BIOST 546 - HW2

John Schoof

2/8/2021

Question 1: Logistic Regression

Question 1A. N = 569 observations. P = 32 (30 predictor variables). There are 357 begign diagnoses and 212 malignant diagnoses.

```
dat <- fread("wdbc.data")
#glimpse(dat)
table(dat$V2)
```

```
##
##   B   M
## 357 212
```

```
which(is.na(dat)) ## no missing data
```

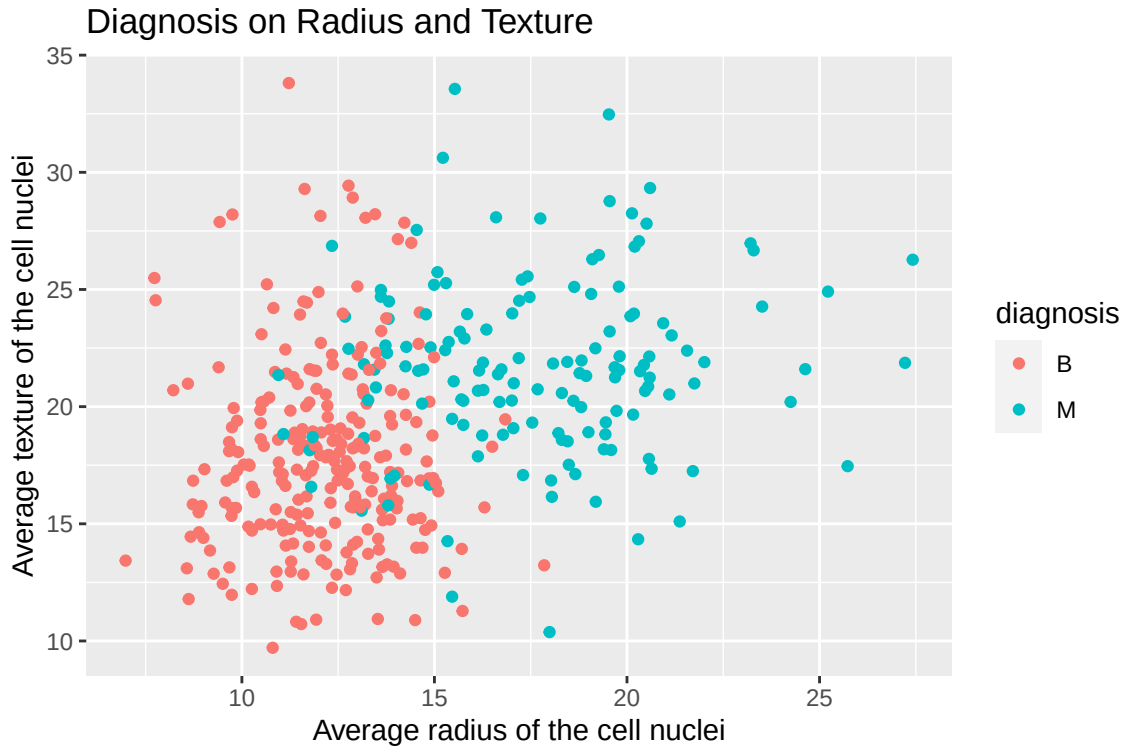
```
## integer(0)
```

```
## keep and rename variables
  dat1 <- dat %>% dplyr::select(V1, V2, V3, V4) %>%
                  rename(id = V1,
                         diagnosis = V2,
                         radius = V3,
                         texture = V4)
  dat1 <- dat1 %>% mutate(diagnosis = as.factor(diagnosis))
  #glimpse(dat1)

## split sample
  set.seed(1)
  train_id = sample(nrow(dat1), 400)
  train = dat1[train_id,]
  test  = dat1[-train_id,]
```

Question 1B.

Question 1C. The below plot suggests that there tend to be clusters of benign and malignant tumors with regard to tumor texture and radius. Malignant tumors appear to have cell nuclei with a larger radius and higher standard deviation of gray-scale values as a measure of texture. However, there is overlap, meaning there are not two completely distinct groups of red and blur dots, which will result in some prediction error.

```
ggplot(data = train) +
  geom_point(aes(x=radius, y= texture, color = diagnosis)) +
  labs(title = "Diagnosis on Radius and Texture",
       x = "Average radius of the cell nuclei",
       y = "Average texture of the cell nuclei")
```



Question 1D. I fit the below logistic regression model. The log odds of a tumor being malignant are 1.01 times greater comparing tumors with a one unit difference in radius on average holding texture constant. The log odds of a tumor being malignant are 0.21 times comparing tumors with a one unit difference in radius on average holding texture constant.

$$\text{Logit}(Diagnosis = Malignant | Radius, Texture = x_1, x_2) = \beta_0 + \beta_1 * Radius + \beta_2 * Texture$$

```
## logistic regession model
mod1d <- glm(diagnosis~radius+texture, data = train, family = "binomial")
summary.1d <- summary(mod1d)


## table of output
table.1d <- data.frame(c("Intercept", "Radius", "Texture"))
table.1d$estimate <- summary.1d$coefficients[1:3]
table.1d$st.error <- summary.1d$coefficients[4:6]
```

```
table.1d$z.stat <- summary.1d$coefficients[7:9]
table.1d$p.value <- summary.1d$coefficients[10:12]

kable(table.1d, format = "pipe", col.names = c("", "Estimate", "Standard Error", "Z-Statistic", "P-valu
```

Table 1: Logistic Regression Model Output

	Estimate	Standard Error	Z-Statistic	P-value
Intercept	-19.201777	2.0217987	-9.497374	0e+00
Radius	1.005809	0.1149819	8.747537	0e+00
Texture	0.215260	0.0435310	4.944986	8e-07

Question 1E.

$$\text{Logit}(Diagnosis = Malignant|Radius, Texture = x_1, x_2) = \beta_0 + \beta_1 * Radius + \beta_2 * Texture$$

```
## Using manual calculation
  coef <- as.vector(mod1d$coefficients)
  log.odds <- coef[1] + coef[2] * 10 + coef[3] * 12
  odds <- exp(log.odds)
  prob <- odds / (1+odds)
## Using predict fctn
  pred <- predict(mod1d, data.frame(radius = 10, texture = 12))
  pred.prob <- (exp(pred))/(1+(exp(pred)))
```

Log odds of diagnosis being malignant = -19.2017774 + 1.0058087 * 10 + 0.21526 * 12 = -6.5605698

Odds of diagnosis being malignant = exp(-6.5605698) = 0.0014151

Probability of diagnosis being malignant = 0.0014151 / (1 + 0.0014151) = 0.0014131

Using the predict function, the estimated probability of a diagnosis being malignant is also 0.0014131.

```
  log.odds <- 0.7
  odds <- exp(log.odds)
  prob <- odds / (1+odds)
```

Question 1F. If the log odds = 0.7, then **the expected probability that a diagnosis is malignant is 0.6681878.**

Question 1G. The prediction accuracy on the training set using Bayes' classifier was 89.7%. This means that my model correctly predicted 89.7% of outcomes correctly for every individual in the training set. The prediction error on the training set using Bayes' classifier was 87%. This means that my model correctly predicted 87% of outcomes correctly for every individual in the training set. It is expected that the model performs better on the training set.

```
## Training set performances
  glm.prob.train <- predict(mod1d, type = "response") # my model's predictions

  ## Binary classification based on probability threshold
  glm.label.train <- rep("B", nrow(train))
  glm.label.train[glm.prob.train > .5] <- "M"

  ## confusion matrix (true positive rate and false positive rate)
  (tt.glm.train <- table(glm.label.train, train$diagnosis))
```

```
##
## glm.label.train   B   M
##               B 243  28
##               M  13 116
```

```
  ## misclassification error (proportion of all 569 that I predicted correctly)
  accuracy.train <- mean(glm.label.train == train$diagnosis)

# Test set performances
  glm.prob.test <- predict(mod1d, type = "response", newdata = test)

  ## Binary classification based on probability threshold
  glm.label.test <- rep("B", nrow(test))
  glm.label.test[glm.prob.test > .5] <- "M"
  ## confusion matrix (true positive rate and false positive rate)
  (tt.glm.test <- table(glm.label.test, test$diagnosis))
```

```
##
## glm.label.test  B  M
##              B 95 16
##              M  6 52
```

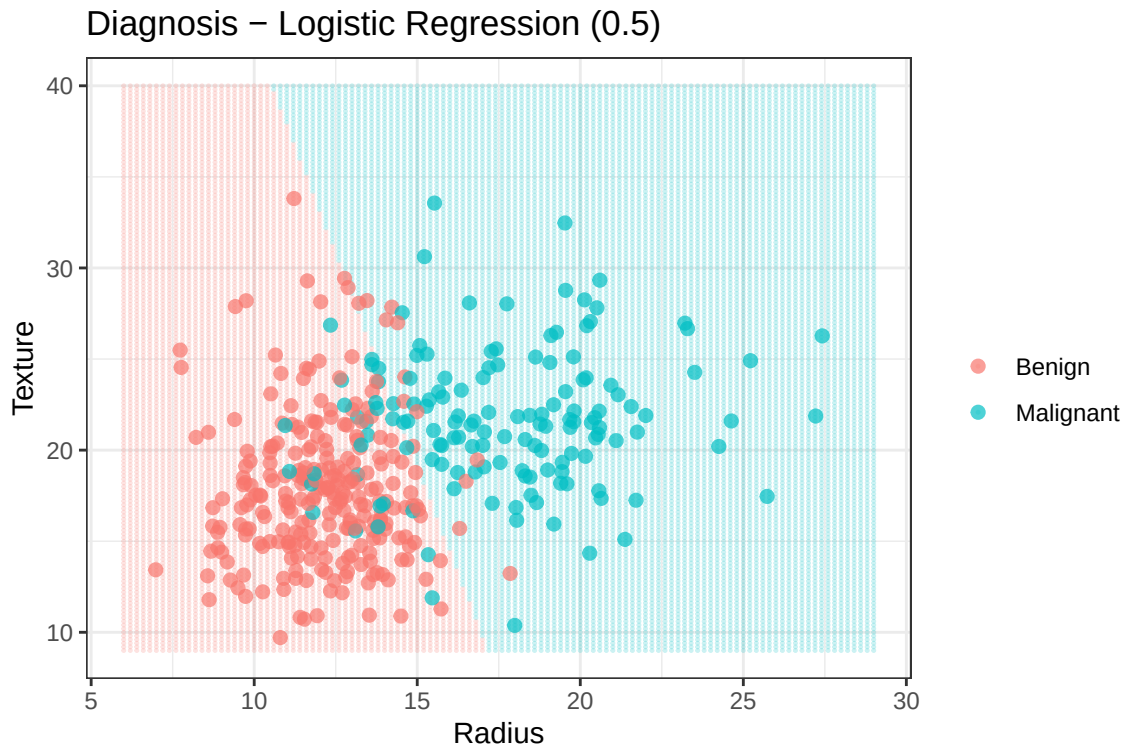
```
  ## misclassification error (proportion of test set that I predicted correctly)
  (glm.accuracy <- mean(glm.label.test == test$diagnosis))
```

```
## [1] 0.8698225
```

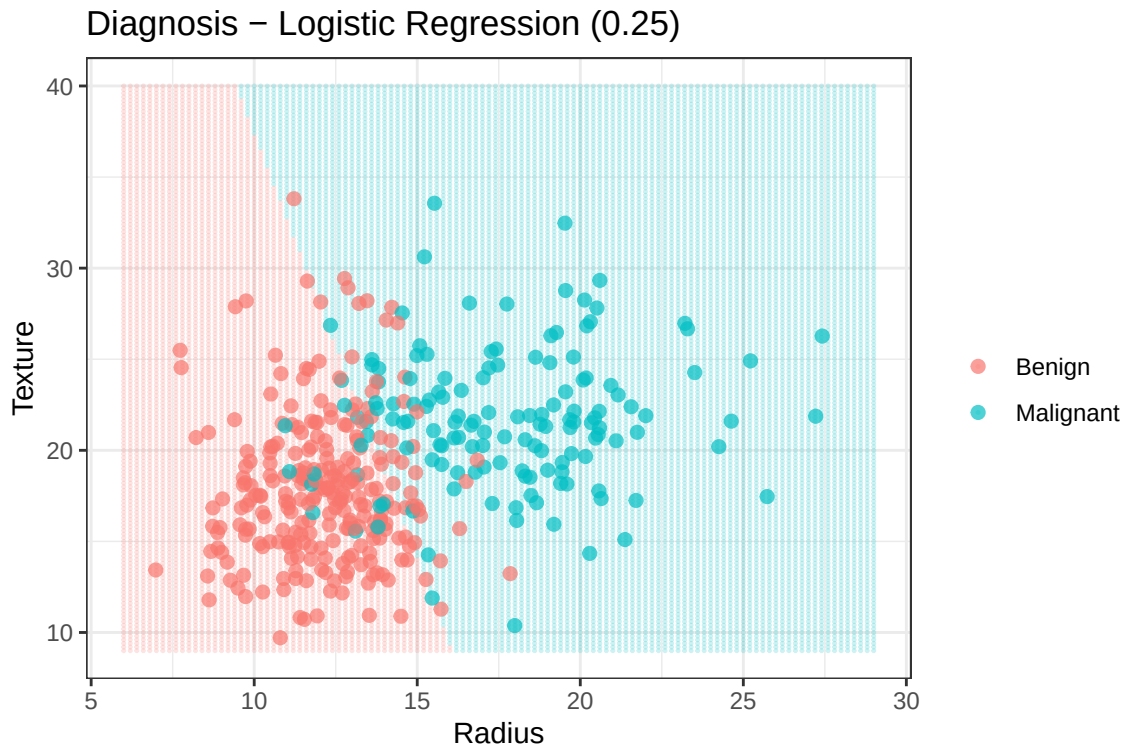
Question 1H. Below are plots of the decision boundaries for probability cut offs of 0.5, 0.25, and 0.75. The decision boundary for a cutoff of 0.25 is shifted to the left of the original 0.5 decision boundary. This means that there are more false positives, or more benign tumors are predicted to be malignant than using the 0.5 cutoff. Alternatively, the decision boundary for a cutoff of 0.75 is shifted to the right of the original 0.5 decision boundary. This means that there are more false negatives, or more malignant tumors are predicted to be benign than using the 0.5 cutoff.

```
##  cut off of 0.5
  dense.dat <- expand.grid(texture = seq(9, 40, by = 0.2), radius = seq(6, 29, by = 0.2))
dense.dat <- as.data.frame(dense.dat)

  pred1h <- predict(mod1d, type = 'response', newdata = dense.dat)

  glm.label.test <- rep("B", nrow(dense.dat))
```

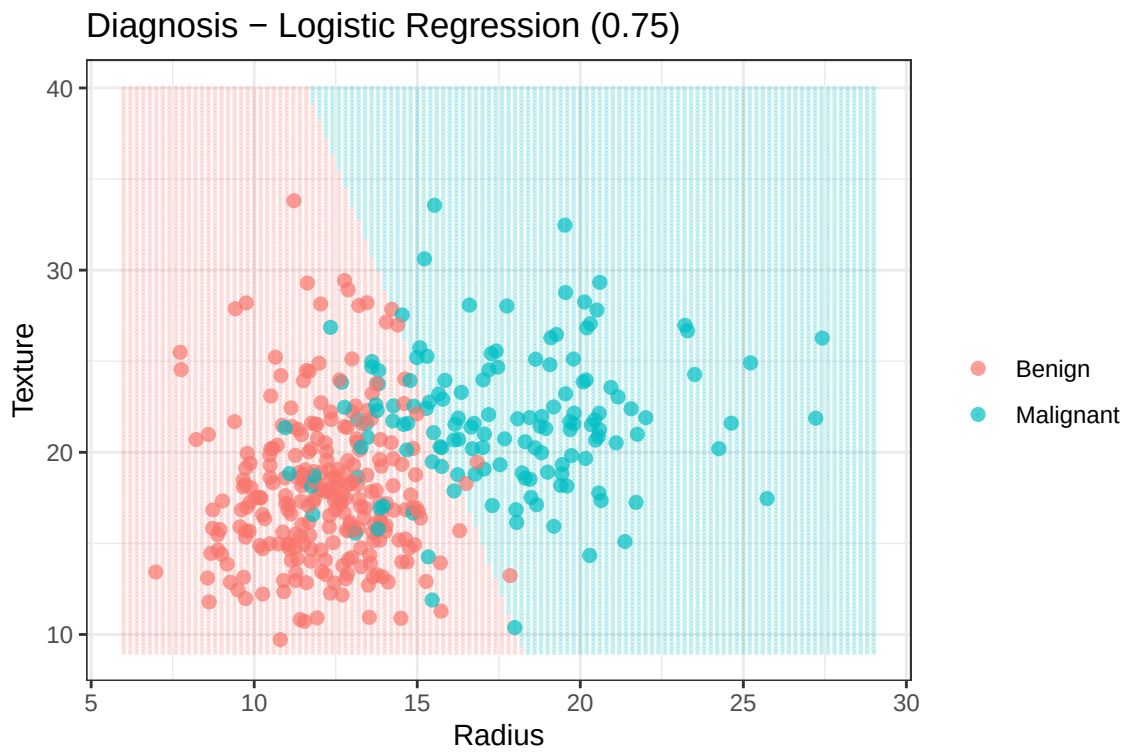
```
  glm.label.test[pred1h > .5] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = glm.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Logistic Regression (0.5)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```




```
##  cut off of 0.25
  pred1h <- predict(mod1d, type = 'response', newdata = dense.dat)

  glm.label.test <- rep("B", nrow(dense.dat))
  glm.label.test[pred1h > .25] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = glm.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Logistic Regression (0.25)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```

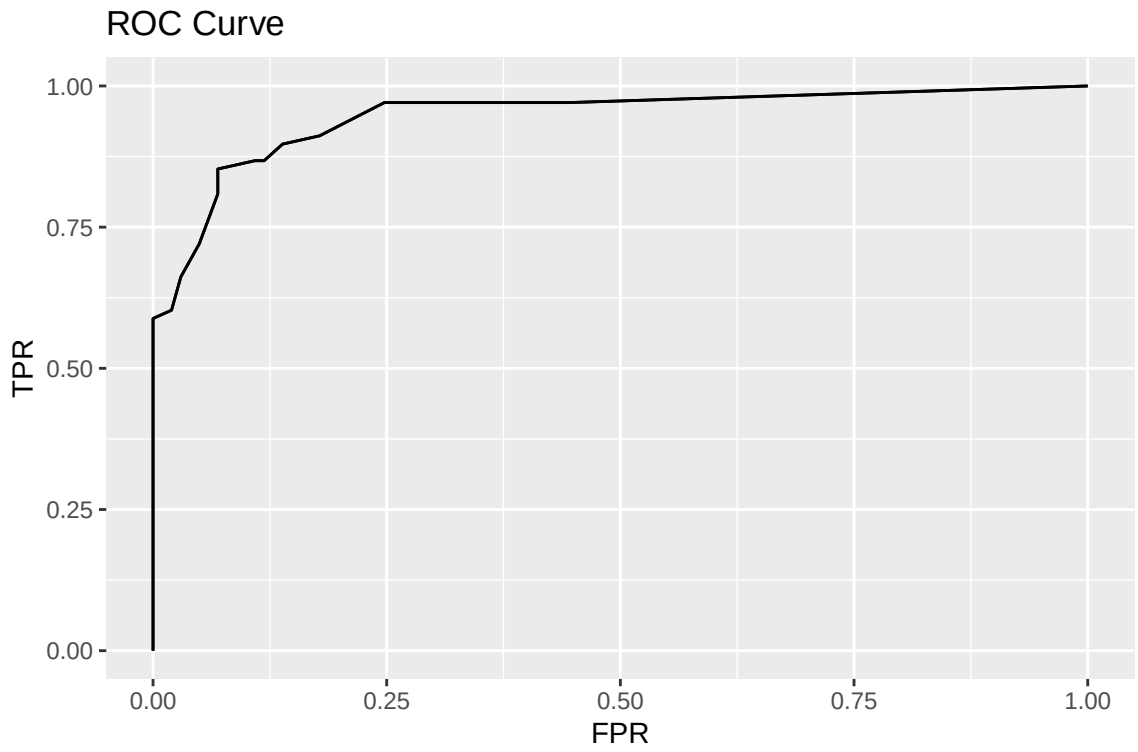



```
##  cut off of 0.75
  pred1h <- predict(mod1d, type = 'response', newdata = dense.dat)

  glm.label.test <- rep("B", nrow(dense.dat))
  glm.label.test[pred1h > .75] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = glm.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Logistic Regression (0.75)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```



Question 1I. Below is the plot for the ROC curve for the logistic regression model.

```
## roc curve
  n_segm = 21
  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("B", nrow(test))
    glm.label.test[glm.prob.test > p_th[i]] = "M"

    tt.glm.test = table(glm.label.test, test$diagnosis)
    TPR[i] = mean(glm.label.test[test$diagnosis == "M"] == test$diagnosis[test$diagnosis == "M"])
    FPR[i] = mean(glm.label.test[test$diagnosis == "B"] != test$diagnosis[test$diagnosis == "B"])
  }

# plot(x = FPR, y = TPR, 'l')
  ggplot() +
    geom_path(aes(x = FPR, y = TPR)) + geom_path(aes(x = FPR, y = TPR))+
    labs(title = "ROC Curve")
```



Question 1J. The estimated area under the curve is 0.947.

```
## area under the curve
  auc.vec <- c()
    for (i in 1:20){
      auc.vec[i] <- ((FPR[i]-FPR[(i+1)]) * ((TPR[i]+TPR[(i+1)])/2))
    }
  (auc <- sum(auc.vec))
```

```
## [1] 0.9465638
```

Question 2: Linear Discriminant Analysis

Question 2A. Below is a table of output from the LDA model. Prior probabilities represent the probability that an individual's tumor is benign or malignant, regardless of outcome. The group means are the mean radius and texture measurements among only individuals with benign tumors and separately among only individuals with malignant tumors. These values are related to each other in Bayes Theorem. More specifically, The prior probabilities are simply the probability of Y and group means are the means of the density function of X given Y that we assume to be normally distributed. These are the components of the numerator in Bayes Theorem.

```
## LDA model
  lda.model <- lda(diagnosis ~ radius + texture, data = train, center = TRUE, scale = TRUE)
  lda.model
```

```
## Call:
## lda(diagnosis ~ radius + texture, data = train, center = TRUE,
```

```
##     scale = TRUE)
##
## Prior probabilities of groups:
##    B    M
## 0.64 0.36
##
## Group means:
##     radius  texture
## B 12.12635 17.83480
## M 17.64986 21.65347
##
## Coefficients of linear discriminants:
##                LD1
## radius  0.37635664
## texture 0.09082027
```

```
## table of prior probs (pi) and group means
  table.2a <- data.frame(c("Prior Probabilities", "Group Mean: Radius", "Group Mean: Texture"))
  table.2a$benign <- c(0.64, 12.1, 17.8)
  table.2a$malignant <- c(0.36, 17.6, 21.7)

  kable(table.2a, format = "pipe", col.names = c("", "Benign", "Malignant"), caption = "LDA Model Outpu
```

Table 2: LDA Model Output

	Benign	Malignant
Prior Probabilities	0.64	0.36
Group Mean: Radius	12.10	17.60
Group Mean: Texture	17.80	21.70

Question 2B. Below are the confusion matrices for the training and testing sets, respectively. The prediction accuracy for the training set is 89.7%. The prediction accuracy for the test set is 85.2%.

```
##
  lda.pred.train <- predict(lda.model, train)
  lda.pred.test <- predict(lda.model, test)

  tt.lda.train <- table(lda.pred.train$class, train$diagnosis)
  kable(tt.lda.train, caption = "Confusion Matrix - Training Set")
```

Table 3: Confusion Matrix - Training Set

	B	M
B	248	32
M	8	112

```
  mean(lda.pred.train$class == train$diagnosis)
```

```
## [1] 0.9
```

```
  tt.lda.test <- table(lda.pred.test$class, test$diagnosis)
  kable(tt.lda.test, caption = "Confusion Matrix - Test Set")
```

Table 4: Confusion Matrix - Test Set

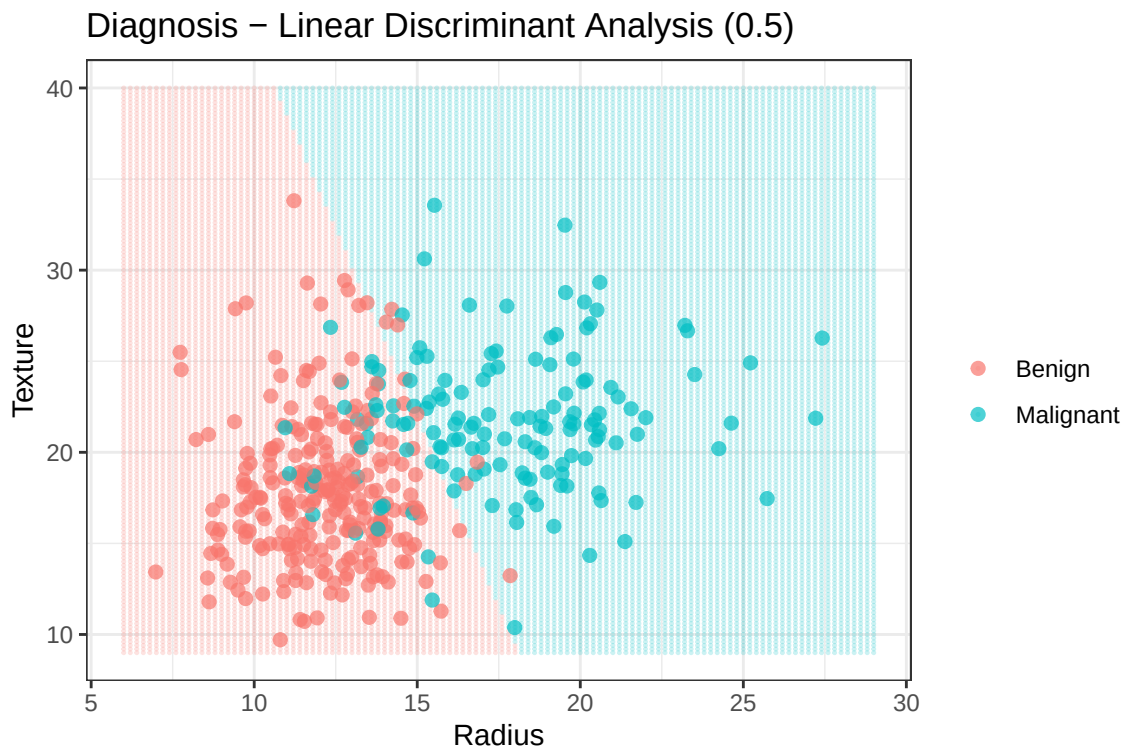
	B	M
B	98	23
M	3	45

```
  (lda.accuracy <- mean(lda.pred.test$class == test$diagnosis))
```

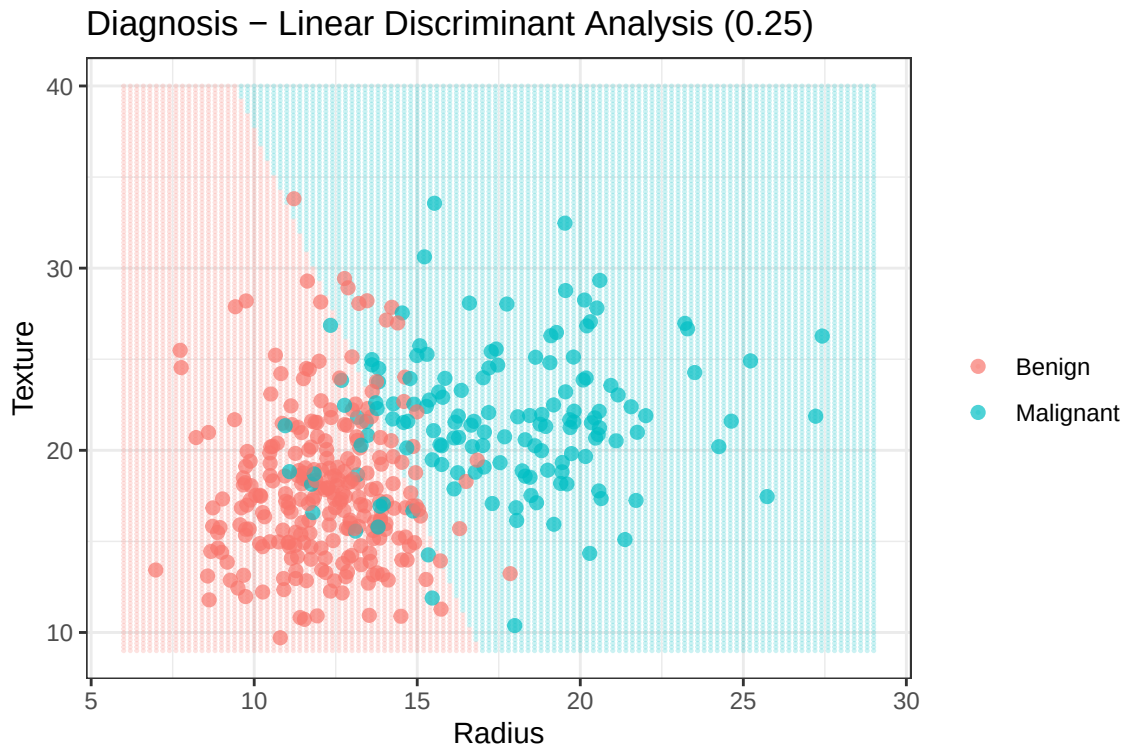
```
## [1] 0.8461538
```

Question 2C. Below are plots of the LDA decision boundaries for probability cut offs of 0.5, 0.25, and 0.75. The decision boundary for a cutoff of 0.25 is shifted to the left of the original 0.5 decision boundary. This means that there are more false positives, or more benign tumors are predicted to be malignant than using the 0.5 cutoff. Alternatively, the decision boundary for a cutoff of 0.75 is shifted to the right of the original 0.5 decision boundary. This means that there are more false negatives, or more malignant tumors are predicted to be benign than using the 0.5 cutoff.

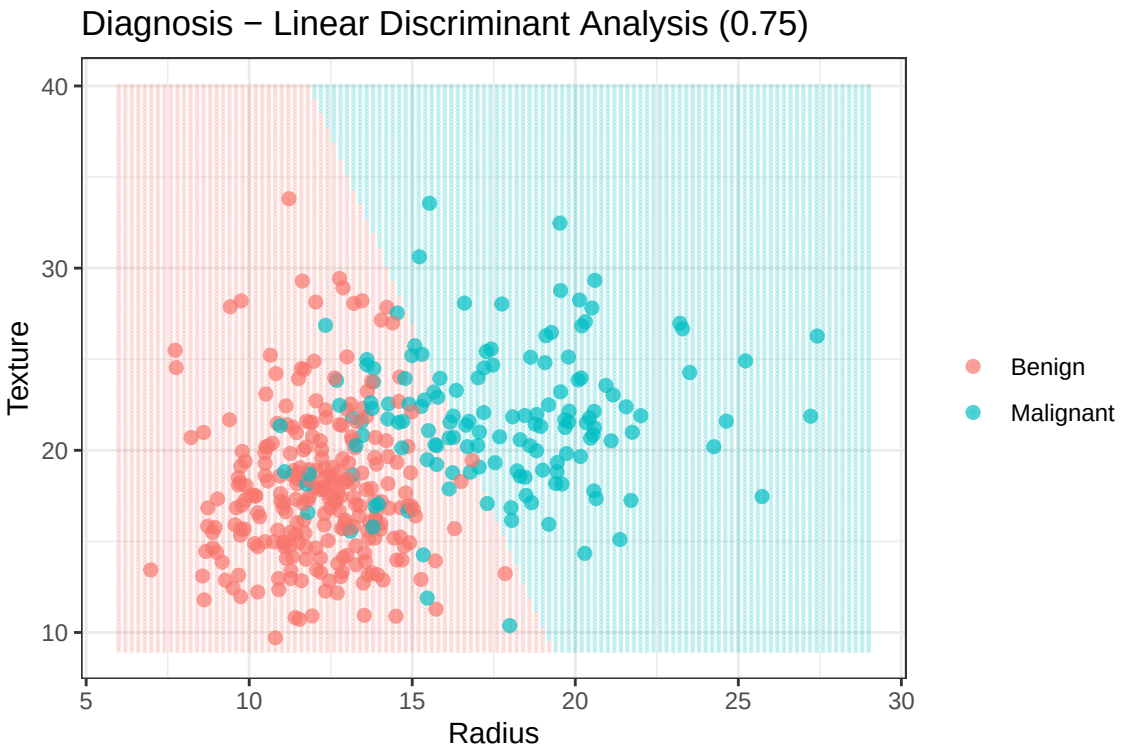
```
##  cut off of 0.5
  pred2c <- predict(lda.model, type = "response", newdata = dense.dat)$posterior[, 2]

  lda.label.test <- rep("B", nrow(dense.dat))
  lda.label.test[(pred2c) > .5] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = lda.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Linear Discriminant Analysis (0.5)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```




```
##  cut off of 0.25
  pred2c <- predict(lda.model, type = "response", newdata = dense.dat)$posterior[, 2]

  lda.label.test <- rep("B", nrow(dense.dat))
  lda.label.test[(pred2c) > .25] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = lda.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Linear Discriminant Analysis (0.25)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```

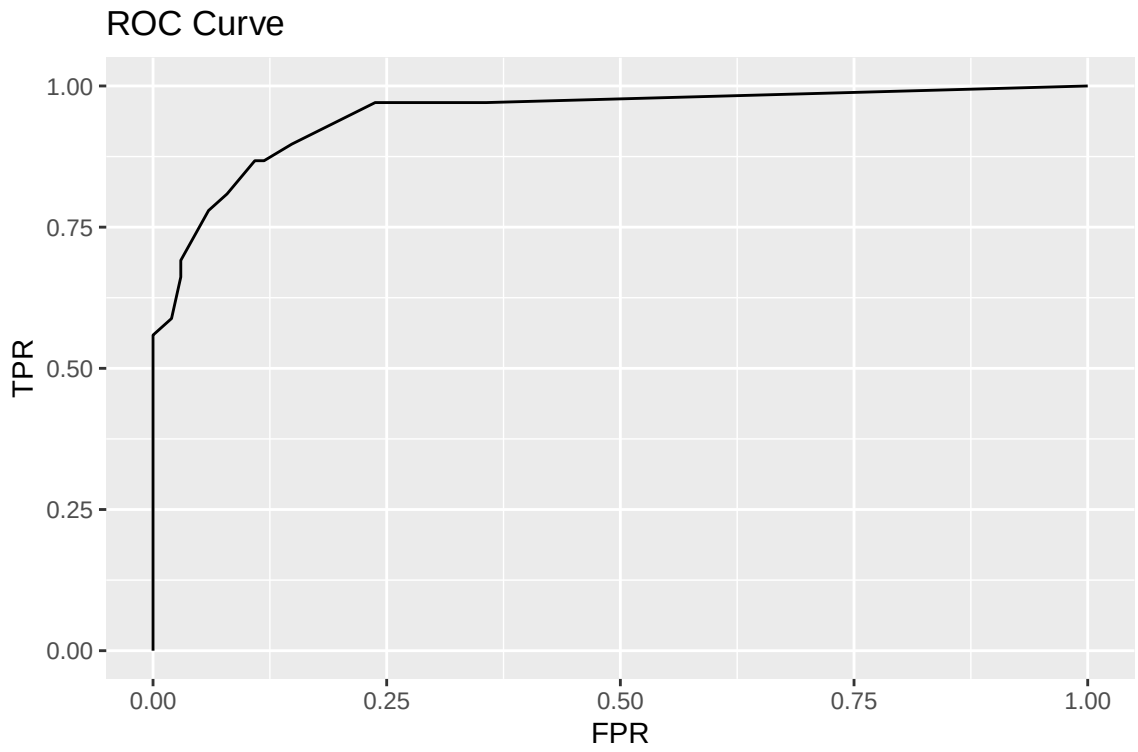



```
##  cut off of 0.75
  pred2c <- predict(lda.model, type = "response", newdata = dense.dat)$posterior[, 2]

  lda.label.test <- rep("B", nrow(dense.dat))
  lda.label.test[(pred2c) > .75] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = lda.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Linear Discriminant Analysis (0.75)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```



Question 2D. Below is the ROC curve for the LDA model.

```
## roc curve
  n_segm = 21
  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("B", nrow(test))
    lda.label.test[lda.pred.test$posterior[,2] > p_th[i]] = "M"

    TPR[i] = mean(lda.label.test[test$diagnosis == "M"] == test$diagnosis[test$diagnosis == "M"])
    FPR[i] = mean(lda.label.test[test$diagnosis == "B"] != test$diagnosis[test$diagnosis == "B"])
  }

# plot(x = FPR, y = TPR, 'l')
  ggplot() +
    geom_path(aes(x = FPR, y = TPR)) +
    labs(title = "ROC Curve")
```



Question 2E. The area under the curve is 0.947 for the LDA model.

```
## area under the curve
  auc.vec <- c()
    for (i in 1:20){
      auc.vec[i] <- ((FPR[i]-FPR[(i+1)]) * ((TPR[i]+TPR[(i+1)])/2))
    }

  (auc <- sum(auc.vec))
```

```
## [1] 0.9474374
```

Question 3: Quadratic Discriminant Analysis

```
## QDA model
  qda.model <- qda(diagnosis ~ radius + texture, data = train, center = TRUE, scale = TRUE)
  qda.model
```

Question 3A.

```
## Call:
## qda(diagnosis ~ radius + texture, data = train, center = TRUE,
##     scale = TRUE)
##
```

```
## Prior probabilities of groups:
##    B    M
## 0.64 0.36
##
## Group means:
##     radius  texture
## B 12.12635 17.83480
## M 17.64986 21.65347
```

```
## table of prior probs (pi) and group means
    table.3a <- data.frame(c("Prior Probabilities", "Group Mean: Radius", "Group Mean: Texture"))
    table.3a$benign <- c(0.64, 12.1, 17.8)
    table.3a$malignant <- c(0.36, 17.6, 21.6)

    kable(table.3a, format = "pipe", col.names = c("", "Benign", "Malignant"), caption = "LDA Model Outp
```

Table 5: LDA Model Output

	Benign	Malignant
Prior Probabilities	0.64	0.36
Group Mean: Radius	12.10	17.60
Group Mean: Texture	17.80	21.60

Question 3B. Below are the confusion matrices for the training and testing sets, respectively. The prediction accuracy for the training set is 89.5%. The prediction accuracy for the test set is 85.6%.

```
##
  qda.pred.train <- predict(qda.model, train)
  qda.pred.test <- predict(qda.model, test)

  tt.qda.train <- table(qda.pred.train$class, train$diagnosis)
  kable(tt.qda.train, caption = "Confusion Matrix - Training Set")
```

Table 6: Confusion Matrix - Training Set

	B	M
B	246	31
M	10	113

```
  mean(qda.pred.train$class == train$diagnosis)
```

```
## [1] 0.8975
```

```
  tt.qda.test <- table(qda.pred.test$class, test$diagnosis)
  kable(tt.qda.test, caption = "Confusion Matrix - Test Set")
```

Table 7: Confusion Matrix - Test Set

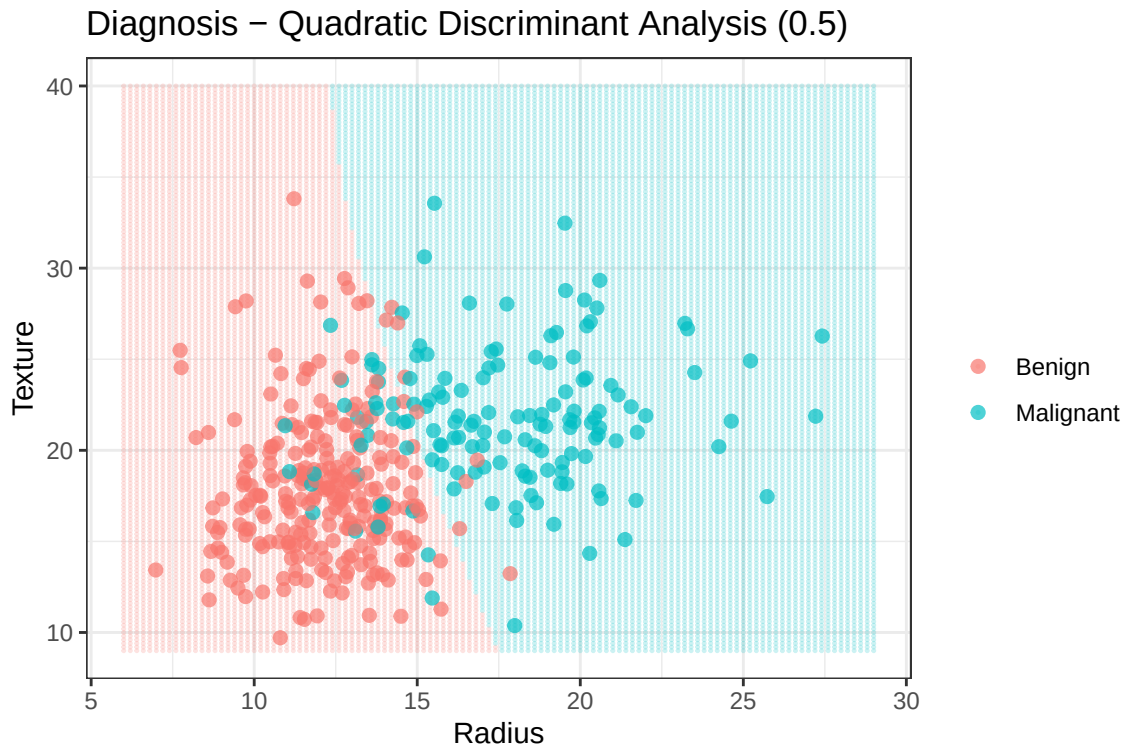
	B	M
B	97	20
M	4	48

```
  (qda.accuracy <- mean(qda.pred.test$class == test$diagnosis))
```

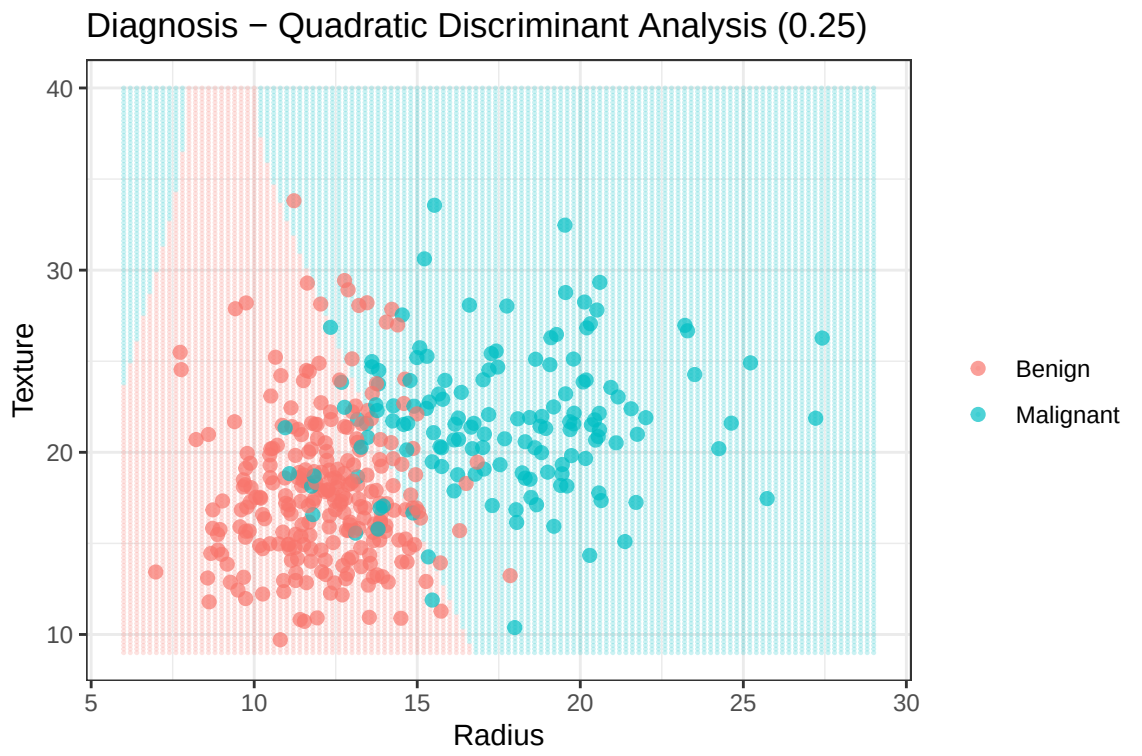
```
## [1] 0.8579882
```

Question 3C. Below are plots of the QDA decision boundaries for probability cut offs of 0.5, 0.25, and 0.75. The main difference between the LDA plots and QDA plots is that the decision boundaries for QDA are curved. The decision boundary for a cutoff of 0.25 is shifted to the left of the original 0.5 decision boundary. This means that there are more false positives, or more benign tumors are predicted to be malignant than using the 0.5 cutoff. There is now a blue section is upper left corner of the 0.25 plot. If a new data point were observed with those levels of texture and radius then it would be predicted to be malignant, unlike with the GLM and LDA models. Alternatively, the decision boundary for a cutoff of 0.75 is shifted to the right of the original 0.5 decision boundary. This means that there are more false negatives, or more malignant tumors are predicted to be benign than using the 0.5 cutoff.

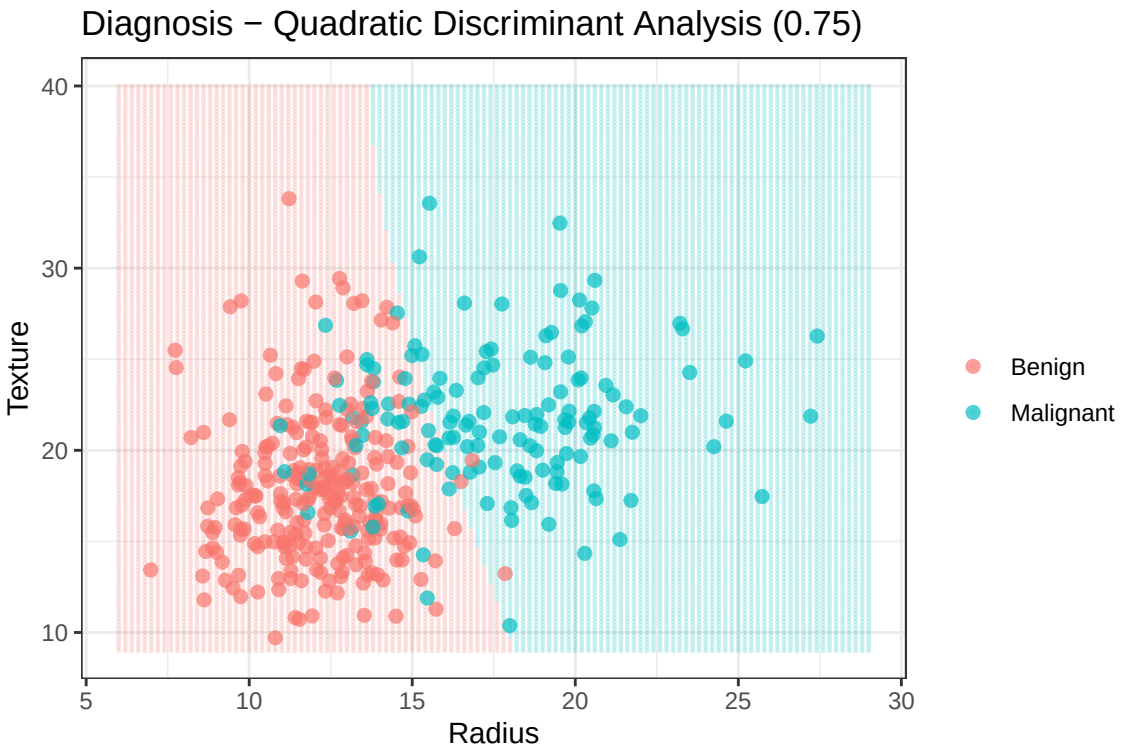
```
##  cut off of 0.5
  pred3c <- predict(qda.model, type = "response", newdata = dense.dat)$posterior[, 2]

  qda.label.test <- rep("B", nrow(dense.dat))
  qda.label.test[pred3c > .5] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = qda.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Quadratic Discriminant Analysis (0.5)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```




```
##  cut off of 0.25
  pred3c <- predict(qda.model, type = "response", newdata = dense.dat)$posterior[, 2]

  qda.label.test <- rep("B", nrow(dense.dat))
  qda.label.test[pred3c > .25] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = qda.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Quadratic Discriminant Analysis (0.25)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```

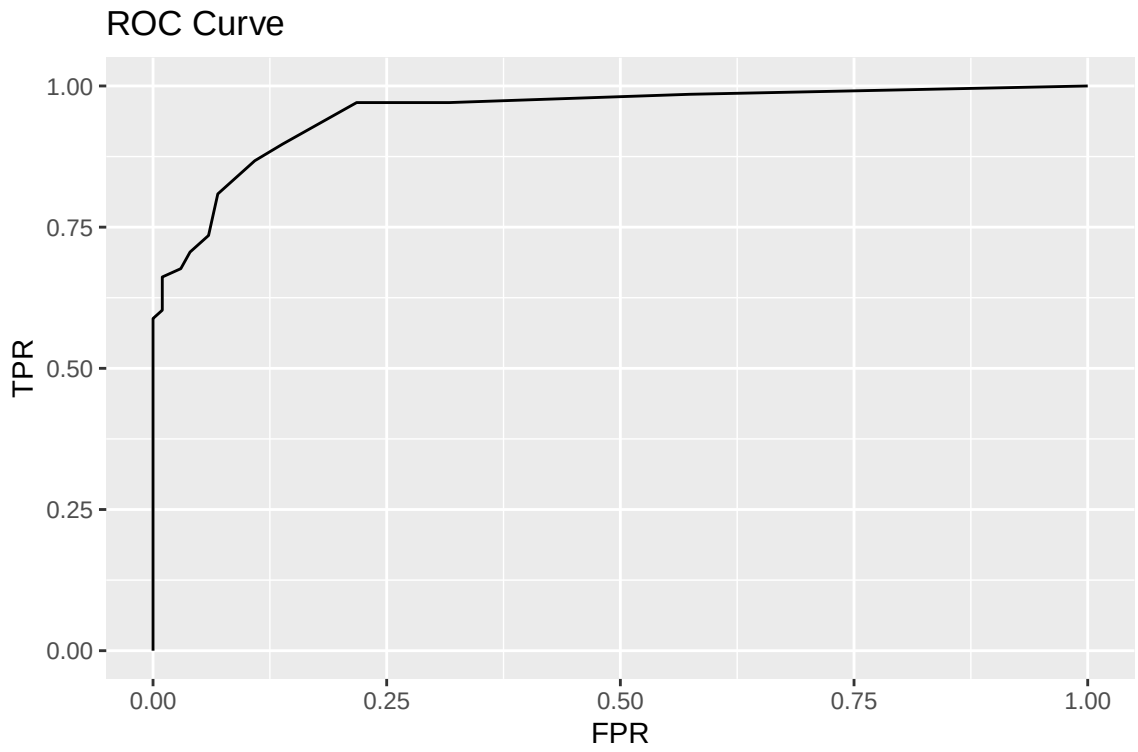



```
##  cut off of 0.75
  pred3c <- predict(qda.model, type = "response", newdata = dense.dat)$posterior[, 2]

  qda.label.test <- rep("B", nrow(dense.dat))
  qda.label.test[pred3c > .75] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = qda.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - Quadratic Discriminant Analysis (0.75)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```



Question 3D. Below is the ROC curve for the QDA model.

```
## roc curve
  n_segm = 21
  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("B", nrow(test))
    qda.label.test[qda.pred.test$posterior[,2] > p_th[i]] = "M"

    TPR[i] = mean(qda.label.test[test$diagnosis == "M"] == test$diagnosis[test$diagnosis == "M"])
    FPR[i] = mean(qda.label.test[test$diagnosis == "B"] != test$diagnosis[test$diagnosis == "B"])
  }

# plot(x = FPR, y = TPR, 'l')
  ggplot() +
    geom_path(aes(x = FPR, y = TPR)) +
    labs(title = "ROC Curve")
```



Question 3E. The area under the ROC curve is 0.946.

```
## area under the curve
  auc.vec <- c()
    for (i in 1:20){
      auc.vec[i] <- ((FPR[i]-FPR[(i+1)]) * ((TPR[i]+TPR[(i+1)])/2))
    }

  (auc <- sum(auc.vec))
```

```
## [1] 0.9517327
```

Question 4: K-Nearest Neighbor

Question 4A. The below table shows the prediction accuracy for the KNN models where k = 1, 2, 3, 4, and 20. In general, the prediction error on the training sets is high relative to that of the test set. This is evidence of overfitting. In particular, when k = 1 the prediction accuracy on the training set is perfect. This is expected when k = 1. The level of k with the highest prediction accuracy on the test set is when k = 20. The prediction accuracy for the knn model with k=20 is about 87%

```
## KNN Model set up
  train.num <- train %>% dplyr::select(-diagnosis)
  train.num <- scale(train.num)
  test.num <- test %>% dplyr::select(-diagnosis)
  test.num <- scale(test.num)

## KNN model
```

```
k.vec <- c(1:4, 20)
knn.mat <- matrix(ncol = 2, nrow = length(k.vec))

for(i in k.vec){
knn.train.pred <- knn(train = train.num,
                      test = train.num,
                      cl = train$diagnosis, k = i)

knn.test.pred <- knn(train = train.num,
                      test = test.num,
                      cl = train$diagnosis, k = i)

tt.knn.train = table(knn.train.pred, train$diagnosis)
knn.mat[(which(k.vec == i)),1] <- mean(knn.train.pred == train$diagnosis)

tt.knn.test = table(knn.test.pred, test$diagnosis)
knn.mat[(which(k.vec == i)),2] <- mean(knn.test.pred == test$diagnosis)
}

colnames(knn.mat) = c("Prediction Accuracy Train", "Prediction Accuracy Test")
knn.df <- as.data.frame(knn.mat)
knn.df$k <- c(1,2,3,4,20)
knn.df <- knn.df[,c(3,1,2)]
kable(knn.df, caption = "Prediction Accuracy - KNN Models", digits = 3)
```

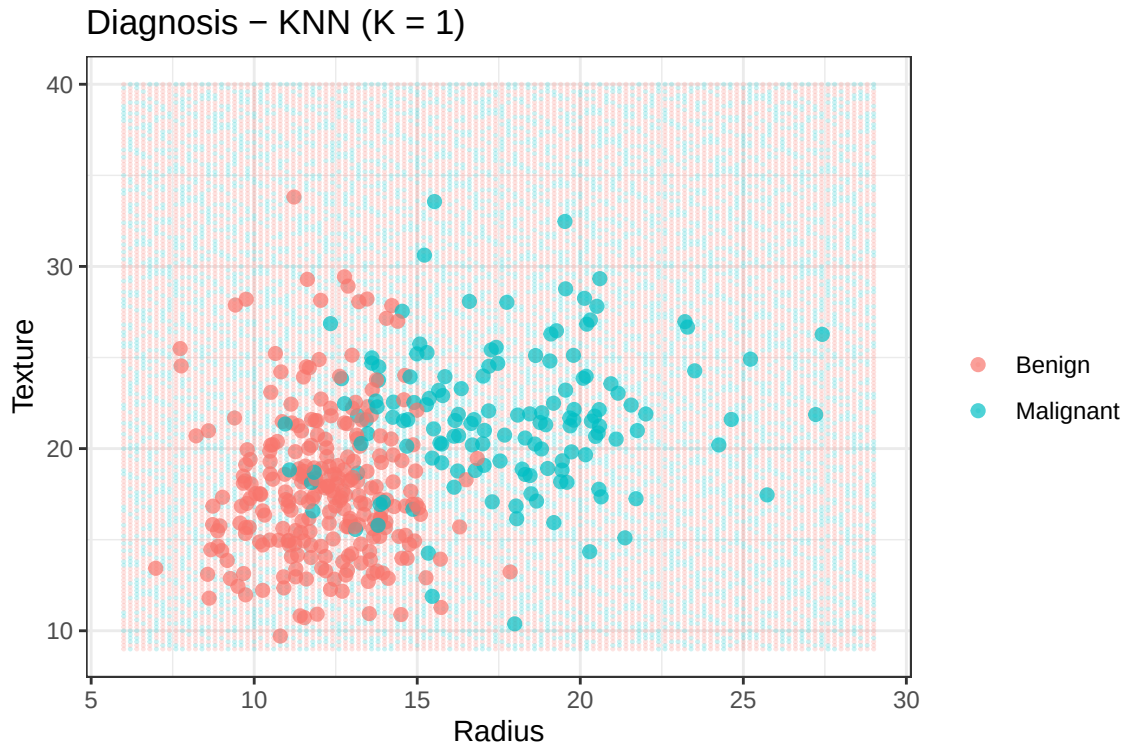
Table 8: Prediction Accuracy - KNN Models

k	Prediction Accuracy Train	Prediction Accuracy Test
1	1.000	0.864
2	0.915	0.822
3	0.927	0.846
4	0.917	0.852
20	0.905	0.864

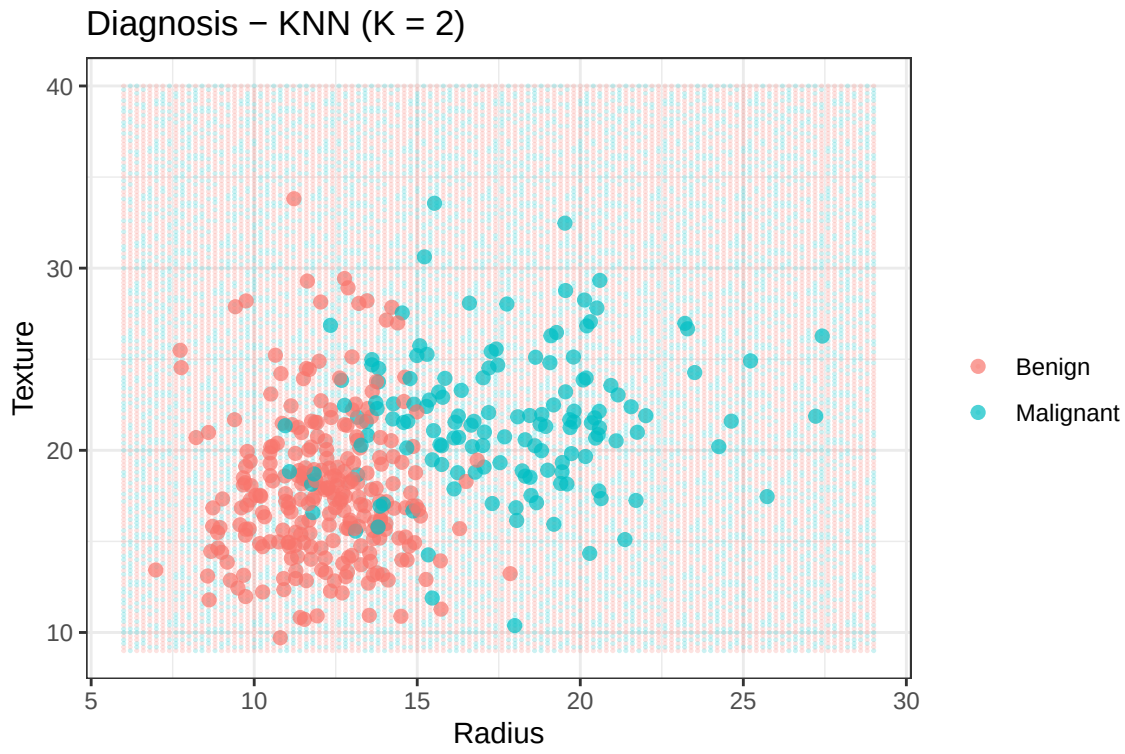
Question 4B. Below are the decision boundary plots for the KNN models of differing values of K. The dense grid that makes up the background of the plot is no longer divided with a decision boundary into red and blue sections like the plots of the previous models were. This makes sense because the KNN model is the only completely non-parametric model of the four. With the KNN model there is no line or quadratic that is estimated by a set of coefficients. Rather, the prediction is based on the values of the K-closest data points.

```
##  k = 1
  knn.test.pred <- knn(train = train.num,
                       test = train.num,
                       cl = train$diagnosis, k = 1)

  knn.label.test <- rep("B", nrow(dense.dat))
  knn.label.test[knn.test.pred == "M"] <- "M"

## plot
  ggplot()+
```

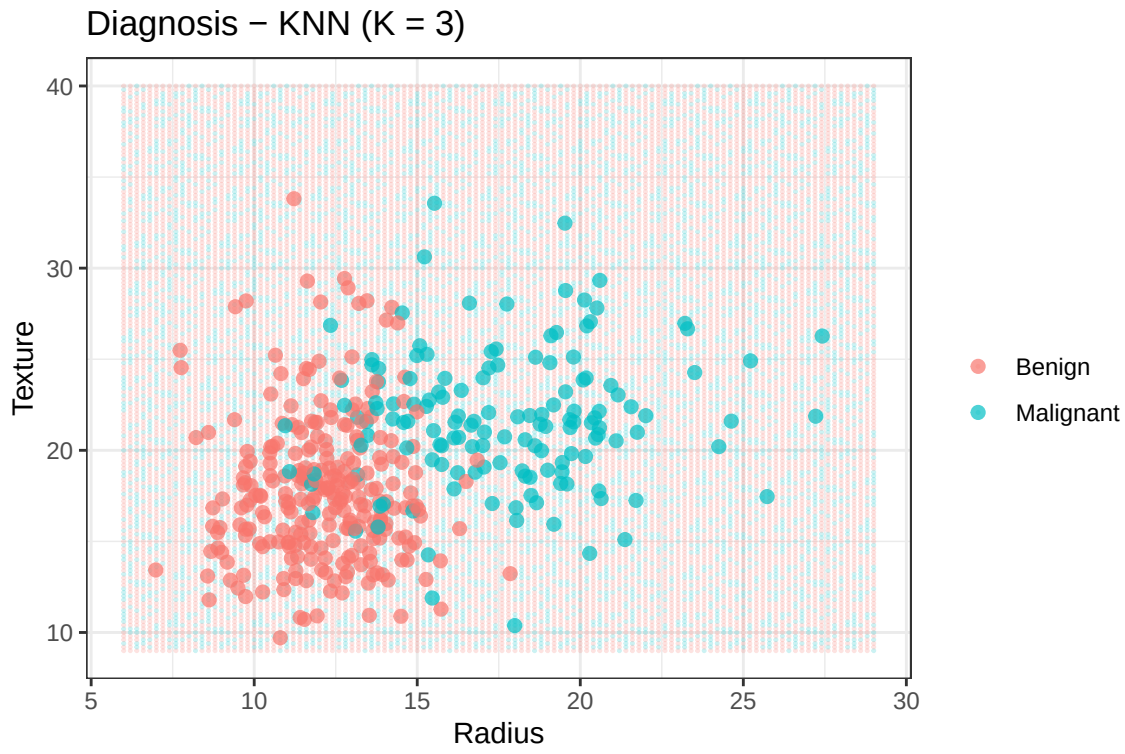
```
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = knn.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - KNN (K = 1)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```



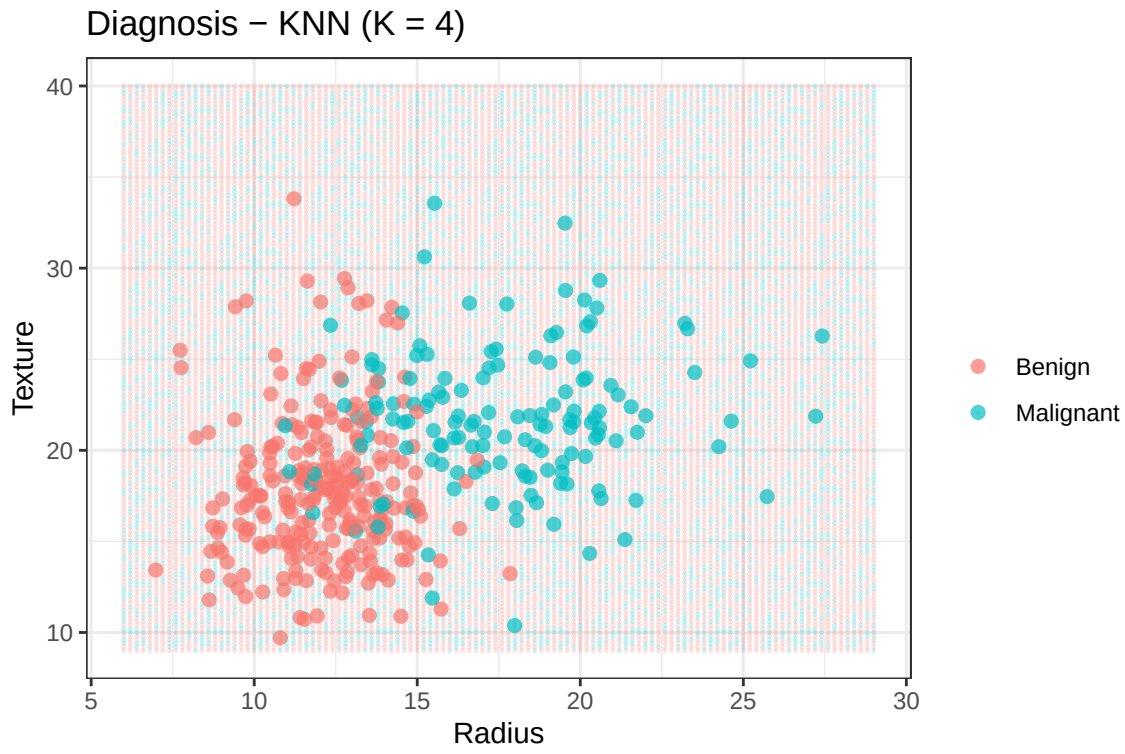

```
##  k = 2
  knn.test.pred <- knn(train = train.num,
                       test = test.num,
                       cl = train$diagnosis, k = 2)

  knn.label.test <- rep("B", nrow(dense.dat))
  knn.label.test[knn.test.pred == "M"] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = knn.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - KNN (K = 2)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```



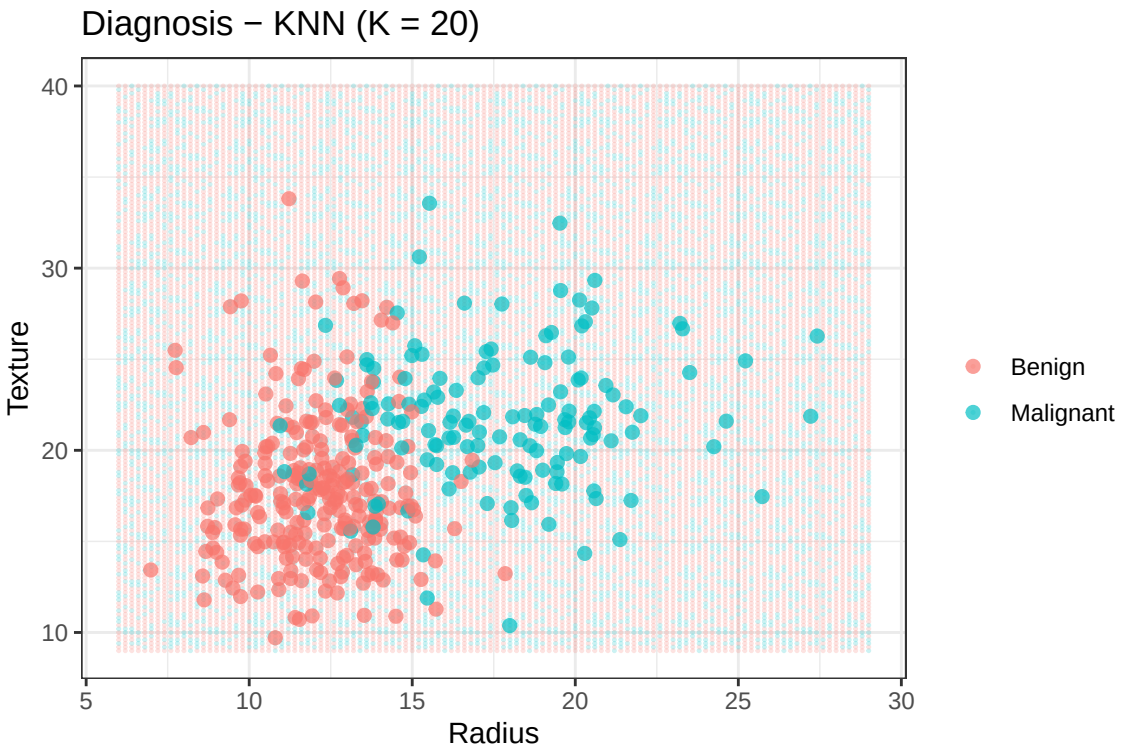

```
##  k = 3
  knn.test.pred <- knn(train = train.num,
                       test = test.num,
                       cl = train$diagnosis, k = 3)

  knn.label.test <- rep("B", nrow(dense.dat))
  knn.label.test[knn.test.pred == "M"] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = knn.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - KNN (K = 3)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```




```
##  k = 4
  knn.test.pred <- knn(train = train.num,
                       test = test.num,
                       cl = train$diagnosis, k = 4)

  knn.label.test <- rep("B", nrow(dense.dat))
  knn.label.test[knn.test.pred == "M"] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = knn.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - KNN (K = 4)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```




```
##  k = 20
  knn.test.pred <- knn(train = train.num,
                       test = test.num,
                       cl = train$diagnosis, k = 20)

  knn.label.test <- rep("B", nrow(dense.dat))
  knn.label.test[knn.test.pred == "M"] <- "M"

## plot
  ggplot()+
    geom_point(aes(x = dense.dat$radius, y = dense.dat$texture, color = knn.label.test), size = 0.2, al
    geom_point(aes(x = train$radius, y = train$texture, color = train$diagnosis), size = 2, alpha = 0.7
    theme_bw() +
    labs(x = "Radius", y = "Texture", title ="Diagnosis - KNN (K = 20)") +
    scale_color_discrete( labels = c("Benign", "Malignant"), name = " ")
```



Question 4C. The below table and plot show the test set prediction accuracy for each value of k from 1 through 20. To maximize prediction, I would choose the KNN model where K = 9. This is the model with the highest prediction accuracy on the test set. The prediction accuracy on the training sets follow a clear downward trend as K increases. In the test set, there is less of a clear trend, however the models where k is 9-12 have the highest prediction accuracies.

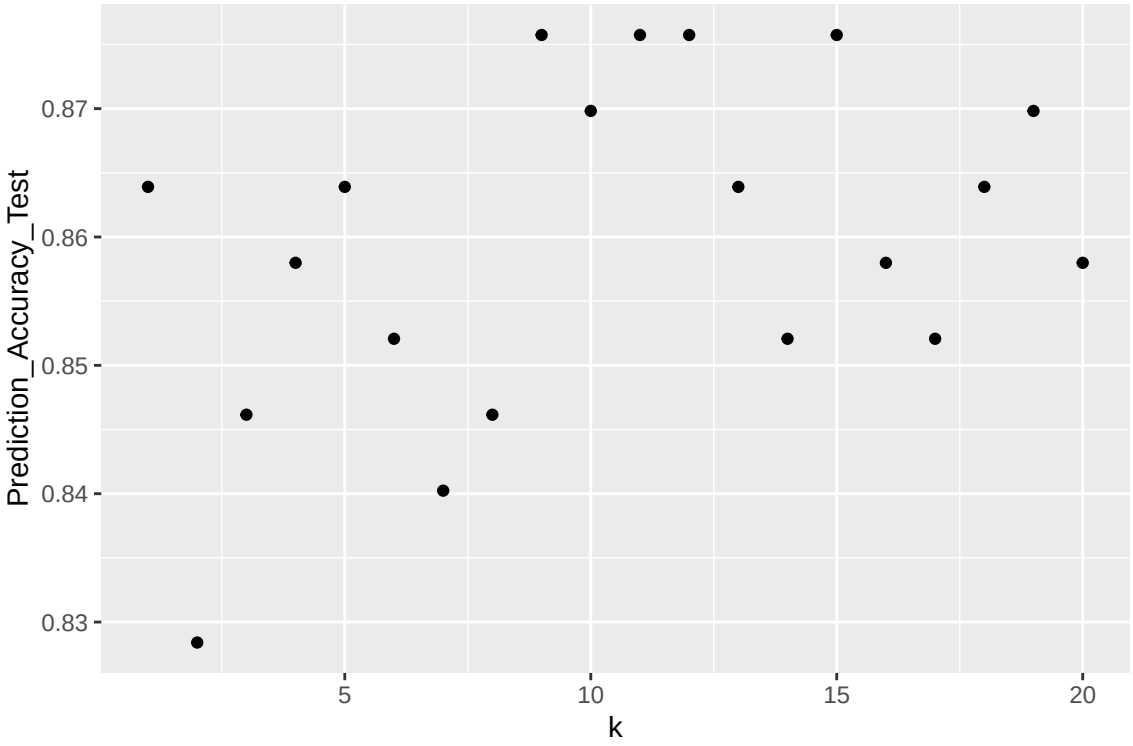
```
## KNN Model set up
  train.num <- train %>% dplyr::select(-diagnosis)
  train.num <- scale(train.num)
  test.num <- test %>% dplyr::select(-diagnosis)
  test.num <- scale(test.num)

## KNN model
  k.vec <- c(1:20)
  knn.mat <- matrix(ncol = 2, nrow = length(k.vec))

  for(i in k.vec){

  knn.train.pred <- knn(train = train.num,
                        test = train.num,
                        cl = train$diagnosis, k = i)

  knn.test.pred <- knn(train = train.num,
                        test = test.num,
                        cl = train$diagnosis, k = i)

  tt.knn.train = table(knn.train.pred, train$diagnosis)
  knn.mat[(which(k.vec == i)),1] <- mean(knn.train.pred == train$diagnosis)
```

```
  tt.knn.test = table(knn.test.pred, test$diagnosis)
  knn.mat[(which(k.vec == i)),2] <- mean(knn.test.pred == test$diagnosis)
  }

  colnames(knn.mat) = c("Prediction_Accuracy_Train", "Prediction_Accuracy_Test")
  knn.df <- as.data.frame(knn.mat)
  knn.df$k <- c(1:20)
  knn.df <- knn.df[,c(3,1,2)]
  kable(knn.df, caption = "Prediction Accuracy - KNN Models", digits = 3)
```

Table 9: Prediction Accuracy - KNN Models

k	Prediction_Accuracy_Train	Prediction_Accuracy_Test
1	1.000	0.864
2	0.935	0.828
3	0.927	0.846
4	0.922	0.858
5	0.925	0.864
6	0.920	0.852
7	0.912	0.840
8	0.910	0.846
9	0.917	0.876
10	0.915	0.870
11	0.915	0.876
12	0.915	0.876
13	0.912	0.864
14	0.920	0.852
15	0.910	0.876
16	0.905	0.858
17	0.905	0.852
18	0.905	0.864
19	0.905	0.870
20	0.897	0.858

```
## plot test prediction accuracies
  ggplot(mapping = aes(x = k, y = Prediction_Accuracy_Test), data = knn.df) +
    geom_point()
```




```
## which k
  which.max(knn.df$Prediction_Accuracy_Test)
```

```
## [1] 9
```

```
  knn9.accuracy <- knn.df$Prediction_Accuracy_Test[9]
```

Question 5: Discussion

The below table summarizes the test set prediction error for all four of the models that I fit. The KNN has the highest prediction accuracy of the four models with an accuracy of 0.88. The logistic regression model is the next most predictive followed by the QDA and then LDA models. We see that logistic regression model has a higher prediction accuracy suggesting that the assumption of a Gaussian distribution within classes does not hold.

The KNN has an advantage because it is the only completely non-parametric model of the four. There is no structure imposed by assumptions placed on the model as is the case with the other three models. Therefore, the KNN model is the most flexible of the four. The predictive advantage of the KNN model is strongest when the data is non-linear. One potential way to mitigate this issue is to use k-fold cross validation so that we can train the model on more training sets and reduce any effects of overfitting.

Table 10: Model Evaluation

Model	Prediction Accuracy on Test Set
GLM	0.870
LDA	0.846
QDA	0.858
KNN (K=9)	0.876