Yu Gao a13937873

Fordham  2016.5.7

machine lerning project

MACD and SVM - machine learning application on finance

**Abstract**

The model in this paper uses SVM to classify data processed by MACD which is an index used in stock pitch explained in next part. Then predict ups and downs given the test data to test the model’s validation. At last applying the prediction to transaction to calculate the cumulative return and maximum drawback ratio. Although the result is not satisfied especially considering the transaction cost. The cumulative return keeps going down. It is still a little step to quant. As my undergraduate major was finance. I’d love to apply the machine learning approaches for the Finance.

**Dataset and concept explain**

Data [[1]](#endnote-1)includes the open price, closing price, highest price and volume of General Electric Co from 2015-5-1 to 2016-5-1. And this paper concentrate much more on the closing price.

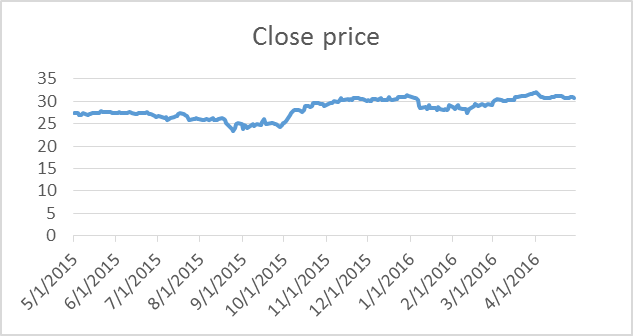


Figure GE's cloing price from 2015-5-1 to 2016-6-1

MACD, short for moving average convergence/divergence, is a trading indicator used in technical analysis of stock prices, created by Gerald Appel in the late 1970s.[1] It is supposed to reveal changes in the strength, direction, momentum, and duration of a trend in a stock's price. The MACD indicator (or "oscillator") is a collection of three time series calculated from historical price data, most often the closing price. These three series are: the MACD series proper, the "signal" or "average" series, and the "divergence" series which is the difference between the two. The MACD series is the difference between a "fast" (short period) exponential moving average (EMA), and a "slow" (longer period) EMA of the price series. The average series is an EMA of the MACD series itself.[[2]](#endnote-2)

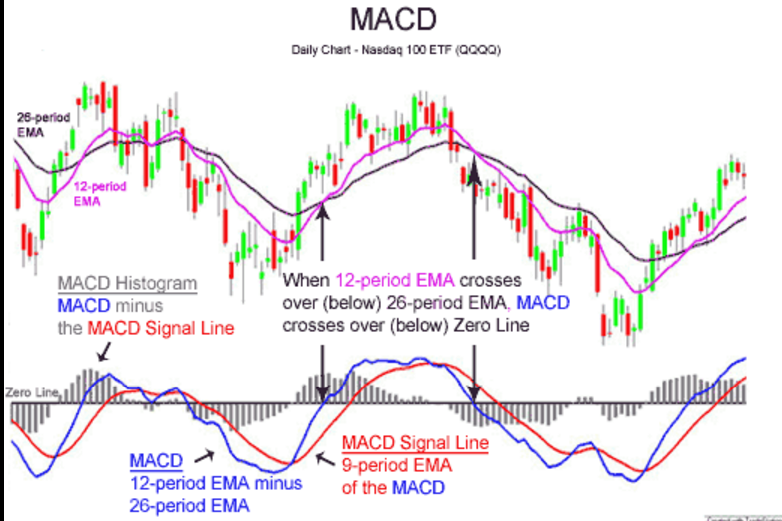


Figure MACD example

Max drawdown is an indicator of the risk of a portfolio chosen based on a certain strategy. It measures the largest single drop from peak to bottom in the value of a portfolio (before a new peak is achieved).[[3]](#endnote-3)

In machine learning, support vector machines (SVMs, also support vector networks[1]) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked for belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier.[[4]](#endnote-4)

**Machine learning approach**

MACD part:

Firstly calculate the short-term (5days) as a "fast" (short period) exponential moving average (EMA) and long-term (25 days) EMA for the each day as a "slow" (longer period) EMA of the price series. Then get the MACD at T-1 and T-2. Based on the data in table.xml, if the price at T bigger than at T-1, set the variable as 1 to represent the rise than the day before, similarly, if the price at T lower than at T-1, set the variable as 0 to represent the decrease than the day before.

SVM part:

* Let y denotes that 1 represents rise than yesterday, 0 represents down than yesterday.
* Add x1 denotes up or down at T, x2 denotes up or down at T-1, x3 denotes the difference of 25-MACD and 5-MACD at T-1 and x4 represent
* Build the svm model as y~x1+x2+x3+x4 in R under the package (e1071)

Prediction:

* Put the training data into the svm and predict.
* Calculate the cumulative return: (assuming the transaction cost is 0.01)
  + When the prediction at T equals 1, buy the GE stock at T and sell the GE stock at T+1
  + When the prediction at T equals 0, sell the GE stock at T and buy the GE stock at T+1. Calculate the returns each day.
* Cumulate the return, generally it keeps going down.

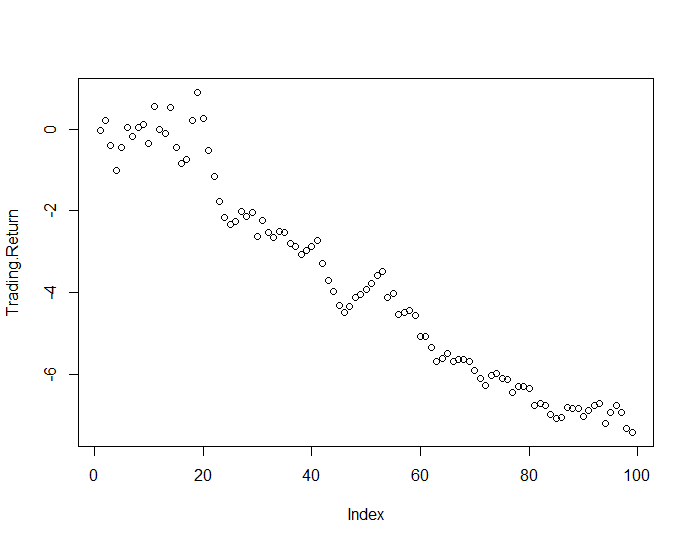


Figure 3 cumulative return

Maximum drawdown ratio is also not satisfying. Firstly, set the maximum return as 0, then we can get the percentage decrease than the maximum return each day.

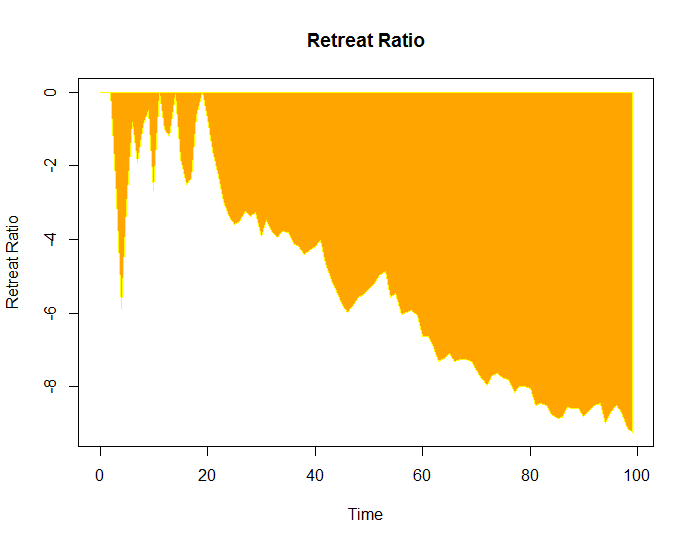


Figure Retreat Ratio

**Result**

The model in this paper uses the SVM to classify data processed by MACD which is an index used in stock pitch explained in next part. Then predict ups and downs given the test data to prove the model’s validation. At last apply the prediction to transaction to calculate the cumulative return and maximum drawback ratio. Although the result is not satisfied especially considering the transaction cost. However, testing the model by using the new data, we find that the cumulative keeps going down and the maximum retreat ratio is big.

Limitations: Only research one stock “GE”; can try more appropriate classify algorithms; MACD is a technical analysis, and we can use more technology in fundamental analysis to beat the market efficiency.

It is still a little step and beginning to quant. As my undergraduate major was finance. I’d love to apply the machine learning approaches for the Finance.

**Reference**

1. Data source, [www.yucezhe.com](http://www.yucezhe.com) [↑](#endnote-ref-1)
2. Wikipedia, <https://en.wikipedia.org/wiki/MACD> [↑](#endnote-ref-2)
3. Ycharts, <https://ycharts.com/glossary/terms/max_drawdown> [↑](#endnote-ref-3)
4. Wikipedia, <https://en.wikipedia.org/wiki/Support_vector_machine>

   **APPENDIX(code):**

   Data<- read.csv("table.csv") #GE's stock from 2015-5-1 to 2016-5-1

   head(Data,30)

   colnames(Data)<- c("time","open","high","low","end","volume")

   Close<- Data[,5]

   # MACD's function

   macd<-function(Stock,short\_period,

   long\_period,dea\_period) {

   # Stock is end price

   # short\_period is short-term

   # long\_periodd is long-term

   # dea\_period

   length\_stock<-length(Stock)

   EMA\_short<-rep(0,length\_stock)

   EMA\_long<-rep(0,length\_stock)

   stock\_diff<-rep(0,length\_stock)

   stock\_dea<-rep(0,length\_stock)

   stock\_macd<-rep(0,length\_stock)

   # initialize he primeter

   EMA\_short[1]<-Stock[1]

   EMA\_long[1]<-Stock[1]

   stock\_diff[1]<-0

   stock\_dea[1]<-0

   stock\_macd[1]<-0

   # calculate the MACD

   for (t in 2:length\_stock) {

   EMA\_short[t]<-Stock[t]\*2/(short\_period+1)+

   EMA\_short[t-1]\*(short\_period-1)/(short\_period+1)

   EMA\_long[t]<-Stock[t]\*2/(long\_period+1)+

   EMA\_long[t-1]\*(long\_period-1)/(long\_period+1)

   stock\_diff[t]<-EMA\_short[t]-EMA\_long[t]

   stock\_dea[t]<-stock\_diff[t]\*2/(dea\_period+1)+

   stock\_dea[t-1]\*(dea\_period-1)/(dea\_period+1)

   stock\_macd[t]<-2\*(stock\_diff[t]-stock\_dea[t])

   }

   return(stock\_macd)

   }

   # The explantory is the Macd for last one day and two day

   # trasfer the prince today as 1 or 0 to represent the rise or down of the stock

   short\_period<-5

   long\_period<-25

   dea\_period<-9

   Macd<-macd(Close,short\_period,long\_period,dea\_period)

   Diff<-diff(Close)

   Diff<-ifelse(Diff>0,1,0)

   # up or down at T (the response variable)

   Diff.T<-Diff

   Diff.T<-c(NA,Diff.T)

   # up or down at T-1(the explantory variable)

   Diff.T1<-Diff[-length(Diff)]

   Diff.T1<-c(NA,NA,Diff.T1)

   # up or down at T-2(the explantory variable)

   Diff.T2<-Diff[-c((length(Diff)-1),length(Diff))]

   Diff.T2<-c(NA,NA,NA,Diff.T2)

   # Macd at T-1(the explantory variable)

   Macd.T1<-Macd[-length(Macd)]

   Macd.T1<-c(NA,Macd.T1)

   # Macd at T-2(the explantory variable)

   Macd.T2<-Macd[-c((length(Macd)-1),length(Macd))]

   Macd.T2<-c(NA,NA,Macd.T2)

   # dataset

   N<-length(Close)

   Data.train<-data.frame(

   y=as.factor(Diff.T[c((N-250):(N-100))]),

   x1=Diff.T1[c((N-250):(N-100))],

   x2=Diff.T2[c((N-250):(N-100))],

   x3=Macd.T1[c((N-250):(N-100))],

   x4=Macd.T2[c((N-250):(N-100))])

   Data.test<-data.frame(

   x1=Diff.T1[c((N-99):(N))],

   x2=Diff.T2[c((N-99):(N))],

   x3=Macd.T1[c((N-99):(N))],

   x4=Macd.T2[c((N-99):(N))])

   # use SVM for modeling

   Data.train$y<- as.numeric(Data.train$y)

   library(e1071)

   DataSvmModel<- svm(y~x1+x2+x3+x4, data=Data.train)

   summary(DataSvmModel)

   plot(DataSvmModel,Data.train)

   # predict

   Predict<-predict(DataSvmModel,Data.test,

   probability=TRUE)

   # up or down for prediction

   Predict<-as.numeric(as.vector(Predict))

   # build the transaction and backtesting

   cost<-0.001 #assume the trasaction cost

   Length<-length(Predict)

   Trading.Return<-rep(0,Length)

   Close.test<-Close[c((N-99):N)]

   for (i in 1:Length) {

   if (Predict[i]==1) {

   # up

   Trading.Return[i]<-

   Close.test[i+1]\*(1-cost)-Close.test[i]\*(1+cost)

   } else {

   # down

   Trading.Return[i]<-

   Close.test[i]\*(1-cost)-Close.test[i+1]\*(1+cost)

   }

   }

   # cummulative sum of return

   Trading.Return<-cumsum(Trading.Return)

   Trading.Return<- Trading.Return[-100]

   # function to calculate the highest retreat ratio.

   Retreat\_Ratio<-function(stock\_return1) {

   N<-length(stock\_return1)

   RetraceRatio<-rep(0,N)

   for (i in 2:N) {

   C<-max(stock\_return1[1:i])

   if (C==stock\_return1[i]) {

   RetraceRatio[i]<-0

   } else {

   RetraceRatio[i]<-(stock\_return1[i]-C)/C

   }

   }

   return(RetraceRatio)

   }

   retreat\_ratio<-Retreat\_Ratio(Trading.Return)

   plot(Trading.Return)

   Length<- length(retreat\_ratio)

   #retreat ratio plot

   c1<-c(c(0:Length),c(Length:0))

   c2<-c(c(0,retreat\_ratio),rep(0,Length+1))

   plot(c1,c2,type='n',xlab='Time',

   ylab='Retreat Ratio')

   polygon(c1,c2,col='orange',border='yellow')

   title('Retreat Ratio')

   # #######################################################

   # ####################################################### [↑](#endnote-ref-4)