Dataset: Voice

library(caTools)

library(glmnet)

library(ROCR)

library(caret)

library(e1071)

library(randomForest)

library (tree)

**Exploratory Data Analysis**

setwd("C:\\Users\\admin\\Downloads")

voiceData<-read.csv("voice.csv", header = TRUE, stringsAsFactors = FALSE)

head(voiceData)

dim(voiceData)

**We have read the dataset**

**#### Check NA values ####**

detectNAs<-function(x){

return(sum(is.na(x)))

}

lapply(voiceData, detectNAs)

**#### Check 0 values ####**

detectZeros<-function(x){

return(sum(x==0))

}

lapply(voiceData, detectZeros)

**###Replace Zero values with mean**

voiceData$dfrange[voiceData$dfrange==0]<-mean(voiceData$dfrange)

voiceData$modindx[voiceData$modindx==0]<-mean(voiceData$modindx)

**###Convert dependent variable into binary terms**

voiceData$label<-ifelse(voiceData$label=="male", 1, 0)

**We have fitted a lasso model in order to identify the significant variables**

**####Fit Lasso Model**

**##### Split Data for Lasso**

x=x=model.matrix(label~.,voiceData)[,-1]

dim(x)

head(x)

y=voiceData$label

grid=10^seq(10,-2,length=100)

train\_lasso=sample(1:nrow(x), nrow(x)/2)

test\_lasso=(-train\_lasso)

lasso.mod=glmnet(x[train\_lasso,],y[train\_lasso],alpha=1,lambda=grid)

plot(lasso.mod)

set.seed(1)

cv.out=cv.glmnet(x[train\_lasso,],y[train\_lasso],alpha=1)

plot(cv.out)

bestlam=cv.out$lambda.min

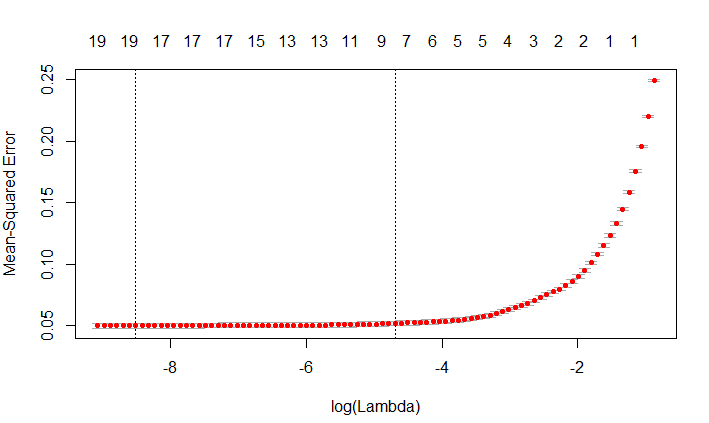
bestlam

out=glmnet(x,y,alpha=1,lambda=grid)

lasso.coef=predict(out,type="coefficients",s=bestlam)[1:20,]

lasso.coef

lasso.coef[lasso.coef!=0]



We choose the best lamda here

**The significant variables came out to be**

Q75 IQR skew sfm mode meanfun minfun maxfun meandom

###Split Data

split<-sample.split(voiceData$label, SplitRatio=0.8)

voice\_train<-subset(voiceData, split==TRUE)

voice\_test<-subset(voiceData, split==FALSE)

dim(voice\_train)

dim(voice\_test)

######Build Model

####Logistics

voice.fit1<-glm(label~(Q75+IQR+skew+sfm+mode+meanfun+minfun+maxfun+meandom), data=voice\_train, family = "binomial")

summary(voice.fit1)

predicted\_voice<-predict(voice.fit1, newdata = voice\_test, type="response")

conf\_voice1<-table(predicted=predicted\_voice>0.5, actual=voice\_test$label)



accuracy\_voice1=sum(diag(conf\_voice1))/sum(conf\_voice1)

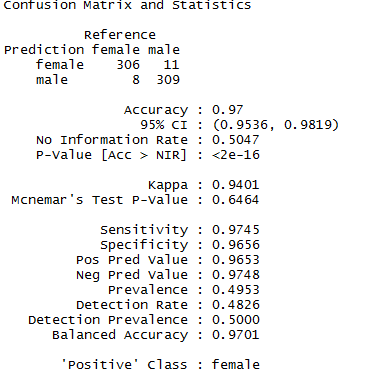
accuracy\_voice1

0.9700315

test\_set\_label<-as.factor(ifelse(voice\_test$label==1, "male", "female"))

log.pred<-ifelse(predicted\_voice>0.5, "male", "female")

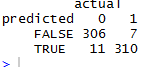
confusionMatrix(test\_set\_label, as.factor(log.pred))



**We have varied the cutoff point in oder to check if accuracy improves.**

**###0.45**

conf\_voice1<-table(predicted=predicted\_voice>0.45, actual=voice\_test$label)



accuracy\_voice1=sum(diag(conf\_voice1))/sum(conf\_voice1)

accuracy\_voice1

0.9716088

**###0.35**

conf\_voice1<-table(predicted=predicted\_voice>0.35, actual=voice\_test$label)



accuracy\_voice1=sum(diag(conf\_voice1))/sum(conf\_voice1)

accuracy\_voice1

0.966877

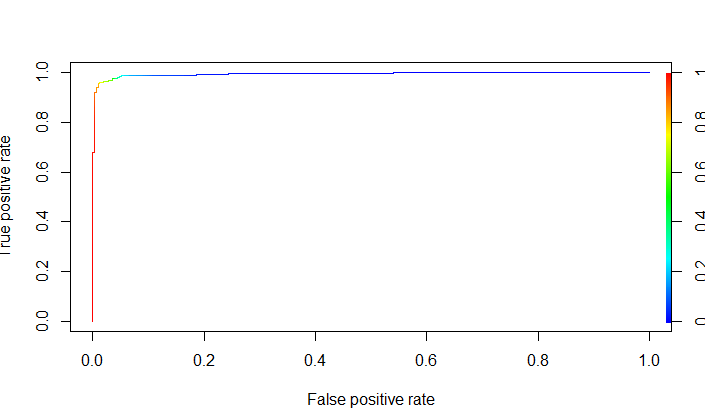
**We checked that the accuracy remains nearly same**

**####ROCR**

ROCRpred\_log <- prediction(predicted\_voice, voice\_test$label)

ROCRperf\_log <- performance(ROCRpred\_log, 'tpr','fpr')

plot(ROCRperf\_log, colorize = TRUE, text.adj = c(-0.2,1.7))



auc.tmp\_log <- performance(ROCRpred\_log,"auc")

auc\_log <- as.numeric(auc.tmp\_log@y.values)

auc\_log

Area under curve came out to be 0.9946263

giniCoeff\_log<-2\*auc\_log-1

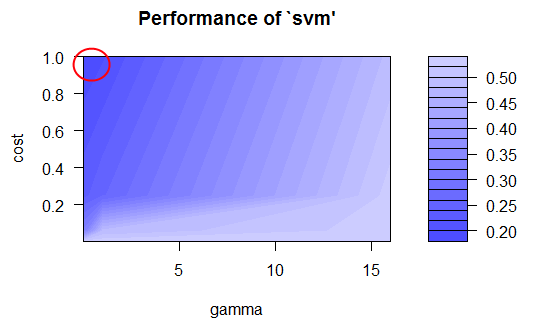
giniCoeff\_log

0.9892526

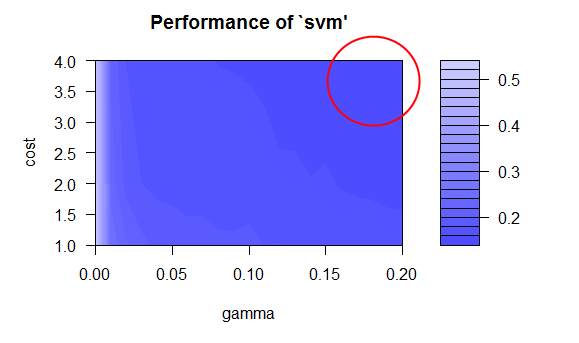
Higher the gini Coeff and auc better is the model

**SVM**

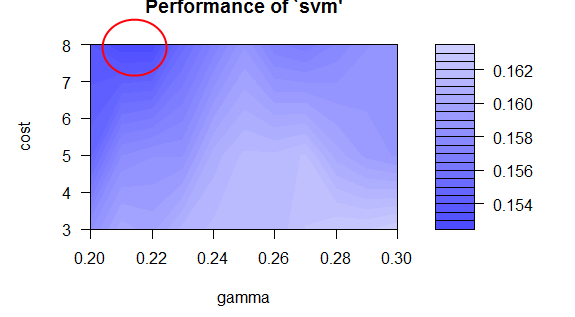
Our next model is a support vector machine, tuned with the best values for cost and gamma. To determine the best fit for an SVM model, the model was initially run with default parameters. A plot of the SVM error rate is then printed, with the darkest shades of blue indicating the best (ie., lowest) error rates. This is the best place to choose a cost and gamma value. You can fine-tune the SVM by narrowing in on the darkest blue range and performing further tuning. This essentially focuses in on the section, yielding a finer value for cost and gamma, and thus, a lower error rate and higher accuracy. The following performance images show how this progresses.



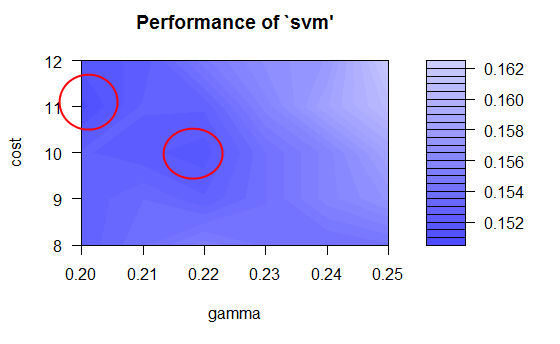
**First pass of tuning the SVM. Our best values are around cost 1 and gamma 0.2**



**Further fine-tuning and our best values are around cost 4 and gamma 0.2**



**Zooming in further, our best values are around cost 8 and 0.21 gamma**



**One more final pass, our best values are around cost 10 and 0.22 gamma**

cat("\014")

voiceData<-read.csv("voice.csv", header = TRUE, stringsAsFactors = FALSE)

voiceData$label<-ifelse(voiceData$label=="male", 1, 0)

**#### Check NA values ####**

detectNAs<-function(x){

return(sum(is.na(x)))

}

lapply(voiceData, detectNAs)

**#### Check 0 values ####**

detectZeros<-function(x){

return(sum(x==0))

}

lapply(voiceData, detectZeros)

**###Replace Zero values with mean**

voiceData$dfrange[voiceData$dfrange==0]<-mean(voiceData$dfrange)

voiceData$modindx[voiceData$modindx==0]<-mean(voiceData$modindx)

split<-sample.split(voiceData$label, SplitRatio=0.8)

voice\_train<-subset(voiceData, split==TRUE)

voice\_test<-subset(voiceData, split==FALSE)

set.seed(777)

svmTune <- tune.svm(label ~ ., data=voice\_train, sampling='fix', gamma = 2^c(-8,-4,0,4), cost = 2^c(-8,-4,-2,0))

**# The darker blue is the best values for a model.**

plot(svmTune)

**# We can re-run the tuning with more specific values for gamma (epsilon) and cost**.

set.seed(777)

svmTune <- tune.svm(label ~ ., data=voice\_train, sampling='fix', gamma = seq(0, 0.2, 0.01), cost = c(1, 2, 4))

genderSvm <- svmTune$best.model

plot(svmTune)

**# Accuracy: 0.91**

predictSvm <- predict(genderSvm, voice\_train)

table(predictSvm, train$label)

**# Accuracy: 0.83**

predictSvm <- predict(genderSvm, voice\_test)

table(predictSvm, test$label)

**# Narrow down one more time**.

set.seed(777)

svmTune <- tune.svm(label ~ ., data=voice\_train, sampling='fix', gamma = seq(0.2, 0.3, 0.01), cost = c(3, 5, 8))

genderSvm <- svmTune$best.model

plot(svmTune)

**# Accuracy: 0.96**

predictSvm <- predict(genderSvm, voice\_train)

table(predictSvm, train$label)

**# Accuracy: 0.85**

predictSvm <- predict(genderSvm, voice\_test)

table(predictSvm, test$label)

**# One final tuning.**

set.seed(777)

svmTune <- tune.svm(label ~ ., data=voice\_train, sampling='fix', gamma = seq(0.2, 0.25, 0.01), cost = seq(8, 12, 1))

genderSvm <- svmTune$best.model

plot(svmTune)

**# Accuracy: 0.97**

predictSvm <- predict(genderSvm, voice\_train)

table(predictSvm, train$label)

**# Accuracy: 0.85 (one less, so very tiny overfitting)**

predictSvm <- predict(genderSvm, voice\_test)

table(predictSvm, test$label)

**By varying different parameters the highest accuracy we obtained is 97%**

**########### ROCR SVM #############**

ROCRpred.svm<-prediction(predictSvm, voice\_test$label)

ROCRpref.svm<-performance(ROCRpred.svm, "tpr", "fpr")

plot(ROCRpref.svm, colorize=TRUE)

auc.temp\_svm<-performance(ROCRpred.svm, "auc")

auc\_svm<-as.numeric(auc.temp\_svm@y.values)

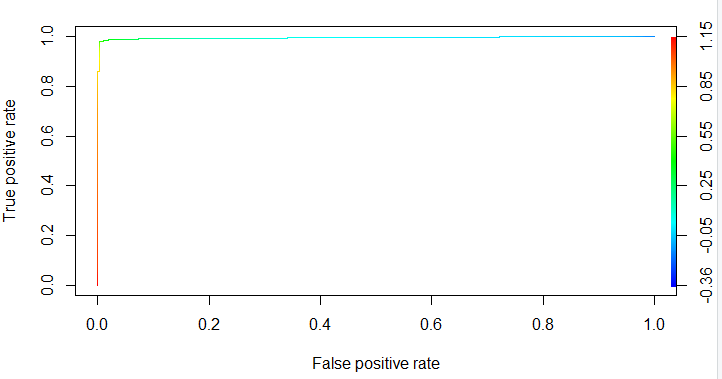
auc\_svm

area under curve is 0.995711

giniCoeff\_svm<-2\*auc\_svm-1

giniCoeff\_svm

0.9914219



**Therfore we can conclude both logistics and sv model give nearly same prediction accuracy**