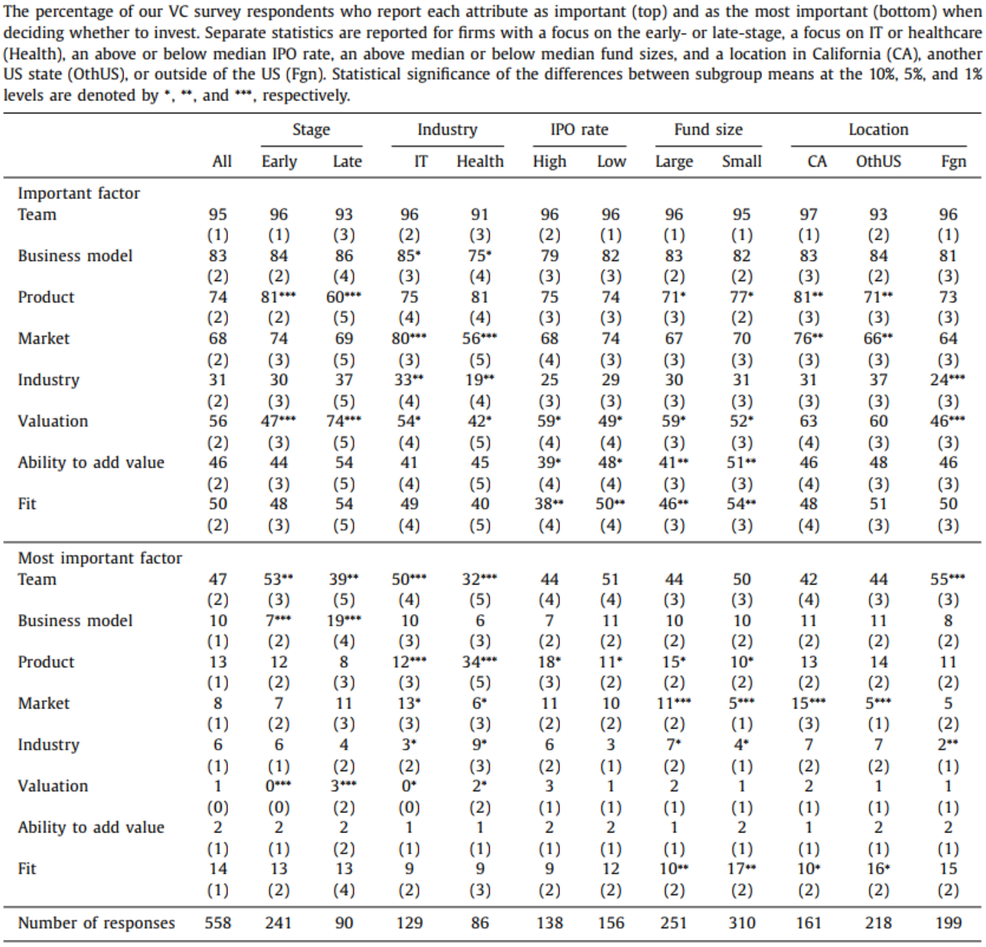

Training Data
(A)
(B)
2010/01*
2011/01
(A)
(B)
2010/02
2011/02
...
(A)
(B)
2014/05
2015/05
(A):
Data snapshot of companies with seed/ angel funding
(B):
1 year outcome of whether company attained series A+ funding
*No starting date is provided for the training data, so we used 2010/01 for simplicity
Test Data
(A)
(B)
2015/05
2016/05
(A)
(B)
2015/06
2016/06
...
(A)
(B)
2016/06
2017/05

Figure A.4: Training and Test Iterations for Models

A.3 Quantitative Models for Venture Capital

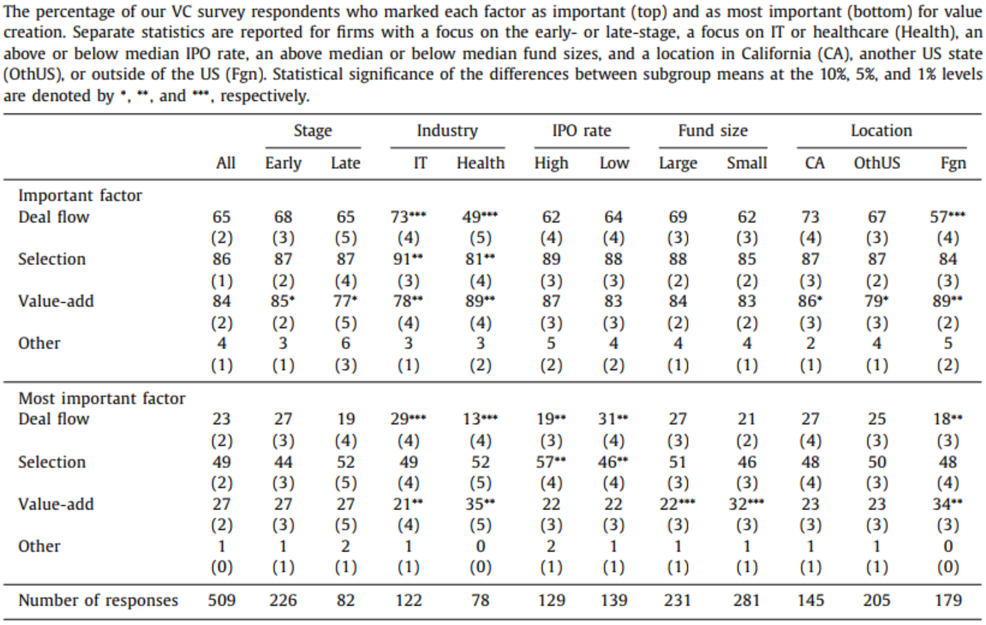
Survey of Venture Capital firms, results taken from [19].



The percentage of our VC survey respondents who report each attribute as important (top) and as the most important (bottom) when deciding whether to invest. Separate statistics are reported for firms with a focus on the early- or late-stage, a focus on IT or healthcare (Health), an above or below median IPO rate, an above median or below median fund sizes, and a location in California (CA), another US state (OthUS), or outside of the US (Fgn). Statistical significance of the differences between subgroup means at the 10%, 5%, and 1% levels are denoted by *, **, and ***, respectively.

		Stage		Industry		IPO rate		Fund size		Location		
	All	Early	Late	IT	Health	High	Low	Large	Small	CA	OthUS	Fgn
Important factor												
Team	95	96	93	96	91	96	96	96	95	97	93	96
	(1)	(1)	(3)	(2)	(3)	(2)	(1)	(1)	(1)	(1)	(2)	(1)
Business model	83	84	86	85*	75*	79	82	83	82	83	84	81
	(2)	(2)	(4)	(3)	(4)	(3)	(3)	(2)	(2)	(3)	(2)	(3)
Product	74	81***	60***	75	81	75	74	71*	77*	81**	71**	73
	(2)	(2)	(5)	(4)	(4)	(3)	(3)	(3)	(2)	(3)	(3)	(3)
Market	68	74	69	80***	56***	68	74	67	70	76**	66**	64
	(2)	(3)	(5)	(3)	(5)	(4)	(3)	(3)	(3)	(3)	(3)	(3)
Industry	31	30	37	33**	19**	25	29	30	31	31	37	24***
	(2)	(3)	(5)	(4)	(4)	(3)	(3)	(3)	(3)	(3)	(3)	(3)
Valuation	56	47***	74***	54*	42*	59*	49*	59*	52*	63	60	46***
	(2)	(3)	(5)	(4)	(5)	(4)	(4)	(3)	(3)	(4)	(3)	(3)
Ability to add value	46	44	54	41	45	39*	48*	41**	51**	46	48	46
	(2)	(3)	(5)	(4)	(5)	(4)	(4)	(3)	(3)	(4)	(3)	(3)
Fit	50	48	54	49	40	38**	50**	46**	54**	48	51	50
	(2)	(3)	(5)	(4)	(5)	(4)	(4)	(3)	(3)	(4)	(3)	(3)
Most important factor												
Team	47	53**	39**	50***	32***	44	51	44	50	42	44	55***
	(2)	(3)	(5)	(4)	(5)	(4)	(4)	(3)	(3)	(4)	(3)	(3)
Business model	10	7***	19***	10	6	7	11	10	10	11	11	8
	(1)	(2)	(4)	(3)	(3)	(2)	(2)	(2)	(2)	(2)	(2)	(2)
Product	13	12	8	12***	34***	18*	11*	15*	10*	13	14	11
	(1)	(2)	(3)	(3)	(5)	(3)	(2)	(2)	(2)	(2)	(2)	(2)
Market	8	7	11	13*	6*	11	10	11***	5***	15***	5***	5
	(1)	(2)	(3)	(3)	(3)	(2)	(2)	(2)	(1)	(3)	(1)	(2)
Industry	6	6	4	3*	9*	6	3	7*	4*	7	7	2**
	(1)	(1)	(2)	(2)	(3)	(2)	(1)	(2)	(1)	(2)	(2)	(1)
Valuation	1	0***	3***	0*	2*	3	1	2	1	2	1	1
	(0)	(0)	(2)	(0)	(2)	(1)	(1)	(1)	(1)	(1)	(1)	(1)
Ability to add value	2	2	2	1	1	2	2	1	2	1	2	2
	(1)	(1)	(2)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)
Fit	14	13	13	9	9	9	12	10**	17**	10*	16*	15
	(1)	(2)	(4)	(2)	(3)	(2)	(2)	(2)	(2)	(2)	(2)	(2)
Number of responses	558	241	90	129	86	138	156	251	310	161	218	199

Figure A.5: Table 5 Reproduced: Important Factors for Investment Selection



The percentage of our VC survey respondents who marked each factor as important (top) and as most important (bottom) for value creation. Separate statistics are reported for firms with a focus on the early- or late-stage, a focus on IT or healthcare (Health), an above or below median IPO rate, an above median or below median fund sizes, and a location in California (CA), another US state (OthUS), or outside of the US (Fgn). Statistical significance of the differences between subgroup means at the 10%, 5%, and 1% levels are denoted by *, **, and ***, respectively.

		Stage		Industry		IPO rate		Fund size		Location		
	All	Early	Late	IT	Health	High	Low	Large	Small	CA	OthUS	Fgn
Important factor												
Deal flow	65	68	65	73***	49***	62	64	69	62	73	67	57***
	(2)	(3)	(5)	(4)	(5)	(4)	(4)	(3)	(3)	(4)	(3)	(4)
Selection	86	87	87	91**	81**	89	88	88	85	87	87	84
	(1)	(2)	(4)	(3)	(4)	(3)	(3)	(2)	(2)	(3)	(2)	(3)
Value-add	84	85*	77*	78**	89**	87	83	84	83	86*	79*	89**
	(2)	(2)	(5)	(4)	(4)	(3)	(3)	(2)	(2)	(3)	(3)	(2)
Other	4	3	6	3	3	5	4	4	4	2	4	5
	(1)	(1)	(3)	(1)	(2)	(2)	(2)	(1)	(1)	(1)	(1)	(2)
Most important factor												
Deal flow	23	27	19	29***	13***	19**	31**	27	21	27	25	18**
	(2)	(3)	(4)	(4)	(4)	(3)	(4)	(3)	(2)	(4)	(3)	(3)
Selection	49	44	52	49	52	57**	46**	51	46	48	50	48
	(2)	(3)	(5)	(4)	(5)	(4)	(4)	(3)	(3)	(4)	(3)	(4)
Value-add	27	27	27	21**	35**	22	22	22***	32***	23	23	34**
	(2)	(3)	(5)	(4)	(5)	(3)	(3)	(3)	(3)	(3)	(3)	(3)
Other	1	1	2	1	0	2	1	1	1	1	1	0
	(0)	(1)	(1)	(1)	(0)	(1)	(1)	(1)	(1)	(1)	(1)	(0)
Number of responses	509	226	82	122	78	129	139	231	281	145	205	179

Figure A.6: Table 13 Reproduced: Important Contributors to Value Creation

A.4 Data and Analysis

US Business Applications and Formations

Table A.3 is based on data from [32].

Year	Business Application (All)	Business Formations (Within 4 Quarters)	Business Formations (Within 4 Quarters)
2005	2,502,014	470,052	19%
2006	2,644,204	425,014	16%
2007	2,659,813	386,987	15%
2008	2,556,042	322,563	13%
2009	2,401,128	288,503	12%
2010	2,463,839	290,965	12%
2011	2,537,102	289,092	11%
2012	2,542,219	280,092	11%
2013	2,582,539	282,436	11%
2014	2,689,139	289,360	11%
2015	2,786,711	291,663	10%
2016	2,945,758	297,778	10%
2017	3,176,109	304,906	10%
2018	3,472,126	324,928	9%

Table A.3: Overview of US Business Applications and Formations, Application Years: 2005-2018

US Employer and Nonemployer Company Data

Table A.4 is based on data from [33].

Year	Employer Firms	Nonemployer Firms	Nonemployer Firms % Total
2012	5,726,160	22,735,915	80%
2013	5,775,055	23,005,620	80%
2014	5,825,458	23,836,937	80%
2015	5,900,731	24,331,403	80%
2016	5,954,684	24,813,048	81%
2017	5,996,900	25,701,671	81%

Table A.4: US Employer and Nonemployer Firms, Years: 2012-2017

Estimated US Startup Coverage by Crunchbase

Table A.3 is based on data from [34] and [18].

Year	Business Formations (Within 4 Quarters)	Haircut (Employer) Business Formations	US Crunchbase Companies	US Crunchbase Companies %
	(A)	(B) = 0.2 x (A)	(C)	(C) / (B)
2012	280,892	56,178	14,138	25%
2013	282,436	56,487	14,385	25%
2014	289,360	57,872	14,625	25%
2015	291,663	58,333	13,322	23%
2016	297,778	59,556	10,998	18%
2017	304,906	60,981	10,166	17%
2018	324,928	64,986	7,716	12%

Table A.5: Estimated US Startup Coverage by Crunchbase, Assuming Only 10% of New Startups are Employer Companies, Founding Years: 2012-2018

UK Business Applications and Formations

Table A.6 is based on data from [35] and [18].

Year	Business Incorporations (All)	UK Crunchbase Companies	UK Crunchbase Companies (% All)
	(A)	(B)	(B) / (A)
2013	208,753	3,061	6.1%
2014	256,225	3,173	5.2%
2015	292,091	3,217	4.6%
2016	327,818	2,778	3.5%
2017	376,180	2,444	2.7%
2018	495,011	1,796	1.5%

Table A.6: Overview of Total UK Business Incorporations vs UK Companies on Crunchbase, Incorporation Years: 2013-2018

The analogous comparison of total US business applications vs. US companies on Crunchbase is provided in Table A.7.

Year	Business Applications (All)	US Crunchbase Companies	UK Crunchbase Companies (% All)
	(A)	(B)	(B) / (A)
2013	2,582,539	14,385	2.8%
2014	2,689,139	14,625	2.7%
2015	2,786,711	13,322	2.4%
2016	2,945,758	10,998	1.9%
2017	3,176,109	10,166	1.6%
2018	3,472,126	7,716	1.1%

Table A.7: Overview of Total US Business Applications vs US Companies on Crunchbase, Applications Years: 2013-2018

Although total business applications (US) and business incorporations (UK) are not completely comparable, these tables still show that there are between 1.4 (2018) to 2.2 (2013) times more companies covered on Crunchbase, as a proportion of all business applications/ incorporations, in the UK compared to the US. This would imply that the proportion of employer UK companies covered by Crunchbase should at least match those figures listed in Table A.5.

UK Employer and Nonemployer Company Data

For completeness, we also include the UK non employer data in Table A.8, sourced from [53].

Year	Employer Firms	Nonemployer Firms	Nonemployer Firms % Total
2013	1,210,910	3,684,745	75%
2014	1,277,360	3,965,775	76%
2015	1,311,865	4,077,585	76%
2016	1,325,485	4,172,185	76%
2017	1,366,835	4,327,680	76%
2018	1,389,285	4,278,225	75%
2019	1,409,950	4,457,820	76%

Table A.8: UK Employer and Nonemployer Firms, Years: 2013-2019

Additional Data Reliability Checks

Data set	Description of data used
Acquisitions	Could be verified by the publicity surrounding the event or from public disclosures by the acquiring company.
Degrees	Cannot be independently validated without accessing the awarding institutions' records. Validating via LinkedIn could also be considered, although the degree information is self-reported on this platform and so wouldn't serve as independent validation.
Funding rounds	Can generally only be validated publicity surrounding the event.
IPOs	Could be verified from the IPO's listing venue.
Jobs	Cannot be independently validated for the same reasons as given for Degrees.
Organizations	Key information used in the data selection and quantitative model builds (such as country, founding date and social media links) can be validated through Companies House (UK), Secretary of State business entity searches (US) and a review of social media sites.
People	Cannot be independently validated for the same reasons as given for Degrees.

Table A.9: Suggested Additional Data Validation Sources and Procedures

Sample Check: US Acquisitions

On a sample of 20 acquisitions, all 20 were verified with only 1 instance where the acquirer name differed due to the acquirer company changing its name.

Sample Check: US IPOs

On a sample of 20 IPOs, 16 were verified via the listing exchange and/ or the SEC, while the remaining 4 had an equity sale event with shares trading via over the counter platforms, but no IPO event registered with the SEC.

Sample Check: Organizations

A sample of 50 US companies' data (42 active and 8 closed) were reviewed on 2020/08/23. The company status was checked against the relevant secretary of state (SOS) database, however not all US SOS databases are freely accessible (or usable) so that only 41 statuses could be checked (33 active, 8 closed). Of the 33 active, only 22 could be located. Of these 22, 16 were indeed active while 6 were closed. 6 of the 8 closed companies could be located and of these all 6 were indeed closed.

Social media profiles were also verified for the 27/ 28/ 32 companies that had data indicating a Facebook/ LinkedIn/ Twitter address however, 4/ 3/ 4 of these landing pages did not exist.

Summary

We can conclude that the data from Crunchbase is relatively reliable, particularly regarding acquisitions. While some questions can be raised on the accuracy of the IPO indicator, whether shares are sold via an equity sale event or IPO, this still provides an exit opportunity for investors and so does not diminish the validity of the outcome.

We cannot draw too many conclusions from the social media checks, as these pages can be taken down over time or the address could change which might not be updated on Crunchbase. However, it is encouraging to see that in ca. 85-90% of the sample, the social media profiles listed on Crunchbase did (still) exist.

With regards to the company status, the SOS databases don't always list the date on which a company became inactive or closed, so it is hard to assess whether companies listed as active on Crunchbase but closed on the SOS database, have been closed for a while or if they closed recently as a consequence of COVID-19's impact on the US economy. Regardless of this, if companies listed as being active are indeed closed, then these companies would not receive any further funding or be acquired and much less have an IPO. As such, any potential shortcomings on the accuracy of the company status do not impact the target variable or the results of the model predictions.

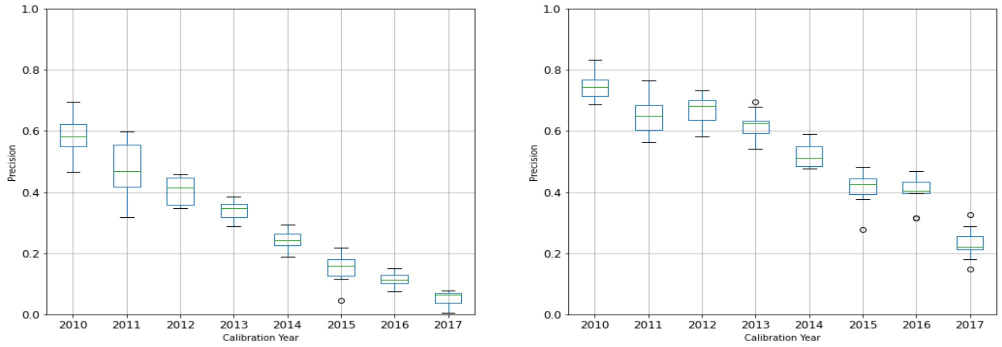
A.5 Model Implementation and Results

In the output that follows, *Training Companies* represents the 70% sample of startups drawn from the Group 1 data which fall within the relevant *calibration window* to train the model. The startups selected in the training data will vary depending on the random draw, however the size of this data will not vary due to being fixed at 70% of Group 1 companies. *(C/F) Test Companies* represents the 30% of startups not drawn from Group 1 to train the model, which have still not been acquired, closed or have had an IPO ('ACIs') in the year after calibration **plus** all new startups founded in the year after calibration (which are not ACIs). The size of this data will vary depending on the random draw, as more/ fewer companies from the 30% of Group 1 companies not used for training will be ACIs in the year after calibration.

Logistic Regression

	(C/F) 2015/16		**(C/F) 2016/17**		**(C/F) 2017/18**	
	S	L	S	L	S	L
Training Companies	29,156		33,557		37,879	
(C/F) Test Companies	$[16,547; 16,640]$		$[18,046; 18,180]$		$[18,001; 18,137]$	
Model Precision: Median	0.16	0.43	0.11	0.40	0.07	0.22
Mean	0.15	0.42	0.11	0.40	0.05	0.23
Range	$[0.05; 0.22]$	$[0.28; 0.48]$	$[0.08; 0.15]$	$[0.32; 0.47]$	$[0.01; 0.08]$	$[0.15; 0.33]$
Model Recall: Median	0.03	0.02	0.03	0.03	0.03	0.03
Mean	0.03	0.02	0.03	0.03	0.03	0.03
Range	$[0.01; 0.04]$	$[0.01; 0.03]$	$[0.01; 0.05]$	$[0.01; 0.04]$	$[0.00; 0.06]$	$[0.02; 0.04]$
Targets: Median	201	233	184	232	155	184
Mean	202	231	170	214	149	190
Range	$[161; 249]$	$[193; 290]$	$[105; 231]$	$[138; 286]$	$[104; 197]$	$[142; 265]$

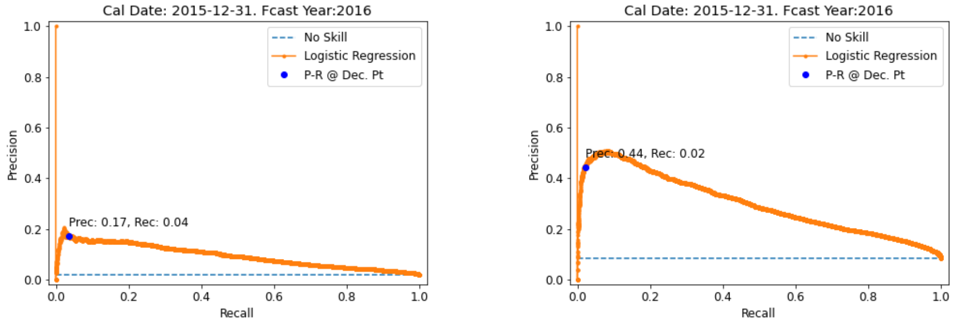
Table A.10: LR Statistics for Strict (S) and Lenient (L) Target Definitions



(a) Strict Target Variable

(b) Lenient Target Variable

Figure A.7: Precision Rate Box-Plots per Calibration-Forecast Pair for 10 Train-Test Iterations



(a) (C/F)2015/16 - Strict

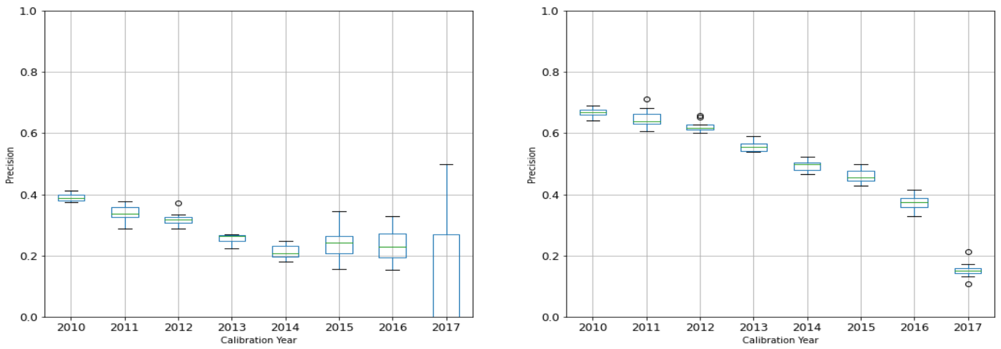
(b) (C/F)2015/16 - Lenient

Figure A.8: Additional Precision-Recall Curve Examples

Random Forest

	(C/F) 2015/16		**(C/F) 2016/17**		**(C/F) 2017/18**	
	S	L	S	L	S	L
Training Companies	29,156		33,557		37,879	
(C/F) Test Companies	$[16,547; 16,640]$		$[18,046; 18,180]$		$[18,001; 18,137]$	
Model Precision: Median	0.24	0.45	0.23	0.37	0.00	0.15
Mean	0.24	0.46	0.23	0.37	0.14	0.15
Range	$[0.16; 0.34]$	$[0.43; 0.50]$	$[0.15; 0.33]$	$[0.33; 0.42]$	$[0.00; 0.50]$	$[0.11; 0.21]$
Model Recall: Median	0.05	0.12	0.03	0.09	0.00	0.02
Mean	0.05	0.12	0.03	0.09	0.00	0.02
Range	$[0.04; 0.05]$	$[0.09; 0.16]$	$[0.02; 0.04]$	$[0.07; 0.11]$	$[0.00; 0.00]$	$[0.02; 0.05]$
Targets: Median	220	1,223	74	791	3	196
Mean	225	1,235	70	829	5	225
Range	[152; 298]	[934; 1,616]	[40; 102]	[559; 1,081]	[0; 17]	[145; 464]

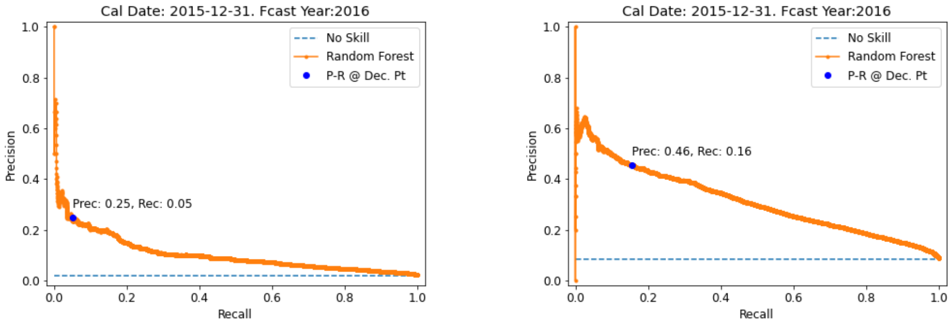
Table A.11: RF Statistics for Strict (S) and Lenient (L) Target Definitions



(a) Strict Target Variable

(b) Lenient Target Variable

Figure A.9: Precision Rate Box-Plots per Calibration-Forecast Pair for 10 Train-Test Iterations



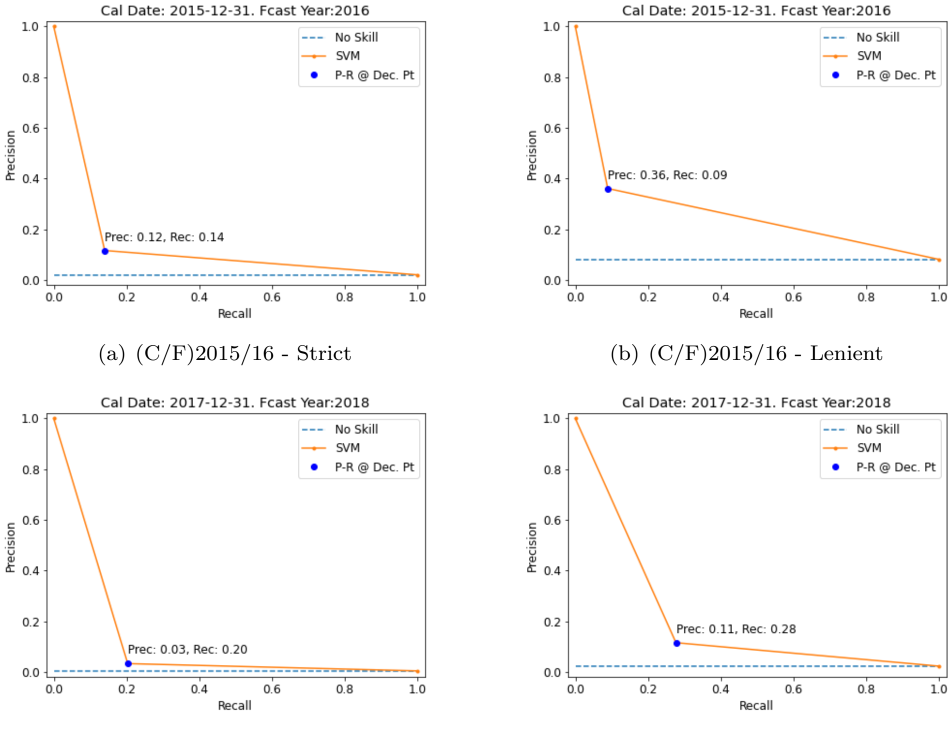
(a) (C/F)2015/16 - Strict

(b) (C/F)2015/16 - Lenient

Figure A.10: Additional Precision-Recall Curve Examples

Support Vector Machines (1000 stratified sample)

Due to the excessive run-times for this model, only a single simulation was performed.



(a) (C/F)2015/16 - Strict

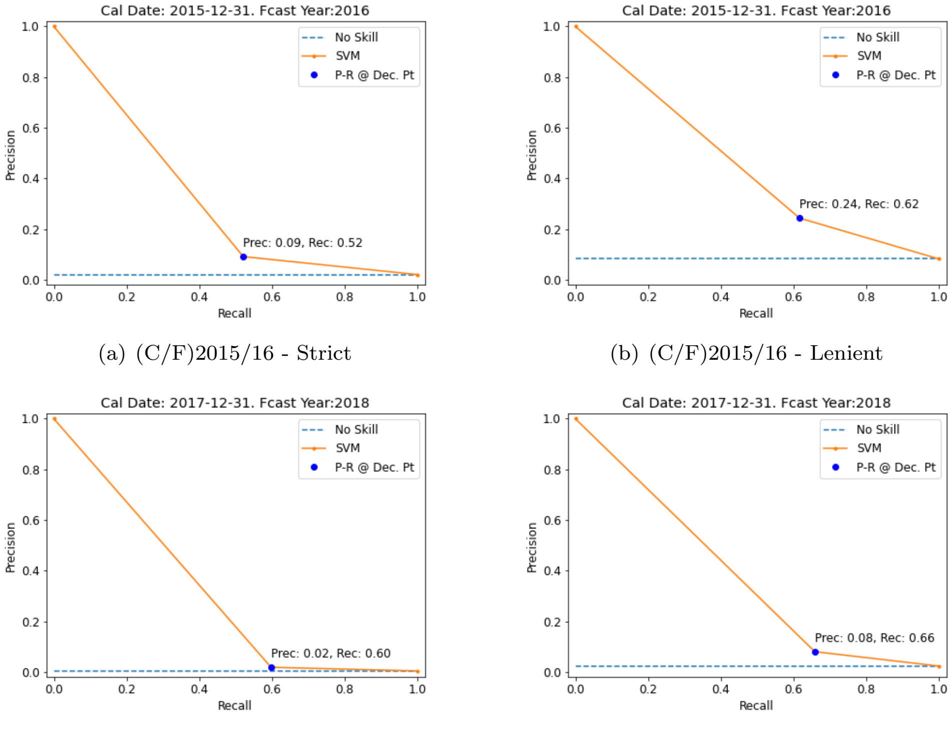
(b) (C/F)2015/16 - Lenient

(c) (C/F)2017/18 - Strict

(d) (C/F)2017/18 - Lenient

Figure A.11: Additional Precision-Recall Curve Examples

Support Vector Machines (5000 stratified sample)



(a) (C/F)2015/16 - Strict

(b) (C/F)2015/16 - Lenient

(c) (C/F)2017/18 - Strict

(d) (C/F)2017/18 - Lenient

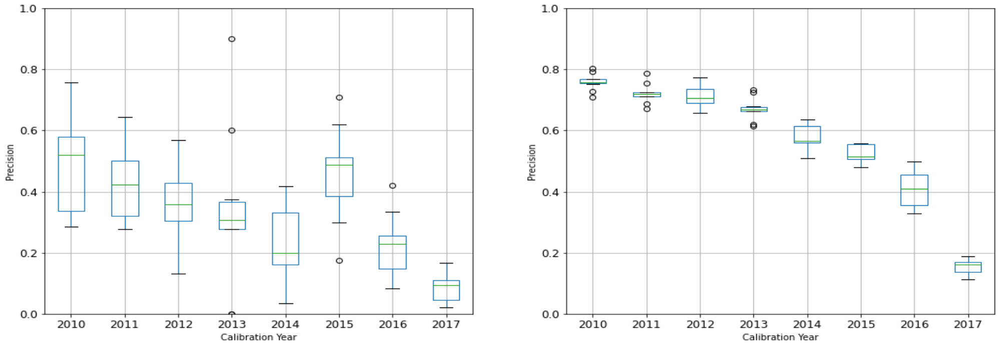
Figure A.12: Additional Precision-Recall Curve Examples

Extreme Gradient Boosting

	(C/F) 2015/16		**(C/F) 2016/17**		**(C/F) 2017/18**	
	S	*L*	*S*	*L*	*S*	*L*
Training Companies	28,708		32,937		37,044	
(C/F) Test Companies	[16, 555; 16, 687]		[18, 024; 18, 180]		[18, 018; 18, 141]	
Model Precision: Median	0.49	0.52	0.23	0.41	0.09	0.16
Mean	0.46	0.52	0.22	0.41	0.09	0.16
Range	[0.19; 0.71]	[0.48; 0.56]	[0.08; 0.42]	[0.33; 0.50]	[0.02; 0.17]	[0.11; 0.19]
Model Recall: Median	0.01	0.04	0.01	0.03	0.01	0.03
Mean	0.01	0.04	0.01	0.03	0.01	0.03
Range	[0.00; 0.03]	[0.03; 0.05]	[0.00; 0.01]	[0.02; 0.04]	[0.00; 0.02]	[0.02; 0.03]
Targets: Median	30	368	16	261	31	267
Mean	31	371	19	259	30	248
Range	[17; 69]	[337; 400]	[6; 31]	[187; 312]	[9; 50]	[184; 287]

Table A.12: XGB Statistics for Strict (*S*) and Lenient (*L*) Target Definitions

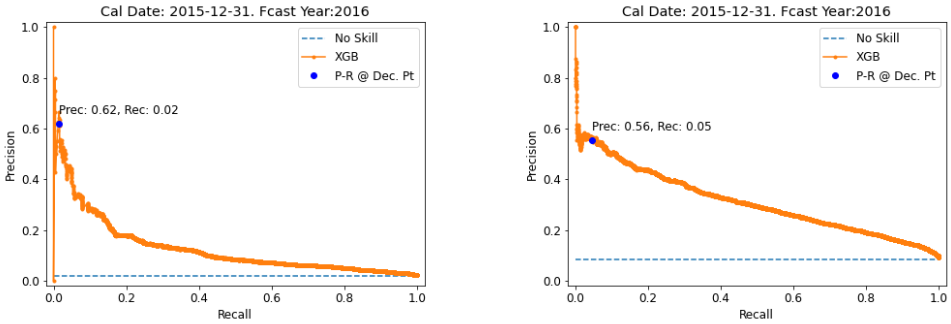
Due to the shorter calibration window (7.5 years), the number of *Training Companies* for XGB is lower than what's reported for LR and RF, while the number of *(C/F) Test Companies* remains quite similar.



(a) Strict Target Variable

(b) Lenient Target Variable

Figure A.13: Precision Rate Box-Plots per Calibration-Forecast Pair for 10 Train-Test Iterations



(a) (C/F)2015/16 - Strict

(b) (C/F)2015/16 - Lenient

Figure A.14: Additional Precision-Recall Curve Examples

A.6 List of Variables Used in Models

Variable	Type	Description
org uuid	Text	Unique organisation identifier (excluded after company train/ test split)
event date coarse	Date	Quarterly time grid (excluded after company train/ test split)
cum debt fin events	Integer	Cumulative number of debt financing events
facebook ind	Binary	Facebook (Y/N)
linkedin ind	Binary	LinkedIn (Y/N)
twitter ind	Binary	Twitter (Y/N)
Empl Int net cumu	Integer	Net number of local employees
Empl Loc net cumu	Integer	Net number of non-local employees
advisor female net cumu	Integer	Net [gender] advisors
advisor male net cumu	Integer	Net [gender] advisors
advisor other net cumu	Integer	Net [gender] advisors
board member female net cumu	Integer	Net [gender] board members
board member male net cumu	Integer	Net [gender] board members
board member other net cumu	Integer	Net [gender] board members
board observer female net cumu	Integer	Net [gender] board observers
board observer male net cumu	Integer	Net [gender] board observers
board observer other net cumu	Integer	Net [gender] board observers
employee female net cumu	Integer	Net [gender] board employees
employee male net cumu	Integer	Net [gender] board employees
employee other net cumu	Integer	Net [gender] board employees
executive female net cumu	Integer	Net [gender] board executives
executive male net cumu	Integer	Net [gender] board executives
executive other net cumu	Integer	Net [gender] board executives
top uni ind net cumu	Integer	Degree from Top University (Y/N)
B net cumu	Integer	Net [degree type] employed at company
BA net cumu	Integer	Net [degree type] employed at company
BBA net cumu	Integer	Net [degree type] employed at company
BCOM net cumu	Integer	Net [degree type] employed at company
BE net cumu	Integer	Net [degree type] employed at company
BENG net cumu	Integer	Net [degree type] employed at company
BSC net cumu	Integer	Net [degree type] employed at company
BTECH net cumu	Integer	Net [degree type] employed at company

Variable	**Type**	**Description**
CERTIFICATE net cumu	Integer	Net [degree type] employed at company
DIPLOMA net cumu	Integer	Net [degree type] employed at company
EMBA net cumu	Integer	Net [degree type] employed at company
JD net cumu	Integer	Net [degree type] employed at company
LLB net cumu	Integer	Net [degree type] employed at company
M net cumu	Integer	Net [degree type] employed at company
MA net cumu	Integer	Net [degree type] employed at company
MBA net cumu	Integer	Net [degree type] employed at company
MD net cumu	Integer	Net [degree type] employed at company
MENG net cumu	Integer	Net [degree type] employed at company
MPHIL net cumu	Integer	Net [degree type] employed at company
MSC net cumu	Integer	Net [degree type] employed at company
MTECH net cumu	Integer	Net [degree type] employed at company
PHD net cumu	Integer	Net [degree type] employed at company
UNKNOWN net cumu	Integer	Net [degree type] employed at company
3ML CPI 2000	Integer (Index, 2000Q1 = 100)	3m lagged CPI Index
3ML CPI 2000 Change	Decimal	3m lagged CPI change
Avg 10y Gov Yield	Decimal	Average 10y Gov Bond Yield Index
Avg 10y Gov Yield Change	Decimal	Average 10y Gov Bond Yield change
3ML M1 2000	Integer (Index, 2000Q1 = 100)	3m lagged M1 Money Value (Local Currency) Index
3ML M1 2000 Change	Decimal	3m lagged M1 Money Value (Local Currency) change
3ML Unemployment	Decimal	3m lagged Unemployment level
3ML Unemployment Change	Decimal	3m lagged Unemployment change
3ML M3 2000	Integer (Index, 2000Q1 = 100)	3m lagged M3 Money Value (Local Currency) Index
3ML M3 Change	Decimal	3m lagged M3 Money Value (Local Currency) change
Asset Purchase bns	Decimal	3m lagged Gov Asset Purchases (Quantitative Easing, Local Currency)

Variable	**Type**	**Description**
Asset Purchase bns change	Decimal	3m lagged Gov Asset Purchases (Quantitative Easing, Local Currency) change
Pvt Cons Exp 2000 bns	Integer (Index, 2000Q1 = 100)	3m lagged Private Final Consumption (Local Currency) Index
Pvt Cons Exp change	Decimal	3m lagged Private Final Consumption (Local Currency) change
Administrative Services	One hot, drop none	Company Industry
Advertising	One hot, drop none	Company Industry
Agriculture and Farming	One hot, drop none	Company Industry
Apps	One hot, drop none	Company Industry
Artificial Intelligence	One hot, drop none	Company Industry
Biotechnology	One hot, drop none	Company Industry
Clothing and Apparel	One hot, drop none	Company Industry
Commerce and Shopping	One hot, drop none	Company Industry
Community and Lifestyle	One hot, drop none	Company Industry
Consumer Electronics	One hot, drop none	Company Industry
Consumer Goods	One hot, drop none	Company Industry
Content and Publishing	One hot, drop none	Company Industry
Data and Analytics	One hot, drop none	Company Industry
Design	One hot, drop none	Company Industry
Education	One hot, drop none	Company Industry
Energy	One hot, drop none	Company Industry
Events	One hot, drop none	Company Industry
Financial Services	One hot, drop none	Company Industry
Food and Beverage	One hot, drop none	Company Industry
Gaming	One hot, drop none	Company Industry
Government and Military	One hot, drop none	Company Industry

Variable	Type	Description
Hardware	One hot, drop none	Company Industry
Health Care	One hot, drop none	Company Industry
Information Technology	One hot, drop none	Company Industry
Internet Services	One hot, drop none	Company Industry
Lending and Investments	One hot, drop none	Company Industry
Manufacturing	One hot, drop none	Company Industry
Media and Entertainment	One hot, drop none	Company Industry
Messaging and Telecommunications	One hot, drop none	Company Industry
Mobile	One hot, drop none	Company Industry
Music and Audio	One hot, drop none	Company Industry
Natural Resources	One hot, drop none	Company Industry
Navigation and Mapping	One hot, drop none	Company Industry
Payments	One hot, drop none	Company Industry
Platforms	One hot, drop none	Company Industry
Privacy and Security	One hot, drop none	Company Industry
Professional Services	One hot, drop none	Company Industry
Real Estate	One hot, drop none	Company Industry
Sales and Marketing	One hot, drop none	Company Industry
Science and Engineering	One hot, drop none	Company Industry
Software	One hot, drop none	Company Industry
Sports	One hot, drop none	Company Industry
Sustainability	One hot, drop none	Company Industry
Transportation	One hot, drop none	Company Industry
Travel and Tourism	One hot, drop none	Company Industry
Video	One hot, drop none	Company Industry
country	Binary	UK(0) / USA (1)

Variable	Type	Description
corporate round	One hot, drop first	Funding Stage
equity crowdfunding	One hot, drop first	Funding Stage
grant	One hot, drop first	Funding Stage
pre seed	One hot, drop first	Funding Stage
private equity	One hot, drop first	Funding Stage
secondary market	One hot, drop first	Funding Stage
seed	One hot, drop first	Funding Stage
series a	One hot, drop first	Funding Stage
series b	One hot, drop first	Funding Stage
series c	One hot, drop first	Funding Stage
series d	One hot, drop first	Funding Stage
series e	One hot, drop first	Funding Stage
series f	One hot, drop first	Funding Stage
series g	One hot, drop first	Funding Stage
series h	One hot, drop first	Funding Stage
series i	One hot, drop first	Funding Stage
series j	One hot, drop first	Funding Stage
series unknown	One hot, drop first	Funding Stage
target	Binary	Successful (1)/ Unsuccessful (0) Investment

Table A.13: List of All Variables Used in Models

Top University Names

The list of top US universities is taken from [15, Appendix C, p. 34], while the top 10 UK universities are taken from The Times Higher Education[1] and US News[2].

UK	**US**
Imperial College London	Berkeley
King's College London	Brown
London School of Economics and Political Science	California Institute of Technology
University College London	Carnegie Mellon
University of Bristol	Columbia
University of Cambridge	Cornell
University of Edinburgh	Dartmouth
University of Manchester	Duke
University of Oxford	Harvard
University of Warwick	Johns Hopkins
	Massachusetts Institute of Technology
	Northwestern
	Princeton
	Stanford University
	University of Chicago
	University of Pennsylvania
	Wharton
	Yale

Table A.14: Overview of Top UK and US Universities Encoded into the Data

[1] https://www.timeshighereducation.com/student/best-universities/best-universities-uk

[2] https://www.usnews.com/education/best-global-universities/united-kingdom

Bibliography

[1] Peter Laurelli. Hedge fund flows remain high amidst market volatility. https://www.evestment.com/news/hedge-fund-flows-remain-high-amidst-market-volatility/, Last accessed 2020/07.

[2] McKinsey & Company. A new decade for private markets, McKinsey Global Private Markets Review 2020. https://www.mckinsey.com/~/media/mckinsey/industries/privateequityandprincipalinvestors/ourinsights/mckinseysprivatemarketsannualreview/mckinsey-global-private-markets-review-2020-v4.ashx, Last accessed 2020/07.

[3] David Coats, Co-Founder of Correlation Ventures. Venture Capital — No, We're Not Normal. https://medium.com/correlation-ventures/venture-capital-no-were-not-normal-32a26edea7c7, Last accessed 2020/07.

[4] Paul Gompers and Silpa Kovvali. The Other Diversity Dividend. *Harvard Business Review*, Aug 2018.

[5] Nicolas Rabener. Venture Capital: Worth Venturing Into? https://blogs.cfainstitute.org/investor/2020/02/17/venture-capital-worth-venturing-into/, Last accessed 2020/07.

[6] Ludovic Phalipp. An Inconvenient Fact: Private Equity Returns & The Billionaire Factory. Jun 2020.

[7] Robert N Lussier and Sanja Pfeifer. A crossnational prediction model for business success. *Journal of small business management*, 39(3):228–239, 2001.

[8] Srinivasan Ragothaman, Bijayananda Naik, and Kumoli Ramakrishnan. Predicting corporate acquisitions: An application of uncertain reasoning using rule induction. *Information Systems Frontiers*, 5(4):401–412, 2003.

[9] C.-P. Wei, Y.-S. Jiang, and C.-S. Yang. Patent analysis for supporting merger and acquisition (M&A) prediction: A data mining approach. volume 22, pages 187–200. Springer Verlag, 2009.

[10] Kaloyan Haralampiev, Boyan Yankov, and Petko Ruskov. Models and tools for technology start-up companies success analysis. *Economic Alternatives*, pages 15–24, 2014.

[11] Diego Camelo Martinez. Startup success prediction in the dutch startup ecosystem. 2019.

[12] Guang Xiang, Zeyu Zheng, Miaomiao Wen, Jason Hong, Carolyn Rose, and Chao Liu. A supervised approach to predict company acquisition with factual and topic features using profiles and news articles on techcrunch. *ICWSM 2012 - Proceedings of the 6th International AAAI Conference on Weblogs and Social Media*, 01 2012.

[13] Dominik Dellermann, Nikolaus Lipusch, Philipp Ebel, Karl Popp, and Jan Marco Leimeister. Finding the unicorn: Predicting early stage startup success through a hybrid intelligence method. 12 2017.

[14] Francisco Ramadas da Silva Ribeiro Bento. Predicting start-up success with machine learning. 2018.

[15] David Scott Hunter, Ajay Saini, and Tauhid Zaman. Picking winners: A data driven approach to evaluating the quality of startup companies, 2017.

[16] Boris Sharchilev, Michael Roizner, Andrey Rumyantsev, Denis Ozornin, Pavel Serdyukov, and Maarten de Rijke. Web-based startup success prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 2283–2291, New York, NY, USA, 2018. Association for Computing Machinery.

[17] Javier Arroyo, Francesco Corea, Guillermo Jimenez-Diaz, and Juan A Recio-Garcia. Assessment of machine learning performance for decision support in venture capital investments. *IEEE Access*, 7:124233–124243, 2019.

[18] Crunchbase, Inc. Company data. Data retrieved on 2020/06/06, `http://www.crunchbase.com/`.

[19] Paul A Gompers, Will Gornall, Steven N Kaplan, and Ilya A Strebulaev. How do venture capitalists make decisions? *Journal of Financial Economics*, 135(1):169–190, 2020.

[20] Crunchbase. The Ultimate Guide to Raise Capital for a Startup. `http://about.crunchbase.com/wp-content/uploads/2019/10/Ultimate-guide-raising-startup-capital.pdf`, Last accessed 2020/07.

[21] Glossary of Funding Types. `https://support.crunchbase.com/hc/en-us/articles/115010458467-Glossary-of-Funding-Types`, Last accessed 2020/07.

[22] KPMG Private Enterprise. Venture Pulse, Q4'19, Global Analysis of Venture Funding. `https://home.kpmg/xx/en/home/campaigns/2020/01/q4-venture-pulse-report-global.html`, Last accessed 2020/07.

[23] Scott W. Menard. Logistic regression from introductory to advanced concepts and applications, 2010.

[24] Jason W. Osborne. Best practices in logistic regression, 2015.

[25] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer Series in Statistics. Springer New York, New York, NY, second edition edition, 2009.

[26] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*, volume 103 of *Springer Texts in Statistics*. Springer New York, New York, NY, 2013.

[27] Joshua Starmer. StatQuest content on "Decision Trees" and "Random Forest". `https://www.youtube.com/c/joshstarmer/videos`, Last accessed 2020/07.

[28] Joshua Starmer. StatQuest content on "Gradient Boost". `https://www.youtube.com/c/joshstarmer/videos`, Last accessed 2020/07.

[29] Joshua Starmer. StatQuest content on "Extreme Gradient Boost". `https://www.youtube.com/c/joshstarmer/videos`, Last accessed 2020/07.

[30] Tianqi Chen and Carlos Guestrin. Xgboost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug 2016.

[31] Vishal Morde. XGBoost Algorithm: Long May She Reign! `https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d`, Last accessed 2020/07.

[32] United States Census Bureau. Business formation statistics (BFS). `https://www.census.gov/programs-surveys/bfs/data/datasets.html`, Last accessed 2020/06/30.

[33] United States Census Bureau. Statistics of u.s. businesses (SUSB). https://www.census.gov/programs-surveys/susb.html, Last accessed 2020/06/30.

[34] United States Census Bureau. NES datasets. https://www.census.gov/programs-surveys/nonemployer-statistics/data/datasets.html, Last accessed 2020/06/30.

[35] Companies House. Free company data product. http://download.companieshouse.gov.uk/en_output.html.

[36] PitchBook. PitchBook 2019 Venture Capital Outlook. https://pitchbook.com/news/reports/4q-2018-pitchbook-analyst-note-2019-venture-capital-outlook, Last accessed 2020/07.

[37] Gary M. Weiss, Kate McCarthy, and Bibi Zabar. Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs? In *DMIN*, 2007.

[38] Alex Graham. Three Core Principles of Venture Capital Portfolio Strategy. https://www.toptal.com/finance/venture-capital-consultants/venture-capital-portfolio-strategy, Last accessed 2020/07.

[39] Sevakula, Rahul Kumar, Verma, Nishchal K. Support vector machine for large databases as classifier". In *Swarm, Evolutionary, and Memetic Computing*, pages 303–313. Springer Berlin Heidelberg, 2012.

[40] Corporate Finance Institute. What are Private Equity Funds? https://corporatefinanceinstitute.com/resources/knowledge/trading-investing/private-equity-funds/, Last accessed 2020/07.

[41] Adam Barone. Buyout. https://www.investopedia.com/terms/b/buyout.asp, Last accessed 2020/07.

[42] Brian DeChesare. Private Equity vs. Venture Capital: Why the Lines Have Blurred. https://www.mergersandinquisitions.com/private-equity-vs-venture-capital/, Last accessed 2020/07.

[43] Investopedia. Private Equity vs. Venture Capital: What's the Difference? https://www.investopedia.com/ask/answers/020415/what-difference-between-private-equity-and-venture-capital.asp, Last accessed 2020/07.

[44] Venero Capital Advisors. Understanding the difference between Growth Equity and Venture Capital. https://pulse.venerocapital.com/understanding-the-difference-between-growth-equity-and-venture-capital-a4a6adc22902, Last accessed 2020/07.

[45] Allen Wagner. Characteristics of Growth Deals. https://pitchbook.com/news/articles/characteristics-of-growth-deals, Last accessed 2020/07.

[46] Dheeraj Vaidya. What is Growth Capital? https://www.wallstreetmojo.com/growth-capital/, Last accessed 2020/07.

[47] Julia Kagan. Private Equity Real Estate. https://www.investopedia.com/terms/p/private-equity-real-estate.asp, Last accessed 2020/07.

[48] Brian DeChesare. Real Estate Private Equity (REPE): The Definitive Guide. https://www.mergersandinquisitions.com/real-estate-private-equity/, Last accessed 2020/07.

[49] PitchBook. What is private debt? https://pitchbook.com/blog/what-is-private-debt, Last accessed 2020/07.

[50] PriceWaterhouseCoopers. Infrastructure. Generating long term value from infrastructure investments. https://www.pwc.co.uk/assets/pdf/infrastructure-funds-brochure-may2010.pdf, Last accessed 2020/07.

[51] Backbay Communications. Infrastructure can also exit. `https://informaconnect.com/infrastructure-can-also-exit/`, Last accessed 2020/07.

[52] Jason Rowley. There Are More VC Funds Than Ever, But Capital Concentrates At The Top. `https://news.crunchbase.com/news/there-are-more-vc-funds-than-ever-but-capital-concentrates-at-the-top/`, Last accessed 2020/07.

[53] Department for Business, Energy & Industrial Strategy. Business Population Estimates. `https://www.gov.uk/government/collections/business-population-estimates`, Last accessed 2020/06/30.