Data Driven Investment Strategy

Shrayansh Jyoti

November 1, 2022

1 MACHINE LEARNING INVESTMENT STRATEGIES

1.1 Introduction:

A machine learning model is an expression of an algorithm that combs through data to find patterns or make predictions. You train a model over a set of data, providing it an algorithm that it can use to reason over and learn from those data. Once you have trained the model, you can use it to reason over data that it hasn't seen before, and make predictions about those data. It can not only learn the complex logical relationship behind the data but also improve its performance in the process of repeated training.

Quantitative finance has a strong foundation in the use of mathematical models, theories, and proofs, essentially moving from abstraction to action. Machine learning takes the opposite approach of focusing on developing models that are based on the empirical data. If quantitative finance is generally seen as a top-down approach, ML offers a complementary bottom-up alternative.

1.2 Benefits:

The model-based approach achieves a better performance with fewer environmental interactions.The critical success factors for traders are access to information, accurate interpretation of that information, and speed in execution. Machine learning can handle vast data sets, global news, visual data, and more to seek out patterns that may be insightful and profitable when applied to trading. Further, it can be put to work in areas that are practically impenetrable for humans, including unstructured data, alternative data sets, and multi-factor analysis, for example. A prime example of this is the use of Natural Language processing for sentiment analysis of financial news which is a great way of analysing market sentiments to find alphas. The biggest advantage of ML is it's capacity to handle and process ginormous amounts of data, which would be incredibly difficult to do through traditional methods.

Another important application of Machine Learning is in Risk Management, particularly in fraud detection and prevention. Credit card fraud, for example, is a significant concern for financial services firms. Machine learning can help detect and flag fraudulent transactions and anomalies, first by learning and retaining a profile of typical consumer behavior and then comparing a current transaction in-process against the set of known data points, including geographical locations, IP addresses, account history, and other customer information. If an intended purchase or withdrawal seems suspicious, the system can decline to complete it, notify the institution and the customer, and wait for validation. Machine learning is also being applied to credit risk modeling, using traditional credit factors, including debt-to-income ratios, banking information, and credit card history. The holistic view of taking a deeper look at patterns of consumer behavior, enhances the predictive power of the models, extending from the individual case to the corporate case.

1.3 Risks associated:

A machine learning algorithm is only as good as the data it is trained on. The better the quality of the data and the larger the volume the lesser the variance in the results. Data with biases in it will also bias the model. For instance, the popular selfie editor FaceApp was initially inadvertently trained to make faces "hotter" by lightening the skin tone-a result of having been fed a much larger quantity of photos of people with lighter skin tones. ML strategies are particularly sensitive to the noise, bias, and missing values in the data and since, the signal to noise ratio in finance is quite low thus ML is quite error prone. Underfitting and Overfitting is also a common problem for ML models.

As compelling as the case is for ML, one must remember the dictum from Ancient Greece, "Moderation in all things." A better approach for most strategies would be to layer both a ML approach and a traditional mathematical approach. For example, an extensive dataset can be subdivided using ML and then worked on individually through traditional quantitative techniques.

1.4 Why has ML suddenly become so popular?

Although, ML techniques have existed for decades, they have only started to become popular in the last decade or so. The main reasons for ML to have moved beyond textbooks is the abundance of data and the availability of cheap computational power. As you must have heard this is the era of Big Data. There is data that is being collected now that has never been collected for individuals or at least at this scale before. Several organisations are exploring all of the ways of tracking and collecting everyday information about people from biological information like heart beats, breaths, steps, to interactions such as conversations and words spoken. Machine learning methods provide the capability to model complex problems using large volumes of seemingly disparate data, the only limitation being the computational power at one's disposal. With powerful computers at everyone's disposal, computation has truly been democratised and therefore, the usage of ML has sky rocketed.

2 DECISION TREE

2.1 Data Description:

We have a dataset with the monthly historical prices of several thousands of stocks from 1980 to the end of 2021. We have another dataset with monthly data from the start of 2008 to the end of 2021 for all the factors being considered, i.e., Gross Profitability, Low Volatility, 12 months Momentum, Quarterly EPS for value, US Federal Funds rate, and, the US unemployment rate. All the data has been normalised. One important feature of the price dataset is that, not all stocks have all the months of data available. For example, stock 10001, may have data from the start of 1980 to the end of 2021, while, stock 10025 may only have the data from May 1997 to Jan 2005. This will turn out to be extremely important later on.

2.2 Methodology:

We want to use decision tree classifier model to predict, for each stock, whether the returns for next month will be above or below the median returns of all the stocks for that month. We will train our model, using a maximum depth, which is the length of the longest path from a root to a leaf, of four, on data from the start of 2008 to the end of 2017. While the model will be tested on data from the start of 2018 to the end of 2021. To perform the modelling we must preprocess the data, i.e., we need to calculate the returns from the price data after which we group all the stocks by date and calculate the median for each month. We then compare, for each stock, whether the return for this month is greater than the median return of all stocks for the next month. This is our target variable. Since, decision trees are error prone for categorical data, we map True as 1 and False as 0, instead.

2.3 Results:

$$\begin{bmatrix} & 0 & 1 \\ 0 & TN & FP \\ 1 & FN & TP \end{bmatrix}$$

```
Confusion Matrix:
 [[35156  9607]
 [30783  8696]]

TPR = 0.22027
TNR = 0.78538

In-Sample Accuracy:
 0.537

Out-of-Sample Accuracy:
 0.521

10-Folds Cross Validation
Average Accuracy:
 0.536
```

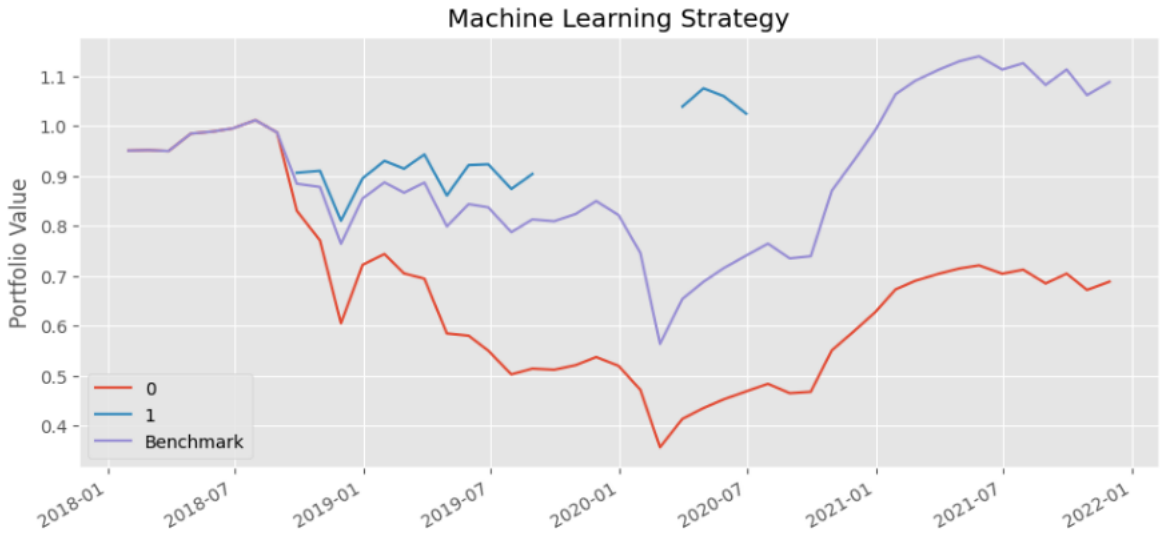
We'll start by interpreting the Confusion Matrix. For better understanding, we can represent the matrix as above . Where, TN is a True negative and FP is a False positive and so on. The horizontal axes represent actual values whereas the vertical axes represent predicted values. This matrix allows us to have more in depth understanding of the model performance over a simple accuracy score. We can quickly see that there is an almost equally large number of both True Negatives and False Negatives. Which is some indication that our accuracy maybe be no better than a coin toss. The sum of the first horizontal axis gives the us the total number of 'False' or 0 in the actual data, and similarly for the next. This will help us understand the next metrics, i.e., TPR or True Positive Rate and TNR or True Negative Rate. TPR is the ratio of the correctly predicted positive

samples divided by all actual positive samples in the data. Similarly, TNR is the ratio of the correctly predicted negative samples divided by all actual negative samples. We see that only around 22% of all true positives have been predicted correctly. While this ratio is much larger for True Negatives, TNR is 78%. Which implies that the model was much more successful in predicting Negatives than predicting Positives. My first instinct to explain this was to search whether the data is significantly biased towards negatives, i.e., negatives far outnumber the positives. However, that is visibly untrue as the sum of the first horizontal axis, i.e., the total number of actual negatives is $44,763$, while the sum of the second horizontal axis, i.e., the total number of actual positives is $39,479$. A difference of only about 12%. This will not account for the massive discrepancy in the TPR and TNR. I am unable to explain this at the moment other than the obvious reason that the factors aren't a reasonable predictor of the target variable, i.e., whether this month's return will be greater than the median for previous month. Another thing to notice is that since the target variable is binary all the actual positives that are predicted wrongly get allotted a negative, which skews the prediction towards negatives. This will become relevant later on. The next metric is our accuracy score which as we expected is only marginally better than a coin flip with in-sample accuracy being at 53.7% and out-of-sample accuracy at 52.1%. The 10-Folds Cross Validation shuffles the train and test datasets and computes the average accuracy of 10 such shuffles. With a score of 53.6%, we can be sure than although accuracy is low in general, our model does perform as accurately out of sample as it does in sample. Which implies that although the signal to noise ratio is low, which is almost always the case in Quantitative Finance, we can be sure that the model we have built is of some practical significance and can be used on real world data. We also see that the model completely ignores Low Volatility Beta factor while accounting for all others. We also see the US Federal Rate and Unemployment Rate are the major predictors with the rest barely substantial. For a pie chart of how important each factor was in the model please see the Appendix (Ref. 2.3.1).

	0	1	Benchmark	Active	Neutral
Mean Return	-0.095434	0.018303	0.021519	-0.040881	0.302670
St. Dev.	0.297707	0.244822	0.248602	0.085981	0.203088
RR Ratio	-0.320564	0.074762	0.086561	-0.475464	1.490334

	0	1	Benchmark	Active	Neutral
Mean Return	-0.095434	0.018303	0.021519	-0.040881	0.302670
St. Dev.	0.297707	0.244822	0.248602	0.085981	0.203088
RR Ratio	-0.320564	0.074762	0.086561	-0.475464	1.490334

For a benchmark, I have also analysed the performance of the actual test data which is the first table. You'll notice that the benchmark is quite an unreasonable standard. We see an annual mean return of 129% for the positive group and an information ratio of almost 6. Which means we're risking a dollar to gain 6, while also more than doubling over investment year on year. Nothing in Quantitative Finance gives this kind of returns with such little risk. Of course there's a lot to be said about Survivourship Bias here. Whereby, our model focuses on the top historical performers neglecting the rest which leads to overestimation. However, that is pretty apparent. Coming back to our decision tree model, we see a below average mean annual return of 1.8% and an Information Ratio of around 0.085. Of course, these numbers are underwhelming, however, the more important issue here is that the Positive Group does not have defined returns for a lot of dates. This is because there are no stocks in this group for those dates. As previously mentioned, all false negatives get allotted a 0 which skews the data in that direction hence, leading to a shortfall of enough positive stocks. This implies we need better data. We need the data for all months in the specified time interval for each stock, which is not true for the data provided as it's missing a lot of months for a lot of stocks. We need more voluminous data as this isn't just a requirement for better performing models but instead is a necessity for performance in any Machine Learning models. Since, these models can analyse ginormous amounts of data it is usually better to feed them the same. The better the data the better the classifier model.



We see for what little data we have that the signal has outperformed the Benchmark. However, there is a lot of data missing, which makes this quite impractical.

3 ECONOMIC INTERPRETATION:

(Ref. Appendix 3) At the start we split the data based on Unemployment rate. With stocks with low Fed rates on the left and stocks with high gross profitability. Low US Federal rates signify an expansionary policy with economy doing well and growing and consumption at a high. High gross profitability would signify businesses with products out-competing the competition and a high turnaround . A safe investment anyhow. The first group is a set of stocks growing because the market in general is optimistic. The second group represents 'good' stocks based on profits.
Among the first group, the next split is even lower Fed rate on the left and low unemployment rate on the right. The employment group would then split into low value 'bad' stocks based on quarterly earning per share and the other based on low momentum. The lower Fed Rate group split into two, first with the lowest Fed rate symbolising a steadily growing economy but nothing about the stocks, second with a lower Unemployment rate which is not symbolic of anything much as can be categorised as marginally 'bad' stocks.
Among the second group representing 'good' stocks based on profits. The first split is again based on gross profitability with profitable stocks on the right and low momentum stocks on the left. However, this profitable stock group on the right, split into a node that got pruned and another that is too low in momentum. The low momentum stocks group on the left then splits based on quarterly earnings per share. Which is the only value group splitting again based on epq. This is however not ideal as these are already characterised as low momentum stocks.

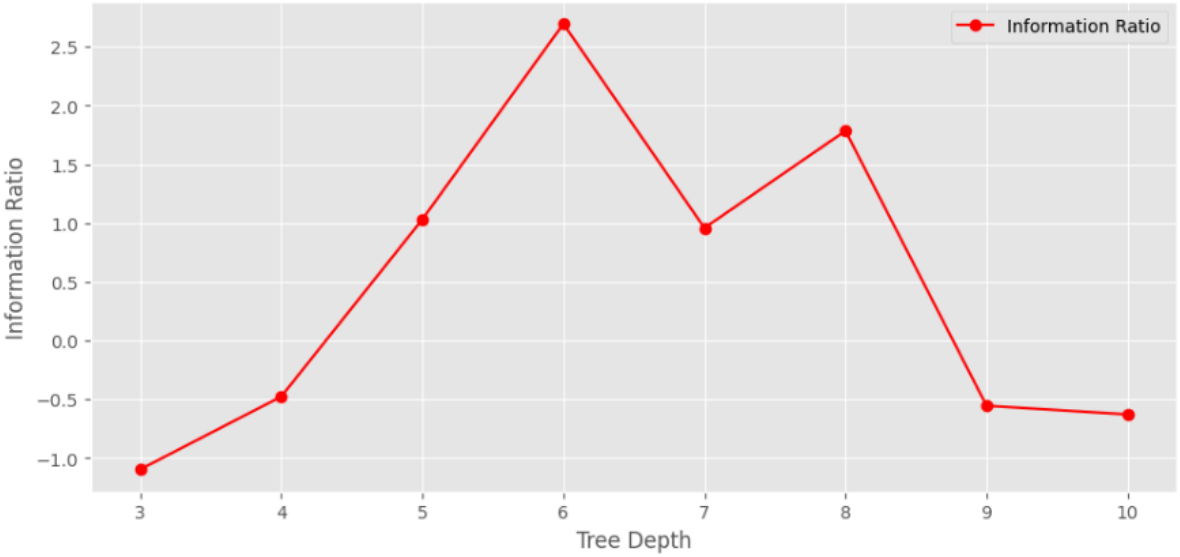
The algorithm has produced an investment strategy that is not in line with a standard theory based approach. In both the groups representing value, the one with the lowest Fed rate and the one with high epq but low momentum are the only two divisions that seems to have some correlation with 'good' stocks. The others have been divided based on factors that make no economic sense and wouldn't be used as metrics to measure 'good' stocks. Low Fed rates imply a booming economy which in turn implies growing stocks. But inverse is not true, growing stocks do not imply low Fed rates. Which means we can't differentiate between actual 'good' stocks and stocks growing simple because the economy is. A classic case of survivorship bias. Similarly, gross profitability is good indicator that a business has promise but isn't indicative of intrinsic value. It must be combined with other factors. The value group high on epq may represent 'good' stocks but they are low on momentum implying low returns although steady growth maybe in the future. The most troubling part is that the value factor earnings per share only has an importance of 1.7% and the same goes for momentum at 0.9%. This is troubling as the top two factors, i.e., Fed rate and Unemployment rate both have no substantial correlation with quality stocks.

4 OPTIMISATION:

4.1 Methodology:

We alter the maximum depth parameter of the decision tree model, which is the number of nodes along the longest path from the root node to the farthest leaf. We'll be testing all depths from one to ten and choose the one with the greatest Information Ratio.

4.2 Results:



We see that a maximum depth of 6 is the most optimal and the graph is surprisingly linear. With the only exception being the maximum depth of 8. This is usually the case with greater complexity. As complexity increases, so does the performance for while but with enough increase in complexity the inverse occurs and after that point the performance decreases most probably due to overfitting.
Analysing various metrics for the optimal depth decision tree:

	0	1	Benchmark	Active	Neutral
Mean Return	-0.124623	0.174747	0.021519	0.086819	0.463793
St. Dev.	0.318376	0.237077	0.248602	0.032210	0.184048
RR Ratio	-0.391433	0.737089	0.086561	2.695412	2.519951

We see that we haven't actually achieved anymore accuracy. But we didn't optimise for that we had instead optimised for the information ratio which has indeed improved to a substantial 2.7. The positive group also seems to produce a mean annual return of 17.5% which is still quite high for the real world but compared to the benchmark metrics it's nothing. Analysing the accuracy metrics we see that the model is still biased towards negative and is more or less similarly successful in the prediction of both positives and negatives with a slight increase in True Positive Rate. The out-of-sample accuracy is slightly less than the previous model however, the cross validation accuracy is exactly the same as the last one which implies that this model is just as successful.

```
Confusion Matrix:
 [[31710 13053]
 [28374 11105]]

TPR = 0.28129
TNR = 0.7084
```

```
Out-of-Sample Accuracy:
 0.508

10-Folds Cross Validation
Average Accuracy:
 0.536
```

5 BAGGING AND BOOSTING:

5.1 Methodology:

We use the optimal depth from the previous section to train a Random Forest Classifier model and a Gradient Boosting Classifier in an attempt to improve upon our metrics.

5.2 Results:

Using Random Forest Classifier:

	0	1	Benchmark	Active	Neutral
Mean Return	-0.158728	-0.032353	0.021519	0.003252	0.285639
St. Dev.	0.337065	0.296890	0.248602	0.062930	0.208998
RR Ratio	-0.470913	-0.108974	0.086561	0.051675	1.366707

```
Confusion Matrix:
 [[24797 19966]
 [20606 18873]]
TPR = 0.47805
TNR = 0.55396
Out-of-Sample Accuracy:
 0.518
```

We see an negative mean return for the positive group and substantial decrease in the Information ratio to 0.1 compared to the optimal decision tree model. The model seems to be approximately equally accurate as the last ones albeit with a slight increase. This is not totally surprising as we are using only six features or information signals, so the distinguishing feature of a random forest is not particularly valuable. We also see that the features have been diversified in the model albeit only marginally as the US Federal Rate and Unemployment Rate are still the major factors. For a pie chart of Feature Importance, see Appendix (Ref. 5.2.1).

Taking a closer look at the confusion matrix, we see that the number of False Positives have drastically increased, almost doubled, compared to the last model. Which implies that a lot of 'bad' stocks have been classified into 'good' category. Which probably explains why there is a sharp decrease in the mean returns of the positive group. We also see the True Positive rate has increased while True Negativity rate has decreased balancing each other out. This model seems to be better at predicting positives than the last one. However, it's also worse at predicting negatives. If I simple had the model metrics without the simulated investment performance metrics, I would much prefer this compared to the previous one as this is more balanced and accurate while averaging over both classes. That's because 'bad' stocks could be in a spectrum and the wrong predictions don't differentiate between really 'bad' stocks and marginally 'bad' ones. Therefore, it would be better to modify and continually train the data on unique but similar datasets to achieve better accuracy after which we can proceed with measuring the simulated investment performance as this would be more realistic.

Using Gradient Boosting Classifier:

	0	1	Benchmark	Active	Neutral
Mean Return	-0.065598	-0.028942	0.021519	-0.021722	0.071335
St. Dev.	0.320246	0.303445	0.248602	0.192724	0.248318
RR Ratio	-0.204838	-0.095377	0.086561	-0.112710	0.287275

```
Confusion Matrix:
 [[17469 27294]
 [15469 24010]]
TPR = 0.60817
TNR = 0.39026
Out-of-Sample Accuracy:
 0.492
```

Again, we see an negative mean return for the positive group and substantial decrease in the Information ratio to 0.19, compared to the optimal decision tree model. The model seems to be approximately equally accurate as the last ones albeit with a slight decrease. The decrease in the True Negative Rate might be helpful in explaining that. Again since Boosting was designed to work on large datasets with a lot of features, it isn't particularly surprising that we got underwhelming results. We also see that the features have been better diversified in this model, compared to the previous one, albeit only

marginally as the US Federal Rate and Unemployment Rate are still the major factors. For a pie chart of Feature Importance, see Appendix (Ref. 5.2.2).

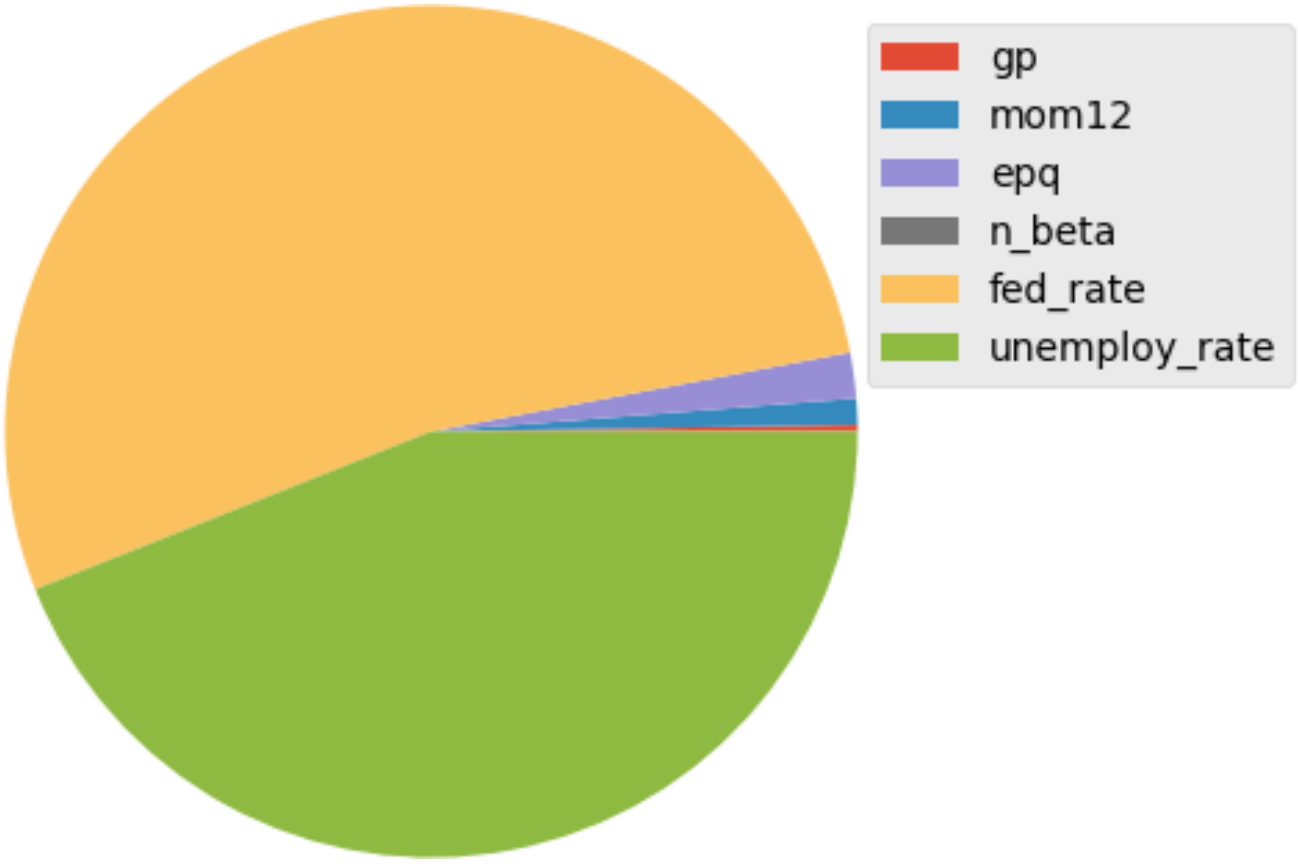
Taking a closer look at the confusion matrix, we see that the number of False Positives have substantially increased, compared to the last model. Which implies that a lot of 'bad' stocks have been classified into 'good' category. Although we don't necessarily see a decrease in the mean returns which have slightly increased. We also see the True Positive rate has increased while True Negativity rate has decreased. Which could explain why the portfolio performed better than the last model as the number of True Positives have increased by about 30%. A decrease in True Negatives and an increase in True Positives balancing each other out might explain these returns. This model seems to be better at predicting positives than the last one. However, it's also worse at predicting negatives. It might be that given the sequential nature of the Boosting, the really 'bad' stocks might have been filtered out leaving only the marginally 'bad' ones in the prediction set. Which could explain the slight increase in the mean returns and the Information Ratio. To make both, Bagging and Boosting, more realistic we would need a much larger dataset.

6 CONCLUSIONS

We see that the Optimal Decision Tree model seems to be the most accurate and the best performer for the simulated investment strategy. However, I keep going back to the Portfolio Value graph and the absolute lack of data therein. As suggested earlier, Machine Learning models are only as good as the data you feed them. And since the data was so small, had so many missing values and perhaps unrecognised biases, we got underwhelming results. The data must have contained all the monthly prices for the specified time interval, for each stock. The data should have been much more voluminous, with several thousands more stocks. The data should have had many more features for the classifier particularly the Boost to work. Machine Learning algorithms are famous for being difficult to interpret and small datasets exacerbate that. In conclusion, Machine Learning strategies have their place in Quantitative Finance, as they are exceptional at identifying trends and patterns but their requirements must be met in order to get a practical, and interpretable results.

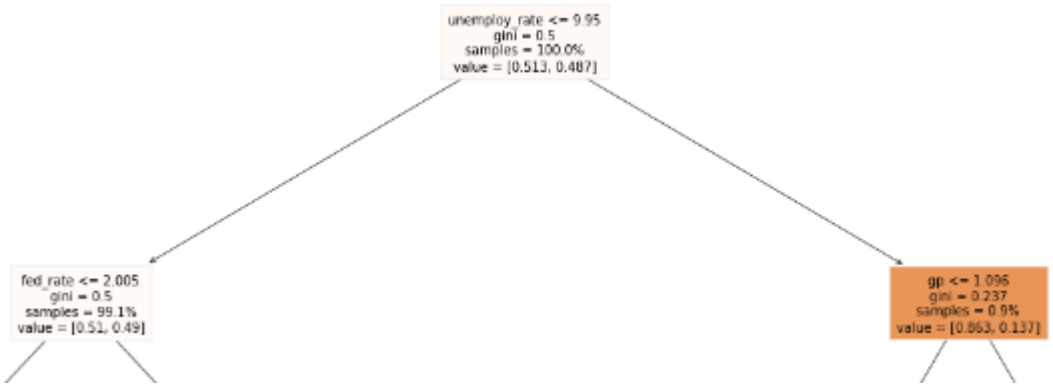
Appendix

2.3.1 Pie Chart

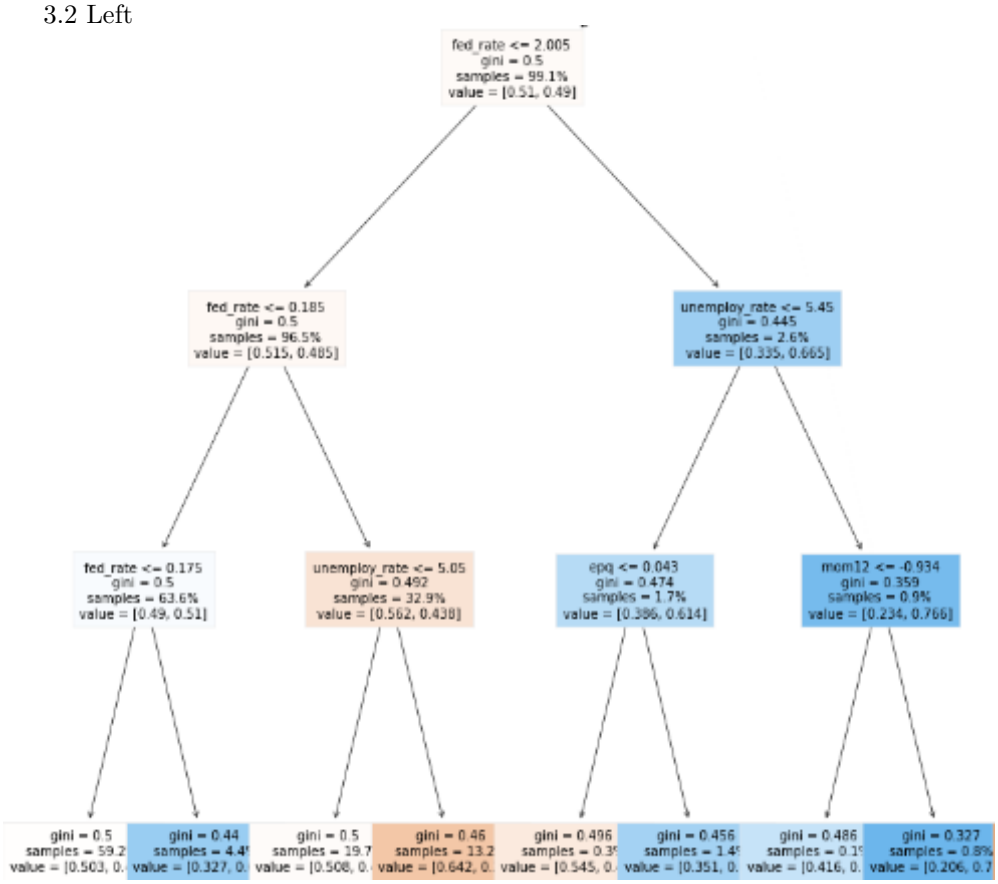


3 Binary Tree

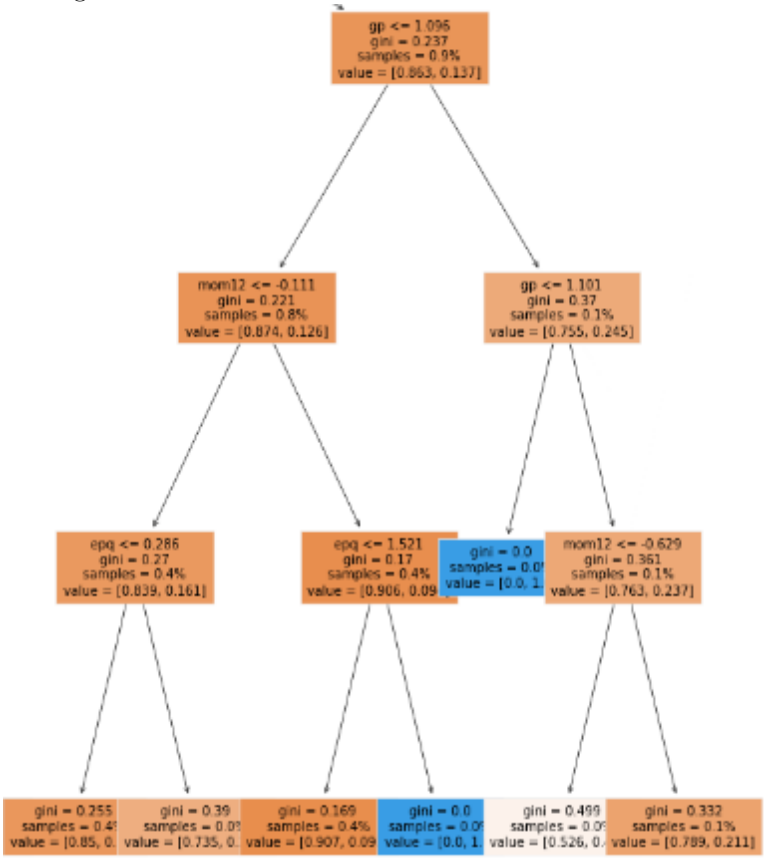
3.1 Upper



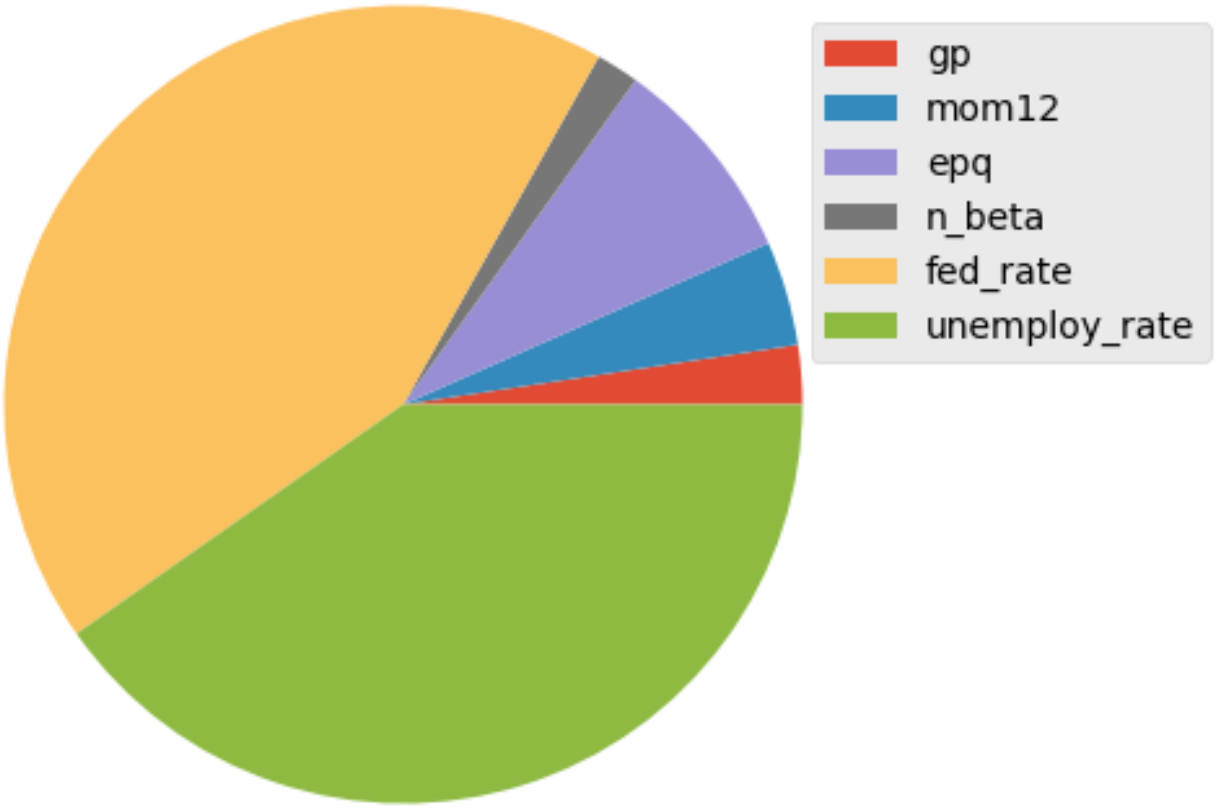
3.2 Left



3.3 Right



5.2.1 Pie Chart RNF



5.2.2 Pie Chart GBT