Machine Learning Engineer Nanodegree

Capstone Project - Use clustering and multiple tabular predictors to predict the 1-day future price of gold

Neil Simon

December 29, 2021

I. Definition

Project Overview

Predicting changes in the value of gold has obvious financial benefit including knowing when to buy or sell and determining future volatility. Historically, gold has been used as a currency, either directly or as backing for a paper currency (Gold's history as a currency standard, 2010) and it has proven to be a useful asset to hold during times of economic downturn and uncertainty. This is despite the relatively small role it plays as a functional material in the modern world. The efficient market hypothesis (EMH) suggests that the market price reflects all information and that it should be impossible to consistently predict future prices of gold, and in particular, that it is impossible to consistently get returns greater than market level returns (Fama, E. F., 1970). Therefore, any system which is able to beat the market consistently would not only provide significant avenues for financial return, but also demonstrate that the EMH is not perfectly supported. In this project, I implemented a system which divides gold prices into clusters using a K-Means predictor and related historical data, and based on the predicted cluster, uses a TabularPredictor trained for that cluster to determine a predicted 1 day future return on gold. Having analysed the results, and compared them with naive trading strategies, I have determined that none of the setups resulted in a system capable of predicting the future value of gold with any significant degree of accuracy.

Problem Statement

The problem I attempted to solve is to predict the next day's gold price based on historical gold price trends and the Standard and Poors 500 index (S&P 500). To do this I performed the following tasks:

1. Retrieve the historical price of gold and historical values of the S&P 500.
2. Combine such into a suitable table of data.
3. Add additional columns of data expressing historical and most recent returns for gold and S&P 500.
4. Train a K-Means clustering system to predict clusters.
5. Train K different TabularPredictors on data based on predicted cluster of said data.
6. Gather predictions on test data.
7. Compare results to actual returns on predicted days.

8. Iterate with different numbers for K (different numbers of clusters).
9. Evaluate final results.

By comparing predicted return and therefore price of gold with the actual market price I determined how accurately the model predicts such and the associated level of risk. Additionally, based on these predictions I extrapolated an expected return of the model over a long period and compare such with the actual market return and with other naive trading strategies.

Metrics

The metrics I have applied in evaluating the predictions are relative cumulative return based on recommended trading, and accuracy. To evaluate the cumulative return over the test data and compare such with the actual return:

$$return = \frac{closing\_price}{opening\_price} - 1$$

The cumulative return is just a matter of getting the product of all actual returns:

$$c\_return = (1 + return_1) * (1 + return_2) * ... * (1 + return_{n-1}) * (1 + return_n)$$

Of course, if we know that the return is going to be positive (gold going up in value), it makes sense to buy or hold, and if it is going down, we sell. Therefore we can bring the concept of placing a buy or sell order and the ideal order is thus:

$$order_{ideal} = \begin{cases} +1 & \text{if } return \geq 0 \\ -1 & \text{if } return < 0 \end{cases}$$

Thus we can change the above to give an idealised return:

$$c\_return_{ideal} = (1+order_1 * return_1) * (1+order_2 * return_2) * ... * (1+order_{n-1} * return_{n-1}) * (1+order_n * return_n)$$

Assuming that the predicted values are the ideal, we can use the above and create a formula for $c\_return_{recommended}$ where we follow the recommendations of the system and use the following order formula:

$$order_{recommended} = \begin{cases} +1 & \text{if } return_{predicted} \geq 0 \\ -1 & \text{if } return_{predicted} < 0 \end{cases}$$

Thus the actual cumulative return, if following this strategy of trading recommended by the predicted values, is:

$$c\_return_{recommended} = (1 + order_1 * return_1) * (1 + order_2 * return_2) * ...$$

Thus we can compare the return that following the recommended system would give us, the return that simply holding gold would (simply the market return) and a fair random coin (an average cumulative return of 1). For comparing

returns, it makes most sense to consider the market as the baseline and view the relative cumulative return of the recommendation system:

$$r\_return_{recommended} = \frac{c\_return_{recommended}}{c\_return}$$

A value greater than 1 would suggest a system that has outperformed the actual market.

As it is possible to outperform the market through random chance and since the cumulative return is a product of all the predictions, I also compare the $order_{ideal}$ and $order_{recommended}$ for accuracy to see if the system does a good job of accurately predicting the direction of gold prices.

$$Accuracy = \frac{count(order_{ideal} == order_{recommended})}{number\_of\_samples}$$

A value consistently greater than 0.5 would indicate a greater degree of accuracy than pure chance.

Other metrics, such as precision and recall do not make sense here as the relative rate of positives and negatives is close to 1. If the market was particularly weighted towards or against days with increases, then we would need to consider precision, recall/sensitivity, and specificity.

II. Analysis

(approx. 2-4 pages)

Data Exploration

I have used a combination of 2 different datasets, one is the price of gold and the other the closing price of the S&P 500 between 2006-01-10 and 2021-12-03. By combining these two I created a third dataset with the date, gold price and S&P 500 closing price. From this I generated the gold return for the ranges of 1 day's return, 2 day's return ending yesterday, 4 day's return ending 3 days back, and 8 day's return ending 7 days back. The general formula for this is:

$$return_{(x,y)} = \frac{p_{n-x} - p_{n-x-y}}{p_{n-x-y}}$$

Where p_n is todays gold price, and p_{n-1} is yesterday's gold price, etc., x is the number of days back we are looking to find the return at, and y is the number of days over which the return is calculated. So for today's return (1 day's return ending today) $x = 0$ and $y = 1$, and for 4 day's return ending 3 days back $x = 3$ and $y = 4$.

I then normalised these features by getting the 16 day standard deviation on daily returns and dividing the return's by that (adjusted for the number of days

over which the return was calculated). The normalization formula for these is:

$$return\_norm_{xy} = \frac{r_{xy}}{\sigma_{16} * \sqrt{2}y}$$

This resulted in 4 normalised return values for each day's gold price and a standard deviation value.

I applied the same approach to the S&P data to create 4 normalised return values for each day, and a standard deviation value. Resulting in a total of 10 features with information for each day. I added these additional features to each row to increase the amount of information about recent gold and S&P 500 returns, while not massively increasing the total amount of information in the dataset. While it would be possible to have just included the returns going back multiple days in each row, this would have increased the degrees over which the K-Means clustering took place, and therefore resulted in a less useful clustering process. My approach was to attempt to maximise the amount of information in each row, without increasing the number of features too greatly, so as to potentially benefit from the use of the K-Means clustering algorithm.

Ultimately, it is the one day future return ($\overline{return}_{(-1,1)}$) we are looking to calculate as from that we can determine tomorrow's price as we know today's price.

After creating the normalised returns, I examined the data to make sure that it was somewhat close to normally distributed. While each does display some degree of variance from the expected normal distribution, it is sufficiently close to be assumed that it is normally distributed for the purposes of this project (see figures 1 through 8).

Ultimately, I determined that for the purposes of this project, the data was well formatted and did not appear to have any abnormalities.

Exploratory Visualization

Figures 1 through 8 show that we are dealing with relatively suitable data for analysis as it is close to normally distributed.

There are few actual features of note about this data, with the following being the most promenant:

1. The return must always be between -1 and ∞ as the value of gold will never go negative. That said, over the range of data tested, the return was in the range $[-0.05, 0.06]$.

In this section, you will need to provide some form of visualization that summarizes or extracts a relevant characteristic or feature about the data. The visualization should adequately support the data being used. Discuss why this visualization was chosen and how it is relevant. Questions to ask yourself when writing this section: - *Have you visualized a relevant characteristic or feature*

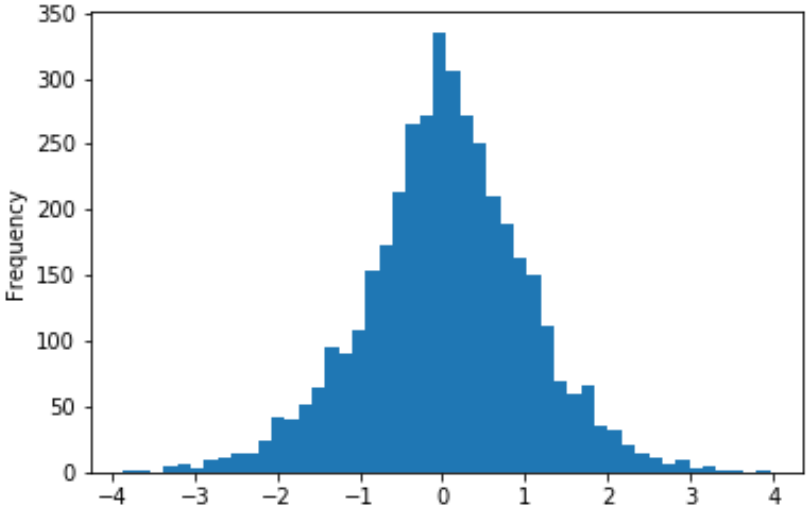

Figure 1: Normalised Gold Return Over 1 Day

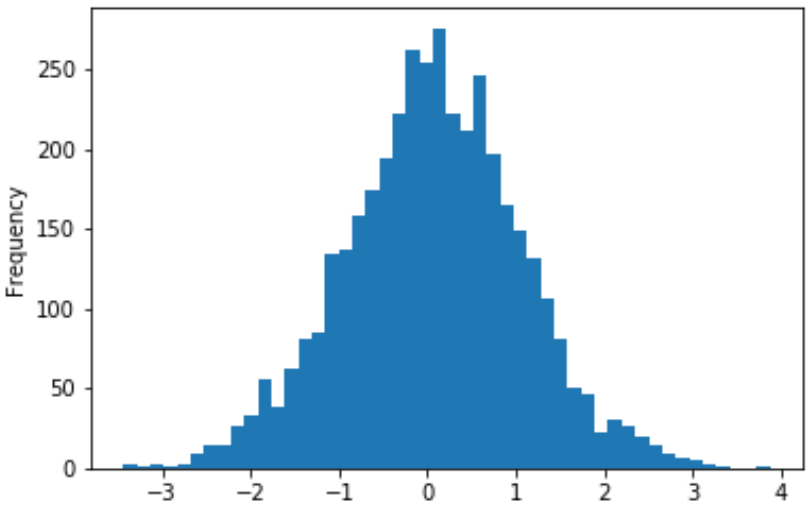

Figure 2: Normalised Gold Return Over 2 Days Ending 1 Day Ago

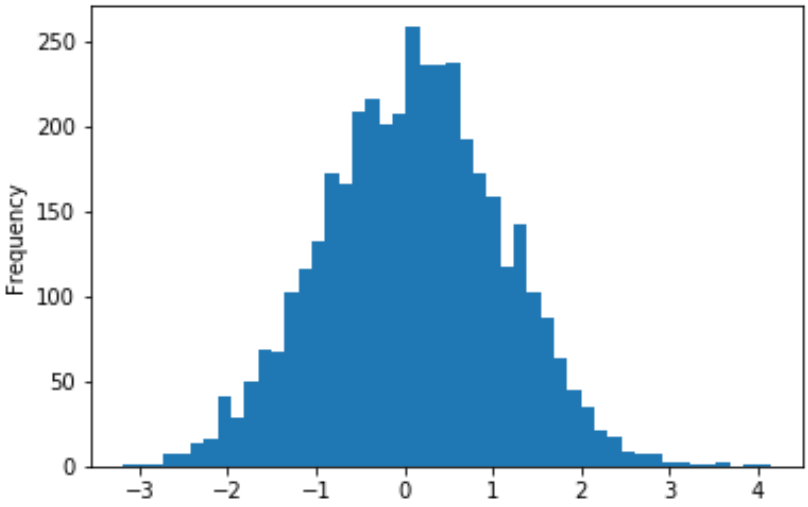

Figure 3: Normalised Gold Return Over 4 Days Ending 3 Days Ago

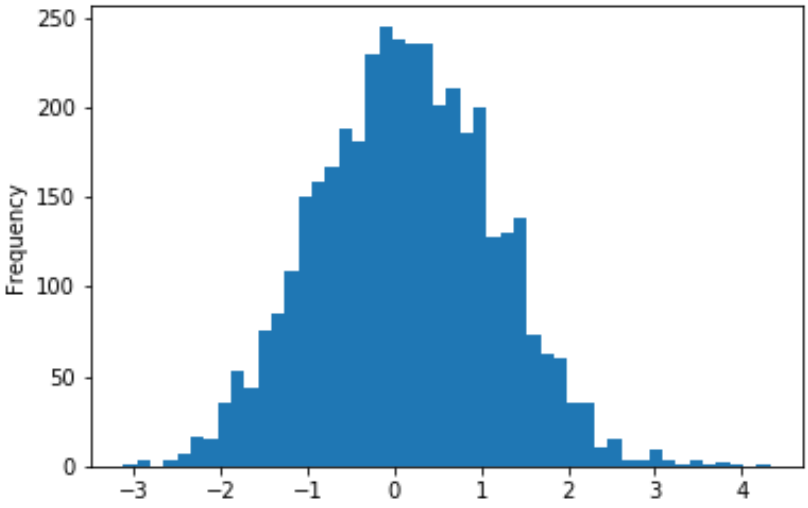

Figure 4: Normalised Gold Return Over 8 Days Ending 7 Days Ago

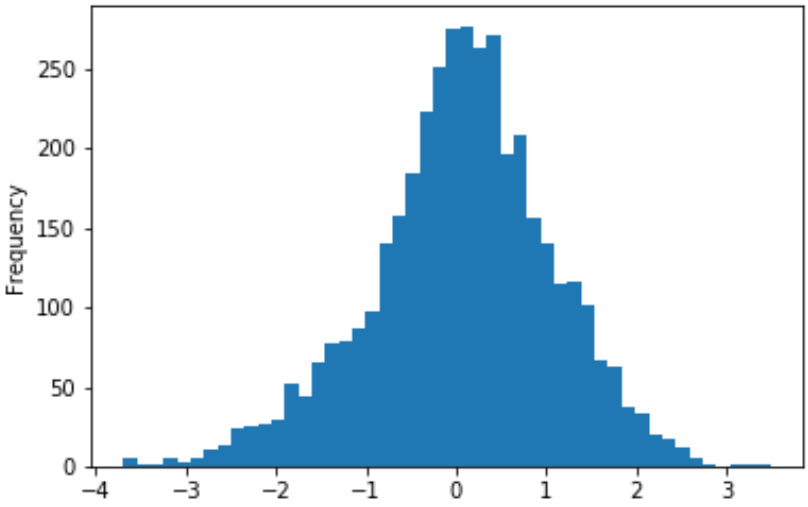

Figure 5: Normalised S&P Return Over 1 Day

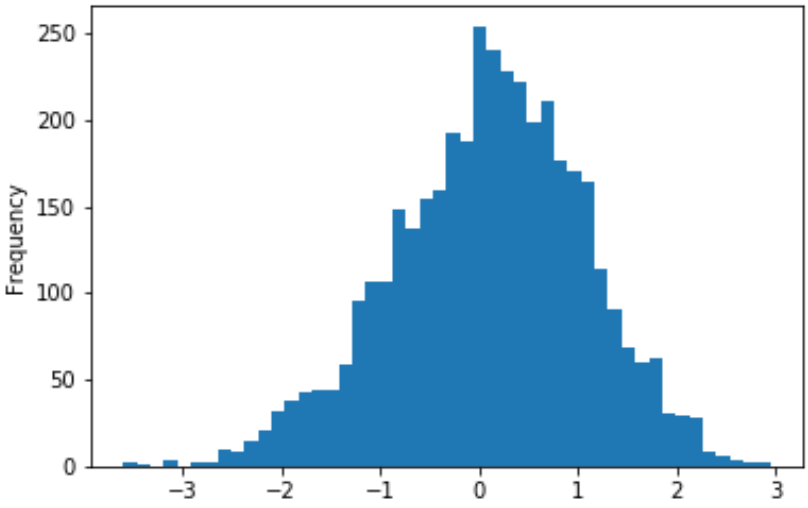

Figure 6: Normalised S&P Return Over 2 Days Ending 1 Day Ago

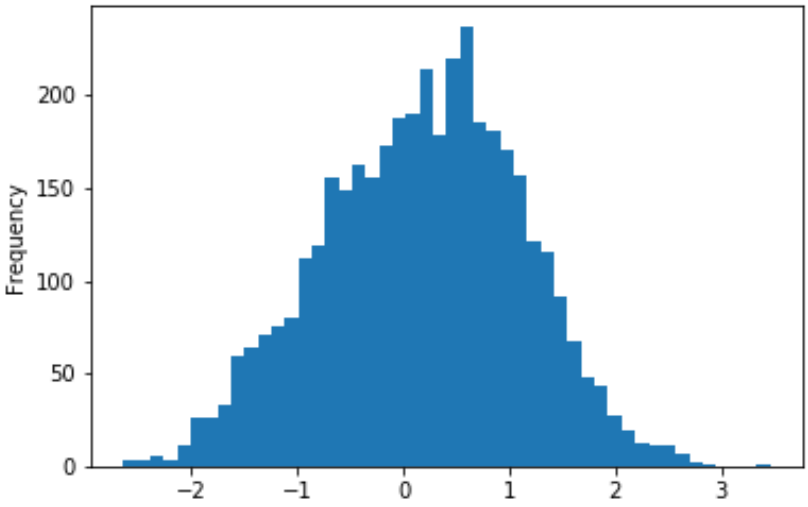

Figure 7: Normalised S&P Return Over 4 Days Ending 3 Days Ago

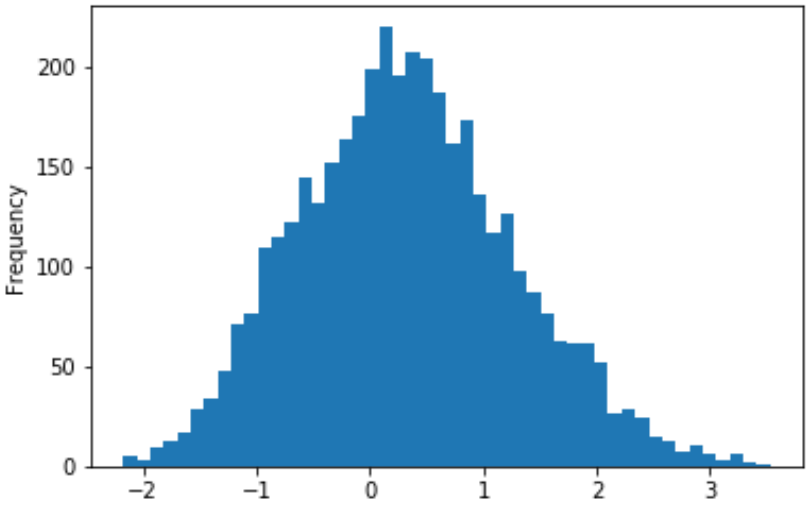

Figure 8: Normalised S&P Return Over 8 Days Ending 7 Days Ago

about the dataset or input data? - Is the visualization thoroughly analyzed and discussed? - If a plot is provided, are the axes, title, and datum clearly defined?

Algorithms and Techniques

In this section, you will need to discuss the algorithms and techniques you intend to use for solving the problem. You should justify the use of each one based on the characteristics of the problem and the problem domain. Questions to ask yourself when writing this section: - *Are the algorithms you will use, including any default variables/parameters in the project clearly defined? - Are the techniques to be used thoroughly discussed and justified? - Is it made clear how the input data or datasets will be handled by the algorithms and techniques chosen?*

Benchmark

In this section, you will need to provide a clearly defined benchmark result or threshold for comparing across performances obtained by your solution. The reasoning behind the benchmark (in the case where it is not an established result) should be discussed. Questions to ask yourself when writing this section: - *Has some result or value been provided that acts as a benchmark for measuring performance? - Is it clear how this result or value was obtained (whether by data or by hypothesis)?*

III. Methodology

(approx. 3-5 pages)

Data Preprocessing

In this section, all of your preprocessing steps will need to be clearly documented, if any were necessary. From the previous section, any of the abnormalities or characteristics that you identified about the dataset will be addressed and corrected here. Questions to ask yourself when writing this section: - *If the algorithms chosen require preprocessing steps like feature selection or feature transformations, have they been properly documented? - Based on the* ***Data Exploration*** *section, if there were abnormalities or characteristics that needed to be addressed, have they been properly corrected? - If no preprocessing is needed, has it been made clear why?*

Implementation

In this section, the process for which metrics, algorithms, and techniques that you implemented for the given data will need to be clearly documented. It should be abundantly clear how the implementation was carried out, and discussion should be made regarding any complications that occurred during this process. Questions to ask yourself when writing this section: - *Is it made clear how the*

algorithms and techniques were implemented with the given datasets or input data? - Were there any complications with the original metrics or techniques that required changing prior to acquiring a solution? - Was there any part of the coding process (e.g., writing complicated functions) that should be documented?

Refinement

In this section, you will need to discuss the process of improvement you made upon the algorithms and techniques you used in your implementation. For example, adjusting parameters for certain models to acquire improved solutions would fall under the refinement category. Your initial and final solutions should be reported, as well as any significant intermediate results as necessary. Questions to ask yourself when writing this section: - *Has an initial solution been found and clearly reported? - Is the process of improvement clearly documented, such as what techniques were used? - Are intermediate and final solutions clearly reported as the process is improved?*

IV. Results

(approx. 2-3 pages)

Model Evaluation and Validation

In this section, the final model and any supporting qualities should be evaluated in detail. It should be clear how the final model was derived and why this model was chosen. In addition, some type of analysis should be used to validate the robustness of this model and its solution, such as manipulating the input data or environment to see how the model's solution is affected (this is called sensitivity analysis). Questions to ask yourself when writing this section: - *Is the final model reasonable and aligning with solution expectations? Are the final parameters of the model appropriate? - Has the final model been tested with various inputs to evaluate whether the model generalizes well to unseen data? - Is the model robust enough for the problem? Do small perturbations (changes) in training data or the input space greatly affect the results? - Can results found from the model be trusted?*

Justification

In this section, your model's final solution and its results should be compared to the benchmark you established earlier in the project using some type of statistical analysis. You should also justify whether these results and the solution are significant enough to have solved the problem posed in the project. Questions to ask yourself when writing this section: - *Are the final results found stronger than the benchmark result reported earlier? - Have you thoroughly analyzed and discussed the final solution? - Is the final solution significant enough to have solved the problem?*

V. Conclusion

(approx. 1-2 pages)

Free-Form Visualization

In this section, you will need to provide some form of visualization that emphasizes an important quality about the project. It is much more free-form, but should reasonably support a significant result or characteristic about the problem that you want to discuss. Questions to ask yourself when writing this section: - *Have you visualized a relevant or important quality about the problem, dataset, input data, or results? - Is the visualization thoroughly analyzed and discussed? - If a plot is provided, are the axes, title, and datum clearly defined?*

Reflection

In this section, you will summarize the entire end-to-end problem solution and discuss one or two particular aspects of the project you found interesting or difficult. You are expected to reflect on the project as a whole to show that you have a firm understanding of the entire process employed in your work. Questions to ask yourself when writing this section: - *Have you thoroughly summarized the entire process you used for this project? - Were there any interesting aspects of the project? - Were there any difficult aspects of the project? - Does the final model and solution fit your expectations for the problem, and should it be used in a general setting to solve these types of problems?*

Improvement

In this section, you will need to provide discussion as to how one aspect of the implementation you designed could be improved. As an example, consider ways your implementation can be made more general, and what would need to be modified. You do not need to make this improvement, but the potential solutions resulting from these changes are considered and compared/contrasted to your current solution. Questions to ask yourself when writing this section: - *Are there further improvements that could be made on the algorithms or techniques you used in this project? - Were there algorithms or techniques you researched that you did not know how to implement, but would consider using if you knew how? - If you used your final solution as the new benchmark, do you think an even better solution exists?*

Before submitting, ask yourself. . .

- Does the project report you've written follow a well-organized structure similar to that of the project template?
- Is each section (particularly **Analysis** and **Methodology**) written in a clear, concise and specific fashion? Are there any ambiguous terms or phrases that need clarification?

- Would the intended audience of your project be able to understand your analysis, methods, and results?
- Have you properly proof-read your project report to assure there are minimal grammatical and spelling mistakes?
- Are all the resources used for this project correctly cited and referenced?
- Is the code that implements your solution easily readable and properly commented?
- Does the code execute without error and produce results similar to those reported?

https://auto.gluon.ai/dev/tutorials/cloud_fit_deploy/cloud-aws-sagemaker-training.html https://auto.gluon.ai/dev/tutorials/cloud_fit_deploy/cloud-aws-sagemaker-deployment.html