Final Project

Lyndon Swarey, Ryan Folk, Alexa Kelly

May 7, 2020

Packages

The following packages are required. Tidyverse is used for data manipulation, cluster is used for k-means clustering, factoextra is used for visualizing the cluster results, caret package is used to split the data, and fit the K-NN model, rpart is used to fit the classification tree and visualize the results, MASS is used to fit the linear and quadratic discriminant analysis, reshape2 is used to convert the data from wide to long and "select <- dplyr::select" avoids conflict with the MASS package.

```r
library(tidyverse)
library(cluster)
library(factoextra)
library(caret)
library(rpart)
library(MASS)
library(reshape2)
select<-dplyr::select
```

Data

The data was loaded using read.csv. To preserve the originial dataset, a new dataset was created from the original. A new ID variable was also created.

```r
# Import data
data_org <- read.csv("/Users/alexa/Desktop/data.csv")

# Create a new file to preserve orginal data
data <- data_org

# Create New_Id column
data <- data %>%
  mutate(new_ID = seq(1:569))
```

Check for Missing Values

The following code checked for missing values. This data set does not have any missing values

```r
anyNA(data)
```

```
## [1] FALSE
```

Variable clustering: mean

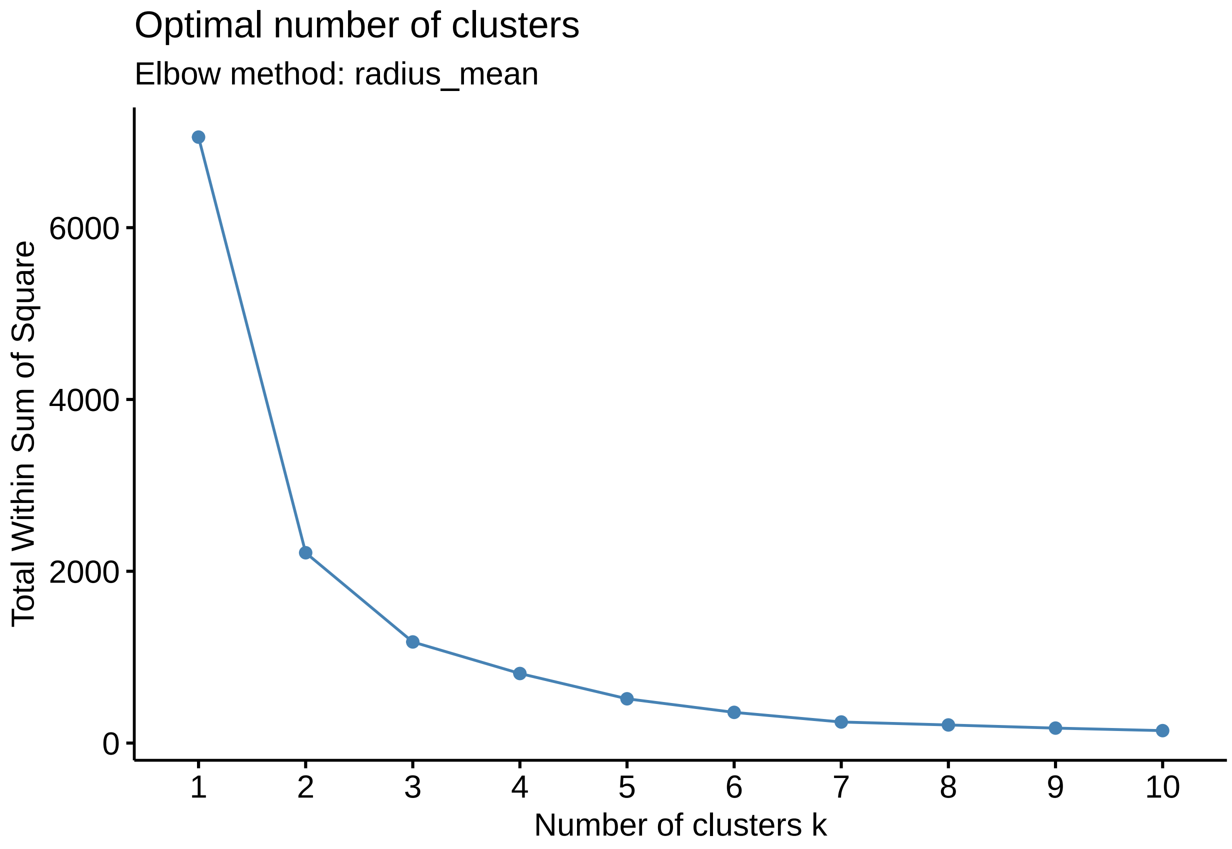
radius_mean

The radius_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_3 was created for the cluster results and a new_ID variable was made as well.

```r
# Select radius_mean
data_3 <- data %>%
  select(radius_mean)

# Set seed for reproducibility
set.seed(123)

# Graph for wws method
fviz_nbclust(data_3, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: radius_mean")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_3, centers = 3, nstart = 10)

# Make new colums in for the new variables generated from kmeans
data_3 <- data_3 %>%
  mutate(radius_mean_wss = as.factor(as.factor(k_wss$cluster)),
         new_ID = seq(1:569))
```

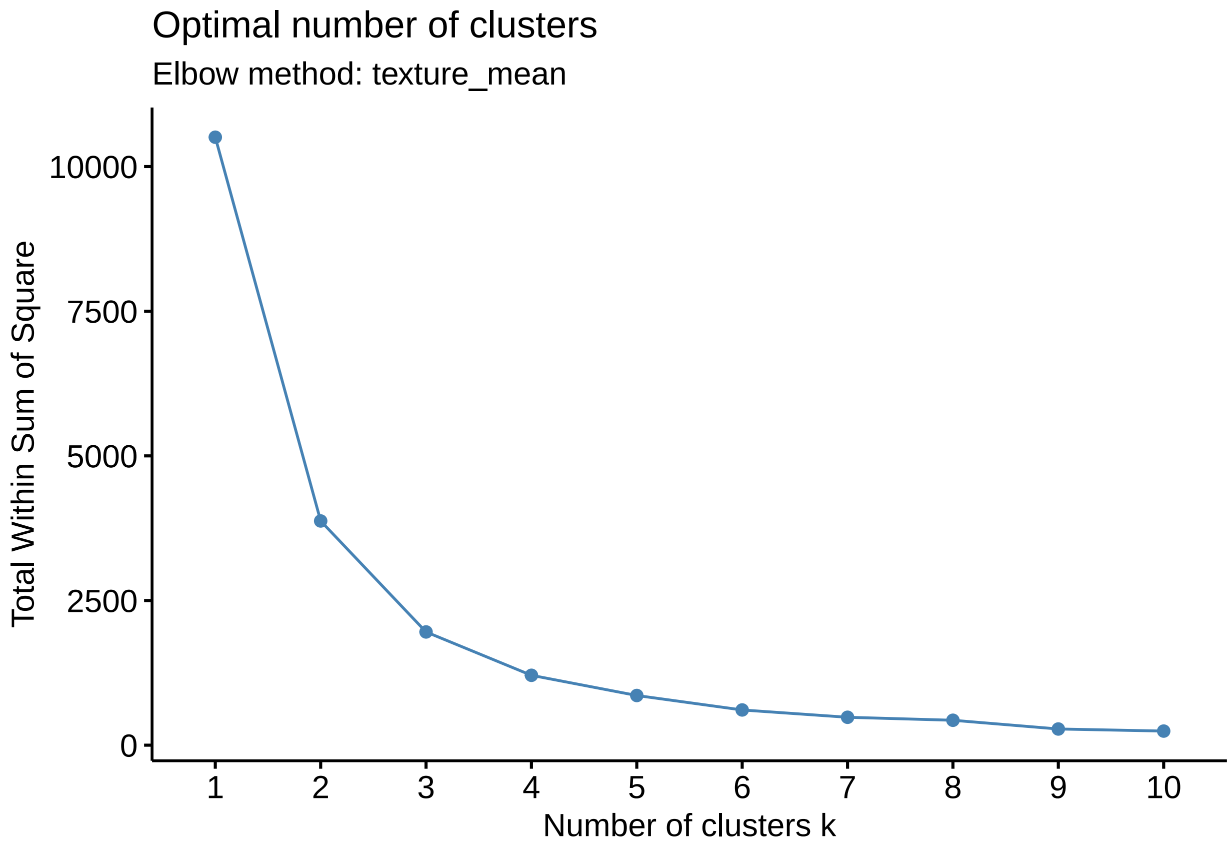
texture_mean

The texture_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_4 was created for the cluster results and a new_ID variable was made as well.

```r
# Select texture_mean
data_4 <- data %>%
  select(texture_mean)

# Set seed for reproducibility
set.seed(123)

# Graph for wws method
fviz_nbclust(data_4, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: texture_mean")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_4, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_4 <- data_4 %>%
  mutate(texture_mean_wss = as.factor(as.factor(k_wss$cluster)),
         new_ID = seq(1:569))
```

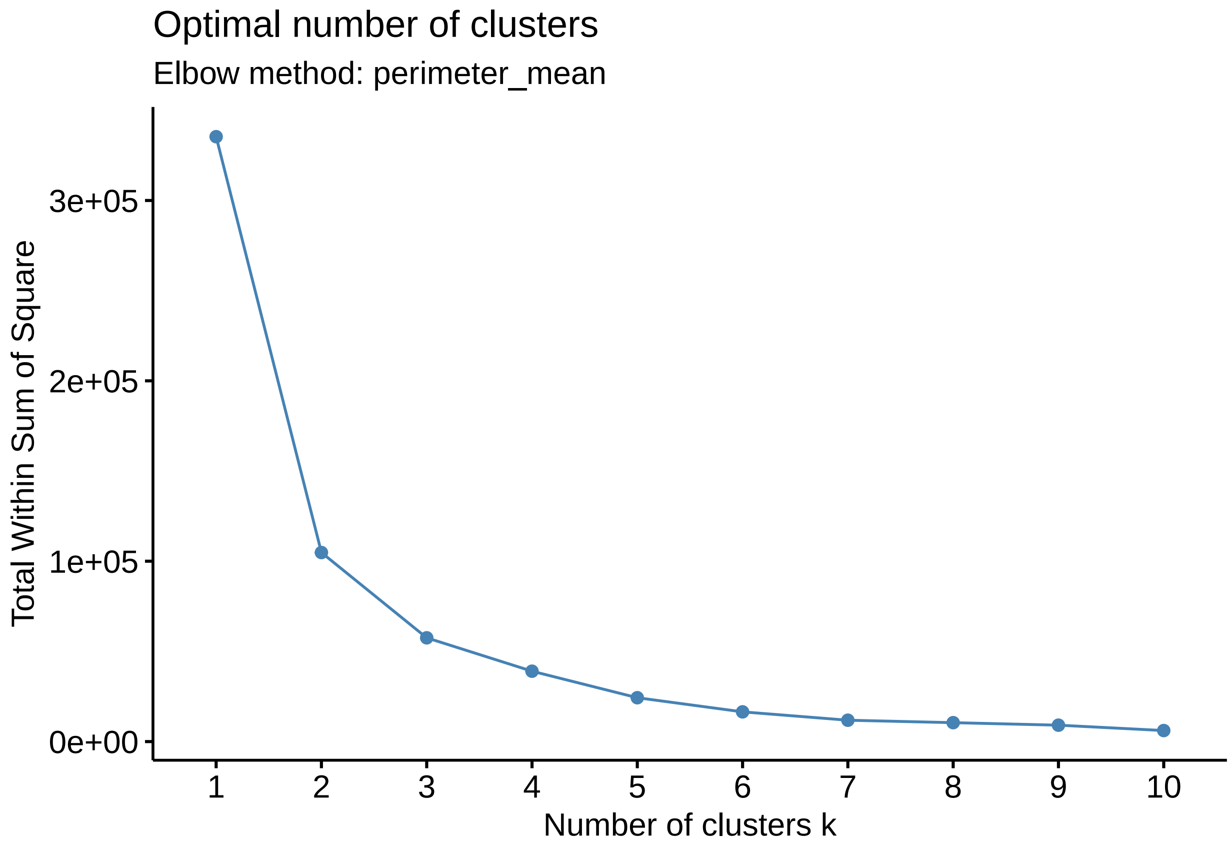
perimeter_mean

The perimeter_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_5 was created for the cluster results and a new_ID variable was made as well.

```r
# Select perimeter_mean
data_5 <- data %>%
  select(perimeter_mean)

# Set seed for reproducibilty
set.seed(123)

# Graph for wws method
fviz_nbclust(data_5, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: perimeter_mean")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_5, centers = 3, nstart = 10)

# Make new colums in data_3 for the new variables generated from kmeans
data_5 <- data_5 %>%
  mutate(perimeter_mean_wss = as.factor(k_wss$cluster),
          new_ID = seq(1:569))
```

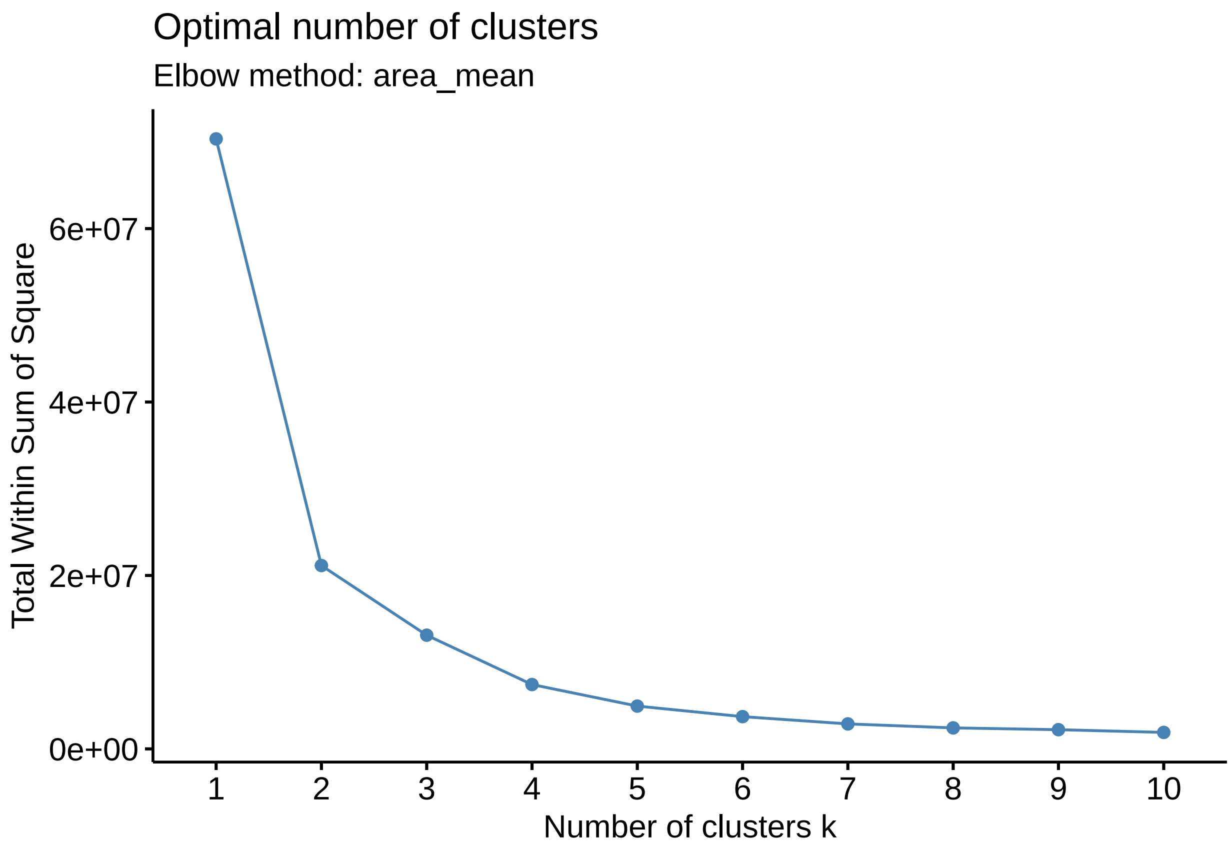
area_mean

The area_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_6 was created for the cluster results and a new_ID variable was made as well. The optimal k appears to be at 4 but needed to be adjusted in order maintain the minimum number of observations for each catorgory for the Chi-square test.

```
# Select area_mean
data_6 <- data %>%
  select(area_mean)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_6, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: area_mean")
```



```
# Kmeans wss method
k_wss <- kmeans(data_6, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_6 <- data_6 %>%
  mutate(area_mean_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

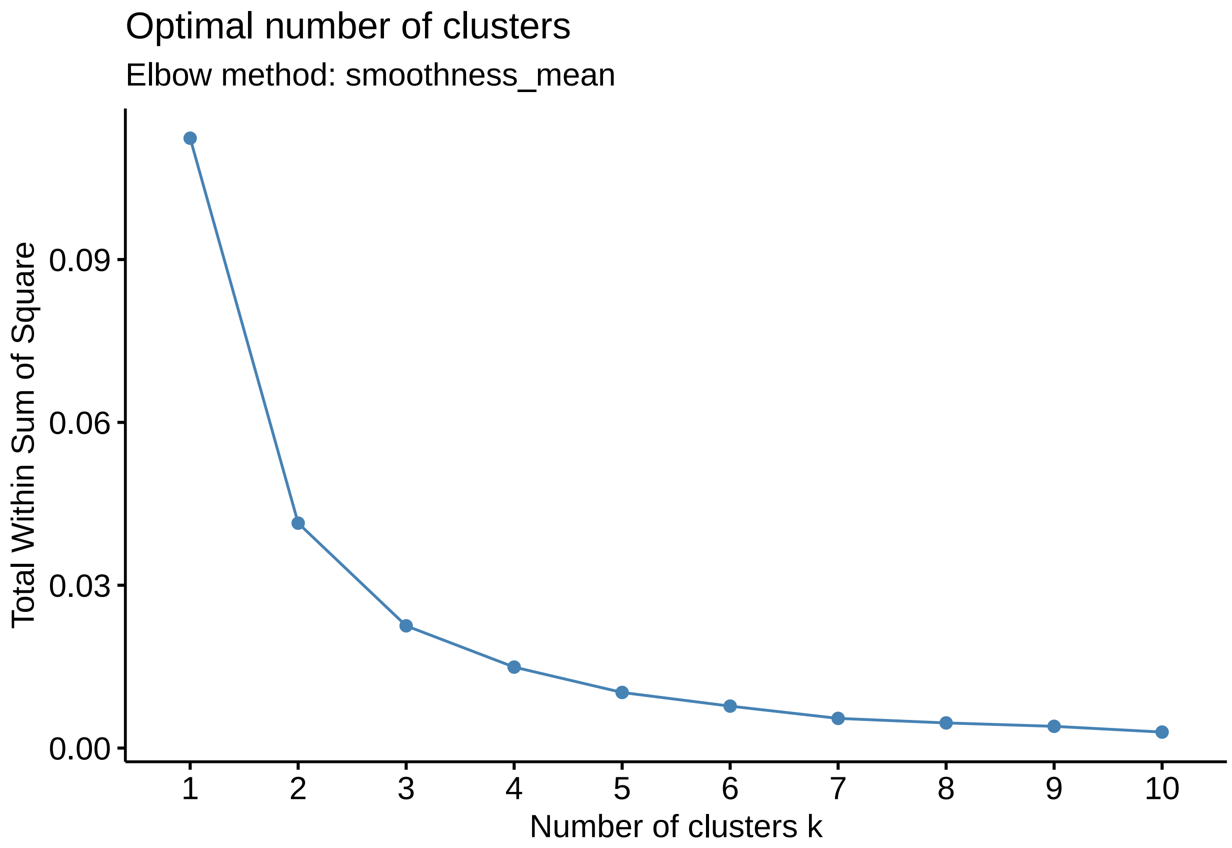
smoothness_mean

The smoothness_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_7 was created for the cluster results and a new_ID variable was made as well.

```r
# Select smoothness_mean
data_7 <- data %>%
  select(smoothness_mean)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_7, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: smoothness_mean")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_7, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_7 <- data_7 %>%
  mutate(smoothness_mean_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

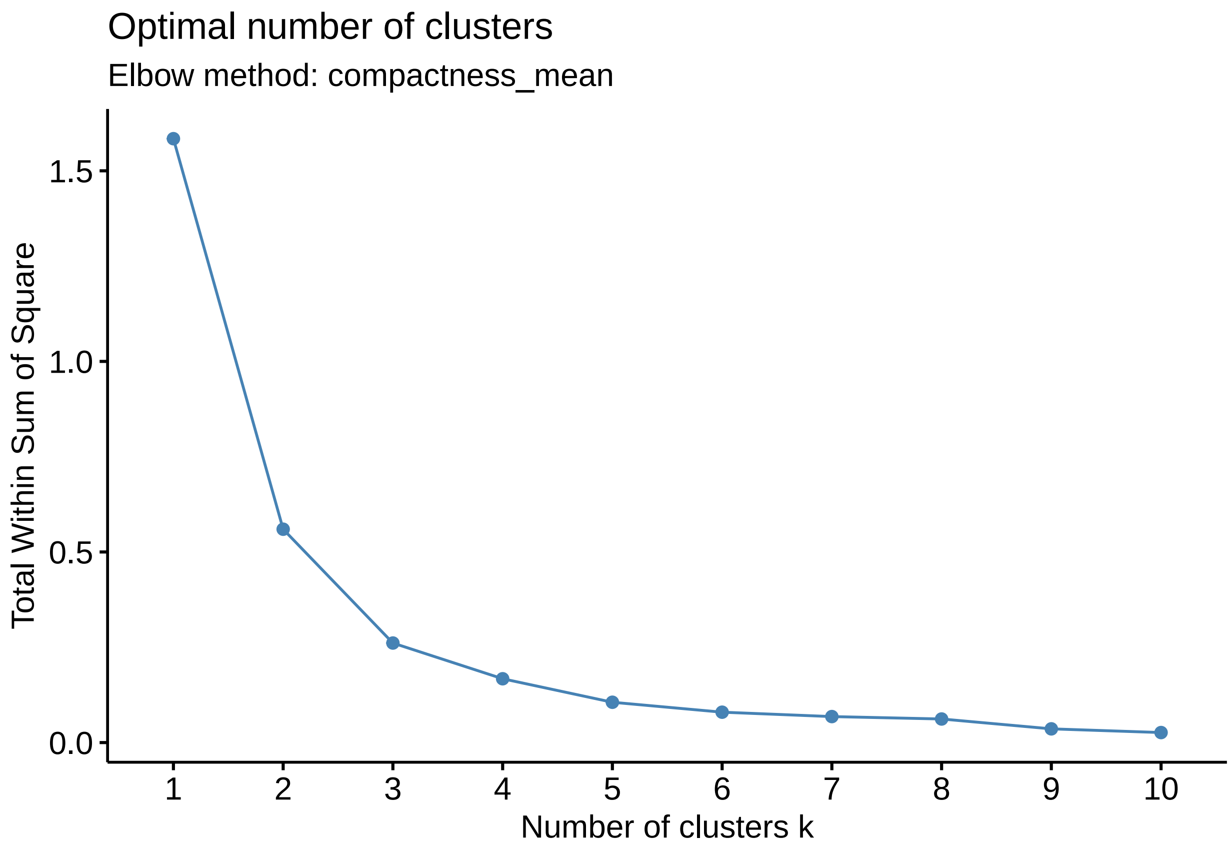
compactness_mean

The compactness_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_8 was created for the cluster results and a new_ID variable was made as well.

```r
# Select smoothness_mean
data_8 <- data %>%
  select(compactness_mean)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_8, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: compactness_mean")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_8, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_8 <- data_8 %>%
  mutate(compactness_mean_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

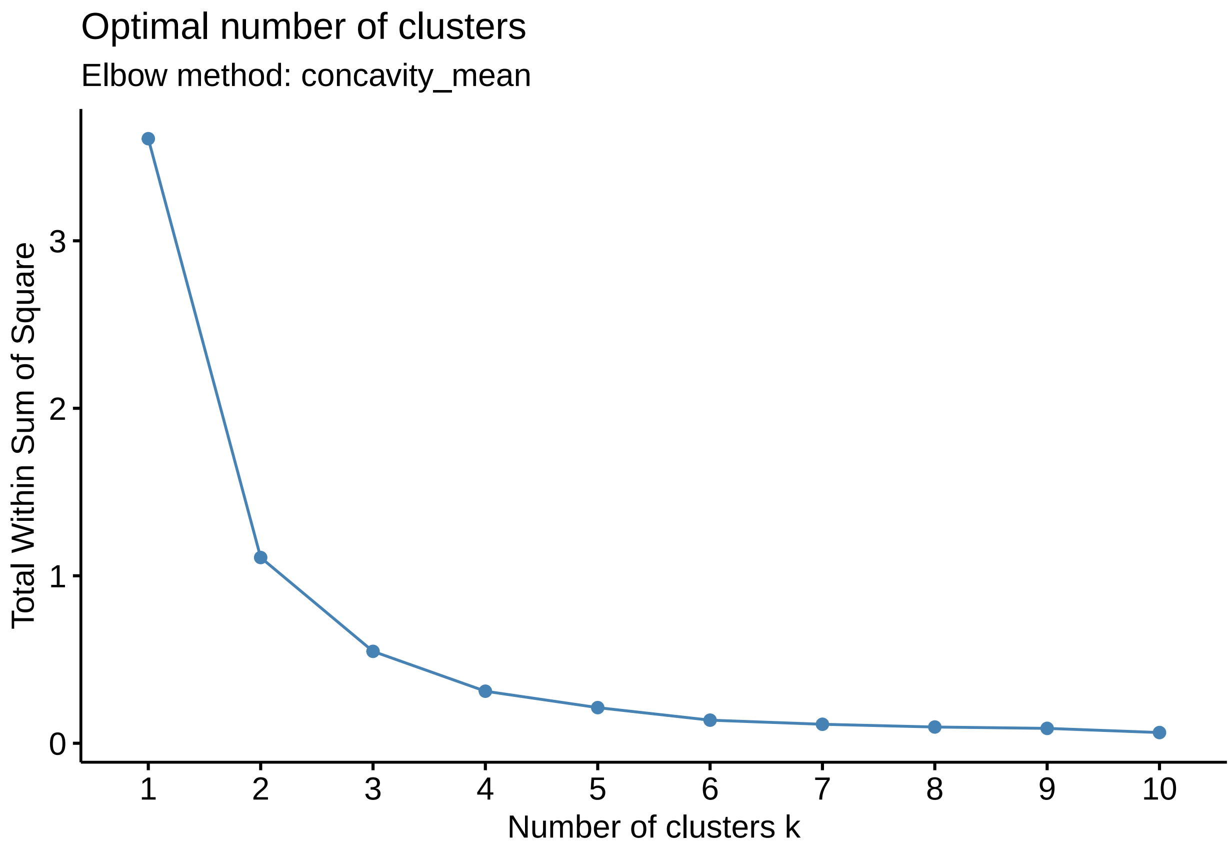
concavity_mean

The concavity_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_9 was created for the cluster results and a new_ID variable was made as well.

```r
# Select concavity_mean
data_9 <- data %>%
  select(concavity_mean)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_9, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: concavity_mean")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_9, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_9 <- data_9 %>%
  mutate(concavity_mean_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

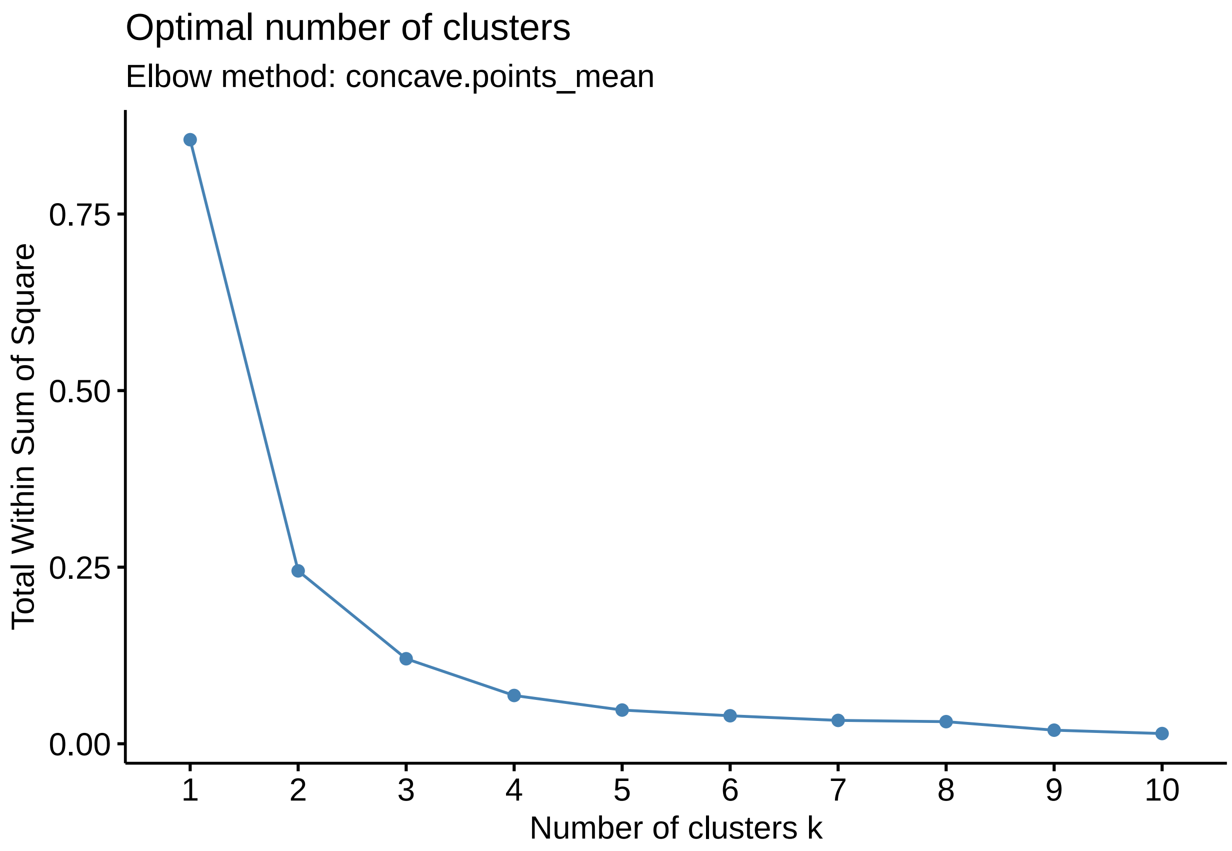
concave.points_mean

The concave.points_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_10 was created for the cluster results and a new_ID variable was made as well.

```r
# Select concave.points_mean
data_10 <- data %>%
  select(concave.points_mean)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_10, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: concave.points_mean")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_10, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_10 <- data_10 %>%
  mutate(concave.points_mean_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

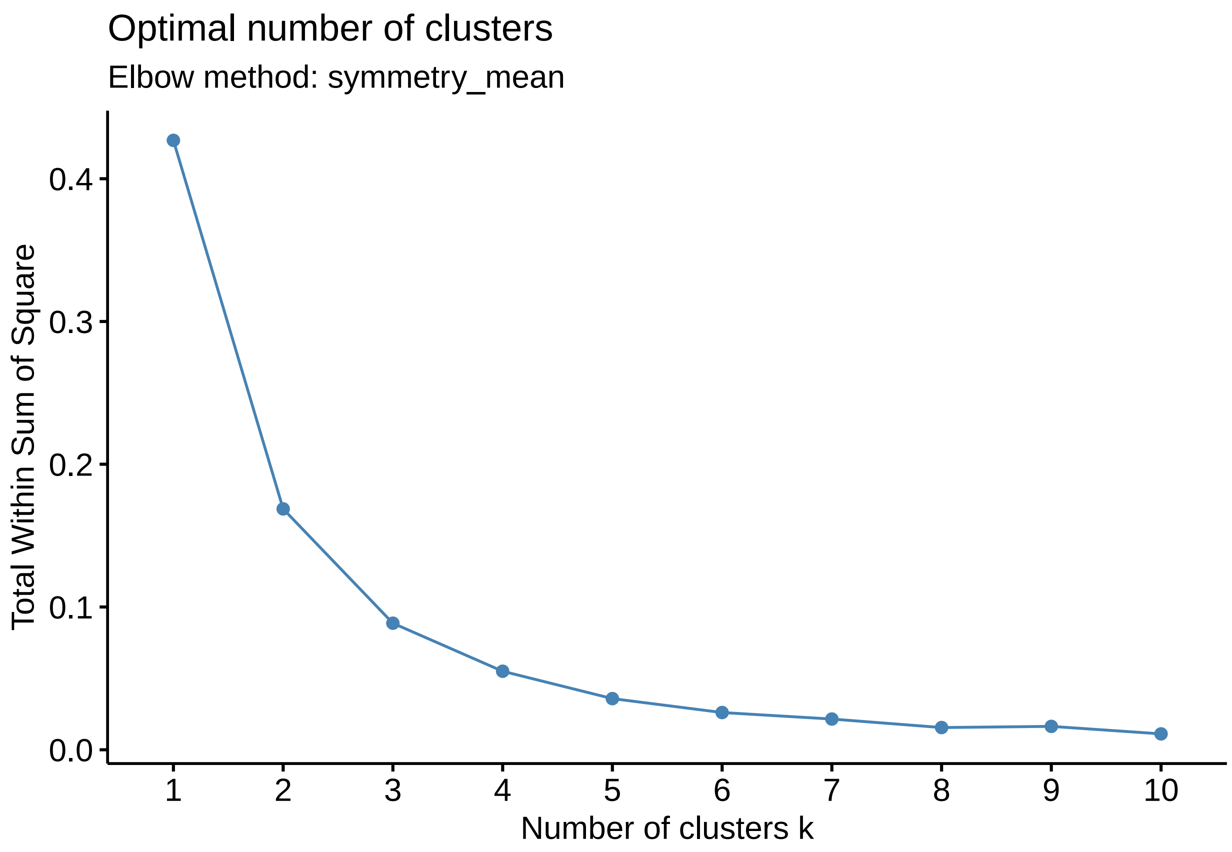
symmetry__mean

The symmetry__mean variable was selected and the seed set for reproducibliity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data__11 was created for the cluster results and a new__ID variable was made as well. The optimal k appears to be at 4 but needed to be adjusted in order maintain the minimum number of observations for each catorgory for the Chi-square test.

```r
# Select symmetry_mean
data_11 <- data %>%
  select(symmetry_mean)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_11, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: symmetry_mean")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_11, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_11 <- data_11 %>%
  mutate(symmetry_mean_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

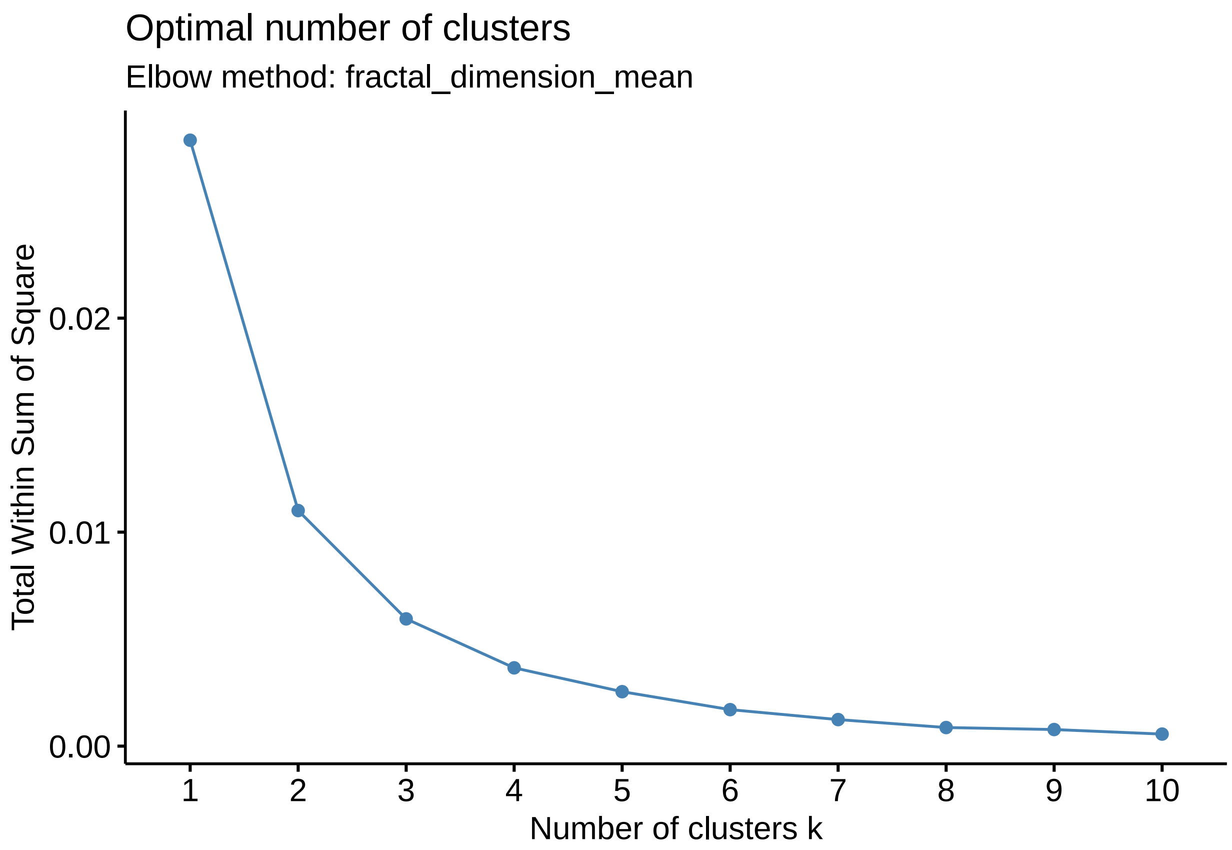
fractal_dimension_mean

The fractal_dimension_mean variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_12 was created for the cluster results and a new_ID variable was made as well.

```
# Select fractal_dimension_mean
data_12 <- data %>%
  select(fractal_dimension_mean)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_12, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: fractal_dimension_mean")
```



```
# Kmeans wss method
k_wss <- kmeans(data_12, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_12 <- data_12 %>%
  mutate(fractal_dimension_mean_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

Variable clustering: se

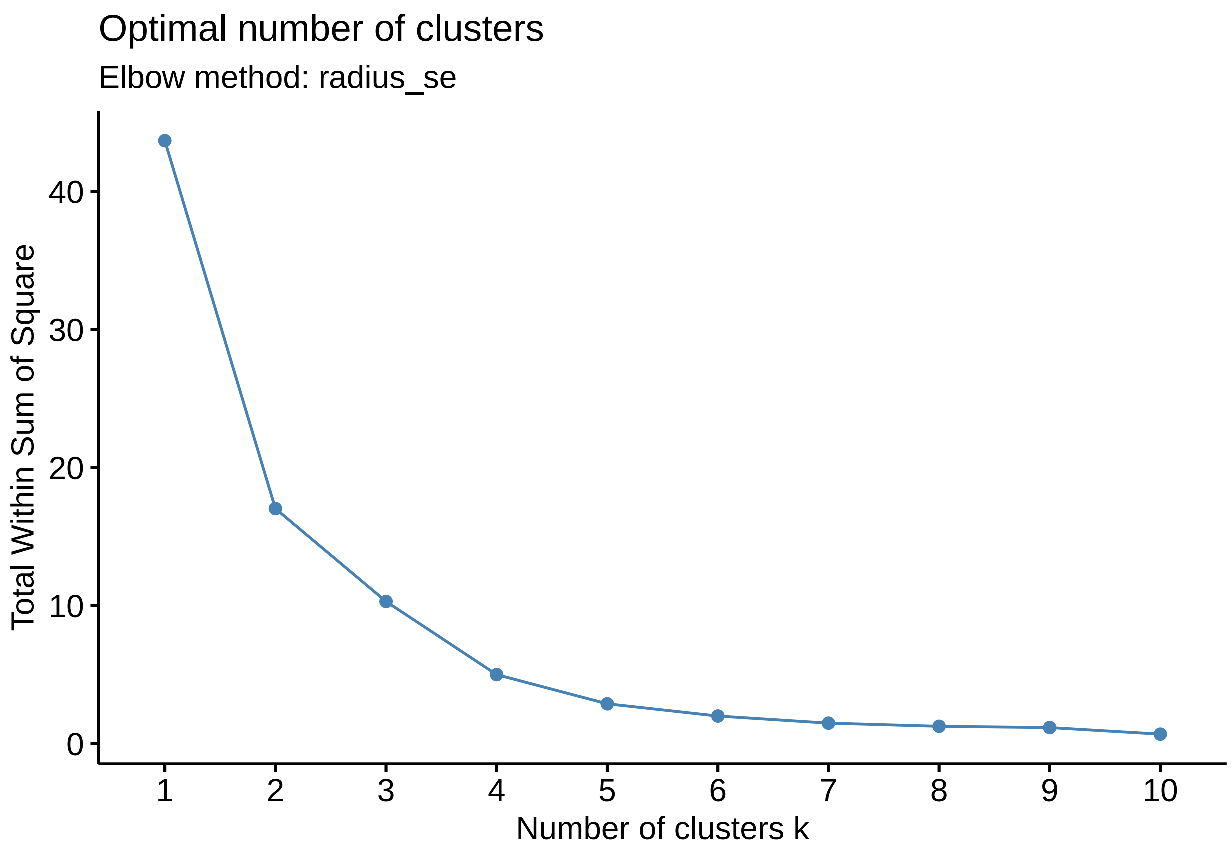
radius_se

The radius_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_13 was created for the cluster results and a new_ID variable was made as well. The optimal k appears to be at 4 but needed to be adjusted in order maintain the minimum number of observations for each catorgory for the Chi-square test.

```
# Select radius_se
data_13 <- data %>%
  select(radius_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_13, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: radius_se")
```



```
# Kmeans wss method
k_wss <- kmeans(data_13, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_13 <- data_13 %>%
  mutate(radius_se_wss = as.factor(k_wss$cluster),
          new_ID = seq(1:569))
```

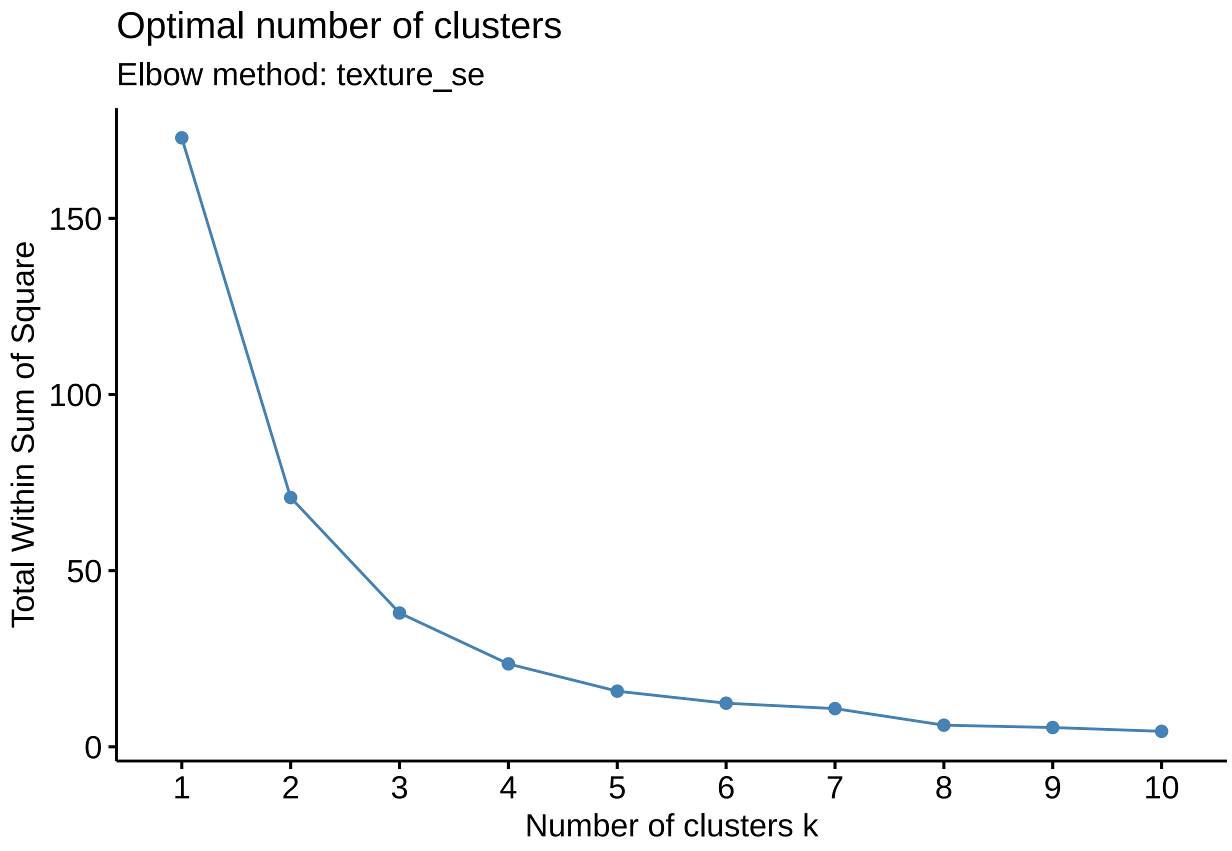
texture_se

The texture_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_14 was created for the cluster results and a new_ID variable was made as well.

```
# Select texture_se
data_14 <- data %>%
  select(texture_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_14, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: texture_se")
```



```
# Kmeans wss method
k_wss <- kmeans(data_14, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_14 <- data_14 %>%
  mutate(texture_se_wss = as.factor(k_wss$cluster),
```

```r
          new_ID = seq(1:569))
```

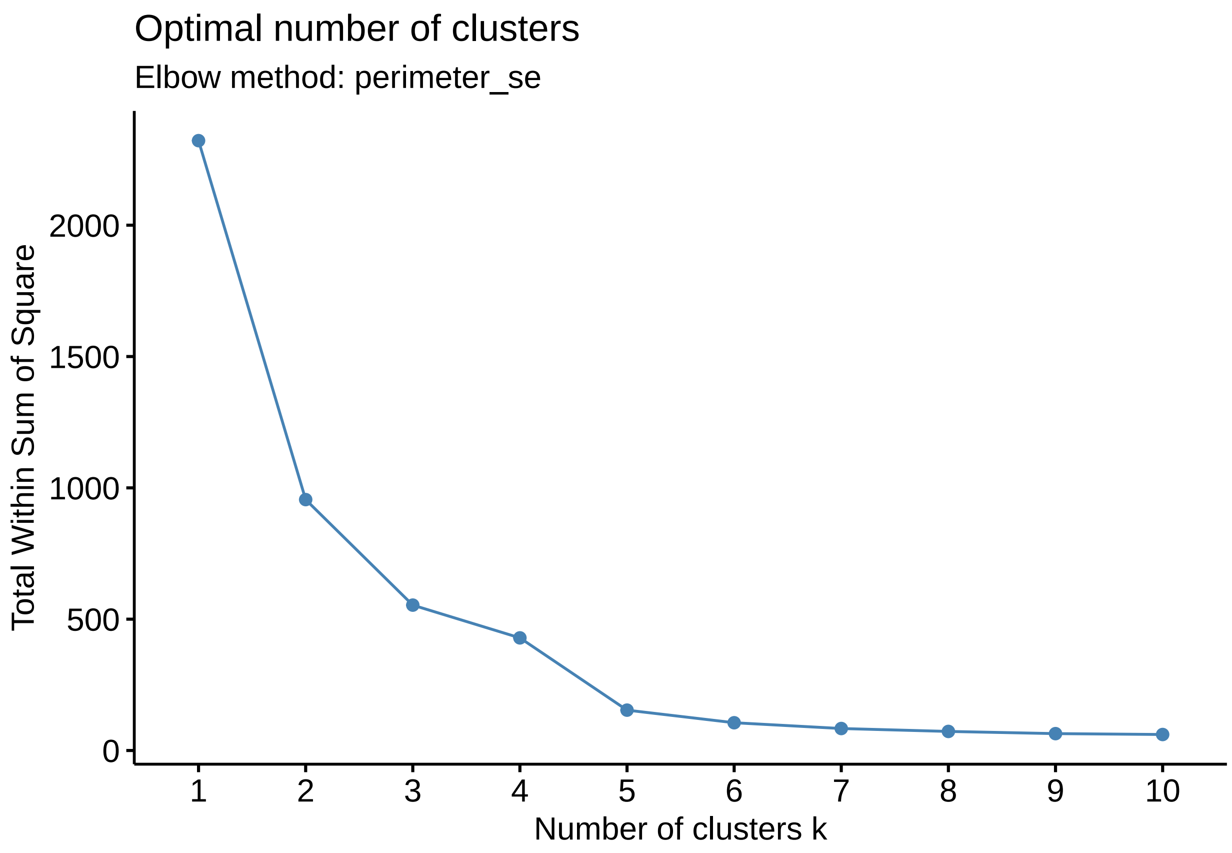
perimeter_se

The perimeter_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_15 was created for the cluster results and a new_ID variable was made as well. The optimal k appears to be at 4 but needed to be adjusted in order maintain the minimum number of observations for each catorgory for the Chi-square test.

```r
# Select perimeter_se
data_15 <- data %>%
  select(perimeter_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_15, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: perimeter_se")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_15, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
```

```r
data_15 <- data_15 %>%
  mutate(perimeter_se_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

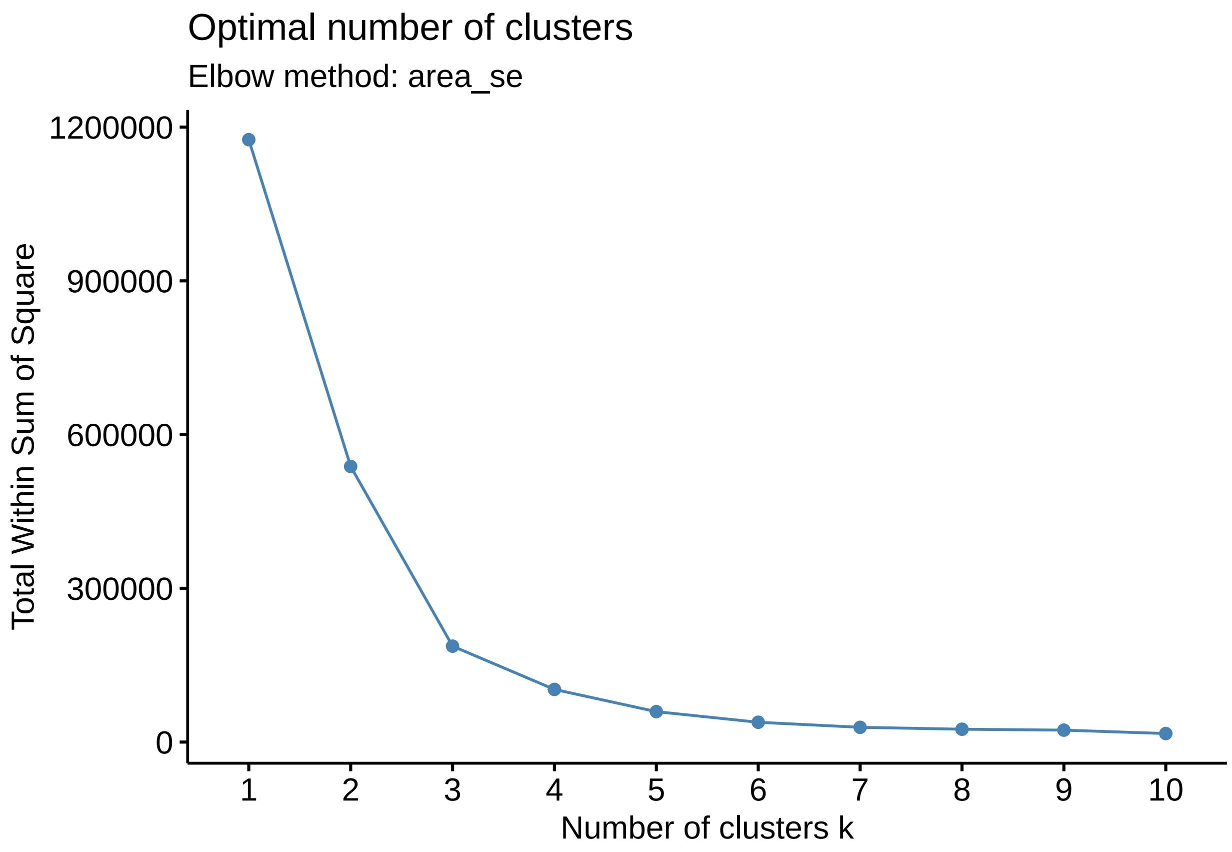
area_se

The area_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=2 with 10 random starts. A new variable in data_16 was created for the cluster results and a new_ID variable was made as well. The optimal k appears to be at 3 but needed to be adjusted in order maintain the minimum number of observations for each catorgory for the Chi-square test.

```r
# Select area_se
data_16 <- data %>%
  select(area_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_16, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: area_se")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_16, centers = 2, nstart = 10)
```

```r
# Make new colums for the new variables generated from kmeans
data_16 <- data_16 %>%
  mutate(area_se_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

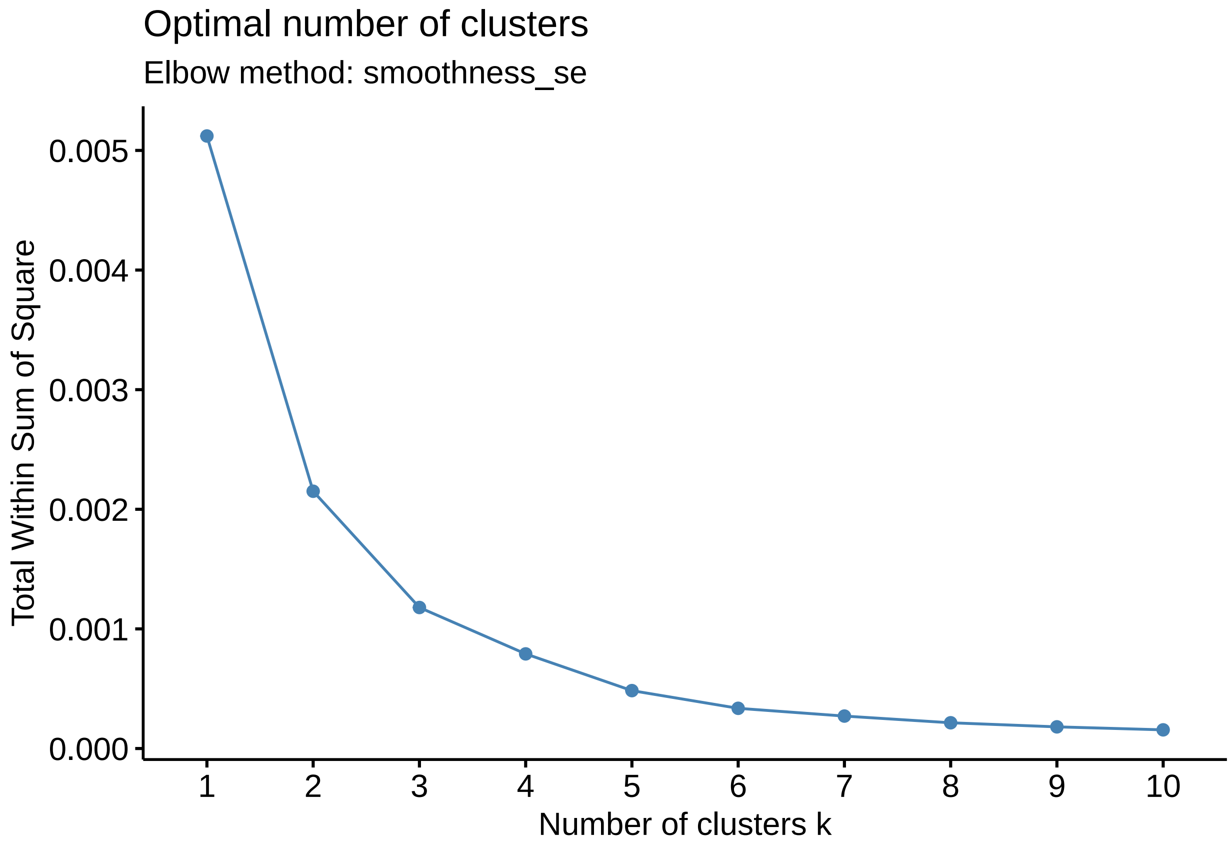
smoothness_se

The smoothness_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_17 was created for the cluster results and a new_ID variable was made as well.

```r
# Select smoothness_se
data_17 <- data %>%
  select(smoothness_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_17, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: smoothness_se")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_17, centers = 3, nstart = 10)
```

```r
# Make new colums for the new variables generated from kmeans
data_17 <- data_17 %>%
  mutate(smoothness_se_wss = as.factor(k_wss$cluster),
          new_ID = seq(1:569))
```

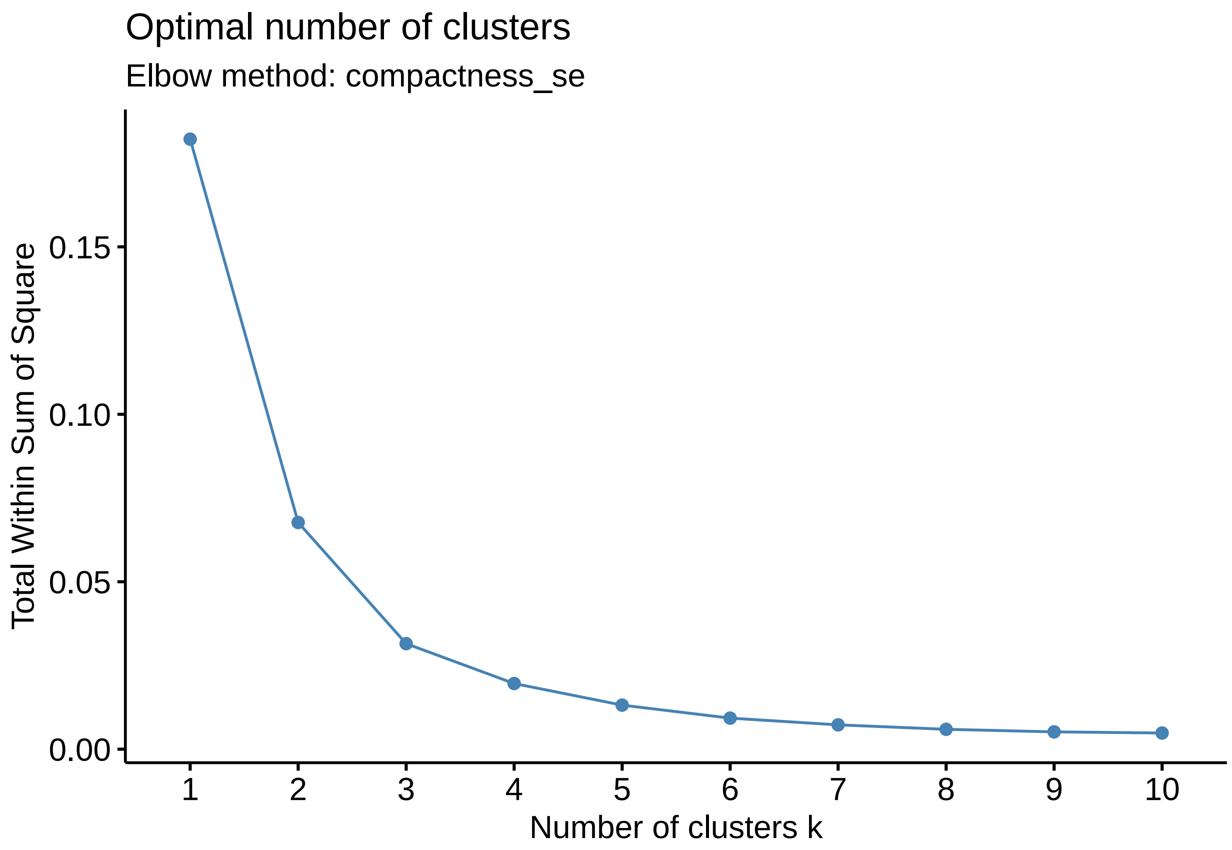
compactness_se

The compactnes_se variable was selected and the seed set for reproducibliity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_18 was created for the cluster results and a new_ID variable was made as well.

```r
# Select compactness_se
data_18 <- data %>%
  select(compactness_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_18, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: compactness_se")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_18, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
```

```
data_18 <- data_18 %>%
  mutate(compactness_se_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

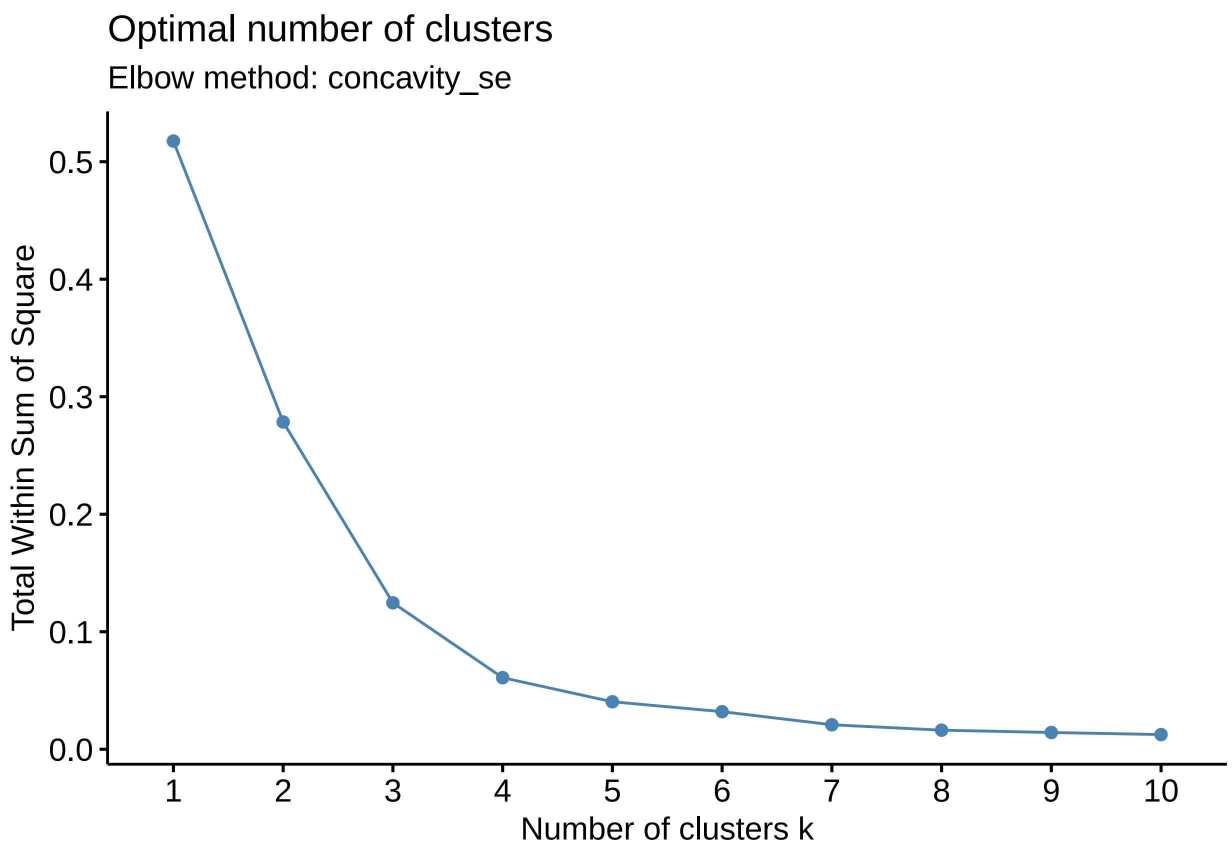
concavity_se

The concavity_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=2 with 10 random starts. A new variable in data_19 was created for the cluster results and a new_ID variable was made as well. The optimal k appears to be at 4 but needed to be adjusted in order maintain the minimum number of observations for each catorgory for the Chi-square test.

```
# Select concavity_se
data_19 <- data %>%
  select(concavity_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_19, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: concavity_se")
```



```
# Kmeans wss method
k_wss <- kmeans(data_19, centers = 2, nstart = 10)
```

```r
# Make new colums for the new variables generated from kmeans
data_19 <- data_19 %>%
  mutate(concavity_se_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

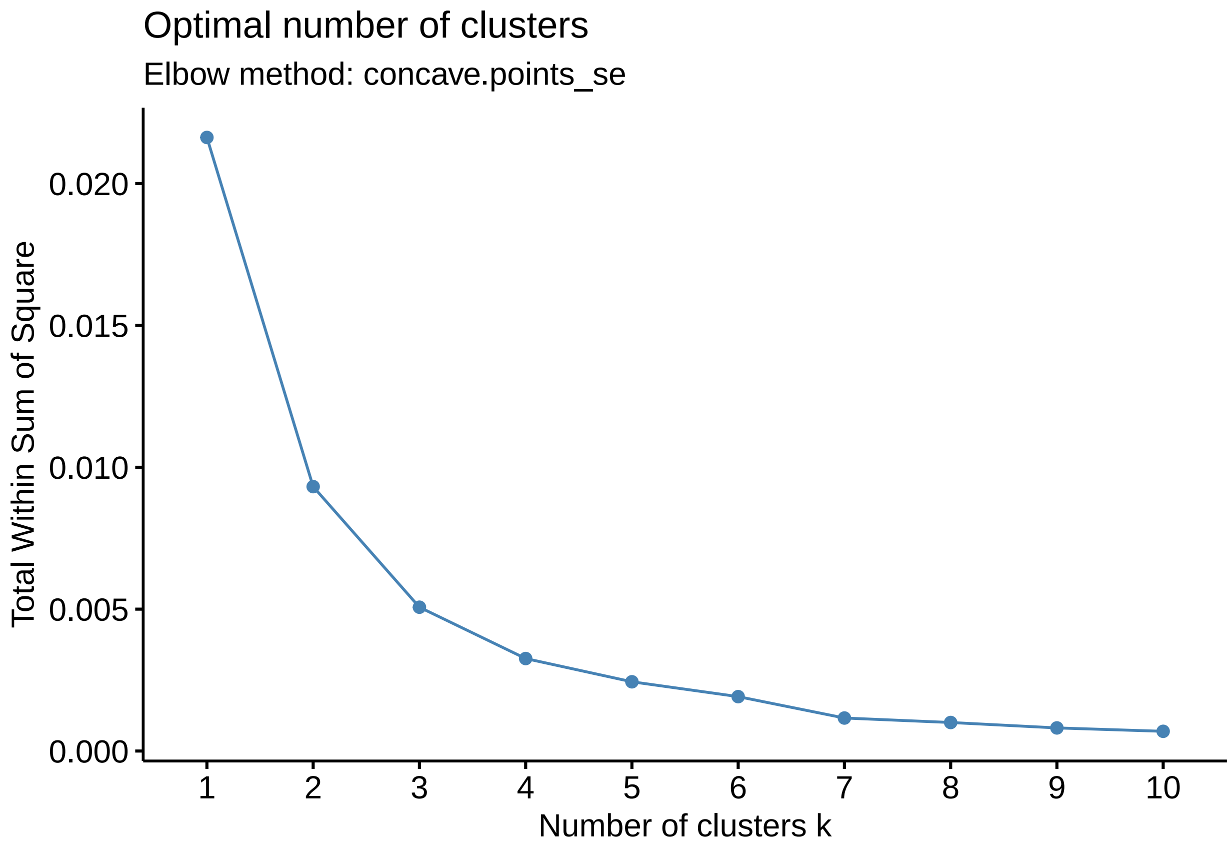
concave.points_se

The concave.points_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_20 was created for the cluster results and a new_ID variable was made as well.

```r
# Select concave.points_se
data_20 <- data %>%
  select(concave.points_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_20, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: concave.points_se")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_20, centers = 4, nstart = 10)
```

```r
# Make new colums for the new variables generated from kmeans
data_20 <- data_20 %>%
  mutate(concave.points_se_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

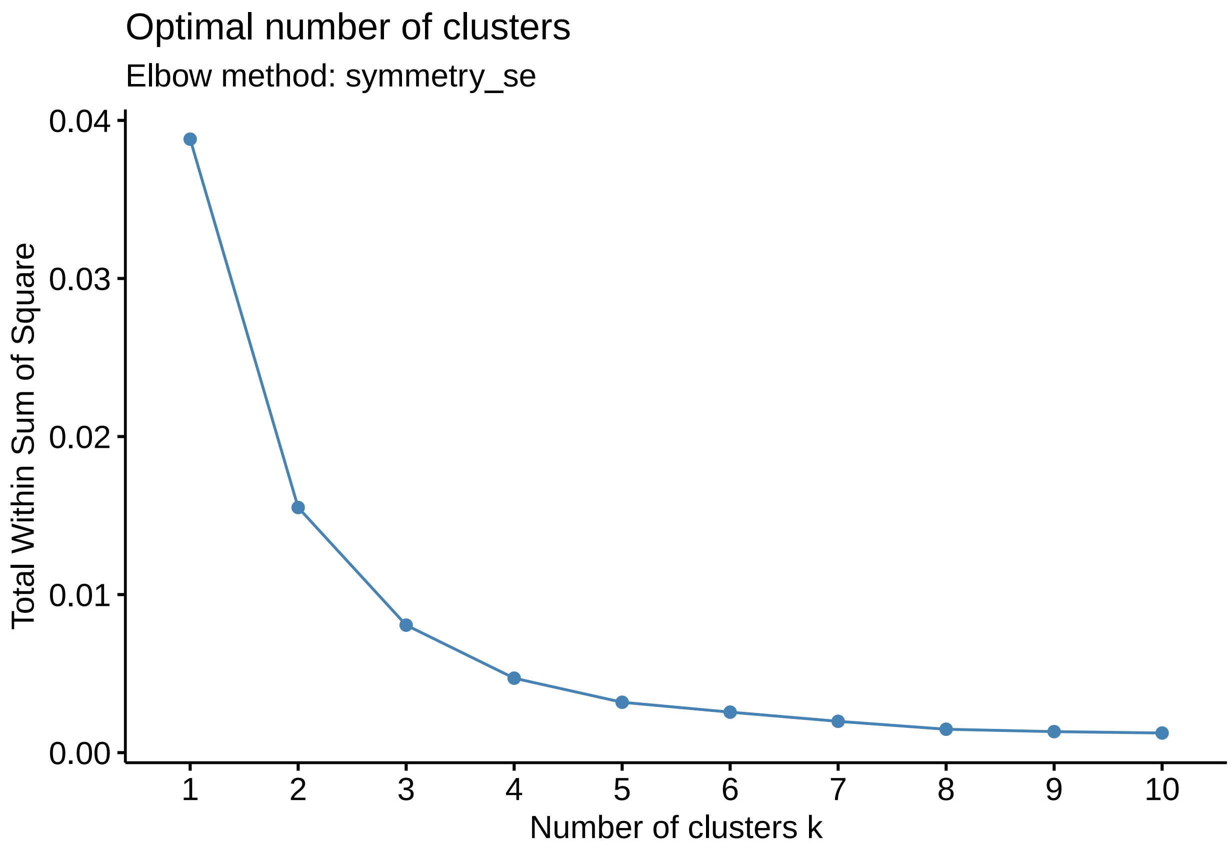
symmetry_se

The symmetry_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_21 was created for the cluster results and a new_ID variable was made as well.

```r
# Select symetry_se
data_21 <- data %>%
  select(symmetry_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_21, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: symmetry_se")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_21, centers = 4, nstart = 10)

# Make new colums for the new variables generated from kmeans
```

```
data_21 <- data_21 %>%
  mutate(symmetry_se_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

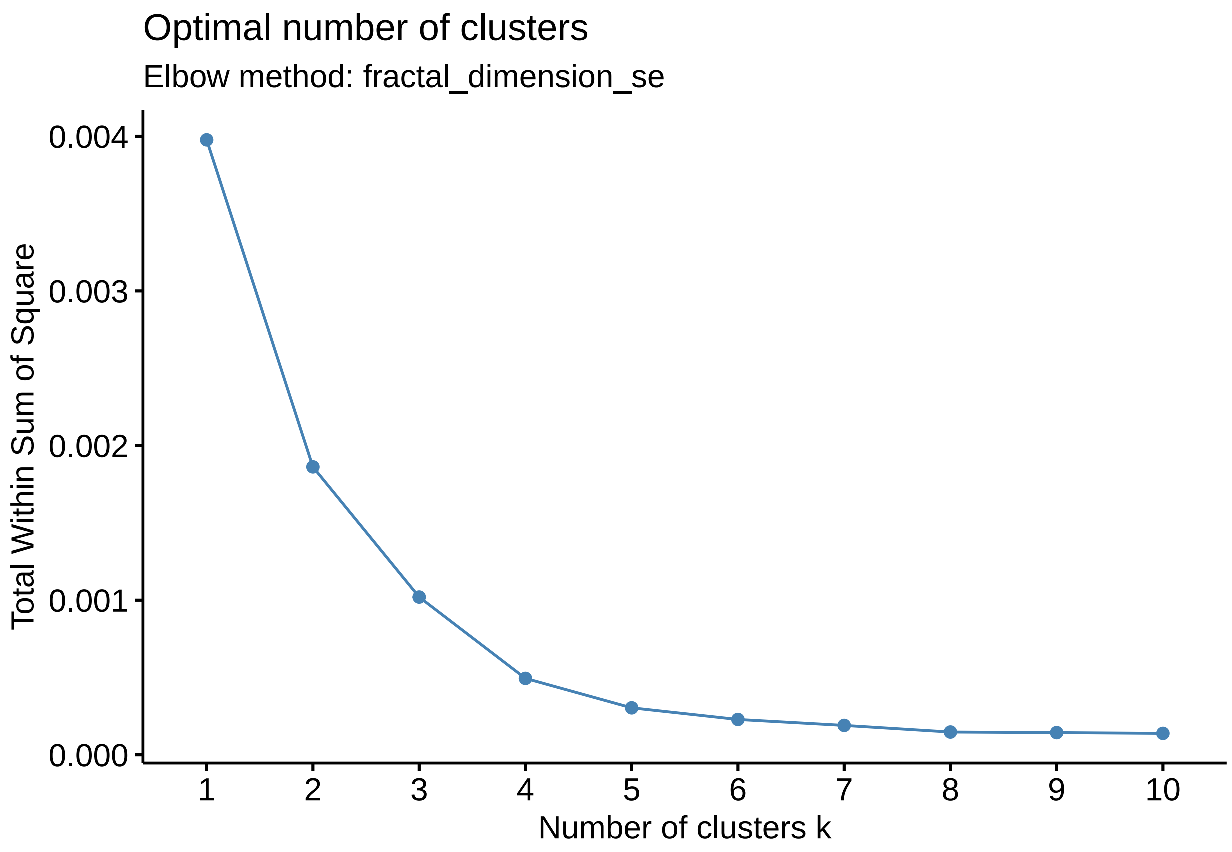
fractal_dimension_se

The fractal_dimension_se variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_22 was created for the cluster results and a new_ID variable was made as well. The optimal k appears to be at 4 but needed to be adjusted in order maintain the minimum number of observations for each catorgory for the Chi-square test.

```
# Select fractal_dimension_mean
data_22 <- data %>%
  select(fractal_dimension_se)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_22, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: fractal_dimension_se")
```



```
# Kmeans wss method
k_wss <- kmeans(data_22, centers = 3, nstart = 10)
```

```r
# Make new colums for the new variables generated from kmeans
data_22 <- data_22 %>%
  mutate(fractal_dimension_se_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

Variable clustering: worst

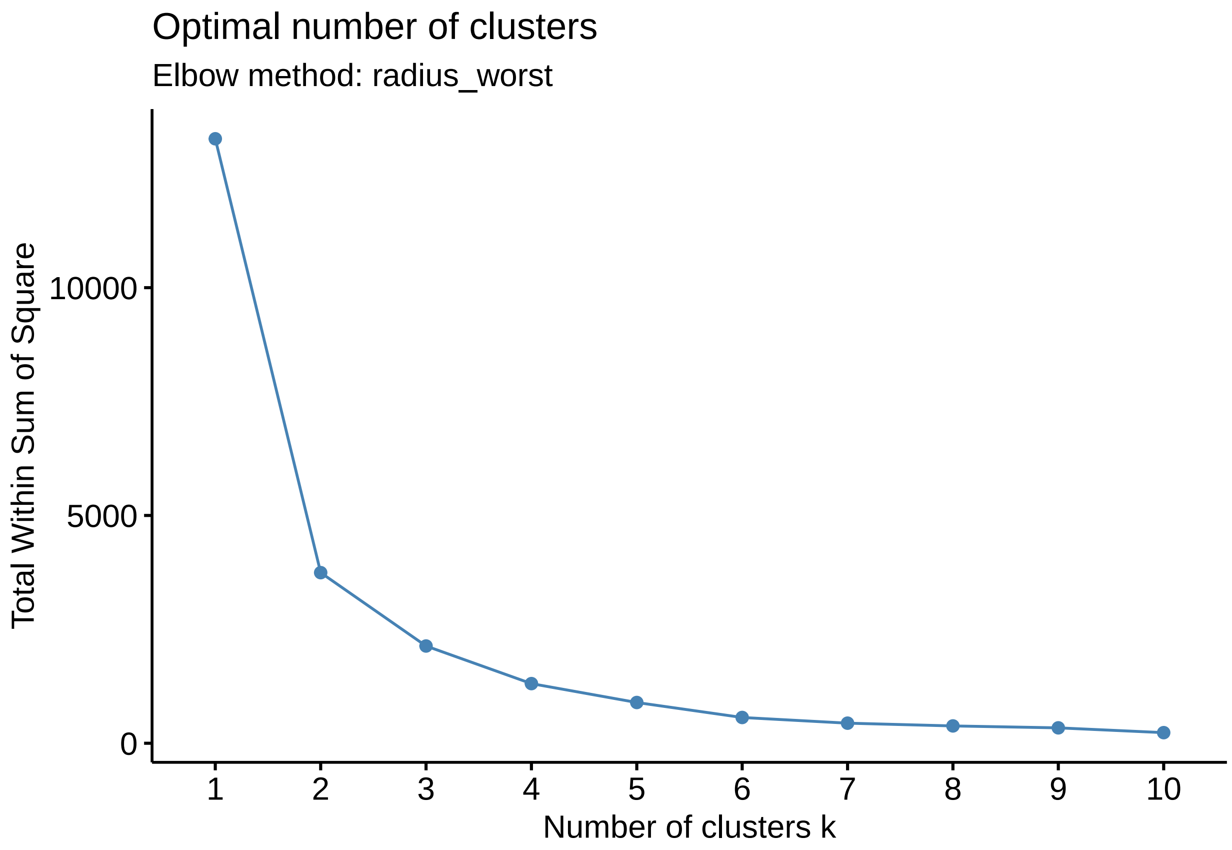
radius_worst

The radius_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_23 was created for the cluster results and a new_ID variable was made as well.

```r
# Select radius_worst
data_23 <- data %>%
  select(radius_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_23, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: radius_worst")
```



```
# Kmeans wss method
k_wss <- kmeans(data_23, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_23 <- data_23 %>%
  mutate(radius_worst_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

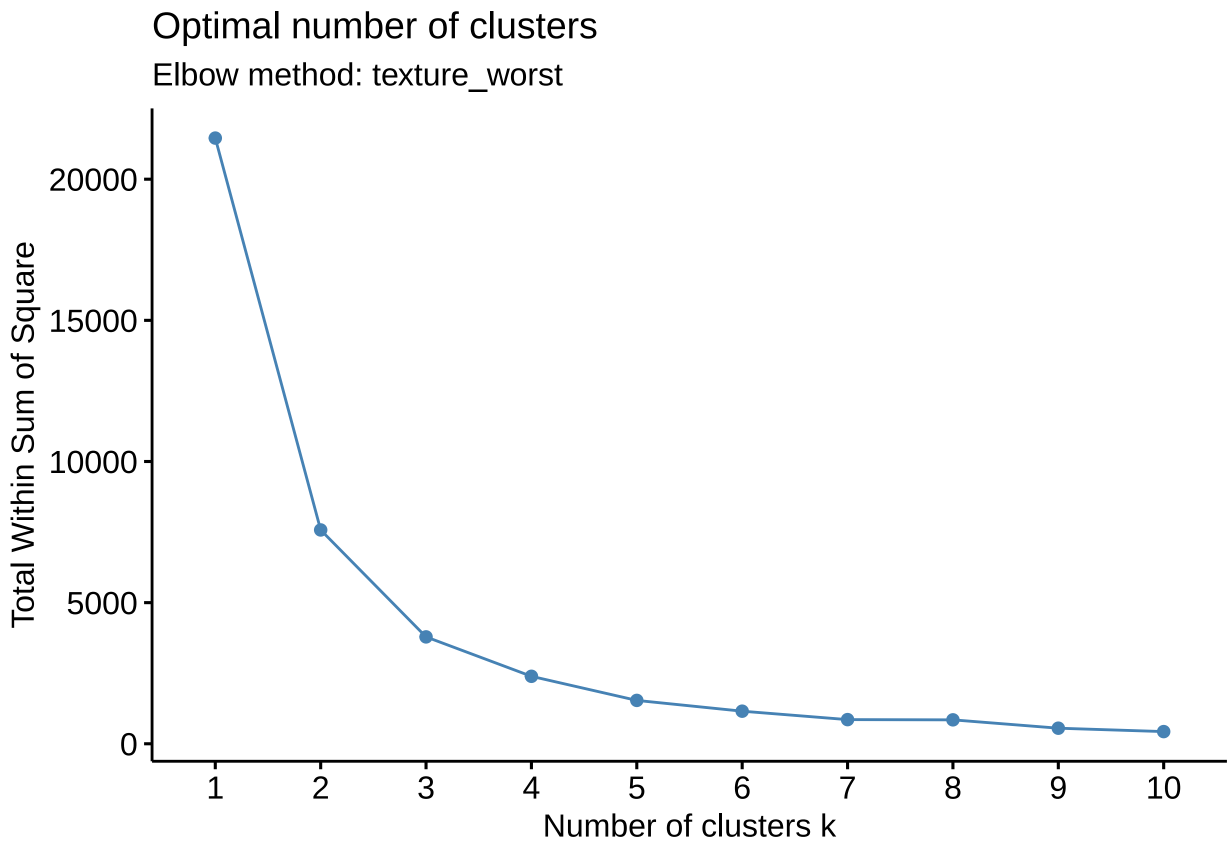
texture_worst

The texture_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_24 was created for the cluster results and a new_ID variable was made as well.

```
# Select texture_worst
data_24 <- data %>%
  select(texture_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_24, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: texture_worst")
```



```r
# Kmeans wss method

 k_wss<- kmeans(data_24, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_24 <- data_24 %>%
  mutate(texture_worst_wss = as.factor(k_wss$cluster),
          new_ID = seq(1:569))
```

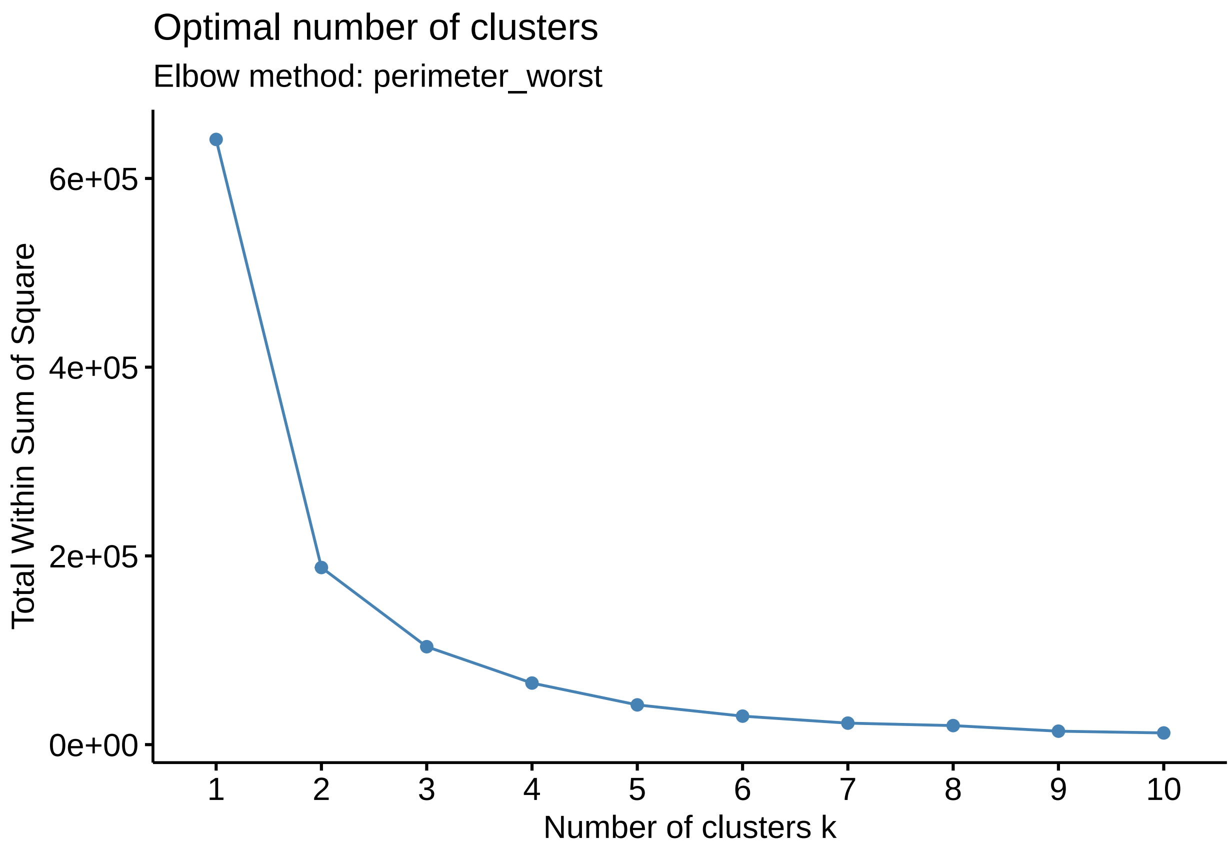
perimeter_worst

The perimeter_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_25 was created for the cluster results and a new_ID variable was made as well.

```r
# Select perimeter_worst
data_25 <- data %>%
  select(perimeter_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_25, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: perimeter_worst")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_25, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_25 <- data_25 %>%
  mutate(perimeter_worst_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

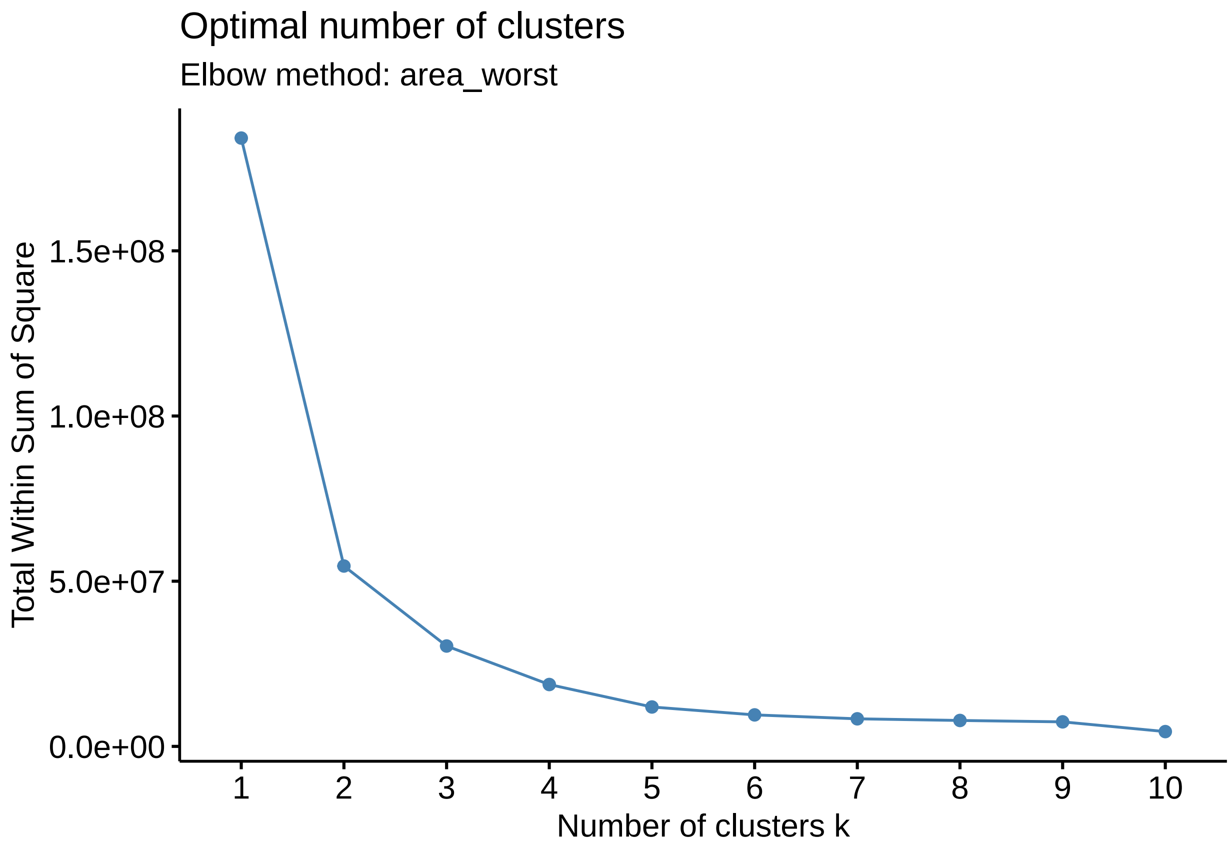
area_worst

The area_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_26 was created for the cluster results and a new_ID variable was made as well.

```r
# Select area_worst
data_26 <- data %>%
  select(area_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_26, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: area_worst")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_26, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_26 <- data_26 %>%
  mutate(area_worst_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

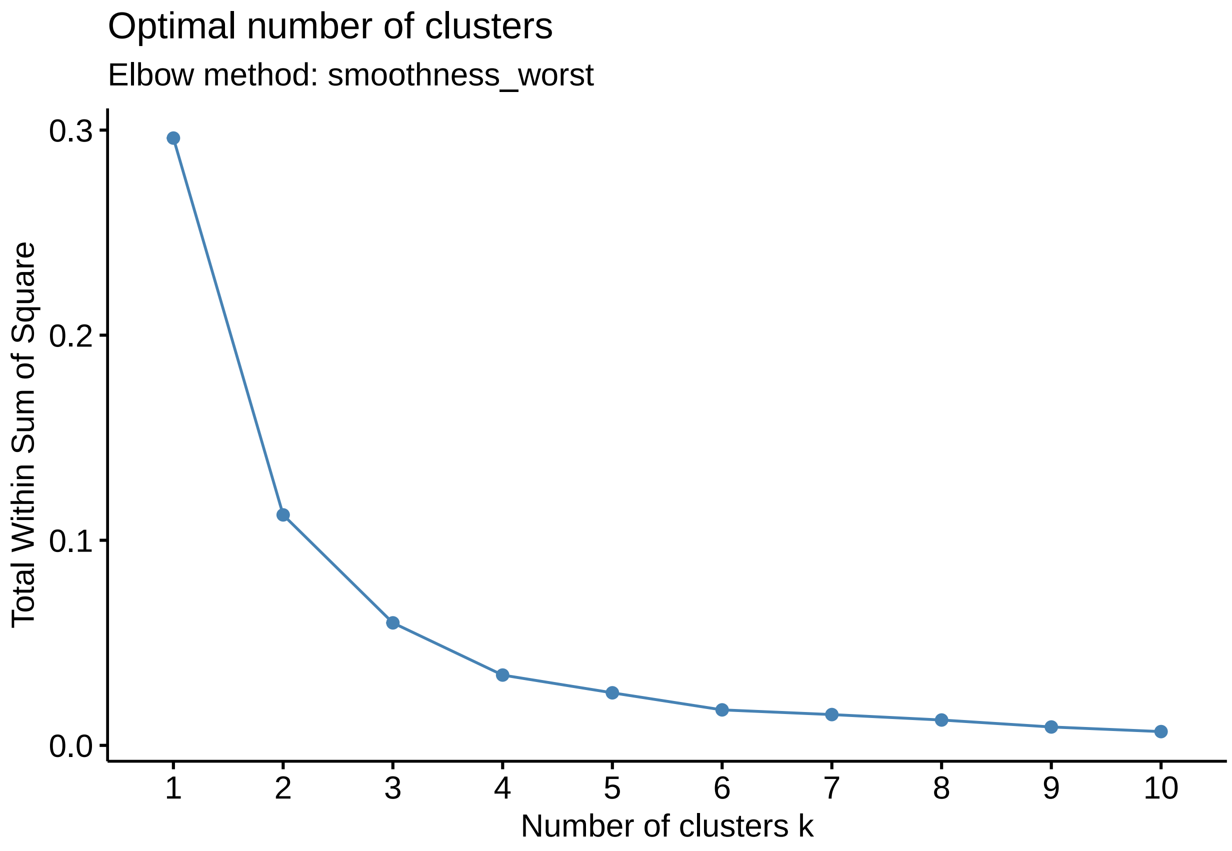
smoothness__worst

The smoothness_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_27 was created for the cluster results and a new_ID variable was made as well.

```r
# Select smoothness_worst
data_27 <- data %>%
  select(smoothness_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_27, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: smoothness_worst")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_27, centers = 4, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_27 <- data_27 %>%
  mutate(smoothness_worst_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

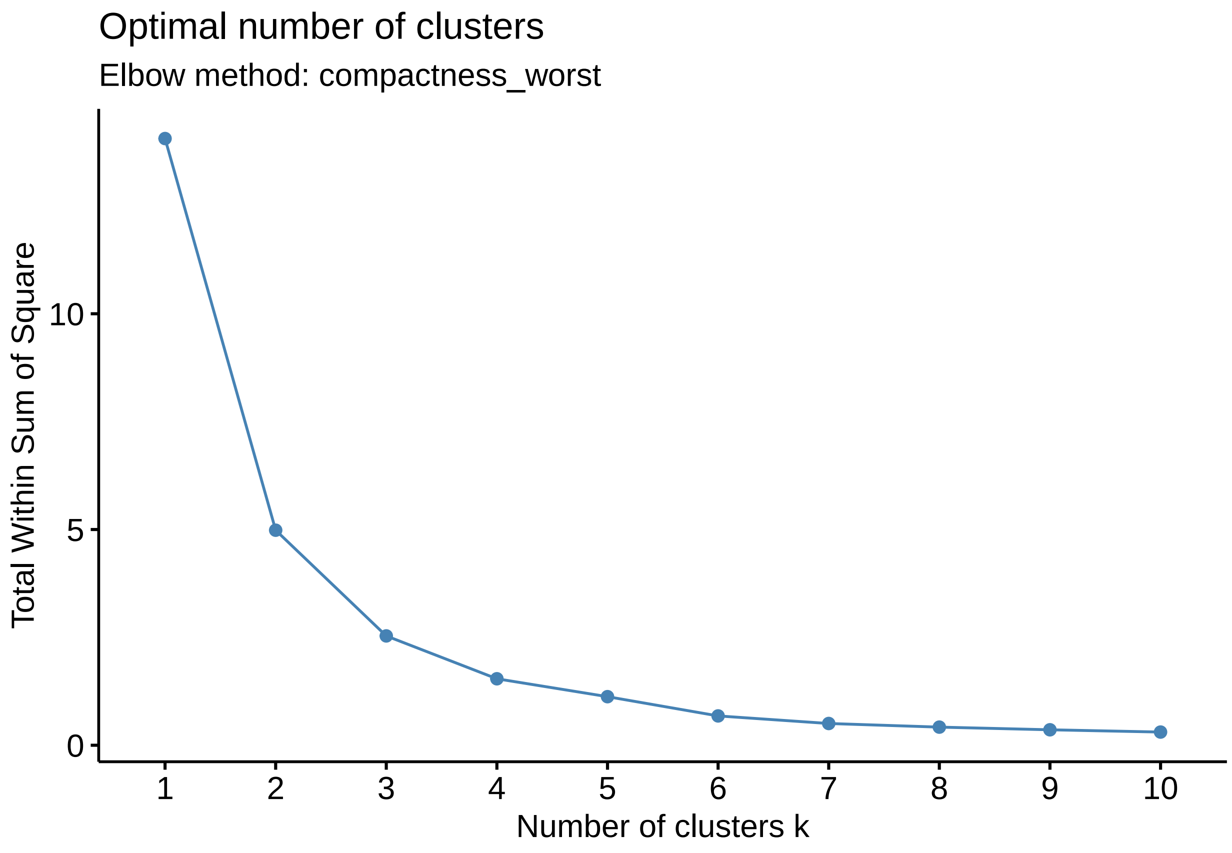
compactness_worst

The compactness_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_28 was created for the cluster results and a new_ID variable was made as well.

```r
# Select compactness_worst
data_28 <- data %>%
  select(compactness_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_28, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: compactness_worst")
```



```
# Kmeans wss method
k_wss <- kmeans(data_28, centers = 4, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_28 <- data_28 %>%
  mutate(compactness_worst_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

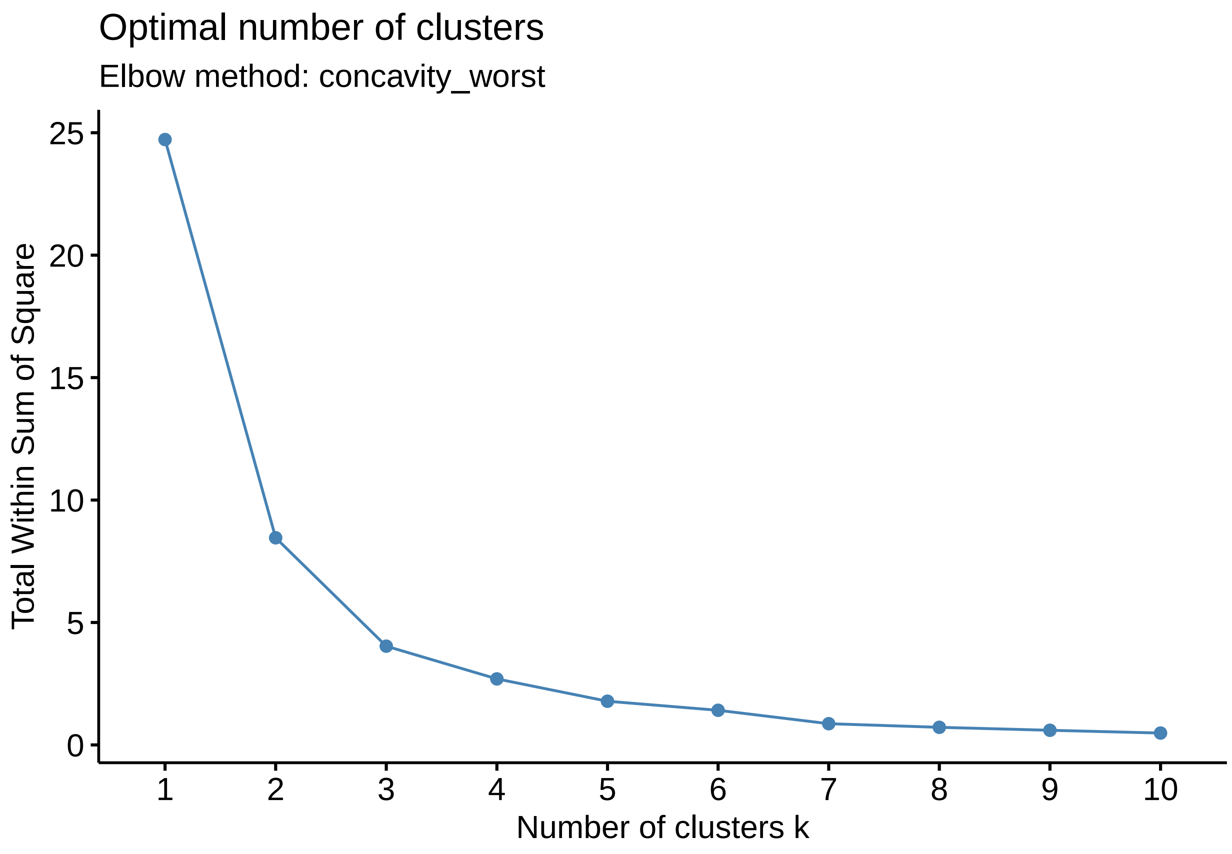
concavity_worst

The concavity_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=3 with 10 random starts. A new variable in data_29 was created for the cluster results and a new_ID variable was made as well.

```
# Select radius_worst
data_29 <- data %>%
  select(concavity_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_29, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: concavity_worst")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_29, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_29 <- data_29 %>%
  mutate(concavity_worst_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

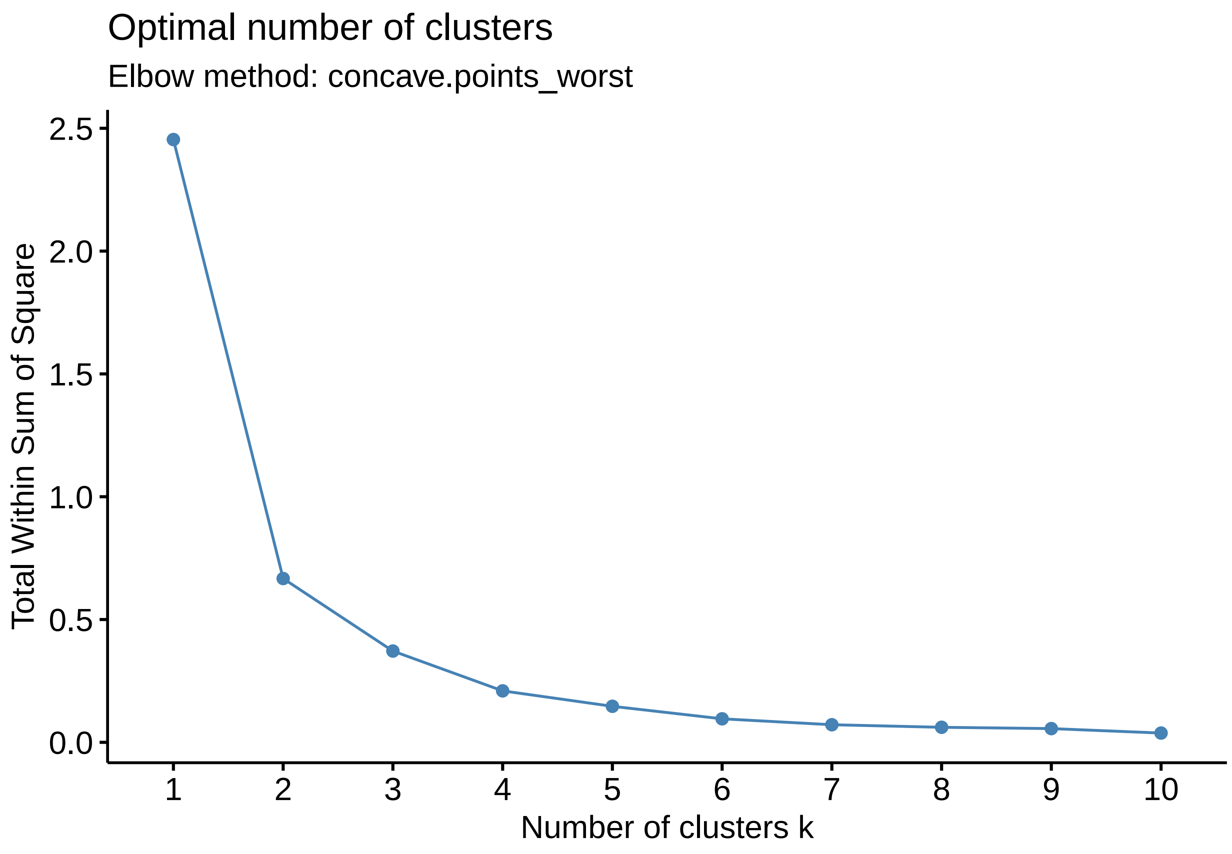
concave.points__worst

The concave.points__worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data__30 was created for the cluster results and a new__ID variable was made as well.

```r
# Select concave.points_worst
data_30 <- data %>%
  select(concave.points_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_30, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: concave.points_worst")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_30, centers = 4, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_30 <- data_30 %>%
  mutate(concave.points_worst_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

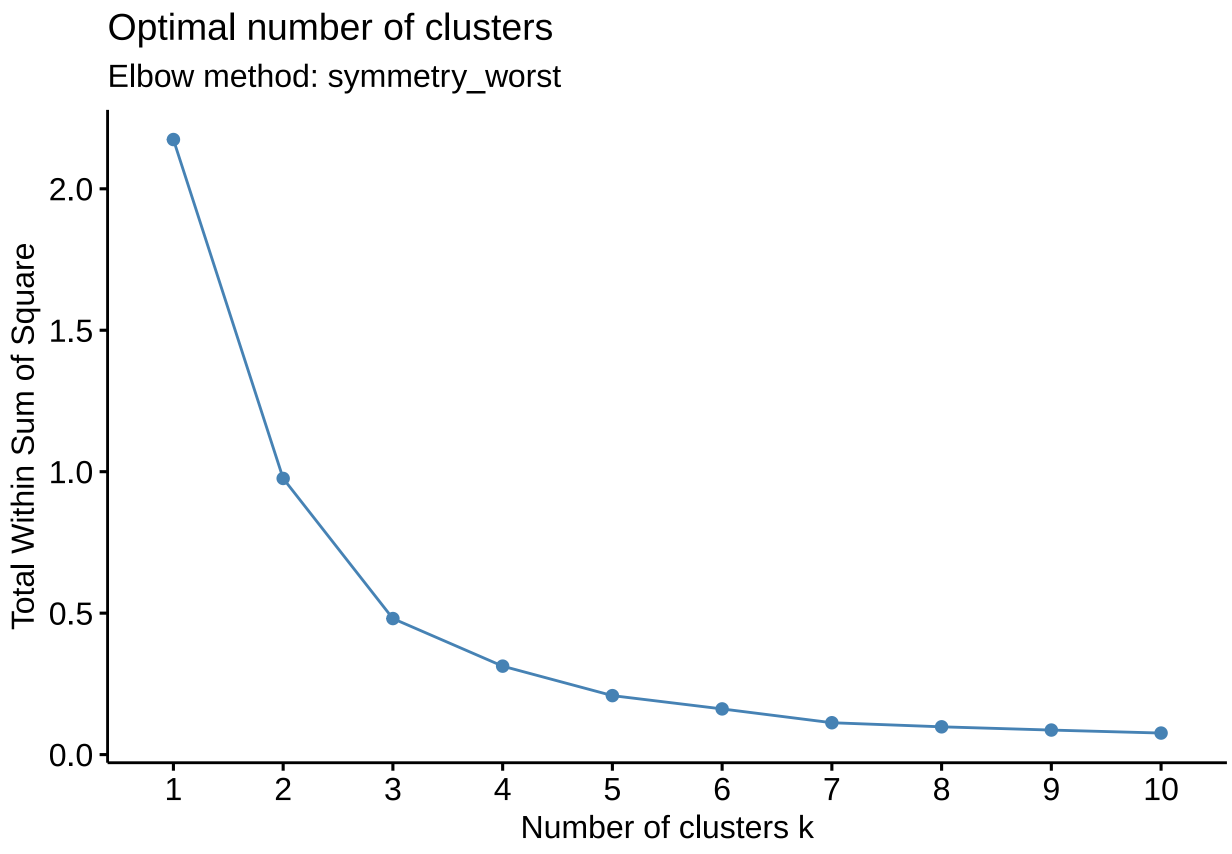
symmetry_worst

The symmetry_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_31 was created for the cluster results and a new_ID variable was made as well.

```r
# Select radius_worst
data_31 <- data %>%
  select(symmetry_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_31, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: symmetry_worst")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_31, centers = 3, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_31 <- data_31 %>%
  mutate(symmetry_worst_wss = as.factor(k_wss$cluster),
         new_ID = seq(1:569))
```

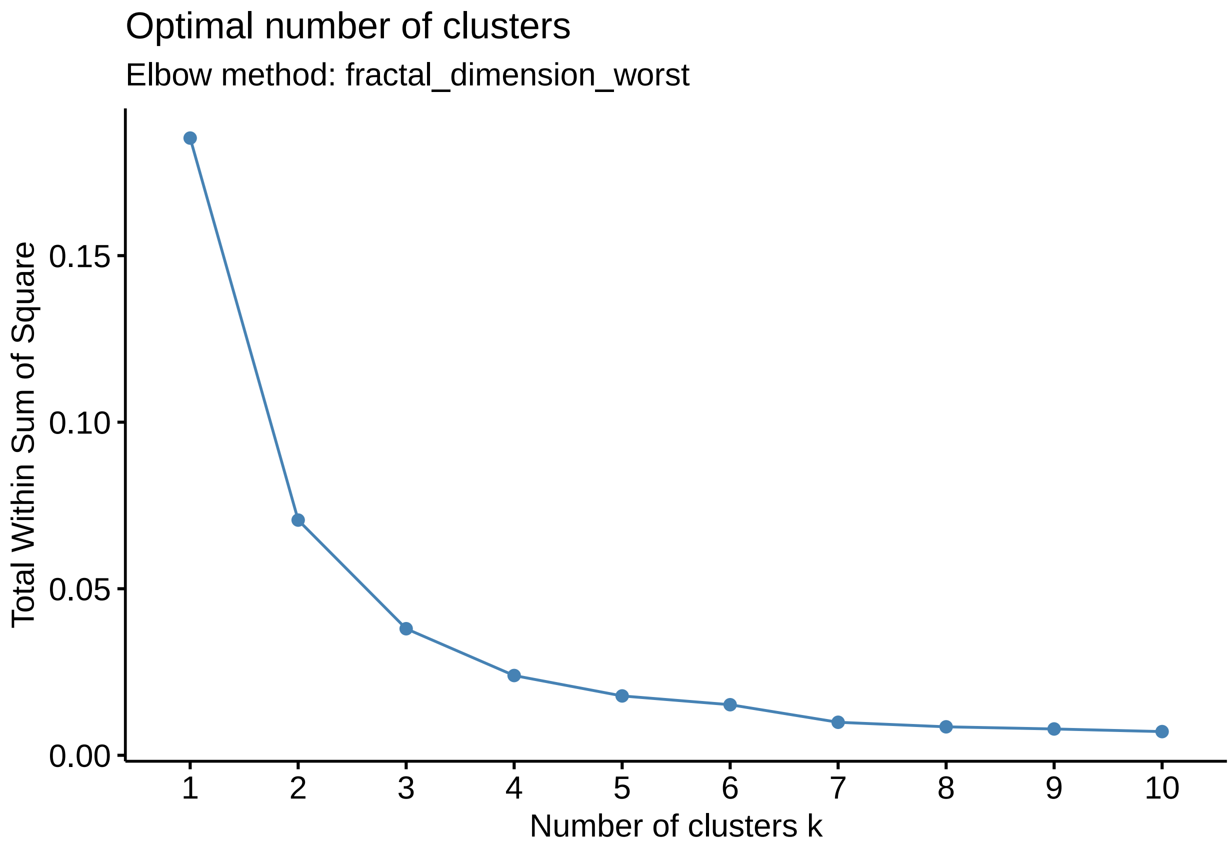
fractal_dimension_worst

The fractal_dimension_worst variable was selected and the seed set for reproduciblity. A graph was created for the different clusters using the within sum of squares for k=1 to k=10. The optimal k was found using the elbow method. The k-means algorithm was run using the optimal k=4 with 10 random starts. A new variable in data_32 was created for the cluster results and a new_ID variable was made as well.

```r
# Select radius_worst
data_32 <- data %>%
  select(fractal_dimension_worst)

# Set seed
set.seed(123)

# Graph for wws method
fviz_nbclust(data_32, kmeans, method = "wss") +
  labs(subtitle = "Elbow method: fractal_dimension_worst")
```



```r
# Kmeans wss method
k_wss <- kmeans(data_32, centers = 4, nstart = 10)

# Make new colums for the new variables generated from kmeans
data_32 <- data_32 %>%
  mutate(fractal_dimension_worst_wss = as.factor(k_wss$cluster),
          new_ID = seq(1:569))
```

Merge Datasets

Id, diagnosis and new_ID were selected and a merge function was made that allowed multiple data frames to be merged at the same time. The the different data frames were merged by the new function and the columns in data_new were rearranged to have all the origninal variables together and all the variables ending in wss together.

```r
#Select the first 2 row the data file
data_new <- data %>%
  select(c(1,2,33))

# Make a merge function
mymerge <- function(x, y){
  df <- merge(x, y, by= "new_ID", all.x= TRUE, all.y= TRUE)
  return(df)}

# Merge files
data_new <-Reduce( mymerge, list(data_new, data_3,data_4, data_5, data_6,
                               data_7, data_8, data_9, data_10, data_11,
                               data_12, data_13, data_14, data_15, data_16,
                               data_17, data_18, data_19, data_20, data_21,
                               data_22, data_23, data_24, data_25, data_26,
                               data_27, data_28, data_29, data_30, data_31, data_32))

# Arrange the columns
data_new <- data_new[c(1,2,3,4,6,8,10,12,14,16,18,20,
                       22,24,26,28,30,32,34,36,38,40,
                       42,44,46,48,50,52,54,56,58,60,62,
                       5,7,9,11,13,15,17,19,21,23,25,27,
                       29,31,33,35,37,39,41,43,45,47,
                       49,51,53,55,57,59,61,63)]
```

Scale Continous Variables

The continues variables were selected and scaled to have a mean of zero and a standard deviation of one. These variables were then removed leaving only the new scaled variables and the new_ID variable was created. data_new_scale was then merged with data_new by new_ID.

```r
# Rescale Variables to have a mean of 0 and standard deviation of 1
data_new_scale <-data_new %>%
  select(c(4:33)) %>%
  mutate_all(funs(scale = ((.)-mean(.))/sd(.))) %>%
  select(-c(1:30)) %>%
  mutate(new_ID = seq(c(1:569)))
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.
```

```r
# Merge data_new_scale with data_new
data_new <- merge(data_new, data_new_scale, by = "new_ID")
```

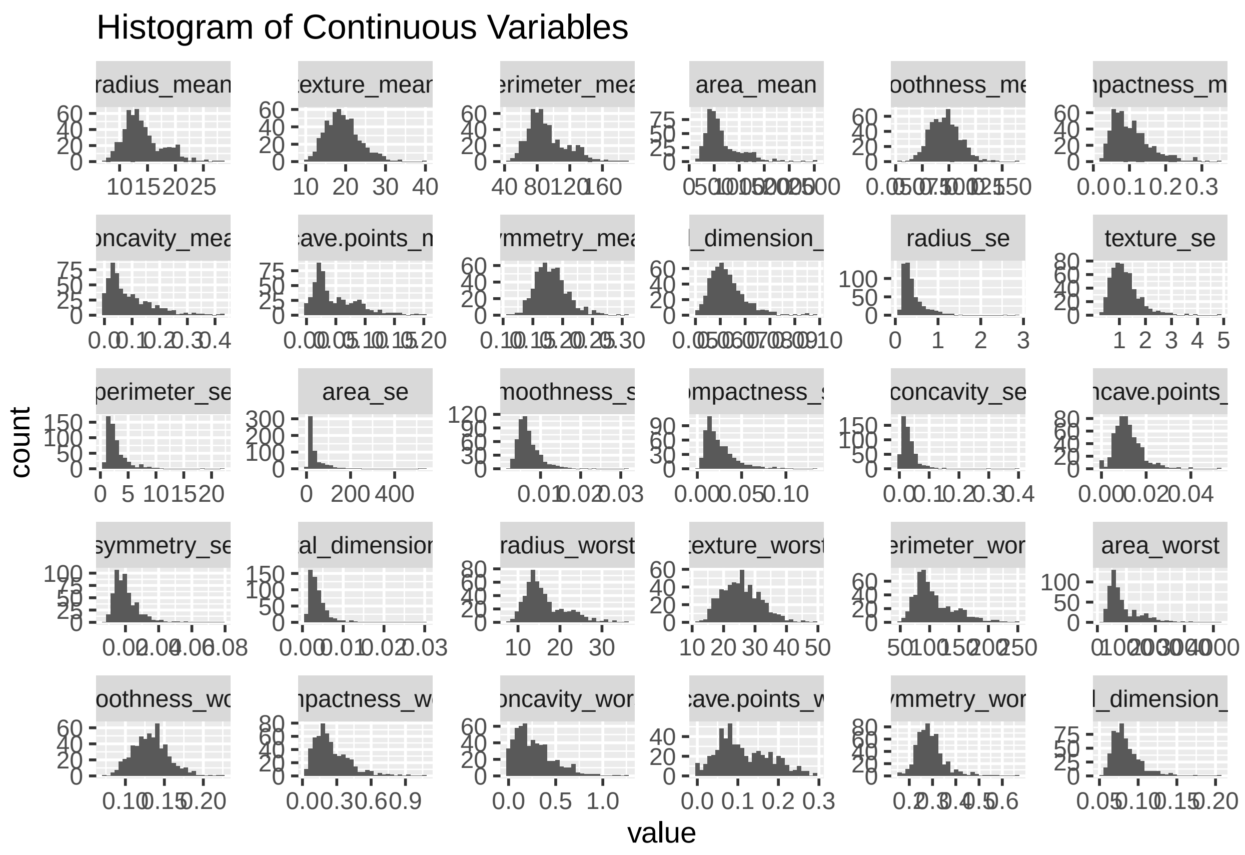
Transform Variables to Make Them Normal

The continuous variables were selected and converted to long form. A histogram was made of the variables to determine if they were were normally distributed. Some of the variables appear to not be normally distributed. The continuous variables were reselected, the variables to transformed were created and the original variables were removed. A Box Cox transformation was applied to these variables. The data was converted to long form again and a histogram of the variables was made. A few of the variables have outliers. Upon farther investigation, these outliers do not appear to be typos, but legitimate data and will be kept as part of the analysis. This may affect the results of the lda and qda models. new_ID variable was created and data_new_trans was merged with data_new by the new_ID variable.

```r
# Select the continous variables
data_new_trans <-data_new %>%
  select(c(4:33))

# Convert to long format
melt_df<-melt(data_new_trans)

# Histogram of all variables
ggplot(melt_df, aes(x=value)) +
  geom_histogram()+
  facet_wrap(~variable, scale="free")+
  labs(title = "Histogram of Continuous Variables")
```

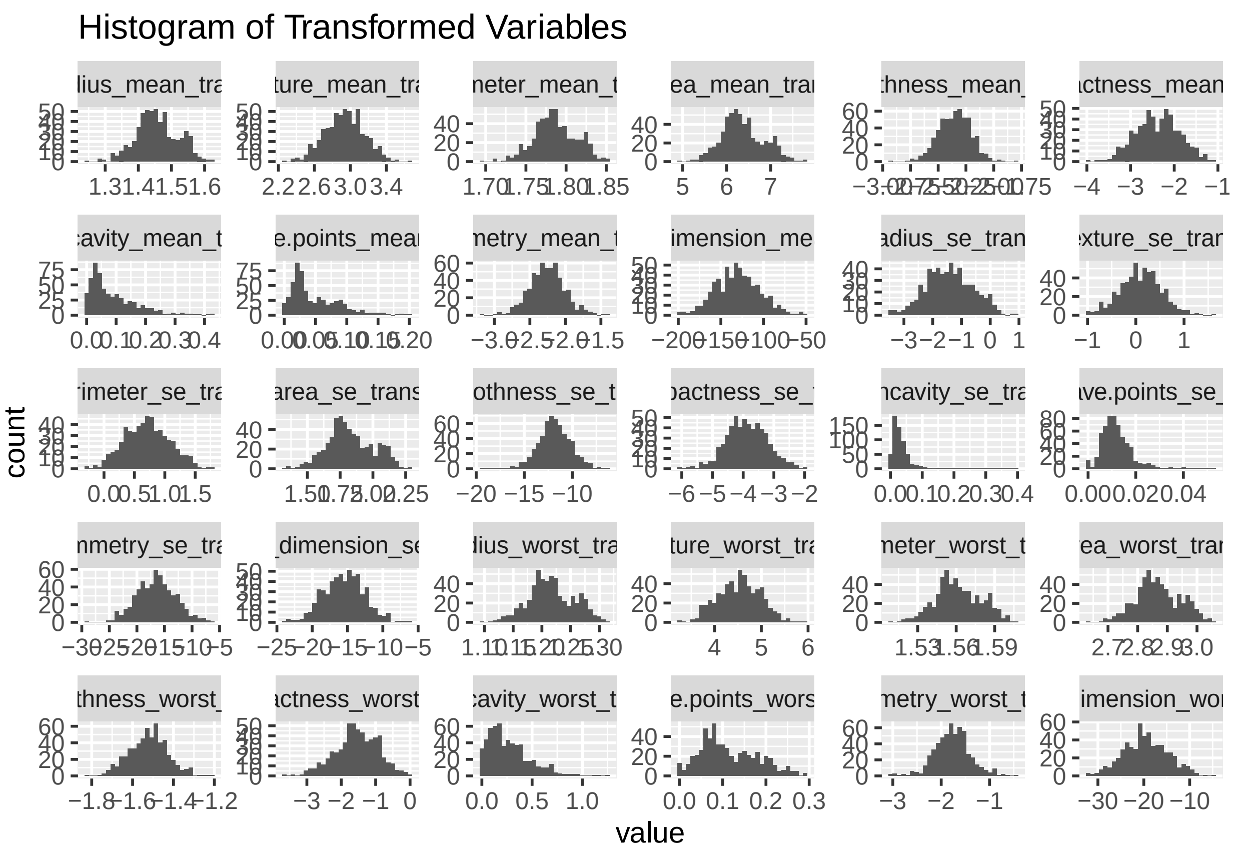


```r
# Select continous variables to be transformed
data_new_trans <-data_new %>%
  select(c(4:33)) %>%
  mutate_all(funs(trans = (.))) %>%
  select(-c(1:30))

# Apply Box Cox transformation
process_vars <- preProcess(data_new_trans, method = c("BoxCox"))
data_new_trans <- predict(process_vars, data_new_trans)

# Convert to long format
melt_df<-melt(data_new_trans)

# Histogram of transformed variables
ggplot(melt_df, aes(x=value)) +
  geom_histogram()+
  facet_wrap(~variable, scale="free")+
  labs(title = "Histogram of Transformed Variables")
```



```
# Create new variable to merge on
data_new_trans <- data_new_trans %>%
  mutate(new_ID = seq(c(1:569)))

# Merge data_new_trans with data_new
data_new <- merge(data_new, data_new_trans, by = "new_ID")
```

Variable Selection and Data Processing

Split Dataset With All Variables

A seed was set for reproducibility and the data was split 70:30 into training and test datasets.

```
# Set seed for reproducibility
set.seed(123)

# Split data into training and test datasets
index <- createDataPartition(data_new$diagnosis,
                             p = 0.7,
                             list = FALSE)

train_data <- data_new[index, ]
test_data <- data_new[-index, ]
```

Chi-square Test

The binned variables along with diagnosis were selected. A special function was made to run the Chi-square test on the all the variables at once. The statistically significant variable were selected. A seed was set for reproducibility and the dataset including only the variables selected by the chi-square test was split 70:30 into training and test datasets.

```
# Select the variable to be use in the Chi-square test
data_chisquare <- data_new %>%
  select(c(34:63,3))

# Run the Chi-square test
myTests <- lapply(data_chisquare[-length(data_chisquare)], function(x) chisq.test(table(as.factor(x), da
unlist(sapply(myTests, "[", "p.value"))
```

```
##             radius_mean_wss.p.value            texture_mean_wss.p.value
##                        4.511102e-65                        3.203777e-22
##          perimeter_mean_wss.p.value               area_mean_wss.p.value
##                        1.687340e-66                        1.023976e-66
##         smoothness_mean_wss.p.value        compactness_mean_wss.p.value
##                        4.052794e-16                        1.725177e-44
##          concavity_mean_wss.p.value     concave.points_mean_wss.p.value
##                        5.900187e-69                        1.174734e-84
##           symmetry_mean_wss.p.value  fractal_dimension_mean_wss.p.value
##                        3.373384e-16                        1.192653e-01
##               radius_se_wss.p.value              texture_se_wss.p.value
##                        2.393589e-45                        6.898377e-01
##            perimeter_se_wss.p.value                 area_se_wss.p.value
##                        3.767510e-41                        7.437912e-32
##           smoothness_se_wss.p.value          compactness_se_wss.p.value
##                        1.195935e-02                        1.282172e-15
##            concavity_se_wss.p.value       concave.points_se_wss.p.value
##                        1.083897e-02                        2.234633e-26
##             symmetry_se_wss.p.value    fractal_dimension_se_wss.p.value
##                        3.777166e-04                        1.804878e-03
##            radius_worst_wss.p.value           texture_worst_wss.p.value
##                        2.450295e-79                        3.480753e-27
##         perimeter_worst_wss.p.value              area_worst_wss.p.value
##                        9.066192e-82                        1.468752e-72
##        smoothness_worst_wss.p.value       compactness_worst_wss.p.value
##                        3.182831e-23                        4.994977e-45
##         concavity_worst_wss.p.value    concave.points_worst_wss.p.value
##                        1.140012e-60                        1.030915e-80
##          symmetry_worst_wss.p.value fractal_dimension_worst_wss.p.value
##                        1.103791e-20                        1.173370e-13
```

```
# Select the statistically significant variables at .001
data_chisquare <- data_new %>%
  select(-c(smoothness_se_wss, concavity_se_wss,
            symmetry_se_wss, fractal_dimension_mean_wss,
            texture_se_wss, fractal_dimension_se_wss,
            smoothness_se, concavity_se, symmetry_se,
            fractal_dimension_mean, texture_se, fractal_dimension_se,
            smoothness_se_scale, concavity_se_scale,
            symmetry_se_scale, fractal_dimension_mean_scale,
```

```r
            texture_se_scale, fractal_dimension_se_scale,
            smoothness_se_trans, concavity_se_trans,
            symmetry_se_trans, fractal_dimension_mean_trans,
            texture_se_trans, fractal_dimension_se_trans))

# Set seed for reproducibility
set.seed(123)

# Split data into training and test datasets
index <- createDataPartition(data_chisquare$diagnosis,
                      p = 0.7,
                      list = FALSE)

train_chisquare <- data_chisquare[index, ]
test_chisquare <- data_chisquare[-index, ]
```

Classification Tree

A seed was set for reproducibility, and a classification tree was run using the default settings of the rpart() function. The resulting tree was viewed and the variables that were selected by the tree were placed into a new dataset. A seed was set again for reproducibility and this dataset was partition 70:30 into training and test datasets.

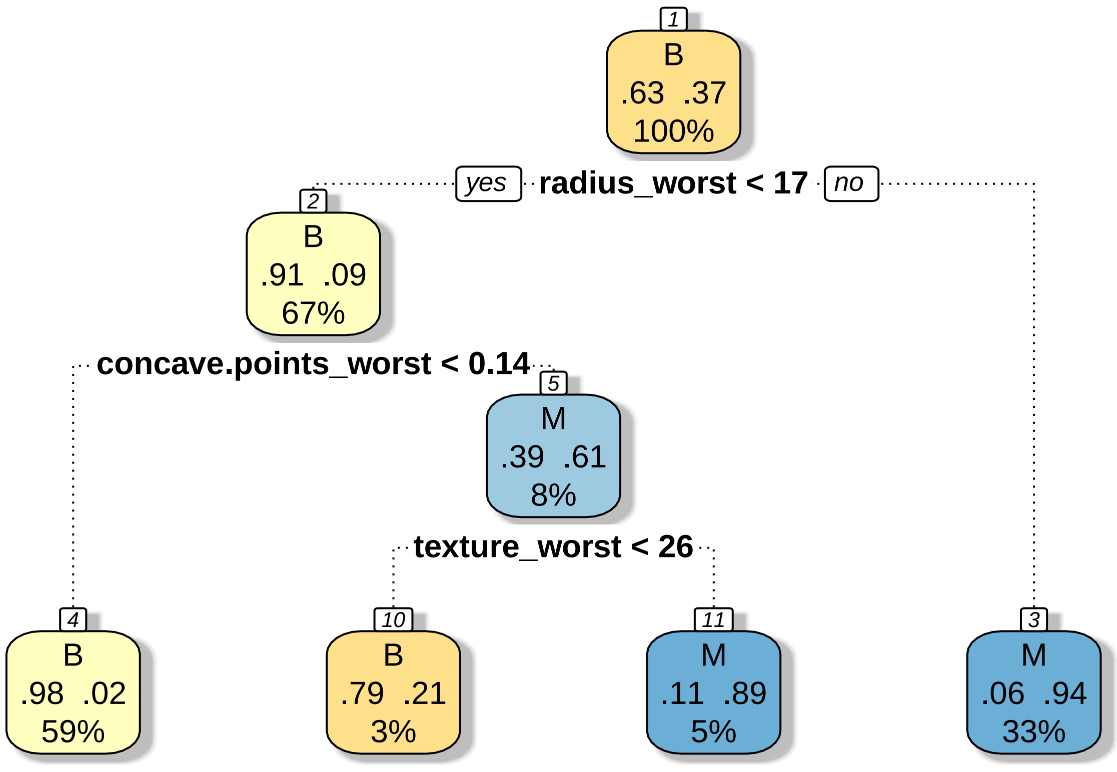
```r
# Set seed for reproducibility
set.seed(123)

# Fit the classification tree model
data_classtree <- rpart(diagnosis ~ ., data = data_new,
                      method = "class")

#plot tree
library(RColorBrewer)
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.3.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```r
fancyRpartPlot(data_classtree, palette="RdYlGn", sub=" ")
```



```
# View the variables selected
print(data_classtree)
```

```
## n= 569
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 569 212 B (0.62741652 0.37258348)
##    2) radius_worst< 16.795 379  33 B (0.91292876 0.08707124)
##      4) concave.points_worst< 0.1358 333   5 B (0.98498498 0.01501502) *
##      5) concave.points_worst>=0.1358 46  18 M (0.39130435 0.60869565)
##       10) texture_worst< 25.67 19   4 B (0.78947368 0.21052632) *
##       11) texture_worst>=25.67 27   3 M (0.11111111 0.88888889) *
##    3) radius_worst>=16.795 190  11 M (0.05789474 0.94210526) *
```

```
# Select the variables
data_classtree <- data_new %>%
  select(c(diagnosis, radius_worst, radius_worst_wss, radius_worst_scale, radius_worst_trans,
           concave.points_worst, concave.points_worst_wss, concave.points_worst_scale,
           concave.points_worst_trans, texture_worst, texture_worst_wss,
           texture_worst_scale, texture_worst_trans))

# Set seed for reproducibility
set.seed(123)

# Split data into training and test datasets
index <- createDataPartition(data_classtree$diagnosis,
                             p = 0.7,
                      list = FALSE)

train_classtree <- data_classtree[index, ]
test_classtree <- data_classtree[-index, ]
```

K-NN

All Variables

K-NN was run on all the variables. The scaled variables were selected and a seed set for reproducibility. The model was fit to the training dataset with 10-fold cross-validation. The resulting model was used to classify the type of cancer on the test dataset and a confusion matrix was generated.

```
# Select scaled variables
train_data_knn <- train_data %>%
  select(diagnosis, ends_with("scale"))

# Set seed for reproducibility
set.seed(123)

# Fit the model
knn_fit = train(diagnosis~.,
                      data = train_data_knn,
                      method = "knn",
                     trControl = trainControl(method = "cv", number = 10))

# Make predictions using the test dataset
knnPredict = predict(knn_fit, newdata = test_data)

# Confusion matrix
confusionMatrix(knnPredict, test_data$diagnosis, positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 106   6
##          M   1  57
##
##                Accuracy : 0.9588
##                  95% CI : (0.917, 0.9833)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9103
##
##  Mcnemar's Test P-Value : 0.1306
##
##             Sensitivity : 0.9048
##             Specificity : 0.9907
##          Pos Pred Value : 0.9828
##          Neg Pred Value : 0.9464
##              Prevalence : 0.3706
##          Detection Rate : 0.3353
##    Detection Prevalence : 0.3412
##       Balanced Accuracy : 0.9477
##
##        'Positive' Class : M
##
```

Chi-square Variables

K-NN was run on the variables selected by the Chi-square test. The scaled variables were selected and a seed set for reproducibility. The model was fit to the training dataset with 10-fold cross-validation. The resulting model was used to classify the type of cancer on the test dataset, and a confusion matrix was generated.

```r
# Select scaled variables
train_data_knn <- train_chisquare %>%
  select(diagnosis, ends_with("scale"))

# Set the seed for reproducibility
set.seed(123)

# Fit the model
knn_fit = train(diagnosis~.,
                     data = train_data_knn,
                     method = "knn",
                    trControl = trainControl(method = "cv", number = 10))

# Make predictions using the test dataset
knnPredict = predict(knn_fit, newdata = test_chisquare)

# Confusion matrix
confusionMatrix(knnPredict, test_chisquare$diagnosis, positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 104   8
##          M   3  55
##
##                Accuracy : 0.9353
##                  95% CI : (0.8872, 0.9673)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.859
##
##  Mcnemar's Test P-Value : 0.2278
##
##             Sensitivity : 0.8730
##             Specificity : 0.9720
##          Pos Pred Value : 0.9483
##          Neg Pred Value : 0.9286
##              Prevalence : 0.3706
##          Detection Rate : 0.3235
##    Detection Prevalence : 0.3412
##       Balanced Accuracy : 0.9225
##
##        'Positive' Class : M
##
```

Classification Tree Variables

K-NN was run on the variables selected by the classification tree. The scaled variables were selected and a seed set for reproducibility. The model was fit to the training dataset with 10-fold cross-validation. The resulting model was used to classify the type of cancer on the test dataset, and a confusion matrix was generated.

```
# Select scaled variables
train_data_knn <- train_classtree %>%
  select(diagnosis, ends_with("scale"))

# Set the seed for reproducibility
set.seed(123)

# Fit the model
knn_fit = train(diagnosis~.,
                      data = train_data_knn,
                      method = "knn",
                     trControl = trainControl(method = "cv", number = 10))

# Make predictions using the test dataset
knnPredict = predict(knn_fit, newdata = test_classtree)

# Confusion matrix
confusionMatrix(knnPredict, test_classtree$diagnosis, positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 103   4
##          M   4  59
##
##                Accuracy : 0.9529
##                  95% CI : (0.9094, 0.9795)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8991
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9365
##             Specificity : 0.9626
##          Pos Pred Value : 0.9365
##          Neg Pred Value : 0.9626
##              Prevalence : 0.3706
##          Detection Rate : 0.3471
##    Detection Prevalence : 0.3706
##       Balanced Accuracy : 0.9496
##
##        'Positive' Class : M
##
```

LDA

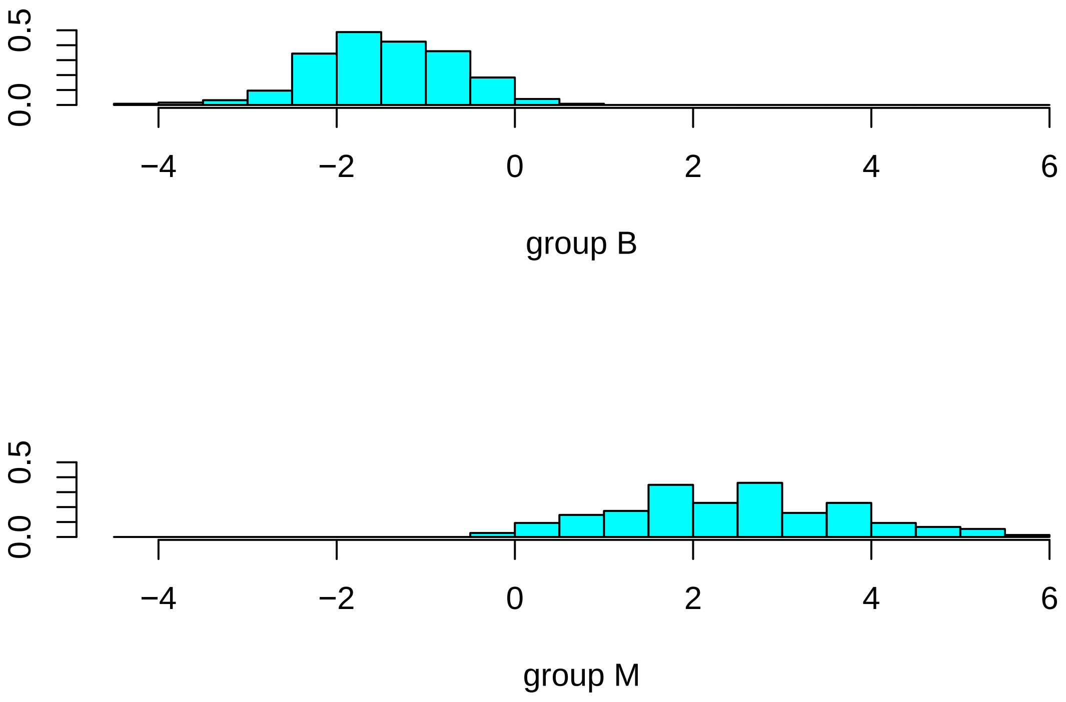
All Variables

Linear discriminant analysis was run on all the variables. The transformed variables were selected, and the model was fit to the training dataset. A stacked histogram was made in order to understand the model's ability to separate malignant and benign cancer. The resulting model was used to classify the type of cancer on the test dataset, and a confusion matrix was generated.

```
# Select the transformed variables
train_new_trans_all <- train_data %>%
  select(diagnosis, ends_with("trans"))

# Fit the linear discriminant analysis model
lda_all = lda(diagnosis~., data = train_new_trans_all)

# Make a stacked histogram
lda_predict_all = predict(lda_all)
ldahist(data = lda_predict_all$x[,1], g=train_new_trans_all$diagnosis)
```



```
# Make prediction using the fitted model on test data
pred_test = predict(lda_all, test_data)

#Confustion matirx
confusionMatrix(as.factor(pred_test$class), as.factor(test_data$diagnosis), positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 107   5
##          M   0  58
##
##                Accuracy : 0.9706
##                  95% CI : (0.9327, 0.9904)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9359
##
##  Mcnemar's Test P-Value : 0.07364
##
##             Sensitivity : 0.9206
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9554
##              Prevalence : 0.3706
##          Detection Rate : 0.3412
##    Detection Prevalence : 0.3412
##       Balanced Accuracy : 0.9603
##
##        'Positive' Class : M
##
```

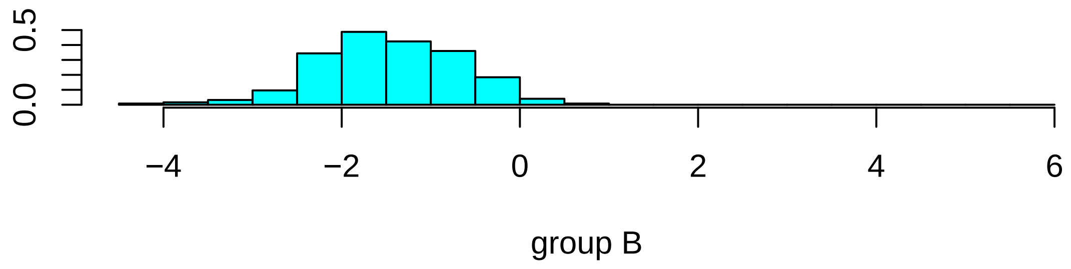
Chi-square Variables

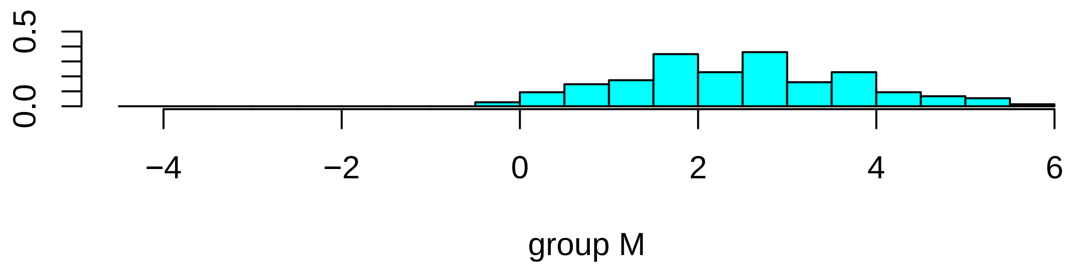
Linear discrimant analysis was run on the variables selected by the Chi-square test. The transformed variables were selected, and the model was fit to the training dataset. A stacked histogram was made in order to understand the model's ability to separate malignant and benign cancer. The resulting model was used to classify the type of cancer on the test dataset, and a confusion matrix was generated.

```
# Select the transformed variables
train_data_lda <- train_chisquare %>%
  select(diagnosis, ends_with("trans"))

# Fit the linear discriminant analysis model
lda_chi = lda(diagnosis~., data = train_data_lda)

# Make a stacked histogram
lda_predict_chi = predict(lda_chi)
ldahist(data = lda_predict_all$x[,1], g=train_data_lda$diagnosis)
```





```r
# Make prediction using the fitted model on test data
pred_test = predict(lda_chi, test_chisquare)

#Confustion matirx
confusionMatrix(as.factor(pred_test$class), as.factor(test_chisquare$diagnosis), positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 105   6
##          M   2  57
##
##                Accuracy : 0.9529
##                  95% CI : (0.9094, 0.9795)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8978
##
##  Mcnemar's Test P-Value : 0.2888
##
##             Sensitivity : 0.9048
##             Specificity : 0.9813
##          Pos Pred Value : 0.9661
##          Neg Pred Value : 0.9459
##              Prevalence : 0.3706
##          Detection Rate : 0.3353
##    Detection Prevalence : 0.3471
##       Balanced Accuracy : 0.9430
##
##        'Positive' Class : M
```

```
##
```

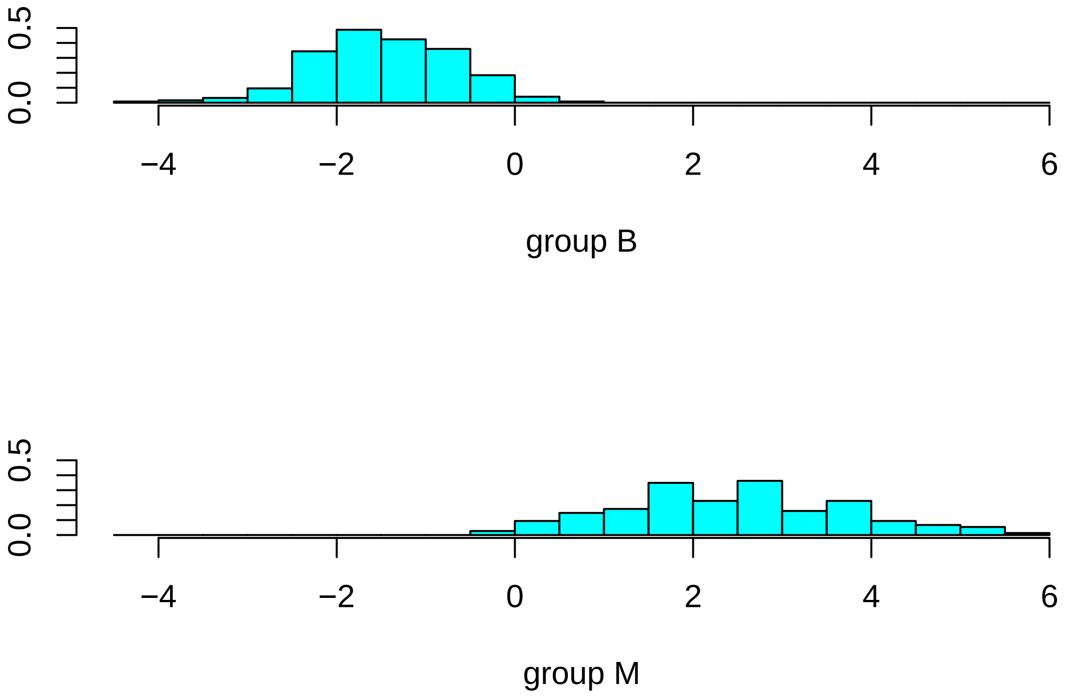
Classification Tree Variables

Linear discriminant analysis was run on the variables selected by the classification tree. The transformed variables were selected, and the model was fit to the training dataset. A stacked histogram was made in order to understand the model's ability to separate malignant and benign cancer. The resulting model was used to classify the type of cancer on the test dataset, and a confusion matrix was generated.

```
# Select the transformed variables
train_data_lda <- train_classtree %>%
  select(diagnosis, ends_with("trans"))

# Fit the linear discriminant analysis model
lda_chi = lda(diagnosis~., data = train_data_lda)

# Make a stacked histogram
lda_predict_chi = predict(lda_chi)
ldahist(data = lda_predict_all$x[,1], g=train_data_lda$diagnosis)
```



```
# Make prediction using the fitted model on test data
pred_test = predict(lda_chi, test_classtree)

#Confustion matirx
confusionMatrix(as.factor(pred_test$class),
                as.factor(test_classtree$diagnosis), positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 104   5
##          M   3  58
## 
##                Accuracy : 0.9529          
##                  95% CI : (0.9094, 0.9795)
##     No Information Rate : 0.6294          
##     P-Value [Acc > NIR] : <2e-16          
## 
##                   Kappa : 0.8985          
## 
##  Mcnemar's Test P-Value : 0.7237          
## 
##             Sensitivity : 0.9206          
##             Specificity : 0.9720          
##          Pos Pred Value : 0.9508          
##          Neg Pred Value : 0.9541          
##              Prevalence : 0.3706          
##          Detection Rate : 0.3412          
##    Detection Prevalence : 0.3588          
##       Balanced Accuracy : 0.9463          
## 
##        'Positive' Class : M               
## 
```

QDA

All Variables

Quadratic discriminant analysis was run on all the variables. The transformed variables were selected, and the model was fit to the training dataset. The resulting model was used to classify the type of cancer on the test dataset, and a confusion matrix was generated.

```r
# Select the transformed variables
train_new_trans_all <- train_data %>%
  select(diagnosis, ends_with("trans"))

# Fit the linear discriminant analysis model
qda_all = qda(diagnosis~., data = train_new_trans_all)

# Make prediction using the fitted model on test data
pred_test = predict(qda_all, test_data)

#Confustion matirx
confusionMatrix(as.factor(pred_test$class), as.factor(test_data$diagnosis), positive = "M")
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   B   M
##          B 101   5
##          M   6  58
## 
##                Accuracy : 0.9353          
##                  95% CI : (0.8872, 0.9673)
##     No Information Rate : 0.6294          
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8617
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9206
##             Specificity : 0.9439
##          Pos Pred Value : 0.9062
##          Neg Pred Value : 0.9528
##              Prevalence : 0.3706
##          Detection Rate : 0.3412
##    Detection Prevalence : 0.3765
##       Balanced Accuracy : 0.9323
##
##        'Positive' Class : M
##
```

Chi-square Variables

Quadratic discriminant analysis was run on the variables selected by the Chi-square test. The transformed variables from the Chi-square test were selected, and the model was fit to the training dataset. The resulting model was used to classify the type of cancer on the test dataset, and a confusion matrix was generated.

```r
# Select the transformed variables
train_data_qda <- train_chisquare %>%
  select(diagnosis, ends_with("trans"))

# Fit the linear discriminant analysis model
qda_chi = qda(diagnosis~., data = train_data_qda)

# Make prediction using the fitted model on test data
pred_test = predict(qda_chi, test_chisquare)

#Confustion matirx
confusionMatrix(as.factor(pred_test$class), as.factor(test_chisquare$diagnosis), positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 104   4
##          M   3  59
##
##                Accuracy : 0.9588
##                  95% CI : (0.917, 0.9833)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9114
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9365
##             Specificity : 0.9720
##          Pos Pred Value : 0.9516
##          Neg Pred Value : 0.9630
##              Prevalence : 0.3706
##          Detection Rate : 0.3471
##    Detection Prevalence : 0.3647
##       Balanced Accuracy : 0.9542
##
##        'Positive' Class : M
##
```

Classification Tree Variables

Quadratic discriminant analysis was run on the variables selected by the classification tree. The transformed variables were selected, and the model was fit to the training dataset. The resulting model was used to classify the type of cancer on the test dataset, and a confusion matrix was generated.

```
# Select the transformed variables
train_data_qda <- train_classtree %>%
  select(diagnosis, ends_with("trans"))

# Fit the linear discriminant analysis model
qda_chi = qda(diagnosis~., data = train_data_qda)


# Make prediction using the fitted model on test data
pred_test = predict(qda_chi, test_classtree)

#Confustion matirx
confusionMatrix(as.factor(pred_test$class),
                as.factor(test_classtree$diagnosis), positive = "M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 104   4
##          M   3  59
##
##                Accuracy : 0.9588
##                  95% CI : (0.917, 0.9833)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9114
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9365
##             Specificity : 0.9720
##          Pos Pred Value : 0.9516
##          Neg Pred Value : 0.9630
##              Prevalence : 0.3706
##          Detection Rate : 0.3471
##    Detection Prevalence : 0.3647
##       Balanced Accuracy : 0.9542
##
##        'Positive' Class : M
##
```