Cryptocurrency Price Prediction

Maxwell Greene

December 14, 2018

Abstract

1 Introduction

1.1 Background

Cryptocurrency is a digital form of currency that has become increasingly popular in the past few years. Market prices are determined by trade values between individuals, which can vary with quite high volatility as supply and demand change. Data had become available on trade history and market prices over the past few years. The transparancy of the market value and trade prices make the price changes somewhat predictable over time. That said, it is possible to build a predictive model, trained on this historic data and the market listings, that can accurately predict future price changes.

1.2 Problem Description

Due to the emotional nature of trading any assets, such as stocks, FOREX, etc. it can be decievingly difficult to make a profit off of initial investments. I believe it is possible to avoid this problem by constructing a predictive model based solely on historic market prices and market listings.

1.3 Objectives

To build a predictive model of cryptocurrency prices using the ‘crypto‘ R package that provides historic and live cryptocurrency prices for a wide variety of coins.

1.4 Data Description

The data provided by the ‘crypto‘ package is 24h trade volumes, open, close, high and low values for the specified market since 2010.

2 Loading Data and Formatting

First load all necessary libraries, as well as the .RData file containing all data.

```
library(tidyverse);
library(crypto);
library(zoo);
```

2.1 Import Data

I will only load the functions I have created in this document, rather than all the prepared data, so that there are no conflicts when I recreate certain variables. The file "Functions.RData" loads all functions from the files "gatherData.R", "mutateData.R", "plotData.R" and "modelData.R".

```
load("Functions.RData")
```

Import data with ‘crypto‘ package using my function ‘GetCoinHistory()‘ from the ‘gatherData.R‘ file:

```
data <- GetCoinHistory("ETH");
```

2.2 Organize Data Using my .R Functions

Add 10 and 50 day moving average of open prices with my function ‘mutateRoll()‘ from the ‘mutateData.R‘ file:

```
data <- mutateRoll(data,data$open,c(5,10,25))
```

Add Yesterday’s open and close price to data with my function ‘mutateLast()‘ from the ‘mutateData.R‘ file:

```
data <- mutateLast(data,data$open,"openLast")
data <- mutateLast(data,data$close,"closeLast")
data <- mutateLast(data,data$high,"highLast")
data <- mutateLast(data,data$low,"lowLast")
data <- mutateLast(data,data$volume,"volumeNext")
```

Add Tomorrow’s open price to data with my function ‘mutateNext()‘ from the ‘mutateData.R‘ file:

```
data <- mutateNext(data,data$open,"openNext")
data <- mutateNext(data,data$close,"closeNext")
data <- mutateNext(data,data$high,"highNext")
data <- mutateNext(data,data$low,"lowNext")
data <- mutateNext(data,data$volume,"volumeNext")
```

Now, convert all applicable columns to ratios with my function ‘convertRatio()‘ from the ‘mutateData.R‘ file:

```
columnNames <- c("close","low","high","openLast","closeLast",
                 "highLast","lowLast","openNext","closeNext",
                 "highNext","lowNext","ma1","ma2","ma3")
dataRatio <- convertRatio(data,data$open,columnNames)
```

Lastly, let’s add an up/down factor column so that we can fit a classification model.

```
dataRatio <- mutate(dataRatio,
              closeNextBool=ifelse(closeNext>close,TRUE,FALSE))
```

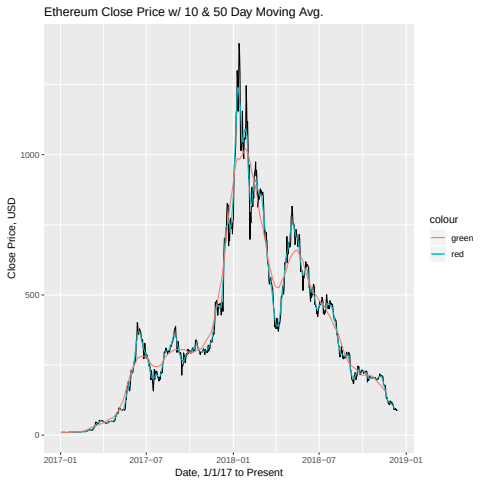
3 Visualizing Data

I have written four plotting functions in my ‘plotData.R‘ file: ‘plotOpen()‘, ‘plotClose()‘, ‘plotHigh()‘, ‘plotLow()‘

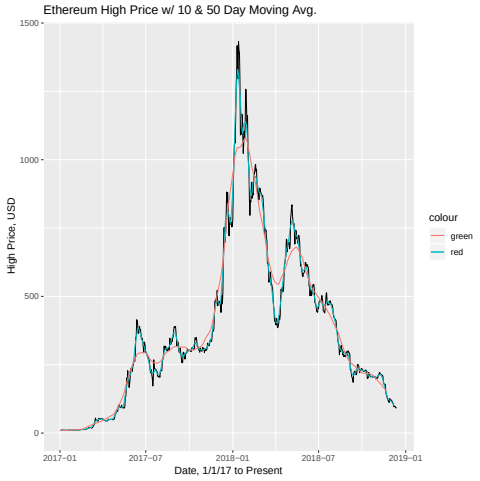
3.1 Moving Forward w/ Model

Now that the data has moving averages as well as past and previous values for open and close, it can be implemented in the form of a model. Yesterday’s open, yesterday’s close, today’s open, today’s close, historically the open and close of the next day, 10-day moving averages and 50-day moving averages can now be predictors used in the model, which could have a variety of useful outputs in trading. Namely, the difference between tomorrow’s open and close values would be a useful prediction for traders, because it would indicate a predicted large shift in the market.

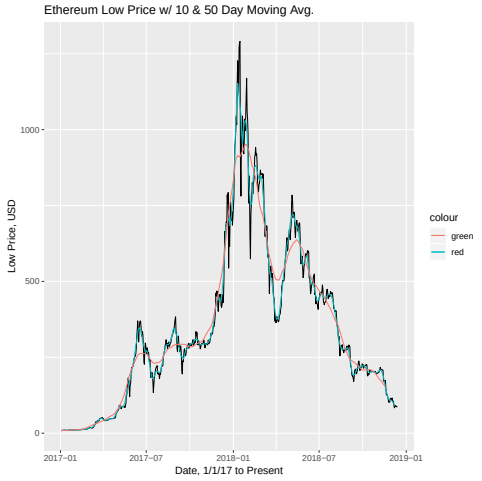
(a) Open



(b) Close



(a) High



(b) Low

4 Model Creation

4.1 Classification Models

First make test and train sets. The formula involving 'closeNextBool' will predict whether tomorrow's close price wll be higher than today's close price.

```
trainData <- sample_n(dataRatio,nrow(dataRatio)*0.8)
testData <- sample_n(dataRatio,nrow(dataRatio)*0.2)

formula <- as.formula("closeNextBool~open*close*ma1*high*low*volume")
lda.fit <- lda(formula,testData)
table<-table(testData$closeNextBool,
             predict(lda.fit,testData)$class)
table

##
##         FALSE TRUE
##   FALSE   100   28
##   TRUE     37   80
```

Unfortunately, this model is not very accurate, showing an accuracy of

```
sum(diag(table))/sum(table)*100

## [1] 73.46939
```

4.2 Regression Models

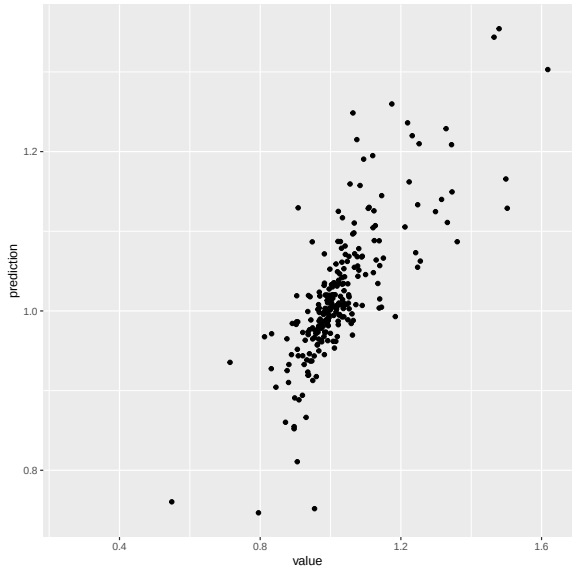
Recreate training and testing datasets.

```
trainData <- sample_n(dataRatio,nrow(dataRatio)*0.8)
testData<- sample_n(dataRatio,nrow(dataRatio)*0.2)

formula <- as.formula("closeNext~open*close*ma1*volume")
glm.fit <- glm(data=trainData,formula=formula)
mean(predict(glm.fit,testData)/testData$closeNext,na.rm=TRUE)

## [1] 0.9990149
```

```
ggplot(data=data.frame(
    prediction=predict(glm.fit,testData),
    value=testData$closeNext)) +
  geom_point(mapping = aes(x=value,y=prediction))
```



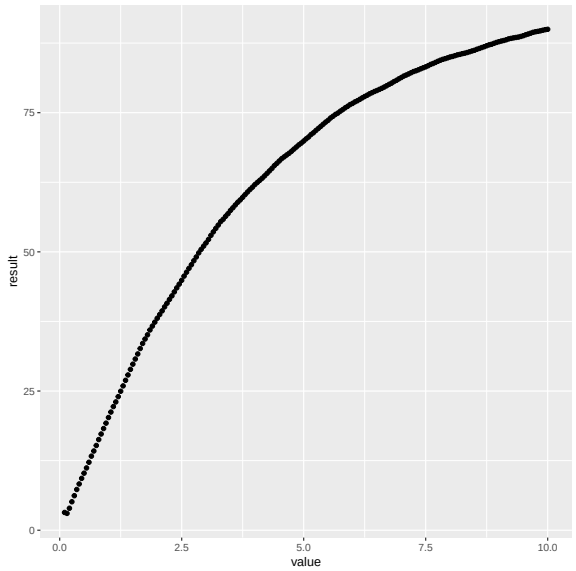
5 Model Evaluation

5.1 Visualizing Accuracy

The following graph is a visual representation of the accuracy of the model, showing the percentage of predictions within the given percentage of the correct test data value. This was done using my function 'glmVect()' in the file 'modelData.R'. Note: The function 'glmPlot()' in the same file performs the same computations, but saves a .png file of the plot, rather than returning a vector.

```
formula <- as.formula("closeNext~open*close*ma1*volume")
percents <- seq(.1,10,.05)
result <- as.vector(
  glmVect(1000,200,percents,formula,forNum=100,data=dataRatio))
```

```
ggplot(data=data.frame(result=result,value=percents)) +
  geom_point(mapping = aes(x=value,y=result))
```



The following is the same plot format as above, except using a compilation of ten different different formulas. In this case, the formulas are just each variable isolated.

```
  formchar1    <- 0
  formchar1[1] <- "closeNext~open"
  formchar1[2] <- "closeNext~high"
  formchar1[3] <- "closeNext~low"
  formchar1[4] <- "closeNext~close"
  formchar1[5] <- "closeNext~volume"
  formchar1[6] <- "closeNext~market"
  formchar1[7] <- "closeNext~close_ratio"
  formchar1[8] <- "closeNext~spread"
  formchar1[9] <- "closeNext~ma1"
  formchar1[10]<- "closeNext~ma2"

resultsdf <- formulaListGLM(formchar1)
```

```
ggplot(data=resultsdf) +
  geom_point(mapping = aes(x=percents,y=results,color=formula))
```

6 Conclusion

References

[1] Yukun Liu and Aleh Tsyvinski. Risks and returns of cryptocurrency. Technical report, National Bureau of Economic Research, 2018.