BIOST 546 Homework 2

Ivy Zhang

1/29/2022

Problem 1

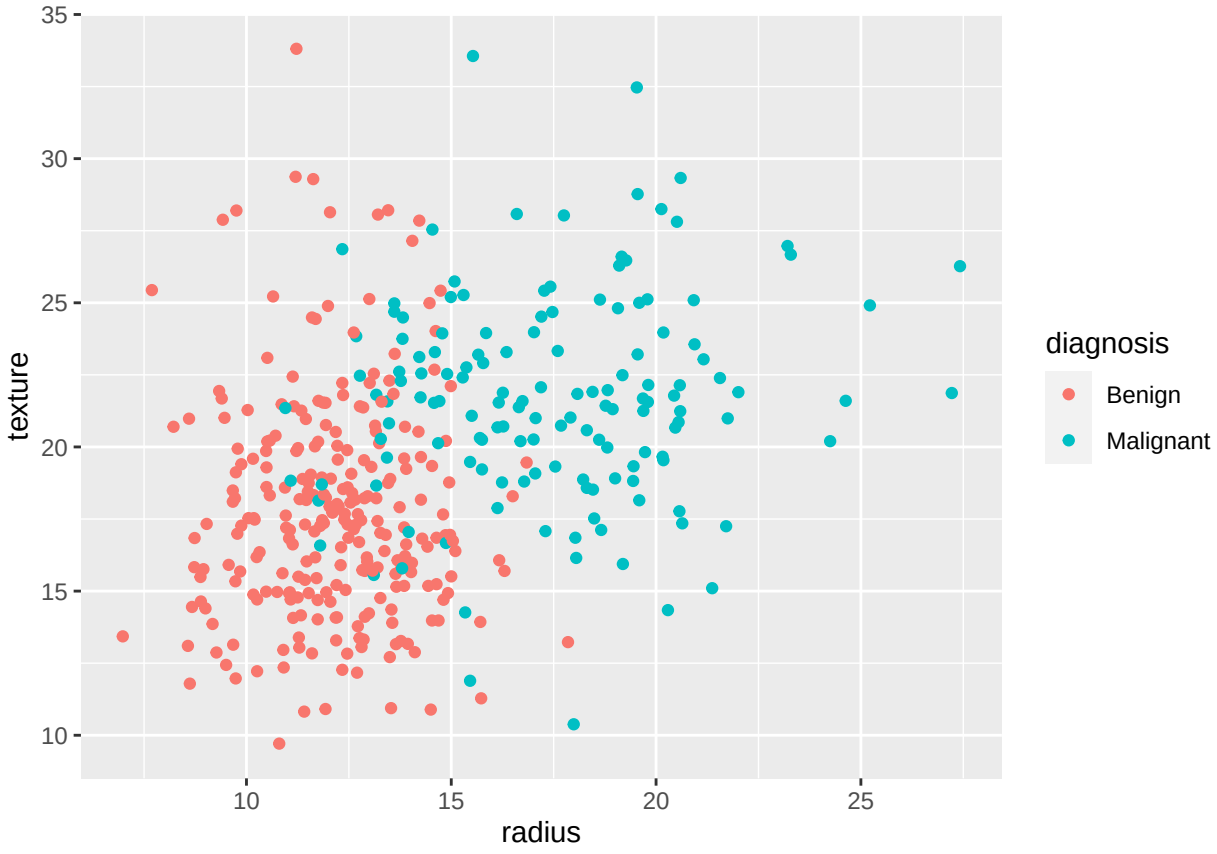
part(a)

The data has sample size of 569, with malignant class has 212 observations and benign class has 367 observations. The data also include 30 predictors.

part(b)

Work is using code attached in the appendix.

part(c)



Based on the scatterplot, I think it is almost impossible to one hundred percent accurately predict the diagnosis using only radius and texture. Although when raidus is over 17 and smaller than 10, the classification is obvious. The observations have radius between 10 and 17 may have different diagnosis but similar radisus and texture. Therefore, it is very hard for us to be one hundred percent accurately to predict the diagnosis with only radius and texture information.

Part(d)

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-19.384	2.015	-9.619	0
radius	0.984	0.111	8.841	0
texture	0.241	0.045	5.309	0

Interpretation:

Intercept: We estimate that when texture equals to 0 and radius equals to 0, the log-odds of participants getting malignant is -19.384.

Radius: We estimate that when radius increase by one unit, the log-odds of participants getting malignant will increase by 0.984.

Texture: We estimate that when texture increase by one unit, the log-odds of participants getting malignant will increase by 0.241.

Part(e)

Based on the part(d), we have only benign and malignant, therefore$Pr(D = Benign) = 1 - Pr(D = Malignant)$ and

$$log(\frac{Pr(D = Malignant)}{1 - Pr(D = Malignant)}) = -19.384 + 0.984 \times radius + 0.241 \times texture$$

. Therefore, if we have radius of 10, and texture equals to 12, then we can estimate that:

$$log(\frac{Pr(D = Malignant)}{1 - Pr(D = Malignant)}) = -19.384 + 0.984 \times 10 + 0.241 \times 12 = -6.652$$

Therefore we have

$$\frac{Pr(D = Malignant)}{1 - Pr(D = Malignant)} = e^{-6.652} = 0.00129$$

Then we can calculus that

$$Pr(D = Malignant) = \frac{0.00129}{1 + 0.00129} = 0.00129$$

Then we can use predict function to compare with our result. Predict function gave an answer that is 0.00129 which is same as our calculation. I believe that may due to the rounding problem in our calculation.

Part(f)

We first compute the confusion matrix of training set and test set.

Table 2: Confusion Matrix of Training Set

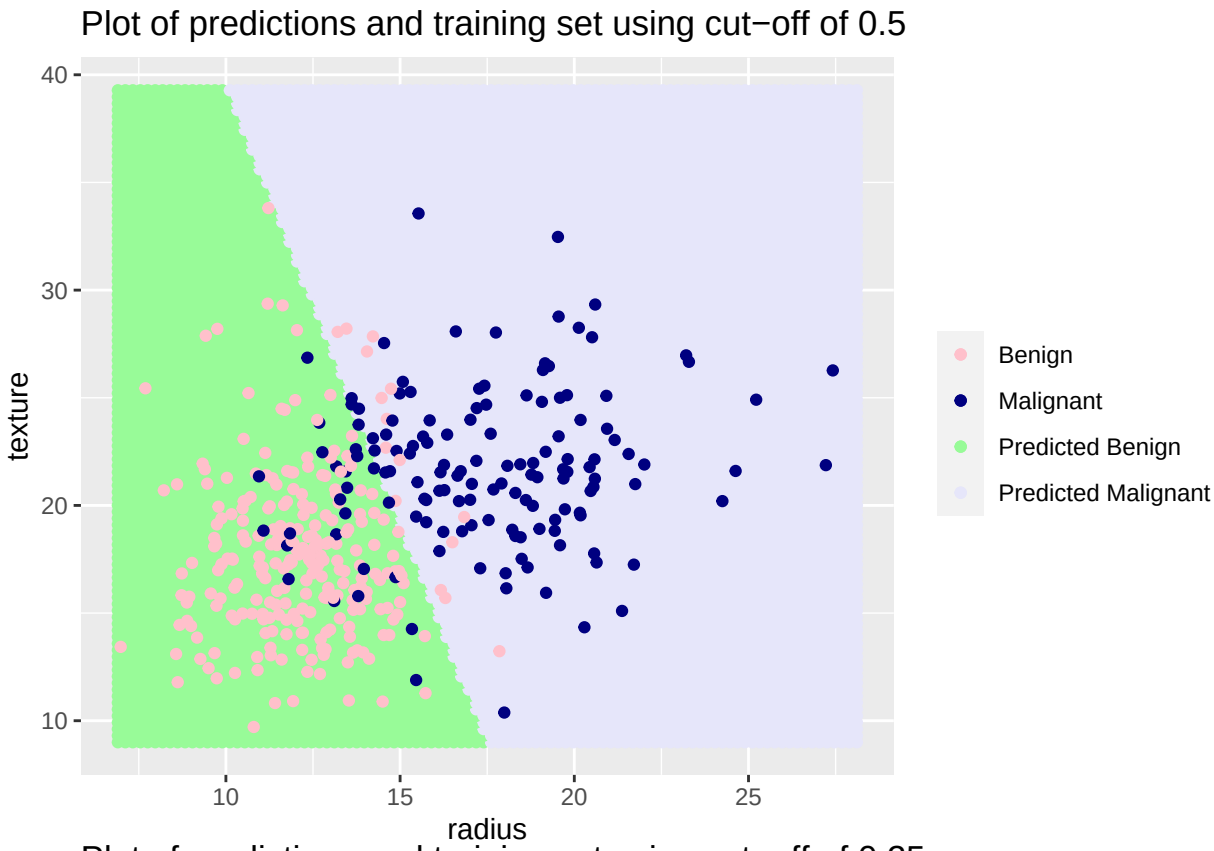
	Benign	Malignant
Benign	242	26
Malignant	15	117

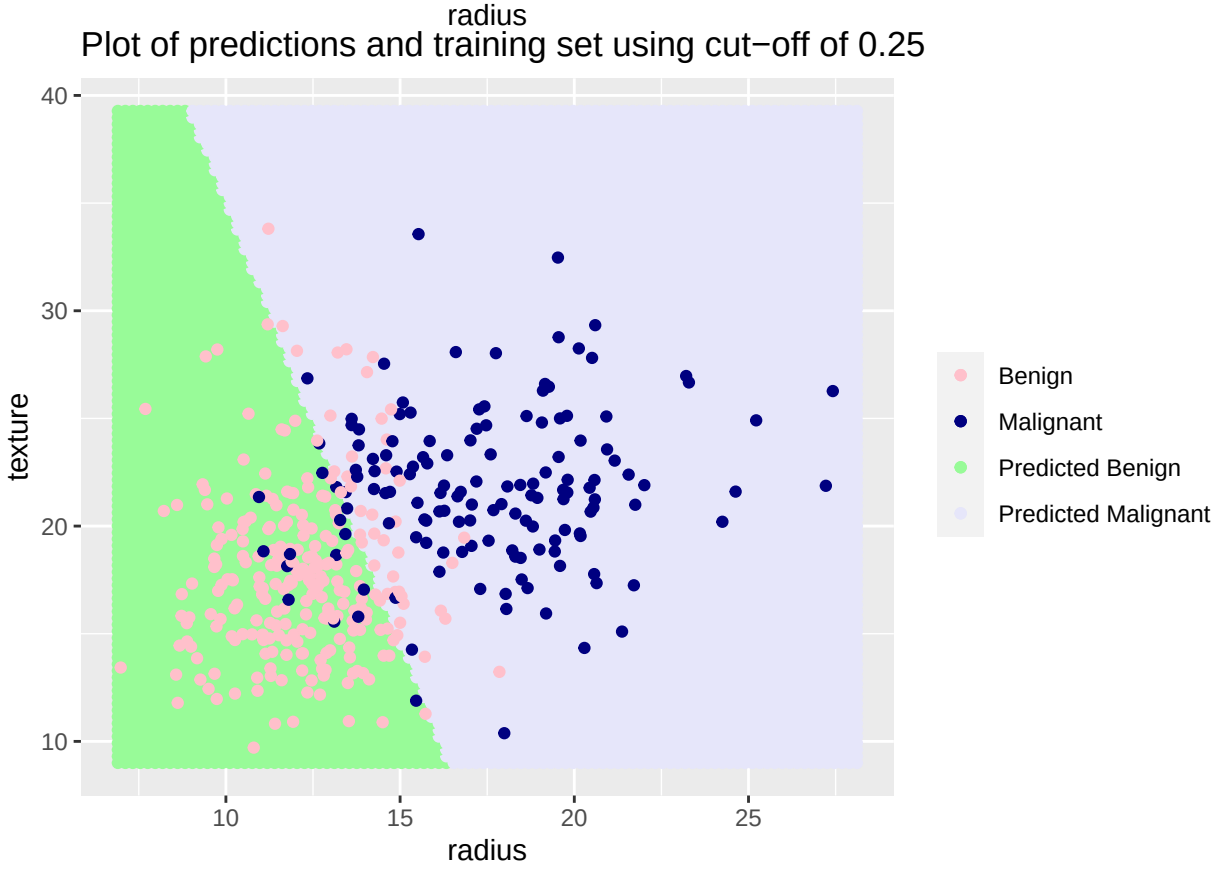
Table 3: Confusion Matrix of Test Set

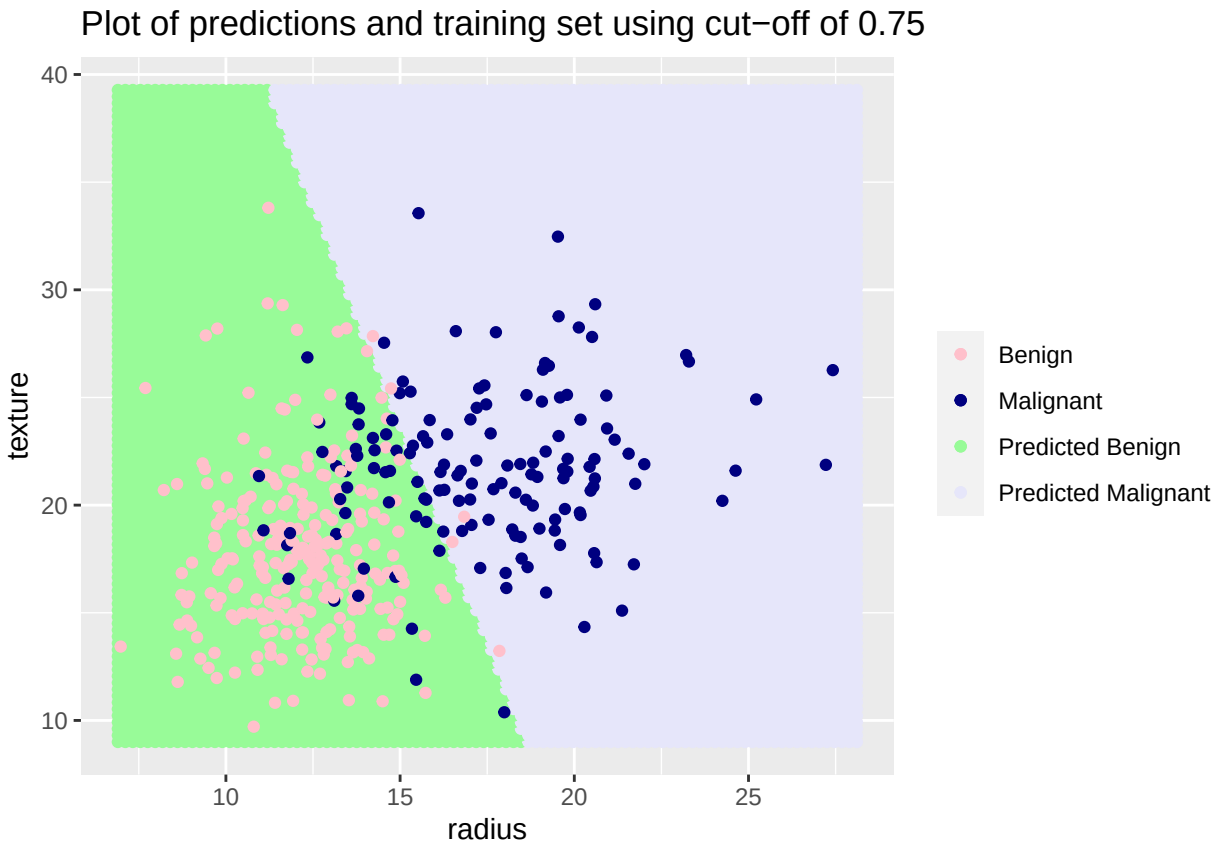
	Benign	Malignant
Benign	94	15
Malignant	6	54

Our prediction accuracy of trainning set is 0.898, but the test set's prediction accuracy is 0.876. Overall, the prediction is doing great job in both training set and test set.The logistic regression model seeme giving false positive rate a little bit higher than the false negative rate.

Part(g)

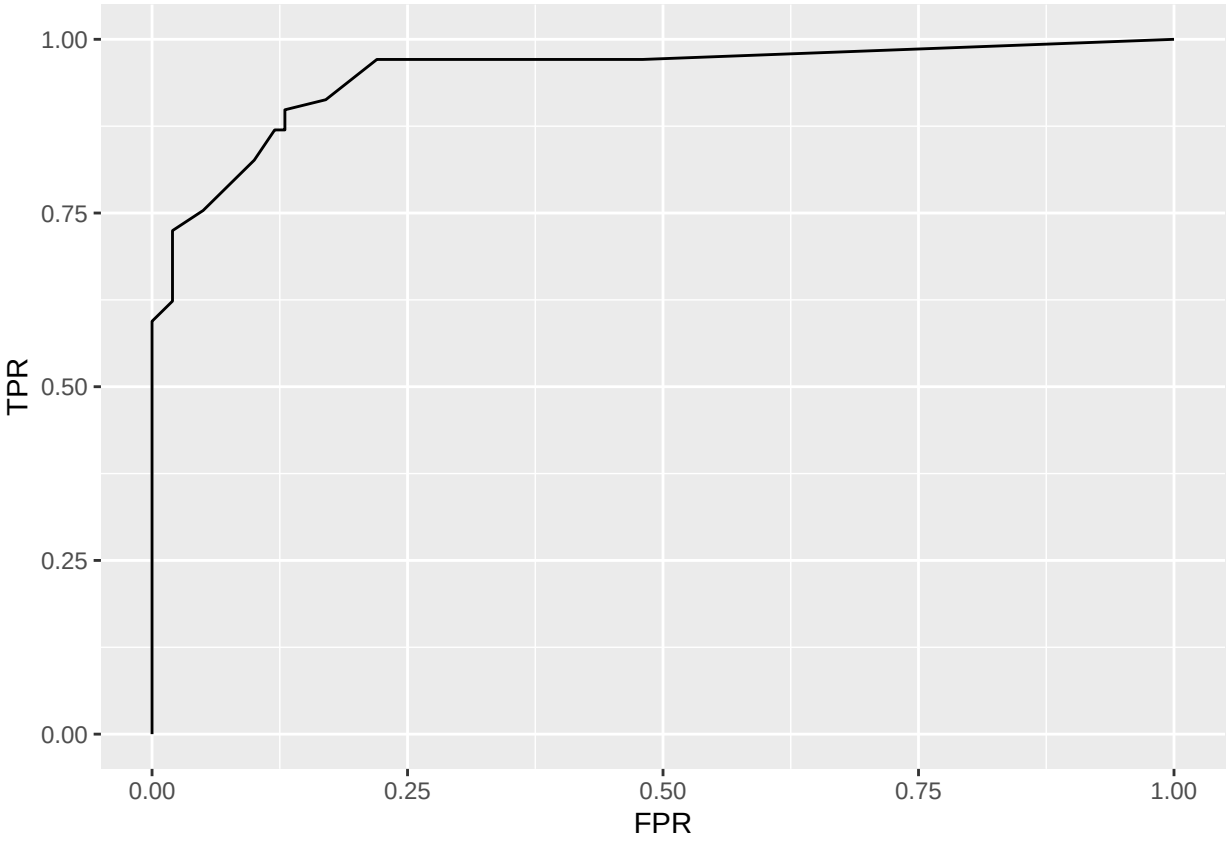

Plot of predictions and training set using cut−off of 0.5
texture
radius
40
30
20
10
10
15
20
25
Benign
Malignant
Predicted Benign
Predicted Malignant


Plot of predictions and training set using cut−off of 0.25
texture
radius
40
30
20
10
10
15
20
25
Benign
Malignant
Predicted Benign
Predicted Malignant



As we can see, the slopes of the decision boundary are very similar for different threshold. When our cut off probability become larger, the decision boundary is moving to the right. In other words, ss cut-off probability become larger and larger, we are classifying more observations to benign instad of malignant.

Part(h)



Problem 2

Part(a)

	Group Means: Texture	Group Mean: Radius	Prior Probabilities
Benign	17.769	12.146	0.6425
Malignant	21.661	17.479	0.3575

We estimate that if we randomly draw an observation from our data will have a probability of 0.36 fall into Malignant and probability of 0.64 falling into benign. The estimated group mean is the mean of the normal distribution of different classes which the density of an observation from that class is draw from, and the distributions have same variance. The products of these two quantities lead to our final estimation of the posterior probabilities.

Part(b)

Table 5: Confusion Matrix of Training Set

	Benign	Malignant
Benign	247	33
Malignant	10	110

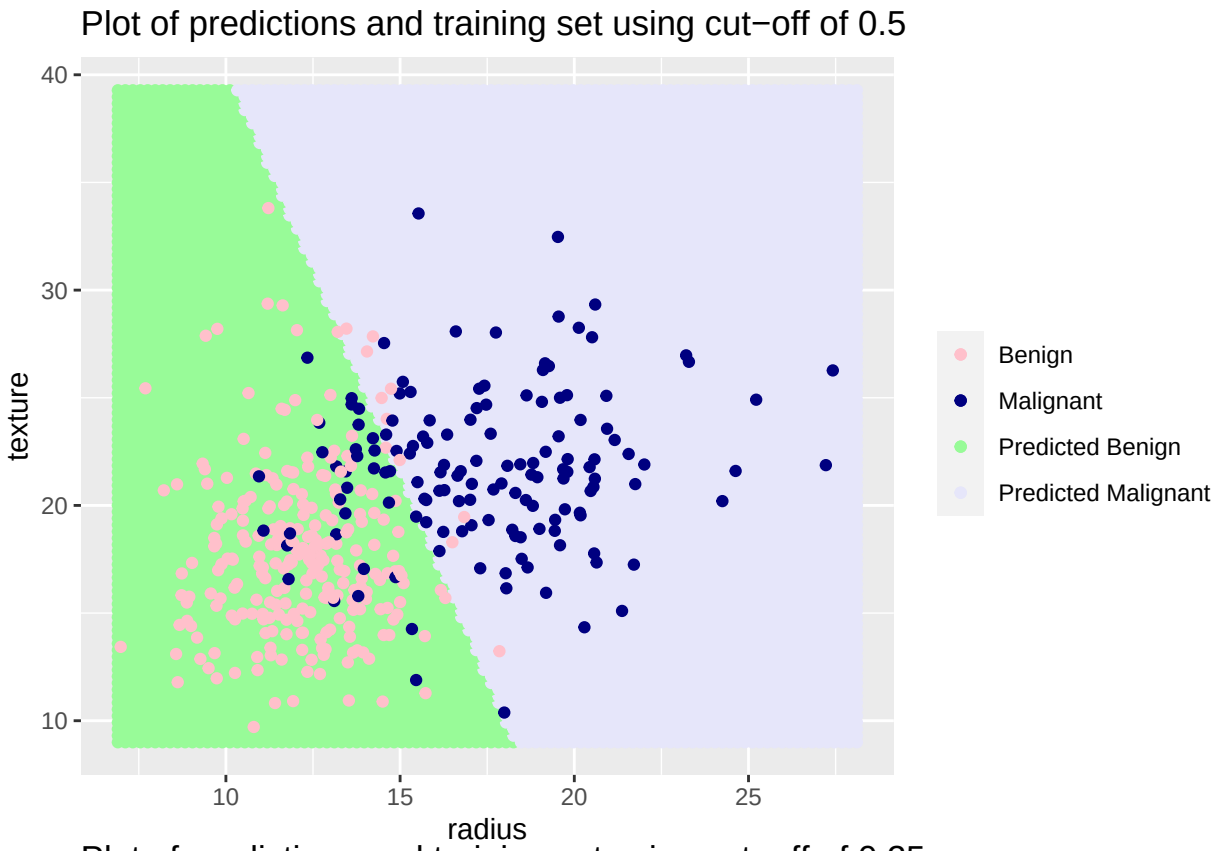
Table 6: Confusion Matrix of Test Set

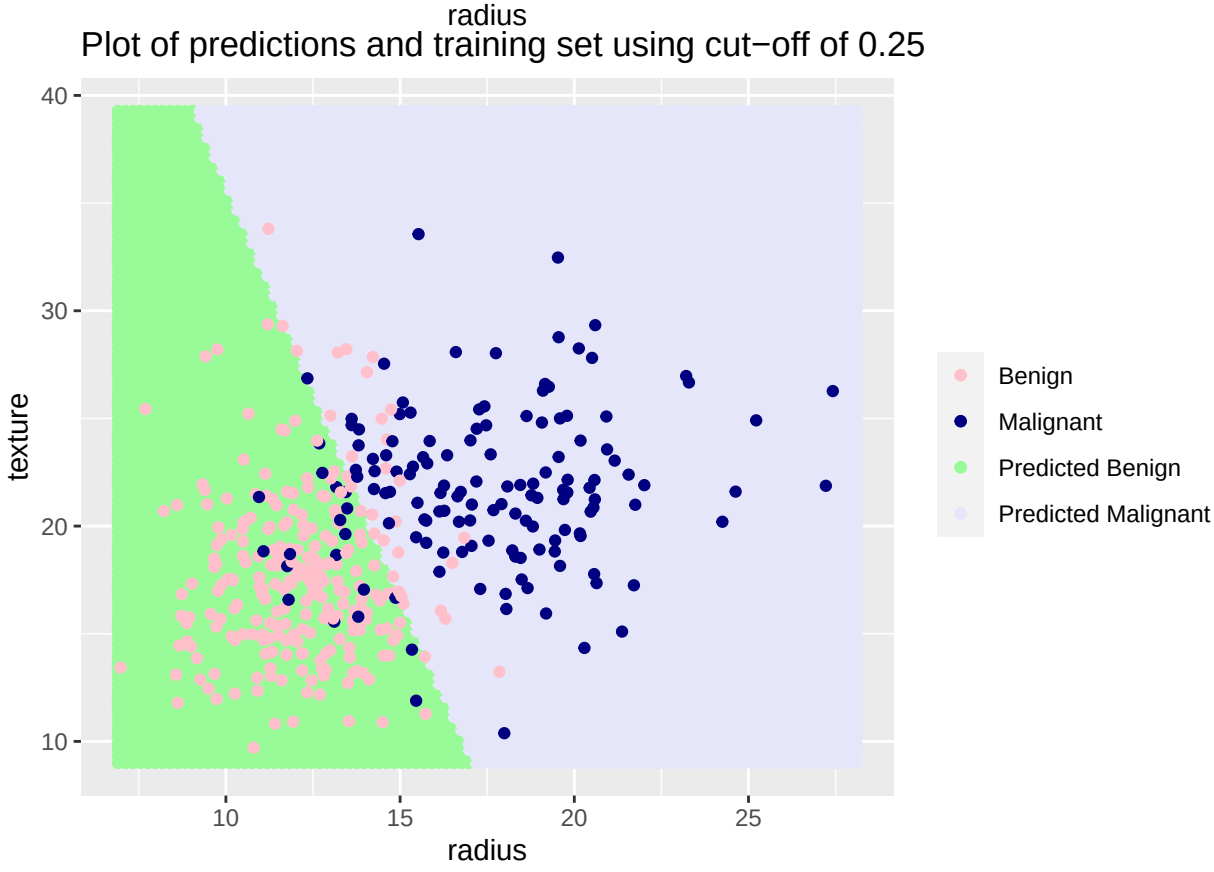
	Benign	Malignant
Benign	98	21
Malignant	2	48

The accuracy of the training set os 0.863 and the accuracy of the test set is 0.929.

As we can see, the test set accuracy is higher than the trainning set accuracy. The model giving false positive rate higher than the false negative rate in the both datasets.

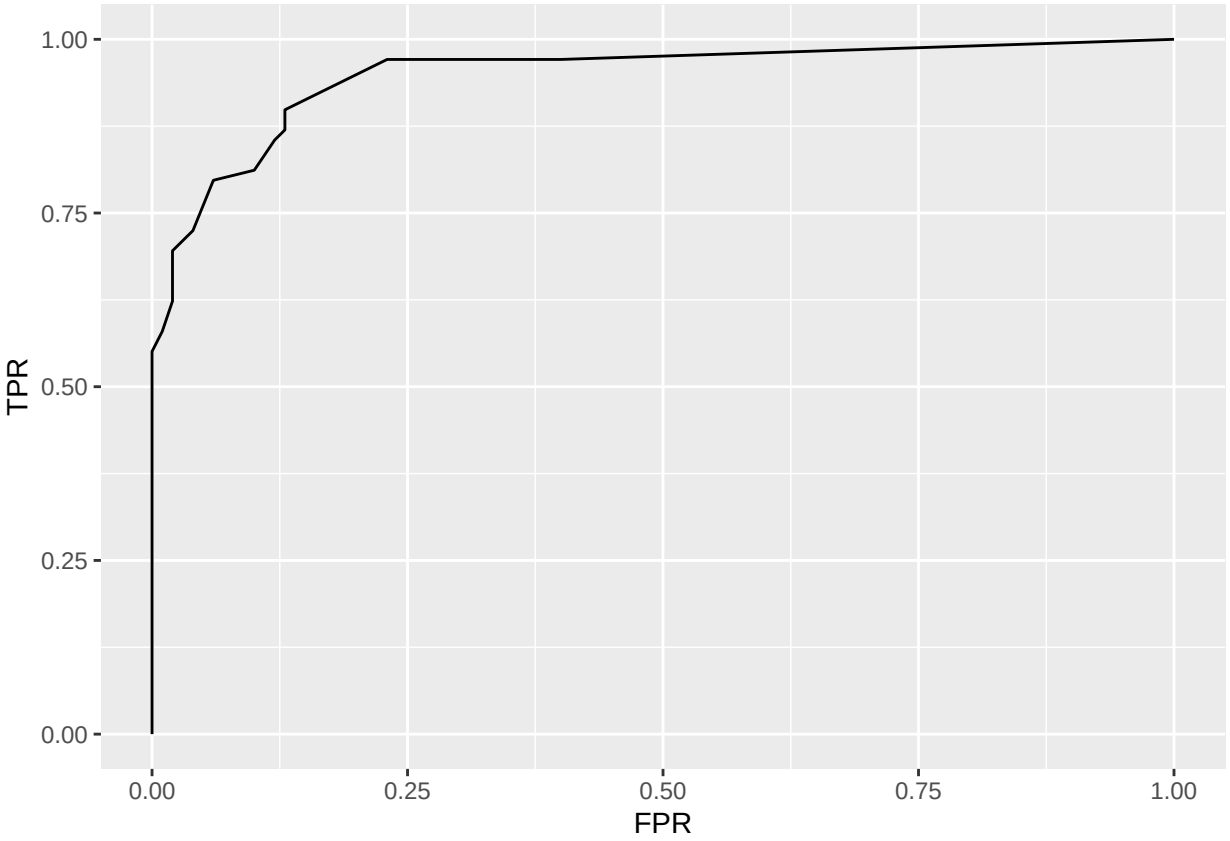
Part(c)


Plot of predictions and training set using cut−off of 0.5
texture
radius
40
30
20
10
10
15
20
25
Benign
Malignant
Predicted Benign
Predicted Malignant


Plot of predictions and training set using cut−off of 0.25
texture
radius
40
30
20
10
10
15
20
25
Benign
Malignant
Predicted Benign
Predicted Malignant

The slopes of the decision boundary are very similar and seems to be linear visisually. As the cut-off become larger and larger, the decision boundary is moving to the right, classifying more observations into benign instead of malignant. When the cut-off probability is 0.75, we can clearly see a relatively high proportion of misclassification.

Part(d)



Part (e)

We calculated the estimated AUC is 0.948.

Problem 3

Part(a)

	Group Means: Texture	Group Mean: Radius	Prior Probabilities
Benign	17.769	12.146	0.6425
Malignant	21.661	17.479	0.3575

We estimate that if we randomly draw an observation from our data will have a probability of 0.64 fall into Malignant and probability of 0.39 falling into benign. The estimated group mean is the mean of the normal distribution of different classes which the density of an observation from that class is draw from and the variance of the distribution are different. The products of these two quantities lead to our final estimation of the posterior probabilities.

Part(b)

Table 8: Confusion Matrix of Training Set

	Benign	Malignant
Benign	244	31
Malignant	13	112

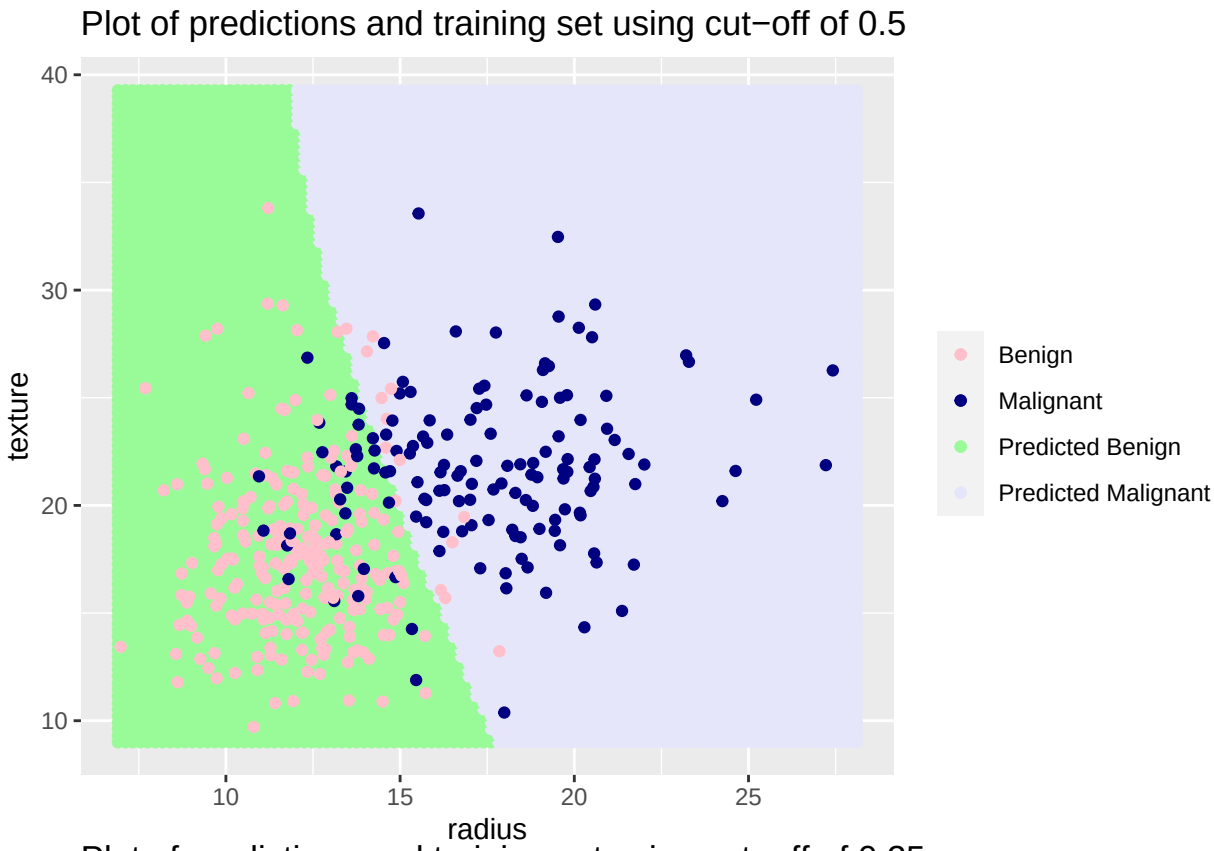
Table 9: Confusion Matrix of Test Set

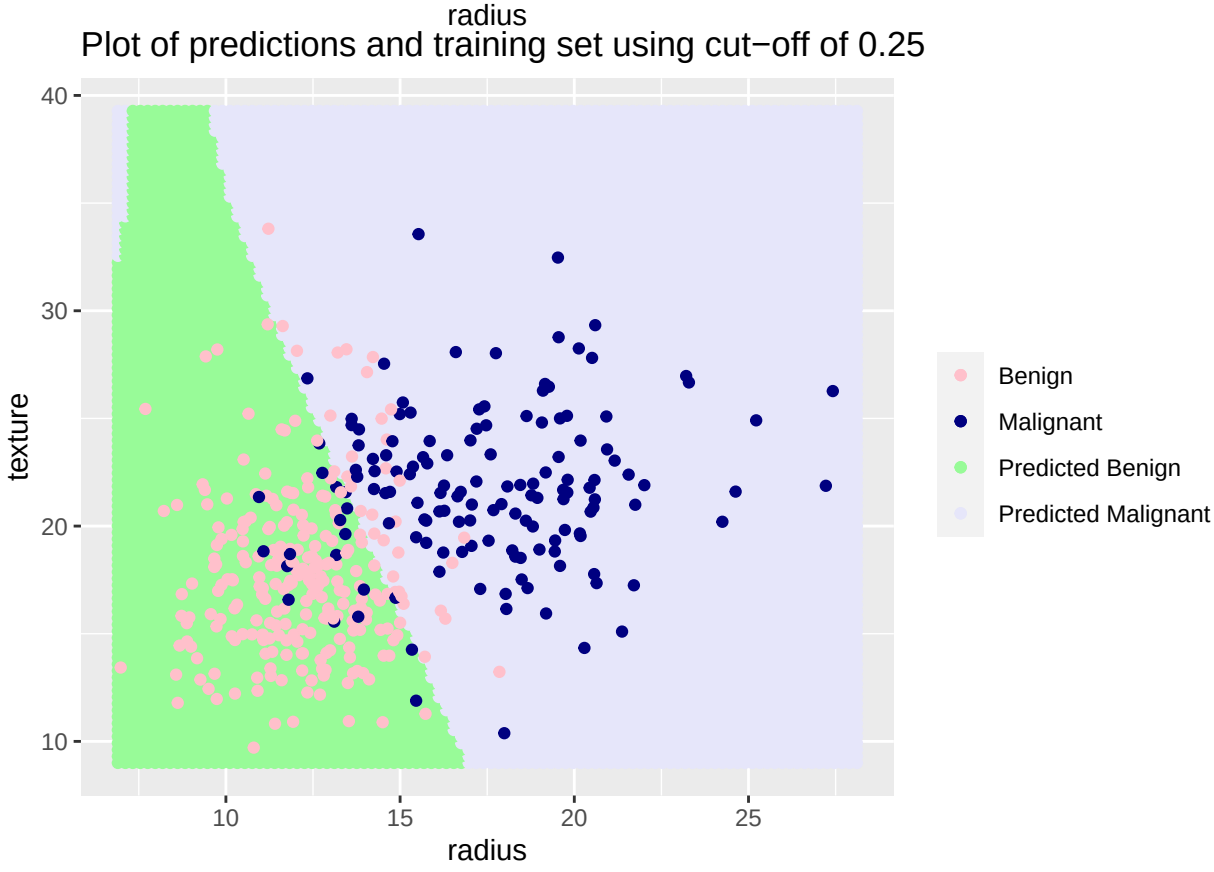
	Benign	Malignant
Benign	98	20
Malignant	2	49

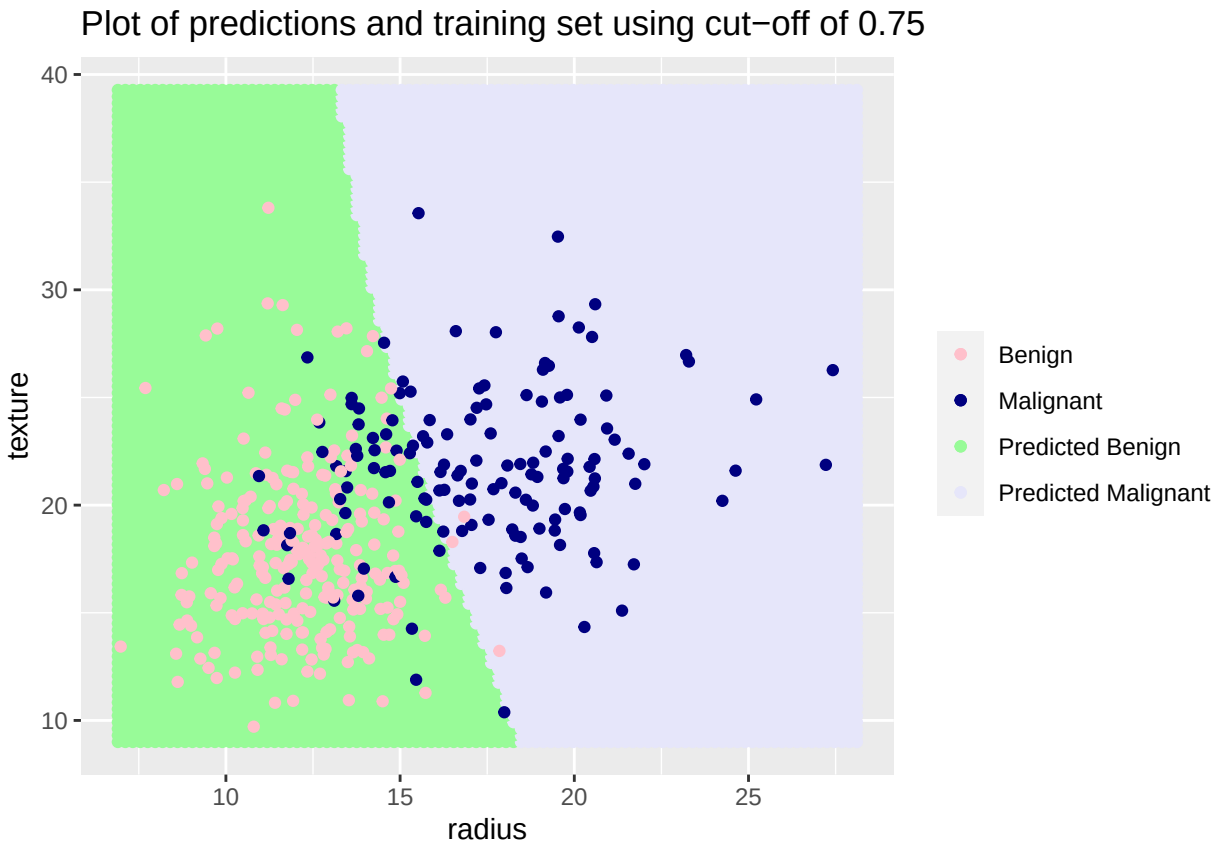
The accuracy of the training set os 0.89 and the accuracy of the test set is 0.870.

As we can see, the test set accuracy is higher than the training set accuracy. The model giving false positive rate higher than the false negative rate in the both datasets. Overall, the table, accuracy and rates are very similar between LDA and QDA.

Part(c)

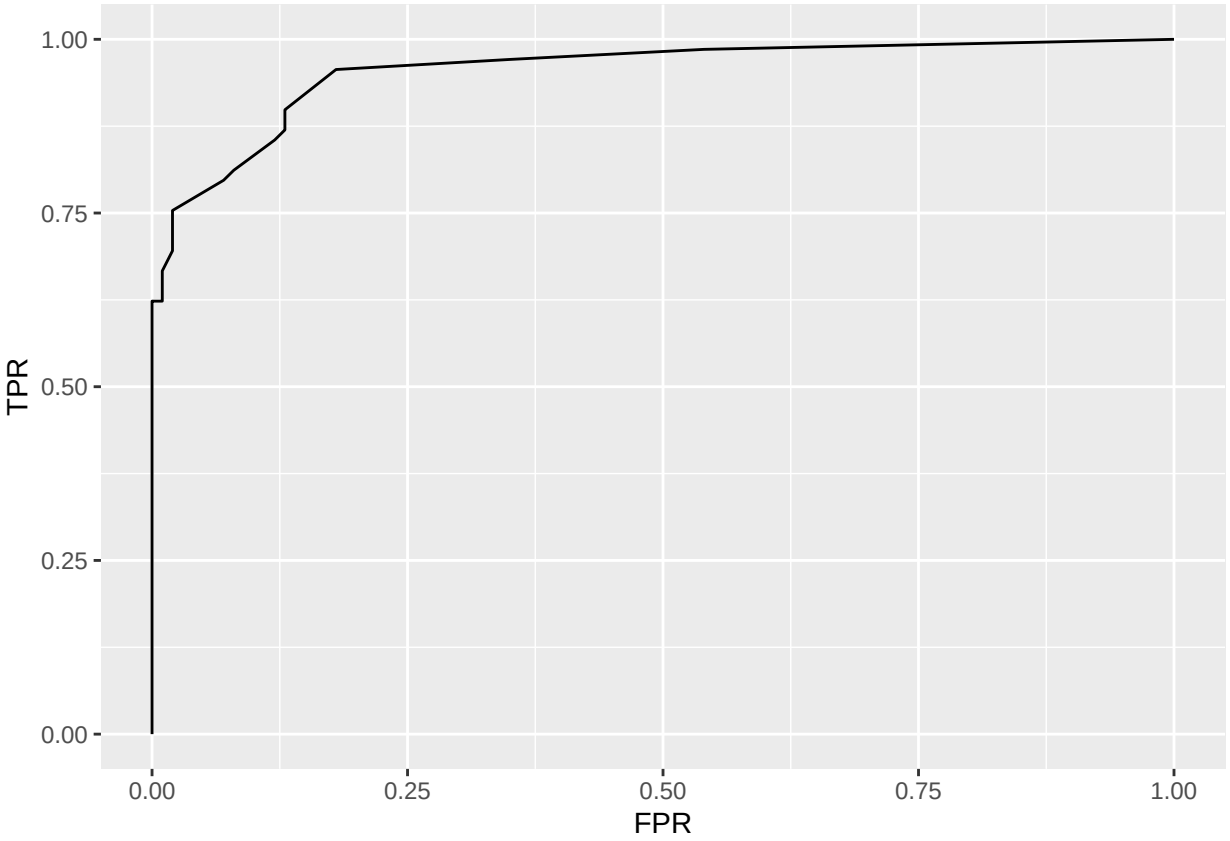

Plot of predictions and training set using cut−off of 0.5
texture
radius
40
30
20
10
10
15
20
25
Benign
Malignant
Predicted Benign
Predicted Malignant


Plot of predictions and training set using cut−off of 0.25
texture
radius
40
30
20
10
10
15
20
25
Benign
Malignant
Predicted Benign
Predicted Malignant



The slopes of the decision boundary are very similar and seems to not be linear anymore visisually, which is very different from LDA and logistic regression. As the cut-off become larger and larger, the decision boundary is moving to the right, classifying more observations into benign instead of malignant. When the cut-off probability is 0.75, we can clearly see a relatively high proportion of misclassification.

Part(d)



Part (e)

We calcualted the estimated AUC is 0.955.

Problem 4

Table 10: Confusion Table for Training Set with K = 1

	Benign	Malignant
Benign	257	0
Malignant	0	143

Table 11: Confusion Table for Test Set with K = 1

	Benign	Malignant
Benign	94	22
Malignant	6	47

Table 12: Confusion Table for Training Set with K = 2

	Benign	Malignant
Benign	244	19
Malignant	13	124

Table 13: Confusion Table for Test Set with K = 2

	Benign	Malignant
Benign	95	22
Malignant	5	47

Table 14: Confusion Table for Training Set with K = 3

	Benign	Malignant
Benign	247	23
Malignant	10	120

Table 15: Confusion Table for Test Set with K = 3

	Benign	Malignant
Benign	100	22
Malignant	0	47

Table 16: Confusion Table for Training Set with K = 4

	Benign	Malignant
Benign	246	22
Malignant	11	121

Table 17: Confusion Table for Test Set with K = 4

	Benign	Malignant
Benign	99	20
Malignant	1	49

Table 18: Confusion Table for Training Set with K = 20

	Benign	Malignant
Benign	246	22
Malignant	11	121

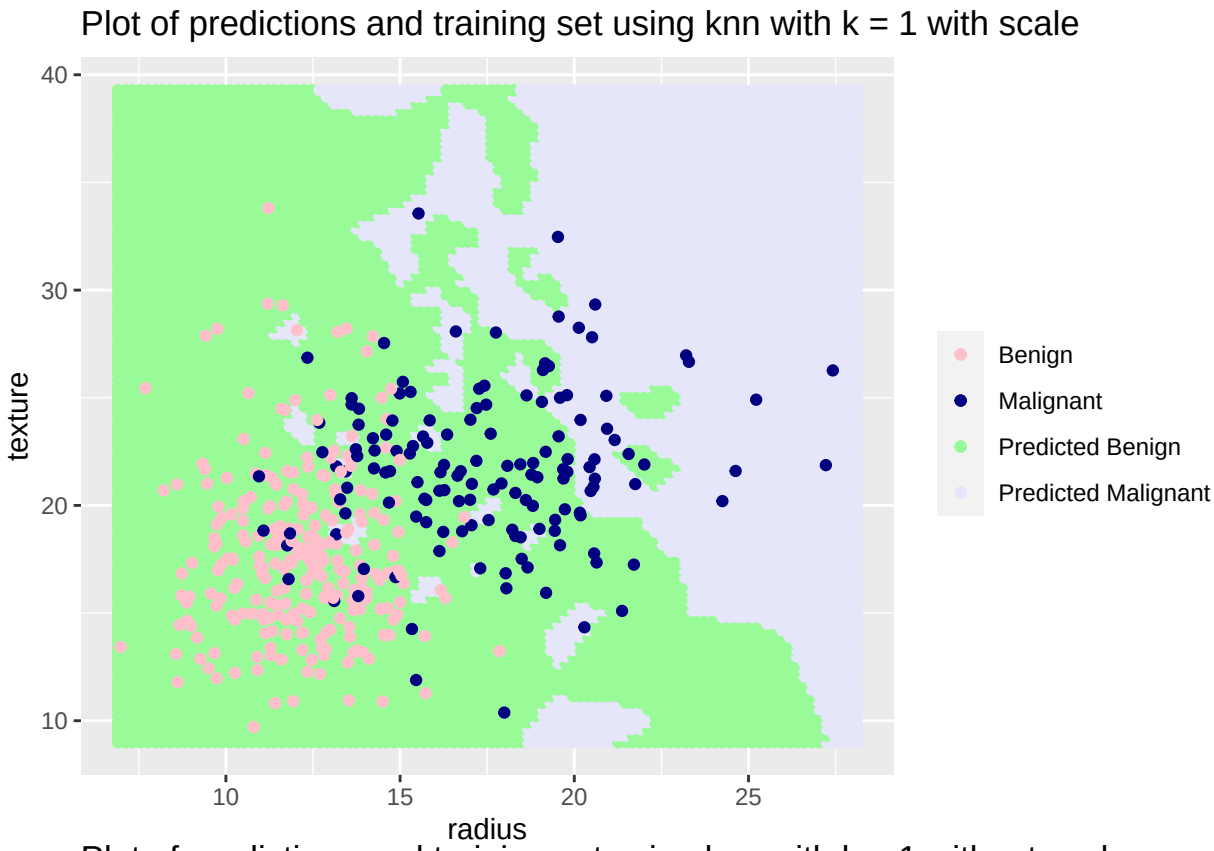
Table 19: Confusion Table for Test Set with K = 20

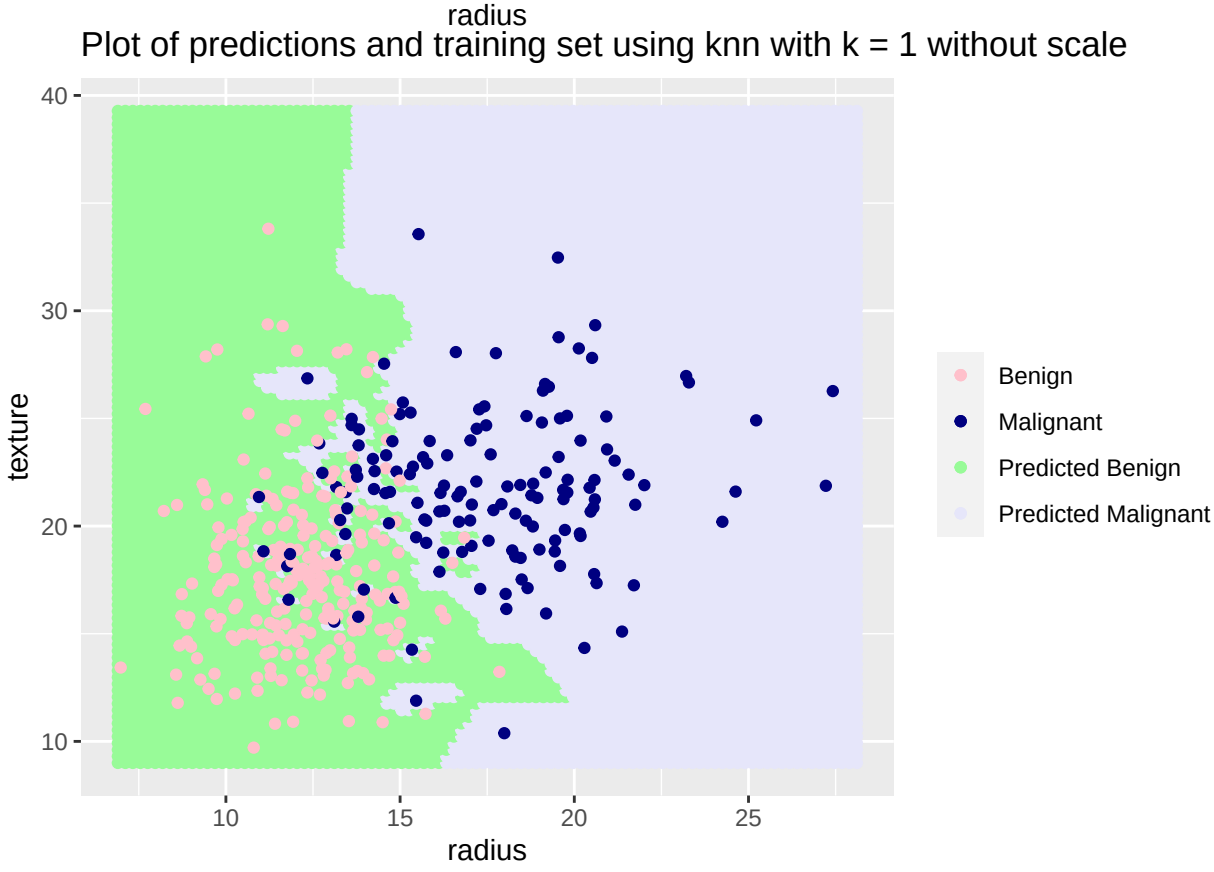
	Benign	Malignant
Benign	98	25
Malignant	2	44

Number of k	Training set accuracy	Test set accuracy
1	1.0000	0.8343195
2	0.9200	0.8402367
3	0.9175	0.8698225
4	0.9175	0.8757396
20	0.9175	0.8402367

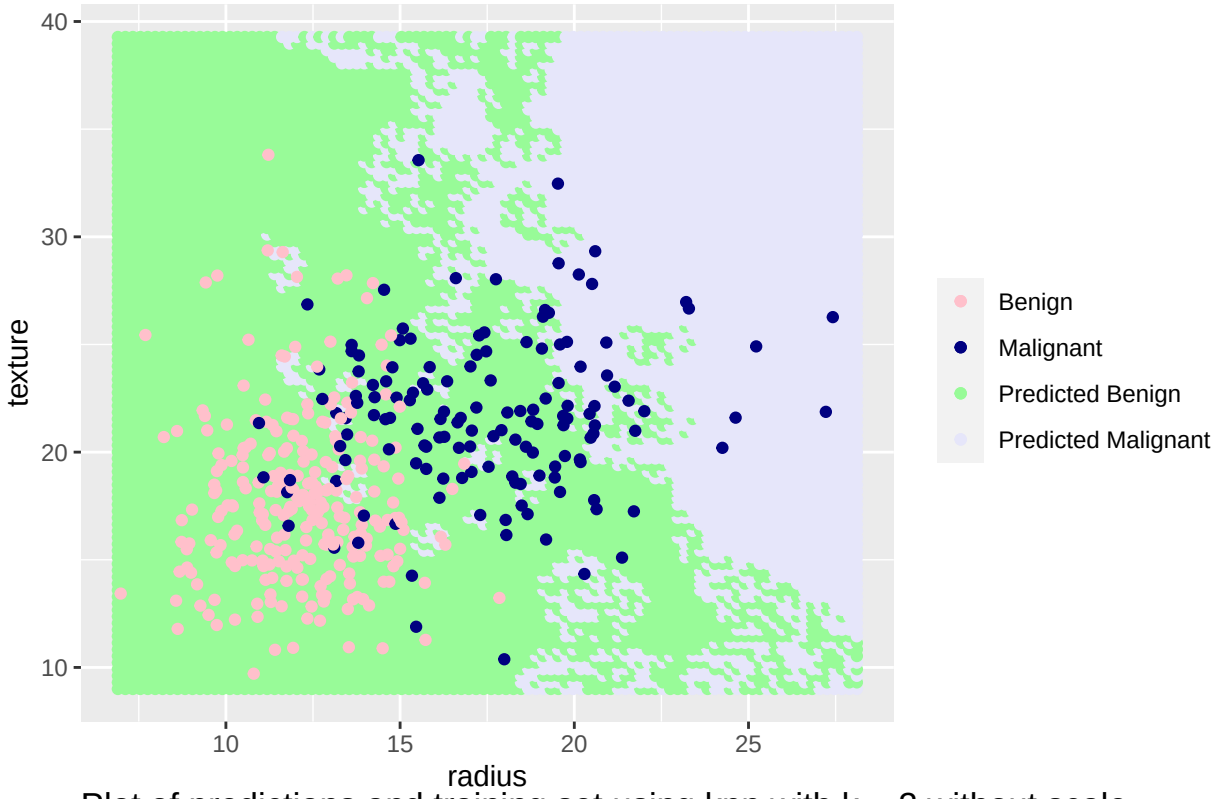
From the previous tables and accuracy predictions, we can see the accuracy is 1 when we use k = 1 in trainning set prediction. As the k goes up, the accuracy of the training set has a decreasing trend, but this trend is not observed at the test set accuracy. When k = 20, both the trainning set accuracy and test set accuracy are relatively low. When k = 4, the testing set accuracy is the highest among all k.

Part(b)

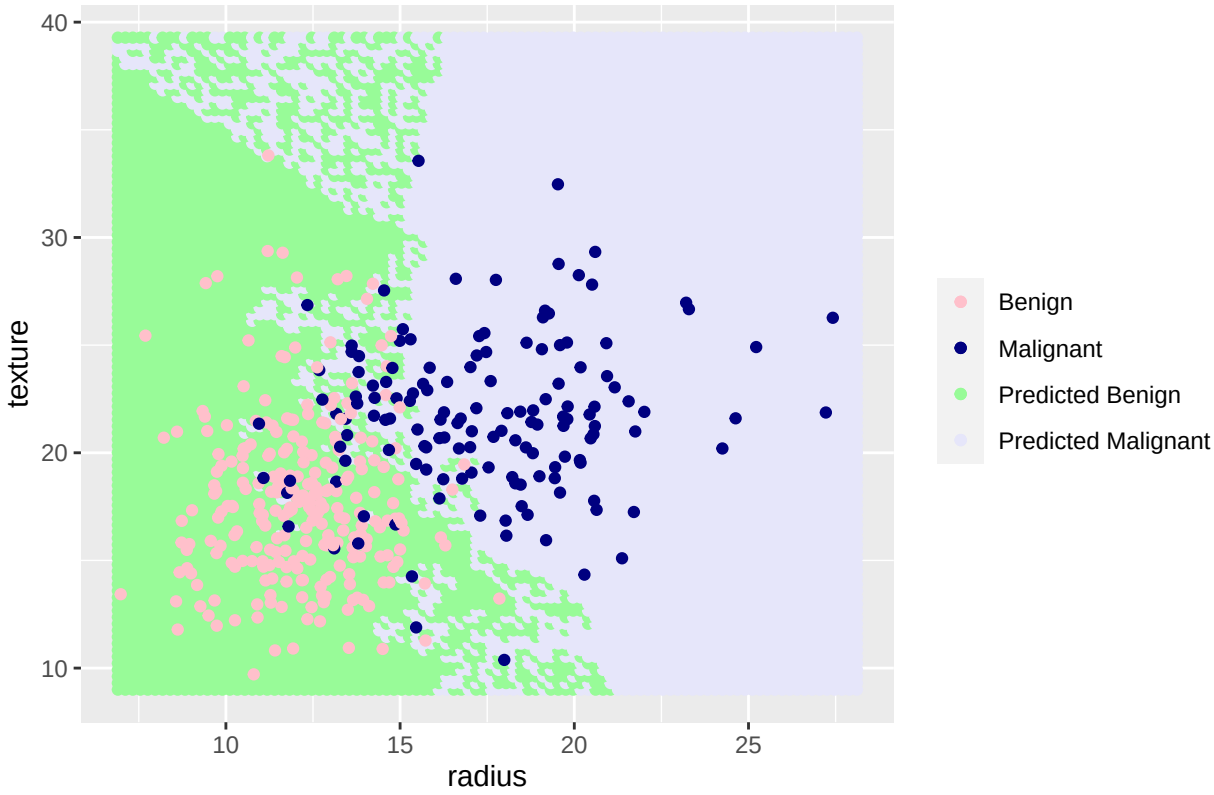

Plot of predictions and training set using knn with k = 1 with scale
texture
radius
40
30
20
10
10
15
20
25
Benign
Malignant
Predicted Benign
Predicted Malignant


Plot of predictions and training set using knn with k = 1 without scale
texture
radius
40
30
20
10
10
15
20
25
Benign
Malignant
Predicted Benign
Predicted Malignant

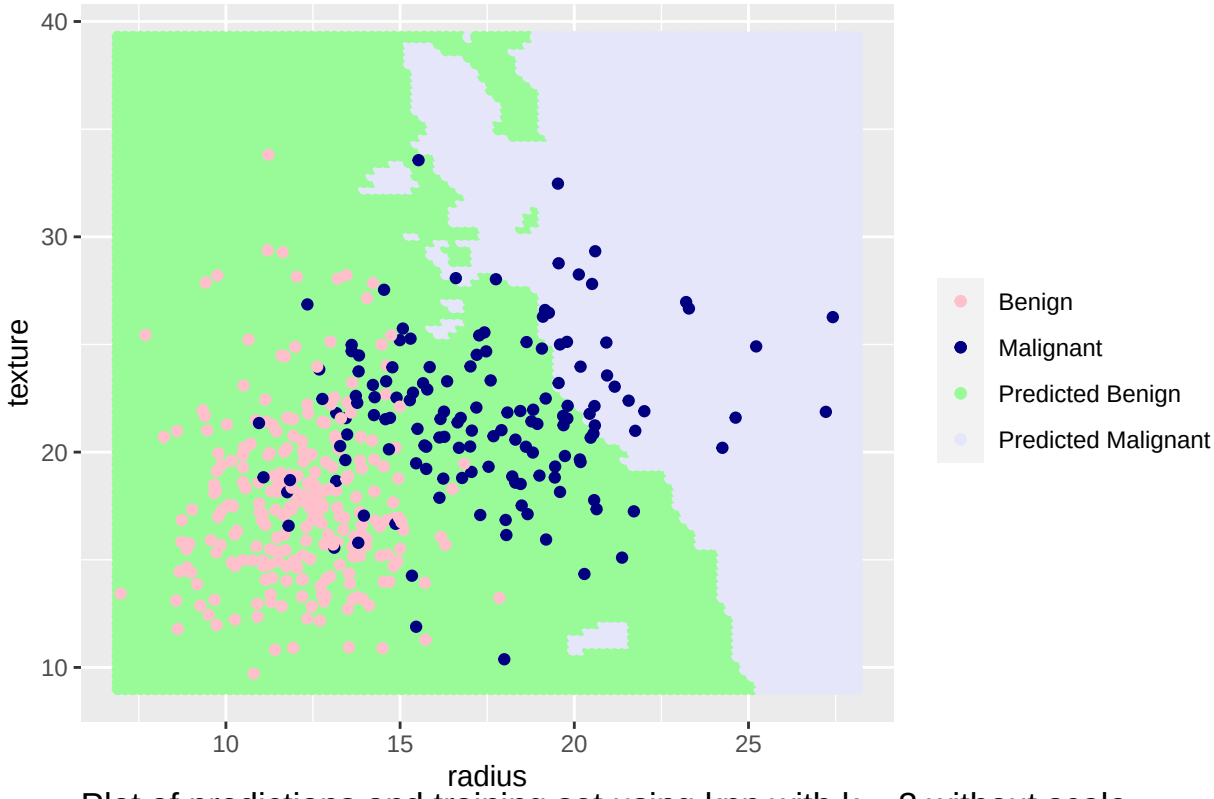
Plot of predictions and training set using knn with k = 2 with scale



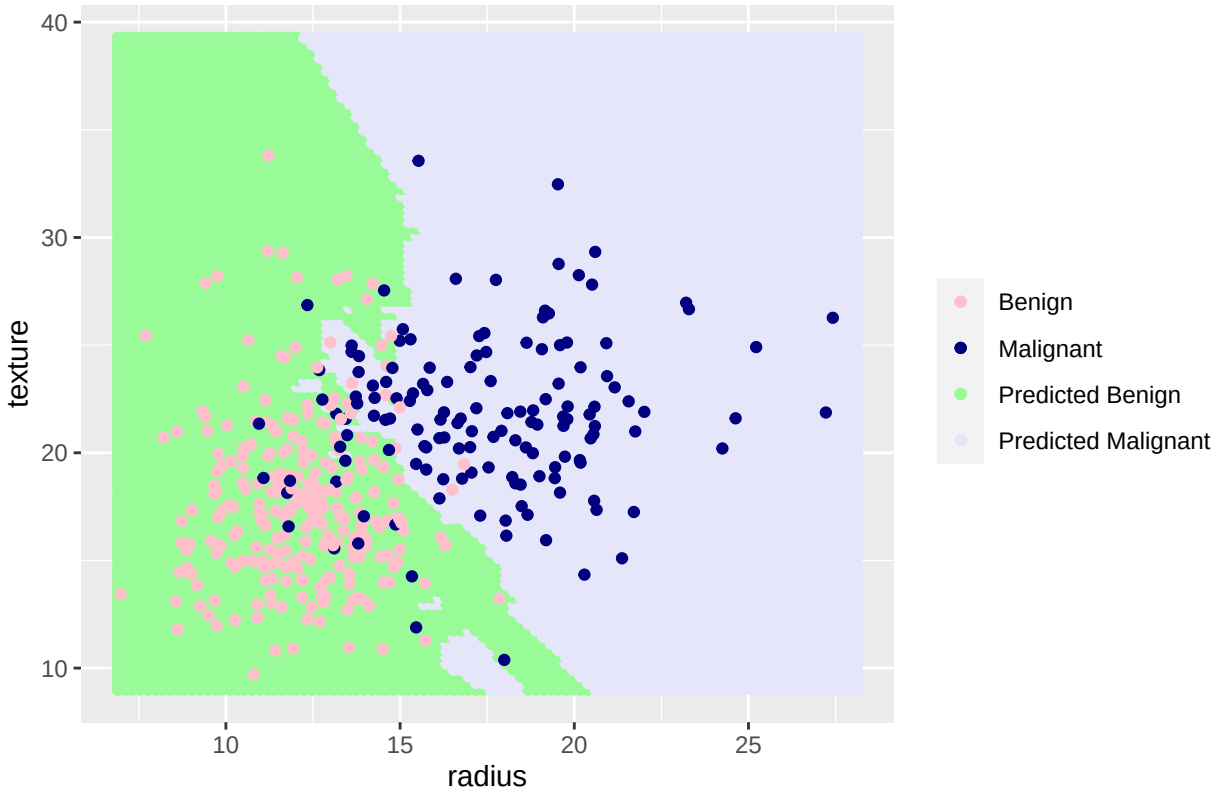
Plot of predictions and training set using knn with k = 2 without scale



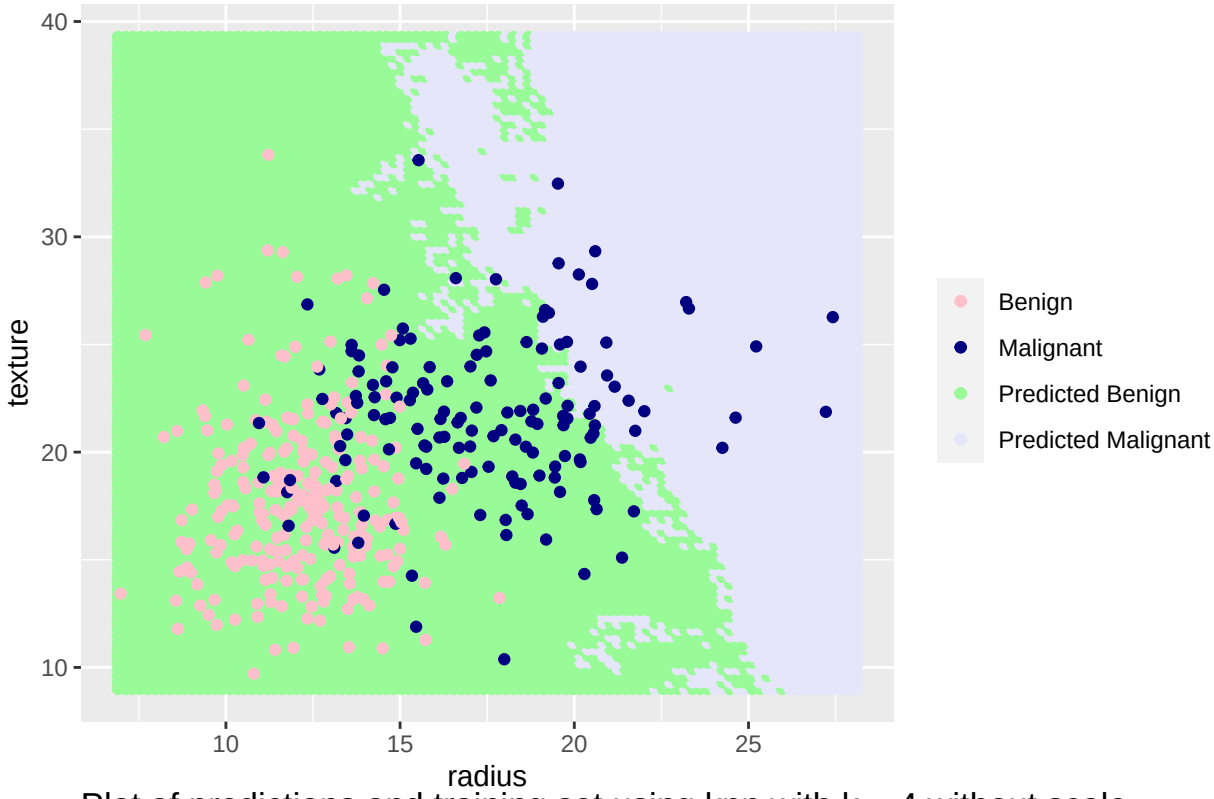
Plot of predictions and training set using knn with k = 3 with scale



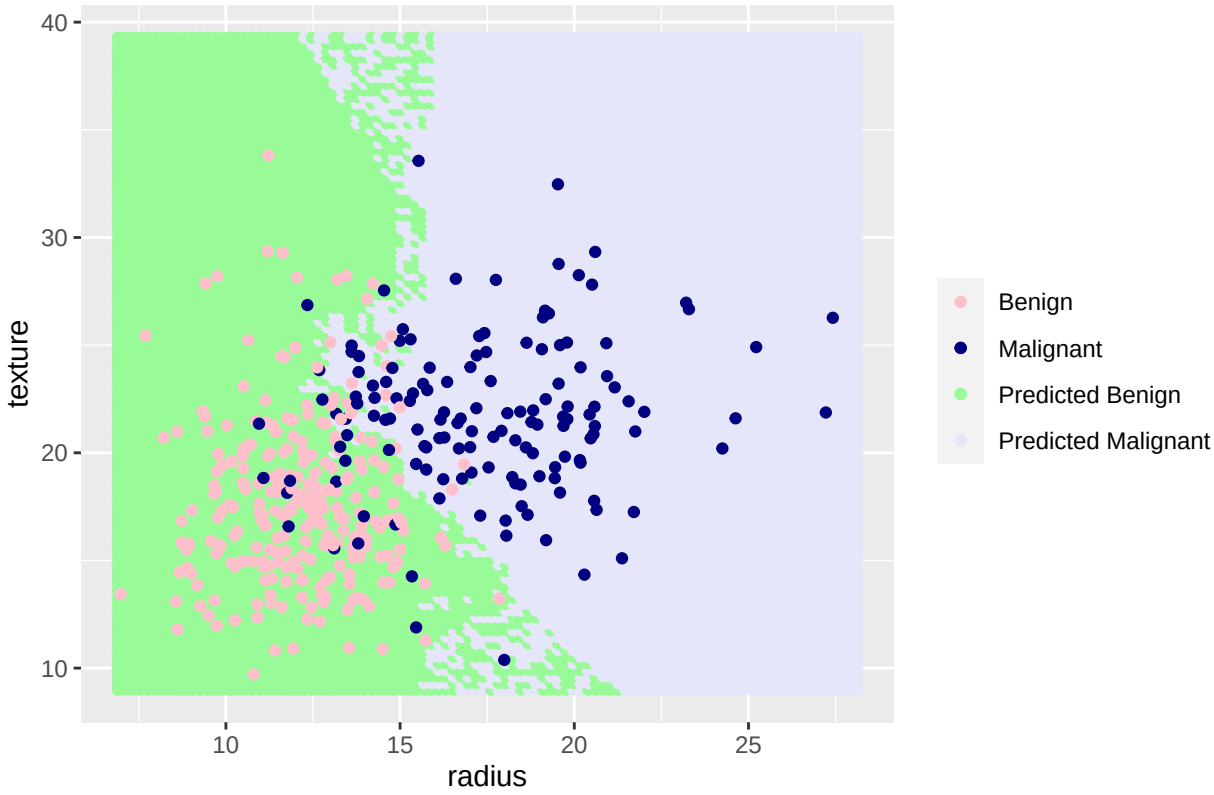
Plot of predictions and training set using knn with k = 3 without scale

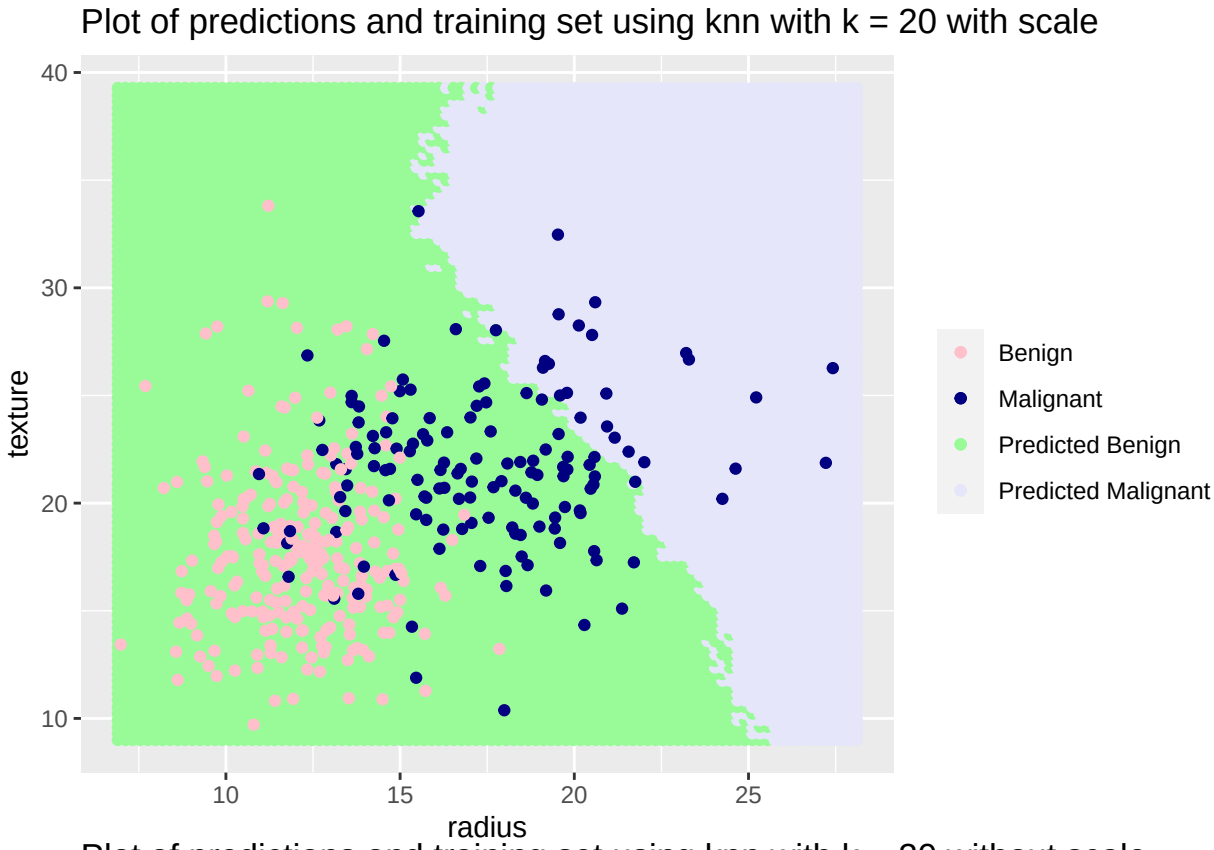


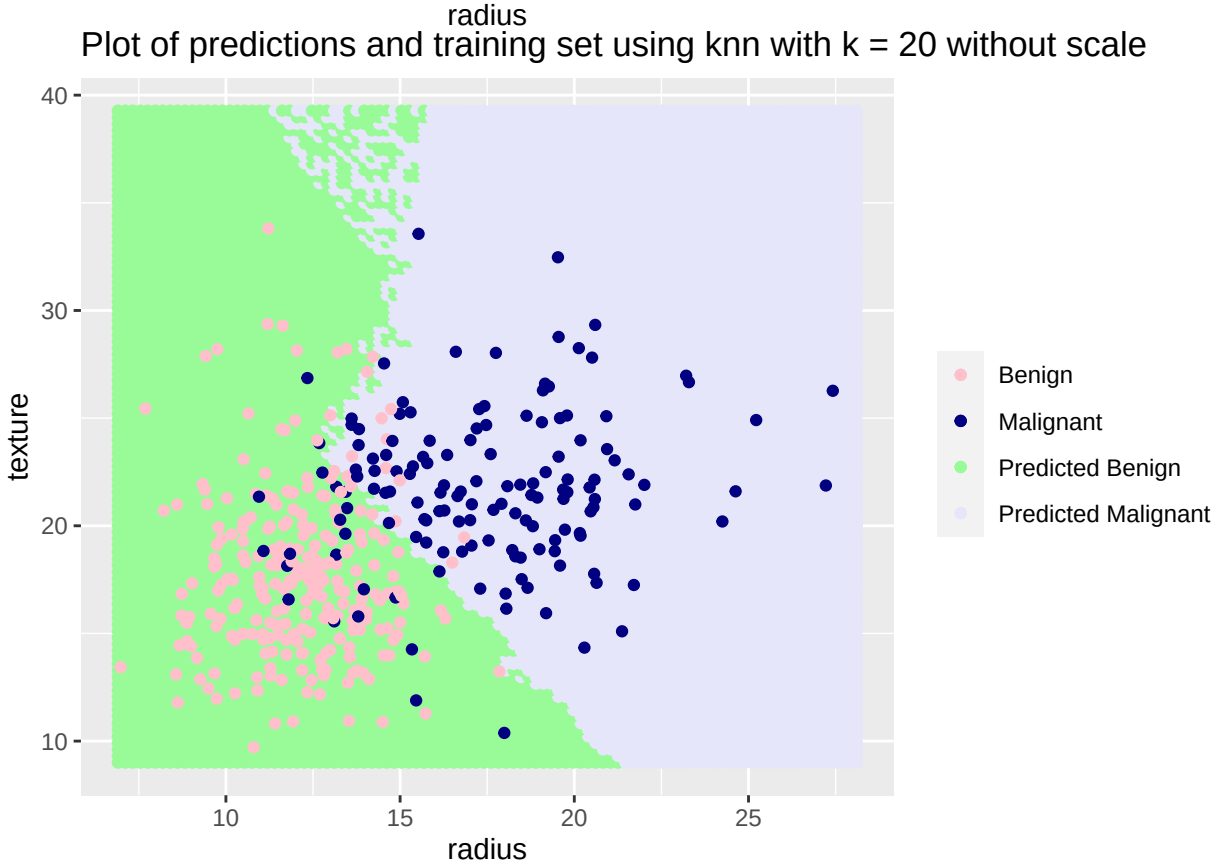
Plot of predictions and training set using knn with k = 4 with scale



Plot of predictions and training set using knn with k = 4 without scale



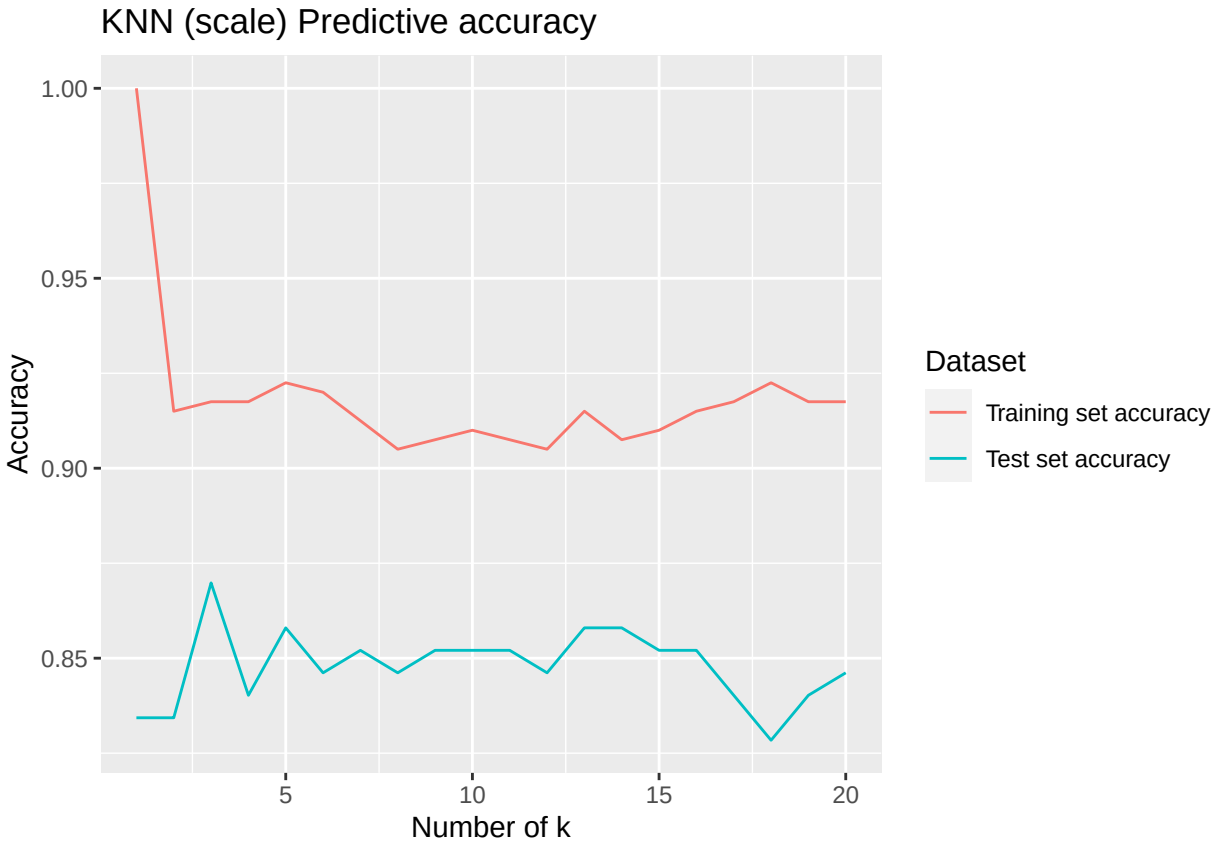


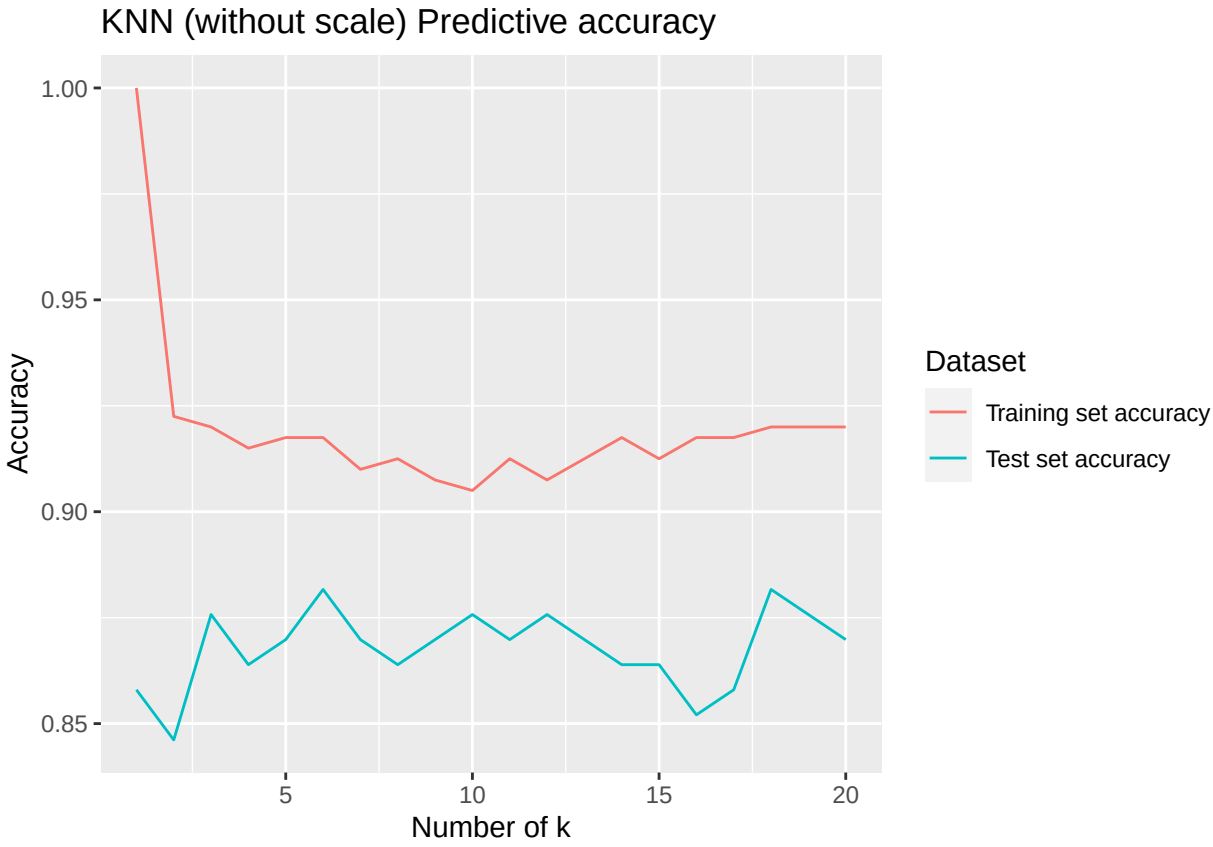


As we can see from the plots, when k is small, there is not a clearly decision boundary that can divided the

graph into two parts. The graph has benign predicted area scattered in different location. When k becomes larger and larger, the predicted area become more and more gathered together and finally when k = 20, the graph are splited into only two areas. Personally, I think we should scale both predictors. However, in this case, it seems KNN works better visually when we don't scale them.

Part(c)


KNN (scale) Predictive accuracy
1.00
0.95
0.90
0.85
Accuracy
5
10
15
20
Number of k
Dataset
Training set accuracy
Test set accuracy


KNN (without scale) Predictive accuracy
1.00
0.95
0.90
0.85
Accuracy
5
10
15
20
Number of k
Dataset
Training set accuracy
Test set accuracy

We can see the training set accuracy has a sharp down at the beginning and maintain as a relatively constant after 5. For the test set accuracy, it increase sharply at the beginning and stay constant at the larger number of k.

If we are using scale data and using kNN, I will choose k equals to 3 where test accuracy is the highest and training accuracy is also relatively good. If we are using raw data and using kNN, I will use k equals to 6 where test set accuracy is the highest and trainning accuracy is also relatively good.

```
knitr::opts_chunk$set(echo = FALSE, warning = F, message = F)
#1a
library(readr)
library(dplyr)
library(ggplot2)
library(knitr)
wdbc <- read.csv("~/Desktop/R hw/wdbc.data", header=FALSE)
colnames(wdbc)[1:4] = c("id","diagnosis","radius","texture")
wdbc$diagnosis = factor(wdbc$diagnosis, levels = c("B","M"), labels = c("Benign","Malignant"))
n = nrow(wdbc)
n_pr = ncol(wdbc) - 2
n_m = wdbc%>%group_by(diagnosis)%>%summarize(n = n())
#1b
set.seed(0)
train_id = sample(nrow(wdbc), 400)
train_df = wdbc[train_id,]
test_df  = wdbc[-train_id,]
#1c
ggplot(train_df, aes(x= radius, y = texture, color = diagnosis)) +
  geom_point()
#1d
glm.model = glm(formula = diagnosis~radius+texture, family = binomial(link = "logit"), data = train_df)
kable(round(summary(glm.model)$coefficients,3))
#1e
predict(glm.model, data.frame(texture = 12, radius = 10), type = "response")
#1f
glm.prob.train = predict(glm.model, type = "response")
glm.label.train = rep("Benign", nrow(train_df))
glm.label.train[glm.prob.train > .5] = "Malignant"
tt.glm.train = table(True = glm.label.train, Predicted = train_df$diagnosis)
kable(tt.glm.train, caption = " Confusion Matrix of Training Set")
glm.prob.test = predict(glm.model,type = "response", newdata = test_df)
glm.label.test = rep("Benign", nrow(test_df))
glm.label.test[glm.prob.test > .5] = "Malignant"
tt.glm.test = table(glm.label.test, test_df$diagnosis)
kable(tt.glm.test, caption = " Confusion Matrix of Test Set")
mean(glm.label.test == test_df$diagnosis)
mean(glm.label.train == train_df$diagnosis)
#1g
df = expand.grid(radius = seq(6.9,28.11, length.out = 100), texture = seq(9, 39.28,length.out = 100))
df = df%>%mutate(pre_prob = predict(glm.model, newdata = df, type = "response"))
df = df %>% mutate(threshold_half = ifelse(pre_prob>0.5,"Predicted Malignant", "Predicted Benign"),
                   threshold_quarte = ifelse(pre_prob>0.25,"Predicted Malignant", "Predicted Benign"),
                   threshold_third = ifelse(pre_prob>0.75,"Predicted Malignant", "Predicted Benign"))
ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_half))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle("Plot of predictions and training set using cut-off of 0.5")+
  theme(legend.title = element_blank())

ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_quarte))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
```

```
  ggtitle("Plot of predictions and training set using cut-off of 0.25")+
  theme(legend.title = element_blank())

ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_third))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle("Plot of predictions and training set using cut-off of 0.75")+
  theme(legend.title = element_blank())
#1h
n_segm = 20
TPR = replicate(n_segm, 0)
FPR = replicate(n_segm, 0)
p_th = seq(0,1,length.out = n_segm)

for (i in 1:n_segm)
{
  glm.label.test = rep("Benign", nrow(test_df))
  glm.label.test[glm.prob.test > p_th[i]] = "Malignant"

  tt.glm.test = table(glm.label.test, test_df$diagnosis)
  TPR[i] = mean(glm.label.test[test_df$diagnosis == 'Malignant'] == test_df$diagnosis[test_df$diagnosis
  FPR[i] = mean(glm.label.test[test_df$diagnosis == 'Benign'] != test_df$diagnosis[test_df$diagnosis ==
}
ggplot() + geom_path(aes(x = FPR, y = TPR))

#2a
library(MASS)
lda.model = lda(diagnosis~texture+radius, data = train_df, centre = TRUE, scale = TRUE)
lda_table = round(lda.model$means,3)
lda_table  = cbind(lda_table, lda.model$prior)
colnames(lda_table) = c("Group Means: Texture","Group Mean: Radius", "Prior Probabilities")
kable(lda_table)
#2b
lda.pred.train = predict(lda.model, train_df)
lda.pred.test = predict(lda.model, test_df)
tt.lda.train = table(Predicted = lda.pred.train$class, Truth=train_df$diagnosis)
kable(tt.lda.train, caption = " Confusion Matrix of Training Set")
tt.lda.test = table(Predicted = lda.pred.test$class, Truth = test_df$diagnosis)
kable(tt.lda.test, caption = " Confusion Matrix of Test Set")
mean(lda.pred.train$class == train_df$diagnosis)
mean(lda.pred.test$class == test_df$diagnosis)
#2c
df = df%>%mutate(pre_prob_lda = predict(lda.model, newdata = df)$posterior[,2])
df = df %>% mutate(threshold_half_lda = ifelse(pre_prob_lda>0.5,
                                               "Predicted Malignant", "Predicted Benign"),
                   threshold_quarte_lda = ifelse(pre_prob_lda>0.25,
                                                 "Predicted Malignant", "Predicted Benign"),
                   threshold_third_lda= ifelse(pre_prob_lda>0.75,
                   "Predicted Malignant", "Predicted Benign"))
ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_half_lda))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle("Plot of predictions and training set using cut-off of 0.5")+
```

```
  theme(legend.title = element_blank())

ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_quarte_lda))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle("Plot of predictions and training set using cut-off of 0.25")+
  theme(legend.title = element_blank())

ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_third_lda))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle("Plot of predictions and training set using cut-off of 0.75")+
  theme(legend.title = element_blank())

#2d
TPR = replicate(n_segm, 0)
FPR = replicate(n_segm, 0)
p_th = seq(0,1,length.out = n_segm)
for (i in 1:n_segm)
{
  lda.label.test = rep("Benign", nrow(test_df))
  lda.label.test[lda.pred.test$posterior[,2] > p_th[i]] = "Malignant"

  tt.lda.test = table(lda.label.test, test_df$diagnosis)
  TPR[i] = mean(lda.label.test[test_df$diagnosis == 'Malignant'] == test_df$diagnosis[test_df$diagnosis
  FPR[i] = mean(lda.label.test[test_df$diagnosis == 'Benign'] != test_df$diagnosis[test_df$diagnosis ==
}
ggplot() + geom_path(aes(x = FPR, y = TPR))
#2e
library(pracma)
auc = abs(trapz(FPR, TPR))
auc
#3a
qda.model = qda(diagnosis~texture+radius, data = train_df,centre = TRUE)
qda_table = round(qda.model$means,3)
qda_table  = cbind(qda_table, qda.model$prior)
colnames(qda_table) = c("Group Means: Texture","Group Mean: Radius", "Prior Probabilities")
kable(qda_table)
#3b
qda.pred.train = predict(qda.model, train_df)
qda.pred.test = predict(qda.model, test_df)
tt.qda.train = table(Predicted = qda.pred.train$class, Truth=train_df$diagnosis)
kable(tt.qda.train, caption = " Confusion Matrix of Training Set")
tt.qda.test = table(Predicted = qda.pred.test$class, Truth = test_df$diagnosis)
kable(tt.qda.test, caption = " Confusion Matrix of Test Set")
mean(qda.pred.train$class == train_df$diagnosis)
mean(qda.pred.test$class == test_df$diagnosis)
#3c
df = df%>%mutate(pre_prob_qda = predict(qda.model, newdata = df)$posterior[,2])
df = df %>% mutate(threshold_half_qda = ifelse(pre_prob_qda>0.5,
                                               "Predicted Malignant", "Predicted Benign"),
                   threshold_quarte_qda = ifelse(pre_prob_qda>0.25,
                                                 "Predicted Malignant", "Predicted Benign"),
```

```
                  threshold_third_qda= ifelse(pre_prob_qda>0.75,
                  "Predicted Malignant", "Predicted Benign"))
ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_half_qda))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle("Plot of predictions and training set using cut-off of 0.5")+
  theme(legend.title = element_blank())

ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_quarte_qda))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle("Plot of predictions and training set using cut-off of 0.25")+
  theme(legend.title = element_blank())

ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = threshold_third_qda))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle("Plot of predictions and training set using cut-off of 0.75")+
  theme(legend.title = element_blank())

#3d
TPR = replicate(n_segm, 0)
FPR = replicate(n_segm, 0)
p_th = seq(0,1,length.out = n_segm)
for (i in 1:n_segm)
{
  qda.label.test = rep("Benign", nrow(test_df))
  qda.label.test[qda.pred.test$posterior[,2] > p_th[i]] = "Malignant"

  tt.qda.test = table(qda.label.test, test_df$diagnosis)
  TPR[i] = mean(qda.label.test[test_df$diagnosis == 'Malignant'] == test_df$diagnosis[test_df$diagnosis
  FPR[i] = mean(qda.label.test[test_df$diagnosis == 'Benign'] != test_df$diagnosis[test_df$diagnosis ==
}
ggplot() + geom_path(aes(x = FPR, y = TPR))
#3e
auc = abs(trapz(FPR, TPR))
auc
library(class)
train_df = train_df[,c(2:4)]
test_df = test_df[,c(2:4)]
train.num = train_df[,!sapply(train_df, is.factor)]
train.num = scale(train_df[,c(2:3)])
test.num = test_df[,!sapply(test_df, is.factor)]
test.num = scale(test.num)

k_num = c(1,2,3,4,20)
accuracy = matrix(nrow = 5, ncol = 3)

for(i in 1:5){
  knn.train.pred <- knn(train = train.num,
                        test  = train.num,
                        cl    = train_df$diagnosis, k = k_num[i])
  knn.test.pred <- knn(train = train.num,
```

```
                      test  = test.num,
                      cl    = train_df$diagnosis, k = k_num[i])
  tt.knn.train = table(knn.train.pred, train_df$diagnosis)


  print(kable(tt.knn.train, caption = paste0("Confusion Table for Training Set with K = ",
                                             k_num[i], sep="")))
  tt.knn.test = table(knn.test.pred, test_df$diagnosis)

  print(kable(tt.knn.test, caption = paste0("Confusion Table for Test Set with K = ",
                                            k_num[i], sep="")))


  accuracy[i,1] = k_num[i]
  accuracy[i,2] = mean(knn.train.pred == train_df$diagnosis)
  accuracy[i,3] = mean(knn.test.pred == test_df$diagnosis)

}
colnames(accuracy) = c("Number of k","Training set accuracy", "Test set accuracy")
kable(accuracy)
train.num = train_df[,!sapply(train_df, is.factor)]
test.num = test_df[,!sapply(test_df, is.factor)]
for(i in 1:5){
  knn.df.pred <- knn(train = scale(train.num),
                      test  = scale(df[,c(1:2)]),
                      cl    = train_df$diagnosis, k = k_num[i])
  knn.df.pred_ns = knn(train = train.num,
                      test  = df[,c(1:2)],
                      cl    = train_df$diagnosis, k = k_num[i])
  knn.df.pred = ifelse(knn.df.pred=="Benign","Predicted Benign","Predicted Malignant")
  knn.df.pred_ns = ifelse(knn.df.pred_ns=="Benign","Predicted Benign","Predicted Malignant")
  print(ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = knn.df.pred))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle(paste0("Plot of predictions and training set using knn with k = ", k_num[i]," with scale",
                 sep = ""))+
  theme(legend.title = element_blank()))
print(ggplot()+geom_point(data = df, aes(x = radius, y = texture, color = knn.df.pred_ns))+
  geom_point(data = train_df, aes(x = radius, y = texture, color = diagnosis))+
  scale_color_manual(values=c("pink", "navy", "palegreen","lavender"))+
  ggtitle(paste0("Plot of predictions and training set using knn with k = ",
                 k_num[i]," without scale",
                 sep = ""))+
  theme(legend.title = element_blank()))
}
library(reshape2)
accuracy = matrix(nrow = 20, ncol = 3)
accuracy_ns = matrix(nrow = 20, ncol = 3)
for(i in 1:20){
  knn.train.pred <- knn(train = train.num,
                      test  = train.num,
                      cl    = train_df$diagnosis, k = i)
  knn.test.pred <- knn(train = train.num,
```

```
                      test  = test.num,
                      cl    = train_df$diagnosis, k = i)
  accuracy_ns[i,1] = i
  accuracy_ns[i,2] = mean(knn.train.pred == train_df$diagnosis)
  accuracy_ns[i,3] = mean(knn.test.pred == test_df$diagnosis)
  knn.train.pred <- knn(train = scale(train.num),
                      test  = scale(train.num),
                      cl    = train_df$diagnosis, k =i)
  knn.test.pred <- knn(train = scale(train.num),
                      test  = scale(test.num),
                      cl    = train_df$diagnosis, k = i)
  accuracy[i,1] = i
  accuracy[i,2] = mean(knn.train.pred == train_df$diagnosis)
  accuracy[i,3] = mean(knn.test.pred == test_df$diagnosis)

}
colnames(accuracy) = c("Number of k","Training set accuracy", "Test set accuracy")
colnames(accuracy_ns) = c("Number of k","Training set accuracy", "Test set accuracy")
accuracy = as.data.frame(accuracy)
accuracy_ns = as.data.frame(accuracy_ns)
accuracy = melt(accuracy, id = "Number of k")
accuracy_ns = melt(accuracy_ns,id = "Number of k")
colnames(accuracy) = c("Number of k","Dataset", "Accuracy")
colnames(accuracy_ns) = c("Number of k","Dataset", "Accuracy")
ggplot(data = accuracy, mapping = aes(x =`Number of k`, y =Accuracy, color = Dataset))+
  geom_line()+
  ggtitle("KNN (scale) Predictive accuracy")
ggplot( data = accuracy_ns, aes(x =`Number of k`, y =Accuracy, color = Dataset))+
  geom_line()+
  ggtitle("KNN (without scale) Predictive accuracy ")
```