### Import data

library(caret)  
library(tidyverse)  
library(plyr)  
library(naivebayes)  
library(rpartScore)  
train\_df=read\_csv("C:\\Users\\Arnold\\OneDrive\\R\_Python\_working\_directory\\IST 707 Data Analytics\\Kaggle-digit-train.csv")  
test\_df=read\_csv("C:\\Users\\Arnold\\OneDrive\\R\_Python\_working\_directory\\IST 707 Data Analytics\\Kaggle-digit-test.csv")

### Convert labels to factors

train\_df$label=factor(train\_df$label)

### Split training data for fitting model and validation.

fit\_idx=createDataPartition(train\_df$label,p = .5,list = F)  
fit\_df=train\_df[fit\_idx,]  
val\_df=train\_df[-fit\_idx,]

### Check the summary to identify some other problems

summary(fit\_df[,c(1,2,sample(3:785,8))])

## label pixel0 pixel183 pixel486   
## 1 :2342 Min. :0 Min. : 0.0 Min. : 0.0   
## 7 :2201 1st Qu.:0 1st Qu.: 0.0 1st Qu.: 0.0   
## 3 :2176 Median :0 Median :128.0 Median : 0.0   
## 9 :2094 Mean :0 Mean :124.7 Mean : 74.7   
## 2 :2089 3rd Qu.:0 3rd Qu.:252.0 3rd Qu.:179.0   
## 6 :2069 Max. :0 Max. :255.0 Max. :255.0   
## (Other):8032   
## pixel697 pixel152 pixel207 pixel36   
## Min. : 0.00000 Min. : 0.00 Min. : 0.00 Min. : 0.00000   
## 1st Qu.: 0.00000 1st Qu.: 0.00 1st Qu.: 0.00 1st Qu.: 0.00000   
## Median : 0.00000 Median : 0.00 Median : 27.00 Median : 0.00000   
## Mean : 0.02047 Mean : 60.21 Mean : 95.14 Mean : 0.04347   
## 3rd Qu.: 0.00000 3rd Qu.:114.00 3rd Qu.:225.00 3rd Qu.: 0.00000   
## Max. :212.00000 Max. :255.00 Max. :255.00 Max. :253.00000   
##   
## pixel782 pixel768   
## Min. :0 Min. : 0.0000   
## 1st Qu.:0 1st Qu.: 0.0000   
## Median :0 Median : 0.0000   
## Mean :0 Mean : 0.4213   
## 3rd Qu.:0 3rd Qu.: 0.0000   
## Max. :0 Max. :255.0000   
##

A lot of pixels are mostly 0, & some are even all 0.

### Remove pixels with all 0’s, because they provide no values.

fit\_df=fit\_df[,c(T,colSums(fit\_df[,-1])>0)]

## Build a decision tree model. Tune the parameters, such as the pruning options, and report the 3-fold CV accuracy.

### The training control method

ctr=trainControl(method = 'cv',number = 3,allowParallel = T)

### Fitting D.T model start.time=Sys.time() Grid=expand.grid(cp =seq(0, 0.005,length.out = 3),split=c('abs','quad'),prune='mc') dt=train(label~.,fit\_df,method='rpartScore',trControl=ctr,tuneGrid = Grid) run.time=Sys.time()-start.time val\_df$dt\_predict=predict(dt,val\_df) dt\_score=postResample(pred = val\_df$dt\_predict,obs = val\_df$label)[1] fit\_df.dt\_predict=predict(dt,fit\_df) dt.train\_score=postResample(pred = fit\_df.dt\_predict,obs = fit\_df$label)[1]

### Create dataframe for models comparison

(mod\_com=data.frame(Model='Decision Tree',`Train Accuracy`=dt.train\_score,`Test Accuracy`=dt\_score,Note=NA,`Run Time`=run.time,row.names = NULL))

## Model Train.Accuracy Test.Accuracy Note Run.Time  
## 1 Decision Tree 0.884207 0.7978283 NA 2.879597 hours

## Build a naïve Bayes model. Tune the parameters, such as the discretization options, to compare results.

### The pixels are numeric, which means by default the probabilities will be calculated using normal distribution. Based on the initial data observation, the pixels are not likely to be normally distributed. A custom function will be created to discretize the pixels.

### N.B without discretization

start.time=Sys.time()  
Grid=expand.grid(laplace = 1:2,usekernel=c(T,F),adjust= 1:2)  
nb=train(label ~ ., data = fit\_df, method = "naive\_bayes",trControl =  
ctr,tuneGrid =Grid)  
run.time=Sys.time()-start.time  
val\_df$nb\_predict=predict(nb,val\_df)  
nb\_score=postResample(pred = val\_df$nb\_predict,obs = val\_df$label)[1]  
fit\_df.nb\_predict=predict(nb,fit\_df)  
nb.train\_score=postResample(pred = fit\_df.nb\_predict,obs = fit\_df$label)[1]

### Add a row to models comparison dataframe

(mod\_com=data.frame(Model='Naive Bayes',`Train Accuracy`=nb.train\_score,`Test  
Accuracy`=nb\_score,Note='Pixels not discretized',`Run Time`=run.time,row.names =  
 NULL) %>% rbind(mod\_com))

## Model Train.Accuracy Test.Accuracy Note  
## 1 Naive Bayes 0.5324001 0.5299328 Pixels not discretized  
## 2 Decision Tree 0.8842070 0.7978283 <NA>  
## Run.Time  
## 1 4.479009 mins  
## 2 2.879597 hours

discretize=function(x){  
 if (length(unique(x[x>0])) < 12){return(factor(ifelse(x==0,'0.','>0')))}  
 else if (length(unique(x[x>0])) < 31){  
 cuts=c(-.1,quantile(x[x>0],seq(0,1,length.out = 4)))  
 cuts[2]=0  
 for (c in 2:length(cuts)){  
 if (cuts[c] %in% cuts[1:c-1]){cuts[c]=cuts[c]+1}  
 }  
 return(factor(cut(x,breaks = cuts,labels = c('0.','Low','Medium','High'))))  
 }  
 else{  
 cuts=c(-.1,quantile(x[x>0],seq(0,1,length.out = 7)))  
 cuts[2]=0  
 for (c in 2:length(cuts)){  
 if (cuts[c] %in% cuts[1:c-1]){cuts[c]=cuts[c]+1}  
 }  
 return(factor(cut(x,breaks = cuts,labels =  
 c('0.','1.','2.','3.','4.','5.','6.'))))  
 }  
}

### N.B with pixels discretized

tdf=fit\_df  
fit.idx=1:nrow(tdf)  
tdf=rbind(tdf,val\_df[,colnames(tdf)])  
tdf[,-1]=tdf %>% select(-label) %>% apply(MARGIN = 2,FUN=discretize) %>%  
as.data.frame()  
tdf.fit=tdf[fit.idx,]  
tdf.val=tdf[-fit.idx,]  
start.time=Sys.time()  
nb.dis=train(label ~ ., data = tdf.fit, method = "naive\_bayes",trControl =  
ctr,tuneGrid =Grid)  
run.time=Sys.time()-start.time  
val\_df$nb.dis\_predict=predict(nb.dis,tdf.val)  
nb.dis\_score=postResample(pred = val\_df$nb.dis\_predict,obs = val\_df$label)[1]  
tdf.fit$predict=predict(nb.dis,tdf.fit)  
nb.dis.train\_score=postResample(pred = tdf.fit$predict,obs = tdf.fit$label)[1]

### Add a row to models comparison dataframe

(mod\_com=data.frame(Model='Naive Bayes',`Train Accuracy`=nb.dis.train\_score,`Test  
Accuracy`=nb.dis\_score,Note='Pixels discretized',`Run Time`=run.time,row.names =  
 NULL) %>% rbind(mod\_com))

## Model Train.Accuracy Test.Accuracy Note  
## 1 Naive Bayes 0.7026615 0.6936229 Pixels discretized  
## 2 Naive Bayes 0.5324001 0.5299328 Pixels not discretized  
## 3 Decision Tree 0.8842070 0.7978283 <NA>  
## Run.Time  
## 1 24.840707 mins  
## 2 4.479009 mins  
## 3 2.879597 hours