MACHINE LEARNING IN THE WILD: SEARCHING FOR SIGNAL IN THE NOISE

Emma Muhleman (emm10005), Keshav Pant (kp2749), Sudharsan Asaithambi (sa6149), Gurneet Bedi (gb2388)

New York University, Center for Data Science (CDS)

December 3, 2020

ABSTRACT

We leverage our team's diverse skill set in combination with an iterative approach and emphasis on bakeoff style testing with minimal to no assumptions to develop a supervised machine learning model that outperforms random prediction with respect to the class expectation for the the 10-day forward returns on the NASDAQ Composite Index (QQQ). We integrate potential predictive features from the realm of technical analysis, volatility modeling, factor exposures, macroeconomic signals, and foreign exchange cross-rates to develop a binary classification algorithm designed to forecast our target variable and place it into one of two categories, Outperform (QQQ must rise at least ten basis points from the start of the 10-day period, from t=0 to t=10) and Underperform otherwise.

This paper seeks to evaluate the hypothesis that financial market prices follow a random walk, with little (if any) persistent signal. While cross-checking with an industry practitioner, we emphasize a purely data-driven (assumption-free) approach and iteratively test thousands of model-hyperparameter permutations to test our hypothesis.

We postulate that any signal that exists in intraday data or over the range of a couple of days degenerates rapidly, crowded out by chaos and complex interconnection of observable features. As such, our goal is to generate a model which produces an AUC significantly higher than 0.5. If successful, this model could be integrated into an algorithmic trading strategy.

BUSINESS UNDERSTANDING [1]

The Big Data and ML Revolution: Advancements in cloud computing technologies, virtualization, 5-G, robotic process automation, and global connectivity coupled with the rapid transition to the provision of services in the virtual world as social-distancing has taken over has only accelerated the so-called "digital revolution" and the impact it will have on society, individual behaviors, customer preferences, organizations, data collection and analytical tools, and so on. The transition itself is nothing new. In fact, it has been underway for years. In the search for uncorrelated investment products and new sources of "alpha" (excess risk-adjusted return relative to a benchmark asset), fund managers are increasingly turning to the vast world of big data and quantitative investment strategies.

Market Evolution and the Outgrowth of Passive Investment Products: Since the turn of the century, the investment management industry has been plagued by one

[1] All business understanding and domain expertise provided by Emma Muhleman (emm10005), CFA, CPA, Junior PM, Equity L/S & Global Macro Strategist at Ascend Investment Management LLC . Her expertise was derived directly from her 10+ years of experience with financial markets

series of problems after the next. The 2007/8 Global Financial Crisis (GFC 1.0) marked the beginning of a massive transformation in the structure of global financial markets. The Federal Reserve's "temporary" adventures down the Quantitative Easing rabbit hole was anything but temporary[2]. Likewise, the Fed's suppression of interest rates during the same period was not intended to continue in perpetuity, as it were, yet here we are in 2020 with the Fed pegging rates to 0% for the "*foreseeable future.*"[3] Why is this relevant? It has dramatically changed the structure of the global financial marketplace, leading us into a world awash in debt (across state and private sectors) precisely as aging populations throughout Europe, China, the United States/Canada, and Latin America (in that order) reach the retirement age. Baby boomers were the first generation to receive the tax-free treatment on savings allotted for retirement via automatic withdrawal from their paychecks, and most used the 401(k) plan as a tax efficient means to save for their eventual retirement. Index funds, ETFs, Target Date funds, and the Vanguards and Blackrocks of the world did not exist during the baby boomers coming of age and as such, most (80%) of baby boomers assets invested in markets today are held in actively managed accounts, with the other 20% in some form or index fund or ETF. This is in stark contrast with the millennials, for whom 95% of assets are held in passive products, the majority of which mimic some market-cap weighted index (a large share of this, given the millennials investment horizon, is likely allocated to "growth equity" strategies and included in the NASDAQ through its ETF that trades by the ticker QQQ).[4]

Here's the problem with this scenario: ETFs and Index Funds managed by Vanguard or BlackRock could make the argument that despite their simple decision rule (take every dollar of net inflow and buy the market-cap weighted NASDAQ, irrespective of price; likewise, for every net dollar of withdrawal requests, sell the same cap-weighted proportion of the QQQ), they pose(d) no significant risks neither caused harm to the functioning of the capital markets. And sure, that argument was acceptable as long as these funds only accounted for a collective 20-30% of average daily trading volumes.

Today, passive products receive over 100% of the net marginal dollar that flows into the U.S. equity markets[5]. Beyond the perverse capital inflow-outflow dynamic that will lead to a collapse in the entire passive complex if and when this reaches its logical conclusion, there are other, more disturbing self-reinforcing feedback loops driven by the rise of the passive trend coupled with the momentum in favor of the NASDAQ. When this scheme eventually breaks down as a logical consequence of imbalanced buyers and sellers, as a hedge fund manager you're forced to reckon with the facts: (1) either you rest on your laurels and allow the process to play its course, slowly losing clients until you are jobless in ten years, or (2)

---

2 https://www.cnbc.com/2017/11/24/the-fed-launched-qe-nine-years-ago--these-four-charts-show-its-impact.html#:~:text=The%20Fed%20launched%20quantitative%20easing,%24900%20billion%20to%20%244.5%20trillion.

3 https://www.hsh.com/finance/mortgage/latest-move-by-the-federal-reserve.html

4 https://www.wsj.com/articles/index-funds-are-the-new-kings-of-wall-street-11568799004

5 https://www.bloomberg.com/news/articles/2019-09-11/passive-u-s-equity-funds-eclipse-active-in-epic-industry-shift

it's time to get familiar with the opportunities of the new age investing model, one increasingly aided by machine learning techniques and artificial intelligence, to build better solutions for the millennials and generations to come.

There are a variety of different ways machine learning can help address the plight of the active manager. Due to time constraints and our currently limited access to computational resources, for practical purposes we will avoid more technologically demanding procedures such as acquiring and analyzing satellite imagery to predict the direction of corn and soybean futures. Instead, we will attempt to evaluate a commonly cited hypothesis among investment managers who suggest that attempting to apply machine learning methodologies to training and fitting a model to noisy financial market data will be an exercise in futility[6].

Our null hypothesis is that the markets are in fact composed of mostly noise, with very little signal, thereby placing them on the intractable or nearly intractable end of the predictability spectrum, especially if one seeks to build an interpretable predictive machine.

DATA UNDERSTANDING

Our data preparation and feature selection process was guided by our domain expert's experience with financial markets and existing strategies within the space[7], yet not dominated by existing assumptions in the space. Our strategy was to identify as many *potential* features as possible, then narrow them down using data mining processes.

The data we used is based primarily on the Invesco QQQ Trust historical dataset, downloadable from Yahoo! Finance, which contains daily open, close, high, and low-price data and trading volume from 1998 to the present day. Our analysis aims to be long-term, so we only include data until December 31st, 2019, to avoid unconventional market behavior during the 2020 COVID-19 pandemic. We then split our data into training and holdout sets. As we are working with time-series data, we cannot split randomly without running into lookahead bias (using future data to predict the past). We therefore selected the most recent year of data (2019) as our final holdout set, and used the rest for training.

Technical indicators: These were a variety of alpha-generating factors derived directly from daily QQQ ticker data, using a variety of functions from the TA-Lib package[8]. They included Average True Range (ATR) - a measure of market volatility using the absolute range of an asset's price during a rolling window, Moving Average Convergence Divergence (MACD) - a momentum based indicator tracking the difference between a long and short-term moving average, Momentum (MOM) - the rate at which price changes, and Bollinger Bands (BB) - the interval created by taking two standard deviations from a rolling average.

Nasdaq top stocks: The Nasdaq is heavily dominated by six large technology companies: Apple (AAPL), Amazon (AMZN), Facebook (FB), Google (GOOG), Microsoft (MSFT) and Tesla (TSLA). We elected to include price data on these

[6] https://zacharydavid.com/2017/08/06/fitting-to-noise-or-nothing-at-all-machine-learning-in-markets/

[7] Ideas for some features, particularly technical indicators, were taken from "Machine Learning for Algorithmic Trading"

[8] https://ta-lib.org/

stocks because movements in their prices would likely correlate with movements in QQQ. Historical price data for these or each of these tickers is also downloadable from Yahoo! Finance; we calculated daily returns and constructed multiple simple rolling averages of those returns (5-day, 10-day, 14-day 21-day and 30-day).

Macroeconomic indicators: We also included price data on macroeconomic tickers which could be indicative of the overall economy and thus provide predictive power on QQQ, including: the Bloomberg Barclays High-Yield Bond ETF (JNK), the U.S. Dollar Index (DXY), the Chicago Board Options Exchange Volatility Index (VIX), and 10-Year Treasury Yields (^TNX)

QQQ Past Returns: An important set of potential indicators of QQQ movement are past returns of QQQ itself. As these returns are highly correlated, we opted to use the geometric mean instead of a simple arithmetic mean when calculating return windows[9]. It is important to note that in order for there to be no leakage, all return feature data was from prior to the date in question.

Target Variables*:* Our analysis is interested in predicting whether or not the QQQ would exceed a certain threshold in a certain window of time. The specifics of that window and threshold are flexible, so long as they are feasible from a trading standpoint. We therefore elected to set up multiple potential target variables, and include them in the feature selection process, in order to find a window of time and binary threshold that is best predicted. In order to generate these variables, we created forward returns by shifting past returns forward (for example, the 5-day return from day 0 to day 5 was shifted back to day 0 to act as a 5-day forward return). Then these returns were binarized depending on whether they crossed a specified threshold.

After the feature engineering phase we had 4986 training data rows, 252 holdout rows, and 141 potential features to predict one of 24 potential target variable window-threshold combinations.

Missing values handling: We pulled ticker data on the full range of available historical data for each stock, but those ranges varied significantly. As a result we came across several instances where we had missing values in our data, specifically if a stock was not traded during our years of interest. All of the missing values were returns (of various stocks), so we imputed the median values in order to not overly affect the distribution. However, a few stocks had such a small time frame (TSLA, FB) that we would have had to impute more than half the data points, as a result we elected to drop them altogether.

DATA PREPARATION & FEATURE SELECTION

Given our dataset's size and the list of potential features, feature selection was a crucial and complex component of our analysis. In our initial exploratory analysis of features, we could not find a single feature with a mutual information score higher than 0.02, and many features had a score of 0. Additionally, many of our features are rolling averages drawn from the same price data source, meaning they are positively correlated. Even features drawn from different sources can be with one another (for example, the price of gold (GLD) and the

[9] https://www.investopedia.com/ask/answers/06/geometricmean.asp

relative strength of the U.S. dollar (DXY) are often negatively correlated). A heatmap of feature correlations is shown below:

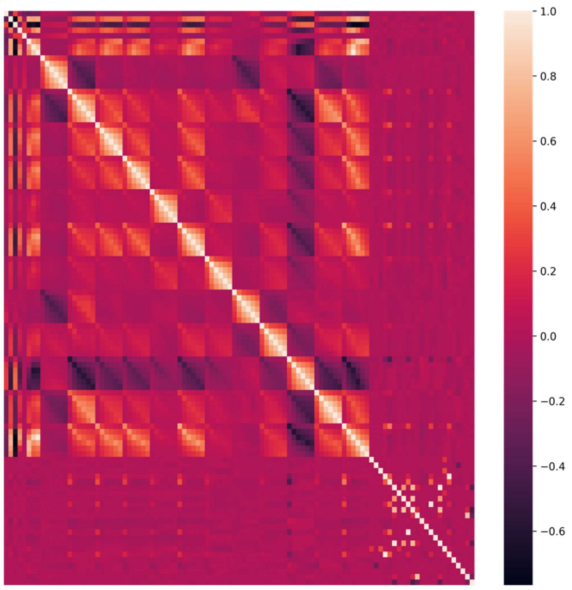

Figure 1. Correlation heatmap for potential features

While we considered dimensionality reduction strategies like singular value decomposition and principal component analysis to condense our features effectively, we believe our model's interpretability will be important during any deployment to allow for sanity checking during trading processes. Our feature selection strategy was therefore broken down into three phases:

Target Variable Selection: As mentioned above, the goal of our analysis was to predict whether returns would cross a certain threshold within a certain window of time, and we decided to vary both threshold and window in order to help us find the most predictable combination (which could be successfully integrated into a trading strategy). As well as predictive power, we wanted to ensure that the binary classes generated by the threshold were relatively balanced, in order to give us the most flexibility when designing our model.

Therefore, our feature selection process needed to include multiple potential window-threshold pairings.

Naïve Filter Methods: As we had so many features, many of which did not seem useful upon initial exploration, we constructed multiple methods of naïve filtering:

1. Pairwise correlation: greedily selects a pair of features with the highest absolute correlation. It drops the feature with less predictive power (defined by either target correlation or mutual information) over the target variable. Continues until all pairwise correlations are below a specified threshold.
2. Mutual information: removes all features with mutual information score below a specified threshold
3. Variance: removes all features with variance below a specified threshold (features with low variance are unlikely to affect the target variable significantly
4. Best rolling window: for many data sources, we constructed rolling averages over multiple windows of time, which would likely be correlated with one another. This function selects the rolling window feature with the most predictive power over the target variable for each data source and drops the rest.

Feature Selection: After initial filtering, we designed several potential feature selection methods, which would reduce our feature counts down to the desired number (which itself was a hyperparameter).

1. k-best feature selection: using either mutual information or target variable correlation, this method selects the k-best features.

2. Recursive feature elimination: this stepwise function used a bare decision tree and incrementally removed the part which minimized the Gini impurity of classification until it only had the desired amount of features remaining

3. Random forest feature selection: this function generated a random forest with all the remaining features and then selected the features with the highest importance to the final model.

Monte Carlo Feature Selection Process: Our feature selection goal was to maximize robustness, given that we were not confident that any particular combination of filters and selection method would produce the "best" set of features. We would have tried every combination of target variable selection, filter method permutations, and selection method in an ideal world, with every possible hyperparameter combination. However, the number of iterations required was unfeasible, so we instead decided to implement a Monte Carlo-style approach. We ran the following algorithm 1000 times:

1. Select a target window and binary threshold

2. Randomly sample without replacement and apply any permutation of naïve filter methods (including none) and their respective hyperparameters

3. Randomly select one feature selection method and its respective hyperparameters (including the number of features to select)

4. Test the selected features using an out-of-the-box decision tree classifier (testing on the most recent 20% of the training data), and record the AUC of the resulting model. We chose a decision tree here because it was simple to execute and did not require additional data transformation. It could also easily handle a varying number of features.

We kept a running log of the iteration with the highest AUC for each target variable permutation. Our first goal was to use this process to fix a target variable setup. We settled on using a 10-day target window with a binary threshold of 0.001 (10 basis points) because it produced a relatively high max AUC while still maintaining somewhat balanced classes (distribution shown below), and being a realistic threshold for a trading strategy We also noticed that nearly all of the best models included just two of the naïve filters: mutual information and best window.. We decided to hold these filter method constants moving forward.

Class	Number of data points
0 - return lower than 0.001 threshold	2309
1 - return higher than 0.001 threshold	2677

Table 1. Distribution of chosen target variable: 10 basis point threshold over a 10-day window.

Letting all other methods and hyperparameters vary, we ran the following algorithm 10,000 times:

1. Randomly select hyperparameters for fixed naïve filter methods, filter data

2. Randomly select one feature selection method and respective hyperparameters

3. Record which features were selected in log.

Rather than an AUC score, each iteration's output was an incrementing count of features selected by the process. The

logic we applied was that if a part was selected frequently, despite random hyperparameters and selection methods, then that feature was likely to be a top predictor of the target variable.

Feature Ranking: The output of our feature ranking efforts is below. Our measure of feature importance was the proportion of the 10,000 randomized runs in which the feature was selected. A total of 98 features were chosen at least once, with their frequency distributed as follows:

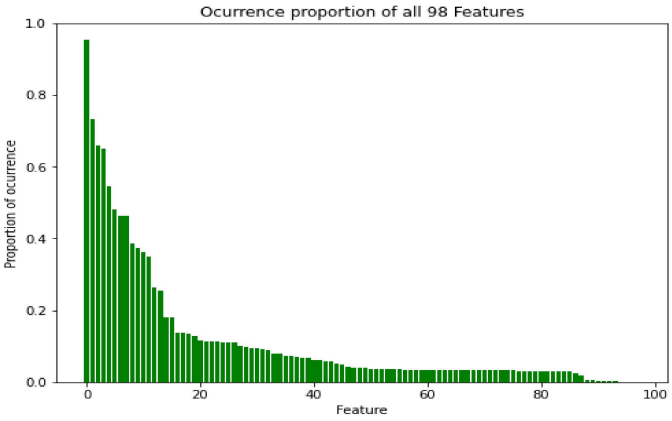

Figure 2. Occurrence proportion of all features

As can be seen, this distribution is quite top-heavy, with a few features occurring disproportionately frequently. The following chart shows the top-10 features and their occurrence proportion.

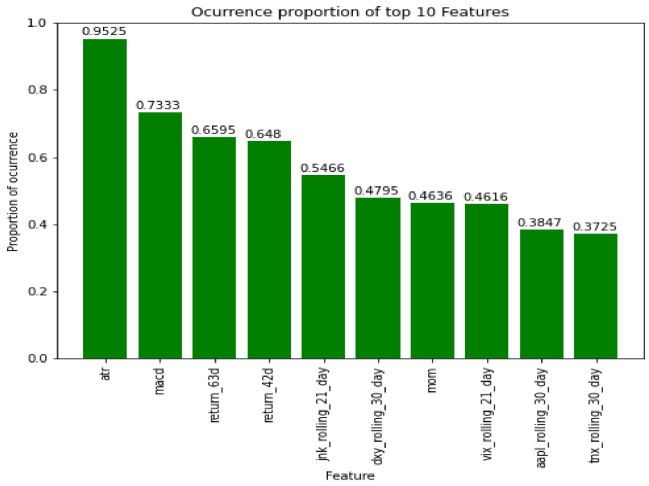

Figure 3. Occurrence proportion of the top-10 features

MODELING & EVALUATION

Baseline Model: To get a sense of how our data predict the target variable before any model selection, we ran an out-of-the-box decision tree with the top 10 features ranked during feature selection.

We chose a decision tree as our baseline model for the same reasons we used it during the feature selection process. It is important to note that for our particular analysis, the specific choice of baseline model is less relevant. We deliberately entered this project the goal of being assumption-blind, and allowing our iterative process to make decisions for us. As a result, although decision trees have been used to predict stock prices[10], that did not play a major role in our choice.

[10] https://link.springer.com/chapter/10.1007/978-3-642-13022-9_19

Our baseline model produced an AUC of 0.52, barely above the 0.5 random mark. We proceeded to try to improve on this model using an iterative bakeoff process.

Model selection overall strategy: We chose to examine a variety of potential model types, which we felt covered the spectrum of model types effectively: Logistic Regression, Decision Tree, K-Nearest Neighbors, Random Forest and Support Vector Machine.

Important to note that all of these models would output *probabilities* rather than simple classifications.

We elected to use AUC as our evaluation metric for two key reasons. Firstly, it is easy to understand and compare across a variety of models and hyperparameter combinations. Secondly, it builds in threshold uncertainty, instead of simply classifying and calculating how many were classed correctly (as in most confusion-matrix based metrics). AUC considers all possible thresholds to give us an “overall” score, which we can use to compare models. Probability thresholds can be used in the deployment phase to handle risk.

Time sensitive cross validation: Cross-validation is an important part of the model selection process, especially in situations like ours, where we do not have an overabundance of training data. However, due to our data's time-series nature, we cannot randomly split data into k-folds and cross-validate. Firstly, our data is positively correlated across time, meaning that if we were to use the future to predict the past (i.e., using returns from next week to predict returns from last month), we would likely overfit the training data.Secondly, when deployed this model will only have access to information up until today, meaning for it to have success, it can only be trained on information that is available at the time of decision making.

To deal with these restrictions, we constructed a time-sensitive cross-validation strategy[11].

1. Select ten non-overlapping 150-day windows, starting in 2013 and ending in 2018. These are our validation sets.
2. For each validation set, use the previous 1000 trading days (~4 years) as a training set.
3. For each set, build a model using the training set, test on the test set, record AUC.
4. Average the AUC scores of all sets, which we use as the "overall" AUC for a specific model-hyperparameter combination.

This data breakdown allows us to cross-validate, re-running on different slices of the same training data, while not engaging in overfitting or using unrealistic data, and sticking to the last ten years of data (2009-2018) to capture the more recent market phenomenon

How many features? The feature selection process produced 98 features with relevancy greater than zero, but there was a significant drop off in importance after the first 5-10. However, some model types may perform better with more features than others. To be safe, we elected to use up to 50 features in our model selection process (this cutoff is shown below)In an ideal world, we would have run every model with every potential number of features (1-50). However, this was way too

[11] https://otexts.com/fpp3/tscv.html

computationally expensive. As a result, we elected to iteratively increase our feature count by 5 (5, 10, 15, etc.)

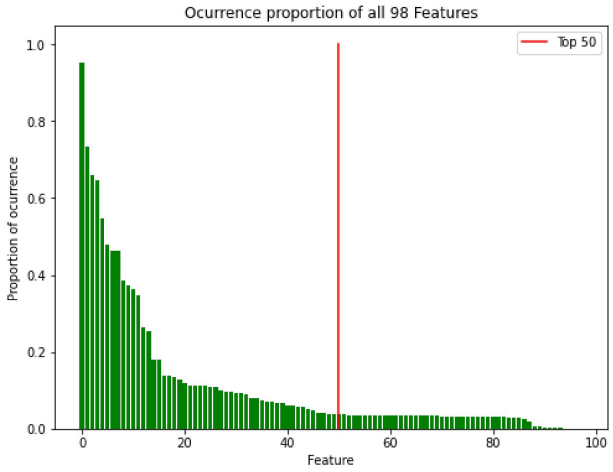

Figure 4. Occurrence proportion of all features with cutoff at 50

Model Bakeoff: As our goal was to minimize external assumptions coming into the analysis, our strategy was to run a bakeoff of multiple model types and potential sets of hyperparameters to choose the model best suited to predicting our data, there were 950 model iterations.

We ran 950 model iterations (each with ten cross-validation folds) to find the best combination for our data. While we will explore the best models later, we first want to discuss overall model performance patterns and the number of iterations in each model

Logistic Regression: We ran 50 iterations of this model, varying both the number of features used and the regularization parameter C (between 10^{-2} and 10^{2}). We found that regardless of C, a model with five features outperformed models with more features. Additionally, when features were less than 20, models with C=0.1 outperformed models with other C values.

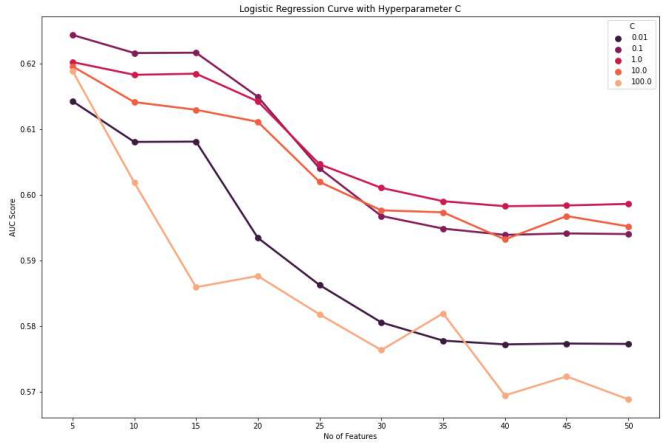

Figure 5. Model performance of Logistic Regressions

Decision Tree Classifier: We ran 80 iterations of this model, varying the number of features used and the minimum samples per leaf (between 2^{1} and 2^{8}). We found that models with 20 features outperformed others on average, and a lower number of samples per leaf increases the likelihood of overfitting and thus reduces model accuracy, while a very high number of samples per leaf leads to underfitting and bias, which also results in lower AUC. The model performs consistently better with a minimum of 8 samples per leaf.

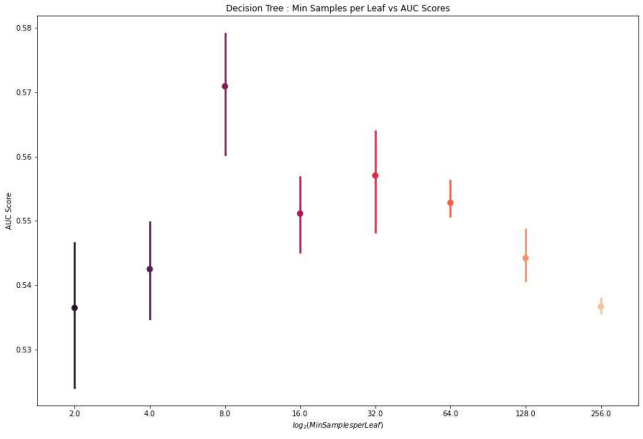

Figure 6. Performance of Decision Tree models broken down by samples per leaf

K Nearest Neighbors: We ran 400 iterations of this model, varying feature numbers, number of neighbors (between 1 and

19), type of weighting (uniform or distance-weighted) and distance metric (Euclidean or Manhattan). Using a distance-weighting of points generally performed better overall than using a uniform weighting, and Euclidean distance (L2 norm) generally outperformed Manhattan distance (L1 norm). We also found that adding features improved performance until about 30, after which it stagnated. Finally, we found that fewer neighbors lead to higher variance, complexity, and overfitting, while more neighbors increase bias, reduces complexity and overfitting. The optimal number of neighbors appears to be 3.

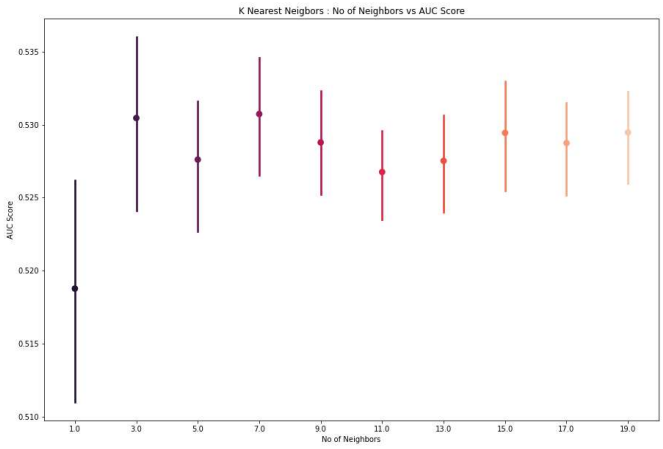

Figure 7. KNN performance broken down by number of neighbors

Random Forest: We ran 300 iterations of this model, varying the number of features, the number of trees in the forest (between 100 and 300) and the minimum samples per leaf of a given tree (between 1 and 19). Increasing the number of trees did marginally improve the model, coming at the expense of efficiency, as increasing the number of trees significantly increases runtime. Increasing the number of features in our training set significantly impacted our cross-validated AUC scores up until 20, after which it plateaued. Furthermore, we examined min samples per leaf but found it did not significantly affect model performance. We believe this is because the randomization of models and multiple iterations help to mitigate overfitting and underfitting.

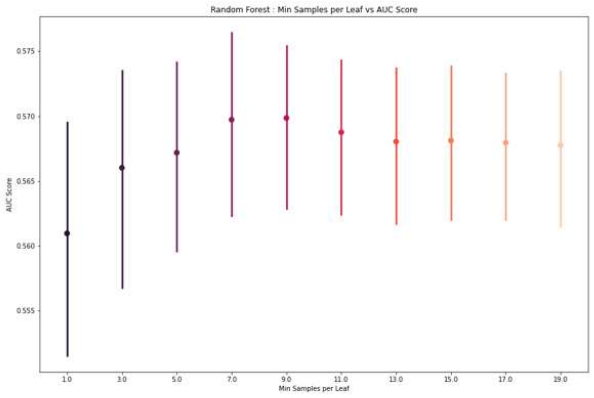

Figure 8. Random Forest performance broken down by min samples per leaf

SVM Classifier: We ran 120 iterations of this model, varying number of features, kernel type (RBF, sigmoid, polynomial and linear) and regularization parameter C (10^{-1} to 10). On average, the linear kernel performs far better than the others on average (look at that spread!). That being said, the best overall performer used a polynomial kernel. Polynomial kernels also appear to have the largest performance spread. Similar to Logistic Regression, models with C=0.1 had the best overall performance, and models with 5 features outperformed models with more.

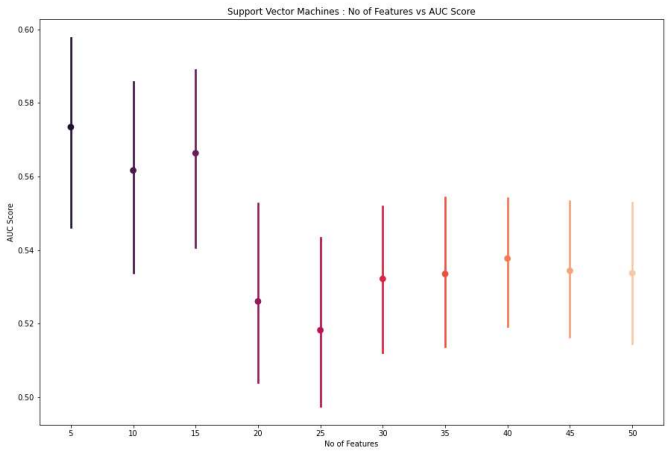

Figure 9. SVM performance by number of features

Modelling summary and evaluation on holdout set: Below are the top performing models of each type, along with their AUC scores. As can be seen, the optimal model setup was an SVM classifier, with a polynomial kernel, top 5 features and a regularization parameter of 0.1.

Model	Hyperparameters	Features	AUC
Logistic Regression	C - 0.1	5	0.624
Decision Tree	Min Leaves - 30	30	0.584
KNN	No of Neighbors - 3 Distance - Euclidean Weighting- Distance	40	0.548
Random Forest	No of Trees - 100 Min Leaves - 7	45	0.593
SVM	C - 0.1 Kernel - Polynomial	5	0.625

Table 2. Final model performance

Finally, we applied the top model to our holdout set, training on the 1000 days prior to 2019 and predicting returns in 2019. Unfortunately, this only produced an AUC of 0.54, which is marginally better than our baseline AUC of 0.52. While this does seem to indicate that our null hypothesis - that financial data is mostly noise - is true, it is also worth considering our models performed so much better on training data, despite model variety and efforts to cross-validate. One reason could be that feature importance shifts over time, and thus features important in 2013 may not be in 2019. We could also be the victims of a particularly tricky holdout set, being constrained both by the time-series nature and limited scope of the data.

DEPLOYMENT

Although our final model did not end up predicting the holdout set with significant improvement, it is worth discussing how such a model would be deployed if it were more effective. We envisaged this model deployed as part of an algorithmic trading strategy. Each day, the model would run on the past 1000 days of data and output the probability of returns exceeding the 10 basis point threshold in 10 days. Traders could then choose thresholds on which to purchase call options on QQQ. They could adjust their probability threshold to vary risk without having to adjust the model at all. This was why we chose to output probabilities instead of basic classifications, and why we used overall AUC as an evaluation metric.

Our aim was to use an assumption-blind approach to try and develop a model capable of isolating the signal from the noise of financial data. Although we were unsuccessful, we believe that our process and strategy, if combined with additional financial insight and existing trading strategies, could be a valuable part of an algorithmic trading tool used by options traders to evaluate decisions.

WORKS CITED

Schulze, Elizabeth. “The Fed Launched QE Nine Years Ago - These Four Charts Show Its Impact.” *CNBC*, CNBC, 25 Nov. 2017, www.cnbc.com/2017/11/24/the-fed-launched-qe-nine-years-ago--these-four-charts-show-its-impact.html.

David, Zachary. “Fitting to Noise or Nothing At All: Machine Learning in Markets.” *Zachary David's*, 14 Mar. 2019, zacharydavid.com/2017/08/06/fitting-to-noise-or-nothing-at-all-machine-learning-in-markets/.

“Decision Trees in Stock Market Analysis: Construction and Validation.” *Trends in Applied Intelligent Systems 23rd International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2010, Cordoba, Spain, June 1-4, 2010, Proceedings, Part I*, by Margaret Miró-Julià et al., Springer Berlin Heidelberg, 2010, pp. 185–194.

Gallant, Chris. “The Difference Between the Arithmetic Mean and Geometric Mean.” *Investopedia*, Investopedia, 16 Sept. 2020, www.investopedia.com/ask/answers/06/geometricmean.asp.

Gittelsohn, John. “End of Era: Passive Equity Funds Surpass Active in Epic Shift.” *Bloomberg.com*, Bloomberg, 2019, www.bloomberg.com/news/articles/2019-09-11/passive-u-s-equity-funds-eclipse-active-in-epic-industry-shift.

Gumbinger, Keith. “The Latest Move by the Federal Reserve - November 5, 2020.” *Hsh.com*, 5 Nov. 2020, www.hsh.com/finance/mortgage/latest-move-by-the-federal-reserve.html.

Jansen, Stefan. *Machine Learning for Algorithmic Trading: Predictive Models to Extract Signals from Market and Alternative Data for Systematic Trading Strategies with Python*. Packt Publishing, 2020.

Lim, Dawn. “Index Funds Are the New Kings of Wall Street.” *The Wall Street Journal*, Dow Jones & Company, 18 Sept. 2019, www.wsj.com/articles/index-funds-are-the-new-kings-of-wall-street-11568799004.

TA-LIB.org. “Lib : Technical Analysis Library.” *TA*, ta-lib.org/.

“Time Series Cross-Validation.” *Forecasting Principles and Practice*, by Rob J. Hyndman and George Athanasopoulos, OTexts, 2018.

APPENDIX A: CONTRIBUTION

Keshav Pant (kp2749): Idea generation/initial planning, general planning, data sourcing, data preparation/feature engineering, feature selection, writing, proofreading and editing.

Emma Muhleman (emm10005): Idea generation/initial planning, general planning, business understanding, data sourcing, writeup

Sudharsan Asaithambi (sa6149): Idea generation/initial planning, general planning, data preparation/feature engineering, model selection/evaluation, writeup

Gurneet Bedi (gb2388): General planning, Model selection/evaluation, writing, proofreading and editing

APPENDIX B: DATA SOURCES

Stock	Ticker	Source	Rows	Start Date	End Date
Invesco QQQ Trust Series 1	QQQ	Yahoo! Finance	5458	1999-03-10	2020-11-12
Apple Inc	AAPL	Yahoo! Finance	5755	1998-01-02	2020-11-12
Amazon.com, Inc.	AMZN	Yahoo! Finance	5755	1998-01-02	2020-11-12
Facebook, Inc. Common Stock	FB	Yahoo! Finance	2137	2012-05-18	2020-11-12
Alphabet Inc Class C	GOOG	Yahoo! Finance	4089	2004-08-19	2020-11-12
Microsoft Corporation	MSFT	Yahoo! Finance	5755	1998-01-02	2020-11-12
Tesla Inc	TSLA	Yahoo! Finance	2614	2010-06-29	2020-11-12
U.S. Dollar Index	DXY	Investing.com	5914	1998-01-01	2020-11-12
iShares MSCI Emerging Markets ETF	EEM	Yahoo! Finance	4429	2003-04-14	2020-11-12
SPDR Gold Trust	GLD	Yahoo! Finance	4025	2004-11-18	2020-11-12
SPDR Bloomberg Barclays High Yield Bon ETF	JNK	Yahoo! Finance	3260	2007-12-04	2020-11-12
CBOE Treasury Yield 10 Years	^TNX	Yahoo! Finance	7035	1998-01-02	2020-11-12
CBOE Volatility Index	^VIX	Yahoo! Finance	7779	1990-01-02	2020-11-12

Table 3. Details on all data sources used

APPENDIX C: POTENTIAL FEATURES

Feature	Time Period	Definition
Volume	Daily	The amount of the stock that was bought and sold during the day.
Relative Strength Index	14-day	A momentum indicator which measures the size of recent price changes to evaluate whether a stock was overbought or oversold.
Bollinger Bands (x2)	20-day	A measure of two standard deviations above and below a simple moving average of price.
Average True Range	14-day	Simple moving average of the daily True Range - the maximum out of the following three calculations: 1) High - low. 2) Absolute value of high - yesterday's close. 3) Absolute value of low - yesterday's close
Moving Average Convergence Divergence	short-term: 12-day, long-term: 26-day	Difference between the short term exponential moving average and the long term exponential moving average.
Momentum	10-day	Speed of price changes
Rate of Change	10-day	((Current price / previous price)-1) x 100
Candlestick Patterns (x61)	Full data	Patterns within "Candlestick" charts: box-and-whisker plots of price, where the open and close price forms the "real body" (box) and the high and low form the "wicks" (whiskers).

Table 4. QQQ Technical indicators

Name	Ticker	Definition
U.S. Dollar Index	DXY	The strength of the U.S. Dollar against a trade-weighted basket of major currencies (EUR, JPY, GBP, CAD, SEK, CHF)
iShares MSCI Emerging Markets ETF	EEM	An ETF tracking an index made up of various emerging market equities
SPDR Gold Trust	GLD	An ETF representing ownership in gold bullion.
SPDR Bloomberg Barclays High Yield Bond ETF	JNK	An ETF tracking the performance of the Bloomberg Barclays High Yield Very Liquid Index
CBOE Treasury Yield 10 Years	^TNX	10-year U.S. Treasury Notes
CBOE Volatility Index	^VIX	A real-time market index representing 30-day forward looking volatility

Table 5. Macroeconomic Indicators

APPENDIX D: FEATURE SELECTION PARAMETERS

Parameter	Definition	Values
Window	The period of time to consider when looking at returns	[1-day, 5-day, 10-day, 21-day]
Threshold	The value around which to base the binarization of the target return variable (0 if below, 1 if above)	[0, 0.0005, 0.001, 0.002, 0.005, 0.1]

Table 6. Target variable selection parameters

Method	Hyperparameter	Definition	Values
Filter by Variance	Threshold	The minimum variance value included in the feature list. Features with variance equal to or below this would be dropped	0, 1e-4, 1e-3, 1e-2, 1e-1
Filter by Mutual Information	Threshold	The minimum MI value included in the feature list. Features with MI equal to or below this would be dropped	0, 0.001, 0.002. 0.005, 0.01
Filter by Correlation	Threshold	The minimum pairwise correlation required in order to filter values.	0.6, 0.7, 0.8, 0.9
Filter by Best Window	Metric	The metric to use when comparing different window/rolling average features based on the same ticker.	Mutual Information, Correlation with target variable

Table 7. Naive filter method parameters

Method	Hyperparameter	Definition	Values
K-Best Feature Selection	Metric	Metric to use in order to rank features	Mutual Information, Correlation with target variable
K-Best Feature Selection, Recursive Feature Elimination, Random Forest Feature Selection	Number of Features	Number of Features to select	5, 8, 15, 20

Table 8. Feature selection parameters

APPENDIX E: MODEL SELECTION PARAMETERS

Model	Hyperparameter	Definition	Values
Logistic Regression, Decision Tree, KNN, Random Forest, SVM	Number of features	The number of features used.	5, 10, 15, 20, 25, 30, 35, 40, 45, 50
Logistic Regression, SVM	C (Regularization Parameter)	Strength of regularization (smaller C = stronger regularization)	0.01, 0.1, 1, 10, 100
Decision Tree, Random Forest	Min samples per leaf	The minimum number of samples required by a leaf node.	2, 4, 8, 16, 32, 64, 128, 256
KNN	Weighting	Whether to weight by distance or uniformly	Uniform, Distance
KNN	Distance	Distance metric to use when evaluating neighbors	Euclidean Distance, Manhattan distance
KNN	Number of neighbors:	Indices of and distances to the neighbors of each point.	1, 3, 5, 7, 9, 11, 13, 15, 17, 19
Random Forest	Number of Trees	Number of trees	100,200,300
SVM	Kernel type	Returns the inner product between two points in a standard feature dimension.	RBF, Sigmoid, Polynomial, Linear

Table 9. Model Selection Hyperparameters