Stat 107: Introduction to Business and Financial Statistics
Homework 7: Due Friday, November 18th

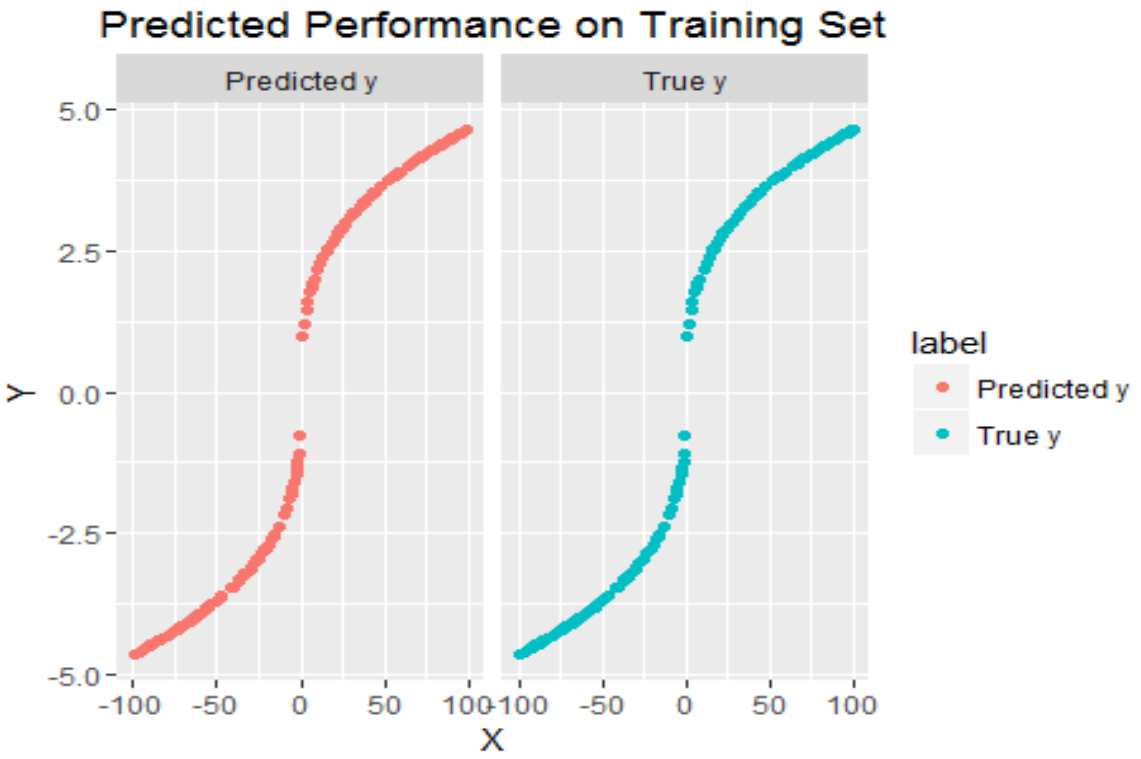
Xiner Zhou

November 17, 2016

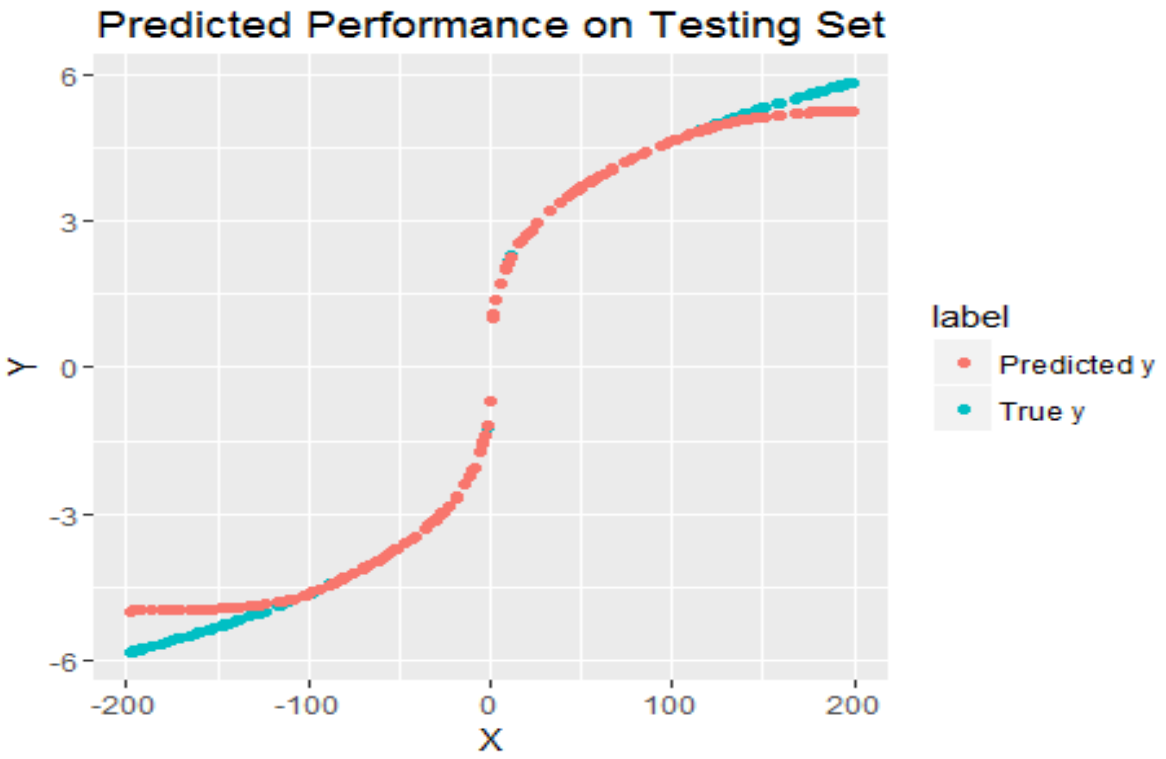
(1)

Use the neural net routine in R shown in the class notes to learn the $y = x^{(1/3)}$ function. Generate training data including both positive and negative x values, then show how it does on new data.

```
library(neuralnet)
library(ggplot2)

# Create training set:
# Generate 200 random numbers uniformly distributed between -100 and 100 as x
# Take the cube root of x as y
x<-runif(200,min=-100,max=100)
y<-sign(x)*abs(x)^(1/3) # cube root of negative number doesn't work for all c
ases

# Store x and y as a dataframe
training<-data.frame(x,y)


# Train the neural network with 10 hidden layers
neural.fit<-neuralnet(y~x,data=training,hidden=10,threshold=0.001,stepmax=1e6
)

# plot the neural net
plot(neural.fit)

# Visualize predictive performance on training set
training.result<-data.frame(x1=c(x,x),y1=c(y,neural.fit$response),label=c(rep
("True y",200),rep("Predicted y",200)))

ggplot(training.result,aes(x=x1,y=y1,color=label))+
  geom_point()+
  facet_grid(.~label)+
  ggtitle("Predicted Performance on Training Set")+
  xlab("X")+
  ylab("Y")
```




```
# Test the Network
test.x<-runif(200,min=-200,max=200)
neural.predicted.y<-compute(neural.fit, test.x)
testing.result<-data.frame(x1=c(test.x,test.x), y1=c(sign(test.x)*abs(test.x)
^(1/3),neural.predicted.y$net.result), label=c(rep("True y",200),rep("Predict
ed y",200)))

# Visualize predictive performance on training set
testing.result<-data.frame(x1=c(test.x,test.x), y1=c(sign(test.x)*abs(test.x)
^(1/3),neural.predicted.y$net.result), label=c(rep("True y",200),rep("Predict
ed y",200)))

ggplot(testing.result,aes(x=x1,y=y1,color=label))+
  geom_point()+
  ggtitle("Predicted Performance on Testing Set")+
  xlab("X")+
  ylab("Y")
```



The neural net predicts extremely well on the training set, and on indepedent testing set within the interval of the training input, but not very satisfactory on values out of the range of training input.

(2)

This problem is a little open ended. Read about volume weighted moving averages here:(and also read the R help for function VMA)

Using data on SPY from 2015-01-01, implement the buy/sell rules discussed in the article:

When **VMA** - (5-day) > **A** then we BUY.
When **VMA** - (5-day) < **B** then we SELL.

For values of A and B that you decide on, run your code to see how this rule does from 2015-01-01 to present. Extra credit if one wants to optimize the values of A and B but a simple run through using values picked from a visual chart inspection is fine.

Since we have no idea how exactly the differnece between VWMA and 5-day MA would look like, I propose that we use 3-year prior's daily return to get the historial distribution first. Then, we let A run through the selected values from minimum given by historical difference to 0, and let B run through the selected values from 0 to maximum given by historical difference.

```
library(quantmod)
library(TTR)

# Function that finds the As and Bs
FindAB<-function(ticker) {

  startdate="2012-01-01"
  todate="2014-12-31"
  malength=200
  #ticker="LCI"

  stockdata=getSymbols(ticker,from=startdate,to=todate, auto.assign=FALSE)
  ndays=nrow(stockdata)

  # 200day MA need 200 days of data to compute, so we go back and get more (2
00-1) trading days' data
  # But the problem is we don't know how many days need to go back
  thetimes=time(stockdata)
  firstdate=thetimes[1]
  goback=thetimes[1]-300

  morestockdata=getSymbols(ticker,from=goback,to=todate, auto.assign=FALSE)

  # Let 200th entry be our startdate
  stockdata<-morestockdata[(which(time(morestockdata)==firstdate)-(malength-1
)):nrow(morestockdata),]

  # Volumn weighted moving average
  vwma<-VWAP(price=Ad(stockdata), volume=Vo(stockdata), n=malength)
```

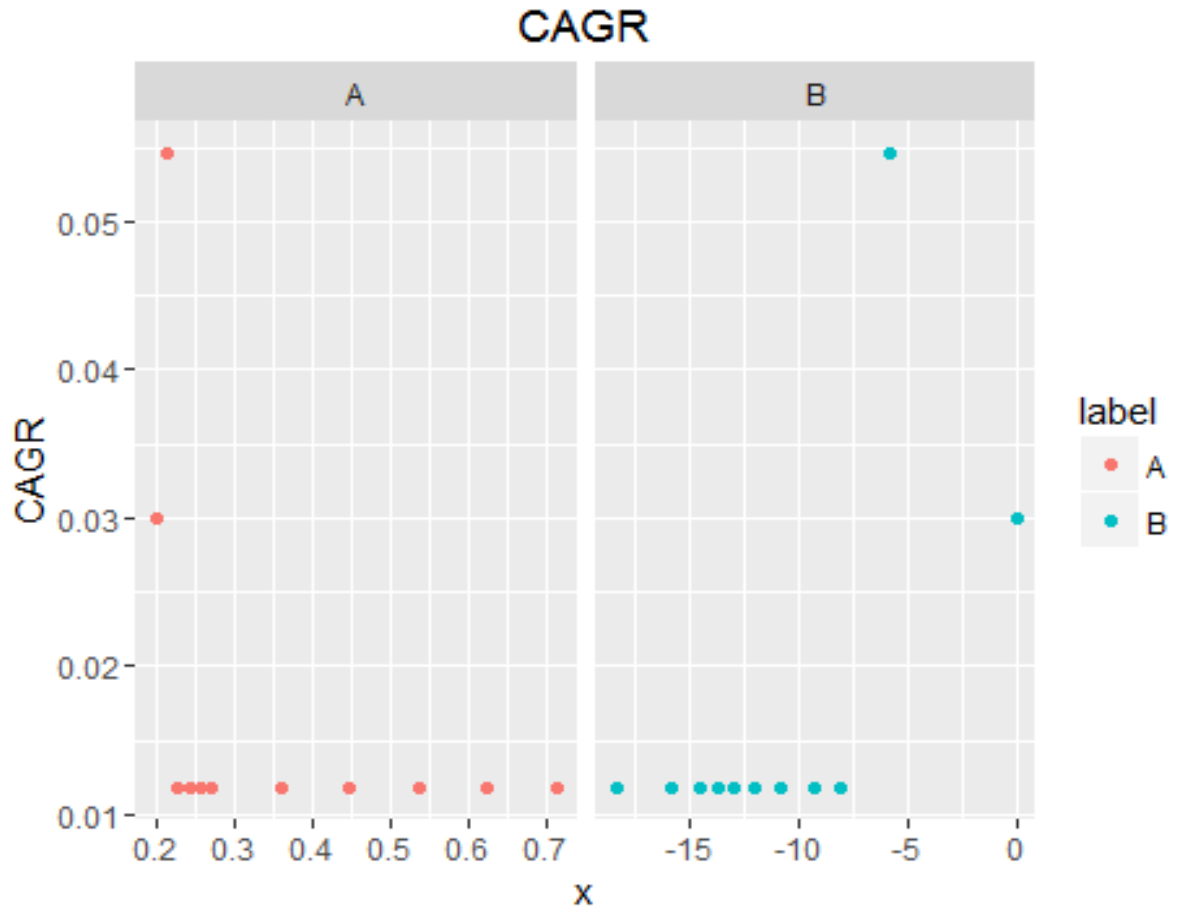
```
  # 5-day moving average
  ma5day<-SMA(Ad(stockdata),n=5)

  # calculate the diff VWMA-5-day MA
  diff<-data.frame(diff=as.numeric(vwma)[-c(1:200)]-as.numeric(ma5day)[-c(1:2
00)])

  # A: upper percentiles of positive difference
  # B: lower percentiles of negative difference
  A<-quantile(diff$diff[diff$diff>0],probs=c(seq(0,1,0.1)))
  B<-quantile(diff$diff[diff$diff<0],probs=c(seq(1,0,-0.1)))

  return(data.frame(A,B))
}
AB<-FindAB("SPY")
```

Second, we let A run through the selected values from minimum given by historical difference to 0, and let B run through the selected values from 0 to maximum given by historical difference, try every combination of A and B, calculate the CAGR for "2015-01-01" up till now, and find the combination of A and B that maximize CAGR.

```
VWMA200<-function(ticker,A,B) {

  startdate="2015-01-01"
  malength=200
  #ticker="LCI"

  stockdata=getSymbols(ticker,from=startdate,auto.assign=FALSE)
  ndays=nrow(stockdata)

  # 200day MA need 200 days of data to compute, so we go back and get more (2
00-1) trading days' data
  # But the problem is we don't know how many days need to go back
  thetimes=time(stockdata)
  firstdate=thetimes[1]
  goback=thetimes[1]-300

  morestockdata=getSymbols(ticker,from=goback, auto.assign=FALSE)

  # Let 200th entry be our startdate
  stockdata<-morestockdata[(which(time(morestockdata)==firstdate)-(malength-1
)):nrow(morestockdata),]
  # stock prince
  sp<-as.numeric(Ad(stockdata))

  # Volumn weighted moving average
  vwma<-VWAP(price=Ad(stockdata), volume=Vo(stockdata), n=malength)

  # 5-day moving average
```

```
  ma5day<-SMA(Ad(stockdata),n=5)

  # on the first day, don't have to buy if no signal
  # but in order to get CAGR, we have to out of stock by the end of the perio
d

  signal="inCash"
  buyprice=0
  sellprice=0
  mawealth=1
  #cat("Buy Price = ",buyprice )

  for(d in (malength):ndays) {
    if((vwma[d]-ma5day[d]>A) && (signal=="inCash")) {
      buyprice=sp[d]
      signal = "inStock"
      #cat("Buy Price = ",buyprice, '\n' )
    }

    if(((vwma[d]-ma5day[d]<B) || (d==ndays)) && (signal=="inStock")) {
      sellprice=sp[d]
      signal = "inCash"
      mawealth=mawealth*(sellprice/buyprice)
      #cat("Sell Price = ",sellprice, '\n'  )
    }
  }

  #bhwealth=sp[ndays]/sp[malength]

  # Need to determine number of years and calculate CAGR from total return
  nyear<-ndays/252
  # return CAGR
  return(mawealth^(1/nyear)-1)
  #print(paste("VWMA CAGR = ",mawealth^(1/nyear)-1))
  #print(paste("BH CAGR = ",bhwealth^(1/nyear)-1))
}

# Initiliza a place holder for CAGR
CAGR<-rep(NA,11)

# loop through A and B
for(i in 1:11){
    CAGR[i]<-VWMA200("SPY",A=AB$A[i],B=AB$B[i])
}


# Visualize and pick the highest CAGR
results<-data.frame(y=c(CAGR,CAGR), x=c(AB$A,AB$B), label=c(rep("A",11),rep("
B",11)))
```

```
ggplot(results, aes(x,y,color=label))+
  geom_point()+
  facet_grid(.~label,scales="free_x")+
  ggtitle("CAGR")+
  ylab("CAGR")
```



The one combinations of A and B gives the highest CAGR which is 5.46%, is: A=0.2144 and B=-5.8045. The way we find A and B depends on the stock we pick and the period we look at, therefore, we can ultimately write an algorithm that accommodates individual stock and time frame with different thresholds of A and B.

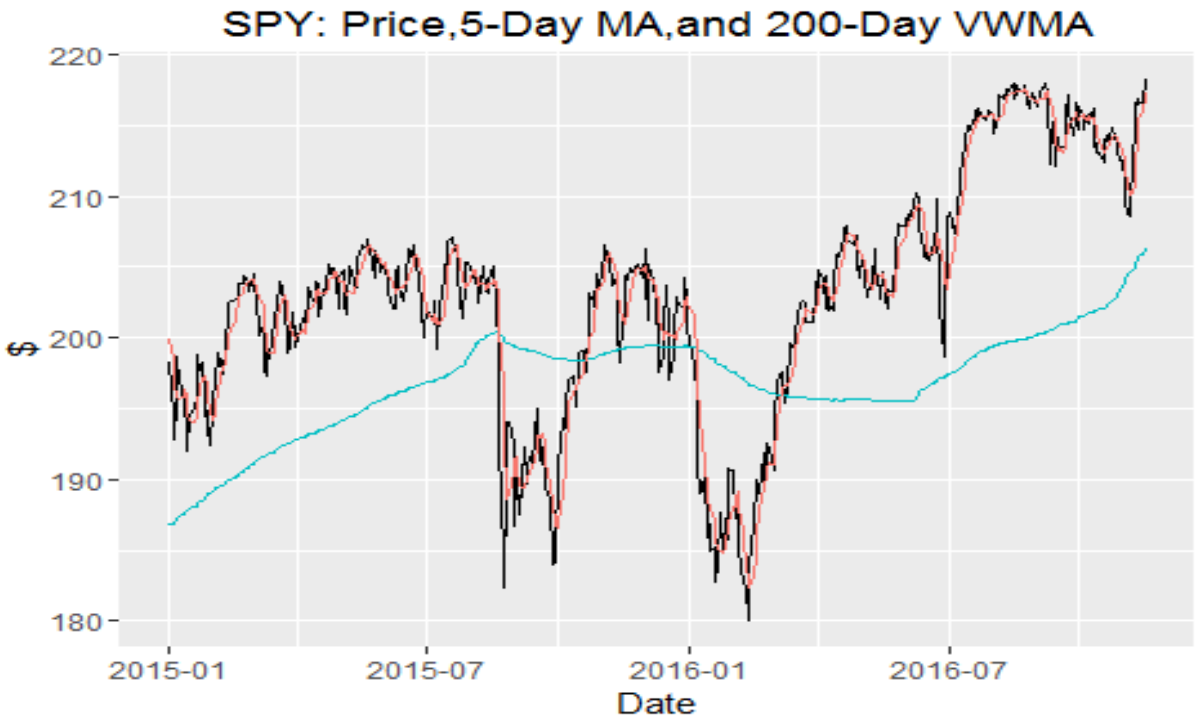
Using the A and B we pick, visualize the SPY stock price, 5-day moving average, 200-day volumn weighted moving average, and buy&sell signals.

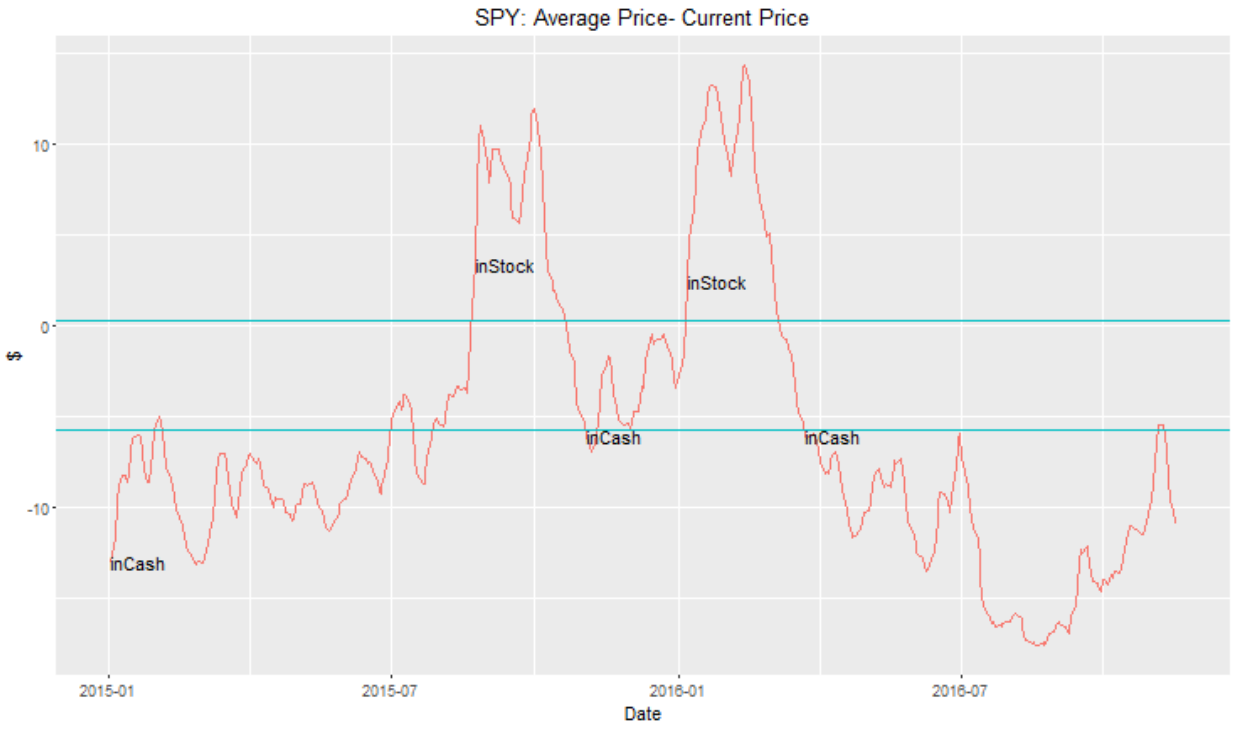
```
startdate="2015-01-01"
malength=200
ticker<-"SPY"
stockdata=getSymbols(ticker,from=startdate,auto.assign=FALSE)
ndays=nrow(stockdata)

# 200day MA need 200 days of data to compute, so we go back and get more (200
```

```r
-1) trading days' data
# But the problem is we don't know how many days need to go back
thetimes=time(stockdata)
firstdate=thetimes[1]
goback=thetimes[1]-300

morestockdata=getSymbols(ticker,from=goback, auto.assign=FALSE)

# Let 200th entry be our startdate
stockdata<-morestockdata[(which(time(morestockdata)==firstdate)-(malength-1))
:nrow(morestockdata),]
# stock prince
sp<-as.numeric(Ad(stockdata))

# Volumn weighted moving average
vwma<-VWAP(price=Ad(stockdata), volume=Vo(stockdata), n=malength)

# 5-day moving average
ma5day<-SMA(Ad(stockdata),n=5)

# Make a data frame contains: adjusted stock price, vwma, sma, buy&sell signa
l
plot.this<-data.frame(time=time(stockdata), price=Ad(stockdata), vwma, ma5day
,vwma-ma5day, signal=rep(NA,nrow(stockdata)) )

signal="inCash"
plot.this$signal[malength] ="inCash"
buyprice=0
sellprice=0
mawealth=1

for(d in (malength):ndays) {
    if((vwma[d]-ma5day[d]>AB$A[2]) && (signal=="inCash")) {
      buyprice=sp[d]
      signal="inStock"
      plot.this$signal[d] = "inStock"
      cat("Buy Price = ",buyprice, '\n' )
    }

    if(((vwma[d]-ma5day[d]<AB$B[2]) || (d==ndays)) && (signal=="inStock")) {
      sellprice=sp[d]
      signal= "inCash"
      plot.this$signal[d] = "inCash"
      mawealth=mawealth*(sellprice/buyprice)
      cat("Sell Price = ",sellprice, '\n'  )
    }
  }
```

```
## Buy Price =  184.546192
## Sell Price =  206.552431
## Buy Price =  191.090249
## Sell Price =  188.677621

nyear<-ndays/252
CAGR<-mawealth^(1/nyear)-1

plot.this<-plot.this[c(200:nrow(stockdata)),]

ggplot(plot.this)+
  geom_line(aes(x=time,y=SPY.Adjusted))+
  geom_line(aes(x=time,y=VWAP,col="red"))+
  geom_line(aes(x=time,y=SMA,col="purple"))+
  theme(legend.position="none")+
  ggtitle("SPY: Price,5-Day MA,and 200-Day VWMA")+
  xlab("Date")+ylab("$")
```

```
ggplot(plot.this)+
  geom_line(aes(x=time,y=VWAP.1,col="red"))+
  geom_hline(aes(yintercept = AB$A[2], colour ="tile"))+
  geom_hline(aes(yintercept = AB$B[2], colour ="tile"))+
  theme(legend.position="none")+
  ggtitle("SPY: Average Price- Current Price")+
  xlab("Date")+ylab("$")+
  geom_text(aes(x=time,y=VWAP.1,label=signal))
```


SPY: Price,5-Day MA,and 200-Day VWMA
220
210
200
190
180
$
2015-01
2015-07
2016-01
2016-07
Date


SPY: Average Price- Current Price
10
0
-10
$
inStock
inStock
inCash
inCash
inCash
2015-01
2015-07
2016-01
2016-07
Date