Statistical & Machine Learning: High Frequency Trading Challenge

Yuyuan Cui
ID: yuyuanc

Problem

Predict the direction of the change in USD/BRL exchange rate 120 seconds in the future, using proprietary "indicators". (Challenge put forth by http://www.circulumvite.com/home/trading-competition)

Data

Use the first day of the data from the challenge; add the first column which gives the rate 120 seconds in the future. File stored as "data_20130103.txt".

Columns of the data are as follows:

1. The price 120 seconds in the future (weighted average price).
2. Time in milliseconds since midnight UTC.
3. Weighted average (weighted by sizes) of the current best bid and ask prices of USD/BRL multiplied by 2000.
4. Current best bid price.
5. Size of current best bid price.
6. Current best ask price.
7. Size of current best ask price.

8~34. ("V8"~"V34") Other proprietary "Trend indicators" and "Book indicators".

Prediction Methods

This is a two-class classification problem. Four statistical methods are compared for this problem: Support Vector Machine, Logistic Regression, Classification Tree, and Random Forests. Of the 53,352 rows of the data, choose a random sample of 10,000 observations to be the same training data set for all four models. The remaining 43,352 rows serve as test data set.

(a) Support Vector Machine

Use all of the proprietary indicators (V8~V34) given for the prediction of the direction of change (classified as increase or decrease in exchange rate). Scale all the predictors. Use radial kernel:

$$K(x_1, x_2) = \exp\left(-\gamma \left|\left|x_1 - x_2\right|\right|^2\right)$$

γ is chosen as one over the dimension of predictor (as default).

The cost parameter C for the optimization is chosen to be from the set {5, 10, 25, 50, 100}

$$\min \frac{1}{2}\left|\left|\beta\right|\right|^2 + \left|\left|\beta\right|\right| C \sum_{i=1}^{n} \xi^*$$

Using ten-fold cross-validation, the best model is chosen with C=50.

(b) Logistic Regression

Using step-wise variable selection based on AIC criteria, arrive at a model with reduced number of predictors:

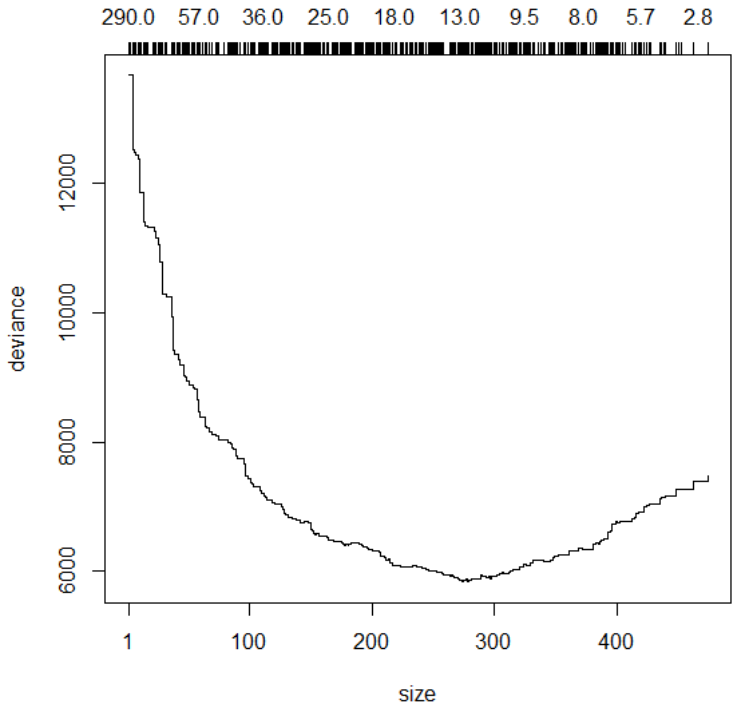
```
> summary(logregout)

Call:
glm(formula = factor(change) ~ V9 + V11 + V13 + V14 + V16 + V18 +
    V19 + V20 + V21 + V22 + V24 + V27 + V28 + V29 + V31 + V32 +
    V34, family = binomial, data = trainset)
```

(c) Classfication Tree

Use 10-fold cross validation to choose the optimal cost complexity parameter α, which is used in the optimization problem when pruning the (full) tree:

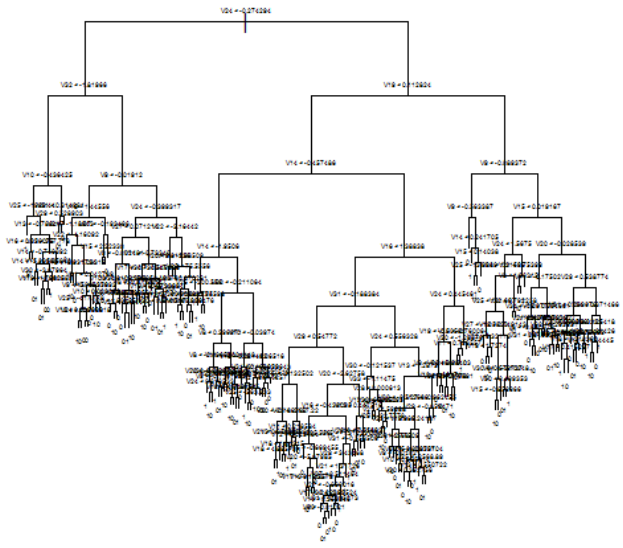
$$\text{min deviance} + \alpha * \text{number of terminal nodes in tree}$$

The alpha corresponding to the minimum deviance is 13.02294.



Set mindev=0 and minsize=2 to split the nodes fully for the full tree model.

The pruned tree (with optimal cost complexity parameter) is as follows:

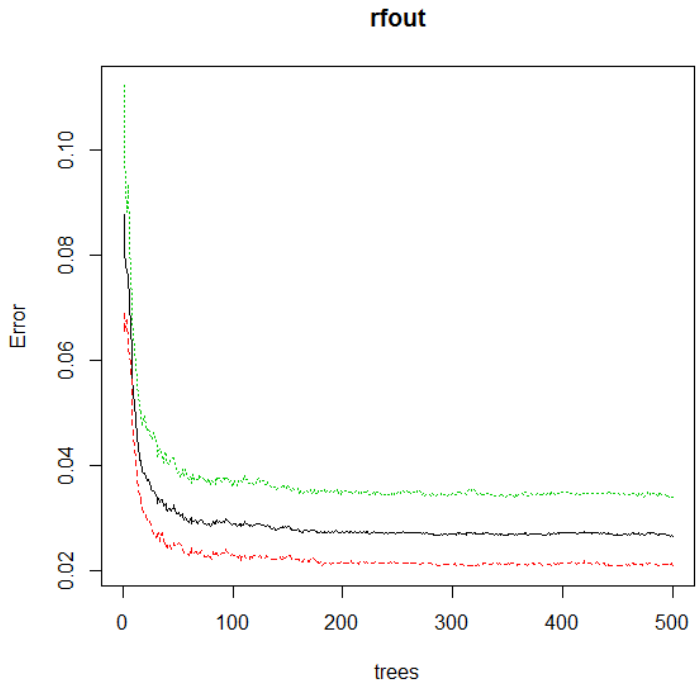


(d) Random Forests

Use B=500 (default) bootstrap samples (each of size 10,000) from the training data and grow a tree using each of the

bootstrap samples. After each split, choose $\lfloor\sqrt{p}\rfloor$ (default) of the p predictors at random and only consider splits from among those $\lfloor\sqrt{p}\rfloor$ predictors. Grow the tree until no further splits can be made without minimum node size dropping below 5 (default).

The out-of-bag (OOB) sample error plot shows that resetting B=100 seems to be enough, as there is no meaningful decrease after 100.



Test Set Prediction Comparison

Using random sample of size 10,000, compare the correctness of prediction in the test set:

SVM	0	1
0	23627	928
1	1066	17731

Logistic Regression	0	1
0	19294	5261
1	11040	7757

Classification Tree	0	1
0	22745	1810
1	1859	16938

Random Forests	0	1
0	24055	500
1	628	18169

Correctness rate:

SVM	95.4%
Logistic Regression	62.4%
Classification Tree	91.5%
Random Forests	97.4%

If instead of using a random sample, use the first 20,000 observations as training and the remaining 33352 observations of the day as test set, the test sample performance is:

SVM	0	1
0	6150	13443
1	3906	9853

Logistic Regression	0	1
0	5297	14296
1	1967	11792

Classification Tree	0	1
0	8588	11005
1	6055	7704

Random Forests	0	1
0	10026	9567
1	6835	6924

Correctness rate:

SVM	48.0%
Logistic Regression	51.2%
Classification Tree	48.8%
Random Forests	50.8%

Conclusion

With random sampling training set, support vector machine, classification tree and random forests predict the classification better than logistic regression. In comparison, with consecutive observations as training set, only logistic regression and random forests have a correctness rate above 50%. A random training sample works much better than training set from consecutive observations. However, it is hard to conduct random sampling over a trading day and train the model for actual trade. Consecutive sampling, on the other hand, is too simple and may fail to capture the time pattern over a trading day. Therefore, to turn the model into real tradable prediction, more sophisticated sampling methods should be applied, which need to strike a balance between feasibility for real trade and representativeness of data.

R Code

```
# Load the library and the data
# install.packages("e1071")
library(e1071)
# install.packages("randomForest")
library(randomForest)
# install.packages("tree")
library(tree)

# function for tree model cross-validation
cv.tree.full <- function (object, rand, FUN = prune.tree, K = 10, mindev=0, mincut=1, minsize=2, fixseed=0,...)
{
    if(!is.null(fixseed))
    {
        set.seed(fixseed)
    }
    require(tree)
    if (!inherits(object, "tree"))
        stop("Not legitimate tree")
    m <- model.frame(object)
    extras <- match.call(expand.dots = FALSE)$...
    FUN <- deparse(substitute(FUN))
    init <- do.call(FUN, c(list(object), extras))
    if (missing(rand))
        rand <- sample(K, length(m[[1]]), replace = TRUE)
    cvdev <- 0
    for (i in unique(rand)) {
        tlearn <- tree(model = m[rand != i, , drop = FALSE],mindev=mindev,mincut=mincut,minsize=minsize)
        plearn <- do.call(FUN, c(list(tlearn, newdata = m[rand ==
            i, , drop = FALSE], k = init$k), extras))
        cvdev <- cvdev + plearn$dev
    }
    init$dev <- cvdev
    init
}

# Read in the data
oneday = read.table("data_20130103.txt")

# Create the indicator for the direction of change.
oneday$change = as.numeric((oneday$V1 - oneday$V3) > 0)
nrow(oneday)
ncol(oneday)


# Divide the data into training and test sets
#set.seed(0)
# random sample
trainrows = sample(1:nrow(oneday),  replace=F, size=10000)
```

```
trainset = oneday[trainrows,8:35]
testset = oneday[-trainrows,8:35]
# first observations  as sample
# trainset = oneday[1:20000,8:35]  # 20000 rows
# testset = oneday[20001:nrow(oneday),8:35]  # 33352 rows

# Support  Vector Machine
svmout = tune.svm(factor(change) ~ ., data=trainset, cost=c(5, 10, 25, 50, 100))
summary(svmout)
# plot(svmout$best.model,  trainset, V10~V9, cex.axis=1.3,cex.lab=1.3)

# Logistic Regression
logregout = glm(factor(change) ~., data=trainset, family=binomial)
logregout = step(logregout)
summary(logregout)

# Regression Tree
fulltree = tree(factor(change) ~ ., data=trainset, mindev=0, minsize=2)
cvout = cv.tree.full(fulltree)
plot(cvout)
alphaopt = cvout$k[which.min(cvout$dev)]
alphaopt
classtreeout = prune.tree(fulltree, k=20)
plot(classtreeout)
text(classtreeout,cex=0.3)

# Random Forest
rfout = randomForest(factor(change) ~ ., data=trainset)
plot(rfout)
rfout = randomForest(factor(change) ~ ., data=trainset, ntree=100)

# Tables showing the results
svmtab = table(testset$change, predict(svmout$best.model,testset))
logregtab = table(testset$change, as.numeric(predict(logregout,newdata=testset,type="response") > 0.5))
classtreetab = table(testset$change, predict(classtreeout,newdata=testset,type="class"))
rftab = table(testset$change, predict(rfout,  newdata=testset))
svmtab
logregtab
classtreetab
rftab
```