Technical and Machine Learning Trading Strategies for Nasdaq Futures

Author
Ao Shen
as5017@ic.ac.uk

Supervisors
Prof. William J Knottenbelt
Alexei Zamyatin

Department of Computing

September 10, 2018

Imperial College London

Why Nasdaq Futures?

Definition

Standardized forward contracts that obligate parties to buy and sell at a predetermined price and date.

- Futures, promptly named, are **leading price indicators** to the underlying asset.
- Popularity of “index investing" whereby investors buy a basket of stocks to diversify risks has **reduced volatility**
- CME's E-mini Nasdaq 100 futures enjoy **immense volume**, and **longer trading hours** than stocks
- Cash-settled, thus **cheap margin** requirements
- Nasdaq 100 has been the **best performing** major US stock index for the past 10 years.

Which Means…

- More data
- Less noise
- Cost efficient
- Leverable

Making it the perfect study subject for machine learning and technical analysis

Imperial College London

Objective

Explore the entire process of trading Nasdaq Futures by:

- Building strategies to **outperform** the **risk-adjusted return** of simply **buying and holding** the future
- Use **machine learning** and **technical analysis** techniques to not predict the exact price of the asset tomorrow (fool’s errand), but **classify** its **direction** (up or down) through historical price and volume data
- Develop a **market simulator** to tie the project together
- Find out which **technical indicators** are the **best** using Sharpe Ratio and **Principal Component Analysis**
- **Validate** our results using **Welch's t-test** and **confusion matrix** statistics analysis.

And of course… as a side objective... to hopefully make money

Machine Learning Results

- **Ensemble** of **4 technical indicators and LSTM** was constructed in Jupyter Notebooks using TensorFlow, and trained on the Google Cloud. The model achieved **64 % MCC** and a **21 % increase in Sharpe**

- The ensemble **outperformed** the indicators and LSTM **individually**, implying **synergies exist** when combing technical analysis and machine learning.

- **Random Forest ensemble** model was implemented from raw **Python and NumPy** to test if **greedy** algorithms can exploit close-interval data dependencies. A statistically significant **61% MCC** and **16%** increase in Sharpe Ratio, implies a **relationship exists** between **recent historical and future** prices.

- **Q-learning ensemble** plagued with **challenges. Non-convergence** during training was overcome with **discretization** using **K-means and Gaussian Mixture Model (GMM)**. Clustering was performed according to **Silhouette Coefficient**. **Dyna-Q** was implemented to speed up **convergence**. However, the model **failed** to achieve statistically significant results.

- Failure may be due to **Markov chains lacking memory**. Discretization reduces problem space but may have reduced the **precision** of our model.

Imperial College London

Technical Analysis Results

- **Principal Component Analysis** and **Correlation Matrix** were produced using SciPy and Seaborne to find the most important factors among the **29** technical indicators. The indicators could be **narrowed down to 6** without compromising the Sharpe Ratio, of which the above **3 were the strongest performers**

- **Moving Average Divergence Convergence, Relative Strength Index, and On Balance Volume**, in descending order are the **strongest** individual predictors for technical analysis

- Designed and implemented a novel technical indicator -- **Max Drawdown Probability**, which allows a strategy to further enhance its performance by **not trading** on days with **potentially high losses**

- On Balance Volume, and Maximum Drawdown Probability enhances signal accuracy at the expense of signal reduction. In other words, fewer but higher probability success trades are made.

- Built a **modular market simulator** from scratch using NumPy and Pandas. The API takes in financial data, transaction costs, technical indicator and machine learning model signals. It then sorts them according to a proprietary algorithm. Subsequently, it backtests the signals, and computes metrics such as MCC, t-tests, and confusion matrix statistics. Finally, it produces an executable trading instructions

Other Results

- **RF** and **LSTM** ensemble were applied to **Bitcoin** and **Ethereum**. **LSTM outperformed** the benchmark on Ethereum, **achieving 56% MCC** and **13% higher Sharpe Ratio**. However, it performed poorly on Bitcoin, which may be due to the high number of unoptimized hyperparameters. Random Forest did not outperform significantly.

- **High classification rates do not translate** perfectly into **higher Sharpe Ratio** according to confusion matrix analysis. The biggest factors were the algorithms' **blindness to the magnitude** of price swings, and its **low** prediction **accuracy on sells**. This was the motivation behind our drawdown indicator.

- Meta analysis show that **lower frequency strategies perform** better than their **high frequency** counterparts. By acting on rare and and seldom occurring events, signals became more robust to noise, thus increasing the probability of profitable outcomes.

- Overall results imply that **learnable features exist** in the Nasdaq Futures

Imperial College London

Performance Metrics

Daily Return
Today's closing price divided by previous day's close. **Higher is better.**

$$Daily\ Return = \frac{Price_1}{Price_0} - 1$$

Volatility
Standard Deviation of Daily Return. **Lower is better**.

$$Volatility = \sigma\ Daily\ Return$$

Sharpe Ratio (SR)
Daily Return over its Standard Deviation. A higher Sharpe is more important than cumulative return because we can use leverage to outperform that cumulative return. **Sortino ratio**, which only uses **negative standard deviation** is a good alternative. Nonetheless, since Sharpe is the industry standard, its optimization is our goal. **Higher is better**.

$$Daily\ Sharpe\ Ratio = \sqrt{Days}\ \frac{Mean\ Return - Risk\ Free\ Rate}{Volatility}$$

We want to max Sharpe, but can we?

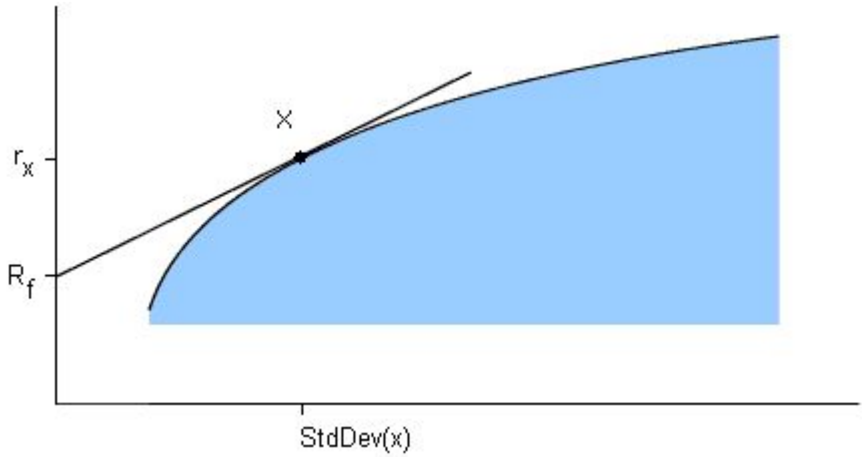
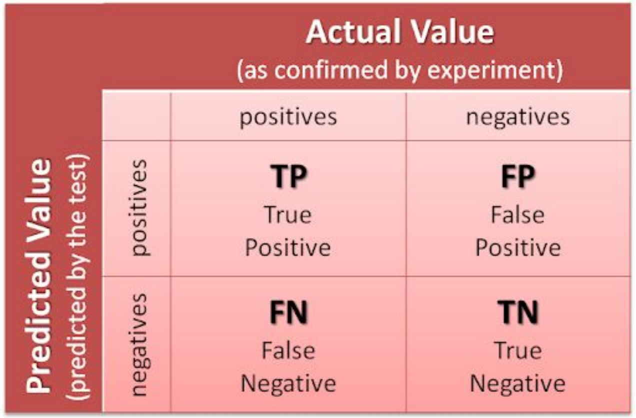


Image showing Sharpe Ratio in its ideal **convex** form.[1]

In complex cases, it can be **quasiconvex** (with some local minimum and flat slopes).
Perfect optimization problem for machine learning!

Imperial College London

Matthews Correlation Coefficient



		Actual Value (as confirmed by experiment)	
		positives	negatives
Predicted Value (predicted by the test)	positives	**TP** True Positive	**FP** False Positive
	negatives	**FN** False Negative	**TN** True Negative

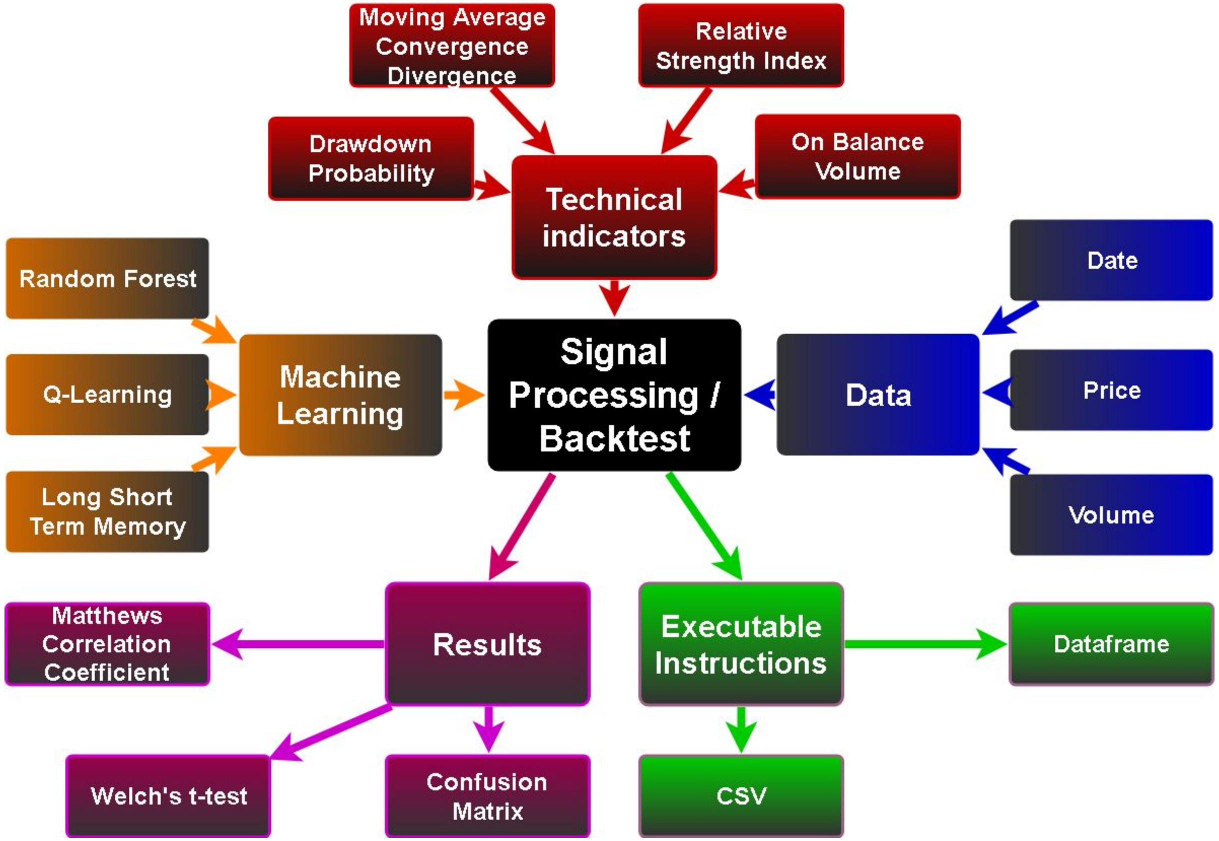
Image showing Confusion Matrix[2]

Matthews Correlation Coefficient (MCC) is the correlation between our prediction and the results. 1 implies perfect correlation, 0 means none, -1 means inverse correlation. No confusion statistic (accuracy, precision, recall, F1) shows as complete of a picture as MCC. I think of it as the Pearson's Correlation Coefficient (r) of machine learning.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

Imperial College London

Modular Market Simulator



Imperial College London

Welch's unequal variances t-test

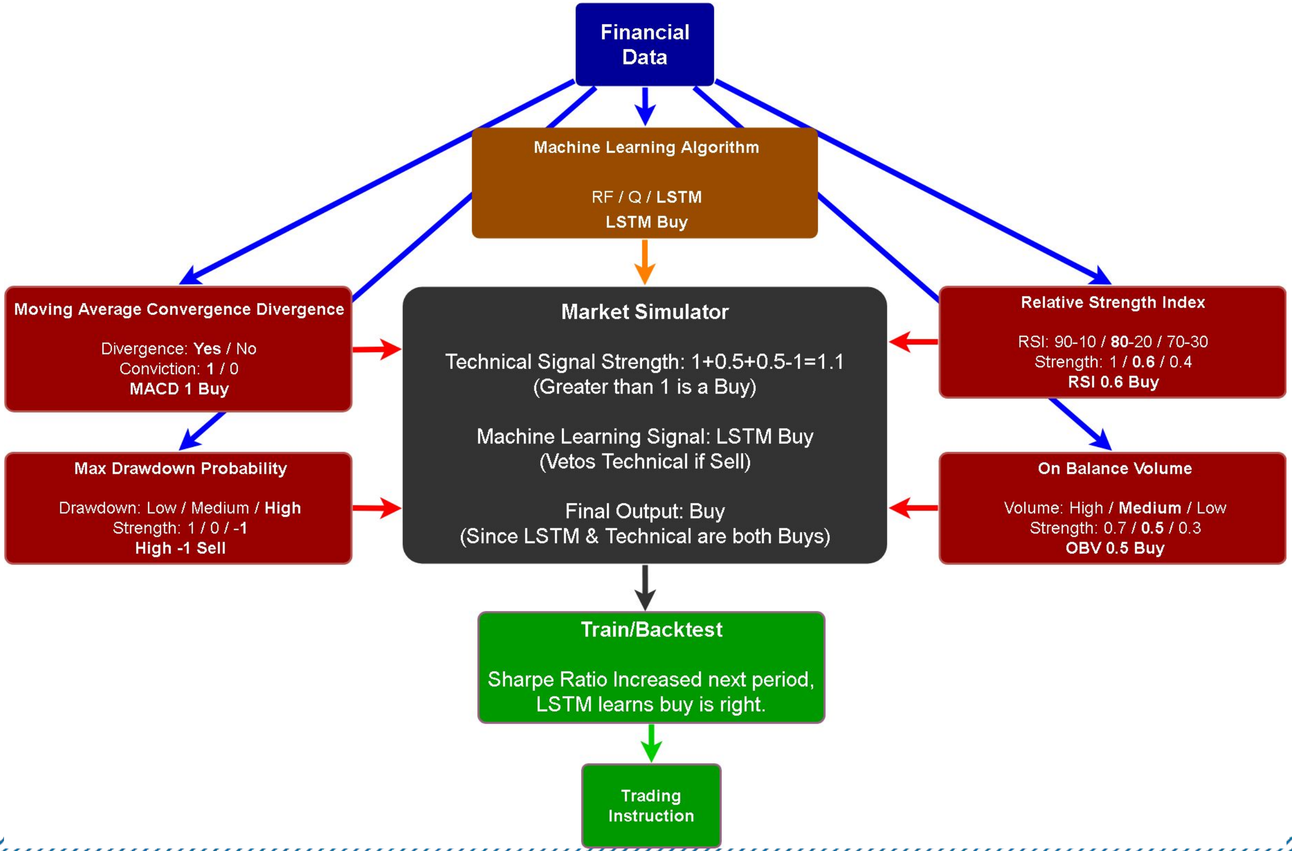
- Good classification may simply come from **luck** or skewed classification distribution
- Welch's unequal variances t-test allows us to compare distributions with **unequal sample size and variance**
- We wish to reject the **null hypothesis** that our strategy does not outperform a random walk with a 95% confidence.
- To past the test, we need to ensure that the t-stat is positive and the critical p-value is < 0.05$.

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}}$$

where X, S, N are the sample means, sample variances and sample sizes, respectively.

Imperial College London

Ensemble



Financial Data
Machine Learning Algorithm
RF / Q / **LSTM**
LSTM Buy
Moving Average Convergence Divergence
Divergence: **Yes** / No
Conviction: **1** / 0
MACD 1 Buy
Max Drawdown Probability
Drawdown: Low / Medium / **High**
Strength: 1 / 0 / **-1**
High -1 Sell
Market Simulator
Technical Signal Strength: 1+0.5+0.5-1=1.1
(Greater than 1 is a Buy)
Machine Learning Signal: LSTM Buy
(Vetos Technical if Sell)
Final Output: Buy
(Since LSTM & Technical are both Buys)
Relative Strength Index
RSI: 90-10 / **80**-20 / 70-30
Strength: 1 / **0.6** / 0.4
RSI 0.6 Buy
On Balance Volume
Volume: High / **Medium** / Low
Strength: 0.7 / **0.5** / 0.3
OBV 0.5 Buy
Train/Backtest
Sharpe Ratio Increased next period, LSTM learns buy is right.
Trading Instruction

Imperial College London

PrinciPAL Component Analysis

Step 1: Covariance matrix

$$Covariance\ Matrix_{ij} = \frac{1}{N-1}\sum_{N=1}^{N} X_{ij}X_{nj} = \frac{1}{N-1}(X^T X)$$

where ij are the Sharpe Ratios of the two indicator, and n is the total number of indicators

Step 2: Linear Transformation

$$Z = XV$$

where Z is the data we want to transform. X is the decoded signal. V is an orthogonal matrix.

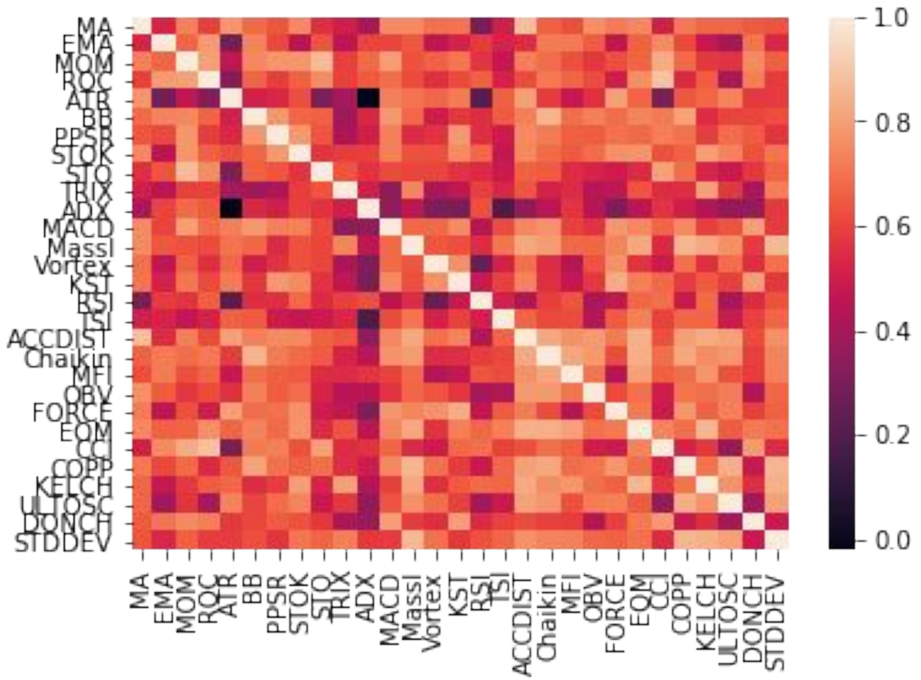
Step 3: Eigenvalue Decomposition and Correlation Matrix

$$Cov_{=}\frac{1}{N-1}Z^T Z = \frac{1}{N-1}V^T X^T XV = V^T CV = V^T UA(V^T U)^T \rightarrow A$$

where C is correlation matrix. A is the diagonal matrix of ordered eigenvalue. U is orthogonal matrix that stores eigenvectors

Imperial College London

Correlation Matrix



Note the light shade of the Correlation Matrix. This implies a significant amount of correlation between the indicators. This is reasonable given that many technical indicators tend to give correlated signals since they are generally linear and based on just price and volume data.

Imperial College London

PCA Results

Features	MCC	SR
29	0.5904	0.5510
12	0.6215	0.5610
6	**0.6205**	**0.5620**
5	**0.4315**	**0.4888**

Note that PCA reduction further than 6, resulted in MCC and SR loss

Imperial College London

MACD

The Moving Average Convergence-Divergence indicator (MACD) is the **strongest** performer
Uses exponential moving average, giving later or more recent values higher weight

$$EMA_t = \frac{Y_1, t=1}{\alpha \cdot Y_t + (1-\alpha) \cdot -1, t>1}$$

where Y is the data series. t is the time window. alpha is the smoothing factor between 0 and 1

$$MACD\ Line = EMA(12) - EMA(26)$$

A third EMA usually of 9 days, is used to creating a crossover point (signal) for the other 2 EMAs (creating an oscillator).

$$Signal\ Line = EMA(MACD\ Line, 9)$$

When the 9-day EMA intersects the MACD, in an ascending formation, is when one should buy. When the lower EMA is descending towards the moving average, is when one should sell.

$$MACD = MACD\ Line - EMA(9)$$

Imperial College London

RSI

- Relative Strength Index (RSI) is a momentum oscillator, our 2nd strong performer.
- Standard usage dictates **>70**, is overbought, implying an impending **drop** in price. When **<30,** a **rally** should ensue.
- Its predictive power however, lies when the oscillator reaches **extremes (>90% or <10%)**.

$$RSI = 100 - \frac{100}{1 + Relative\ Strength}$$

$$Relative\ Strength = \frac{Simple\ Moving\ Average(Up, n)}{Simple\ Moving\ Average(Down, n)}$$

Imperial College London

OBV

- A breakout or down is only significant if sufficient volume was used to push it through
- High volume is often (but not always) the **consensus** of a large number of market participants
- Low volume environments are also subject to **market manipulation**.
- While volume itself is not a signal, it is a great validator to other indicators, confirming high volume trends, and eliminating low volume noise, increasing technical indicators' probability of success.
- Purchasing during high volume also **reduces impact**, as the bid-ask spread shrinks.

$$OBV = OBV_{prev} \begin{cases} volume & if close > close_{prev} \\ -volume & if close < close_{prev} \end{cases}$$

OBV In Conjunction With...

Indicator	Before MCC	After MCC	Before SR	After SR
SMA 200	0.5315	0.5715	0.5778	0.5798
BB 200	**0.5415**	**0.4812**	0.5431	0.5725
MACD 12-26-9	0.6158	0.6219	0.6088	**0.6090**
RSI 10/90	0.6072	0.6171	0.6122	**0.6291**
Sup/Res	0.5756	0.5823	0.5698	0.5722
Benchmark				**0.5591**

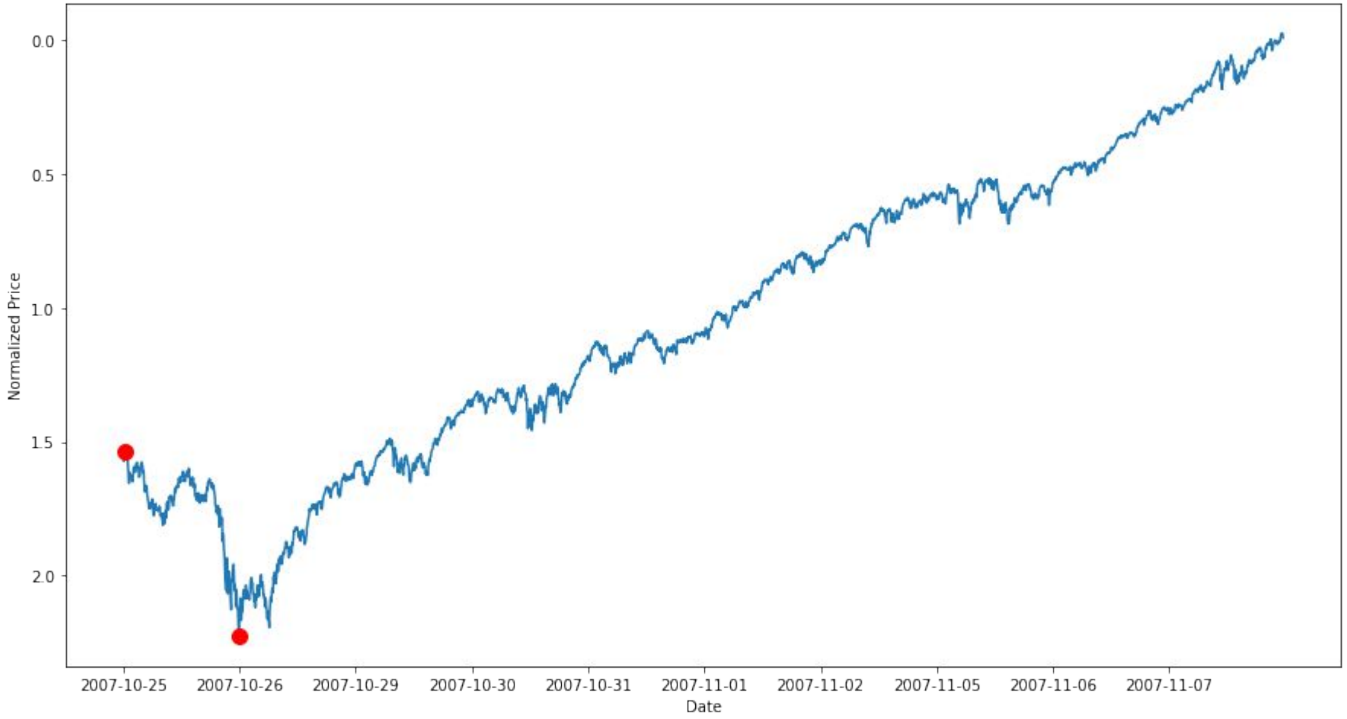
Note that MACD and RSI were the best performers before and after OBV boost.

Also note that Bollinger Bands actually performed worse after volume addition. This likely has to due to the fact that volatility are high during sell-offs, and BB uses standard deviation as its bands.

Imperial College London

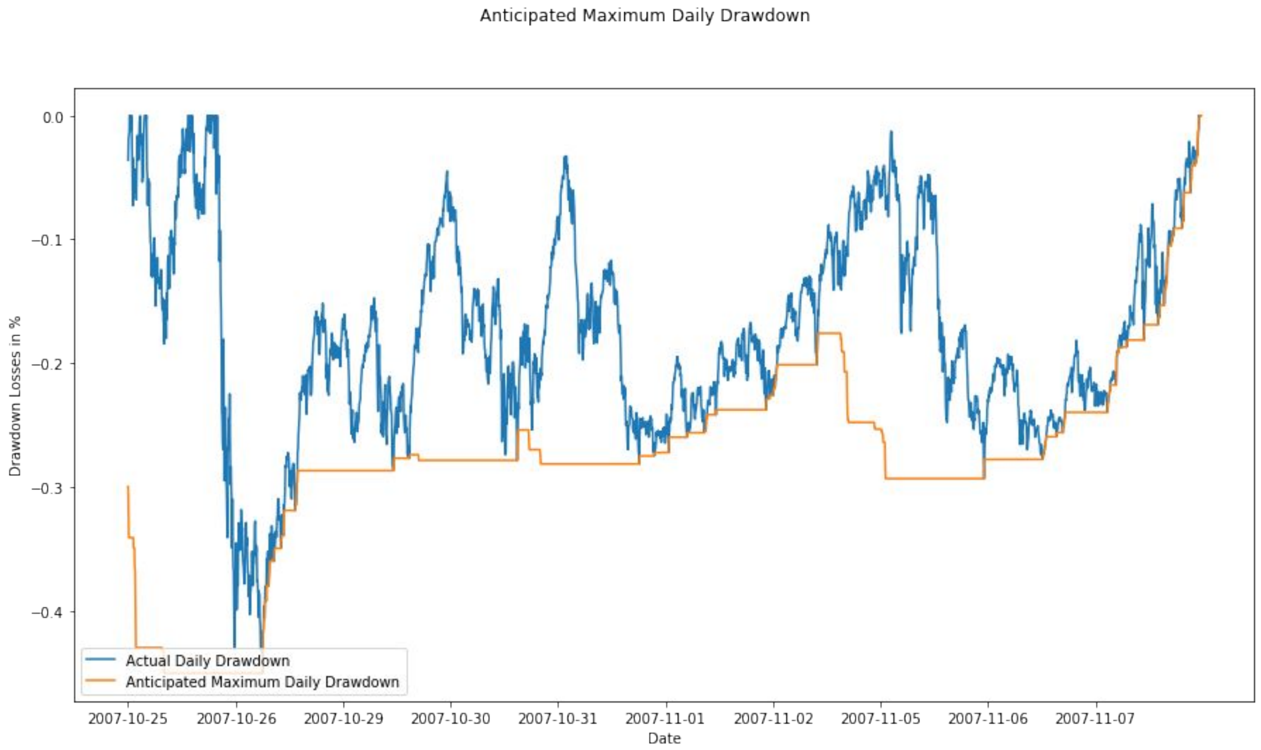
MDP

- Max Drawdown Probability (MDP) measures the probability of max peak to trough declines
- The magnitude of the gap and its frequency can inform us of large potential losses.



Imperial College London

MDP



Note that the prediction is poor in the beginning but gradually improves. On periods of extreme volatility, MDP tells us to stay put, avoiding losses at the cost of reduced upside.

Why Random Forest?

- Works well on **almost everything**
- Minimal data preprocessing
- Few hyperparameter tuning
- **Greedy**, meaning we can study the relationship between recent historical and future prices
- It overcomes the problem of decision trees' tendency to overfit. (low bias, but very high variance). Random Forest averages multiple trees through **bagging**, **reducing variance**, but introducing only a **slight bias**.

Imperial College London

Random Forest Parameters

- **Feature selection** based on **Information Gain** to reduce **entropy**. **Random** split occurs when features are tied.

$$I(N) = I(N) - P_L * I(N_L) - P_R * I(N_R)$$

where I(N) is Information Gain of the node. P_L is the population in node N that splits into the left child. P_R is the population in node N that splits into the right child. N_L N_R is the left and right child of N respectively.

- Features include **closing, high, low, change%, volume**

- We allow the tree to split until all data are classified, hence the minimum leaf size is 1.

- However, we begin to **prune** after about 300 nodes in depth.

- Bootstrap aggregating or **Bagging** randomly samples the training set with replacement to decrease the resultant tree variance without increasing much bias. While the predictions of a single tree are highly sensitive to noise in its training set, the average of many trees is not, as long as the trees are not correlated.[3]

$$B\hat{a}gs = \frac{1}{B}\sum_{b=1}^{B} Bags_b(\grave{x})$$

Imperial College London

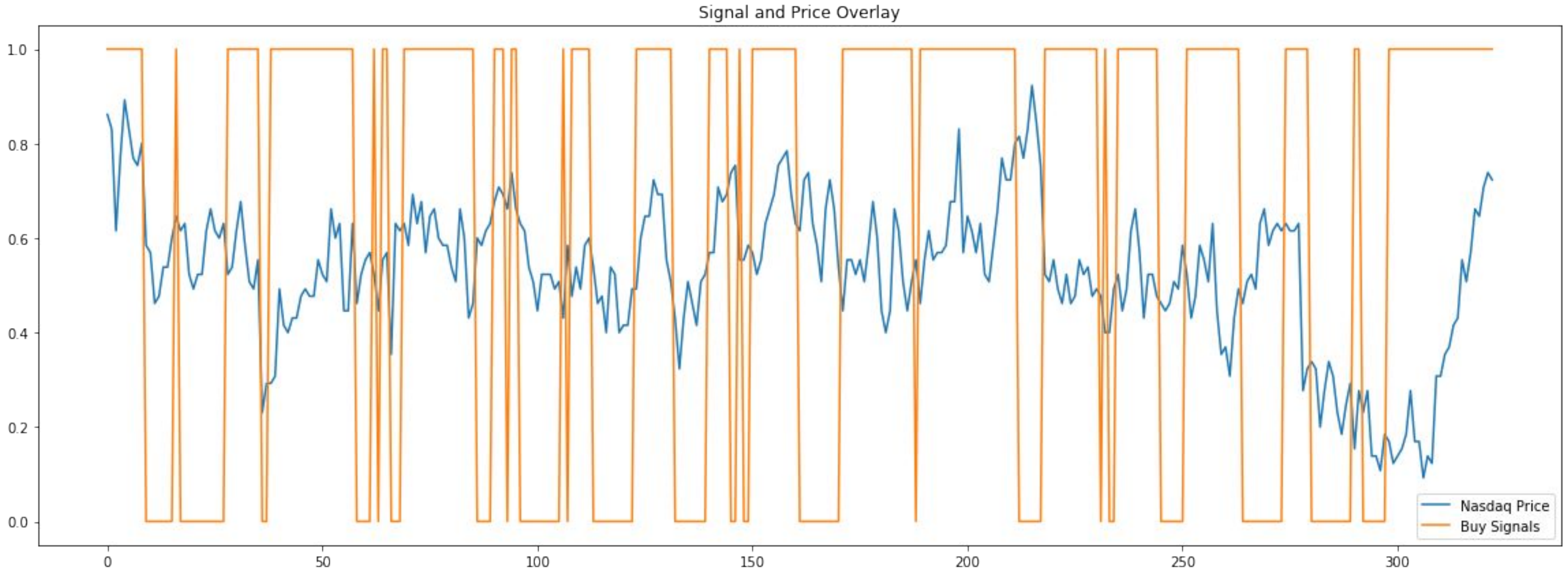
Random Forest Results

Parameter	Signals	MCC	SR
Random Forest	**413**	0.5536	0.5791
RF Ensemble	**87**	0.6089	**0.6421**
Benchmark	N/A	N/A	**0.5591**

Note how the Ensemble outperformed the Benchmark by achieving 61% MCC and almost 16% increase in Sharpe Ratio. This came at the expense of reduced signals (not acting on potential false positives/negatives). These results are statistically significant according to the t-test.

Imperial College London

Random Forest Results



Stock price (blue) overlaid with trade signals (orange). We want the orange lines to remain on top when stocks go up, and on bottom when stocks go down.

Imperial College London

Why Q-learning?

In reinforcement learning, the model trains from **reward and punishment**, rather than labeled data. This gives it some awareness of **magnitude** (accumulating rewards)

Model-free approximation of **dynamic programming**. Not greedy, but searches the problem space. The drawback is that the maximum reward chain may be overfitted and not applicable to future market environments.

Q-learner first '**explores**' all actions randomly. As the table filles up, Q-learner can choose to '**exploit**' the best action from the Q table. As a Q-function converges, it returns more consistent Q-values and the amount of **exploration decreases**.

DeepMind actually decreases randomness over time from 1 to 0.1 – in the beginning the system makes completely random moves to explore the state space maximally, and then it settles down to a fixed exploration rate.[4]

In a Markov chain, for any given state the process has a certain probability of transitioning to another state with every step. A famous application of Markov chains is **Google's PageRank algorithm**, which models link traversal as a Markov process, with each link on a webpage having a certain probability of being utilized.

Markov Decision Process

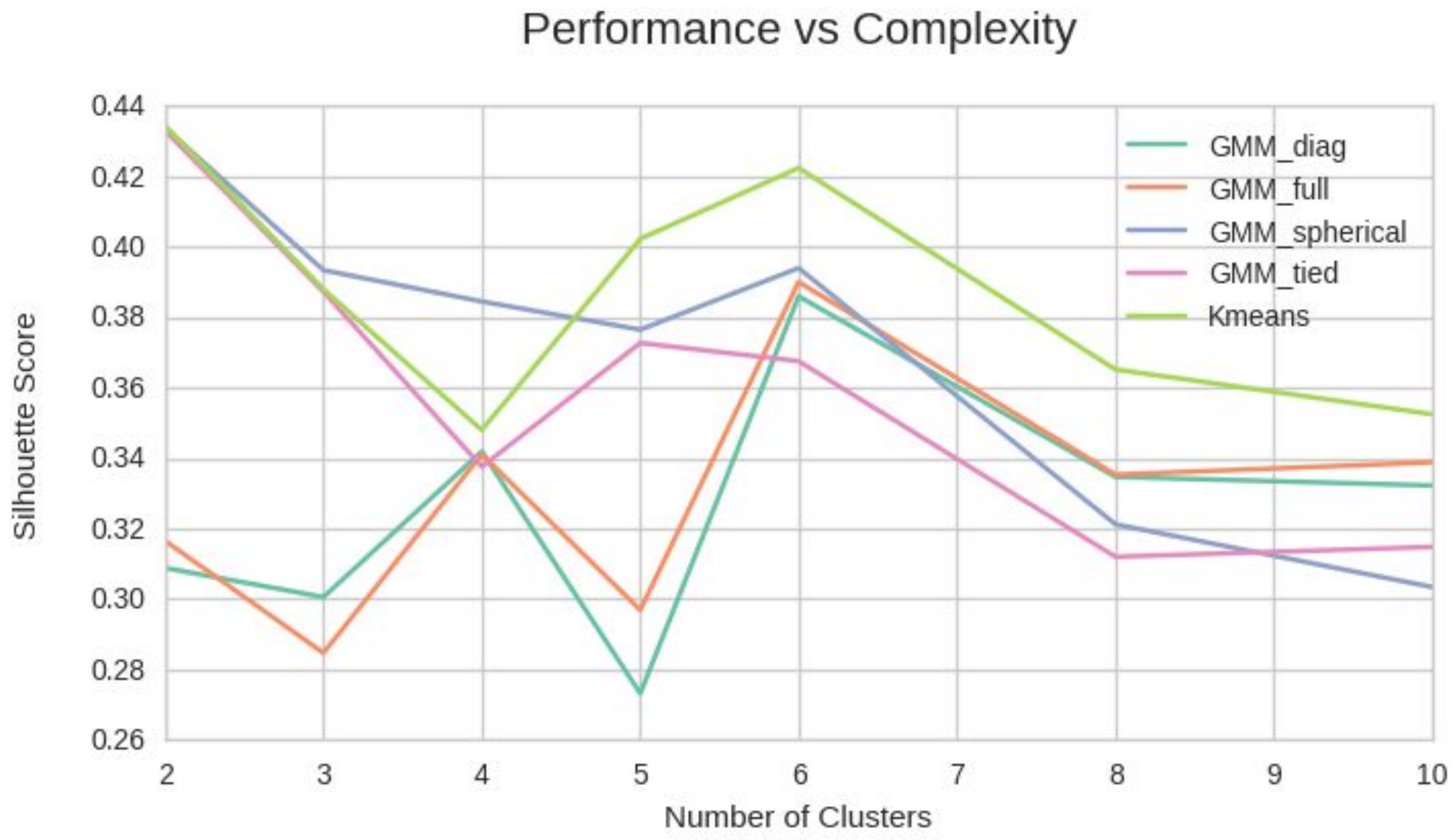
According to Chen[5], MDP is the most effective model for implementing reinforcement learning. The MDP environment contains discrete set of States S, and a discrete set of Actions A. At each step t it is allowed to choose an action a_i from a subset of action space A. In our case, the actions is buy/sell/nothing

At each discrete time step t, the agent reads the current state s_t and chooses to take action a_t. The simulator responds by providing the agent a reward. It is memoryless meaning that one should be able to predict the future of a given process based on its present state just as well as if given the entire history of the process.

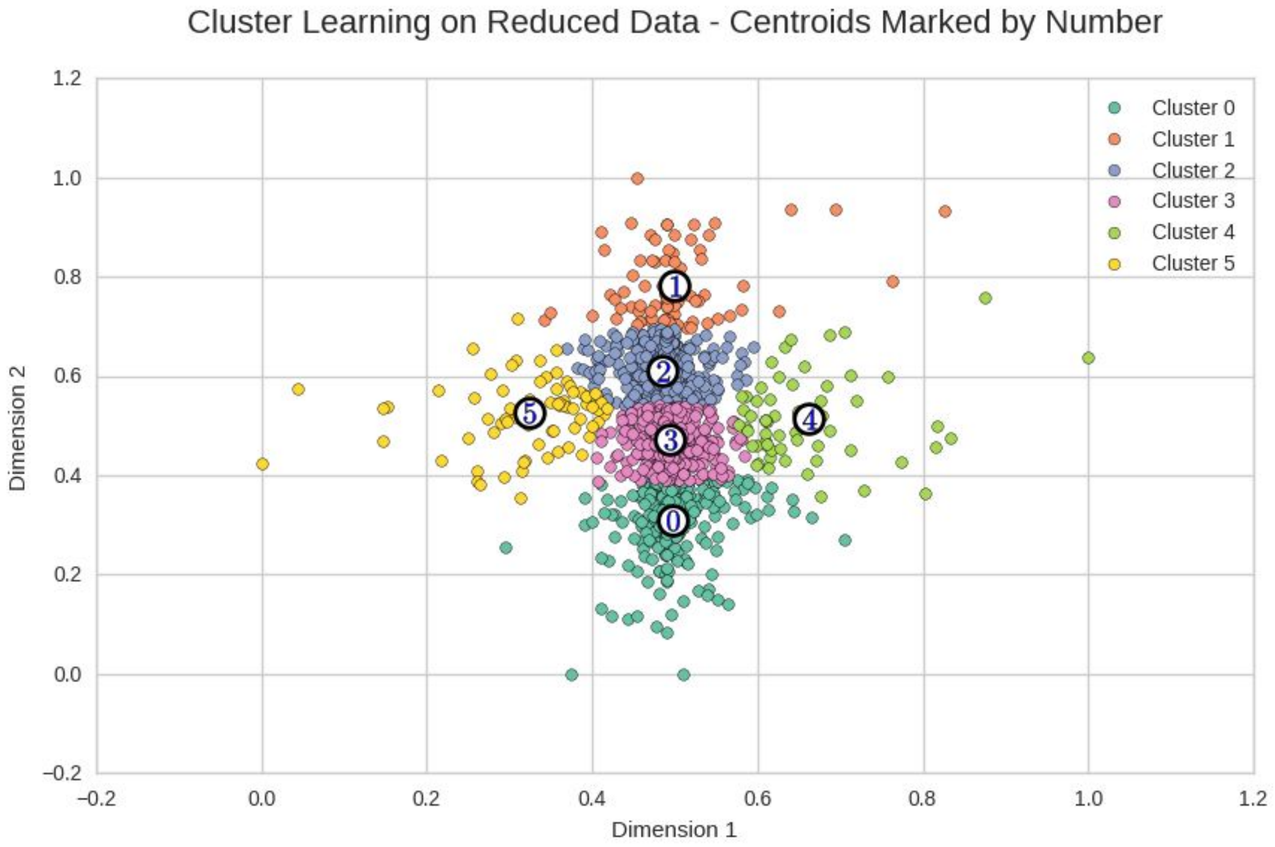
Since our **state is discrete**, we need to discretize our data. **Simply binning** it by a fixed size is not useful, so we will try both **K-means** (based on **silhouette coefficient** distance), and **Gaussian mixture model** (many probabilistic distributions).

We can gain additional insight into the data. (ML project within ML). Since we cannot assume distribution parameters, different covariance structures to GMM are also tested.

Imperial College London



Imperial College London



Imperial College London

Training

$$Q(s,\ a) = r(s, a) + \gamma \max Q(\delta(s,\ a),\ a')$$

The recursive nature of the function above implies that our agent doesn't know the actual Q function. It only estimates Q, which we refer to as Q^. It represents the hypothesis Q^ as a large table that attributes each pair Q(s, a) to a value for Q^(s, a). The γ is the discount factor, 1 implies long-term, 0 implies greedy decisions.

The agent could over commit to actions that presented positive Q values early in the simulation, failing to explore other actions that could present even higher values. Mitchell [6] proposed a **probabilistic approach** to select actions, assigning higher probabilities to action with high Q^ values, but given to every action at least a nonzero probability. So, we implement the following relation:

$$P(a_i \mid s) = \frac{k^{\hat{Q}(s,a_i)}}{\sum_j k^{\hat{Q}(s,a_j)}}$$

Where $P(a_i\, | \,s)$ is the probability of selecting the action $a_i$ given the state $s$. The constant $k$ is positive and determines how strongly the selection favors action with high $\hat{Q}$ values.

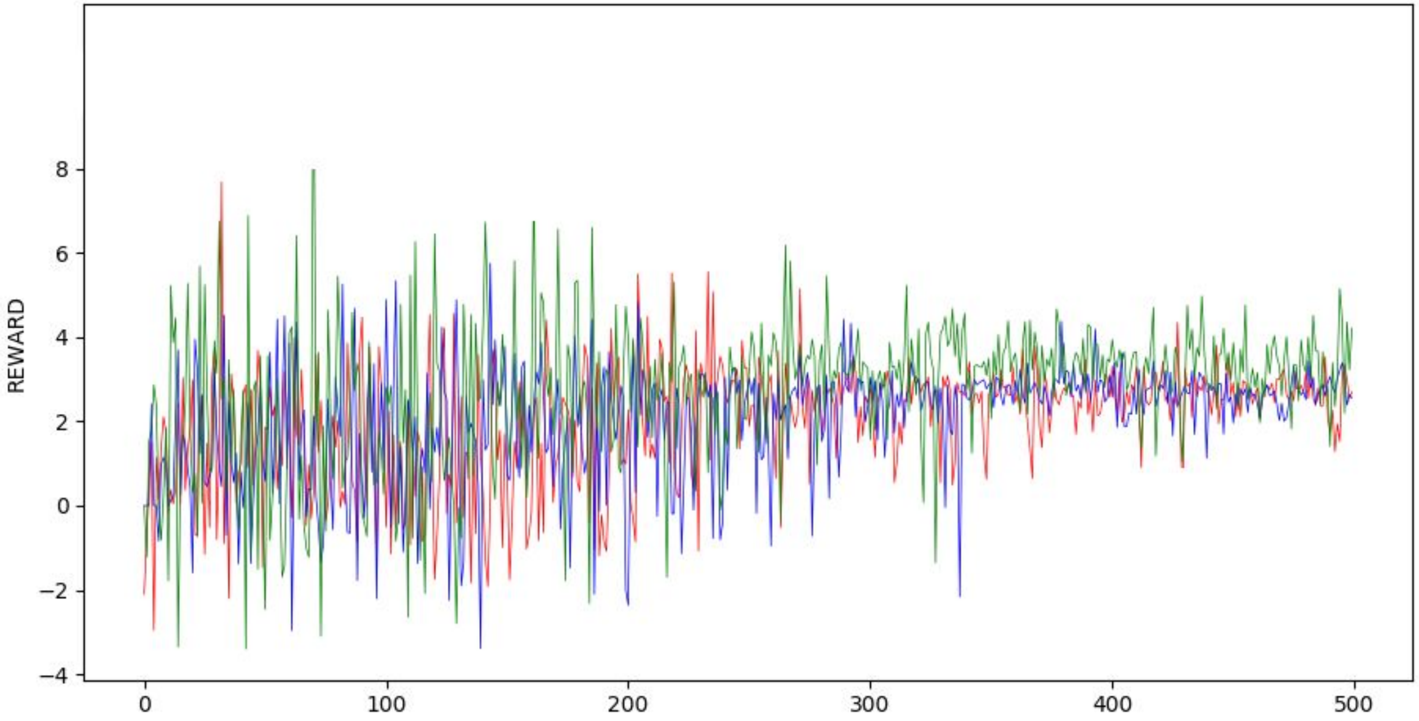
Imperial College
London

Dyna-Q

Even after discretization the problem space, **convergence was extremely slow**. Dynamic Q essentially creates another table that counts the number of times each state is reached after an action in a given state. All values are initialized to 0.00001 to avoid division by 0 errors. Dyna-Q then updates a state's expected reward as weighted sum of reward in each s' weighted by their relative probability.

$$\frac{T_C[S, a, s']}{\sum_i T_C[s, a, i]}$$

where the numerator is the number of times s,a leads to each s'. The denominator is the sum of T_c[s,a,:] (normalizes each s' to its probability)



Notice how the exploration is wide and low in reward. As Q converges, it explores less, and arrives at a higher reward.

Imperial College London

Q-learning Results

Discount Factor	Signals	MCC	SR
0.9	890	0.5125	0.5663
0.6	921	0.5476	0.5723
0.3	1246	0.4639	0.5583
0.1	1321	0.4356	0.2458
0.6 Ensemble	221	0.4923	0.5484
Benchmark	N/A	N/A	0.5591

Notice that there is a slight increase in signals as discount factor decreased. This means that search deep into the past has a slight effect in smoothing out noise. However, results were very **inconsistent** due to the **probabilistic exploration**. It appears that even our **ensemble could not help** Q beat the benchmark, despite almost cutting its signals down by 5 fold.

Possible Reasons for Failure

t statistic of -1.93 suggests that we can not reject the null hypothesis with 95% confidence that the Q-learner does not beat the random agent in risk-adjusted return.. This is the **only instance** in which a **learner cannot beat** the random agent.

Markov chains are designed to work on a discrete set of possible states. While price history can be discretized by segmenting the price range in a predetermined fashion, this might have greatly **reduces the potential precision** of the model.

The memoryless-ness of the MDP means that they work well if the only state influencing future states is the current one. This may not be the case with stocks, as there may be many variable term dependencies and patterns in the market fluctuations that Markov chains could simply not pick up on due to **lack of memory**. This foreshadows of our next method.

Imperial College London

Why LSTM?

- In an LSTM network, the **neurons** in an RNN are instead replaced with memory "**cells**", which consist of an internal memory state and several "gates"

- **Gates are similar to neural network layers**, learning a set of weights to determine what to forget (the "forget gate"), what to update the internal state with (the "input gate") and what to output (the "output gate").

- The cell always remember the information (**hence memory**) while the gates can sample from closer or farther data points in a time series.

- Memory might allow LSTM to predict future asset prices by understanding **short and long term dependencies**.

Imperial College London

LSTM Preprocessing

Normalize using MinMaxScaler, scaling to between 0 and 1.

$$z_i = \frac{x_i - \min X}{\max X - \min X}$$

Exponential smoothing is performed to reduce noise.

$$X_{t+1} = \frac{1}{N} \sum_{i=t-N}^{t} X_i$$

where z is the transformed variable. x_i is the variable to be transformed. X = a vector with all x that is transformed

Imperial College London

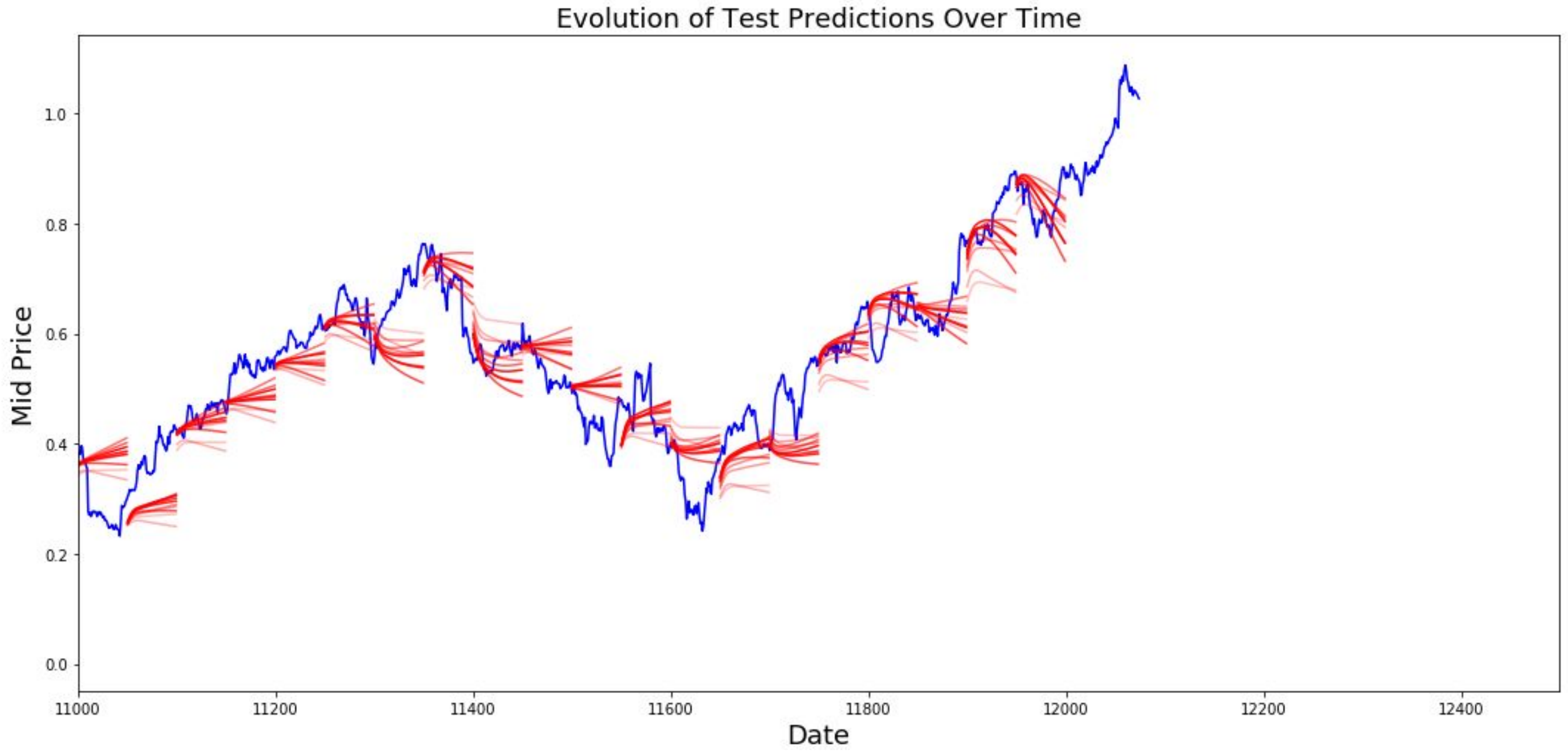
LSTM Architecture

- 3 layers with 1st as linear regression layer (denoted by w and b), that takes the output of the last LSTM cell and output the prediction for the next time step.

- **D:** The dimensionality of input or 1 since we matrix-ed our data.
- **NumLayer:** Searched between 3 to 5. More layers did not improve Sharpe Ratio, but were much slower to train. l
- **NumUnrolling**: Instead of optimizing the LSTM by looking at a single time step, we can optimize the network by backpropagating through time (BPTT) NumUnrolling number of steps. Higher number means we look farther into the past. Practically, the number of arrays sent from input. Optimized to 50.
- **BatchSize**: How many data samples per single time steps. Practically, the number of values per array sent from the input. Optimized to 500.
- **Dropout**: Optimized to 0.1 to reduce overfitting and improve out-of-sample training result
- **NumNodes**: Represents the number of hidden neurons in each LSTM cell. Optimized to 200,200,150 for each layer respectively.
- **Alpha**: Learning Rate optimized by Adam
- **Alpha~Decay**: Learning Rate Decay optimized by Adam

Imperial College London

LSTM Results



Blue is normalized price, Red is predicted trajectory. Close is better

Imperial College London

LSTM Results

Parameter	Signals	MCC	SR
30 Epoch Average	724	0.6274	0.6163
30 Epoch Ensemble	122	0.6423	**0.6779**
Benchmark	N/A	N/A	**0.5591**

A t-stat of 6.71 and p-value of < 0.05 suggests that we can reject the null hypothesis with 95\% confidence that the Q-learner does not statistically outperform the random agent. Therefore, our results are statistically significant.

Imperial College London

LSTM Results

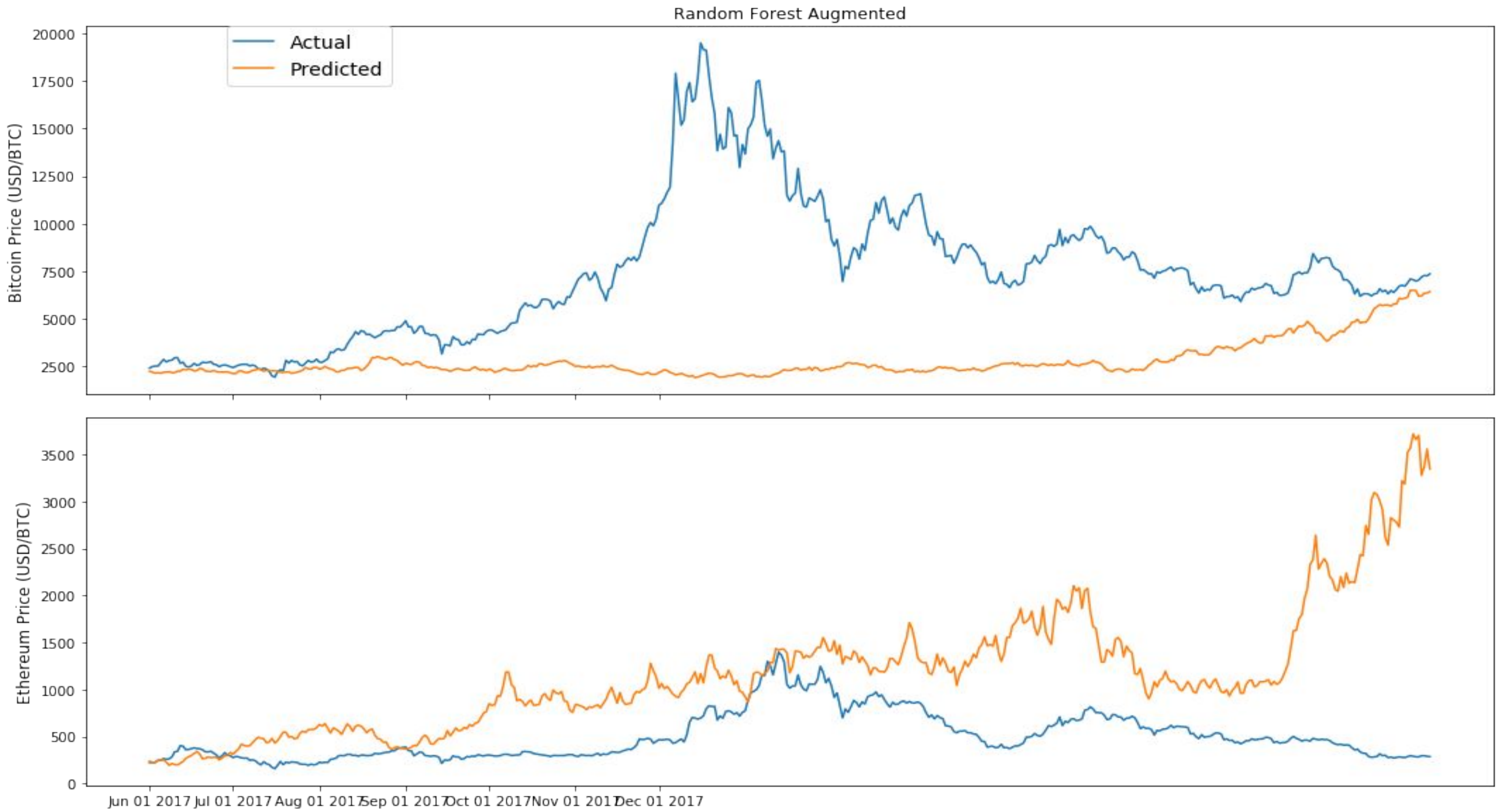
Confusion	Actual Buys	Actual Sells
Predicted Buys	41	28
Predicted Sells	15	38

Accuracy	Precision	Recall	F1
0.65	**0.59**	0.73	0.66

A high precision (classifying buys correctly) implies our model predicts buys better than sells. This makes sense since it is very difficult to predict sudden changes in the underlying stock price.

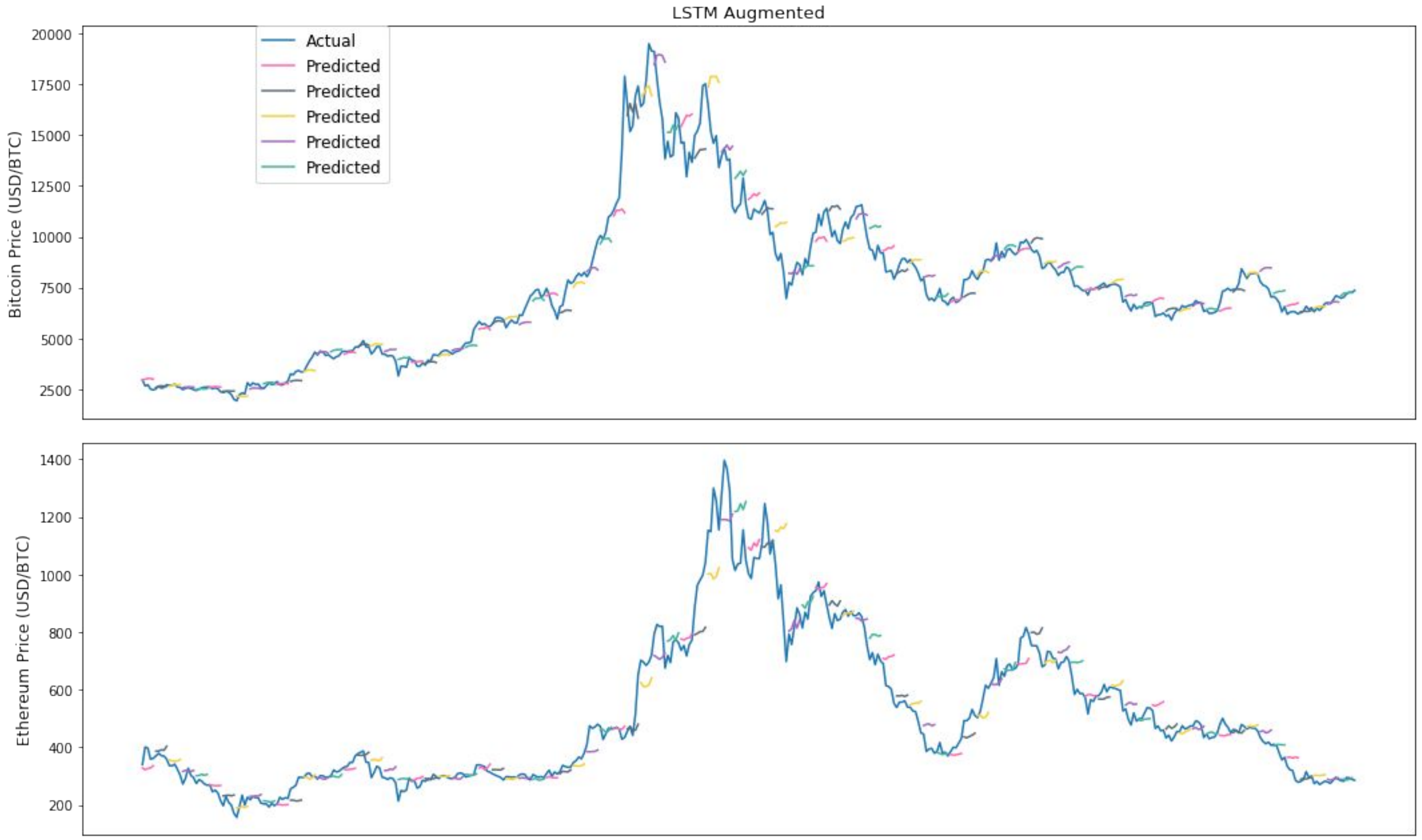
Imperial College London

RF Ensemble (BTC ETH)



Imperial College London

LSTM Ensemble (BTC ETH)



Imperial College London

BTC ETH

Indicator	Signals	MCC	SR
RF BTC	953	0.3835	1.1563
Ensemble RF ETH	465	0.4625	1.2573
LSTM BTC	652	0.5234	1.3687
Ensemble LSTM ETH	**315**	**0.5623**	**1.4234**
Benchmark BTC			1.6581
Benchmark ETH			**1.2631**

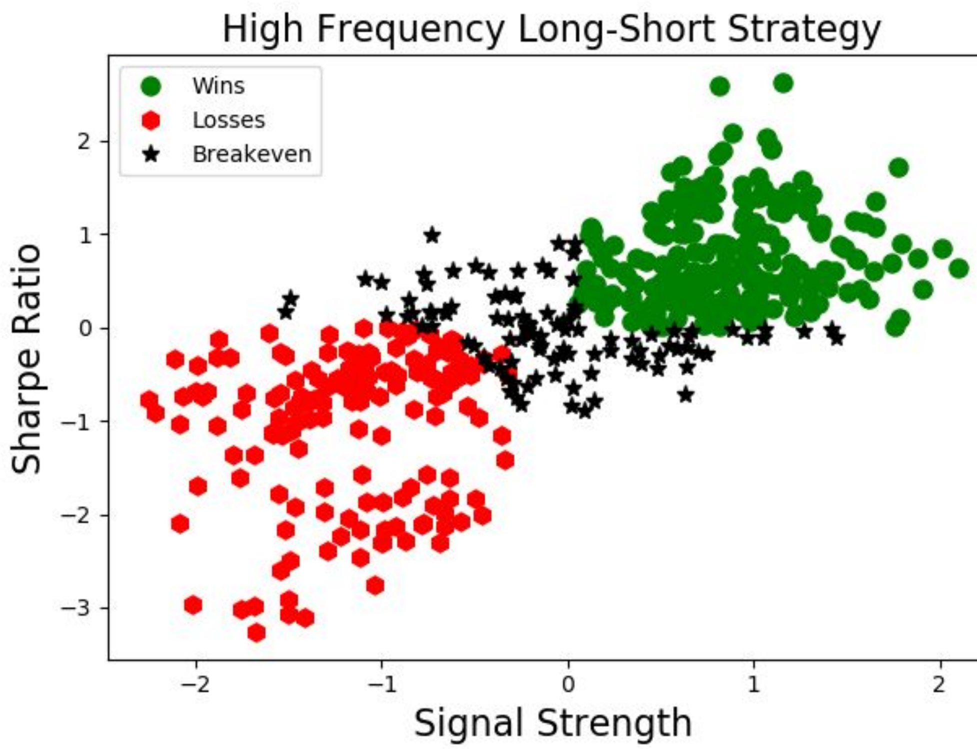
Imperial College London

BTC ETH Discussion

- Ensemble **Random Forest** performs **poorly** for both **Bitcoin and Ethereum**.
- **LSTM** fared much **better**, especially with **Ethereum**.
- This may indicate that **greedy algorithms** that depend on next day returns **may not do** so well.
- This may also i**mply a week correlation between next-day returns** in the data.
- This is a strong indication that this model cannot learn the underlying patterns in Bitcoin.
- The Bitcoin underperformance may be from the high number of **hyperparameters** that remain **unoptimized** in LSTM.
- Further investigation is warranted given the **correlation of the two assets**
- Additionally, the **lack of scaling in random forest** may have contributed greatly to its underperformance given the huge **absolute price swings** experienced by Bitcoin.

Imperial College London

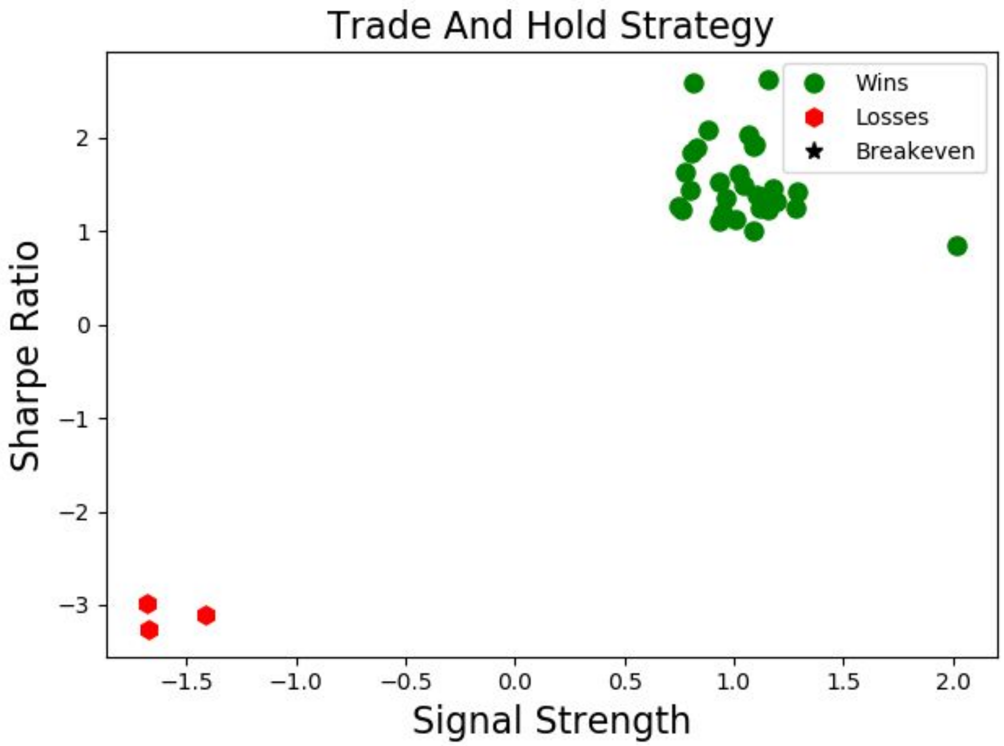
Meta Analysis



High frequency trades tend to use signals with low strength. Strength is the consensus measure among all signals for this trade, similar to entropy

Imperial College London

Trade and Hold



Low Frequency trades tend to use signals with high strength. Strength is the consensus measure among all signals for this trade, similar to entropy

Imperial College London

Future Work

1. Parse the data to uncover, how much of our model's performance derived **from machine learning**, and how much was **from eliminating noisy signals**.

2. **Optimize** the model for other applications like **cryptos** and **individual stocks**

3. Adding **more features** to take full advantage of LSTM neural nets.

- **Economic Data**: GDP, Unemployment, Wages, Manufacturing Index, Housing Starts
- **Market-specific** Data: order Book, Trading, Equity Research Recommendations
- **Sentiment Data**: Twitter, Reddit, Citigroup Global Surprise Index, and Investor Intelligence Newsletter (contrary indicator)

Imperial College London

References

[1] moneychimp. The Sharpe Ratio. URL: http://www.moneychimp.com/articles/ risk/sharpe_ratio.htm (visited on 08/28/2018)

[2] Rohan. Confusion matrix. URL: https://alearningaday.com/2016/09/14/ confusion-matrix/ (visited on 08/31/2018)
[3] L Breiman. “Random Forests”. In: Machine Learning 45.5 (2001), pp. 5–32

[4] Volodymyr Mnih et al. “Playing Atari with Deep Reinforcement Learning”. In: CoRR abs/1312.5602 (2013). URL: http://arxiv.org/abs/1312.5602

[5] Nicholas Tung Chan and Christian Shelton. “An electronic market-maker”. In: AI-MEMO-2001-005 (2001)

[6] Thomas M. Mitchell. Machine Learning. McGraw-Hill, Inc, 1997

Imperial College London

20 Minutes Later...

You: Eh, backtest results…
You: Put your money where your mouth is.

Me: I sure did.

(Now you also know the real motivation behind this thesis)