A long/short index rebalance portfolio

by

E.P. Bryce

Supervisor: Andreas Park
April 2019

A long/short index rebalance portfolio

by

E.P. Bryce

Supervisor: Andreas Park
April 2019

Abstract

A long/short index rebalance portfolio

E.P. Bryce
Bachelor of Applied Science in Engineering Science (Physics)
Division of Engineering Science
University of Toronto
2019

I identify a factor-based methodology for testing whether abnormal returns exist during index rebalances. Using that approach, I find that normal returns do exist around rebalance events, although those abnormal returns may not truly be statistically significant. In light of those abnormal returns, I identify an approach to trading rebalances by predicting which securities out of the entire TSX universe are most likely to be added to the S&P/TSX Composite Index in the next month. Trade signals were identified by training a logistic binary classifier on the same factors which the index committee uses to determine eligibility for inclusion. The low likelihood of any random security being added to an index was mitigated through random over-sampling. Finally, I used these signals to construct a hypothetical portfolio which performed well in back-testing.

Acknowledgements

I'd like to first thank Andreas Park for his supervision of this thesis. As well, I'd like to thank Rotman building management for granting me access to their space and ITG Canada – particularly Ivan Cajic and Doug Clark – for providing very helpful data and direction early in this project. Finally, a very special thanks to all of my friends and colleagues who have listened to me opine about index rebalances for the past year.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Passive investing has grown in popularity in the decade since the financial crisis: in 2016, passive investors accounted for roughly 30% of global equity holdings ("Slow-motion revolution" 2016). Passive funds are considered inexpensive as there are few research or management costs associated with passive portfolio management, but there are nevertheless believed to be hidden costs associated with tracking an index. Inclusion of a stock in an index produces abnormal single-stock returns in the short term(Arnott, Kalesnik, and Wu 2018) with curiously negative longer-term returns(Chan, Kot, and Tang 2013)(Chen, Noronha, and Singal 2004).

As index changes are announced in advance of their effective dates, and because those changes are made on known dates with specific criteria, abnormal returns are thought to emerge months before – and revert days after – the actual rebalance.(Arnott, Kalesnik, and Wu 2018) The magnitude and timescale of these abnormal returns is however not well understood in the academic literature. In particular, these trends have been only minimally studied in a Canadian context.

Quantifying and predicting the presence and magnitude of reversion may enable index fund managers to minimise their transaction costs and improve returns.

1.1 Research question

As a security is added to an index, how much of that security's returns are attributable specifically to its inclusion in the index? If there are abnormal returns, is it possible to construct a portfolio to capitalise on this inefficiency?

1.1.1 Objectives

As can be seen from the research question, the objective of this thesis is three-part:

1. identify the cumulative abnormal returns associated with index rebalances;
2. apply machine learning (random forests, gradient boosted trees, support vector machines, etc.) to identify securities which are likely to be added to the Composite – ideally, before these changes are priced-into the securities; and
3. use the foregoing to construct a portfolio.

1.2 Outline

This report will first provide a review of the available literature, including official specifications governing index rebalances. Next, the Methods chapter discusses how the data was aggregated and outlines the tests used for the event study, the machine learning tools used to identify index changes, and the techniques used to construct and back-test the portfolio. Then, the Event Study chapter covers the event study and identifies whether there are abnormal returns during index rebalances that would be worth trying to trade. Finally, the Model Implementation chapter uses machine learning tools to identify rebalance candidates and construct a potential portfolio. That same chapter will also provide the backtest results.

1.2.1 Approaches scoped-out

The appendices of this report contain both the model used to develop the strategy, as well as some data and investigations on other effects of index rebalances.

Some of the questions which were considered in this thesis but not answered due to time constraints were:

1. How does volatility affect the extent or timing of trade pre-positioning? Anecdotally, portfolio managers are believed to compress index trades during periods of high volatility and spread-out trades during periods of relatively low volatility. Can we confirm this anecdotal belief? Can the contribution of σ_M to returns be isolated?

2. Beyond volatility, can the abnormal returns be decomposed into factors?

3. Did volatility during rebalances change after the TSX introduced limit on close orders to the MOC facility? The introduction of LOC orders reduced volatility in the close generally speaking(Clark, Cajic, et al. 2016) – but how did this affect index rebalances specifically?

4. Is there an increase in liquidity risk commensurate with abnormal returns?

5. Did reversion decrease when S&P changed the benchmark price to 5-day VWAP?[1]

6. If returns on a security are decoupled from industry returns, is this a predictor of index changes? Specifically, can we regress excess returns on each security against returns on an appropriate basket and identify divergence?

[1] Some names appear to trade up during the benchmark window, then revert afterwards – e.g., TVE on 28 Aug.

Chapter 2

Literature review

As of yet, there has been little academic research into investor value destruction through index fund rebalances. Most academic research has focused on the effects of indexing and passive investing on competition. There is considerable academic research on event studies and some industry research exists in modelling transaction costs. At the nexus of these topics lies the application of event study analysis in index rebalances, the focus of this paper.

2.1 Event studies

The event study methodology which seems to prevail in the literature is consistent with that used by Espen Eckbo and Karin 2000. Abnormal returns are computed by comparing the return on a security to the expected return from the market model:

$$r_{jt} - r_{ft} = \alpha_j + \beta_j(r_{mt} - r_{ft}) + \xi_{jt} \tag{2.1}$$

where r_{jt} is the continuously compounded return on security j in month t, r_{ft} is the risk-free rate, α_j is the regression constant, β_j is the systemic risk, and ξ_{jt} is an error term assumed to be zero on average.

Thus, Espen Eckbo and Karin 2000 constructs a regression to compare the returns before and after the event:

$$\hat{\gamma}_{j\tau} = \begin{cases} r_{j\tau} - \left(r_{f\tau} + \hat{\alpha}_j^b + \hat{\beta}_j^a(r_{m\tau} - r_{f\tau})\right) & \text{for } -12 \leq \tau \leq 0 \\ r_{j\tau} - \left(r_{f\tau} + \hat{\alpha}_j^a + \hat{\beta}_j^a(r_{m\tau} - r_{f\tau})\right) & \text{for } 1 \leq \tau \leq 12 \end{cases} \tag{2.2}$$

where $\hat{\theta}_j^a$ and $\hat{\theta}_j^b$ are the OLS-estimates *after* and *before* the event.

In this model, the abnormal return for month τ relative to the event is computed as the arithmetic mean, and tthe resulting Z-statistic is:

$$Z_T = \frac{1}{\sqrt{N_\tau}} \sum_{j=1}^{N_\tau} \frac{\gamma_{j\tau}}{\hat{\sigma}_{\gamma j}} \tag{2.3}$$

where $\gamma_{j\tau}$ is the abnormal return attributable to the event, assuming a normal distribution $N(0, 1)$.

To test statistical significance, another common method is to compare the abnormal returns to the variance in returns. However, Boehmer, Masumeci, and Poulsen 1991 found this method too often rejects

the null hypothesis where the event itself induces variance. A more rigorous approach would instead to be to conduct a cross-sectional study. However, if the induced volatility is limited, then a simple significant test would likely still be appropriate. My own statistical background is also likely too limited to effectively perform a rigorous cross-sectional study.

Earlier, Fama et al. 1969 proposed a method involving comparing the the discrete returns on a security with the market:

$$\ln R_{jt} = \alpha_i + \beta_j \ln L_t + u_{jt} \tag{2.4}$$

where $R_{jt} = \frac{P_{jt} - Djt}{P'j,t-1}$ is the price of the security *ex divided* versus the prior period, P_{jt} is the price of the security, P'_{jt} is the price of the security adjusted for capital changes, D_{jt} is the dividends on the security in that period, and L_t is the market index return.

Alternatively, Healy, Palepu, and Ruback 1992 used a method which compared the returns on security after a merger announcement to the expected present-valued change in cash flows, by estimating the equation:

$$CFRET_i = \alpha + \beta\ ARET_i + \xi_i \tag{2.5}$$

where $CFRET$ is the pre-tax cash flow return improvement per year, $ARET$ is the capitalised value of future cash flow return improvements. That is to say, the model estimates the change in discount factor before and after the merger. This approach is not directly relevant, as there are no probable cash flow changes resulting from inclusion in an index; however, it could be possible to construct a model which uses a similar comparison to estimate how much the market values index inclusion.

In spirit, these approaches are all similar to each other: they involve comparing the returns on a security to the returns on some reference portfolio.

In industry, returns and risk are often modelled with a factor model based on returns and correlations to a security's market, sector, and industry.(Cavaglia, Brightman, and Aked 2000) This has been studied in the academic literature by Gu and Zeng 2014, but a literature search finds little else.

2.2 Canadian equity indices

In general, there are two large Canadian index providers: S&P Dow Jones Indices (S&P Global) and MSCI. Largely due to availabity of data, this project focused specificially on S&P Dow Jones Indices.

2.2.1 Rebalance Specifications

S&P/TSX Composite

Inclusion in the S&P/TSX Composite index is dependant largely on the price and market capitalisation of the security, but does include other factors. To be included in the S&P/TSX Composite Index, the security must meet *all* of the following requirements:

1. The volume weighted average price of the security (VWAP) must be at least $1 over both the past three months *and* the last three days of prior month, where the VWAP is defined as:

$$\bar{P} = \frac{\sum P\,V}{\sum V} \tag{2.6}$$

2. The market capitalisation of the stock in the last three trading days of the prior month, based on the VWAP of the security and the total float ($[Market\ Cap] = [VWAP][Float]$) must represent at least 0.05% of the index, after including the market capitalisation of the candidate security in the index. That is to say:

$$\%\ contribution = \frac{[Market\ Cap]_{stock}}{[Market\ Cap]_{stock} + [Market\ Cap]_{index}} \geq 0.05\% \tag{2.7}$$

.

3. The stock must be liquid. Specifically, the float turnover must be at least 50% annually: the total annual volume traded must be at least 1/2 the total floating shares available.

4. The stock must not be ineligible for inclusion; in practice, this largely means that inclusion in the composite is restricted to only common stock.

5. Finally, the stock must be Canadian. In the context of S&P indices, this means it must: be incorporated, formed or established in Canada; file financial statements and disclosures with a Canadian regulator; be listed on the Toronto Stock Exchange; and have a substantial presence in Canada based on the location of its head office or principal executive offices or a substantial portion of its fixed assets and revenues.

The relevant timelines are summarized in the graphic below.

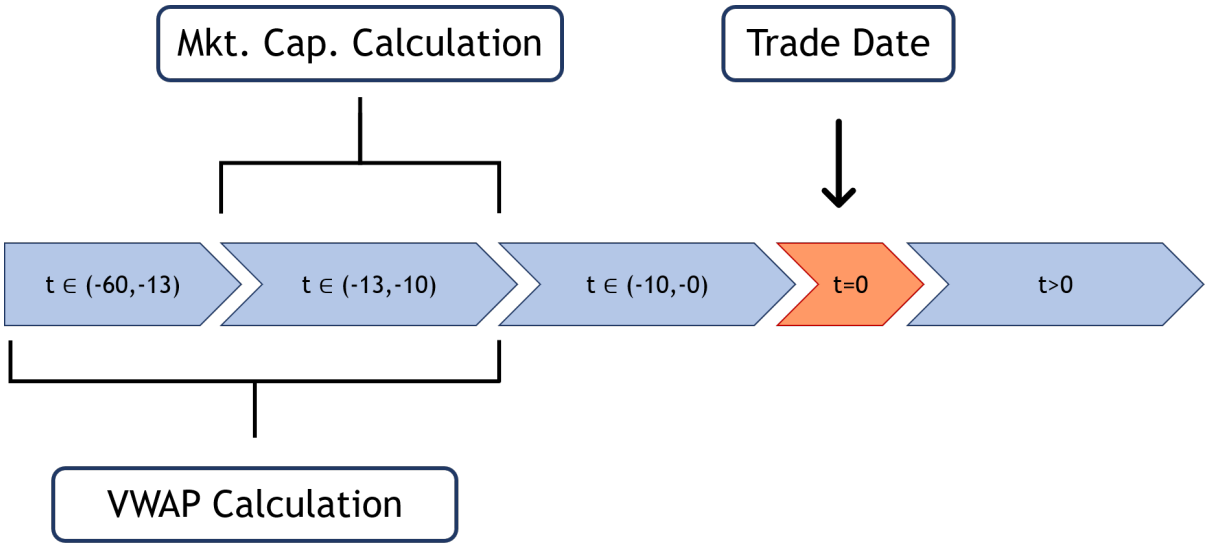


Figure 2.1: Graphic representation of key dates leading up to Composite rebalances.

S&P/TSX Sector Indices

There are eleven S&P TSX Sector indices for each of the GICS (Global Industry Classification Standard) sectors, as is the case for all S&P Dow Jones index products. Specifically, these sectors are: consumer discretionary, consumer staples, energy, financials, health care, industrials, information technology, materials, real estate, telecommunication services and utilities. These sector indices contain all securities included in the Composite index are included in one of the eleven sector indices.So, an addition to the

S&P/TSX Composite index necessarily coincides with an addition to one of the sector indices. Accordingly, it is likely not straightforward to decompose the abnormal return contribution from the overall change versus the contribution from inclusion in the sector index.

S&P/TSX 60

The requirements for inclusion in the 60 are not as straightforward as the composite, as changes are made on an ad hoc basis and on no fixed schedule. Additions and deletions are generally made on a basis of market capitalisation, although the committee also seeks to achieve a sector balance. There are also liquidity requirements associated with the 60. Modelling changes to the 60 is a much more interesting machine learning probelm for that reason, but is out of scope for this project.

2.3 Effects of passivity on stock-specific returns

From first principles, the addition of a stock to an index will lead to a net demand in shares and a commensurate out-performance. Conversely, the removal of a stock from an index will lead to a net over-supply for those shares and a commensurate under-performance.

The inclusion of a stock in an index produces abnormal returns in the short term.(Arnott, Kalesnik, and Wu 2018)

Specifically, in the United States addition of stocks to the S&P 500 or 400 led to returns of approximately 5% from announcement to the effective date and addition of stocks to the S&P 600 led to returns of approximately 7% from the announcement to the effective date. Conversely removal of a stock from the S&P 600 was followed by returns of approximately -15% from announcement to the effective date.(Clark and Cajic 2018) However, in the longer-term, there are curiously negative returns for inclusions(Chan, Kot, and Tang 2013)(Chen, Noronha, and Singal 2004).

Most published research has not to date decomposed these returns into the impact of the change versus the returns which would have existed *a priori*. That is to say, many industry models assume that all returns between announcement and the effective date are attributable entirely to the index change, rather than external factors.

Clark and Cajic 2018 observed a curious trend wherein securities underperform when promoted from the S&P 500 to the more exclusive S&P 400. This is likely attributable to a net decrease in ownership of the S&P 400 versus the more widely-tracked 500. So, the driving factor between abnormal returns during index rebalances is the change in funds tracking the security.

From time to time, the scale of speculative prepositioning is greater than natural indexer flow, and thus the security trades "wrong way" (i.e., reverts) between the announcement and the effective date. In practice, prepositioning is inferred in the trading days leading up to the effective change by measuring daily volatility during that window.

However the shares traded in the marketplace increasingly do not reflect the total number of shares which would have to be exchanged in order to satisfy the total indexing demand. This is likely because much index exposure is hedged in the derivatives or swaps market, rather than the cash equities market.(Clark, Cajic, et al. 2016)

Overall, at the fund-level, these transaction costs can be inferred by observing the tracking error for index products. Theoretically, assuming nil transaction costs associated with fund inflows or outflows and assuming stock loan revenue per Blocher, Whaley, and Blair n.d., the remaining tracking error

should be attributable specifically to transaction costs associated with rebalances. Using this approach, Frazzini, Israel, and Moskowitz 2018 calculated the annual transaction costs for the S&P 500 and Russel 2000 to be 4.72 bps and 12.87 bps, respectively.

2.4 Effects of passivity on volatility

An interesting but yet unanswered question is whether passive investing adversely affects volatility convexity.

There is an established link between certain volatility products (notably the XIV inverse volatility ETP) and equities futures market. Sushko and Turner 2018 implicated volatility products in exacerbating volatility on 5 February 2018. (Sushko and Turner 2018) found causal relation between ETF arbitrage and volatility contagion. For example, high-frequency ETF arbitrage was linked to the 2010 flash crash. However, the link is merely linear or is related to some aspect of the ETF market rather than indexing itself.

As far as I can tell, there is little research on the effects of passivity on volatility contagion.

Going in the other direction, Borkovec and Tyurin 2015 found these abnormal volatility surprises – and thus volume surprises – increase transaction costs for institutional investors. conclusions have not been extended to investigate the management of large passive portfolios specifically.

2.5 Effects of passivity on firms

Although not directly relevant to the market structure aspects of this thesis, the effects of index tracking and common ownership has been the subject of much academic research and investigation by regulators.(*Transcript of FTC Hearings Session #8: Competition and Consumer Protection in the 21st Century* 2018)

In one long-run study, Backus, Conlon, and Sinkinson 2019 found that common ownership of firms reduces competitive behaviour. Where two competing firms are held by the same shareholders, those firms adopt strategies which maximise the aggregate shareholder value of both firms in spite of strategies which may further enrich their own shareholders at the expense of their competitors. This has been observed by Azar, Schmalz, and Tecu 2014 specifically to lead to anti-competitive behaviour in the airline industry – though that particular study has been critiqued for failing to account for divestiture by fund managers during periods of delisting following bankruptcy.(*Transcript of FTC Hearings Session #8: Competition and Consumer Protection in the 21st Century* 2018)

Although typically thought to be related to the rise of passive index investing (Backus, Conlon, and Sinkinson 2019), this phenomena has also been linked by Posner, Scott Morton, and Weyl 2016 to higher levels of institutional investment by, for example, pension funds.

Passive investing has also been studied for its effects on corporate governance, being linked by Boehmer, Masumeci, and Poulsen 1991 to more independent directors and removal of takeover protections but tend to otherwise vote with management on other shareholder resolutions.(Fichtner, Heemskerk, and Garcia-Bernardo 2017)

These matters are unfortunately more closely related to corporate finance and governance and do not directly relate to the focus of this thesis, which is the effect of relative passivity on market structure.

2.6 Machine learning applications

There has been a trend in recent years to adapt machine learning to trading. Recently, this effort has focused on optimising trade scheduling.(Kissell and Bae 2018)

The "J.P. Morgan Big Data & AI Strategies Report" by Kolanovic and Krishnamachari 2017 is a comprehensive reference text on the application of machine learning in finance, but unfortunately says little about modelling index changes.

Some machine learning models also require optimisation of hyper-parameters. In a binary classification context, random forest and gradient boosted tree models should be optimised using a structured hyper-parameter search technique, such as the approach in Bergstra, Yamins, and Cox n.d.

2.6.1 Applications in trade scheduling

In trading, Park and Van Roy 2012 applied reinforcement learning methods to market impact modelling. Reinforcement learning has been further applied to trade schedule optimisation in academia using simulated data (Ritter 2017), data from the Almgren and Chriss 1999 market impact model(Hendricks and Wilcox 2014), and historical market data(Nevmyvaka, Feng, and M. Kearns 2006) as well as in industry using real-world execution data (Pearson et al. 2018). In short, these models work by comparing potential actions at each state (i.e., the algo may either trade or not trade at each discrete time step) and rewarding the algo for reducing residual risk and penalising the algo for incurring costs. Over time, the algo learns to take actions which minimise penalties and maximise rewards. Reinforcement learning is ideally suited for optimisation of trade schedules. Reinforcement learning models for optimal trade execution typically model the state of the program trade with factors relating to the order (time left to trade, remaining inventory, limit price), the security (bid-ask spread, volume imbalance on the book, costs to hit the far-side quote), the market (market-relative price movements), and historical order characteristics (volume participation rates, slippage)(M. Kearns and Nevmyvaka 2013) However, the time span of my research is norably longer than those considered in a trade scheduling context.

2.6.2 Other trading applications

A classic example of machine learning applied to trading is in predicting the next tick based on the existing limit order book. (Cont and Kukanov 2012)(Guéant, Lehalle, and Tapia 2011)

Another application of machine learning is in optimising order sizes on dark venues. This has been done by using censored exploration algorithms to estimate volumes available on dark pools, based on an individual trader's historical fill rates.(Agarwal, Bartlett, and Dama 2010)(K. J. Kearns et al. 2010)

2.7 Gaps in knowledge

From this literature review, a number of key knowledge gaps arise, in particular:

1. Bearing in mind existing research conducted on the American S&P 500, can these results be replicated in the Canadian marketplace?

2. Given that these tools have already been applied to trading, can machine learning be used to predict index changes?

3. What effect does volatility have on index rebalances? What effect do index rebalances have on volatility?

4. Can fund-level transaction costs associated with rebalances be explained by measuring abnormal returns in the run-up to index rebalances?

Chapter 3

Methods

This section will describe some of the methods used to collect and analyze the data, and then will lay out the exact methods used to complete the event study and index change prediction portions of the project.

3.1 Data collection

A great deal of time was spent optimising the data structures and workflow for efficient and scalable data analysis. Some of my key code is included in the appendix; though the entire project is also available online on GitHub.[1] Code for computation of some factors related to liquidity was borrowed from related research on the TSX dual market making program.

3.1.1 Obtaining index change data

Canadian index change data was difficult to come by at first. The databases on Wharton Research Data Services (WRDS) were evaluated to identify information on Canadian indices. Unfortunately, the Center for Research in Security Prices (CRSP) dataset[2] contains only data on American indices.

Additionally, the RavenPack dataset[3] was evaluated to determine whether index constituent changes could be identified from reporting on the PR Newswire or DowJones feed. Although the changes may be reported in either of these feeds, index changes are not specifically identified: the dataset shows events relating to board changes, analyst rating changes, and dividends, but without content on index inclusions.

Approximately 10 years of index change data covering 5376 changes was obtained from S&P Capital IQ using the following search criteria:

1. Key Developments by Type: Index Constituent Drops OR Index Constituent Adds [All History]
2. Exchange Country (All Listings): Canada

[1] https://github.com/ebryce/rebal-costs/blob/master/thesis.ipynb
[2] https://wrds-web.wharton.upenn.edu/wrds/query_forms/navigation.cfm?navId=118
[3] https://wrds-web.wharton.upenn.edu/wrds/query_forms/navigation.cfm?navId=370

Current index holdings covering 827 names were also obtained from S&P Capital IQ, for all TSX indices with Canadian names. ITG Canada also provided a list of 208 TSX Composite changes dating back to Q1 2012.

Reverse-engineering index changes from ETF reporting

Exchage traded fund providers are required to issue daily holdings reports. These reports are aggregated by Bloomberg [4] and form a reasonable proxy for index holdings. This was also used to estimate the size of trades that would have to be executed by fund managers in order to liquidate a position after a deletion, for example, and was useful for illustrative purposes such as during the thesis presentation.

By comparing the portion of the fund represented by a particular investment to the overall capitalisation of the fund

$$[\text{fraction of portfolio in security } j] \times [\text{market capitalisation of fund } F] = [\text{position of } j \text{ in fund } F] \quad (3.1)$$

it is possible to obtain a quick estimate for scale of market impact.

3.1.2 Integrating financial data

Because the identification of candidate securities for inclusion in the TSX Composite can be largely reduced to a question of predicting future market capitalisations, the possibility of financial/accounting data predicting future changes was investigated.

Several key quarterly ratios were also pulled from S&P Capital IQ for the time period ranging from Q1 2010 through Q4 2018, inclusive. This data was integrated into some of the rebalance predictive models, but ultimately did not affect the model's predictive power.

Initially, I pulled financial data from The Globe and Mail's website by exploiting a vulnerability in their website which would permit HTTP GET requests to their data vendor to return key accounting data which would ordinarily only be available by manually copying the data from a dynamic portion of their website.

3.1.3 Market and factor data

Market and risk-free returns – as well as SML and HMB factors – were obtained from French's online database.[5]

Some industry comparables were obtained from Aswath Damodaran's online database.[6] These were used in an earlier version of the model, but added little to the model's predictive power.

Volatility data

The historical volatility for the Composite index was obtained from Bloomberg. This data was used in an attempt to compare degree of abnormal returns to market volatility. The thought was that, under greater volatility, speculators are less likely to pre-position themselves ahead of index trades.

[4] `XIC CN Equity HOLD <GO>`
[5] `http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html`
[6] `http://people.stern.nyu.edu/adamodar/New_Home_Page/datacurrent.html`

In an earlier iteration of this project, the price of a Canadian implied volatility ETF (HUV) was used as a proxy for market volatility. There are two problems with this approach. First, this is an American product hedged to CAD/USD. So, it does not appropriately reflect risk in the Canadian marketplace. Second, this product is an ETP and therefore can be victim to contango or backwardation. Because this is a daily product, the instrument does not track the VIX for time horizons longer than one day(*BetaPro SP500 VIX Short-Term Futures ETF Prospectus* 2018). A more appropriate metric will be investigated.

3.1.4 Price data

Daily close prices

Daily close prices from Q3 2012 through Q3 2018 were obtained from the Canadian Financial Markets Research Centre (CFMRC)[7] annual file via the CHASS Data Centre. This dataset has several limitations; namely, it does not contain a field for VWAP, and its tickers are current only to end-of-year 2018. So, matching price data to index change data from Capital IQ had to first account for ticker changes. Those ticker changes were obtained from Capital IQ.

Intraday trade & quote data

For some market data used in early market structure investigations, trade and quote data was obtained from the TMX Grapevine platform. Some technical limitations of this system unfortunately limited my ability to run useful experiments. Time spent trying to make effective use of these datasets also reduced time available for other research.

3.2 Event study

The event study formed the first half of this project and sought to determine whether abnormal returns exist around rebalance events. The methods used in this half of the project are largely adapted from those in event study literature described above.

3.2.1 Determining abnormal returns

In the initial iteration, a three-factor CAPM approach was used to calculated expected daily returns. These daily expected returns were then used to adjust single-stock returns and obtain their cumulative abnormal returns (ξ):

$$\xi = R - R_{rf} - \beta_m\big(R_m - R_{rf}\big) \tag{3.2}$$

for each day, where R_m and R_f are the French market and risk-free returns, R is the return on each security, and β is the security's systemic risk such that:

$$\beta_i = \frac{Cov(R_i, R_m)}{Var(R_m)} \tag{3.3}$$

[7] `http://clouddc.chass.utoronto.ca.myaccess.library.utoronto.ca/ds/cfmrc`

computed separately for the period before and after the index change – in case returns on a given security become more (less) closely correlated to the market return after the security is added to (removed from) the composite.

However, this approach was found to be quite limited. In particular, it left exposure to returns separate from the variable I wanted to investigate: inclusion in the index.

The next method that was used was a 5-factor Fama-French approach:

$$\xi = R - R_{rf} - \beta\big(R_m - R_{rf}\big) + \beta_s SMB + \beta_v HML \tag{3.4}$$

where SMB refers to the excess return on small cap stocks versus large cap stocks and HML refers to the excess return on value stocks over growth stocks.

This approach was determined to be inappropriate for this study because securities which are added to the Composite are almost by definition large-cap securities, which invalidates the usefulness of the SMB factor. Ultimately, a market-sector-industry factor model was used:

$$\begin{aligned} E[R] &= R_{rf.} + \beta_m(R_m - R_{rf}) + \beta_{sec.}(R_s - R_m) + \beta_{ind.}(R_{ind.} - R_{sec.}) + \xi \\ \Rightarrow \xi &= R - \Big(R_{rf.} + \beta_m(R_m - R_{rf}) + \beta_{sec.}(R_s - R_m) + \beta_{ind.}(R_{ind.} - R_{sec.})\Big) \end{aligned} \tag{3.5}$$

where ξ is the abnormal return, $\beta_{ind.}$ and $\beta_{sec.}$ refer to the correlation between a stock's daily returns and the returns on an equal-weighted portfolio of stocks in the same GICS sector or industry respectively, and $R_{ind.}$ and $R_{sec.}$ are the returns on those equal-weight reference portfolios.

An estimation window was constructed for all time between a given index change and the most recent index change for that same security. If the security had never before been added to/removed from the Composite, then the estimation window was all trading days back to 2010.

3.2.2 Testing for cumulative abnormal returns

From here, we can aggregate the daily returns in the period leading up to the rebalance:

$$CAR[\tau_1, \tau_2] = \prod_{t=\tau_1}^{\tau_2} (1 + \xi_t) \tag{3.6}$$

where τ_1 and τ_2 are the start and end of the event study window, and ξ_t is the abnormal return on day t from the five-factor model earlier.

The significant of the CAR was computed by comparing it to the volatility in the estimation window:

$$SCAR[\tau_1, \tau_2] = \frac{CAR[\tau_1, \tau_2]}{\sigma[\tau_1, \tau_2]} \tag{3.7}$$

A positive alpha implies the price will fall (rise) before the security is added to (removed from) the Composite. The convergence of alpha during the ranking week implies that, on average, the market prices-in all rebalance information before the announcement. Naturally, as the ranking week progresses, the certainty a trader has in whether a security will be added/removed becomes higher.

3.3 Machine learning

The overall method used to develop the machine learning models in this project was adapted from work done for the Engineering Science Capstone Design Course and is described in the appendix. This methodology was created from a survey of industry best practice, academic design literature, and academic machine intelligence literature.

The primary stakeholder for this algorithm would be an individual investor. Although this work could likely be extended to an institutional buy-side manager, there would almost certainly be large transaction costs associated with implementing this strategy.

The objective of the model is to identify securities which will be added to the Composite index. Then, the result of that classification will feed into a portfolio construction script which will back-test the trade signals against historical returns. The results of the classification model should be interpretable in order to aid future research and in order to minimise risk.

Solutions will be scored on a combination of:

1. area under the receiver-operator characteristic curve;
2. balance between the false-negative and false-positive rate on the confusion matrix; and
3. overall return on the back-tested strategy.

Ultimately, that last metric – return on the back-tested strategy – links directly to the overall objective of the model.

3.3.1 Class imbalance

On average, there are relatively few securities which will be added to or removed from an index on any given month; said otherwise, in each month, it is very unlikely that any randomly selected security will be added to or removed from the Composite index.

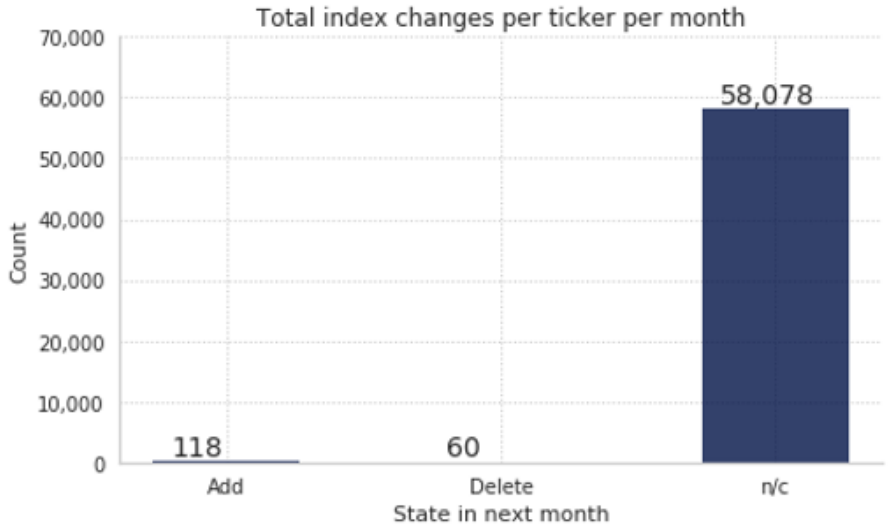
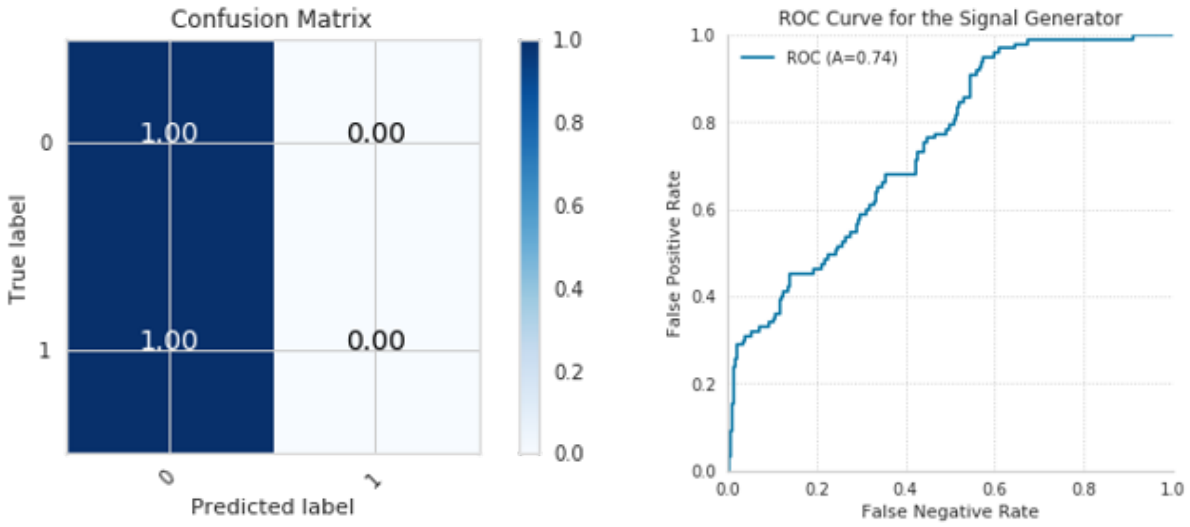


Figure 3.1: Index changes per month, compared with the status quo.

So, any binary classifier will reach very high accuracy – exceeding 95% by many scoring metrics – by simply always specifying that there will never be any change to an index.



(a) Confusion matrix for a binary classifier.

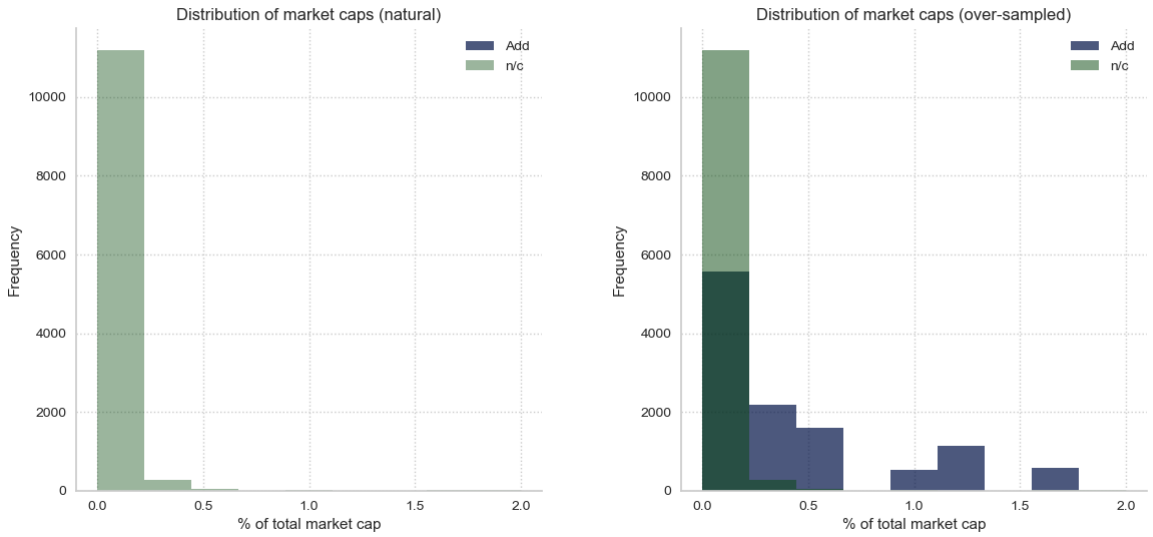
(b) Receiver-operator characteristic for the classifier.

Figure 3.2: Performance measures for a random forest binary classifier trained on the imbalanced dataset.

Over-sampling methods were used to deal with the significant class imbalance.

Random oversampling

Random over-sampling was applied to this problem, as described in the literature review section, using the `imlearn` Python library. The distribution of the sample, in market capitalisation space, is shown in the histogram below.



(a) Natural distribution for the dataset.

(b) Randomly over-sampled distribution.

Figure 3.3: Distribution of market capitalisations of securities which were added to and not added to the index, before and after random oversampling.

One of the issues with this random over-sampling approach is the tendency to preserve the bimodal distribution present in the training set. For example, this means that large capitalisation, high price securities which are not domiciled in Canada – and thus are ineligible for inclusion in the Composite index – are nevertheless included in the final sample.

However, this can be easily corrected by applying a threshold to candidate securities and assuming that any un-included securities with extremely large capitalisations (above $\approx 0.6\%$ of the index market capitalisation) are unlikely to be included in the Composite at any point in time, and therefore exclude them from consideration.

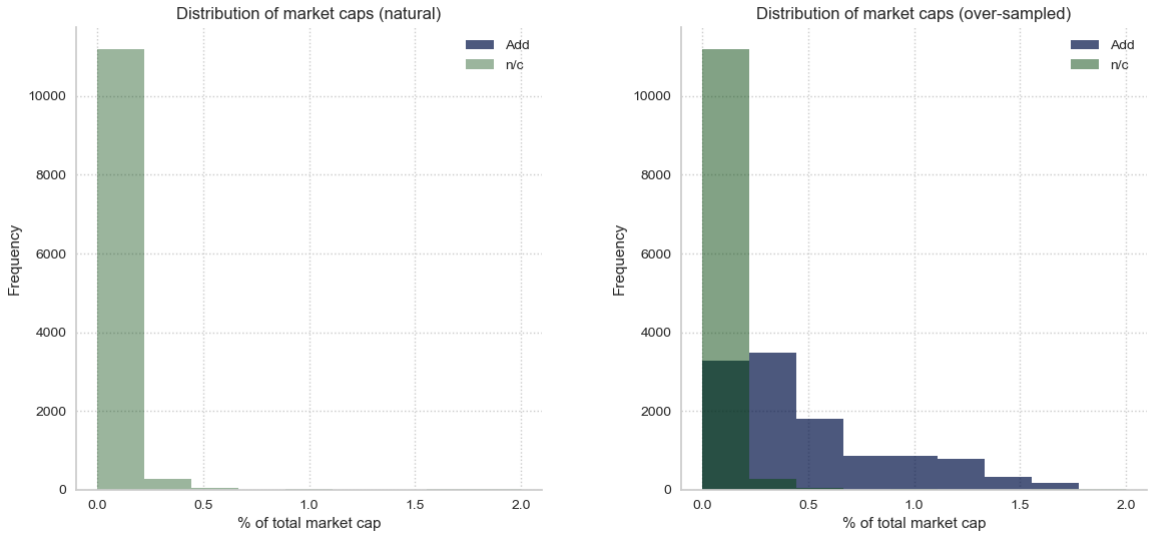
Alternatively, some machine learning techniques such as random forests or gradient boosted trees may perform well with this type of approach, given that they are able to produce non-linear responses to the input matrices.

Synthetic minority over-sampling

Synthetic minority over-sampling (SMOTE) produces a training set which contains additional entries, not identical to those in the original set, but which could conceivably be in a training set. It does this by interpolating points.(Chawla et al. 2002)

Specifically, the `imlean.SMOTE()` function was used to construct the over-sampled dataset.

This method, which produced the distribution in the figures below, actually led to poorer performance in the test set.



(a) Natural distribution for the dataset.

(b) Randomly over-sampled distribution.

Figure 3.4: Distribution of market capitalisations of securities which were added to and not added to the index, before and after applying the synthetic minority over-sampling technique.

This method produced inferior classification results because it created new outlier securities in between the securities that were close to inclusion and the very large capitalisation securities which were nevertheless ineligible for inclusion due to domicile requirements.

That is to say, instead of simply having two clusters in the training set – one in the low-cap regime with actual candidates and one in the high-cap regime with ineligible securities – SMOTE created additional securities which could not reasonably be included in the index.

3.3.2 Test set construction

In order to minimise over-fitting, the sample set was divided into a training set and a test set. The training set constituted 80% of the total population (i.e., 80% of all security-month entries), with the test set constituting the remaining 20%. The binary classifier and algorithm were training on the training dataset: in other words, the regression coefficients were constructed using the training set. The model was then evaluated with the test set, which had not been used to train the regression. The algo was finally backtested using the test dataset.

3.3.3 Feature engineering

Feature engineering should be conducted to ease in model training. The features used by S&P to rank candidates for Composite inclusion(*S&P/TSX Canadian Indices Methodology* 2018) were built into the training data frame. Previous papers investigating this topic, such as those by Arnott, Kalesnik, and Wu 2018, Chan, Kot, and Tang 2013, and Chen, Noronha, and Singal 2004 were be reviewed to identify the methods used by those authors.

Ultimately, based on the features used by S&P Indices to construct their lists, the following features were selected:

Input Parameters	Output Parameters
Float turnover (%)	Added to Composite (binary)
Share of index market capitalisation (%)	Removed from Composite (binary)
VWAP ($)	No-change in status (binary)
Volume traded ($)	
Market capitalisation ($)	

Table 3.1: Model parameters for machine learning fitting.

These were computed on a monthly basis for the days which were used for rankings, based on the S&P Methodology.

3.3.4 Classification problem

As discussed, a primary objective is creating a hypothetical strategy with perfect Composite-related foresight: machine learning may provided some valuable insights.

At the first order, prediction of index changes is a binary classification problem. Either the security is added to an index or it is not; either the security is removed from the index or it is not.

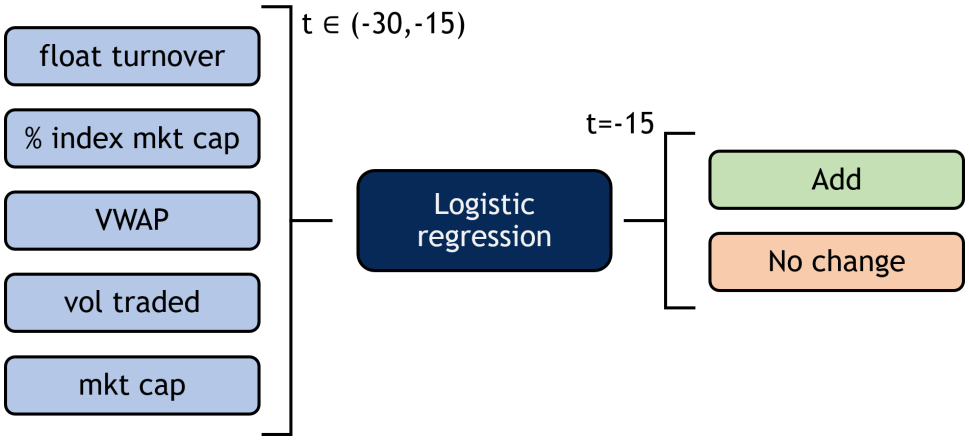


Figure 3.5: Graphic representation of the model inputs/outputs.

This class of problem is well-suited to various types of binary classifiers such as random forests, gradient boosted trees, or the classic logistic regression. This type of problem can also be solved with artificial neural networks, but that would be well out-of-scope for this project.

"Auto ML" frameworks were also considered for selecing an appropriate model, as the models selected for the interim draft were selected largely out of convenience. This can be done with additional machine learning tools such as the Zoph and Le 2016 AutoML framework. However, this approach was not ultimately adopted, as the binary classification problem was much better constrained than initially thought, especially after the class imbalance was resolved.

3.4 Portfolio construction

A custom "signal generation" script uses the trained binary classifier to construct a list of securities which will probably be included in the Composite index. This list is then mapped to a set of trades on specific dates. The algorithm then assumes that the security is shorted on the first day of the rebalance month. The position is then covered at some later point in time after the rebalance occurs.

If the chosen classification model is probabilistic – i.e., if each of the binary results is assigned a probability, such as a logistic regression – then the relative probability of each signal becomes the weight of that trade in the portfolio. If the model is not probabilistic – i.e., if it returns only a binary outcome, such as a support vector machine – then each trade is weighted equally in the portfolio.

The script is modular such that different signal generation scripts can be tested against the same portfolio construction script and vice-versa.

Chapter 4

Results & discussion

This chapter describes the event study conducted for TSX Composite additions. Some additional research was done into other factors which may be affected by the event; however, these are relegated to the appendix as they do not directly contribute to the development of the binary classifier or therefore the resulting trading strategy.

4.1 Constructing the factor model

Before any abnormal returns could be calculation, a factor model was constructed as described in the methods section.

The equal-weighted daily returns on the market and each GICS sector and industry was computed. An example is shown before, for each GICS sector in September 2017.

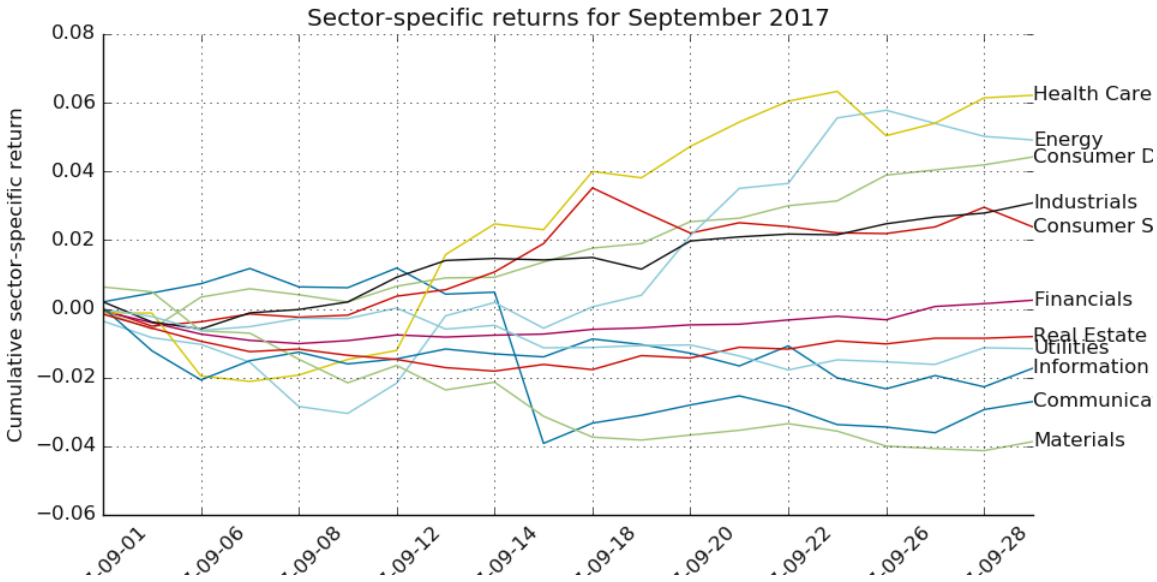


Figure 4.1: Total sector-specific cumulative returns for September 2017.

These daily returns were also computed for the individual GICS industries.

Then, given the daily returns for each sector and industry, as well as the daily market return, coefficients were calculated for each security using a linear regression. Some example coefficients are provided below for TSX-listed companies belonging to the "Copper" GICS industry code:

Ticker	Market Return	Sector Return	Industry Return	Residual
ERO	0.42	0.06	0.03	-0.00
FM	-0.24	0.37	0.17	-0.01
LUN	-0.42	0.26	0.10	-0.01
NSU	-0.15	0.26	0.09	-0.01
TKO	0.06	0.42	0.11	-0.01

Table 4.1: Coefficients for names in the GICS "Copper" industry.

This was done as a more straightforward and manageable approach than creating clusters or peers for each security.

From here, those factors are used to compute the expected return per security, which we will compare to the actual realized returns.

4.2 Abnormal returns

Given the foregoing factor model, I computed the expected return on each treated security, based on the performance of the market and its sector/industry peers during the test window. These expected returns were compared to the actual daily return on each security to obtain the abnormal return. These abnormal returns are plotted below.

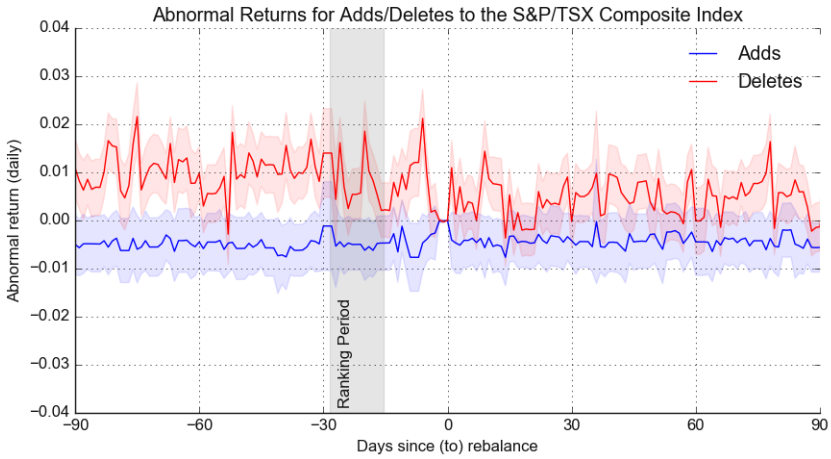


Figure 4.2: Abnormal returns on a portfolio of stocks being added to or removed from the S&P TSX Composite Index.

Interestingly, through the entire study period, the abnormal return on adds seems to be somewhat negative, with a positive uptick around the rebalance day. The standard deviation in daily returns also seems to be consistent across the study period, which suggests that the rebalance does not itself induce volatility.

So, since their is not induced volatility, our simple test statistic of cumulative abnormal return over return variance should be valid.

4.3 Cumulative abnormal returns

Once the daily abnormal returns are available, it is trivial to compute the aggregated cumulative abnormal return of the portfolio.

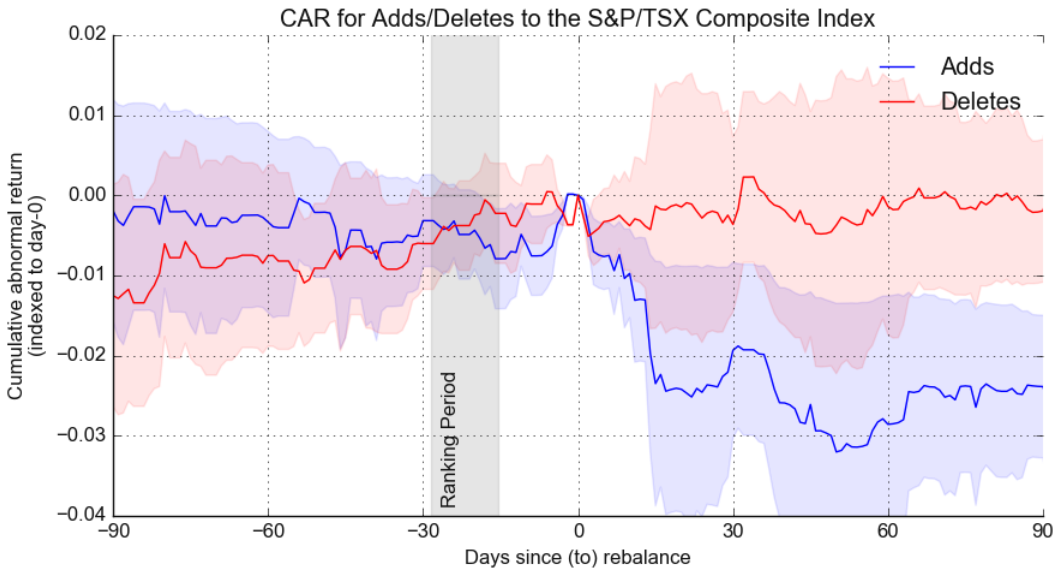


Figure 4.3: Cumulative abnormal returns on a portfolio of stocks being added to or removed from the S&P TSX Composite Index.

There clearly exists significant returns leading into the addition of securities to the Composite, followed by reverse back to the pre-trade prices, and ultimately well below the initial price level.

4.4 Testing for statistical significance

And, with this, we can proceed to compute the quick significance ratio.

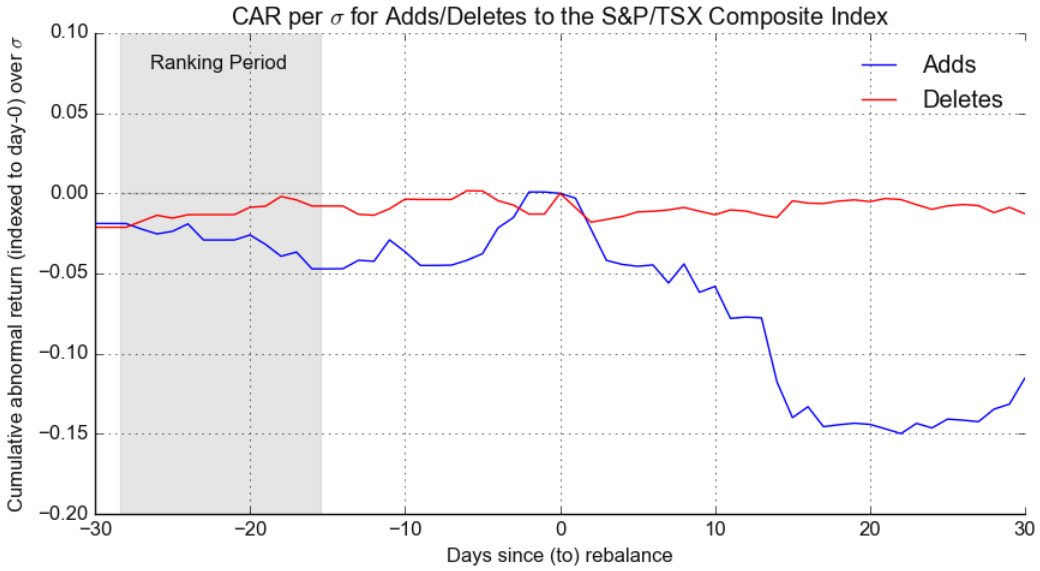


Figure 4.4: Cumulative abnormal returns on a portfolio of stocks being added to or removed from the S&P TSX Composite Index, over the mean standard deviation of returns during the test period.

We see that the cumulative abnormal returns on the portfolio approach -15% of the standard deviation of daily returns by 20 days after the rebalance is effective. So, the cumulative abnormal returns are still less than the standard deviation in daily returns. Without a more robust check for statistical significance, I conclude that there are abnormal returns associated with index rebalances. For that reason, a more rigorous econometric study should be conducted to confirm these qualitative results.

4.5 Volume surprises

These abnormal returns are accompanied with volume surprises. Although trading volumes during the rebalance period are on-par with the norm, volumes on the rebalance day are typically much higher, as in the plot below.

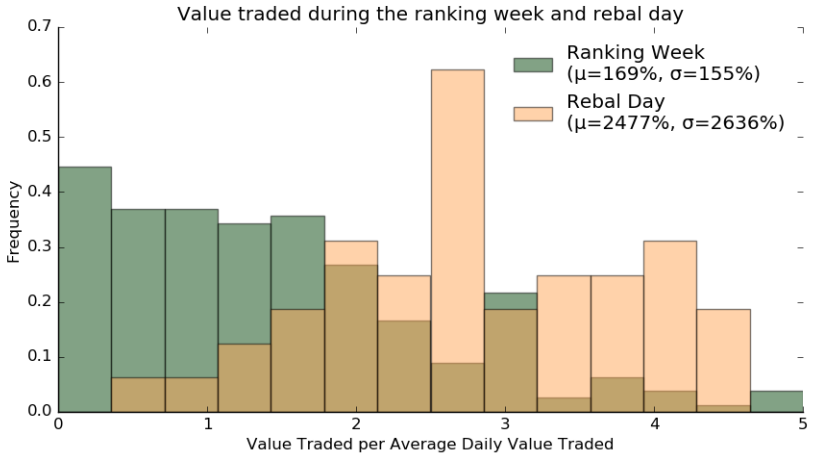


Figure 4.5: Average value traded during the given period versus ADV for all time.

On average, trading volumes on the rebalance day are 2477% average daily volume – however, this figure is skewed considerably by a relatively small number of trading days with very significant volume surprises. Typically, rebalance days will see trading volumes of $2 - 4$ times ADV.

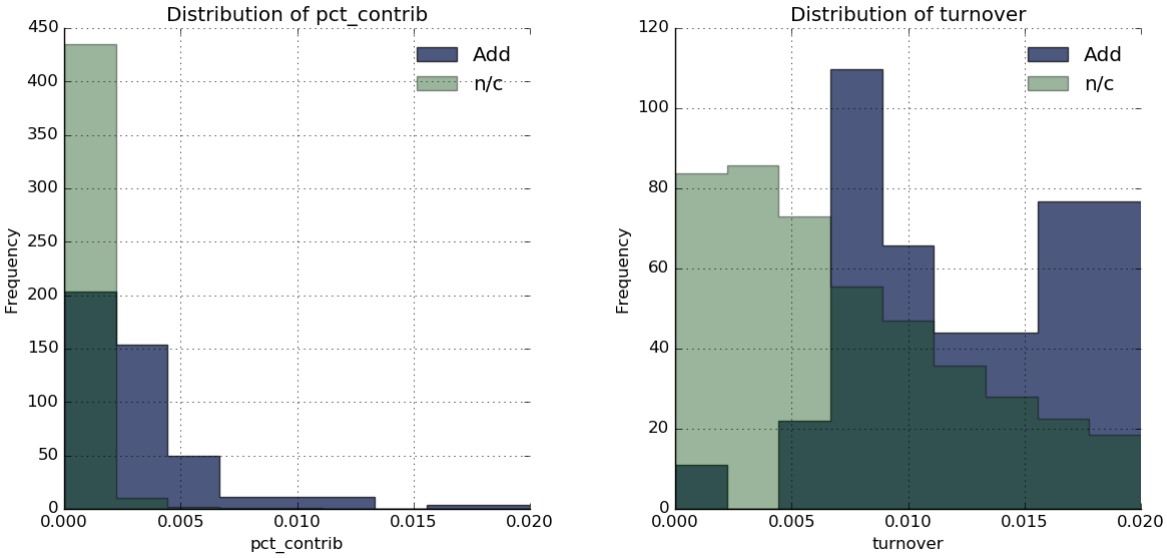
Chapter 5

Model implementation

If it were possible to accurately predict index changes, one could construct a hypothetical strategy in which adds (deletes) are bought (shorted) at least 15-days prior to the rebalance and sold (covered) on the rebalance day.

5.1 Feature exploration

As many of the factors are correlated, for example volume weighted market cap is closely related to the percentage contribution a security's market cap has to the index market capitalisation, there are two main factors which are orthogonal to each other and form a basis for the rest of the features. These are the turnover and the market capitalisation contribution, both of which are plotted below, showing the distribution of security-months which preceded an addition and which preceded a non-change.



(a) Distribution of % mkt. cap. contributions.

(b) Distribution of float turnovers.

Figure 5.1: Distribution of key factors affecting index additions.

This provides some insight into the features which the classification algorithms are likely to identify as significant. In particular, we see that added securities have somewhat higher market capitalisation relative to the index, but they have significantly higher float turnovers. It may be that liquidity is a better predictor of index status change than market capitalisation. This lends support to the theory that there may be other predictors, not associated with those used by the ranking committee, which could have predictive power.

5.2 Predicting index changes

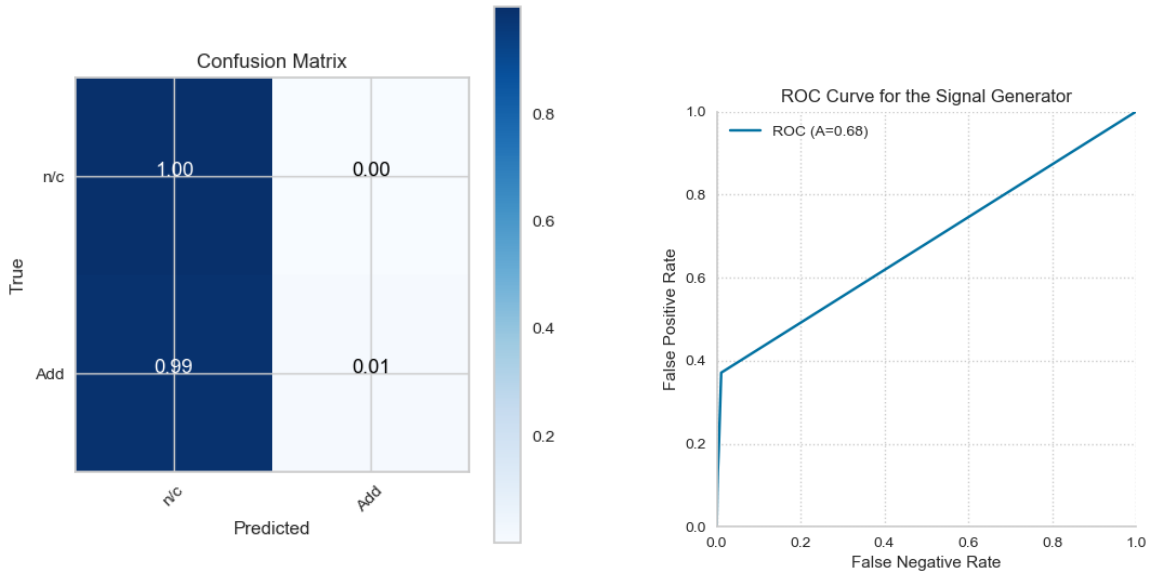
Using the features selected in the metrics section, several binary classification algorithms were applied to the dataset in order to identify securities which are likely to be added to the Composite index.

5.2.1 Binary classification model performance

For each of the models discussed, performance was evaluated on a basis of maximisation of both true positives and true negatives. Optimising on a basis of overall accuracy would tend to produce false negatives, given the imbalanced classes in the sample.

Random forest binary classifier

The random forest classifier performed relatively poorly on the test data. The random forest classifier produced many false negatives. The confusion matrix below shows that the model often classified index additions as not changing. This is similarly reflected in the ROC curve on the right which shows that obtaining a low false negative rate would still necessarily bring along a large number of false positives. Overall, the model trained itself to always predict the status quo result.



(a) Confusion matrix for a binary classifier.

(b) Receiver-operator characteristic for the classifier.

Figure 5.2: Performance measures for a random forest trained on the randomly over-sampled dataset.

One of the advantages to a random forest is that it is possible to obtain the relative importance of the features. More important features are those whose values were more central to the resulting classification.

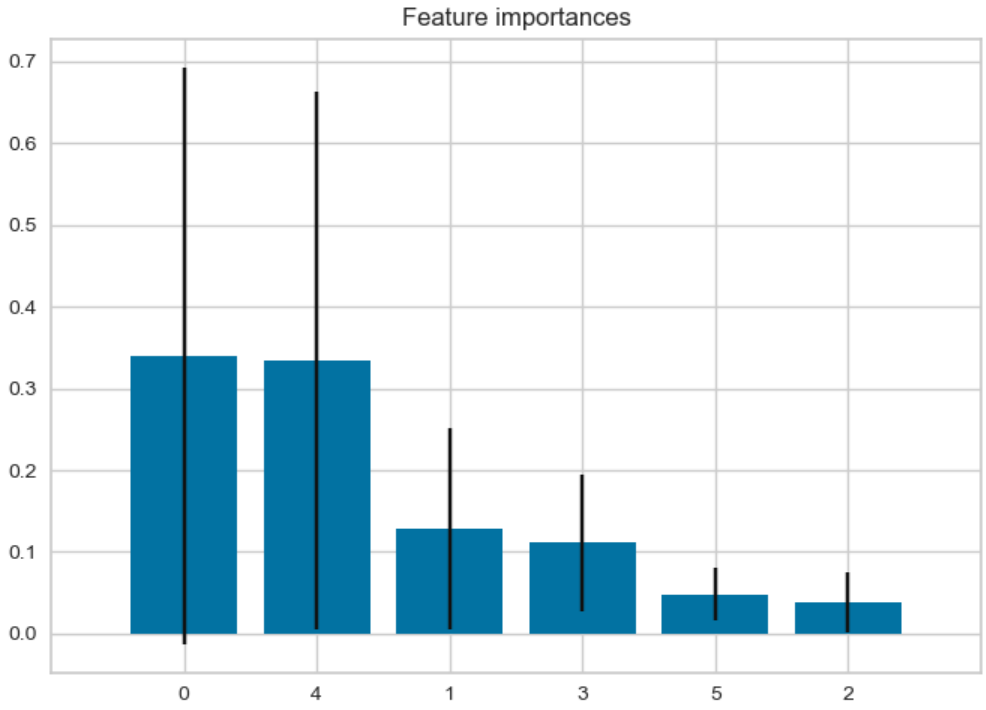


Figure 5.3: Relative importance of the features in the random forest model.

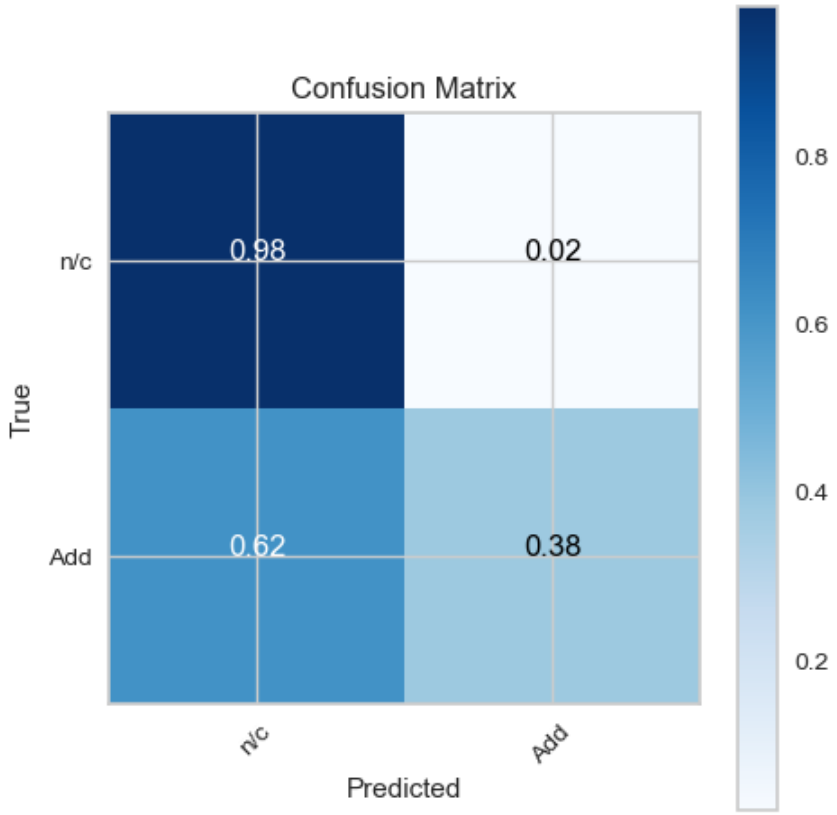
In this plot, the individual features correspond to, in order of highest to least importance:

- percent contribution to the overall index market capitalisation;
- volume-weighted market capitalisation of the security;
- monthly float turnover;
- volume traded in the month;
- and the volume weighted average price of the security.

This shows that the contribution of the security to the index market cap and, surprisingly, liquidity are the most important factors to the classifier.

Support vector machine

The support vector machine model performed reasonably well. Although this model was markedly better than the random forest, it still demonstrated a very high false negative rate of 62%.

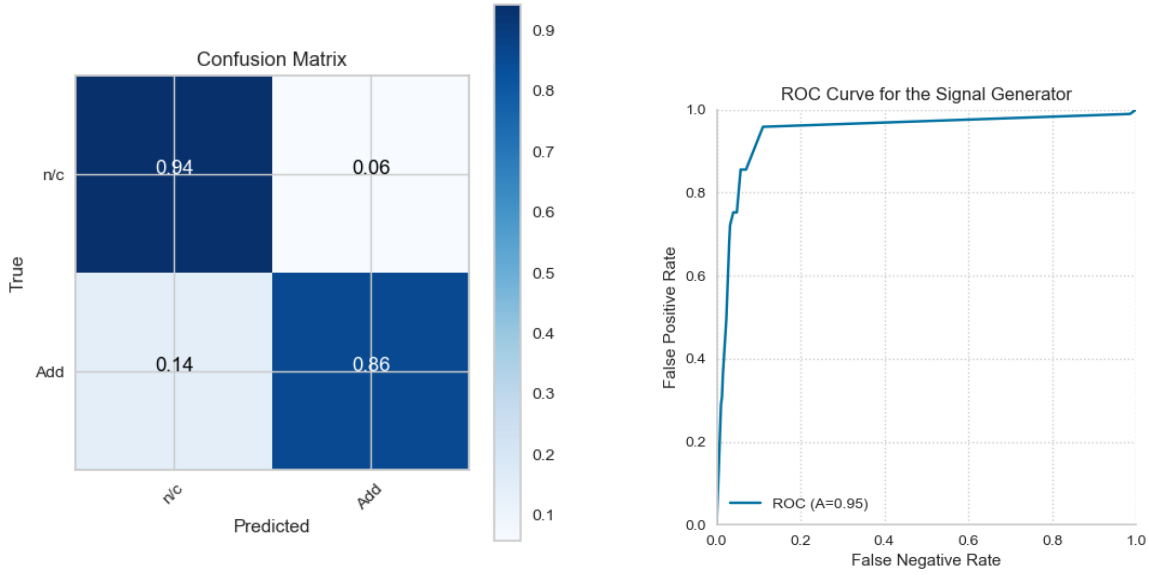


(a) Confusion matrix for a binary classifier.

Figure 5.4: Performance measures for a support vector machine binary classifier trained on the randomly over-sampled dataset.

Gradient boosted tree

A gradient boosted tree is a refinement of the random forest classifier which trains each successive iteration of the tree on previous predictions. This leads to lower bias and over-fitting. This can be implemented much the same as a random forest classifier.



(a) Confusion matrix for a binary classifier.

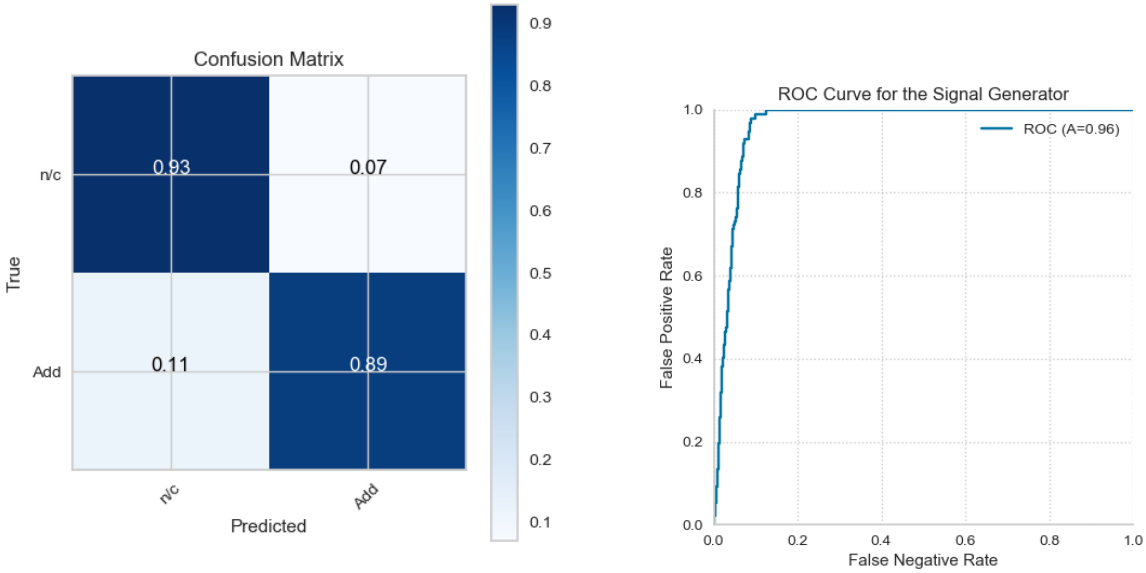
(b) Receiver-operator characteristic for the classifier.

Figure 5.5: Performance measures for a gradient boosted tree classifier trained on the randomly over-sampled dataset.

This model performed relatively well with an area under its ROC curve of 0.95, but still experienced a fairly high false negative rate of 14%: it missed many of the added securities.

Logistic regression

The logistic regression binary classifier, despite being the simplest of the models considered, actually performed the best.



(a) Confusion matrix for a binary classifier.

(b) Receiver-operator characteristic for the classifier.

Figure 5.6: Performance measures for a logistic classifier trained on the over-sampled dataset.

Although the false positive rate was slightly higher than the gradient boosted tree(7& versus 6%), it had a much better false negative rate (only 11% versus 14%) and thus a superior area under its ROC curve (96% versus 95%).

The steepness of the receiver operator characteristic curve also indicates that a near-zero false positive rate could be achieved with only a minor introduction of false negatives.

5.2.2 Model selection

Based on the performance in the classification task – as is summarized below – the logistic regression is likely the best prediction algorithm for index rebalances.

Model	False Negative Rate	False Positive Rate	Area Under ROC Curve
Random Forest	1.00	0.00	0.68
Support Vector Machine	0.62	0.02	n/a
Gradient Boosted Tree	0.14	0.06	0.95
Logistic Regression	0.11	0.07	0.96

Table 5.1: Summary of model performances.

In the next section, these classifiers will be used to construct portfolios, which can be back-tested.

5.3 Strategy implementation

The trading signals for each model were back-tested to identify which of the classification models produced the greatest return on investment. The plots below show the result of $10,000 invested on the first of the month of the rebalance.

Portfolios were constructed from the lists of signals in accordance with the procedure set out in the methods section.

5.3.1 Classifier re-evaluation

Each of the classifiers were tested against historical returns to evaluate whether the decision made in the last section holds.

Random forest classifier

In-line with the poor performance on the classification task, the random forest classifier also produced relatively weak returns. On a risk-adjusted basis, the random forest classifier would likely under-perform the market portfolio.

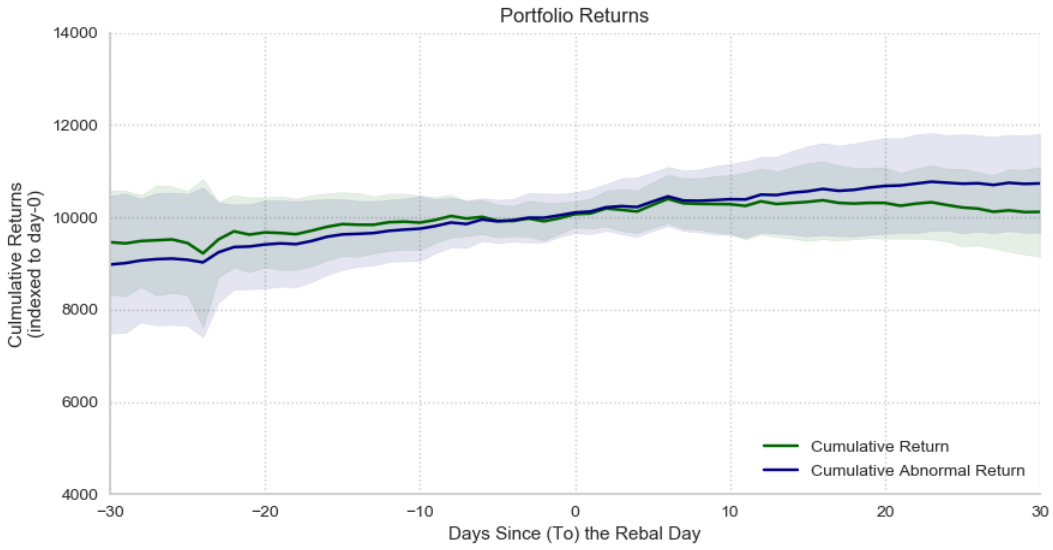


Figure 5.7: Returns on the random forest model, trained on the randomly over-sampled dataset.

Support vector machine classifier

Despite performing relatively poorly on the classification problem, the support vector machine performed well in the portfolio backtest. There does not seem to be an obvious reason for this, as returns on individual securities was not used to train the model.

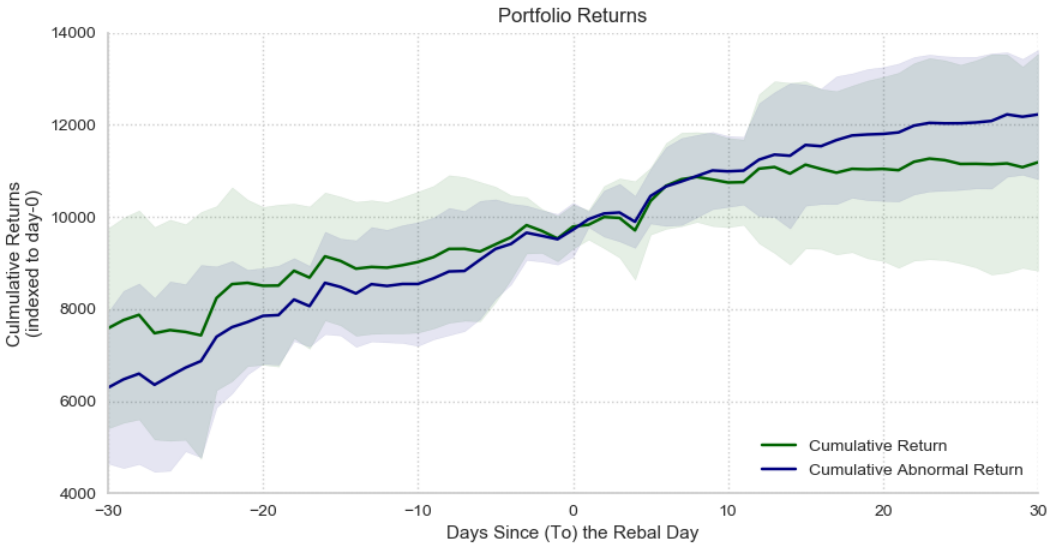


Figure 5.8: Returns on the support vector machine model, trained on the randomly over-sampled dataset.

Working through this problem again, instead training the regressor to future performance (rather than addition) would likely make good use of this unexpected trait of the support vector machine. This may make it useful in a reinforcement learning context.

Gradient boosted tree classifier

Despite doing relatively well on the classification problem, the gradient boosted tree performed relatively poorly on the portfolio backtest – especially when compared to the support vector machine or logistic regression.

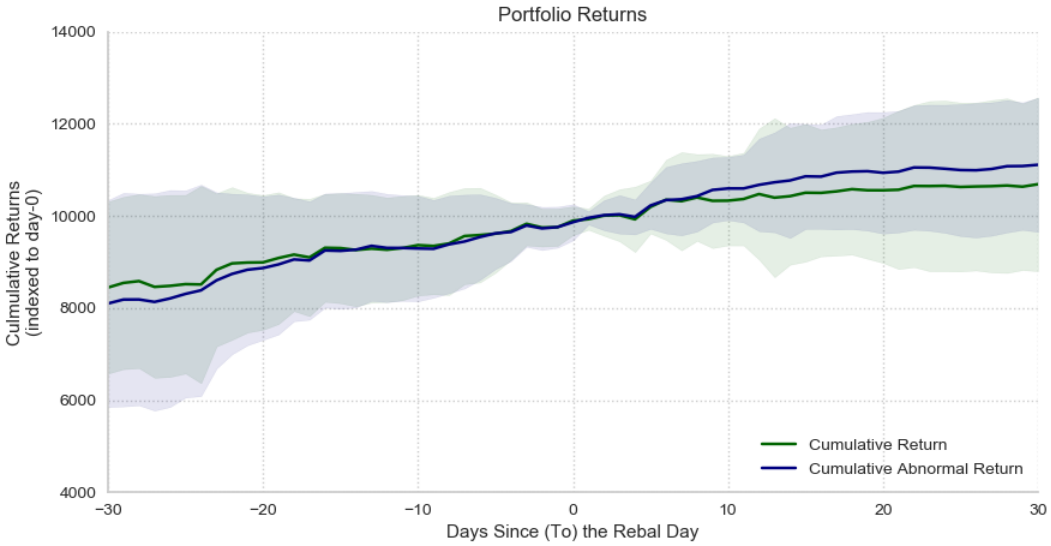
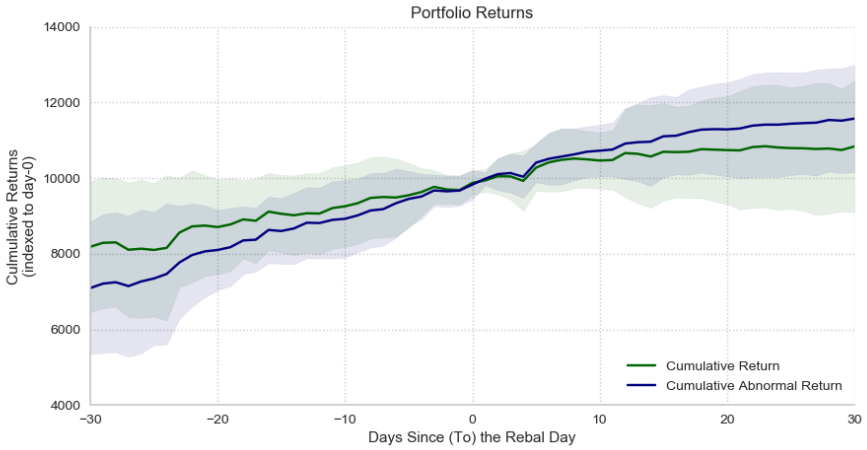


Figure 5.9: Returns on the gradient boosted tree model, trained on the randomly over-sampled dataset.

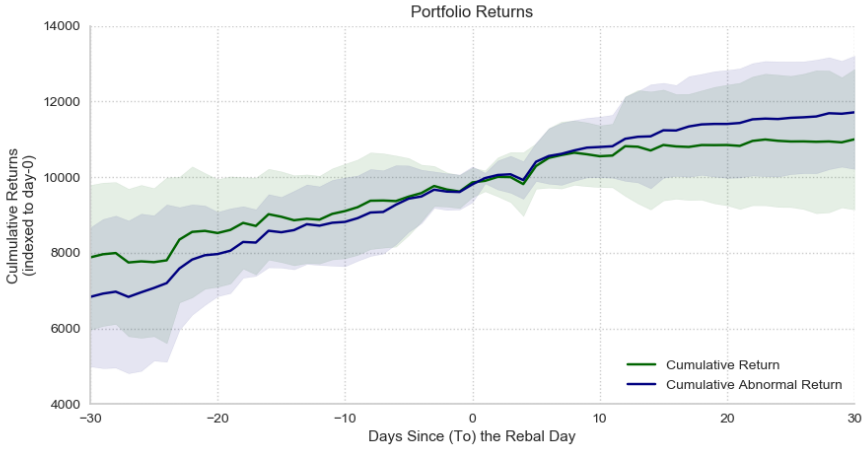
It is probable that the gradient boosted tree was over-fitting to the training dataset, or identifying other extraneous factors which did not contribute to abnormal returns.

Logistic classifier

Despite the marked differences in sampling methodology, both ROS and SMOTE produced comparable return profiles. It is likely that the securities which the logistic regression classified differently under ROS and SMOTE were those which were predicted with small probabilities, and thus would have made up a smaller share of the overall portfolio.



(a) Returns on the logistic regression, trained on the randomly over-sampled dataset.



(b) Returns on the logistic regression, trained on the synthetic minority over-sampled dataset.

Figure 5.10: Backtest performance for the logistic model, for each of the sampling methodologies.

Surprisingly, however, these models actually under-performed the support vector machine model, which did much more poorly on the classification part of the problem.

5.3.2 Model selection

The performances of these models are summarised below:

Model	Cumulative portfolio alpha at d_{30}	Std. dev. of portfolio returns in holding period	Portfolio sharpe ratio
Random Forest	7.21%	6.17%	1.16
Support Vector Machine	14.75%	18.33%	0.80
Gradient Boosted Tree	11.11%	9.55%	1.16
Logistic Regression	15.76%	14.32%	1.10

Table 5.2: Summary of backtest performances.

The sharpe ratios for the Random Forest model is likely relatively high due to its low varience owing to the fact that it produced very few signals.

Despite the support vector machine performing better in the portfolio backtest on an un-adjusted return basis, the logistic classifier on the randomly over-sampled dataset has the most explainable backtest performance and also out-performs SVM on its Sharpe ratio.

For that reason, the logistic classifier trained on the randomly over-sampled dataset is chosen as the final model.

Chapter 6

Conclusions

First, this report identified a methodology for testing whether abnormal returns exist during index rebalances. I found that abnormal returns do indeed exist around rebalance events, although those results may not truly be statistically significant.

Notwithstanding that, second, this report identified an approach to trading on those abnormal returns by predicting which securities out of the entire TSX universe are most likely to be added to the Composite in the next month. This was done using a logistic binary classifier trained on the same factors which the S&P Dow Jones index committee uses to determine eligibility for inclusion. The low likelihood of any random security being added to an index was mitigated through the use of random over-sampling.

Finally, those signals were used to construct a hypothetical portfolio which performed well in a well-constrained back-test.

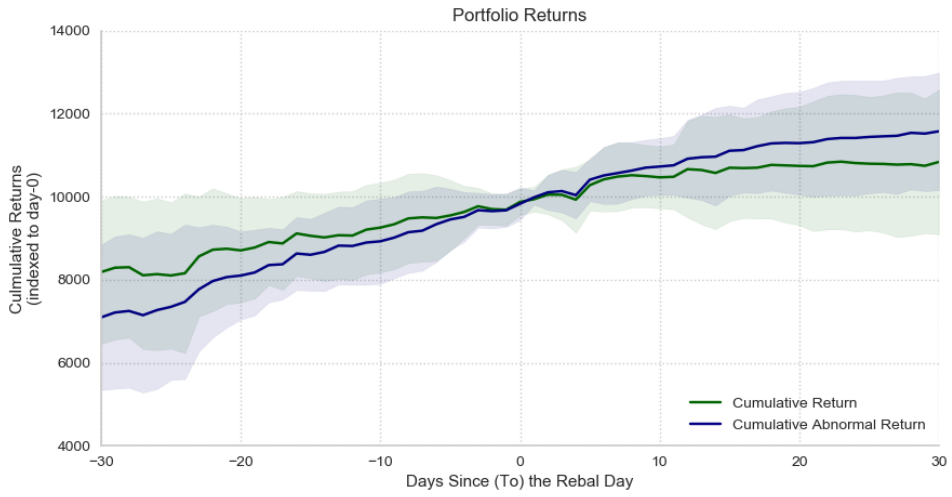


Figure 6.1: Result of a back-test on the portfolio with $\$10,000$ invested, indexed to the trade date.

There is still considerable work which would need to be done to bring such a trading algorithm into production; but this research serves as a proof of concept that a strategy may exist to capitalise on the market inefficiencies around index rebalances.

6.1 Future work

There are several limitations to the proposed strategy. These limitations could be mitigated through further research:

1. First, the hedging assumption is overly burdensome. It may be possible to instead hedge to the overall market, rather than specific GICS sectors.

2. Second, this model does not cover deletions. Deletions occur on an irregular schedule and are more difficult to predict. It may be possible to construct a rolling window where signals are generated.

3. Third, the model generates relatively few signals – only about 20 since 2014. This leads me to suspect that the model may be over-fitting in the backtest. As well, a model with soo few signals is unlikely of economic value, as cash would have to be set aside in the account in the event that a positive signal is identified. However, loosening the threshold for entering a trade produces many erroneous results and actually degrades the overall alpha profile of the strategy. More work is needed to tweak the portfolio construction part of this exercise.

4. Finally, the strategy misses out on another $\approx 10\%$ potential return by not entering a trade until the end of the ranking window. Of course, earlier in the ranking period there is naturally less certainty about which securities may or may not be added to the index. This uncertainty may imply a commensurate increase in risk as I try to seek that additional marginal alpha. An experiment could be constructed by again running the model on a rolling window and allowing it to identify potential trade signals before the end of the ranking period, if it does so with sufficient confidence.

As well, I was approached by Two Sigma, who expressed interest in a partnership in which I would deliver buy/sell signals, which they would test and evaluate with the expectation of building a trading strategy around my work. I was not convinced of the value proposition for me (I was not keen to hand over my signals for free), and so I decided not to pursue this opportunity any further. That may be interesting, nevertheless.

Bibliography

Agarwal, Alekh, Peter Bartlett, and Max Dama (2010). "Optimal Allocation Strategies for the Dark Pool Problem". In: URL: http://arxiv.org/abs/1003.2245.

Almgren, Robert (2009). *High Frequency Volatility*. Tech. rep. New York: New York University. URL: https://pdfs.semanticscholar.org/e9ac/48acf227dd64a518061cab5f7736b9e99218.pdf.

Almgren, Robert and Neil Chriss (1999). *Optimal Execution of Portfolio Transactions*. Tech. rep. URL: https://pdfs.semanticscholar.org/3d2d/773983c5201b58586af463f045befae5bbf2.pdf.

Arnott, Rob, Vitali Kalesnik, and Lillian Wu (2018). *Buy High and Sell Low with Index Funds!* Tech. rep. June. Newport Beach: Research Affiliates, LLC.

Azar, Joss, Martin C. Schmalz, and Isabel Tecu (2014). "Competitive Effects of Common Ownership: Evidence from the Airline Industry". In: *SSRN Electronic Journal*. ISSN: 1556-5068. DOI: 10.2139/ssrn.2427345. URL: http://www.ssrn.com/abstract=2427345.

Backus, Matthew, Christopher Conlon, and Michael Sinkinson (2019). *Common Ownership in America: 1980-2017*. Tech. rep. 25454. Cambridge, MA: National Bureau of Economic Research. DOI: 10.3386/w25454. URL: http://www.nber.org/papers/w25454.

Bergstra, J., D. Yamins, and D. D. Cox. "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures". In: *Proceedings of the 30th International Conference on Machine Learning*. Vol. 28. Journal of Machine Learning Research.

BetaPro SP500 VIX Short-Term Futures ETF Prospectus (2018). Tech. rep. Toronto: Horizons Exchange Traded Funds. URL: www.sedar.com..

Blocher, Jesse, Robert E Whaley, and Valere Blair. *Passive Investing: The Role of Securities Lending*. Tech. rep. URL: http://online.wsj.com/news/article_email/investors-pour-into-vanguard-eschewing-stock-pickers-1408579101-.

Boehmer, Ekkehart, Jim Masumeci, and Annette B. Poulsen (1991). "Event-study methodology under conditions of event-induced variance". In: *Journal of Financial Economics* 30.2, pp. 253–272. DOI: 10.1016/0304-405X(91)90032-F. URL: https://www.sciencedirect.com/science/article/pii/0304405X9190032F.

Borkovec, Milan and Konstantin Tyurin (2015). "Is Volatility Your Nemesis or Best Friend? It Depends on Who You Ask". In: *The Journal of Trading* 11.1, pp. 13–25. DOI: 10.3905/jot.2016.11.1.013. URL: http://jot.iijournals.com/lookup/doi/10.3905/jot.2016.11.1.013.

Cavaglia, Stefano, Christopher Brightman, and Michael Aked (2000). "The Increasing Importance of Industry Factors". In: *Financial Analysts Journal* September/October, pp. 41–54. URL: https://www.researchaffiliates.com/documents/FAJ_Sep_Oct_2000_The_Increasing_Importance_of_Industry_Factors.pdf.

Chan, Kalok, Hung Wan Kot, and Gordon Y.N. Tang (2013). "A comprehensive long-term analysis of S&P 500 index additions and deletions". In: *Journal of Banking & Finance* 37.12, pp. 4920–4930. ISSN: 03784266. DOI: 10.1016/j.jbankfin.2013.08.027. URL: http://linkinghub.elsevier.com/retrieve/pii/S0378426613003592.

Chawla, N. V. et al. (2002). "SMOTE: Synthetic Minority Over-sampling Technique". In: *Journal of Artificial Intelligence Research* 16, pp. 321–357. ISSN: 1076-9757. DOI: 10.1613/jair.953. URL: https://jair.org/index.php/jair/article/view/10302.

Chen, Honghui, Gregory Noronha, and Vijay Singal (2004). *The Price Response to S&P 500 Index Additions and Deletions: Evidence of Asymmetry and a New Explanation*. Tech. rep. 4. The American Finance Association, pp. 1901–1929. URL: https://www-jstor-org.myaccess.library.utoronto.ca/stable/pdf/3694882.pdf?refreqid=excelsior%3A254227701e007556c0bd626ffd6832b1.

Clark, Doug and Ivan Cajic (2018). *Phantom Promotions, Defiant Demotions*. Tech. rep. November. Toronto: Investment Technology Group, pp. 1–7.

Clark, Doug, Ivan Cajic, et al. (2016). *The Curious Case of Missing Equity Volumes*. Tech. rep. January. Toronto: Investment Technology Group, pp. 1–8.

Cont, Rama and Arseniy Kukanov (2012). "Optimal order placement in limit order markets". In: URL: http://arxiv.org/abs/1210.1625.

Espen Eckbo, B and S Thorburn Karin (2000). ""Gains to Bidder Firms Revisited: Domestic and Foreign Acquisitions in Canada"". In: *"Journal of Financial and Quantitative Analysis"* 35, pp. 1–25.

Fama, Eugene F. et al. (1969). "The Adjustment of Stock Prices to New Information". In: *International Economic Review* 10. DOI: 10.2139/ssrn.321524. URL: http://www.ssrn.com/abstract=321524.

Fichtner, Jan, Eelke M. Heemskerk, and Javier Garcia-Bernardo (2017). "Hidden power of the Big Three? Passive index funds, re-concentration of corporate ownership, and new financial risk". In: *Business and Politics* 19.02, pp. 298–326. ISSN: 1469-3569. DOI: 10.1017/bap.2017.6. URL: https://www.cambridge.org/core/product/identifier/S1469356917000064/type/journal_article.

Frazzini, Andrea, Ronen Israel, and Tobias J. Moskowitz (2018). "Trading Costs". In: *SSRN Electronic Journal*. ISSN: 1556-5068. DOI: 10.2139/ssrn.3229719. URL: https://www.ssrn.com/abstract=3229719.

Gu, Angela and Patrick Zeng (2014). "Sector-Based Factor Models for Asset Returns". In: URL: http://arxiv.org/abs/1408.2794.

Guéant, Olivier, Charles-Albert Lehalle, and Joaquin Fernandez Tapia (2011). "Optimal Portfolio Liquidation with Limit Orders". In: DOI: 10.1137/110850475. URL: http://arxiv.org/abs/1106.3279%20http://dx.doi.org/10.1137/110850475.

Healy, Paul M., Krishna G. Palepu, and Richard S. Ruback (1992). "Does corporate performance improve after mergers?" In: *Journal of Financial Economics* 31.2, pp. 135–175. ISSN: 0304405X. DOI: 10.1016/0304-405X(92)90002-F. URL: http://linkinghub.elsevier.com/retrieve/pii/0304405X9290002F.

Hendricks, Dieter and Diane Wilcox (2014). "A reinforcement learning extension to the Almgren-Chriss framework for optimal trade execution". In: *2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)*. IEEE, pp. 457–464. ISBN: 978-1-4799-2380-9. DOI: 10.1109/CIFEr.2014.6924109. URL: http://ieeexplore.ieee.org/document/6924109/.

Kearns, K J et al. (2010). "Censored Exploration and the Dark Pool Problem". In: *CACM*. DOI: 10.1145/1735223.1735247. URL: http://repository.upenn.edu/cis_papershttp://repository.upenn.edu/cis_papers/658.

Kearns, Michael and Yuriy Nevmyvaka (2013). *High Frequency Trading - New Realities for Traders, Markets and Regulators*. Ed. by David Easley, Lopez Marcos de Prado, and Maureen OHara. Risk Books. Chap. Machine Learning for Market Microstructure and High Frequency Trading. ISBN: 9780124859678,0124859674. URL: http://riskbooks.com/book-high-frequency-trading.

Kissell, Robert and Jungsun "Sunny" Bae (2018). "Machine Learning for Algorithmic Trading and Trade Schedule Optimization". In: *The Journal of Trading* 13.4, pp. 138–147. ISSN: 1559-3967. DOI: 10.3905/jot.2018.13.4.138. URL: https://jot.iijournals.com/content/13/4/138.

Kolanovic, Marko and Rajesh Krishnamachari (2017). *Morgan Stanley report—Big Data and AI Strategies*. Tech. rep. May. J.P. Morgan.

Mohanani, Rahul et al. (2017). "Cognitive Biases in Software Engineering: A Systematic Mapping Study". In: DOI: 10.1109/tse.2018.2877759. URL: https://arxiv.org/abs/1707.03869.

Multi-criteria analysis (2009). Tech. rep. London: Department for Communities and Local Government. URL: www.communities.gov.ukcommunity,opportunity,prosperity.

Nevmyvaka, Yuriy, Yi Feng, and Michael Kearns (2006). *Reinforcement Learning for Optimized Trade Execution*. Tech. rep. URL: https://www.seas.upenn.edu/~mkearns/papers/rlexec.pdf.

Park, Beomsoo and Benjamin Van Roy (2012). "Adaptive Execution: Exploration and Learning of Price Impact". In: URL: http://arxiv.org/abs/1207.6423.

Pearson, Phil et al. (2018). *Optimal Order Execution: A reinforcement learning approach*. Tech. rep. New York: ITG. URL: www.itg.com.

Posner, Eric A., Fiona M. Scott Morton, and E. Glen Weyl (2016). "A Proposal to Limit the Anti-Competitive Power of Institutional Investors". In: *Ssrn*. ISSN: 1066-2243. DOI: 10.2139/ssrn.2872754. URL: http://www.ssrn.com/abstract=2872754.

Ritter, Gordon (2017). "Machine Learning for Trading". In: *SSRN Electronic Journal*. ISSN: 1556-5068. DOI: 10.2139/ssrn.3015609. URL: https://www.ssrn.com/abstract=3015609.

"Slow-motion revolution" (2016). In: *The Economist*, p. 11. URL: http://viewswire.eiu.com/index.asp?layout=VWArticleVW3&article_id=874300071.

S&P/TSX Canadian Indices Methodology (2018). Tech. rep. S&P Dow Jones Indices. URL: https://ca.spindices.com/indices/equity/sp-tsx-60-index.

Sushko, Vladyslav and Grant Turner (2018). "The equity market turbulence of 5 February - the role of exchange-traded volatility products". In: *BIS Quarterly Review*. URL: https://www.bis.org/publ/qtrpdf/r_qt1803t.htm.

TMX Group (2017). *Order Types and Functionality Guide*. Tech. rep. Toronto: TMX Group. URL: http://tmx.complinet.com/en/tsxv_rulebook.html.

Transcript of FTC Hearings Session #8: Competition and Consumer Protection in the 21st Century (2018). New York. URL: https://www.ftc.gov/system/files/documents/public_events/1422929/ftc_hearings_session_8_transcript_12-6-18.pdf.

Zoph, Barret and Quoc V. Le (2016). "Neural Architecture Search with Reinforcement Learning". In: URL: https://arxiv.org/abs/1611.01578.

Appendix A

Other rebalance effects

This appendix describes some effects of rebalances on other market structures and also identifies some potential areas for future research.

A.1 Other market structures considered

Although the main focus of this event study was the cumulative abnormal returns associated with index rebalances, there were several other effects that were identified.

A.1.1 Securities that are added and then subsequently removed

There were 26 securities in the same which were both added to and removed from the Composite during the study period. It was more typical that a security be added and then removed (e.g., in the case of ACQ, AKG, AR, AVO, BDI, BXE, CF, CXR, FRC, HNL, IAE, KDX, NAL, P, PLI, PSG, PSN, RIO, SGY, WJX) than to be removed and then re-added (e.g., CLC, FVI, GUY, NVA, POU, TCW).

For most analyses, these securities were simply removed from the sample space. However, it would be interesting to study the difference in returns during the addition and the deletion. This would provide a direct comparable.

A.1.2 Assessing liquidity risk

First, we observed that the top of the book in the security is considerably deeper on the rebalance day than during the ranking week. This is suggestive of the increased liquidity commensurate with indexing flows.

There are no clear trends for spreads or time at NBBO for this single security.

This is consistent with the unusually high and variable volumes in securities when they are added to or removed from an index.

Side	Ranking Week	Rebal Day
Deletes	94% ($\sigma = 79\%$)	761% ($\sigma = 578\%$)
Adds	234% ($\sigma = 176\%$)	3970% ($\sigma = 2814\%$)

Table A.1: Average value traded during the given period versus ADV for all time.

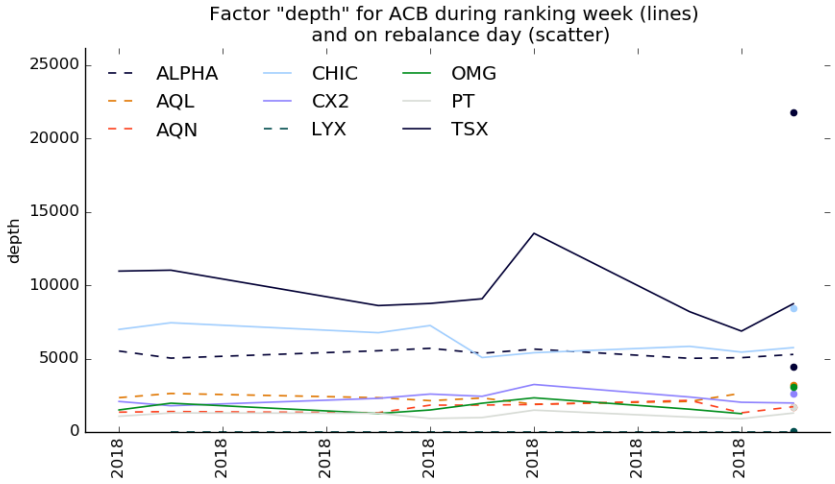


Figure A.1: Time-weighted top-of-book depth on ACB during the ranking weeks (lines) and on the rebalance day (scatter on the right showing values for 16 March 2018).

It is somewhat surprising that considerably greater volumes are traded on the rebalance day, despite the fact that all available information about the security's index inclusion should already have been priced-in.

As well, using this dataset, we can compute other typical metrics for liquidity, such as spread. However, presenting this data in a sensible form has proved challenging.

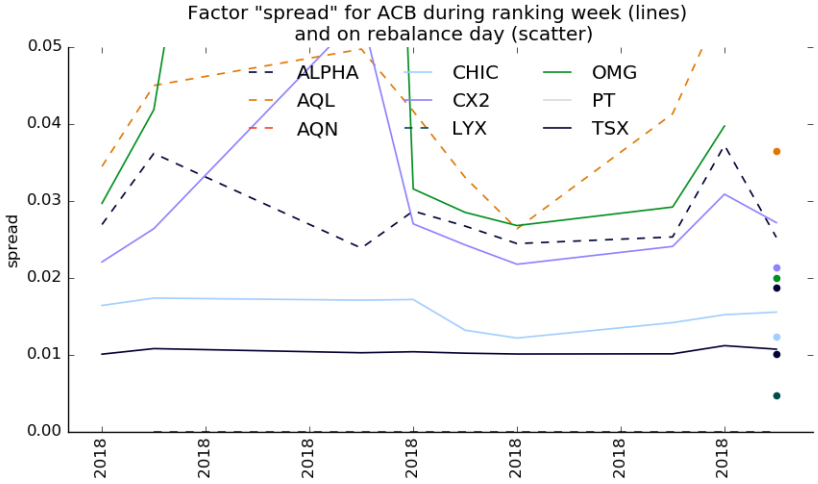


Figure A.2: Time-weighted spread on ACB during the ranking weeks (lines) and on the rebalance day (scatter on the right showing values for 16 March 2018).

Next steps on this front will include costing the liquidity risk of trading the rebalance.

A.1.3 Changes in β after rebalances

Logically, when a security is added to an index, it will be purchased by indexers whose inflows (outflows) are felt across the entire market. Accordingly, securities should be more closely correlated to the market when included in an index. However, inclusion in an index is likely indicative of stability in returns.

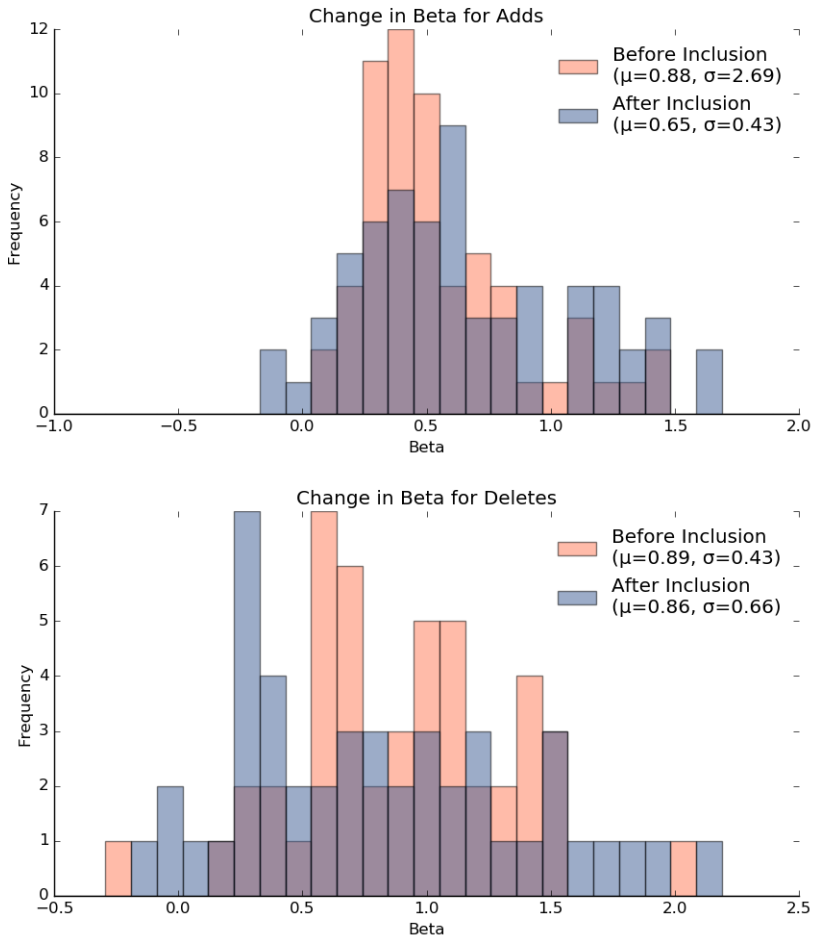


Figure A.3: Distribution of security betas before and after addition/removal from the TSX Composite.

Empirically, we see securities on average have lower and more consistent betas after they are added to the TSX Composite. That is to say, as securities join the Composite, their returns come in-line with the market.

On the delete side, counter-intuitively, we see much greater variability: their returns seem to trend away from the market both positively and negatively. This also emphasises the bifurcation in returns on deletes we observed in the overall alpha visualisation.

A.1.4 Reversion after ranking

These alpha trends beg the question of whether there exists reversion from the end of the ranking week to the rebalance day. I do indeed observe reversion on adds between the end of the ranking week and the rebalance day.

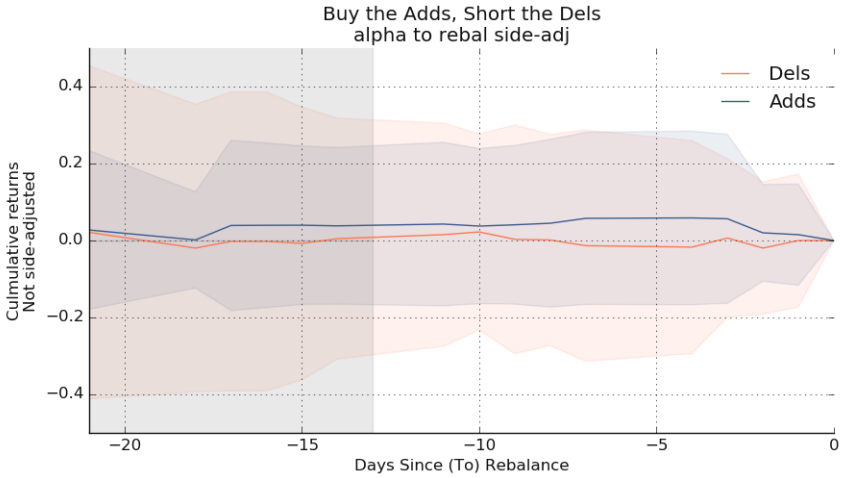


Figure A.4: Returns on adds and deletes from the end of the ranking week to the rebalance close.

The fuzziness of this plot obscures the distribution of returns.

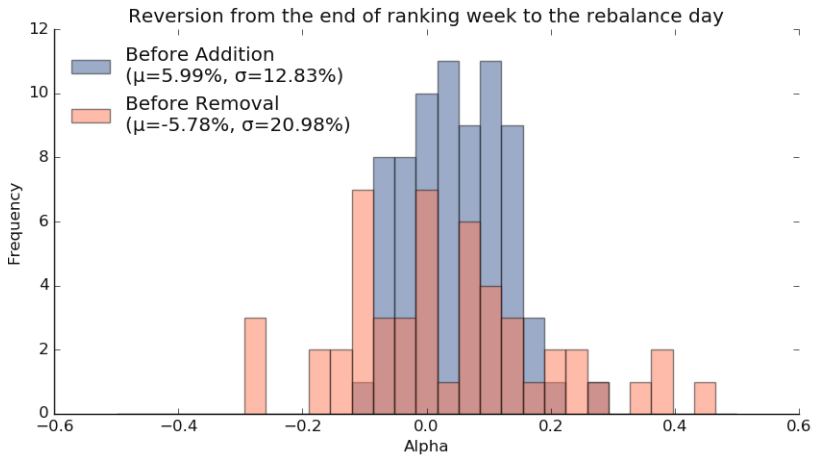


Figure A.5: Returns on adds and deletes from the end of the ranking week to the rebalance close.

Adds trade down an average of 6.0% from the end of the ranking week to the rebalance day, while deletes trade up an average of 5.8%[1]. Granted, the distributions are still rather wide: I will have to use a more rigorous test of statistical significance.

[1] Unintuitive sign notation strikes again!

A.1.5 Pre-positioning under volatility

There is an anecdotal belief that volatility reduces the extent of prepositioning, because there is greater risk associated with taking a position ahead of a less certain rebalance.

Volatility during rebalance events was investigated to determine whether the degree of pre-positioning or reversion is affected.

Alpha as a proxy for pre-positioning

First, we see that greater volatility leads to a greater absolute value on alpha in the week prior to the rebalance. However, directionally, there is no clear relationship. It is possible that greater risk incidentally increases the magnitude of returns if volatility drives price delta.

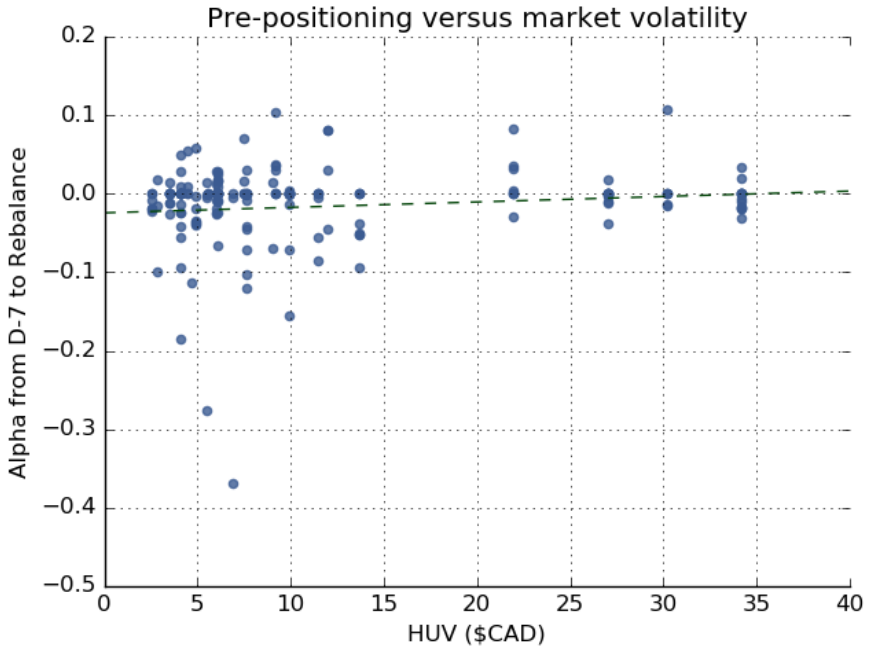


Figure A.6: Alpha from one week prior to rebalance until rebalance day, regressed against the price of the BetaPro S&P 500 VIX Short-Term Futures ETF product, as a proxy for market volatility.

This fit is given by the insignificant relationship ($R^2 = 0.002$):

$$\alpha = 692\ P_{HUV} \times 10^{-6} - 0.0243 \tag{A.1}$$

Volume traded as a proxy for pre-positioning

However, plotting volatility against the average daily value traded in the ranking week tells a different story. There is greater value traded in the ranking week when the average volatility is higher. This seems to suggest a greater degree of prepositioning when volatility is high. However, again, the volume may be only incidental to volatility, as greater value is typically traded when volatility is higher, regardless of indexing flows.

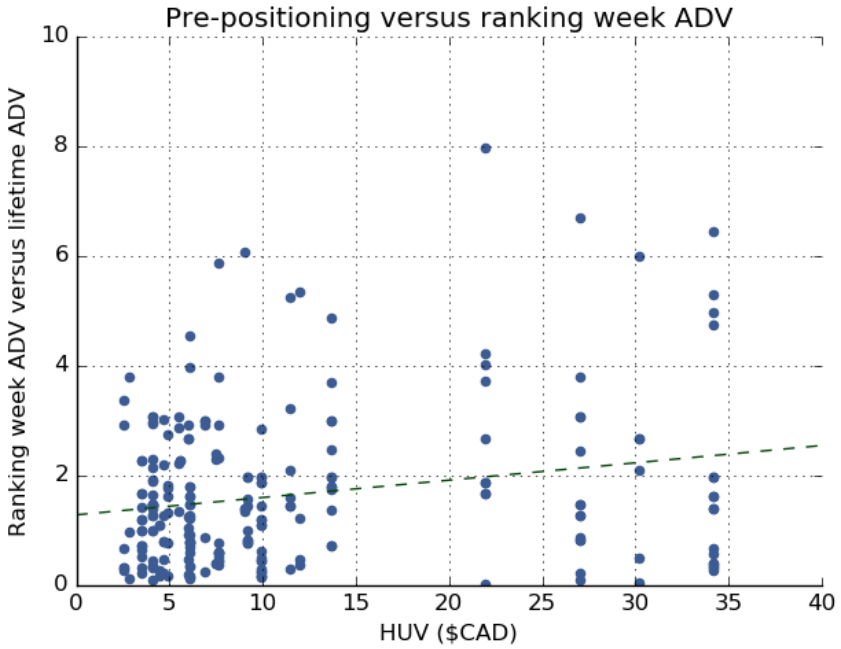


Figure A.7: Average daily volume from the ranking week versus lifetime ADV, regressed against the price of the BetaPro S&P 500 VIX Short-Term Futures ETF product, as a proxy for market volatility.

This fit is given by the relationship, which may in fact be worth investigating further ($R^2 = 0.046$):

$$ADV = 3.16\%\ P_{HUV} + 128.5\% \tag{A.2}$$

Notably, this would imply that in the minimal-risk limit, value traded on the rebalance day would only barely exceed the average daily value traded.

A.1.6 Bifurcations

There is, however, a remarkable bifurcation in returns between adds and deletes.

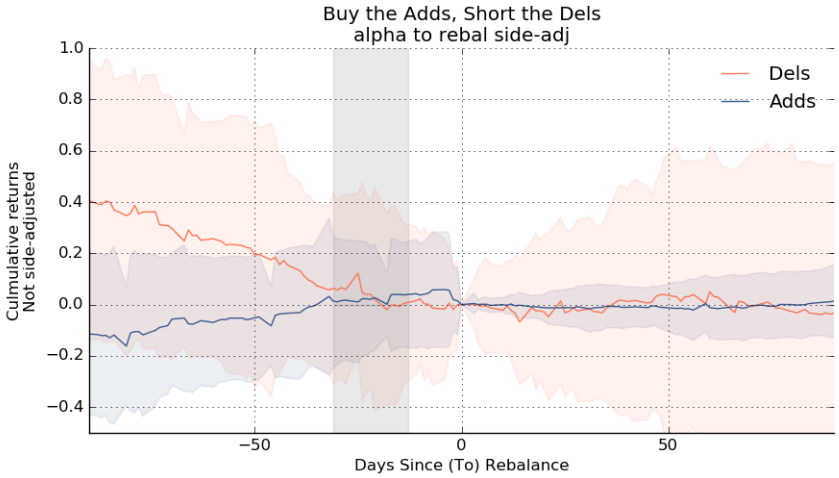


Figure A.8: Returns (not β-adjusted) and alpha (per above) for the entire rebalance list, per side, where Day 0 is the rebalance day and the shaded period is the ranking period. The shaded regions adjacent to the alpha and return reflect a standard deviation.

Here, we see prices on adds trending slowly upwards during the period leading up to the ranking week.[2] That is to say, securities are added to the index after a period of only moderate out-performance. Once added to an index, the returns on the adds are very consistent.

However, we see the deletes perform quite badly during the period leading up to their removal. This may be explained by the fact that the S&P rules are rather lenient on deletes – it is much more difficult to be added to the Composite than to be removed from it.

Also note that the returns on deletes are much more variable after their removal from the index. This may be explained by external factors, such as the fact that securities which are removed from the Composite are likely going to continue to experience considerable volatility.

[2]This plot uses an opposite side convention to the one on the previous page: a rising trend implies a rising price while a falling trend – such as in the deletes – shows a falling price.

A.2 Market structures not explored in depth

The two most significant factors which remain unexplored are the effects of volatility on prepositioning and the degree of liquidity risk during rebalance events.

Other smaller factors to investigate include changes in volatility/reversion after the introduction of the limit-on-close order type and changes in reversion after the benchmark price VWAP window was extended.

At a high-level, I should also weight my calculations by some sensible factor (possibly share of the index). At the moment, I use an arithmetic mean for aggregating alphas.

A.2.1 Volatility

I still do not entirely understand the relationship between volatility, prepositioning, and abnormal returns. There are some incidental correlations, but I am not very confident in these.

First, I should identify a better proxy for market volatility than a VIX-tracking exchange traded product. I will probably pull intraday price data from Grapevine and calculate the intraday volatility per Almgren's time series lecture notesAlmgren 2009, though this will be a time consuming endeavour.

This investigation will likely necessitate conducting a literature review on the effects of various portfolio hedging strategies on the cash equities market. I do not know exactly where to begin with this research, but it will probably begin by reading about Vega/Vomma hedging reminiscent of the breakdown of XIV ETPs on 5 February 2018.

Liquidity risk

Liquidity risk will be explored more thoroughly. Now that much of the groundwork has been laid to pull data from the TMX Analytics Grapevine platform, it should be relatively straightforward to script batch jobs to pull data about market structure around rebalance events, such as:

1. which institutions are most active (i.e., do broker numbers leak information from more informed traders); and

2. are there increased trading costs during the ranking week or rebalance day – this could potentially implement the Almgren-Chriss modelAlmgren and Chriss 1999, although I am not particularly partial to any given approach.

A.2.2 Divergence of security returns from the index

I did not yet make progress on regressing excess returns per security against its peer group. At first, I attempted to compute returns on baskets of securities based on the group in the TMX Grapevine issuers table; however, these groupings are far too granular to be of particular use. My next approach was to cluster securities based on their covariances, but this raises the question of how to split the data that is used to calculate covariances from the data used to predict index changes. I will probably have to read some econometrics texts to better understand this.

A.2.3 Other factors not considered

Introduction of the limit on close order type

In June 2015, the TSX introduced Limit-on-Close (LOC) order types[3]. There may have been a reduction in volatility in the close on rebalance day after this order type was introduced. I have not made much progress on this so far.

Lengthening the period of VWAP calculation

Increasing the period in which VWAP is computed to determine eligibility for inclusion in the Composite may have reduced reversion, as it would have been more difficult to manipulate a stock's price in order to force inclusion.

Accounting for financing flows

When a security is added to the Composite, the total value of the funds which track the index must necessarily remain the same. Therefore, funds must sell other index constituents in order to raise funds to purchase the new addition. Similarly, when a security is removed from the Composite, funds purchase shares in the remaining stocks. The scale of this effect must be investigated to determine whether it affects my model, but I have not yet done so.

Liquidity data

Liquidity summary statistics were obtained from the TMX Grapevine platform for each of the days in the ranking week and for the rebalance day for each security added to (removed from) the TSX Composite in the study period. These summary statistics included the time-weighted spread, top-of-book depth, depth at NBBO, and presence of a quote for each Canadian venue.

This script was made to work only recently and is presently still running, as the dataset is extremely large. Since the set runs alphabetically, it is first possible to study liquidity shifts in ACB when it was added to the TSX Composite on 16 March 2018, at the height of the TSX cannabis market.

[3]The functionality of limit on close orders is described in the TSX Order Types and Functionality Guide.TMX Group 2017

Appendix B

The machine learning design process

This design process is adapted from work done for the Engineering Science (Physics) Capstone Design project. The overall design process used in this thesis involves the four distinct *phases* which are in turn broken down into a total of six distinct *stages*. For each of these *stages* we provide a description of the engineer's activities, as well as a set of necessary considerations to complete the stage within the broader design process.

B.1 Problem definition

1. Determine relevant stakeholders
    - Relevant stakeholders may include potential users, data sources, those affected by the model's decisions, and the engineering designers themselves
2. Build engineering requirements model for the design problem
    - Define objective(s) for the MI model, including:
        - Categorize the problem as classification or regression
        - Determine how interpretable the results need to be
    - Establish metrics to score solutions
    - Establish criteria to connect solution scores (based on metrics) to design objectives
    - Define constraints based on stakeholder requirements, including performance, accuracy and risk
        - Risk constraints should be based on the acceptable worst-case consequences

B.2 Data definition

1. Understand the available data
    - Understand the recency of data, method of collection, biases of data collector, and statistical biases present

2. Determine the minimum viable dataset
    - Consider availability bias (data that is easier to collect influences choice of data)
    - If a representative minimum viable dataset is not available, employ techniques to rebalance data distributions (regression) or classes (classification)
3. Acquire the data
    - Determine whether the data is readily available or if it needs to be collected
    - If using primary data, employ traditional methods of data acquisition
    - If using secondary data, download or acquire license
4. Process the data and feature engineer
    - Format the data into a usable form (e.g. CSV, XML, SQL/NoSQL database)
    - Understand the features of the data set by plotting trends and correlations
    - Use singular value decomposition, regression trees and f-scores to understand most significant features and perform dimensionality reduction
    - Generate sample statistics to compare to population statistics to reveal any biases
    - Normalize and standardize data
    - Perform data split for training, test and cross-validation by determining split and cross-validation technique (number of folds in cross-validation)
    - Explore whether data requires aggregation, discretization, abstraction, or computation to generate new input features to the model

B.3 Model archetype exploration

1. Generate design space of possible model archetypes
    - Ensure finiteness of the design space by applying constraints from engineering requirements model
    - Address anchoring bias by simultaneously refining requirements and generating potential solutions (co-evolution) Mohanani et al. 2017
    - Avoid a biased selection of archetypes by examining the prioritization of use cases and considering that past model performance is not indicative of future model performance
2. Consult the latest literature for the state of the art in specific or similar domains (academic or practical literature)
    - Explore transfer learning by using an existing out-of-the-box model in new domain
    - Start with a pre-trained model and train model further with domain dataset
3. Iterate and refine design space of model archetypes
    - Consider bias-variance tradeoffs
    - Consider precision and recall

B.4 Model archetype hyperparameter optimization

1. Create multiple potential solutions for each potential model archetype
    - Use appropriate search for hyperparameter space Bergstra, Yamins, and Cox n.d.
    - Consider available computational and time resources to allocate to the hyperparameter optimization
2. Utilize traditional and Auto ML methods to select optimal hyperparameters and model architecture
    - Select appropriate AutoML search technique for chosen model Zoph and Le 2016
    - Compare Auto ML generated models to hand-tuned models, in relation to engineering requirements model

B.5 Model selection

1. Use defined metrics to score potential solutions
    - Quantitative metrics may include Classification Accuracy, Logarithmic Loss, Confusion Matrix, Area under Curve, F1 Score, Mean Absolute Error, and Mean Squared Error
2. Use well-developed multiple-criteria decision analysis tools and defined criteria to compare solutions appropriately and holistically
    - E.g. Pugh charts, pairwise comparison matrices *Multi-criteria analysis* 2009
3. Make decisions about the acceptability of trade-offs
    - Evaluate models against constraints and re-adjust thresholds as necessary
    - Base decisions off how representative the training data set is of the population, expected variability of input, and how robust the model needs to be based on bias- variance tradeoffs for each model type
4. Select model for implementation

B.6 Model deployment

1. **Staging**: Deploy to target platform
    - Export model once it meets design requirements and constraints (recording model architecture, exporting weights, library versions, OS version)
    - Containerize the above into Docker/Vagrant for sharing amongst team/ deployment
2. **Pre-production**: Evaluate model on production hardware + software
    - Ensure that models undergo scalability testing based on expected usage
    - Consider future infrastructure and library support, portability of the model
3. **Production**: Deploy product & iterate based on gathered metrics

- Assess deployed product against problem requirements
- Gather metrics to identify gaps in utility

Appendix C

Presentation Slides

A long/short index rebalance portfolio

Engineering Science (Physics) Thesis

Eric Bryce, 9 April 2019

Passive indices are managed

S&P Capital IQ

Capital IQ Key Development Screening Report

Key Developments By Date	Action	Company Name	Key Development Headline
Apr-01-2019	Index Constituent Drops	SNC-Lavalin Group Inc. (TSX:SNC)	SNC-Lavalin Group Inc.(TSX:SNC) dropped from S&P/TSX Canadia Dividend Aristocrats Index
Apr-01-2019	Index Constituent Drops	IDM Mining Ltd. (TSXV:IDM)	IDM Mining Ltd.(TSXV:IDM) dropped from S&P/TSX Venture Composite Index
Mar-25-2019	Index Constituent Adds	Brookfield Property Partners L.P. (NasdaqGS:BPY)	Brookfield Property Partners L.P.(NasdaqGS:BPYP.P) added to NASDAQ Composite Index
Mar-21-2019	Index Constituent Drops	LSC Lithium Corporation (TSXV:LSC)	LSC Lithium Corporation(TSXV:LSC) dropped from S&P/TSX Ventur Composite Index
Mar-19-2019	Index Constituent Drops	Aleafia Health Inc. (TSX:ALEF)	Aleafia Health Inc.(TSXV:ALEF) dropped from S&P/TSX Venture Composite Index
Mar-18-2019	Index Constituent Drops	Xanadu Mines Limited (ASX:XAM)	Xanadu Mines Limited(ASX:XAM) dropped from S&P/ASX All Ordinaries Index
Mar-18-2019	Index Constituent Drops	Cardinal Resources Limited (ASX:CDV)	Cardinal Resources Limited(ASX:CDV) dropped from S&P/ASX Emerging Companies Index
Mar-18-2019	Index Constituent Adds	NorthWest Healthcare Properties Real Estate Investment	NorthWest Healthcare Properties Real Estate Investment

Passive indices are managed, e.g.

SNC-Lavalin Group Inc.(TSX:SNC) dropped from S&P/TSX Canadian Dividend Aristocrats Index

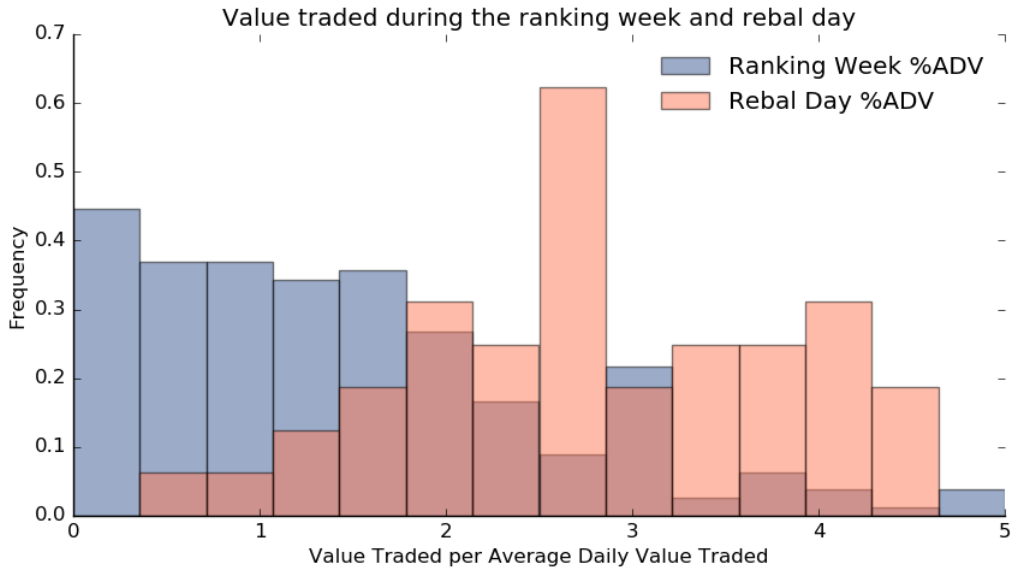
Index changes have market impact, e.g.

- December 2017: Just Energy (JE) dropped from TSX Comp
  - 0.02% of XIC (BMO S&P/TSX Capped Composite Index ETF)
  - Total XIC Mkt Cap: $3.8 billion
- So, 0.02% × $3.8b = $7.7mm to sell

Index changes have market impact, e.g.

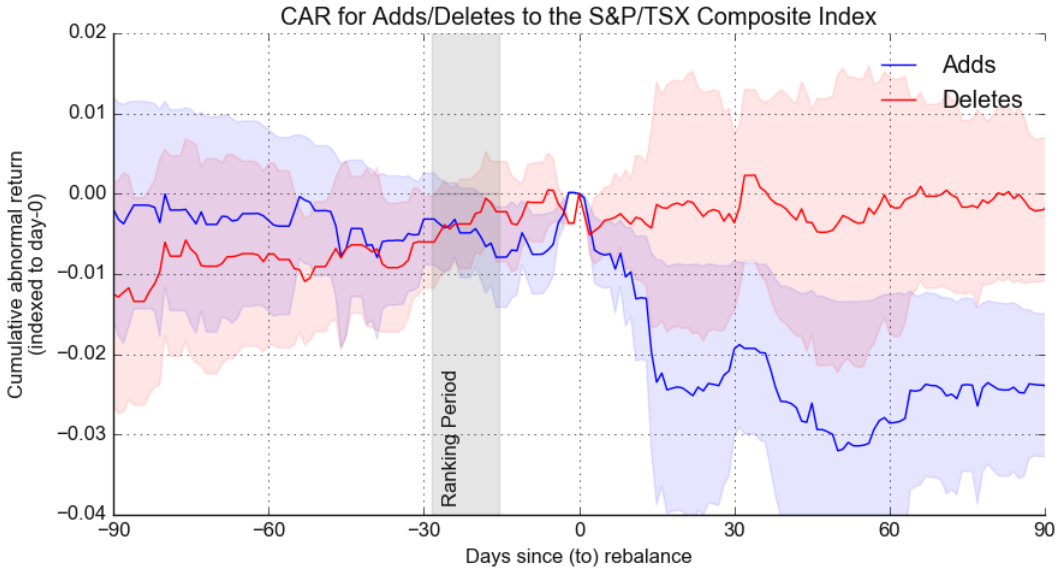
- December 2017: Just Energy (JE) dropped from TSX Comp
  - 0.02% of XIC (BMO S&P/TSX Capped Composite Index ETF)
  - Total XIC Mkt Cap: $3.8 billion
- So, 0.02% × $3.8b = $7.7mm to sell

Index changes have market impact



- Equities trade well above their ADV on rebalance day

Index changes have market impact



- Securities underperform their peer group by ~2.5% after being added to the S&P/TSX Composite Index

7

Composite adds driven by mkt cap

For inclusion in the S&P/TSX Composite Index:

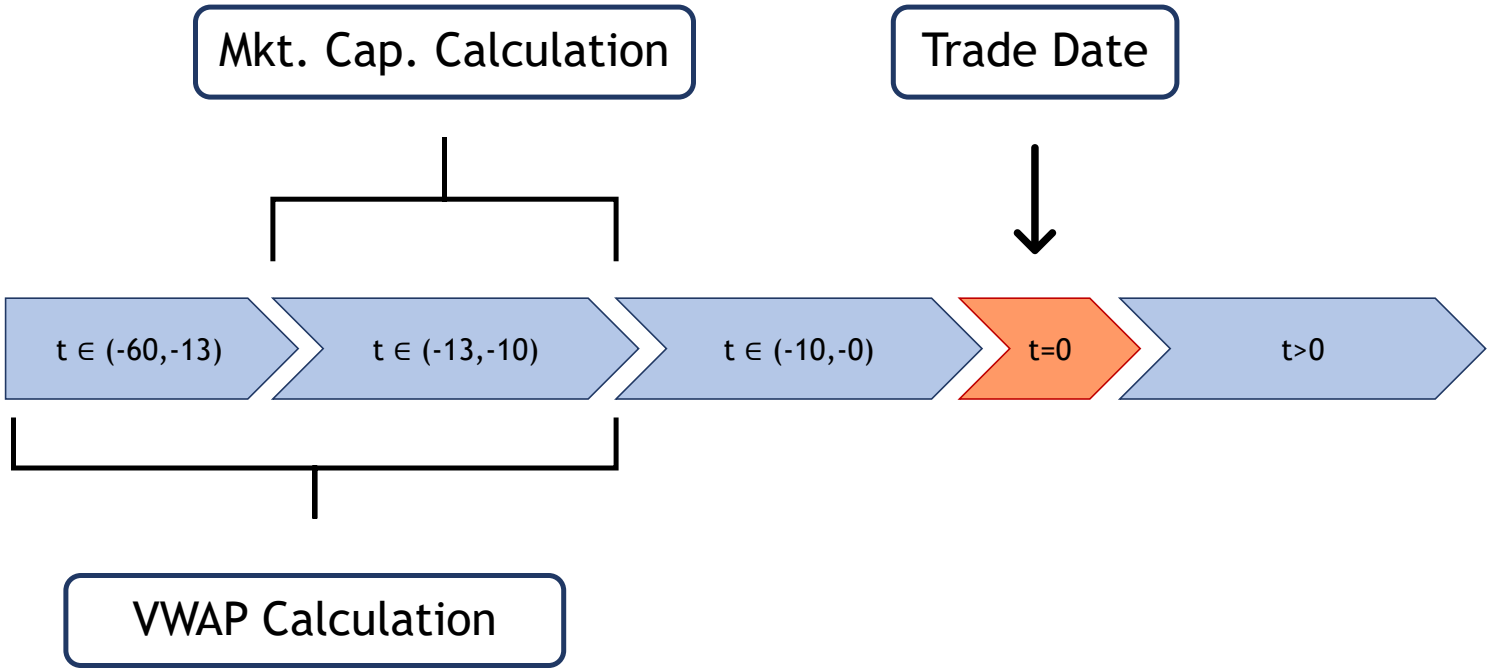
1. **Market capitalisation** at least of 0.04% of the index
   - Approx. $297 mm as of end CY 2019
2. **VWAP** of at least $1
3. **Float turnover** at least 0.5
   - i.e., annual volume traded ≥ ½ float
4. Domiciled in Canada
5. Not an ineligible security
   - Practically, restricted to common shares (voting or non-voting)

Composite adds driven by mkt cap

For inclusion in the S&P/TSX Composite Index:

1. **Market capitalisation** at least of 0.04% of the index
   - Approx. $297 mm as of end CY 2019
2. **VWAP** of at least $1
3. **Float turnover** at least 0.5
   - i.e., annual volume traded ≥ ½ float
4. Domiciled in Canada **not an easily defined attribute!**
5. Not an ineligible security
   - Practically, restricted to common shares (voting or non-voting)

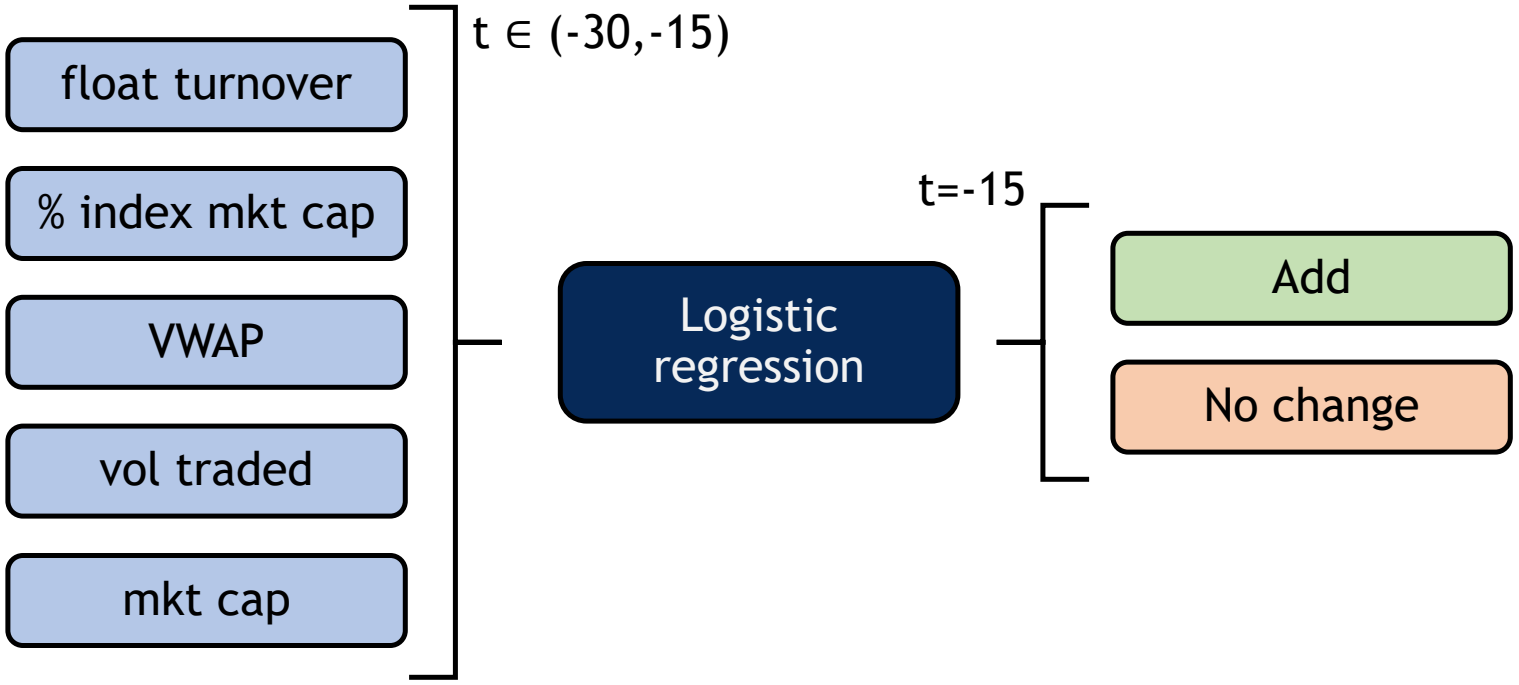
Adds occur on fixed schedule



10

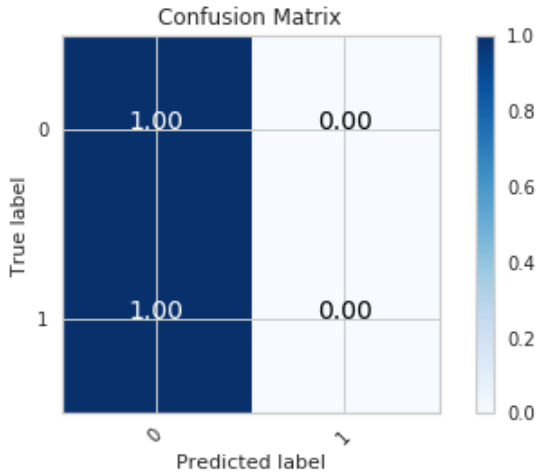
Can we trade this?

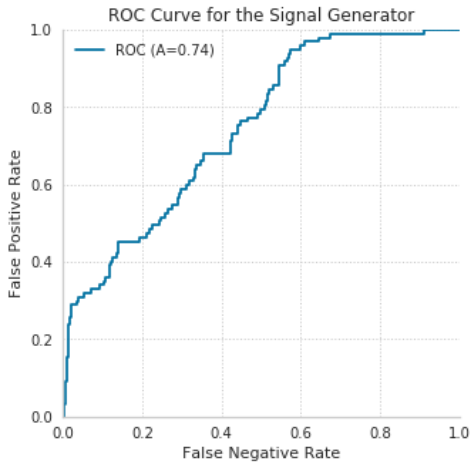
Creating a naïve binary classifier



12

Naïve model performance

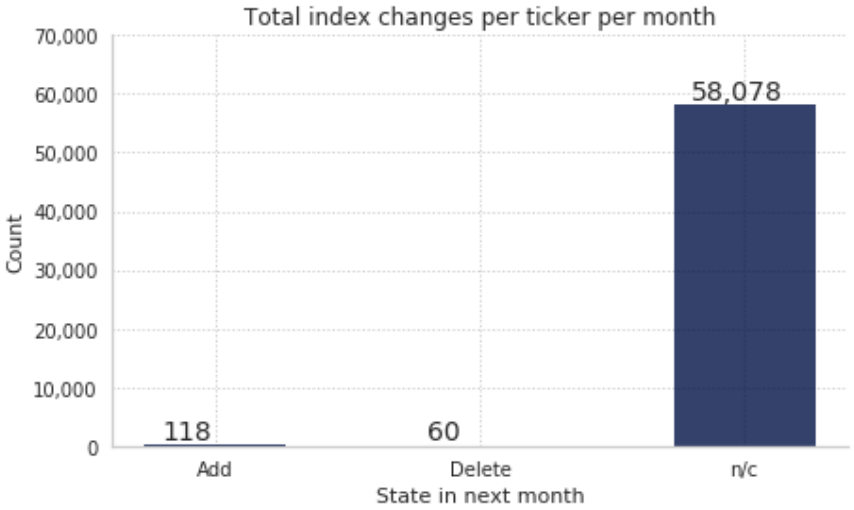




Challenges:

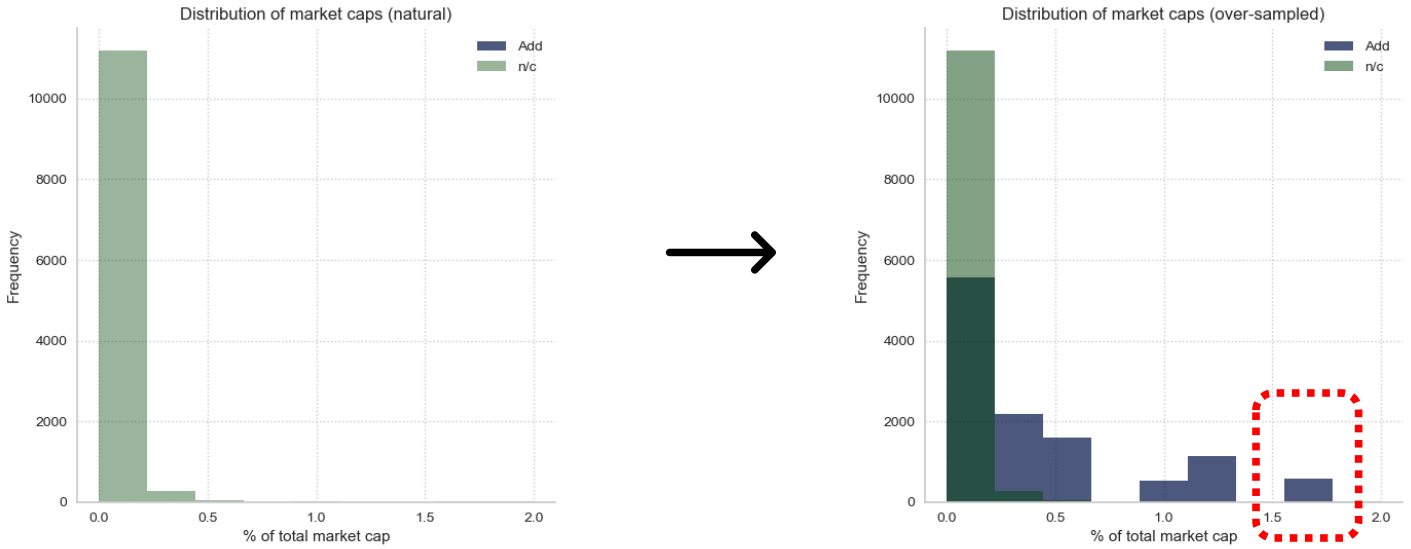
1. Significant class imbalance
2. Non-numeric factors

Class is extremely imbalanced



- On average, the index will not change
  - Model will have ~98% accuracy by predicting no change every time

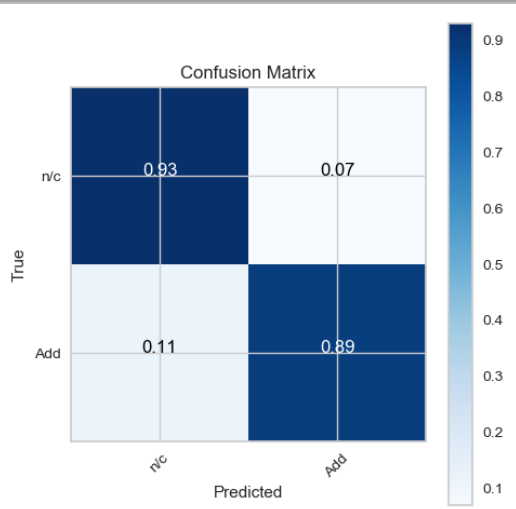
Resolving class imbalances

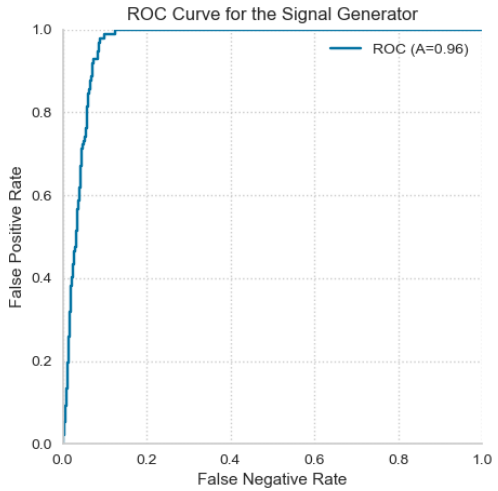


- Random over-sampling over-weights ineligible securities
  - Many positive outliers are not “domiciled” in Canada

15

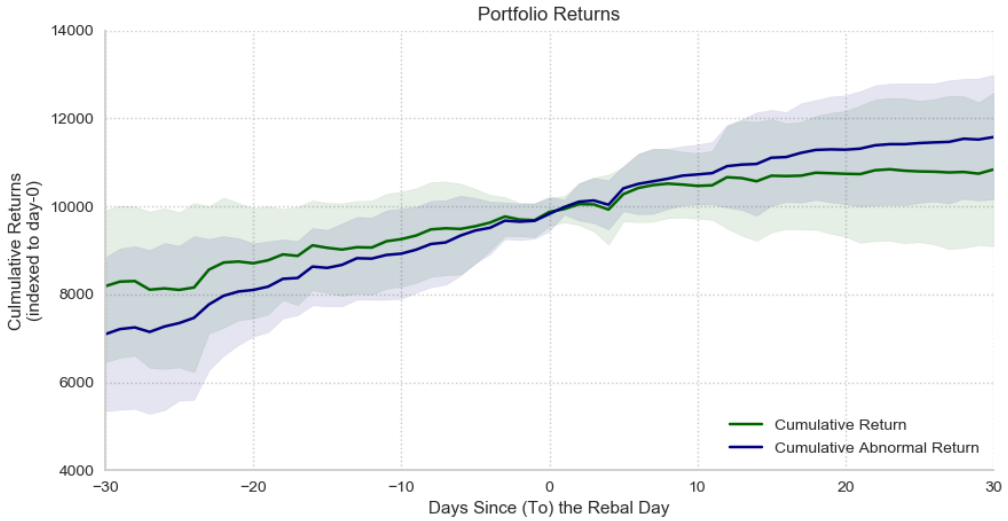
Model accurately predicts index changes





- Binary classifier correctly predicts 89% of adds
- Often predicts adds months ahead of the change
  - Likely due to market cap. calculations

Constructed portfolio performs well



- Hedged "**sell the adds**, buy the industry" strategy entered at the end of the ranking period returns ~16%
  - But, most returns have already been priced-into the security

Limitations

1. Hedging assumption is overly burdensome
   - Could reasonably hedge to the overall index
2. Does not cover deletions
   - Occurs on an irregular schedule
3. Generates relatively few signals
   - Approx. 20 since 2014
   - May be over-fitting in the backtest
4. Misses out on another ~10% potential return
   - Likely brings commensurate increase in risk?

Appendix D

Model

D.1 Factor model

```
# System libraries
import sys, os, gc
import datetime

# Math libraries
import math, random
import pandas as pd, numpy as np
import scipy
from scipy import stats
from datetime import timedelta
from datetime import datetime
import itertools

# Data storage libraries
import pickle, sqlite3, simpledbf, boto3

REFRESH = '--refresh' in [arg.lower() for arg in sys.argv]
BUILD_CHILDREN =  '--refactor-msi' in [arg.lower() for arg in sys.argv]

db = ['C:','Datasets','thesis.db']
overleaf = ['C:','Users','bryce','OneDrive','Documents','Overleaf','Thesis'
    ]

conn = sqlite3.connect('\\'.join(db))
c = conn.cursor()

hdf_path = ['C:','Datasets','ferstenberg.h5']
hdf = pd.HDFStore('\\'.join(hdf_path))
```

```
print('Using SQLite database at %s' % '\\'.join(db))
print('Loaded hdf at %s with keys: %s' % ('\\'.join(hdf_path), ', '.join(
    hdf.keys())))
if REFRESH:
        print(' > Will force full dataset refresh')

sector_query = '''
    SELECT ticker, exchange, share_type,
        sector, industry
    FROM sectors
    WHERE exchange='TSX'
    '''


px_query = '''
    SELECT date, ticker,
        o, h, l, c, vol
    FROM cfmrc
    WHERE ticker IN ('{Tickers}')
    '''

mapping_query = '''SELECT DISTINCT sector, industry FROM sectors'''

df_sector = pd.read_sql(sector_query, conn)

fields = ['industry','sector']

if REFRESH:
        print('Dropping existing tables')
        for field in fields:
                key = '%s_returns' % field
                try:
                        print(' > Dropping {Table}'.format(Table=key))
                        c.execute('DROP TABLE {Table};'.format(Table=key))
                        conn.commit()
                except:
                        print(' > Could not drop {Table}'.format(Table=key)
                            )

for field in fields:
        key = '%s_returns' % field
        if (('/%s' % key) not in hdf.keys()) or REFRESH:
                print('Computing %s returns' % field)
                # SECTOR RETURNS
```

```
all_returns = []
i = 1
for sector in df_sector[field].unique():

        try:

                if sector=='-':
                        raise Exception('Ignoring useless %
                            s: %s' % (field, sector))

                print('%s: %s (%d of %d)' %(field, sector,
                    i, len(df_sector[field].unique())))
                tickers = df_sector.loc[df_sector[field]==
                    sector]['ticker'].unique()
                print(' > %d tickers' % (len(tickers)))

                if len(tickers)==0:
                        raise Exception('Too few tickers;
                            skipping')

                # Get px data from our SQLite instance
                px = pd.read_sql(px_query.format(Tickers="
                    ','".join(tickers)), conn)
                print(' > %d rows of market data returned
                    for %d tickers' % (len(px.index), len(
                    px['ticker'].unique())))

                if len(px.index)<2:
                        raise Exception('Too little market
                            data; skipping')

                # Calculate intraday returns
                px['r_intraday']=(px['c']-px['o'])/px['o']

                # Calculate overnight returns
                returns = px.pivot_table(index=['date'],
                    columns=['ticker'],values=['o','c','
                    r_intraday','vol'])
                for ticker in px['ticker'].unique():
                        returns['r_overnight',ticker] = (
                            returns['o',ticker].shift(1)-
                            returns['c',ticker])/returns['o
                            ',ticker].shift(1)
```

```
# No longer need open and close
returns.drop(['c','o'], inplace=True, axis
    =1)
returns = returns.replace([np.inf, -np.inf
    ], np.nan)

# Compute averages
returns['r_overnight','mean'] = returns.
    iloc[:, (returns.columns.
    get_level_values(0)=='r_overnight') & (
    returns.columns.get_level_values(1).
    isin(tickers))].mean(axis=1)
returns['r_overnight','std'] = returns.iloc
    [:, (returns.columns.get_level_values
    (0)=='r_overnight') & (returns.columns.
    get_level_values(1).isin(tickers))].std
    (axis=1)
returns['r_intraday','mean'] = returns.iloc
    [:, (returns.columns.get_level_values
    (0)=='r_intraday') & (returns.columns.
    get_level_values(1).isin(tickers))].
    mean(axis=1)
returns['r_intraday','std'] = returns.iloc
    [:, (returns.columns.get_level_values
    (0)=='r_intraday') & (returns.columns.
    get_level_values(1).isin(tickers))].std
    (axis=1)
returns['vol','mean'] = returns.iloc[:, (
    returns.columns.get_level_values(0)=='
    vol') & (returns.columns.
    get_level_values(1).isin(tickers))].
    mean(axis=1)
returns['vol','std'] = returns.iloc[:, (
    returns.columns.get_level_values(0)=='
    vol') & (returns.columns.
    get_level_values(1).isin(tickers))].std
    (axis=1)
returns['vol','sum'] = returns.iloc[:, (
    returns.columns.get_level_values(0)=='
    vol') & (returns.columns.
    get_level_values(1).isin(tickers))].sum
    (axis=1)
```

```
                        returns['vol','count'] = returns.iloc[:, (
                            returns.columns.get_level_values(0)=='
                            vol') & (returns.columns.
                            get_level_values(1).isin(tickers))].
                            count(axis=1)

                        # Drop all stock specific data
                        returns = returns.iloc[:, ~(returns.columns
                            .get_level_values(1).isin(tickers))]
                        returns[field] = sector

                        print(returns.tail())

                        all_returns.append(returns)
                        print()
                    except Exception as e:

                        error_msg = ['------',
                                datetime.now().strftime('%Y-%m-%d %
                                    H:%M:%S'),
                                '  ERROR: %s' % e,
                                '  %s'%', '.join(sys.argv),
                                '  %s/%s'%(field, sector),'','']

                        error_msg = '\n'.join(error_msg)
                        print(error_msg)
                        with open('errlog.txt', 'a') as f:
                                f.write(error_msg)
                    i+=1
            print('Done compiling tables; concatenating')
            all_returns = pd.concat(all_returns)
            print(all_returns.head())
            print(all_returns.tail())

            hdf.put(key=key, value=all_returns, format='t', append=
                False)

            all_returns = all_returns.reset_index()

            all_returns.columns = ['date','r_overnight_mean','
                r_overnight_std','r_intraday_mean','r_intraday_s',
                    'vol_mean','vol_std','vol_sum','count',field]
```

```
                all_returns.to_sql(key, conn, index_label=['date',field],
                    if_exists='replace', index=False)
        else:
                print('Skipping %s return calculations' % field)

hdf.close()

if BUILD_CHILDREN and REFRESH:
        print('Fama/French Data')
        import parsers.parse_findata

        stock_specific=True

        print('Computing market-sector-industry returns')

        msi_query = '''
        SELECT date, d.sector, d.industry,--STOCK_SPECIFICm.ticker,
            fr AS riskfree_return,
            mr AS market_return,
            mr-fr AS market_excess_return,
            sr AS sector_return,
            sr-mr AS sector_excess_return,
            ir AS industry_return,
            ir-sr AS industry_excess_return
        FROM(
            SELECT
                i.date, i.industry, s.sector,
                COALESCE(i.r_overnight_mean,0)+COALESCE(i.r_intraday_mean
                    ,0) AS ir,
                COALESCE(s.r_overnight_mean,0)+COALESCE(s.r_intraday_mean
                    ,0) AS sr,
                COALESCE(f.RF,0) as fr, COALESCE(f.Rm,0) AS mr
            FROM industry_returns i
            INNER JOIN (SELECT DISTINCT industry, sector FROM sectors) m
                ON m.industry=i.industry
            LEFT JOIN sector_returns s
                ON s.sector=m.sector AND s.date=i.date
            LEFT JOIN french f
                ON f.date=i.date
            ORDER BY i.date
            ) d
        --STOCK_SPECIFICINNER JOIN (SELECT industry, sector, ticker FROM
            sectors) m
```

```
        --STOCK_SPECIFIC      ON m.industry=d.industry AND m.sector=d.sector
        '''

        #index_label = ['date','ticker']
        if stock_specific:
                msi_query = msi_query.replace('--STOCK_SPECIFIC','')
        #else:
        #       index_label = ['date','sector','industry']

        df = pd.read_sql(msi_query, conn)
        print(df.head())
        df.to_sql('msi', conn, if_exists='replace', index=False)

if REFRESH:
        print('Refreshing daily returns')
        import parsers.parse_daily_returns
else:
        print('Not reloading daily returns')

study_indices = ['S&P/TSX Composite Index']
print('Preparing dataset for regression')
print('Studying only %s' % ', '.join(study_indices))
regression_query = '''
        SELECT
                s.ticker, s.[date],
                fs.[date] AS rebal_date,
                fs.Action AS action,
                fs.[index] AS [index],
                s.r_daily,
                msi.riskfree_return,
                msi.market_excess_return,
                msi.sector_excess_return,
                msi.industry_excess_return
        FROM daily_returns s
        INNER JOIN msi
                ON s.[date]=msi.[date]
                AND s.ticker=msi.ticker
        INNER JOIN factset_index_changes fs
                ON fs.ticker=s.ticker
                AND fs.[index] IN ('{StudyIndex}')
        --WHERE msi.industry='Asset Management and Custody Banks'
        '''.format(StudyIndex="','".join(study_indices))
```

```
print('Querying for price data')
df = pd.read_sql(regression_query, conn)
print('Queried price data')

print(df.head())

if REFRESH:
        print('Calculating correlation coefficients for factors')
        X_cols = df.columns[-4:]
        y_col = 'r_daily'
        label = 'ticker'
        print('Per %s: regresing %s against %s' % (label, y_col, ', '.join(
            X_cols)))
        from sklearn import linear_model
        import warnings
        warnings.simplefilter('ignore')


        regressions = []
        i = 1
        n = len(df[label].unique())
        for ticker in df[label].unique():

                subset = df.loc[df[label]==ticker]

                actions = subset['action'].unique()
                rebal_dates = subset['rebal_date'].unique()
                cutoff_date = rebal_dates.max()
                print('%s: %s on %s (%d of %d), only evaluating pre-%s'% (
                    ticker, '/'.join(actions), '/'.join(rebal_dates), i, n,
                     cutoff_date))
                subset = subset.loc[subset['rebal_date']==cutoff_date]
                action = subset['action'].unique()[0]

                # Data cleaning, remove all strings
                subset[y_col] = subset[y_col].apply(lambda value: np.nan if
                     type(value)!=float else value)
                for col in X_cols:
                        subset[col] = subset[col].apply(lambda value: np.
                            nan if type(value)!=float else value)

                # Remove all extreme values
                subset = subset.replace(np.inf, 1)
```

```
            subset = subset.replace(-np.inf, -1)
            subset = subset.fillna(0)
            subset[y_col] = subset[y_col].apply(lambda value: max(value
                ,-1))
            for col in X_cols:
                    subset[col] = subset[col].apply(lambda value: max(
                        value,-1))

            reg = linear_model.LinearRegression()

            for pre_event in [False, True]:
                    regression = pd.Series(name=ticker)
                    if pre_event:
                            subset = subset.loc[subset['date']<
                                cutoff_date]

                    if len(subset.index)==0:
                            print('  Too little market data for the pre
                                -event %s; skipping' % pre_event)
                            continue

                    reg.fit(subset[X_cols].values, subset[y_col].values
                        )

                    for i in range(len(X_cols)):
                            regression[X_cols[i]] = reg.coef_[i]
                    regression['residual'] = reg.intercept_

                    regression = pd.DataFrame(regression).T
                    regression['pre_event'] = pre_event
                    regression['action'] = action
                    regressions.append(regression)
                    print(regression)
            #break
            i+=1
    regressions = pd.concat(regressions).reset_index()
    regressions.columns = ['ticker','riskfree_return','
        market_excess_return','sector_excess_return','
        industry_excess_return','residual','pre_event','action']

    bounds = [-5,5]
    bounds_rf = [-1,1]
```

```
        for column in ['market_excess_return','sector_excess_return','
            industry_excess_return']:
                regressions[column].clip(upper=max(bounds), lower=min(
                    bounds))
        regressions['riskfree_return'] = regressions['riskfree_return'].
            clip(upper=max(bounds_rf), lower=min(bounds_rf))

        regressions.to_sql('factor_coefficients', conn, if_exists='replace'
            , index=False)
else:
        print('Skipping calculation of correlation coefficients')
        regressions = pd.read_sql('''SELECT * FROM factor_coefficients ''',
            conn)
print(regressions.head())


if REFRESH:
        print('Calculating abnormal returns')
        df['pre_event'] = df['date']<df['rebal_date']

        df = df.merge(regressions, on=['ticker','pre_event','action'],
            suffixes=('','_corr'), how='inner')

        bounds = [-1,10]

        df['er_daily'] = df['riskfree_return']*df['riskfree_return_corr'] +
             df['market_excess_return']*df['market_excess_return_corr'] +
            df['sector_excess_return']*df['sector_excess_return_corr'] + df
            ['industry_excess_return']*df['industry_excess_return_corr']

        df['er_daily'] = df['er_daily'].clip(upper=max(bounds), lower=min(
            bounds))
        df['r_daily'] = df['r_daily'].clip(upper=max(bounds), lower=min(
            bounds))

        df['ar_daily'] = df['r_daily'] - df['er_daily']

        df.to_sql('daily_abnormal_returns', conn, if_exists='replace',
            index=False)

        print(df.head())
else:
        print('Skipping abnormal return calculation')
```

D.2 Signal generation

```
# System libraries
import sys, os, gc
import datetime

# Math libraries
import math, random
import pandas as pd, numpy as np
import scipy
from scipy import stats
from datetime import timedelta
from datetime import datetime
import itertools

# Data storage libraries
import pickle, sqlite3, simpledbf, boto3

# Custom financial data libraries
import utils.findata_utils as fd
import utils.ml_utils as ml_utils

# Plotting libraries
import matplotlib.pyplot as plt
from matplotlib import rcParams

import warnings
#if not sys.warnoptions:
#    warnings.simplefilter("ignore")

from importlib import reload
fd = reload(fd)

import sklearn as sk
import tensorflow as tf
import xgboost as xgb
import keras

from imblearn.over_sampling import RandomOverSampler

from sklearn import svm
from sklearn import preprocessing
from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier
```

```
from sklearn.linear_model import ElasticNet, LogisticRegression
from sklearn.metrics import explained_variance_score, mean_squared_error,
    confusion_matrix, classification_report, accuracy_score
from sklearn.model_selection import cross_val_score, KFold, GridSearchCV
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.pipeline import Pipeline
from sklearn.externals import joblib

from keras.models import Sequential
from keras.optimizers import SGD
from keras.layers import Dense, Dropout
from keras.wrappers.scikit_learn import KerasRegressor

from yellowbrick.regressor import ResidualsPlot, PredictionError

# Connect to databases
db = 'C:\\Datasets\\thesis.db'
overleaf = ['C:','Users','bryce','OneDrive','Documents','Overleaf','Thesis'
    ,'assets','exports']
conn = sqlite3.connect(db)
c = conn.cursor()

hdf_path = 'C:\\Datasets\\thesis.h5'
hdf = pd.HDFStore(hdf_path)

import warnings
if not sys.warnoptions:
        warnings.simplefilter("ignore")

index_mkt_cap = pd.read_sql('''SELECT * FROM index_mkt_cap_bbgmethod ''',
    conn)
index_mkt_cap.columns = ['date','price','pct_day0','SPTSXComp']
#index_mkt_cap['threshold'] = index_mkt_cap['SPTSXComp']*0.0004
#index_mkt_cap['lower'] = index_mkt_cap['threshold']*(1-limit)
#index_mkt_cap['upper'] = index_mkt_cap['threshold']*(1+limit)

index_mkt_cap['datetime'] = index_mkt_cap['date'].apply(lambda date:
    datetime.strptime(str(date)[:10], '%Y-%m-%d'))
index_mkt_cap['month'] = index_mkt_cap['datetime'].apply(lambda date: date.
    strftime('%Y-%m'))
index_mkt_cap['date'] = index_mkt_cap['datetime'].apply(lambda date: date.
    strftime('%Y-%m-%d'))
```

```
index_mkt_cap['is_ranking_week'] = index_mkt_cap['datetime'].apply(lambda
    date: (date+timedelta(days=14)).month!=date.month)

index_mkt_cap.drop(['datetime'],axis=1, inplace=True)

print(index_mkt_cap.head())

avg_mkt_caps = pd.DataFrame(index_mkt_cap.loc[index_mkt_cap['
    is_ranking_week']].groupby(by=['month'])['SPTSXComp'].mean()).
    reset_index()
print(avg_mkt_caps.head())

# MARGINAL SECURITIES ONLY

index_changes = pd.read_sql('''
        SELECT date, action, ticker,
                SUBSTR(date,0,8) AS month
        FROM factset_index_changes
        WHERE [index] IN ('{StudyIndex}')
        '''.format(StudyIndex='S&P/TSX Composite Index'), conn)

index_changes['ranking_month'] = index_changes['month'].apply(lambda month:
     (datetime.strptime(month, '%Y-%m')-timedelta(days=28)).strftime('%Y-%m
    '))
#tickers = index_changes['ticker'].unique()

px_cfmrc = pd.read_sql('''
        SELECT c.date, c.ticker, c.c, c.vol--, s.shares_out, c.c*s.
            shares_out AS mkt_cap
        FROM cfmrc c
        --WHERE mkt_cap>{MinMktCap}*0.75
                --AND c.ticker IN ('{Tickers}')
        '''.format(Tickers='', MinMktCap='')
        , conn)

so = pd.read_sql('''
        SELECT so.ticker, so.shares_out, so.float
        FROM shares_out so
        --WHERE mkt_cap>{MinMktCap}*0.75
        --WHERE c.ticker IN ('{Tickers}')
        '''.format(Tickers='', MinMktCap=''),
                                conn)
```

```
print(index_changes.head())
print(px_cfmrc.head())
print(so.head())

# TICKER CHANGES

ticker_changes = pd.read_sql('''
        SELECT * FROM ticker_changes
        WHERE date>'{cfmrc_date}'
        '''.format(cfmrc_date=px_cfmrc.date.max())
        , conn)[['from','to']]

#print(ticker_changes.head())

for i, change in ticker_changes.iterrows():
        px_cfmrc['ticker'].replace(change['from'],change['to'], inplace=
            True)

#print(px_cfmrc.head())

px = px_cfmrc.merge(so, on=['ticker'], how='inner')

# Deal with nonsense
px['c'] = px['c'].apply(lambda c: float(c))
px['shares_out'] = px['shares_out'].replace(' ',0)
px['shares_out'] = px['shares_out'].apply(lambda so: float(so))

#print(px.head())
px['mkt_cap'] = px['shares_out']*px['c']
print(px.head())

#SECURITY LIST
# Do not look at securities already in the index
indexed = pd.read_sql('SELECT * FROM in_the_comp', conn).set_index(['date'
    ]).unstack().reset_index()
indexed.columns = ['ticker','date','indexed']
indexed['indexed'] = indexed['indexed'].replace(1,True).replace(0,False)
indexed['month'] = indexed['date'].apply(lambda date: date[:7])

# Aggregated indexing per month
#indexed = indexed.merge(index_changes, on=['ticker','month'])#
#indexed = indexed.drop(['date_x','date_y'], axis=1)
indexed = indexed.drop_duplicates().groupby(by=['ticker','month'])
```

```
indexed = indexed['indexed'].max().reset_index()#.drop(['action'], axis=1)

print(indexed.head())

sample = px.copy(deep=True)
sample['datetime'] = sample['date'].apply(lambda date: datetime.strptime(
    str(date)[:10], '%Y-%m-%d'))
sample['is_ranking_week'] = sample['datetime'].apply(lambda date: (date+
    timedelta(days=14)).month!=date.month)
sample['month'] = sample['date'].apply(lambda date: str(date)[:7])

# Only look at the ranking week
sample = sample.loc[sample['is_ranking_week']]

sample = sample.merge(index_mkt_cap, on=['date'], how='inner', suffixes=('
    ','_idc'))
sample.drop(['datetime','price','pct_day0','month_idc','is_ranking_week_idc
    '], axis=1, inplace=True)

print(sample.head())

sample['mkt_cap_X_vol'] = sample['mkt_cap']*sample['vol']
sample['c_X_vol'] = sample['c']*sample['vol']

grouped = sample.groupby(by=['ticker','month'])[['vol','mkt_cap_X_vol', '
    c_X_vol']].sum().reset_index()
grouped['VWMC'] = grouped['mkt_cap_X_vol']/grouped['vol']
grouped['VWAP'] = grouped['c_X_vol']/grouped['vol']
grouped.drop(['mkt_cap_X_vol','c_X_vol'], axis=1, inplace=True)
print(grouped.head())

# Merge to list of changes
grouped = grouped.merge(index_changes, left_on=['month','ticker'],
                                        right_on=['ranking_month','ticker
                                            '],
                                        how='outer', suffixes=('','_chg')
                                            )

# Merge to list of indexed securities
grouped = grouped.merge(indexed, left_on=['ticker','month'],
                                       right_on=['ticker','month'],
                                       how='outer', suffixes=('','_idx'))
```

```
grouped = grouped.merge(avg_mkt_caps, on=['month'], how='left')

grouped = grouped.merge(so[['ticker','float']], on=['ticker'], how='inner')
grouped['turnover'] = grouped['vol']/grouped['float']
grouped['turnover'] = grouped['turnover'].replace([np.inf, -np.inf], np.nan
    )
grouped['turnover'].fillna(grouped['turnover'].mean(), inplace=True)

#sample['indexed']=sample['indexed'].fillna(sample['action']=='Delete')
grouped['indexed'].fillna(False, inplace=True)
grouped['pct_contrib'] = grouped['VWMC']/(grouped['VWMC']+grouped['
    SPTSXComp'])

grouped.drop(['date','month_chg','ranking_month'],axis=1,inplace=True)

print(grouped.head())

# AGG PER RANKING PERIOD

# Apply restrictions
filtered = grouped.loc[grouped['month']>'2010-01']
filtered = filtered.loc[(~filtered['indexed']) | ~(filtered['action'].isna
    ())]
filtered = filtered.loc[filtered['VWAP']>0.5]
print(filtered.head())

# TRAIN THE MODEL

filtered['add']=filtered['action']=='Add'
filtered['del']=filtered['action']=='Delete'

from imblearn.over_sampling import SMOTE

SEED=0xDEADBEEF

y_col = 'add'
X_cols = ['pct_contrib','turnover','VWAP','vol','VWMC','SPTSXComp']
X = filtered[X_cols]
y = filtered[y_col]

X_test, X_train, y_test, y_train  = sk.model_selection.train_test_split(X.
    values, y.values, test_size=0.2, random_state=SEED)
```

```
#oversampler = SMOTE(random_state=SEED)
oversampler = RandomOverSampler(random_state=SEED)

X_train_resample, y_train_resamle = oversampler.fit_resample(X_train,
    y_train)

print(len(X_train), len(X_test))

log_clf = LogisticRegression()#xgb.XGBClassifier(max_depth=4,
    min_child_weight=50, learning_rate=0.01, n_estimators=50, gamma=1)
 #RandomForestClassifier()#LogisticRegression()
log_clf.fit(X_train_resample, y_train_resamle)

print(log_clf.score(X_train, y_train))

y_pred = log_clf.predict(X_test)
y_pred_prob = log_clf.predict_proba(X_test)
ml_utils.clf_model_eval(y_test, y_pred, classes=['n/c','Add'])
#plt.show()
#plt.close()

from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(5,5))

if len(X_cols)>1:
        ax = fig.add_subplot(111, projection='3d')
        ax.scatter(X.values[:,0],X.values[:,1],y.values)
        ax.set_zlabel(y_col)
else:
        ax = fig.add_subplot(111)

        ax.spines['left'].set_visible(True)
        ax.spines['top'].set_visible(False)
        ax.spines['right'].set_visible(False)
        ax.spines['bottom'].set_visible(True)
        ax.grid(True,axis='both',linestyle=':')

        ax.scatter(X.values, y.values)
        ax.set_xlabel(X_cols[0])
        ax.set_ylabel(y_col)
plt.show()
fig.savefig('\\'.join(overleaf+['scatterplot.png']))
```

```
plt.close()

#from sklearn.metrics import roc_curve, auc
import sklearn.metrics as metrics

fpr, tpr, threshold = metrics.roc_curve(y_test, y_pred_prob[:,1])
roc_auc = metrics.auc(fpr, tpr)

# Plot the ROC curve
fig = plt.figure(figsize=(5,5))
fig.patch.set_facecolor('white')
ax = fig.add_subplot(1, 1, 1)
ax.spines['left'].set_visible(True)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(True)
ax.grid(True,axis='both',linestyle=':')

ax.plot(fpr, tpr, label='ROC (A=%0.2f)' % roc_auc)
plt.legend(frameon=False, loc='best')

plt.title('ROC Curve for the Signal Generator')
plt.ylabel('False Positive Rate')
plt.xlabel('False Negative Rate')
plt.xlim(0,1)
plt.ylim(0,1)
plt.show()
fig.savefig('\\'.join(overleaf+['roc_curve.png']))
plt.close()

# PLOT sample sizes
font = {'family' : 'Arial',
        'weight' : 'normal',
        'size'   : 12}

plt.rc('font', **font)

fig = plt.figure(figsize=(6,6))
fig.patch.set_facecolor('white')
ax = fig.add_subplot(1, 1, 1)
ax.spines['left'].set_visible(True)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
```

```
ax.spines['bottom'].set_visible(True)
ax.grid(True,axis='both',linestyle=':')

ax.hist(X_train[:,0][y_train], density=False, bins=np.linspace(0,0.02,10),
    alpha=0.7, histtype='stepfilled', label='Add', color=['xkcd:navy blue'
    ])
ax.hist(X_train[:,0][~y_train], density=False, bins=np.linspace(0,0.02,10),
     alpha=0.4, histtype='stepfilled', label='n/c', color=['xkcd:forest
    green'])
#plt.hist(y_train)
plt.title('Distribution of market caps (natural)')
plt.ylabel('Frequency')
plt.xlabel('% of total market cap')
plt.legend(frameon=False, loc='top right')
plt.xticks(np.linspace(0,0.02,5), np.linspace(0,2,5))
fig.savefig('\\'.join(overleaf+['distribution_natural.png']))
plt.show()

fig = plt.figure(figsize=(6,6))
fig.patch.set_facecolor('white')
ax = fig.add_subplot(1, 1, 1)
ax.spines['left'].set_visible(True)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(True)
ax.grid(True,axis='both',linestyle=':')

ax.hist(X_train_resample[:,0][y_train_resamle], density=False, bins=np.
    linspace(0,0.02,10), alpha=0.7, histtype='stepfilled', label='Add',
    color=['xkcd:navy blue'])
ax.hist(X_train_resample[:,0][~y_train_resamle], density=False, bins=np.
    linspace(0,0.02,10), alpha=0.5, histtype='stepfilled', label='n/c',
    color=['xkcd:forest green'])
#plt.hist(y_train)
plt.title('Distribution of market caps (over-sampled)')
plt.ylabel('Frequency')
plt.xlabel('% of total market cap')
plt.legend(frameon=False, loc='top right')
plt.xticks(np.linspace(0,0.02,5), np.linspace(0,2,5))
fig.savefig('\\'.join(overleaf+['distribution_oversample.png']))
plt.show()
```

```
# CONSTRUCT A PORTFOLIO

signals = filtered.copy(deep=True)
try:
        signals['probability'] = log_clf.predict_proba(X.values)[:,1]
except:
        print('Using flat probability distn')
        signals['probability'] = 1
signals['prediction'] = log_clf.predict(X.values)

signals = signals.loc[signals['prediction']]

signals['trade_date'] = signals['month'].apply(lambda month: (datetime.
    strptime(month, '%Y-%m')+timedelta(days=31)).replace(day=1))
signals = signals[['ticker','trade_date','probability']]

signals.to_sql('signals', conn, if_exists='replace', index=False)

import backtest
```

D.3 Portfolio construction

```
# System libraries
import sys, os, gc
import datetime

# Math libraries
import math, random
import pandas as pd, numpy as np
import scipy
from scipy import stats
from datetime import timedelta
from datetime import datetime
import itertools

# Data storage libraries
import pickle, sqlite3, simpledbf, boto3

# Custom financial data libraries
import utils.findata_utils as fd
import utils.ml_utils as ml_utils
```

```
# Plotting libraries
import matplotlib.pyplot as plt
from matplotlib import rcParams

import warnings
#if not sys.warnoptions:
#    warnings.simplefilter("ignore")

from importlib import reload
fd = reload(fd)

import sklearn as sk
#import tensorflow as tf
import xgboost as xgb
#import keras

from imblearn.over_sampling import RandomOverSampler

from sklearn import svm
from sklearn import preprocessing
from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier
from sklearn.linear_model import ElasticNet, LogisticRegression
from sklearn.metrics import explained_variance_score, mean_squared_error,
    confusion_matrix, classification_report, accuracy_score
from sklearn.model_selection import cross_val_score, KFold, GridSearchCV
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.pipeline import Pipeline
from sklearn.externals import joblib

#from keras.models import Sequential
#from keras.optimizers import SGD
#from keras.layers import Dense, Dropout
#from keras.wrappers.scikit_learn import KerasRegressor

#from yellowbrick.regressor import ResidualsPlot, PredictionError

# Connect to databases
db = 'C:\\Datasets\\thesis.db'
overleaf = ['C:','Users','bryce','OneDrive','Documents','Overleaf','Thesis'
    ,'assets','exports']
conn = sqlite3.connect(db)
c = conn.cursor()
```

```
hdf_path = 'C:\\Datasets\\thesis.h5'
hdf = pd.HDFStore(hdf_path)

import warnings
if not sys.warnoptions:
        warnings.simplefilter("ignore")

holding_period = 30
look_back = 30
short = True

signals = pd.read_sql('''
        SELECT * FROM signals
        ''',
        conn)

signals['trade_date'] = pd.to_datetime(signals['trade_date'])

returns = pd.read_sql('''
        SELECT * FROM daily_abnormal_returns
        WHERE ticker in ('{Tickers}')
        '''.format(Tickers="','".join(signals['ticker'].unique())),
        conn)

returns['datetime'] = returns['date'].apply(lambda date: datetime.strptime(
    date, '%Y-%m-%d'))


port_return = []
skipped_signals = []


action = 'Sell' if short else 'Buy'

try:
        for i, signal in signals.iterrows():

                print('%s %s (p=%f)' % (action, signal['ticker'], signal['
                    probability']))

                look_back_to = signal['trade_date']-timedelta(days=
                    look_back)
```

```
# Choose the day to open the position on
#  if the first day of the month is a weekend, find the
    next monday
open_position_on = signal['trade_date']
while open_position_on.strftime('%Y-%m-%d') not in returns[
    'date'].values:
        open_position_on = open_position_on+timedelta(days
            =1)
print(' > Open position on %s'%open_position_on.strftime('%
    Y-%m-%d'))

close_position_on = signal['trade_date']+timedelta(days=
    holding_period)
ss_return = returns.loc[(returns['ticker']==signal['ticker'
    ]) & (returns['datetime'].between(look_back_to,
    close_position_on))]

# Choose the most appropriate abnormal return calculation
rebal_dates = returns.loc[returns['ticker']==signal['ticker
    ']]['rebal_date'].unique()
try:
        rebal_date_to_use = datetime.strptime(max(
            rebal_dates)[:10],'%Y-%m-%d')
except:
        print(' > No abnormal return data for %s; skipping'
             % signal['ticker'])
        signal['reason'] = 'No factor model data for this
            security'
        skipped_signals.append(pd.DataFrame(signal).T)
        continue
for possible_date in rebal_dates:
        #print(possible_date)
        if signal['trade_date'] < datetime.strptime(
            possible_date[:10],'%Y-%m-%d'):
                rebal_date_to_use = possible_date
ss_return = ss_return.loc[ss_return['rebal_date']==
    rebal_date_to_use]

print(' > Using CAR calculations for %s' % str(
    rebal_date_to_use)[:10])

ss_return['d'] = pd.to_datetime(ss_return['date']) - signal
    ['trade_date']
```

```
ss_return['culm_return'] = (ss_return['r_daily']+1).cumprod
    ()
ss_return['car'] = (ss_return['ar_daily']+1).cumprod()

try:
        return_index = ss_return.loc[ss_return['datetime'
            ]==open_position_on].iloc[0]
except:
        if str(open_position_on)[:10] < returns.loc[(
            returns['ticker']==ticker)]['date'].min():
                signal['reason'] = 'No market history to
                    this date'
                print(' > Do not have market history to
                    this date; skipping')
                skipped_signals.append(pd.DataFrame(signal)
                    .T)
                continue
        else:
                raise(' > Something went wrong')
                signal['reason'] = 'Unknown error'
                skipped_signals.append(pd.DataFrame(signal)
                    .T)
                #break
                continue

for field in ['culm_return','car']:
        ss_return[field] = ss_return[field] - return_index[
            field]

for field in ['culm_return','car']:
        ss_return[field] = ss_return[field]*-1 if short
            else ss_return[field]
        ss_return[field]=(ss_return[field]+1)*10000

ss_return = ss_return[['ticker','date','d','culm_return', '
    car']]

ss_return['d'] = ss_return['d'].apply(lambda d: d.days)

ss_return['trade'] = i
ss_return['probability'] = signal['probability']
port_return.append(ss_return)
```

```
except Exception as e:
        print(ss_return)
        print(signal)
        print()

port_return = pd.concat(port_return)
try:
        skipped_signals = pd.concat(skipped_signals)
        print('Skipped signals:')
        print(skipped_signals)
except:
        pass


#probabilities = port_return['probability']
pivot = port_return.pivot_table(index=['d'], columns=['trade'], values=['
    culm_return','car','probability'], aggfunc=np.mean)
pivot.fillna(method='ffill', inplace=True)
pivot.fillna(method='bfill', inplace=True)

for trade in pivot.columns.get_level_values(1):
        pivot['weight',trade] = pivot['probability',trade]/pivot['
            probability'].sum(axis=1)


for metric in ['culm_return','car']:
        pivot[metric,'portfolio'] = (pivot[metric]*pivot['weight']).sum(
            axis=1)
        pivot[metric,'portfolio_std'] = pivot[metric].std(axis=1)
print(pivot)


# Plot the results
font = {'family' : 'Arial',
        'weight' : 'normal',
        'size'   : 12}

plt.rc('font', **font)

fig = plt.figure(figsize=(10,5))
fig.patch.set_facecolor('white')
```

```
ax = fig.add_subplot(1, 1, 1)

ax.spines['left'].set_visible(True)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(True)
ax.grid(True,axis='both',linestyle=':')

colors = {'car':'navy',
        'culm_return':'darkgreen'}

nice_names = {'car':'Cumulative Abnormal Return',
        'culm_return':'Cumulative Return'}

for metric in ['culm_return','car']:
        ax.plot(pivot.index, pivot[metric,'portfolio'], label=nice_names[
           metric], color=colors[metric])
        ax.fill_between(pivot.index,
                pivot[metric,'portfolio']-pivot[metric,'portfolio_std'],
                pivot[metric,'portfolio']+pivot[metric,'portfolio_std'],
                color=colors[metric],
                alpha=0.1)

ax.fill_between([4000,14000], [1,1], color='grey', alpha=0.1)
ax.fill_between([4000,14000], [-1,-1], color='grey', alpha=0.1)

#ax.xticks(np.linspace(-30,30, ))

plt.legend(frameon=False, loc='lower right')

plt.title('Portfolio Returns')
plt.ylabel('Culmulative Returns\n(indexed to day-0)')
plt.xlabel('Days Since (To) the Rebal Day')
plt.xlim(-look_back, holding_period)
plt.ylim(4000,14000)
plt.show()
fig.savefig('\\'.join(overleaf+['portfolio_backtest.png']))
plt.close()
```