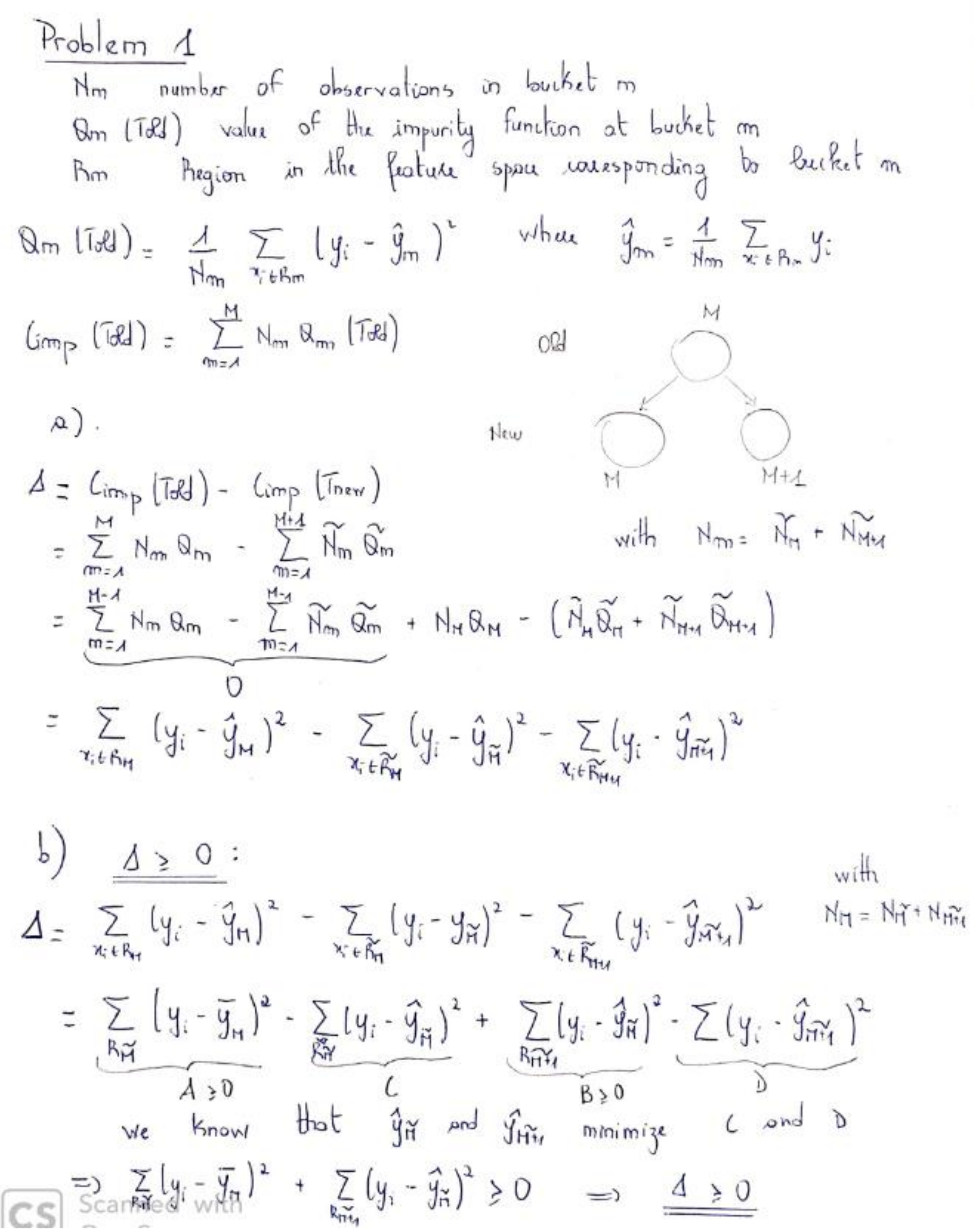
Homework 3



Problem 1

N_m number of observations in bucket m

$Q_m(T_{old})$ value of the impurity function at bucket m

R_m Region in the feature space corresponding to bucket m

$$Q_m(T_{old}) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{y}_m)^2 \quad \text{where} \quad \hat{y}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i$$

$$C_{imp}(T_{old}) = \sum_{m=1}^{M} N_m Q_m(T_{old})$$

Old
M
New
M
M+1

a).

$$\Delta = C_{imp}(T_{old}) - C_{imp}(T_{new})$$

$$= \sum_{m=1}^{M} N_m Q_m - \sum_{m=1}^{M+1} \tilde{N}_m \tilde{Q}_m \qquad \text{with} \quad N_m = \tilde{N}_M + \tilde{N}_{M+1}$$

$$= \underbrace{\sum_{m=1}^{M-1} N_m Q_m - \sum_{m=1}^{M-1} \tilde{N}_m \tilde{Q}_m}_{0} + N_M Q_M - (\tilde{N}_M \tilde{Q}_M + \tilde{N}_{M+1} \tilde{Q}_{M+1})$$

$$= \sum_{x_i \in R_M} (y_i - \hat{y}_M)^2 - \sum_{x_i \in \tilde{R}_M} (y_i - \hat{y}_{\tilde{M}})^2 - \sum_{x_i \in \tilde{R}_{M+1}} (y_i - \hat{y}_{\tilde{M+1}})^2$$

b) $\underline{\underline{\Delta \geq 0}}$:

$$\Delta = \sum_{x_i \in R_M} (y_i - \hat{y}_M)^2 - \sum_{x_i \in \tilde{R}_M} (y_i - y_{\tilde{M}})^2 - \sum_{x_i \in \tilde{R}_{M+1}} (y_i - \hat{y}_{\tilde{M+1}})^2 \qquad \text{with} \quad N_M = N_{\tilde{M}} + N_{\tilde{M+1}}$$

$$= \underbrace{\sum_{R_{\tilde{M}}} (y_i - \bar{y}_M)^2}_{A \geq 0} - \underbrace{\sum_{R_{\tilde{M}}} (y_i - \hat{y}_{\tilde{M}})^2}_{C} + \underbrace{\sum_{R_{\tilde{M+1}}} (y_i - \hat{y}_{\tilde{M}})^2}_{B \geq 0} - \underbrace{\sum (y_i - \hat{y}_{\tilde{M+1}})^2}_{D}$$

we know that $\hat{y}_{\tilde{M}}$ and $\hat{y}_{\tilde{M+1}}$ minimize C and D

$$\Rightarrow \sum_{R_{\tilde{M}}} (y_i - \bar{y}_M)^2 + \sum_{R_{\tilde{M+1}}} (y_i - \hat{y}_{\tilde{M}})^2 \geq 0 \quad \Rightarrow \quad \underline{\underline{\Delta \geq 0}}$$

CS Scanned with

c) Show that $C_\alpha(T_{new}) \leq C_\alpha(T_{old}) \Rightarrow R^2_{new} - R^2_{old} \geq \alpha$

$C_\alpha' \leq C_\alpha$

$C_{\alpha'} \leq C_\alpha \Rightarrow C' + \alpha\, SST^{+}|T+1| < C + \alpha . SST|T|$

$\Rightarrow C' - C < -\alpha . SST$

$\Rightarrow \dfrac{C' - C}{SST} < -\alpha$ with $R^2_{new} = \dfrac{1 - \sum_{m=1}^{M+1} \tilde{N}_m \tilde{Q}_m}{SST}$

$\Rightarrow \dfrac{SSR' - SSR}{SST} < -\alpha$ $R^2_{old} = 1 - \dfrac{\sum_{m=1}^{M} N_m Q_m}{SST}$

$\Rightarrow 1 - R'^2 - (1 - R^2) < -\alpha$

$\Rightarrow -R'^2 + R^2 < -\alpha$

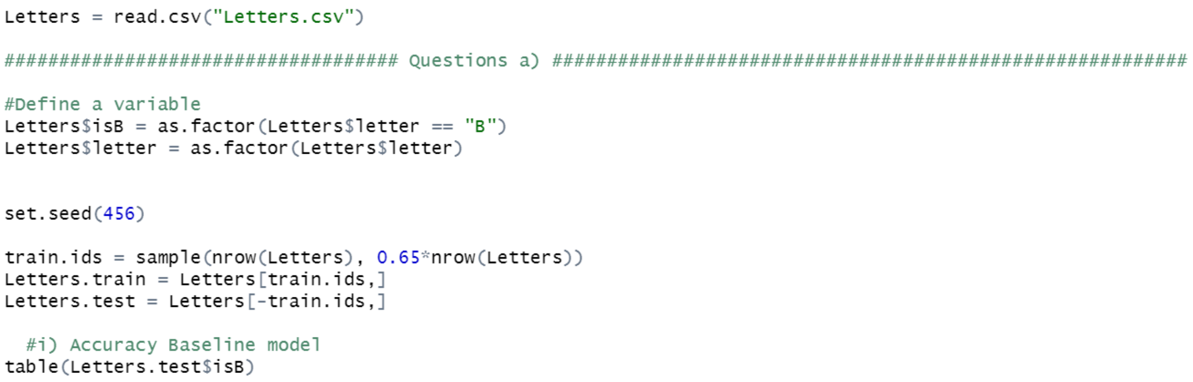
$\Rightarrow R'^2 - R^2 > \alpha$

$C_\alpha(T_{new}) \leq C_\alpha(T_{old}) \Rightarrow R^2_{new} - R^2_{old} \geq \alpha$

Problem 2: Framingham Heart Study

a)

i) Baseline method

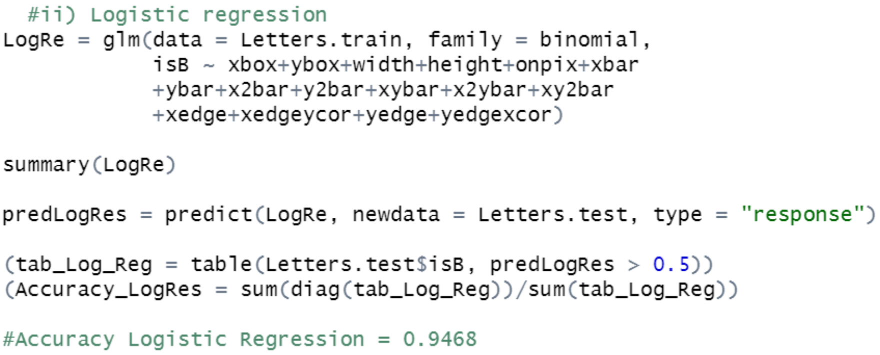


```
Letters = read.csv("Letters.csv")

################################### Questions a) ##########################################################

#Define a variable
Letters$isB = as.factor(Letters$letter == "B")
Letters$letter = as.factor(Letters$letter)


set.seed(456)

train.ids = sample(nrow(Letters), 0.65*nrow(Letters))
Letters.train = Letters[train.ids,]
Letters.test = Letters[-train.ids,]

  #i) Accuracy Baseline model
table(Letters.test$isB)
```

It always predict the most frequent outcome, “**not B”.**

$$\boldsymbol{Accuracy} = \frac{\mathbf{788}}{\mathbf{788+303}} = \mathbf{72.22\%}$$

ii) Logistic regression to predict whether or not the letter is B



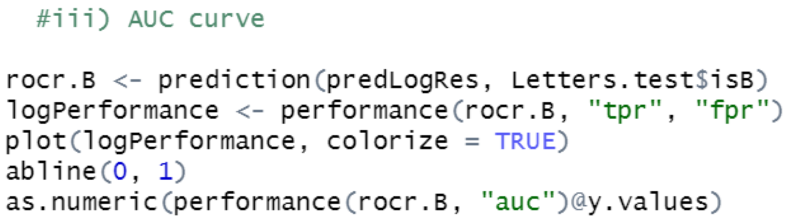
```
  #ii) Logistic regression
LogRe = glm(data = Letters.train, family = binomial,
            isB ~ xbox+ybox+width+height+onpix+xbar
            +ybar+x2bar+y2bar+xybar+x2ybar+xy2bar
            +xedge+xedgeycor+yedge+yedgexcor)

summary(LogRe)

predLogRes = predict(LogRe, newdata = Letters.test, type = "response")

(tab_Log_Reg = table(Letters.test$isB, predLogRes > 0.5))
(Accuracy_LogRes = sum(diag(tab_Log_Reg))/sum(tab_Log_Reg))

#Accuracy Logistic Regression = 0.9468
```

P(Y=1/X=x)=1/(1+exp(-(xbox+ybox+width+height+onpix+xbar+ybar+x2bar+y2bar+xybar+x2ybar+xy2bar+xedge+xedgeycor+yedge+yedgexcor)))

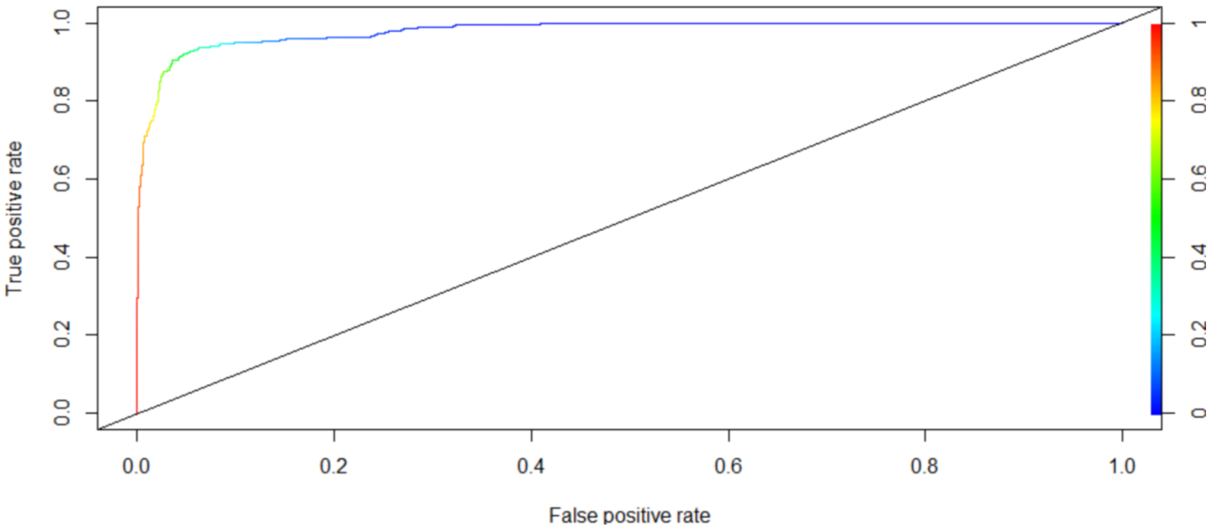
	No B P(Y=1)<0.5	**B P(Y=1)>0.5**
No B	**760**	**28**
B	**30**	**273**

$$\boldsymbol{Accuracy} = \frac{\mathbf{760+273}}{\mathbf{760+273+30+28}} = \mathbf{0.9468}$$

iii) AUC of the logistic regression



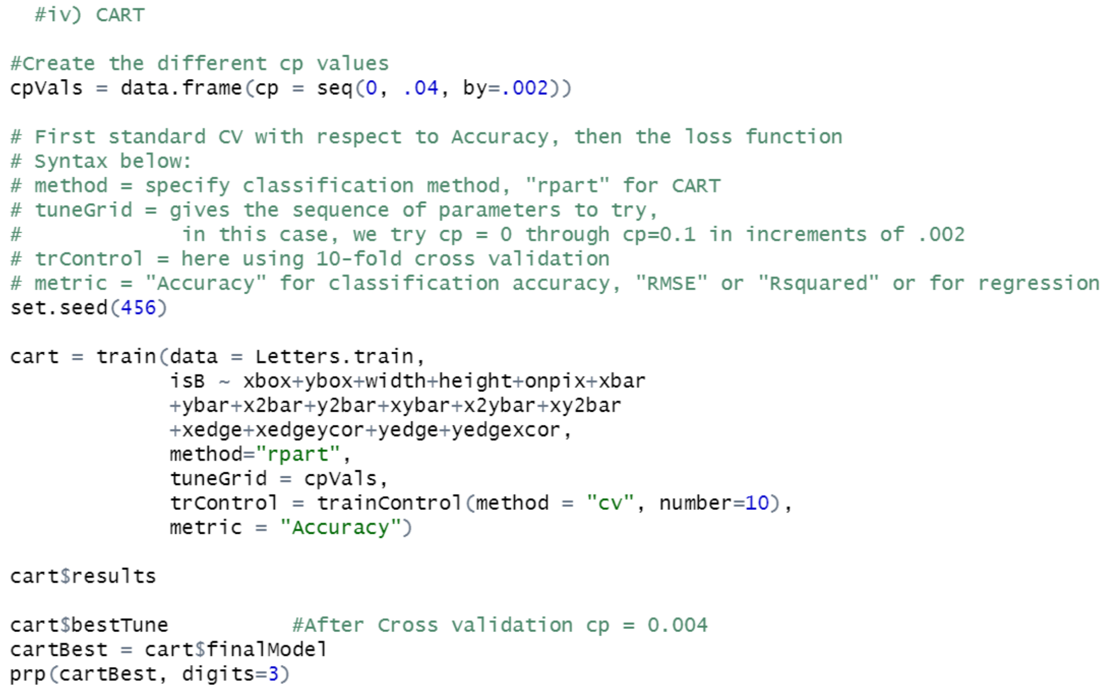
```
  #iii) AUC curve

rocr.B <- prediction(predLogRes, Letters.test$isB)
logPerformance <- performance(rocr.B, "tpr", "fpr")
plot(logPerformance, colorize = TRUE)
abline(0, 1)
as.numeric(performance(rocr.B, "auc")@y.values)
```



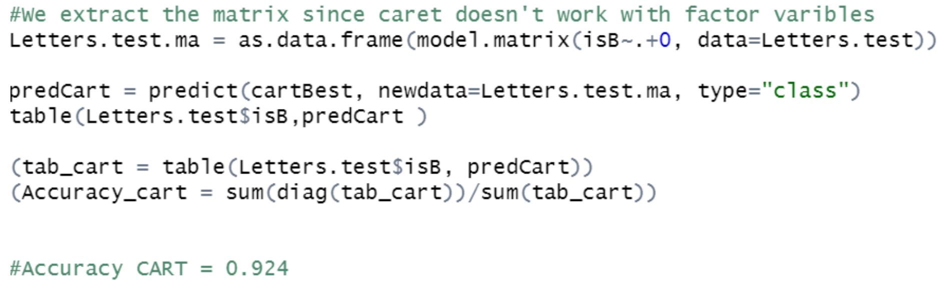
AUC = 0.9796

The regression model is a good fit to our data.

iv) Construct a CART tp predict whether or not a letter is a B



```
  #iv) CART

#Create the different cp values
cpVals = data.frame(cp = seq(0, .04, by=.002))

# First standard CV with respect to Accuracy, then the loss function
# Syntax below:
# method = specify classification method, "rpart" for CART
# tuneGrid = gives the sequence of parameters to try,
#              in this case, we try cp = 0 through cp=0.1 in increments of .002
# trControl = here using 10-fold cross validation
# metric = "Accuracy" for classification accuracy, "RMSE" or "Rsquared" or for regression
set.seed(456)

cart = train(data = Letters.train,
             isB ~ xbox+ybox+width+height+onpix+xbar
             +ybar+x2bar+y2bar+xybar+x2ybar+xy2bar
             +xedge+xedgeycor+yedge+yedgexcor,
             method="rpart",
             tuneGrid = cpVals,
             trControl = trainControl(method = "cv", number=10),
             metric = "Accuracy")

cart$results

cart$bestTune           #After Cross validation cp = 0.004
cartBest = cart$finalModel
prp(cartBest, digits=3)
```

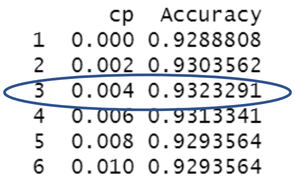


```
#We extract the matrix since caret doesn't work with factor varibles
Letters.test.ma = as.data.frame(model.matrix(isB~.+0, data=Letters.test))

predCart = predict(cartBest, newdata=Letters.test.ma, type="class")
table(Letters.test$isB,predCart )

(tab_cart = table(Letters.test$isB, predCart))
(Accuracy_cart = sum(diag(tab_cart))/sum(tab_cart))


#Accuracy CART = 0.924
```

In this study we decide to focus on accuracy as our metric. We will a 10-fold Cross validation to choose the best cp values.

In this case, we try cp = 0 through cp=0.1 in increments of 0.002. Then we select the cp that yield to the best accuracy on the trainning set.

We find cp = 0.004

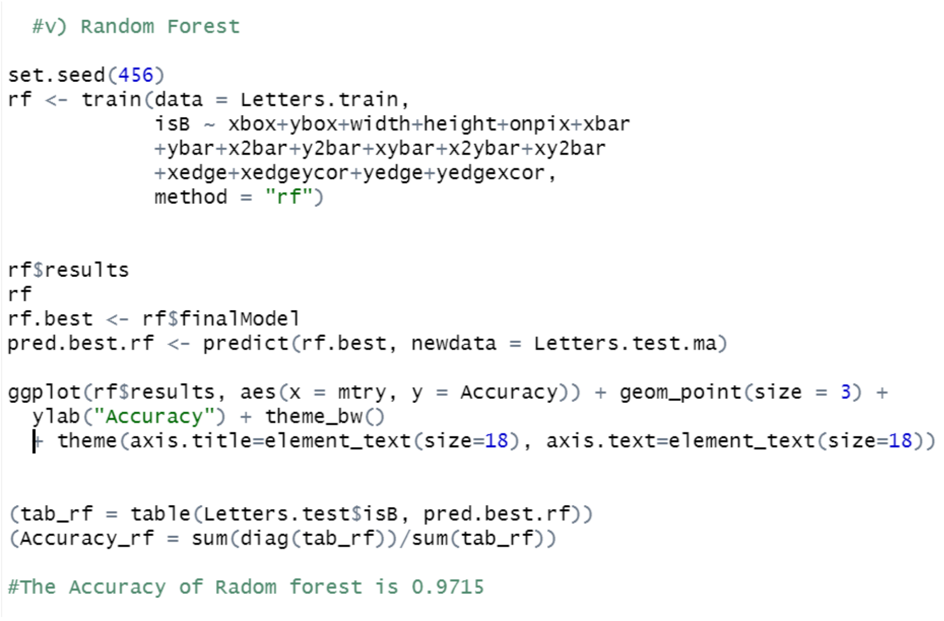


```
     cp  Accuracy
1 0.000 0.9288808
2 0.002 0.9303562
3 0.004 0.9323291
4 0.006 0.9313341
5 0.008 0.9293564
6 0.010 0.9293564
```

	No B P(Y=1)<0.5	B P(Y=1)>0.5
No B	**768**	**20**
B	**62**	**241**

$$\boldsymbol{Accuracy = \frac{768+241}{768+241+62+20} = 0.9248}$$

v) Random Forest



```
  #v) Random Forest

set.seed(456)
rf <- train(data = Letters.train,
            isB ~ xbox+ybox+width+height+onpix+xbar
            +ybar+x2bar+y2bar+xybar+x2ybar+xy2bar
            +xedge+xedgeycor+yedge+yedgexcor,
            method = "rf")


rf$results
rf
rf.best <- rf$finalModel
pred.best.rf <- predict(rf.best, newdata = Letters.test.ma)

ggplot(rf$results, aes(x = mtry, y = Accuracy)) + geom_point(size = 3) +
  ylab("Accuracy") + theme_bw()
  + theme(axis.title=element_text(size=18), axis.text=element_text(size=18))


(tab_rf = table(Letters.test$isB, pred.best.rf))
(Accuracy_rf = sum(diag(tab_rf))/sum(tab_rf))

#The Accuracy of Radom forest is 0.9715
```

	Good risk P(Y=1)<0.16	**Bad risk P(Y=1)>0.16**
No CHD	**781**	**7**
CHD	**24**	**279**

$$Accuracy = \frac{781 + 279}{781 + 279 + 24 + 7} = 0.9715$$

vii) Compare the models

In this case accuracy is much more important than interpretability since there is no clear meaning to pixels

	Baseline model	**Regression model**	**CART**	**Random Forest**
Accuracy	0.7222	0.9468	0.9248	0.9715

Baseline model < CART < Regression model < Random Forest

Random Forest has the best performance on the test set but is the less interpretable. In this case we are going to choose accuracy over interpretability because there no much information we could take from the features; they are not explicit.

b) Original problem of interest

We will try to find the best model to predict “Letter”

i) Baseline model

```
table(Letters.test$letter)

#The most frequent outcome is A with 546 occurences
```

A	B	P	R
546	**463**	**520**	**496**

Considering a base line model that would always predict the most frequent class A, we would have this accuracy on the test set

$$\boldsymbol{Accuracy = \frac{243}{243+303+283+262} = 0.2227}$$

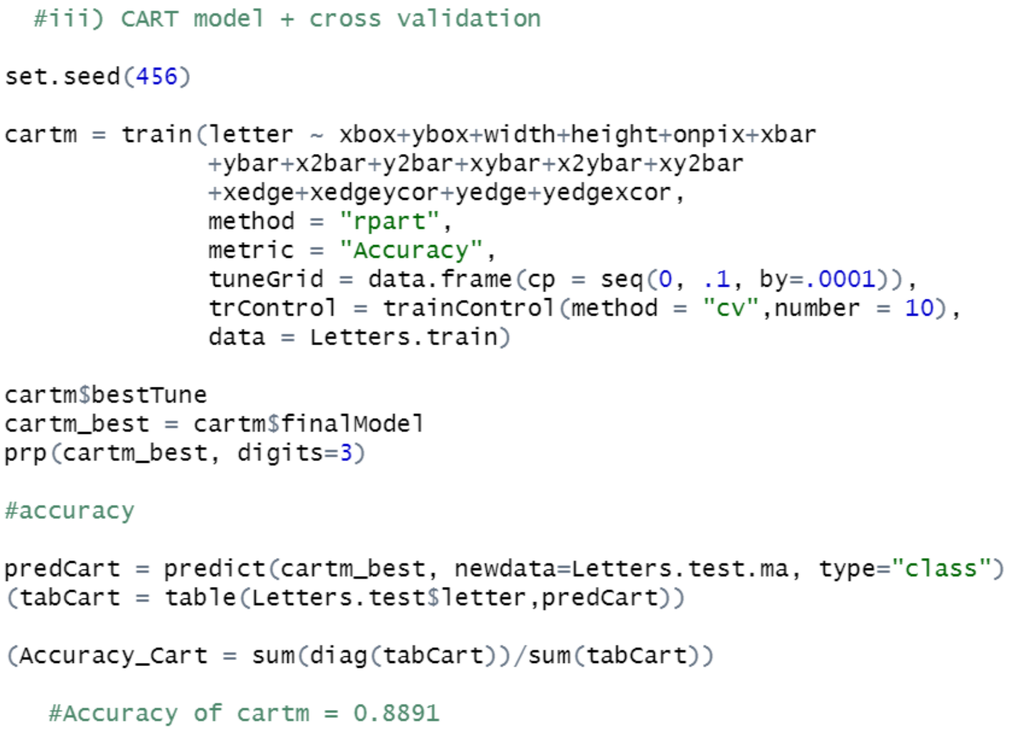
ii) LDA

```
  #ii)LDA model

ldam = lda(data = Letters.train,
           letter ~ xbox+ybox+width+height+onpix+xbar
           +ybar+x2bar+y2bar+xybar+x2ybar+xy2bar
           +xedge+xedgeycor+yedge+yedgexcor)

predLda = predict(ldam, Letters.test)
predLda_class = predLda$class

(tabLda = table(Letters.test$letter,predLda_class))

(Accuracy_Lda = sum(diag(tabLda))/sum(tabLda))

# Accuracy of Lda = 0.918
```

	A	B	P	R
A	**227**	**5**	**1**	**10**
B	**0**	**272**	**0**	**31**
P	**0**	**6**	**273**	**4**
R	**0**	**31**	**1**	**230**

$$\boldsymbol{Accuracy = 0.9184}$$

iii) CART model + Cross Validation



```
  #iii) CART model + cross validation

set.seed(456)

cartm = train(letter ~ xbox+ybox+width+height+onpix+xbar
              +ybar+x2bar+y2bar+xybar+x2ybar+xy2bar
              +xedge+xedgeycor+yedge+yedgexcor,
              method = "rpart",
              metric = "Accuracy",
              tuneGrid = data.frame(cp = seq(0, .1, by=.0001)),
              trControl = trainControl(method = "cv",number = 10),
              data = Letters.train)

cartm$bestTune
cartm_best = cartm$finalModel
prp(cartm_best, digits=3)

#accuracy

predCart = predict(cartm_best, newdata=Letters.test.ma, type="class")
(tabCart = table(Letters.test$letter,predCart))

(Accuracy_Cart = sum(diag(tabCart))/sum(tabCart))

   #Accuracy of cartm = 0.8891
```

- Cross validation

First I decided to do a 10-fold cross validation to find the cp parameter with the best accuracy on the training set. I tried the same values of the first CART cp = 0 through cp=0.1 in increments of 0.002. We found cp = 0 with an accuracy of 0.886

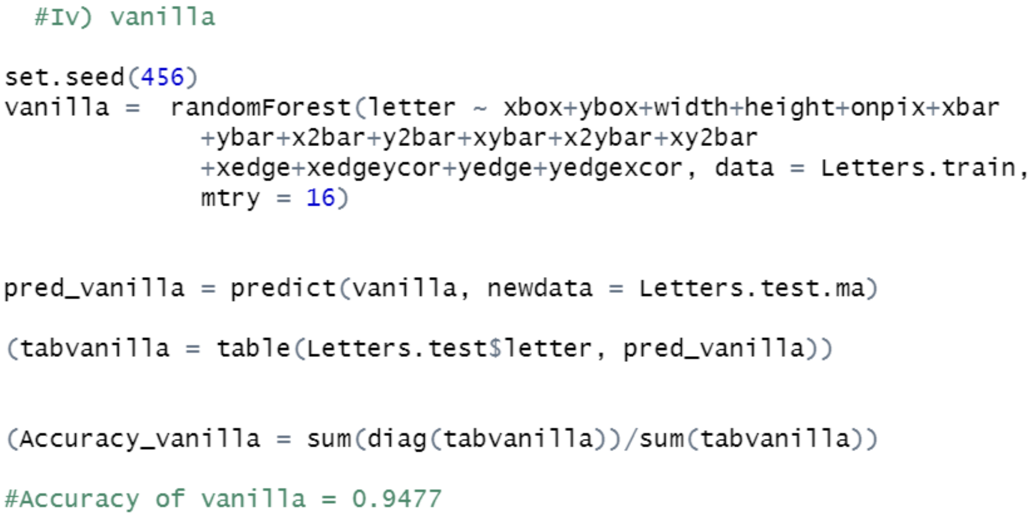
Then, we changed the increment to 0.0001. We find a **cp = 0.0007** with an accuracy of 0.91 on the trainning set.

- Results

	A	**B**	**P**	**R**
A	**230**	**5**	**1**	**7**
B	**10**	**253**	**10**	**30**
P	**2**	**13**	**265**	**3**
R	**10**	**27**	**3**	**222**

$$\boldsymbol{Accuracy = 0.8891}$$

iv) Vanilla



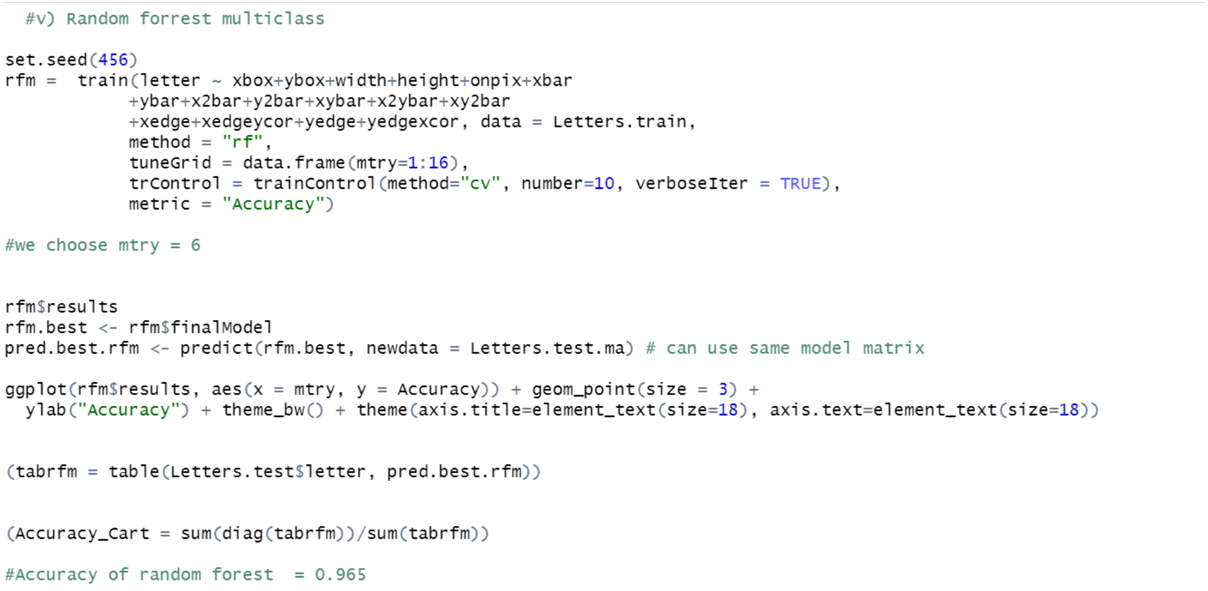
```
  #Iv) vanilla

set.seed(456)
vanilla =  randomForest(letter ~ xbox+ybox+width+height+onpix+xbar
             +ybar+x2bar+y2bar+xybar+x2ybar+xy2bar
             +xedge+xedgeycor+yedge+yedgexcor, data = Letters.train,
             mtry = 16)


pred_vanilla = predict(vanilla, newdata = Letters.test.ma)

(tabvanilla = table(Letters.test$letter, pred_vanilla))


(Accuracy_vanilla = sum(diag(tabvanilla))/sum(tabvanilla))

#Accuracy of vanilla = 0.9477
```

Through the Random Forest model we will set **mtry = 16** to examine all the variables at each spit of fitted CART trees

	A	**B**	**P**	**R**
A	**236**	**4**	**1**	**2**
B	**2**	**280**	**2**	**19**
P	**0**	**2**	**278**	**3**
R	**1**	**19**	**2**	**240**

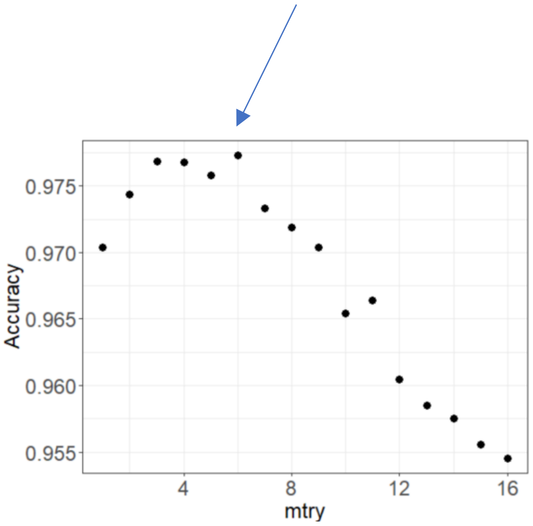
$$Accuracy = \mathbf{0.947}$$

v) Random Forest + Cross validation



```
  #v) Random forrest multiclass

set.seed(456)
rfm =  train(letter ~ xbox+ybox+width+height+onpix+xbar
          +ybar+x2bar+y2bar+xybar+x2ybar+xy2bar
          +xedge+xedgeycor+yedge+yedgexcor, data = Letters.train,
          method = "rf",
          tuneGrid = data.frame(mtry=1:16),
          trControl = trainControl(method="cv", number=10, verboseIter = TRUE),
          metric = "Accuracy")

#we choose mtry = 6


rfm$results
rfm.best <- rfm$finalModel
pred.best.rfm <- predict(rfm.best, newdata = Letters.test.ma) # can use same model matrix

ggplot(rfm$results, aes(x = mtry, y = Accuracy)) + geom_point(size = 3) +
  ylab("Accuracy") + theme_bw() + theme(axis.title=element_text(size=18), axis.text=element_text(size=18))


(tabrfm = table(Letters.test$letter, pred.best.rfm))


(Accuracy_Cart = sum(diag(tabrfm))/sum(tabrfm))

#Accuracy of random forest  = 0.965
```

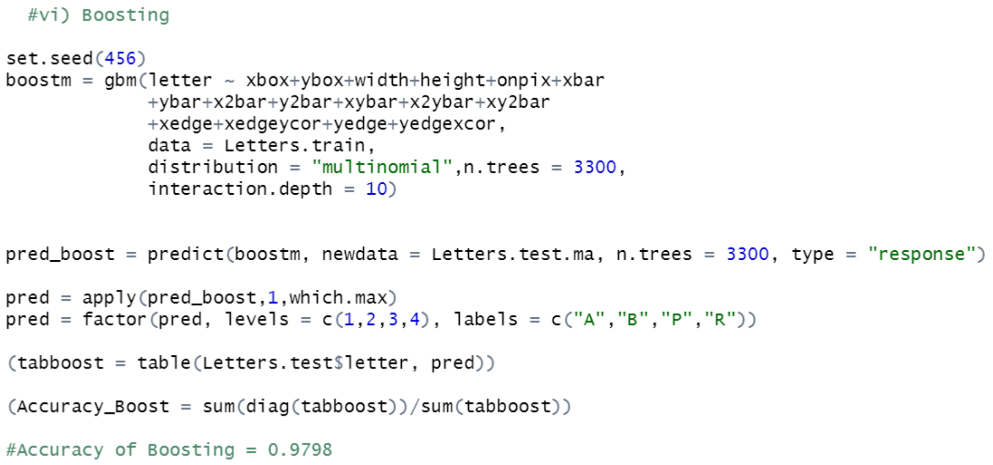
We will use a 10 fold cross validation to select mtry (number of variables examined at each split 1..16), number of trees = 500. We find mtry = 6 that yield to accuracy 0.9787 on the training set.



	A	B	P	R
A	**238**	**2**	**1**	**2**
B	**1**	**286**	**0**	**16**
P	**0**	**2**	**279**	**2**
R	**0**	**10**	**2**	**250**

$$Accuracy = 0.9651$$

vi) Boosting



```
  #vi) Boosting

set.seed(456)
boostm = gbm(letter ~ xbox+ybox+width+height+onpix+xbar
             +ybar+x2bar+y2bar+xybar+x2ybar+xy2bar
             +xedge+xedgeycor+yedge+yedgexcor,
             data = Letters.train,
             distribution = "multinomial",n.trees = 3300,
             interaction.depth = 10)


pred_boost = predict(boostm, newdata = Letters.test.ma, n.trees = 3300, type = "response")

pred = apply(pred_boost,1,which.max)
pred = factor(pred, levels = c(1,2,3,4), labels = c("A","B","P","R"))

(tabboost = table(Letters.test$letter, pred))

(Accuracy_Boost = sum(diag(tabboost))/sum(tabboost))

#Accuracy of Boosting = 0.9798
```

We set the interaction depth to 10, run the method for 3300 iterations. With a 5-fold cross validation we have:

	A	B	P	R
A	**240**	**2**	**0**	**1**
B	**0**	**295**	**0**	**8**
P	**0**	**1**	**277**	**5**
R	**0**	**5**	**0**	**257**

$$\boldsymbol{Accuracy = 0.9798}$$

vii) Comparison

	LDA	**CART**	**Vanilla (bagging)**	**Random Forest**	**Boosting**
Accuracy	0.918	0.889	0.947	0.965	0.980

CART < LDA < Vanilla < Random Forest < Boosting

Boosting is more accurate than Random Forest on the test set. Thus we recommend using the first one since for the same reasons as before we choose to give more importance in this study to accuracy than interpretability (Even if we find the most impactful features, it would be hard to give them a concrete explanation).