Predicting Gold Prices

Megan Potoski

Abstract
Given historical price data up to and including a given day, I attempt to predict whether the London PM price fix of gold the following day will increase or decrease relative to the current day's price fix. My predictions are based primarily on technical indicators calculated from historical price data for gold as well as for a variety of financial variables. Using logistic regression for binary classification, I attain over 54% accuracy using only features calculated from the gold price, and over 69% accuracy after adding certain intermarket variable features that display a high correlation with gold prices. I also simulate executing 1-day trades based on this prediction, where I can either buy or sell 1 ounce of gold each day, and achieve an average daily profit exceeding $0.65.

1. Introduction

Machine learning has often been applied to the prediction of financial variables, but usually with a focus on stock prediction rather than commodities. In my project, I chose to apply supervised learning to the prediction of gold prices in order to see what kind of success I could achieve. For the purposes of profitability, it is more important to predict the relative change in price tomorrow (i.e. whether the price will go up or down) than to predict the absolute price tomorrow, so I have formulated this as a classification problem: given historical price data up to a given day, my algorithm attempts to predict whether the gold price tomorrow will be higher or lower than it is today.

The most reliable source of gold price data that I found was the London afternoon (PM) price fix, which I obtained from the USA Gold website. As gold is not traded in the same sense as stocks, there is no separate data available for, for instance, daily open vs. close prices or trading volume. I have gathered the daily price fix data spanning early 2007 to late 2013 - about 7 years, where 5 out of every 7 days are eligible data points, which yields about 1700 training examples.

Another question that I sought to address in this project is whether or not intermarket financial variables, like stock index prices, exchange rates, bond rates, and other commodity prices, can be effective inputs to the prediction of gold prices. Historically, these variables have exhibited a significant correlation; but I was curious as to the reliability of these relationships as well as their applicability in extremely short-term situations.

2. Feature Definition

From the outset, I expected that feature definition would be the most challenging component of this project, as the price data alone lends little insight into future price movement. The goal thus becomes to find and to calculate the features that best capture the movement of gold prices and provide information on not only their past and current movements, but also their future movements.

2.1 Technical Indicators

Quantitative research on stock prediction often uses an assortment of calculated technical indicators. As gold prices should follow many of the principles of predicting financial markets in general, I have studied past work on stock prediction to learn about these methods [1, 2]. Below I summarize the indicators I have chosen to use and how I calculate them:

Trendlines The ability to recognize the current price trend is vital to the accurate prediction of future prices. Not only can determining the trend over the previous time period provide a rough estimate of continued price movement, but it can also prove useful in spotting a trend reversal (uptrend to downtrend, or vice versa). Since prices can fluctuate considerably, a trendline is not an ideal way to describe the data, but it is still a valuable tool. I have chosen to use simple linear regression with a least-squares cost function to fit a trendline over the last n days, by determining a line of the form

$$y = ax + b$$

that minimizes

$$\sum_{i=1}^{n} (ax^{(i)} + b - y^{(i)})^2$$

Rate of Change Momentum measures the difference between the price on day x and the price n days before. Rate of change (ROC) is essentially normalized momentum; I calculate this as follows:

$$\mathrm{ROC}_n(x) = \frac{P(x) - P(x-n)}{P(x-n)}$$

Momentum and ROC are often used to establish trends, i.e. $\mathrm{ROC} > 0$ signals an overall uptrend, and $\mathrm{ROC} < 0$ signals a downtrend. However, it is also useful to compare rate of change over varying time periods: for instance, if $0 < \mathrm{ROC}_5(x) < \mathrm{ROC}_{10}(x)$, or $0 < \mathrm{ROC}_5(x) < \mathrm{ROC}_5(x-5)$, this may signal a weakening and potentially a reversal in the uptrend.

Ratios The ratio between ROC calculated over different time intervals (particularly ROC_n / ROC_m for $m > n$) is informative because it lends insight into how the change in price (similar to the first derivative of price) is changing over time (similar to the second derivative of price).

Stochastic Oscillator The stochastic oscillator is used to determine overbought or oversold levels of a stock or commodity. Overbought basically means that the price has increased significantly over a short period of time and may be artificially high; this might mean that the underlying asset is overvalued and the market will soon adjust, bringing the price back down. The stochastic oscillator assumes that, in uptrends, prices will close near the upper end of the recent price range, and in downtrends, near the lower end. Adapting this to use the daily price fix rather than close prices, I calculate the oscillator on day x over an n-day period as follows:

$$L_n = \text{lowest price over the past } n \text{ days}$$
$$H_n = \text{highest price over the past } n \text{ days}$$
$$P(x) = \text{price on day } x$$
$$\%K = \frac{P(x) - L_n}{H_n - L_n} \times 100\%$$

In general, $\%K$ generates a buy signal if it is less than 20, or a sell signal if it is greater than 80.

3. Initial Tests

There are a variety of algorithms which can be used for classification; the two I chose to focus on for this project are logistic regression and SVM. First, I explain the evaluation metric used throughout; the training/testing strategy; and preliminary feature selection.

3.1 Evaluation Metric

I have formatted this as a classification problem, where $y^{(i)} = 1$ means the gold price goes up from day i to day $i+1$, and $y^{(i)} = 0$ means the gold price goes down (or stays the same). Now, denote any example with label 1 a "positive" example, and with label 0 a "negative" example. To evaluate the success of a given algorithm, I plan to take into account all three of the following metrics:

$$\textbf{Precision} = \frac{\text{true positives}}{\text{true positives + false positives}}$$
$$\textbf{Recall} = \frac{\text{true positives}}{\text{true positives + false negatives}}$$
$$\textbf{Accuracy} = \frac{\text{true positives + true negatives}}{\text{all examples}}$$

3.2 Train and Test Sets

As explained above, I have approximately 1700 data points spanning early 2007 to late 2013. I believe that the best way to train and test the algorithm on this data is using k-fold cross-validation. Not only does this make efficient use of the data by using every example for both training and testing at some point, but it also avoids the not-unlikely situation of choosing a comparatively small test window consisting of an uneven representation of all possible examples. For example, if I chose to test on one 100-day period in which gold happened to be exhibiting a strong uptrend, this would not adequately test my algorithm's ability to accurately predict what happens in a downtrend. Thus, for all algorithms described below, I train and test on the full data set with k-fold cross-validation for $k = 10$.

3.3 Basic Feature Selection

In the feature definition section above, I described the usefulness of trendlines and rate of change, but never specified the exact periods over which I would calculate these. According to past research, a reasonable period would range from about 2 days to 20 days. Let `nums` hold the array of periods I include as features. Varying only the items of `nums`, I ran forward selection with logistic regression on this model using precisely the following features (calculated only for gold price fix, but none of the intermarket variables):

- Slope of trendline at day x and $x-1$ for each n in `nums`
- ROC_1; ROC_n for each n in `nums`
- Ratios $\text{ROC}_1/\text{ROC}_n$ for each n in `nums`
- Value of %K oscillator for period 14

I obtained an optimal value `nums` = [2, 3, 5, 8, 15, 17].

3.4 Logistic Regression

Using the set of features selected above, the first algorithm I tried was logistic regression, using the linear model from Python's scikit-learn library, in an attempt to classify as accurately as possible whether the following day's London PM gold price fix would be higher or lower than the current day's. I used the full set of gold price fix technical indicator features but none of the intermarket variables. I also used *l-1* norm for penalization rather than the default *l-2*, as it achieved better results. The average results of running logistic regression with 10-fold cross-validation were as follows:

Table 1. Logistic Regression (Gold Price Features Only)

Precision	55.03%
Recall	76.27%
Accuracy	54.23%

So the recall of logistic regression is over 75% on average, which is very high. This means that almost all positive examples were correctly classified as positive. However, the precision is just over 55%, which is relatively low. This is because the algorithm produced many false positives; even for negative examples, it is more likely to predict 1 than 0.

3.5 SVM

The second algorithm that I tried was SVM, using the SVC model from scikit-learn. I tried four variants of SVM: the default RBF (radial basis function) kernel; the linear kernel with *l-1* regularization; the linear kernel with *l-2* regularization; and the polynomial kernel. After getting disappointing accuracy (around 51-52%) with all four, I normalized all of the input features by scaling, setting

$$X_j^{(i)} = \frac{X_j^{(i)} - \text{mean}_j}{\max_j - \min_j}$$

Ultimately, the highest accuracy that I was able to achieve with any kernel with or without normalization was 53.13%, which occurred when the SVM predicted a positive label for every example. Unfortunately, all variants of SVM displayed a strong propensity to favor predicting whichever class occurred more frequently in the training set (i.e. if `X_train` contained slightly more positive examples than negative, then `Y_test` would contain almost all positive predictions).

For this reason, I decided that SVM is likely not the best algorithm for this task; regardless of the kernel, and even if the input features are normalized, the SVM always favors one class. Thus I proceed with a primary focus on logistic regression. However, I do attempt SVM once again with the final feature set, and see a vast improvement in results.

3.6 Bias vs. Variance

Before proceeding, it is important to determine whether the high test error is due to high bias or high variance, i.e. underfitting or overfitting the training set. Even when trained and tested on the entire data set, logistic regression still achieves only 55.51% accuracy and 55.88% precision using the gold price-only feature set. This means that the high error is occurring due to high bias, not high variance.

4. Additional Features

As the error above is due almost entirely to the bias of the classifier, the current feature set clearly does not provide the algorithm with adequate information to make the right prediction. In this section, I explain how I add features to improve both train and test accuracy.

4.1 Intermarket Variables

Gold prices, and commodity prices in general, may also be related to other financial variables. For instance, gold prices are commonly thought to be related to stock prices; interest rates; the value of the dollar; and other factors. Therefore, I wanted to explore whether these other variables could be effective inputs to the prediction of gold prices. I collected the following data over the same time period as the gold price fix data, from early 2007 to late 2013:

- **US Stock Indices:** Dow Jones Industrial Average (DJI); S&P 500 (GSPC); NASDAQ Composite (IXIC)
- **World Stock Indices:** Ibovespa (BVSP); CAC 40 (FCHI); FTSE 100 (FTSE); DAX (GDAXI); S&P/TSX Composite (GSPTSE); Hang Seng Index (HSI); KOSPI Composite (KS11); Euronext 100 (N100); Nikkei 225 (N225); Shanghai Composite (SSEC); SMI (SSMI)
- **COMEX Futures:** Gold futures; Silver futures; Copper futures; Oil futures
- **FOREX Rates:** EUR-USD (Euro); GBP-USD (British Pound); USD-JPY (Japanese yen); USD-CNY (Chinese yuan)
- **Bond Rates:** US 5-year bond yield; US 10-year bond yield; Eurobund futures
- **Dollar Index:** Measures relative value of US dollar

As with gold prices, the absolute values of stock indices, bond rates, etc. mean very little without an understanding of current trends and the mechanics of their recent price movement. For instance, for Dow Jones index values to be useful for gold price prediction, the algorithm must first thoroughly understand, and to some extent predict, the movement of the DJI itself. With this in mind, I am applying the same technical indicators as outlined above with gold prices.

However, I am also adding two additional features per variable: (1) rate of change of volume, since daily volume was not available with the gold price fix data; and (2) rate of change from open price to close price on the current day, which gives deeper insight into the most recent price movement data that is available.

4.2 Results with Additional Features

I ran logistic regression (the same format as above, with 10-fold CV) with the full set of features calculated for both gold prices and all intermarket variables listed above. The table below displays the results:

Table 2. Logistic Regression (Full Feature Set)

Precision	63.76%
Recall	63.89%
Accuracy	61.92%

As shown above, adding extra features provided a major boost to the accuracy of the classifier: when I included the full feature set, accuracy jumped from 54.23% to 61.92%, and precision from 55.03% to 63.76%. These additional intermarket variables are thus very useful for predicting the rate of change in gold price.

4.3 Computing Correlation

As adding a multitude of features that actually have little or no effect on gold prices would add noise and negatively affect my predictions, I first want to quantify the correlation that

certain variables have with gold prices to determine whether they might be beneficial to include [3]. For all of the intermarket variables listed above, I calculated the Pearson correlation coefficient over the roughly 1700-day data period between (a) the London PM gold price fix on a given day, and (b) the close price of the variable. The table below displays all quantities by descending absolute-value correlation coefficient with the gold price fix:

Table 3. Correlation Coefficients

1. Gold Futures	0.72
2. Silver Futures	0.50
3. Copper Futures	0.27
4. EUR-USD	0.24
5. Dollar Index	-0.21
6. Oil Futures	0.21
7. S&P/TSX Composite	0.18
8. USD-CNY	-0.17
9. GBP-USD	0.16
10. KOSPI Composite	0.13
11. Hang Seng Index	0.11
12. FTSE 100	0.10
13. Nikkei 225	0.10
14. Shanghai Composite	0.09
15. USD-JPY	-0.08
16. DAX	0.07
17. CAC 40	0.07
18. Euronext 100	0.06
19. Eurobund	0.05
20. US 10-Year Bond	-0.02
21. US 5-Year Bond	-0.02
22. S&P 500	0.01
23. NASDAQ Composite	0.01
24. Dow Jones Industrial Average	0.01
25. SMI	0.01

The correlation values lead to a variety of very interesting observations, and here I discuss several salient ones. First, commodity futures are very highly correlated with the gold spot price, especially futures for gold itself.

Also, when the dollar is valued high, it takes comparatively less dollars to buy the same amount of gold, so the gold price (measured in dollars) will be low. For this reason, when variables which increase with the value of the dollar (i.e. the dollar index or the USD-CNY exchange rate) are high, the dollar is high so the gold price is low. Conversely, when variables which decrease with the value of the dollar (i.e. the EUR-USD exchange rate) are high, the dollar is low so the gold price is high.

In addition, every single stock index I tested exhibited a positive correlation with the gold price fix. Certain indices appear much more correlated than others: for instance, the Canadian S&P/TSX index and all of the Asian stocks were quite highly correlated. Interestingly, all three of the US stock indices I tested had an extremely low correlation with gold prices, with coefficients under 0.01.

4.4 Feature Selection Based on Correlation

Consider, for instance, the three US stock indices; these all have a very low correlation with gold prices, and thus are likely not helpful for gold price prediction. In fact, they may even add noise and cause the classifier to overfit the training data, leading to higher test error. Indeed, training and testing on the full data set with all of these features achieves a train accuracy of 81.69%, which is high compared to the test accuracy of 61.92%. Thus the variance of this model must be high relative to the previous model with gold price-only features. I suspected that if I could find a model somewhere in between these two, I could boost accuracy even further.

I decided to run feature selection on the intermarket variable features. To do so, I try adding all features associated with each intermarket variable in order of descending absolute-value of its correlation coefficient with gold prices. The goal is to continue adding features in a strategic order as long as they help rather than hurt accuracy. If the additional features increase accuracy, I keep them in the model; otherwise, I discard them.

The outcome of this experiment was extremely positive. Ultimately, the optimal feature set included 10 of the top 15 most correlated variables (of 25 total). These are:

- 4 commodities futures: gold, silver, copper, oil
- Dollar index
- 3 exchange rates: EUR-USD, GBP-USD, USD-CNY
- Hang Seng Index and Nikkei 225

The results are as follows:

Table 4. Logistic Regression (Optimal Feature Set)

Precision	69.90%
Recall	72.31%
Accuracy	69.30%

This illustrates a very significant increase in accuracy, from 61.92% when using the entire feature set to 69.30% when using this modified feature set. Moreover, precision and recall suggesting that the classifier is no longer strongly favoring one class over another (as in the initial tests, where there were disproportionately many false positives).

4.5 Varying the Algorithm

At this point, I consider that I have found a highly satisfactory feature set to work with, and briefly examine the effects of making changes in the algorithm to potentially obtain higher accuracy.

4.5.1 Revisiting SVM

Running SVM on the gold-price-only feature set was disappointing, as was running SVM on the feature set with all intermarket variables. However, a linear-kernel SVM with *l-1*-regularization actually performed quite well on the final feature set (normalized by scaling, as in the SVM section above), achieving accuracy of 69.08% with 10-fold CV. Apparently these new features provide the SVM just enough information to make predictions with accuracy very close to, but still not quite matching, that of logistic regression.

4.5.2 Logistic Regression

There are not too many parameters of the logistic regression model to experiment with, but I tested variations on (a) the norm used in penalization, and (b) the C parameter, which specifies inverse of regularization strength. I found that the optimal combination occurred with the *l-1* penalty and the default C = 1.0, precisely the parameters of the model in Table 4. Thus this model attains the highest accuracy, precision, and recall of anything I tried, making my overall peak classification accuracy 69.30%.

5. Trading Simulation

For practical purposes, predicting the direction of change is important primarily insofar as it leads to profitable decisions. To evaluate the success of the algorithm with regard to its potential profitability, I ran a small simulation that executes 1-day trades based on my algorithm's predictions:

- **Predict 1:** buy 1 ounce of gold
- **Predict 0:** sell 1 ounce of gold

I broke the set of training data up into 10 folds, and separately for each fold, I trained the classifier described above on all other examples, then simulate trading on the test set. The results my algorithm obtained are as follows:

Table 5. Simulation Results

Features	Average Daily Profit
1. Only gold features	$0.65
2. All optimal features*	$7.08

*The simulation with the 2nd feature set assumes that gold can still be purchased at the London PM price fix even after market close and all data has been obtained, which occurs several hours later. (I was not able to find reliable data for the gold spot price at market close, when the algorithm would be ready to make the buy/sell decision.)

I expect the algorithm's actual profit would fall somewhere between these two estimates. At any rate, my algorithm's predictions allow the trading agent to net a significant profit over the course of the simulation.

6. Conclusions

Ultimately, my algorithm was able to predict with a good degree of accuracy whether the following day's London PM gold price fix would be higher or lower than the current day's. I was surprised that the relatively simple model of logistic regression could achieve such high accuracy, even outperforming all variants of SVM that I tried.

Moreover, I was able to answer many of the questions I originally set out to explore. For instance, technical indicators are usually applied to stock prices or, in general, financial quantities for which daily high, low, open, and close prices are tracked. By making slight modifications (for instance, to the stochastic oscillator formula) and extending this logic to a quantity like the gold price fix, which is measured only once per day, I was still able to achieve the desired effect of understanding how prices have been moving.

I was also very curious about whether intermarket variables would be effective inputs for gold price prediction. The results demonstrate that they indeed were. A strategic combination of these variables based on the magnitude of their correlation with gold prices provided a major increase in accuracy and precision. Indeed, feature selection was a highly useful process in determining not only what intermarket variables I should use but also what specific technical indicators I should use. In general, adding features tended to help, but in other cases it led the model to overfit the training data, thus reducing test accuracy.

Finally, the simulation illustrates that this algorithm could actually have significant profitability. With a classification accuracy of nearly 70%, even the extremely simple trading agent set forth in the simulation was able to attain a considerable average daily profit.

Overall, these experiments have further piqued my interest in the applications of machine learning to financial prediction, as determining which features to add and which algorithms to use was quite interesting, and the end results were better than I expected. I would be interested in continuing to optimize this model and perhaps putting its predictions to practical use.

References

[1] Jan Ivar Larsen. *Predicting stock prices using technical analysis and machine learning*. NTNU, 2010.

[2] Vatsal H. Shah. *Machine learning techniques for stock prediction*. NYU, 2007.

[3] Shunrong Shen, Haomiao Jiang, and Tongda Zhang. *Stock market forecasting using machine learning algorithms*. Stanford University, 2012.