PREDICTION COMPETITION 5

Q1

Training data : Error estimation using 10 fold cross validation : 0.1779548

Test data : Error estimation of ‘svm’ using 10-fold cross validation: 0.2038239

```r
library(readr)
dat <- read.csv("Desktop/aug_train (1).csv", header = TRUE)
dat2<- dat
library(e1071)
library(dplyr)

### data cleaning :
dat$city = as.factor(dat$city)
dat$city_development_index = as.factor(dat$city_development_index)
dat$gender = as.factor(dat$gender)
dat$relevent_experience = as.factor(dat$relevent_experience)
dat$enrolled_university = as.factor(dat$enrolled_university)
dat$education_level = as.factor(dat$education_level)
dat$major_discipline = as.factor(dat$major_discipline)
dat$experience= as.factor(dat$experience)
dat$company_size = as.factor(dat$company_size)
dat$company_type = as.factor(dat$company_type)
dat$last_new_job = as.factor(dat$last_new_job)
dat$city<- factor(dat$city,levels = c("city_103","city_21","city_16","city_114","city_160"))
dat$enrollee_id = as.factor(dat$enrollee_id)
### creating a data frame for easier use :::
frame<- tbl_df(dat)
### dealing with missing values to reduce the imbalance in the data::
frame$gender[frame$gender == ""] <- NA
frame$enrolled_university[frame$enrolled_university == ""] <- NA
frame$education_level[frame$education_level == ""] <- NA
frame$major_discipline[frame$major_discipline == ""] <- NA
frame$company_size[frame$company_size == ""] <- NA
frame$company_type[frame$company_type == ""] <- NA
frame$last_new_job[frame$last_new_job == ""] <- NA


```

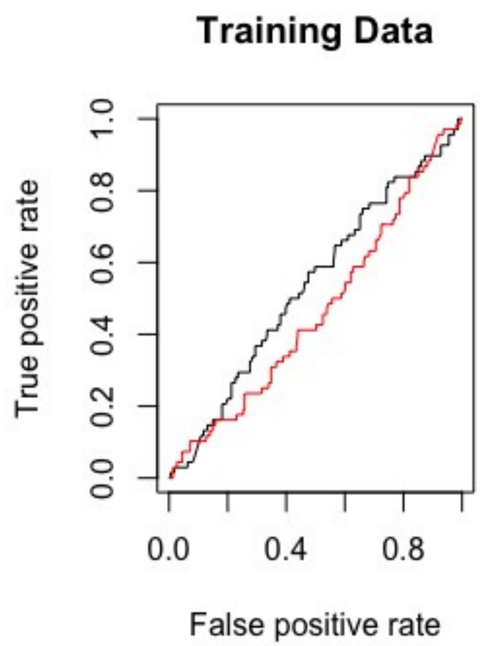
```
train <- sample(1:nrow(frame),1000)
training <- frame[train,]
testing <- frame[-train,]
#### creating a SVM function using target to see whether one is a potential
#### candidate or not.

svmf = svm(target~., data = frame[train,], kernel="radial",cost=1, gamma= 5.177323e-05)
#### tuning out :::
tune.out <- tune(svm , target~., data = frame[train,] , kernel = "linear",cost = 0.001)

xtest <- frame[-train,]
ytest<- frame$target
bestmod<- tune.out$best.model
ypred <- predict(bestmod,xtest)
ytest<- ytest[c(1:length(ypred))]
testdat<- data.frame(x = xtest,y= as.factor(ytest))
pred = predict(bestmod,newdata = frame[-train,])
prediction_model = predict(bestmod,newdata = frame[train,])
```

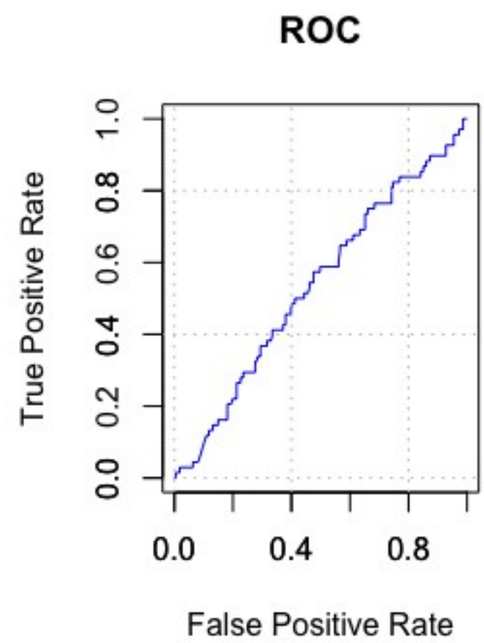
```
> summary(tune.out$best.model)

Call:
best.tune(method = svm, train.x = target ~ ., data = frame[train, ], kernel = "linear",
    cost = 0.001)


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  linear
       cost:  0.001
      gamma:  5.177323e-05
    epsilon:  0.1


Number of Support Vectors:  147
```

```
> summary(tune.out)

Error estimation of 'svm' using 10-fold cross validation: 0.1779548
```

Q2
A)

```r

##### QUESTION 2 :::
rocplot <- function(pred , truth , ...) {
  predob <- prediction(pred , truth)
  perf <- performance(predob , "tpr", "fpr")
  plot(perf , ...)
}

library(ROCR)
svmfit.opt <- svm(target~.,data=frame[train,],kernel="radial",cost = 1, gamma=5, decision.values= T)
fitted<- attributes(predict(svmfit.opt,frame[train,],decision.values = TRUE))$decision.values
par(mfrow=c(1,2))
var <- frame[train,"target"]
len_f <- length(fitted)
rocplot(fitted,var[1:len_f,1], main = "Training Data")
rocplot(-fitted , var[1:len_f,1], add = T, col = "red")


### ROC :::
rocplot(fitted, var[1:len_f,1], type = "l", col = "blue", xlab = "False Positive Rate"
        , ylab = "True Positive Rate", main = "ROC")
axis(1, seq(0.0,1.0,0.4))
axis(2, seq(0.0,1.0,0.4))
abline(h=seq(0.0,1.0,0.4), v=seq(0.0,1.0,0.4), col="gray", lty=3)

```





B)

Calculating the AUC using glm and the pROC package to get the auc.

```
### Calculating the AUC :::
glm.fit = glm(target~., data = training, family="binomial")
predicted <- predict(glm.fit,data = frame[-train,],type= "response")
library(pROC)
testing <- frame[-train,]
AUC <- auc(testing$target[1:len_f],predicted)

```

```
> AUC
Area under the curve: 0.5655
>
```

D)

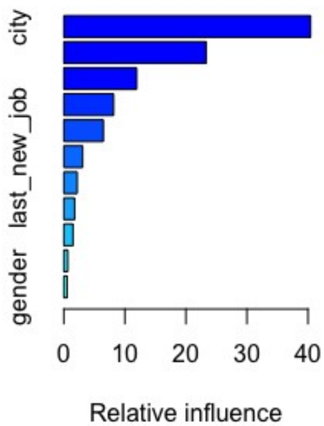
Calculating the variable importance using the gbm method and boosting ::

```
#### part d)
boosting <- gbm(target~.-city_development_index-enrollee_id, data= testing,
                distribution = "gaussian", n.trees= 1000, interaction.depth = 4, shrinkage = 0.01)
summary(boosting)


```

```
> summary(boosting)
                                     var    rel.inf
city                                city 40.4247585
company_size                company_size 23.3045664
experience                    experience 11.9304117
education_level          education_level  8.1528005
company_type                company_type  6.4705238
last_new_job                last_new_job  3.0812341
major_discipline        major_discipline  2.1827220
enrolled_university enrolled_university  1.7676838
training_hours            training_hours  1.4975057
relevent_experience relevent_experience  0.6497003
gender                            gender  0.5380932
```

The most important variables are as follows from the training data^
However, the testing data gives a better importance :

```
> summary(boosting)
                                     var    rel.inf
experience                    experience 35.0245521
city                                city 19.0175342
company_size                company_size 16.6230017
last_new_job                last_new_job  8.4786199
company_type                company_type  5.7174332
education_level          education_level  5.5987691
training_hours            training_hours  3.9712284
enrolled_university enrolled_university  2.7384052
major_discipline        major_discipline  1.9324182
gender                            gender  0.6377761
relevent_experience relevent_experience  0.2602621
```



(training data variable importance)

Confusion matrix using the best model prediction and the target which determines whether a potential candidate or not.

```
> table(predict= ypred, candidate= ytest)
                    candidate
predict              0 1
  0.0396450554191474 0 1
  0.0400642377264664 0 1
  0.0401501786651821 1 0
  0.0401992322490509 0 1
  0.0402438334891641 0 1
  0.0404269597208246 1 0
```