Blog Post #4

Thee Meensuk (tmeensuk), Petros Dawit (pdawit), Dikshyant Rai (drai), Kasemsan Kongsala (kkongsal)

Macroeconomic Scheme Reworked

After learning more about machine learning in class, we want to improve on our macroeconomic approach to predict Bitcoin prices using machine learning. Thus, we redesigned our label and feature set to find out how to best fit the model to reality.

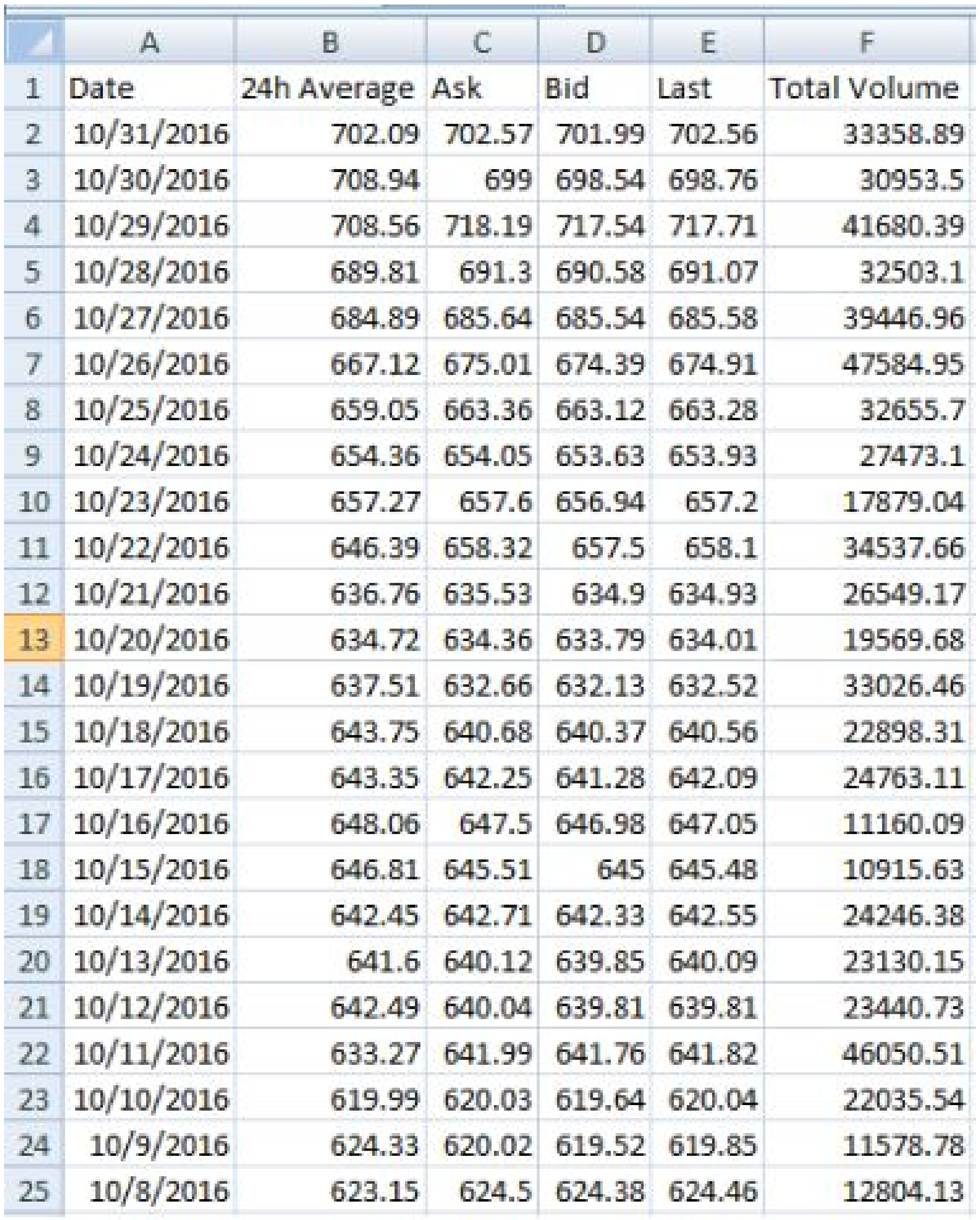
Model and Feature Optimization

Instead of randomly guessing a model and hoping that it will work well to predict Bitcoin prices, we want to use machine learning to figure out a good approach before proceeding to craft the real model. To accomplish this, there are two things we need to consider: the time span of past data and the types of assets we want to incorporate in our model. Since we are trying to use the past data to predict the future prices of Bitcoin, we want to consider how far we should go back in the past. Because Bitcoin is essentially a financial asset, too far back in the past will have little impact on the present or future, and using those data will count against us. On the other hand, only a few days prior to the date of prediction might not provide us with enough data to make good predictions. Thus, we expect the optimal time frame to be in the middle, not too short or too long. In order to find out how long, we try different time frames and compare the results.

The types of assets to be used as features are important as well. We have brainstormed and came up with 4 major assets whose prices will likely correlate with Bitcoin's. We want to use those assets as the main bases for features to be used in our machine learning model. However, we will also transform some of these features to make the models more realistic, which we hope will help increase the accuracy of our prediction.

Raw Data

The data extracted from https://www.quandl.com/ give us the historical Bitcoin prices in US Dollar (USD), Chinese Yuan (CNY), and Euro (EUR), as well as the interest rates for 10-year US Treasury Bonds and Gold Prices in USD. For the Bitcoin prices, the data goes daily from September 2013 to October 2016, daily. There are 365 days in a year, so we have about 1400 data points.



	A	B	C	D	E	F
1	Date	24h Average	Ask	Bid	Last	Total Volume
2	10/31/2016	702.09	702.57	701.99	702.56	33358.89
3	10/30/2016	708.94	699	698.54	698.76	30953.5
4	10/29/2016	708.56	718.19	717.54	717.71	41680.39
5	10/28/2016	689.81	691.3	690.58	691.07	32503.1
6	10/27/2016	684.89	685.64	685.54	685.58	39446.96
7	10/26/2016	667.12	675.01	674.39	674.91	47584.95
8	10/25/2016	659.05	663.36	663.12	663.28	32655.7
9	10/24/2016	654.36	654.05	653.63	653.93	27473.1
10	10/23/2016	657.27	657.6	656.94	657.2	17879.04
11	10/22/2016	646.39	658.32	657.5	658.1	34537.66
12	10/21/2016	636.76	635.53	634.9	634.93	26549.17
13	10/20/2016	634.72	634.36	633.79	634.01	19569.68
14	10/19/2016	637.51	632.66	632.13	632.52	33026.46
15	10/18/2016	643.75	640.68	640.37	640.56	22898.31
16	10/17/2016	643.35	642.25	641.28	642.09	24763.11
17	10/16/2016	648.06	647.5	646.98	647.05	11160.09
18	10/15/2016	646.81	645.51	645	645.48	10915.63
19	10/14/2016	642.45	642.71	642.33	642.55	24246.38
20	10/13/2016	641.6	640.12	639.85	640.09	23130.15
21	10/12/2016	642.49	640.04	639.81	639.81	23440.73
22	10/11/2016	633.27	641.99	641.76	641.82	46050.51
23	10/10/2016	619.99	620.03	619.64	620.04	22035.54
24	10/9/2016	624.33	620.02	619.52	619.85	11578.78
25	10/8/2016	623.15	624.5	624.38	624.46	12804.13

Label and Feature Transformation

From the raw data, we want to craft a set of features and label to train with machine learning. For the label, we will just want to classify whether or not the price of Bitcoin on a given day is higher than the day before. We give 1 for higher, -1 for lower, and 0 for the same. Theoretically we should get mostly 1 and -1, because having the same price of Bitcoin today as yesterday would mean that given the price yesterday, the price today has to be exactly equal to that value, out of more than a million possibilities. Having 1 and -1 for label should not be much different from 1 and 0, and having a few 0's thrown into the mix of 1 and -1 should not be much different from purely 1 and -1. Thus, we chose 1, 0, -1 for our labels, to make it intuitive (1 for higher, -1 for lower, and 0 for no change).

To get features, we will consider each asset individually, and one additional case where we use all the 4 assets to do the machine learning. First, for the Chinese Yuan, CNY, the raw data has the price of one Bitcoin in terms of CNY. However, this feature contains two sets of information both in itself: the exchange rate between CNY and USD, or CNY/USD Exchange rate, and the value of Bitcoin in USD. In other words, <CNY/BTC> = <CNY/USD> * <USD/BTC>. Thus, if the value of Bitcoin appreciates (which it did), while the exchange rate fluctuates within a small range, the resulting CNY/BTC will go up wildly as the <USD/BTC> part goes up. However, we want to find features to predict the label. Our labels, +1 if the Bitcoin price goes up, and -1 if the price goes down, should not depend on whether or not the absolute price of Bitcoin is high or low. If that were to be the case, since the price of Bitcoin has appreciated sharply in the recent years, the more recent prices would all indicate one direction of price change, and almost all prices in the past would indicate the other direction. That means our model would be predicting decreases in Bitcoin prices every day in the long past up until a point in time when the model switches to predicting increases in prices every day until today.

To eliminate the <USD/BTC> part that represents the absolute value of Bitcoin (in USD), we can divide <CNY/BTC> on each day by <USD/BTC> of the same day. This leaves us with only <CNY/USD>, which could be a good predictor of whether or not the price of Bitcoin in USD would go up or down on a particular day. To think about economic realities, the CNY/USD represents the exchange rate between USD and CNY. If CNY/USD decreases, it means $1 is equivalent to less CNY, or that the Dollar has depreciated. If the Dollar depreciates, $1 should be able to buy fewer Bitcoins. Conversely, 1 BTC will be equivalent to more Dollars. That means higher USD/BTC. Given this logic, it makes sense that we would be able to predict whether or not USD/BTC will increase or decrease just by looking at CNY/USD, calculated from <USD/BTC> and <CNY/BTC>.

Now that we have extracted the CNY/USD out from the raw data, we also want to include the concept of “time” into our feature. Currently there is no “time” in our CNY/USD data, but we want to predict the future from the past. Thus, we include time into our feature by creating more CNY/USD at different times. As a result, we will have CNY/USD at 1day before today, 2days before, 3days before, and so on. Because we are running machine learning with different time frames, the number of features will vary with the length of our time frame. For example, if we are to do the 40-day time frame case, there would be 40 features representing each of the 40 previous days of CNY/USD exchange rates. However, the corresponding label remains the USD/BTC price.

The next case is to predict the same set of labels using EUR/USD on previous days. The process is the same as in the CNY/USD case except that we change CNY to EUR.

Next is to use the 10-year US Treasury Yield on previous days. The Treasury Yield is straightforward, so we don’t need to do extra feature transformation here.

Finally, we use the Gold price in USD/Ounce. The Gold price is tricky. If 1000 USD/Ounce is the norm, having the price increase to 1050 is a price increase in Gold. Since Gold and Bitcoin are similar assets that are competing for money from investors, Gold price increasing would mean Bitcoin price decreasing, so we could predict the label this way. However, if 1500 USD/Ounce is the norm, the price of 1050 would be a total disaster, and we expect not the Bitcoin price to spike up as Gold plummets. Since the price of gold is largely stable, and we only consider the time span of 3 years, we will assume that the norm is constant, and assume it is sufficient to simply use the plain gold price in our machine learning.

Finally, we combine every feature into one batch of ML. That is, in the first 4 parts we predict labels from only one of CNY/USD, EUR/USD, 10-year Treasury Interest Rate, and Gold. If we are considering the 30-day learning period, we will have 30 features, each corresponding to each of the previous 30 days. In this final machine learning attempt, if we consider the 30-day period, we will have 120 features, broken down into 30 CNY/USD, 30 EUR/USD, 30 10-year Treasure Interest Rates, and 30 Golds.

Feature Scaling

We also want to scale the features down to carry similar weights. Ideally we would find out the lowest value of each feature and set it to 0, and the highest value to 1. After that we would interpolate the intermediate values. Because the nature of the financial variables makes it possible for new lowest and highest values to appear when we include new information like spanning across more dates, and it will not be more generalizable to scale from 0 to 1 based only on the training data that we consider. Thus, we think it will not make much difference in terms of accuracy and reliability to scale the numbers down using just rough estimations, instead of precise scaling from 0 to 1. Therefore, we decide to divide the CNY/USD feature values by 6, EUR/USD by 0.75, Interest Rate by 10, and Gold by 1000.

Model's Accuracy Scores

We did machine learning with three different models: Naive Bayes, Logistic Regression, and SVM (Support Vector Machine).

Using the aforementioned features, we trained our models with different sets of features described above. Then we tested the trained models against the original data labels and arrived at the following accuracy results. As a side note, since we are testing the model with the training data, the accuracy scores will be unusually high, since we are overfitting the model. However, the difference should not be significantly high enough. Later we will test it with another set of data and report updated accuracy scores.

Naive Bayes

Feature Set	CNY/USD Alone	EUR/USD Alone	10-year US Treasury Yield (%) Alone	Gold ($/Ounce) Alone	All of them combined
10 Days	46.76%	46.76%	47.95%	48.31%	48.94%
20 Days	46.73%	46.73%	47.52%	47.49%	48.80%
30 Days	46.70%	46.70%	47.70%	47.47%	48.79%
40 Days	46.76%	46.76%	47.99%	47.23%	48.84%
50 Days	46.92%	46.92%	47.97%	47.52%	48.97%
60 Days	47.08%	47.08%	48.70%	47.92%	49.97%
70 Days	47.42%	47.42%	49.01%	48.54%	49.65%
80 Days	47.68%	47.68%	45.22%	45.81%	49.92%

Logistic Regression

Feature Set	CNY/USD Alone	EUR/USD Alone	10-year US Treasury Yield (%) Alone	Gold ($/Ounce) Alone	All of them combined
10 Days	54.03%	55.44%	54.60%	54.60%	58.04%
20 Days	54.24%	55.94%	54.85%	54.44%	58.11%
30 Days	54.55%	55.09%	55.74%	54.17%	57.25%
40 Days	54.32%	55.50%	56.85%	57.76%	57.91%
50 Days	54.45%	55.46%	56.82%	57.63%	57.72%
60 Days	54.95%	55.51%	55.92%	57.18%	58.28%
70 Days	55.75%	56.50%	54.79%	56.94%	57.60%
80 Days	57.41%	57.03%	54.41%	56.25%	58.79%

Support Vector Machine (SVM)

Feature Set	CNY/USD Alone	EUR/USD Alone	10-year US Treasury Yield (%) Alone	Gold ($) Alone	All of them combined
10 Days	57.39%	55.62%	54.80%	54.49%	59.63%
20 Days	56.60%	56.38%	55.05%	54.54%	59.76%
30 Days	57.03%	56.71%	54.87%	54.69%	61.22%
40 Days	58.48%	57.54%	55.41%	56.30%	61.86%
50 Days	58.49%	57.49%	55.21%	57.32%	63.35%
60 Days	59.94%	58.02%	54.49%	56.86%	63.79%
70 Days	58.71%	58.28%	54.36%	58.01%	63.53%
80 Days	60.48%	58.79%	53.19%	55.93%	63.51%

Using coin tosses to predict the Bitcoin price performance would have resulted in approximately 50% accuracy. Thus, any score above 50% is better than a coin toss. We don't expect any score higher than 70% because any model with such high accuracy and simplicity should have been discovered by someone else long before, and that person must have exhausted most profit opportunities to the point that there is little economic benefit left for us. If any score higher than 70% shows up, we expect to surpass Bill Gates' net worth within 10 years.

As the results show, the Naive Bayes yield horrible results in predicting the price change of Bitcoin. All accuracy scores are below 50%, meaning that our Naive Bayes models underperform even chimpanzees throwing darts to predict the change in price. In addition, when computing using the CNY/USD and EUR/USD, our model had trouble with some data points. When we investigated, we found that this problem was caused by certain probabilities being too low and causing negative values inside logarithmic operations that the model used to train. This discovery strengthens our conclusion that the Naive Bayes model is not suitable for our feature set to predict the Bitcoin price.

The Logistic Regression and SVM models perform better, with accuracy scores ranging from 54% to as high as almost 64%. Generally the SVM model performs better than the Logistic Regression model. The typical pattern is that the accuracy score goes up when we consider a longer time window, up until a point when the accuracy drops as we increase the length. Individual feature sets perform differently with the CNY/USD and EUR/USD being generally superior to the other two feature sets (10-year US Treasury Interest Rate and Gold). However, when we combine all the features, the accuracy scores go up to as high as 63.79% in the SVM case. This means that out of 10 guesses, we get about 3.6 times wrong and 6.4 times right. This accuracy will be more than enough for a professional algorithmic trader to make splendid living.

Plan on Additional Work

Now that we have done machine learning and computed accuracy scores of each case in our Macroeconomic Scheme, we know that the best approach is to combine all 4 feature sets and have a time frame of around 50-60 days. Right now we have the SVM with 60-day time frame that can provide an accuracy score of 63.79%. However, we can only tell whether or not the price will go up or down. We want to use what we learned here to craft out a predictive model that can tell us to at least 2 decimal places the expected future price of Bitcoin, using just past data. We plan to do that with Artificial Neural Networks and are investigating how to implement it.

Furthermore, we want to improve on our current ML implementation of the Macroeconomic Scheme's 1/0/-1 label classification by:

1. Consider another feature, which is the USD/BTC's historical prices. Originally, we thought that since we are doing the Macroeconomic approach, we should use the prices of other financial assets to predict USD/BTC price. However, as we brought the concept of time into our model, we found that it will be possible to use USD/BTC's previous day prices in our model, and adding that feature should help improve our accuracy score.
2. Scaling. Right now we are doing only rough scaling of feature values by dividing them by 'beautiful' hard-coded numbers. In the future we want to do a better job at scaling by turning the lowest and highest values to 0 and 1, even thought it might not make much difference.
3. Change Gold Feature. Right now we are using plain Gold prices in the machine learning. We plan to change it to 0 and 1, with 0 being the price has decreased from the benchmark time, and 1 being the price has increased. We are not sure how much impact this change will have on the accuracy of our model, and we want to find out.

Future Work: Trading Trend Approach

In addition, we plan on revising our trading trend approach to get more meaningful data. Last time, our only feature we looked at was the past prices over a set amount of days to used ML to predict the current price. We ended up with nearly 55% accuracy, but we've realized that we can better our results by looking at more features and covering a larger span of days.

We could also have better visualization and compare the accuracy (numbers of days correctly classifying the testing data/ total number of testing days) for the different amount of combinations and different spans of days.

To do this, with our data, we will probably use 30% of it as testing data. We will have the features we are looking at be the current weighted price (USD), low/high price, total volume, transaction fees, bitcoin hash rate, number of bitcoin transactions, bitcoin volume. We will compare the results with different combinations of the features and with different MLs (SVM, Bayes, Log) along with days before the current price of 2, 5, 10, 25, 50, 100, 200. We will, if the results show, go ahead and visualized in the form of graphs a linear regression line on number of days vs accuracy and the number of features vs accuracy on the different MLs.