A Model-Free Reinforcement Learning Trading Agent

Udacity Machine Learning Nano Degree Capstone Project Report

Sebastian Janisch

March 14, 2017

Summary

Quantitative trading strategies make up a large part of financial investing. Over decades trading models have been developed that seek to find market inefficiencies by evaluating an ever growing amount of financial data.

Machine learning techniques offer a new suite of tools to tackle the challenge of generating superior investment returns. This project aims to outline the usage of a **model-free** reinforcement learning technique called *Q-Learning* to build a self learning trading agent based on quantiative input data.

1 Domain Background

In quantitative asset management any available information that can be quantified is used to uncover potential market opportunities. These opportunities are typically caused by mispricing, where a stock's price is not reflective of its true (intrinsic) value. The *Efficient Market Hypothesis* (henceforth referred to as *EMH*) suggests that a "market in which prices always 'fully' reflect available information is called 'efficient'"[1] indicating that in such efficient markets opportunities due to mispricing don't exist. Empirical analysis suggests however that this theory can, at least in part, be refuted[2], opening the playing field for a broad array of quantitative approaches to find said market inefficiencies.

Theories such as the *EMH* and the *Random Walk Hypothesis*[3] call into question the right of existence for active portfolio management (the discipline that seeks to outperform against a chosen benchmark by uncovering market inefficiencies). Whilst the jury on the matter is still out there is an understanding that said market inefficiencies are difficult to expose[4]. Finding yet another clever way of generating alpha (i.e. return greater than that of the market benchmark) is the problem that quantitative active portfolio management tries to solve.

Much has happened in the machine learning space in recent years. As a discipline, machine learning techniques lend themselves well to problems such as finding trading strategies that outperform the market.

2 Problem Statement

The assumption of all active asset management is that markets are not always efficient and can be outperformed (i.e. risk adjusted outperformance for given benchmark). Much of traditonal quantitative asset management uses a model based approach, where investment ideas are cast into a model which generates trading signals[4]. This implies that a concrete statement is made as to whether or not a given piece of quantitative information is positive or negative (e.g. high corporate leverage indicates poor financial health and possibility of financial distress). The interpreation of a given quantitive metric is however not always so clear (e.g. some industries are characterised by higher leverage without posing a risk of financial distress). The system proposed in this project sets forward an alternative model-free reinforcement learning technique approach using Q-Learning, where no statement is made as to the interpretation of a quantitative metric but it is rather left to the system to learn it autonomously.

3 Datasets and Inputs

Fortunately in the finance domain quantiative data is readily available and abundant. It is understood that more complete, timely and extensive data can be obtained from vendors at cost, however, for the purpose of this project data readily available and free data sources shall suffice.

One of the most crucial information about a traded stock is surely its stock price. As pointed out by [2] the *EMH* can be strongly refuted for daily stock prices, evidence for longer frequencies is thinner though. It is hence that for this project daily asset returns will be used as a primary data input.

Based on the daily (adjusted, i.e. corrected for stock splits and dividend payments) stock price alone alongside with daily trading volumes a whole field has emerged engaging in the discipline referred to as Technical Analysis [5]. Using metrics that are essentially a function of stock price together with trading volume a whole battery of different metrics is created that is thought to help find market inefficiencies (e.g. Momentum, Bollinger Bands, etc.).

There is much controversy as to whether or not plain technical analysis is truly a strong predictor of future stock prices. Enriching the data set with information other than the stock price alone can add more insight into the details of a company behind a stock and as such yield a more predictive set of information. Fundamental quantitative data derived from financial statements (e.g. profit margin, dividend payout ratio, etc.) is hence used as a second set of input data into the learning algorithm.

The data source used will be the Quandl API to retrieve stock prices and financial statement data for given stocks. To access the necessary data feeds the below Quandl databases will be used.

- End of Day US Stock Prices (Quandl Code *EOD*) - to retrieve stock prices for the US

equity market - historical data up to 1996.

- Core US Fundamentals Data (Quandl Code *SF1*) - to retrieve point in time fundamental metrics for the US equity market - historical data up to 2005.

Given that the earliest available date of both data inputs is the year 2005 this gives 11 years of data to base the system on.

Data Retrieval Example

Assuming Alphabet Inc. was a US equity stock covered in our investment universe and we wanted to obtain daily stock prices and also the profit margin (expressed as the ratio of net income and gross revenue) of the firm to include in our model, then the Quandl database would be utilised for the below dataset codes.

- *EOD/GOOGL* - prices, dividends, splits for Alphabet Inc. (GOOGL).
- *SF1/GOOGL_NETINC_MRQ* - net income, the portion of profit or loss for the period for Alphabet Inc. (GOOGL).
- *SF1/GOOGL_REVENUE_ARQ* - revenues, amount of revenue recognized from goods sold, services rendered, insurance premiums, or other activities that constitute an earning process.

Data Sample

Below table in figure 1 shows a sample of the raw input data. Specifically, the $PRICE$ and $NETMARGIN\_ART$ (net profit margin, rolling twelve months) data items are given for six trading days for the $AAPL$ (Apple Inc.). What is noteworthy is that for the net profit margin there is only one actual data point present in the given sample. This is due to the slow moving nature of corporate fundamental data which is published on a quarterly, semi-annual and annual basis. Here it also becomes apparent that this data is best adjusted by fill forwarding it, since a published fundamental data point is valid until the next data point is published in the future.

4 Solution Statement

Given the above data source appropriate quantiative metrics will be selected to form the state space for the Q-Learning algorithm[8]. Generally, Q-Learning is a model-free reinforcement learning algorithm which allows to realise delayed rewards and find optimal policies for selecting an action out of a universe of possible actions (e.g. buy vs. sell). This notion of a

Date	NETMARGIN_ART	PRICE
2017-01-27	–	121.42
2017-01-30	–	121.10
2017-01-31	–	120.83
2017-02-01	0.21	128.19
2017-02-02	–	127.98
2017-02-03	–	128.52

Figure 1: Data Sample

'delayed reward' is much in contrast to traditional quantitative methods of investing where models are based on an ex-ante prediction of future asset prices.

More formally, the Q-Learning algorithm is based on a finite Markov Decision Process and maximises total reward. A reward is achieved by transitioning from a given state S_t to S_{t+1} by taking an action a_t. By specifying an appropriate reward function that rewards desired and penalises undesired behaviour the algorithm will learn a policy of behaviour. The speed of learning can be calibrated using an α parameter.

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right) \tag{1}$$

In this project we take a no control perspective where the trading agent is acting as a small market participant with no market influencing capabilities (see below). As such this problem can not be represented as a Markov process since no state transitions can be achieved. This however does not impact the Q-Learning process per se as it can still be represented as a chain of actions and rewards.

4.1 Simplifications and Assumptions

This project does not make any claims of producing a real world trading agent, recognising that there are many considerations and real world complications to take into account which are out of the scope of this project.

- **No trading costs:** trading costs are an important aspect to consider in real life investing as such cost can make up a significant part of the maintenance cost of a portfolio for high turnover portfolios or illiquid assets. In this project we simplify and ignore trading costs.
- **Perfectly divisible shares:** this project uses a mean variance optimisation to find

optimal holding weights which are computed in continuous space. In the real world a fraction might not easily be tradeable. This restriction is relaxed.

- **No control perspective:** the agent acting in this space takes the perspective of a minor market participant with no pricing influence or market movement capabilities. This assumptions is most closely related to the real world where a financial market is made up of a large number of individual participants, each individually only comprising marginal importance to the overall market. This has implications for the Q-learner in as much as there are no actual state transitions modeled (as the agent can't influence the state of the environment).

- **No shorting:** the portfolio will never over invest itself or use shorting strategies. On the other hand the portfolio might be fully invested in cash if the agent decides that no stocks at all should be held. This is a convenient simplification so as to not force the learner to buy at least one asset but is is to be noted that this is a departure from real world requirements of fully invested portfolios (an active manager is not paid to hold cash).

State Space

The state space will be formulated based on the below two premises.

1. provide sufficiently large value range covered by the state space to discriminate well over different states of the system.

2. do not overly bloat up the state space to prevent causing high dimensionality issues (referred to as the *Curse of Dimensionality* [6]).

In addition to the traditional price derived metrics we are going to pull various fundamental data items which will be used to gain insight into an asset's profitability, financial health, etc.

Asset Universe

As a reference universe of assets we will use the DOW Jones Industrial Average (DJIA) constituents. There are primarily two reasons for this choice:

1. the DJIA is made up of 30 assets which is a small enough number to not overly complicate this project and introduce excess computational complexity (e.g. by using the S&P500) yet is made up of enough assets to achieve portfolio diversification effects (see chart below)][9].

2. The DJIA is a well understood price-weighted index and performance data is readily available for comparison purposes.

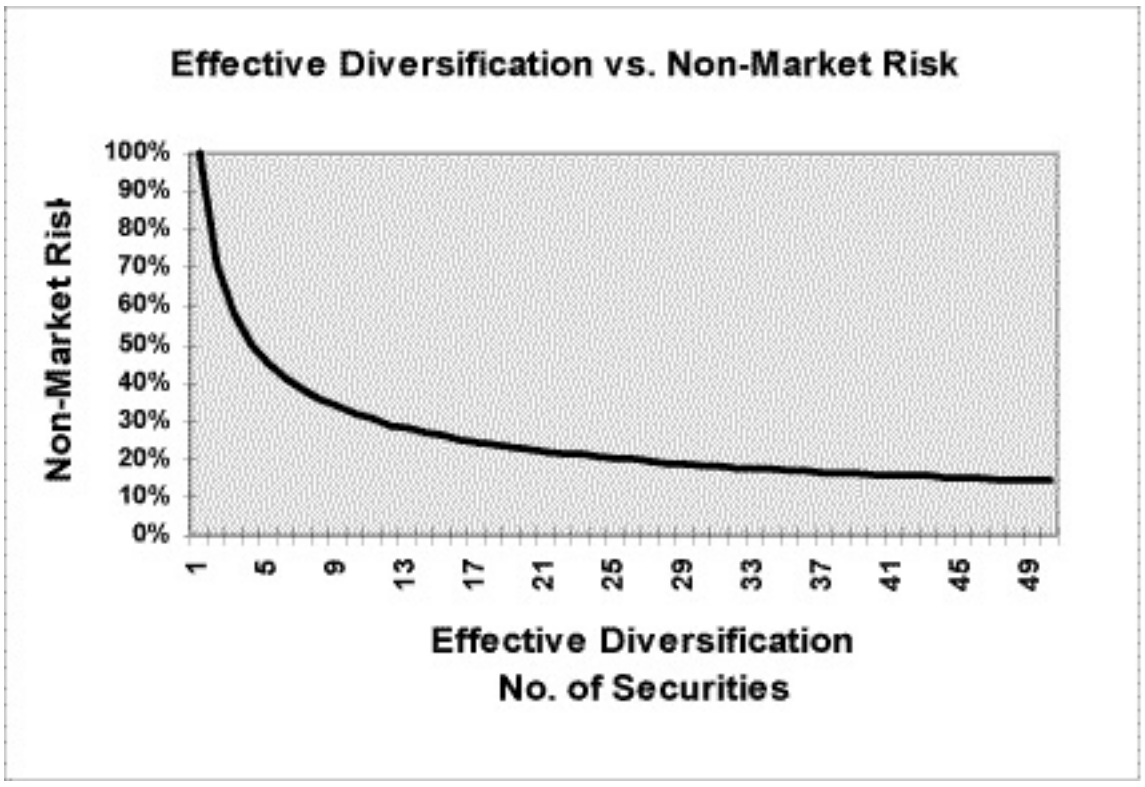

Figure 2: Diversification Effects

Fundamental Data Items

Financial statement data is mostly expressed in monetary units (e.g. a US based company's revenues are expressed in terms of US dollars). Different companies vary in size and scale hence it is not appropriate to compare financial statement data on a dollar to dollar basis.

Financial statement ratios provide a common scale for comparison (e.g. a company's profit ratio is a percent based figure, irrespective of size and scale). Financial statement ratios can be broadly categorised into four (note that different sources define these categories differently) different themes. Each of these can be computed in different ways (described later).

1. **Profitability Ratios:** a company's profitability relative to some other measure (e.g. net income compared to revenue).
2. **Liquidity Ratios:** a company's ability to pay off its ongoing liabilities and expenditures (interest payments, project investments, etc.).
3. **Leverage Ratios:** a company's ability to sustain ongoing operations using its mix of debt and assets.
4. **Efficiency Ratios:** a company's ability to make efficient use of its capital and assets.

Each of these broad categories can be expressed in different ways. A company's liquidity could be measured by looking at its cash reserves or its debt levels (the less debt the less interest a company needs to pay which in turn means the company is more liquid). In a real world scenario each of these themes would be expressed using a variety of different metrics which would be standardised to a common scale and subsequently be combined into a weighted average. This weighted average is understood to capture the different characteristics of the enclosing category and condense them into one figure. For the purposes of this project a single representative metric will be used per category.

Financial statement data is published in different frequencies, usually quarterly, semi-annual and annual. Annual statements are undergoing a regulatory audit process whereas quarterly and semi-annual data is not necessarily audited. Thus the assumption is made that annual data is more reliable and accurate and annual data will be used. This again is a departure from a real world scenario where different filtering and preference logic in line with availability of data would be employed.

Profitability Ratio: As a measure of profitability the profit margin is chosen which is computed as the firm's *net income / net sales*. This prominent figure takes into account the costs a firm incurs for producing its goods and services and provides a useful indication of overall profitability.

Liquidity Ratio: Interest coverage is a metric that represents a company's ability to pay

Data Item	Description	Data Points	Mean	σ	Min	Max
PRICE	Daily Asset Price	2085	61.82	10.99	5.14	252.89
NETMARGIN_ART	Net Profit Margin	2085	13.93%	2.73%	-0.05%	48%
INTEREST_COVERAGE_ART	Interest Coverage	2085	0.08	0.08	-0.01	1.92
LEVERAGE_ART	Leverage Ratio	2085	4.30	6.89	-96.12	200
ASSETTURNOVER_ART	Asset Turnover	2085	0.75	0.08	0.02	2.45
MOMENTUM	Momentum	2085	1.00	0.008	0.7	1.49

Figure 3: Descriptive Statistics

off its ongoing interest expenses and is computed as *ebit / interest expense*. As such this figures is read as 'how many times could the company pay off its annual interest obligations'.

Leverage Ratio: The most popular figure to express a company's leverage is the debt asset ratio, calculated as *average assets / average equity*. This ratio is chosen to represent this category.

Efficiency Ratio: The asset turnover is a financial ratio that describes the ability of a company to turn its assets into revenue and is calculated as *net sales / average assets*.

Price Based Data Items

As price based input data we chose a basic mometum indicator over one trading week, defined as $price_t$ / $price_{t-5}$.

Data Adjustments

In this section data items are adjusted to prevent issues such as division by zero or other extreme case.

- interest coverage is prone to a division by zero problem if a company does not have any debt outstanding. Without loss of information we can take the inverse of the interest coverage (i.e. 1 / *interest coverage* = *interest expense / ebit*) to achieve a metric that is always defined (unless in the unrealistic event where ebit is zero).

Descriptive Statistics

Figure 3 shows basic descriptive statistics of the raw input data set before standardisation and discretisation. Note that ART refers to trailing twelve months. σ signifies the standard deviation and note that the interest covereage values are the inverse of the actual interest coverage figure.

Discretisation

The input data needs to be discretised to allow for a useful state space. The discretisation will be performed by applying the below two transformations to the data.

For each date and each data item:

1. form a z-score for data item I and asset i defined as $z_{I,i} = \frac{x_i - \hat{x}}{\sigma_I}$. This will transform the value of an asset into a continuous standardised score with mean 0 and standard deviation of 1.

2. discretise the z-score into three different buckets.

Here three buckets were chosen to provide a sufficiently small amount of possible states inside the state space whilst still providing information where an asset's value places with respect to all other assets (near the mean, above the mean or below the mean).

Specifacally, using above definition of a state space using the four different fundamental items and momentum indicator we form a state space that can take 243 different states ($3^5 = 243$).

Risk Adjusted Returns and the Q Reward Function

The reward function will be based on the **risk adjusted** return generated for a given decision the agent took. In active portfolio construction there are many techniques being employed. The premise is mostly to maximise the expected return of a set of assets under a set of constraints. These constraints are typically (but not restricted to) risk constraints and transaction cost constraints, which can be expressed with below figure. [4].

$$\alpha_P - \lambda_A \cdot \Psi_P^2 - TC \tag{2}$$

Here, α represents the return of a portfolio of stocks in excess of its benchmark, Ψ represents some representation of active risk (i.e. risk in excess of the benchmark), λ represents the penalty scalar for taking on active risk and TC represents the transaction costs incurred by the strategy.

Taking into consideration the transaction costs is understood to be a fundamental corner stone of any trading strategy (especially important when investing in low liquidity assets). Since this project however does not make any claim on presenting a realistic real world implementation of a trading strategy but merely aims to outline a more novel way of utilising machine learning techniques transaction costs will be disregarded.

The point to make is that it is easy to achieve high returns by taking excessive active risk. Devising a reward function that is based on pure return would incentivise the Q-Learning algorithm to take risky bets, a behaviour we want to control. By correcting for risk in the

reward function we can achieve a similar effect compared to the characeristics of formula 2.

A suitable candidate for a reward function is the *Sharpe Ratio*, the industry standard for calculating risk adjusted return. More formally, the *Sharpe Ratio* is outlined below.

$$S_i = \frac{r_i - r_f}{\sigma_i} \tag{3}$$

Here r_i is the return of asset i, r_f is the risk free rate and σ_i is the standard deviation of the asset's return.

The risk free rate will be assumed to be 0 which for one thing would not be a large departure from current market circumstaces and for another thing does not impact the efficacy of the Q-Learner.

5 Benchmark Model

To evaluate the performance of the actions carried out by the agent will be benchmarked against the return of a representative stock index. Given that the learner for the purpose of this project will be based on the Dow Jones Industrial Average index this will also be used as a benchmark.

As return figures of active portfolio and benchmark the cumulative return will be used. If we start measuring performance starting at time $t = 0$ and ending at time $t = T$ then we compute the cumulative return for any given portfolio as below.

$$R_c = \prod_{t=1}^{T} \left[\frac{P_t}{P_{t-1}} \right] - 1 \tag{4}$$

Here P_t is the price at time t (or current value) of the portfolio we are looking at (i.e. benchmark or active portfolio).

6 Evaluation Metrics

To ensure the outcome of the learner is compared apples to apples a risk adjusted measure will be used to quantify the risk adjusted return of both benchmark index and our stock portfolio. The *Sharpe Ratio* (see above) will be used as the metric of choice to compare portfolio performances.

As mentioned above, it is understood that a more realistic system would incorporate transaction costs into the evaluation. For simplicity however this project disregards transaction costs.

7 The Trading Agent

The trading agent encapsulates the underlying learning system. It sets on top of the data space defined earlier and has the following responsibilities:

- train the learner without actually acting and investing.
- use the recommendations of the learner to form a holding portfolio which's weights are recorded in time to later produce performance figures.

For each of the two items above the agent needs to be able to do the below as it steps through time:

- take the raw input data at time t and compute the variables that form the state space.
  - standardise variables (zscore)
  - discretise variables (bucketing)
- for each asset at time t obtain the recommended action from the Q-learner (either Buy or Sell).
- obtain holding weights for assets at time t by executing a minimum variance optimisation.
  - only consider those assets recommended as Buy.
  - compute asset returns for given assets up to time t (using the 'PRICE' data item).
  - compute covariance matrix for asset returns.
  - use covariance matrix to compute minimum variance portfolio[9].
  - rebalance portfolio at time t with given optimal weights (not in learning mode).
- compute reward and update the Q-learner.
  - portfolios are rebalanced at a given frequency (see below). The reward is based on the sharpe ratio of the bought or sold asset over the chosen frequency. E.g. if the learner recommended to buy an asset and the asset has been performing well with a low standard deviation the reward would be high. Holding all else constant but the asset would have been more volatile then the reward would be lower. Given the learner recommended to sell an asset and the asset actually declined in value the reward would be high (i.e. the negative sharpe ratio is used if the learner recommended a sell).

Random Portfolio

In addition to the real learning portfolio we will trade a random portfolio to compare the results to the benchmark and actual Q-Learner portfolio.

7.1 Training the Agent

To prevent starting off a portfolio with completely uninformed trades and bets the system is pre-trained by stepping through a predefined number of periods and simulate a portfolio without actually recording the performance (i.e. holdings are computed and simualted as but the results of those holdings won't enter the final performance comparison).

In addition we specify a holding period which indicates in which intervals assets are re-assessed and and the portfolio is rebalanced. In this example we choose one week as a portfolio rabalance frequency.

7.2 Trading the Portfolio

After initially training the agent the actual portfolio is traded going forward and the Q-Learner learns as it moves through time. The steps involved are identical to the learning above with the exception that the portfolio weights are now recorded through time and evaluated later.

8 Result Evaluation

Following the trading period the performance and other analytics of the agent can be evaluated.

8.1 Cumulative Return of Portfolio, Benchmark and Random Portfolio

The most important performance measure is the realised return the portfolio achieved compared to the benchmark.

The benchmark portfolio values for the Dow Jones can be directly observed and simply need to be cumulated. The returns of our portfolio will be calculated using the weights obtained by the trading agent during the trading phase.

Figure 4 shows that over the trading period between 2010 and 2017 the portfolio achieved a better cumulative return compared to the benchmark and to the random portfolio.

8.2 Portfolio Risk

The portfolio risk is approximated by using a rolling standard deviation of both portfolio and benchmark.

The chart in figure 5 shows that the portfolio assumed lower levels of risk compared to

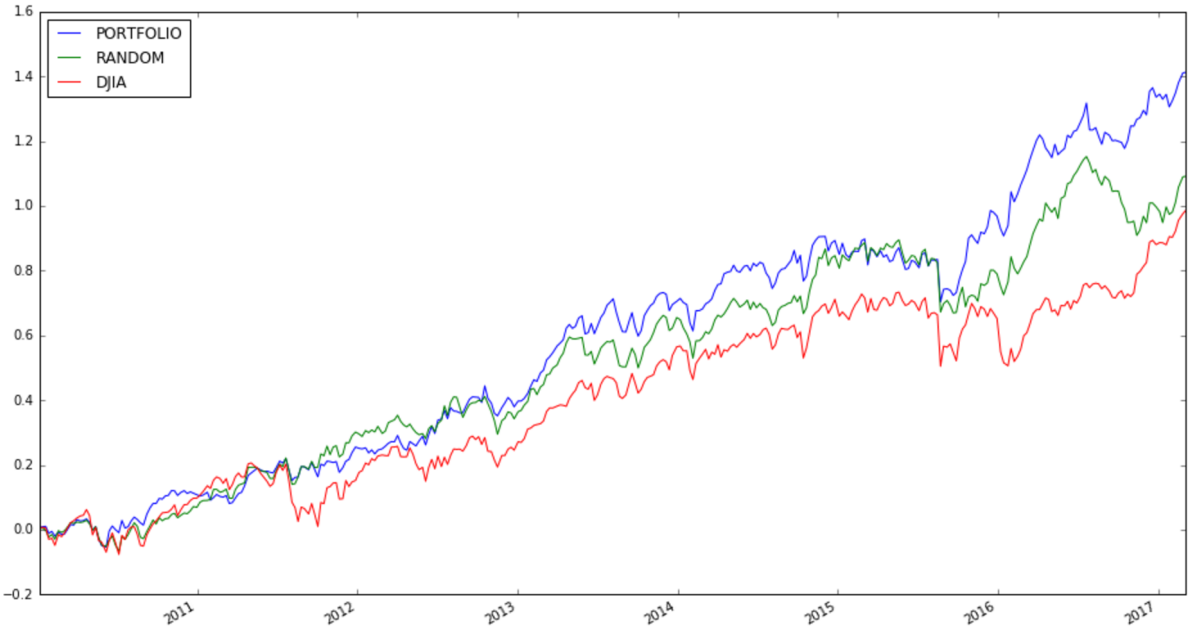

Figure 4: Portfolio Returns

the benchmark and the random portfolio. This is not surprising as the reward function incorporated the level of risk assumed by the trading behaviour (the Sharpe Ratio was the basis for the reward function).

8.3 Portfolio Sharpe Ratio

Finally the risk adjusted measure of return (the Sharpe Ratio) is computed for the active portfolio, the benchmark as well as the random portfolio.

The overall Sharpe Ratio for portfolio, benchmark and random portfolio gives an indication of the risk adjusted return achieved over the trading period. We find that the portfolio could achieve a higher Sharpe Ratio compared to the benchmark and the random portfolio.

$$S_{Portfolio} = 94.31 \tag{5}$$
$$S_{Benchmark} = 51.44 \tag{6}$$
$$S_{Random} = 68.93 \tag{7}$$

8.4 Model Parameters & Robustness

The model has been calibrated with the below parameters.

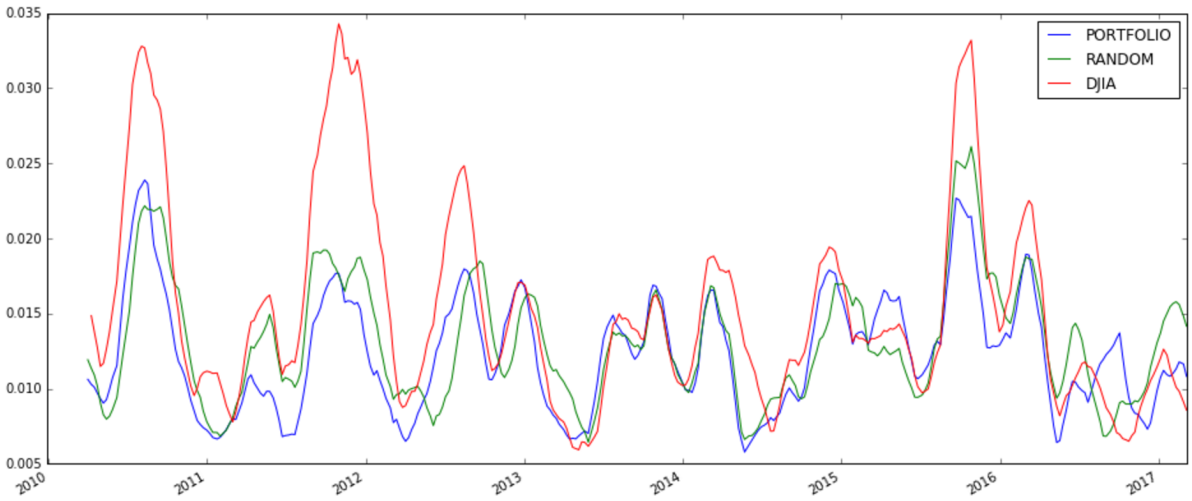

Figure 5: Portfolio Risk

- Rebalance Frequency: Weekly (5 days) - This frequency was chosen to somewhat stay within realistic territory in terms of transaction costs, even though in this model those are not explicitly measured. Rebalancing a portfolio on a daily basis would create excessive churn and transaction costs. Moreover asset prices fluctuate over the short term. It stands to argue what exactly can be defined as a short term fluctuation, however, using a weekly window seems a reasonable timeframe to provide a robust reward estimate as given by the reward function.
- Training Period: 1 Year (255 days) - The input data ranges from late 2008 to early 2017. Training the Q-Learner on 1 year is equivalent to using around 10% of the data for initial training purposes which is understood to be a reasonable number.

To test the robostness of the model the agent was also calibrated using a 10 day rebalancing window. The resulting portfolio, too, outperformed benchmark portfolio and random portfolio.

The trained model also does not outperform the benchmark by an excessive and unrealistic margin which further increases the confidence that the model is on reasnonable grounds.

9 Conclusion

9.1 Free-Form Visualisation

Figure 6 shows the number of assets the portfolio held through time, showing that on average 20 assets were held in the portfolio with low points of 9 assets and maximum of 29 assets.

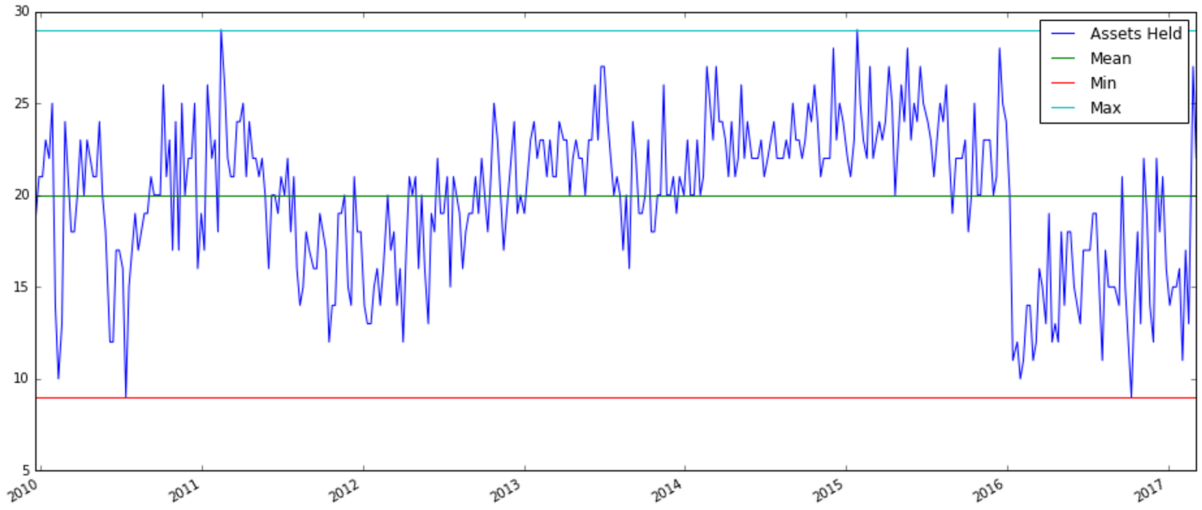

Figure 6: Assets Held

This visualisation confirms that the agent was holding a sufficiently large number of assets throughout to achieve portfolio diversification effects.

9.2 Reflection and Improvement

This capstone project has demonstrated an application of Q-Learning as a trading agent by creating a learner that makes buy and sell recommendations and creates minimum variance optimised portfolios through time. For that purpose price based as well as corporate fundamental data was used. The output demonstrated that the trading agent could achieve higher returns over a trading period of 8 years.

Of particular interest was the implementation of the Q-Learner and forming a relevant state-space, as well as evaluating the model results. It proved particaluary difficult to obtain and curate the input data from different sources to transform them into a suitable space for this problem.

For the purpose of this project several real life complications were relaxed and assumptions were made (see above). A more realistic version of this trading agent would one by one incorporate these complications and evaluate the behaviour of the agent.

Further the state space for the agent can be elaborated. As discussed above for each fundamental ratio category one representative ratio was chosen. This can be improved by utilising more than one ratio per category and combining these using a weighted average. Further technical indicators can be added as well.

Earlier the term diversification was mentioned. In this project the trading agent held 20 assets on avaerage but it was not explicitly designed to do so. To gain more confidence that the agent would not hold a number of assets that is too low to achieve basic diversification effects the model could be improved. One way of achieving this would be to incorporate the number of assets held in the reward function.

Lastly, statistics of the internal state of the agent could be summarised to gain insight into how well the data discriminates over the state space as the agent moves through time. Specifically, knowing how often a particular state is hit enables us to analyse whether or not the state is hit enough to provide a reasonable learning effect and accumulation of rewards. This could then be utilised as an indicator to either contract (if individual states are not hit often enough) or expand (if individual states are hit often enough) the state space.

References

[1] Eugene F. Fama, Efficient Capital Markets: A Review of Theory and Empirical Work, (1970).

[2] Martin Sewell, The Efficient Market Hypothesis: Empirical Evidence, International Journal of Statistics and Probability, Vol. 1, No. 2, (2012).

[3] Paul H. Cootner, The random character of stock market prices, M.I.T. Press, (1964).

[4] Richard C. Grinold and Ronald N. Kahn, Active Portfolio Management : A quantative approach for producing superior returns and selecting superior money managers, (1999).

[5] John J. Murphy, Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications, New York Institute of Finance, (1998).

[6] Raul Rojas, The Curse of Dimensionality, (2015).

[7] Keith C. Brown and Frank K. Reilly, Analysis of Investments and Management of Portfolios

[8] Richard Sutton, Reinforcement Learning: An Introduction, (1998).

[9] William Sharpe, Risk, Market Sensitivity and Diversification, Financial Analysts Journal, January/February 1972, pp. 74-79.