Final Project

```
setwd("~/NotreDame/IntroDataScience/Final")
```

```
FNA <- read.csv("FNA_cancer.csv", header = T)
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(rpart)
library(partykit)
```

```
## Loading required package: grid

## Loading required package: libcoin

## Warning: package 'libcoin' was built under R version 4.0.4

## Loading required package: mvtnorm

## Warning: package 'mvtnorm' was built under R version 4.0.3
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.4

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(class)
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.4

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
glimpse(FNA)
```

```
## Rows: 569
## Columns: 33
## $ id                      <int> 842302, 842517, 84300903, 84348301, 8435840...
## $ diagnosis               <chr> "M", "M", "M", "M", "M", "M", "M", "M", "M"...
## $ radius_mean             <dbl> 17.990, 20.570, 19.690, 11.420, 20.290, 12....
## $ texture_mean            <dbl> 10.38, 17.77, 21.25, 20.38, 14.34, 15.70, 1...
## $ perimeter_mean          <dbl> 122.80, 132.90, 130.00, 77.58, 135.10, 82.5...
## $ area_mean               <dbl> 1001.0, 1326.0, 1203.0, 386.1, 1297.0, 477....
## $ smoothness_mean         <dbl> 0.11840, 0.08474, 0.10960, 0.14250, 0.10030...
## $ compactness_mean        <dbl> 0.27760, 0.07864, 0.15990, 0.28390, 0.13280...
## $ concavity_mean          <dbl> 0.30010, 0.08690, 0.19740, 0.24140, 0.19800...
## $ concave.points_mean     <dbl> 0.14710, 0.07017, 0.12790, 0.10520, 0.10430...
## $ symmetry_mean           <dbl> 0.2419, 0.1812, 0.2069, 0.2597, 0.1809, 0.2...
## $ fractal_dimension_mean  <dbl> 0.07871, 0.05667, 0.05999, 0.09744, 0.05883...
## $ radius_se               <dbl> 1.0950, 0.5435, 0.7456, 0.4956, 0.7572, 0.3...
## $ texture_se              <dbl> 0.9053, 0.7339, 0.7869, 1.1560, 0.7813, 0.8...
## $ perimeter_se            <dbl> 8.589, 3.398, 4.585, 3.445, 5.438, 2.217, 3...
## $ area_se                 <dbl> 153.40, 74.08, 94.03, 27.23, 94.44, 27.19, ...
## $ smoothness_se           <dbl> 0.006399, 0.005225, 0.006150, 0.009110, 0.0...
## $ compactness_se          <dbl> 0.049040, 0.013080, 0.040060, 0.074580, 0.0...
## $ concavity_se            <dbl> 0.05373, 0.01860, 0.03832, 0.05661, 0.05688...
## $ concave.points_se       <dbl> 0.015870, 0.013400, 0.020580, 0.018670, 0.0...
## $ symmetry_se             <dbl> 0.03003, 0.01389, 0.02250, 0.05963, 0.01756...
## $ fractal_dimension_se    <dbl> 0.006193, 0.003532, 0.004571, 0.009208, 0.0...
## $ radius_worst            <dbl> 25.38, 24.99, 23.57, 14.91, 22.54, 15.47, 2...
## $ texture_worst           <dbl> 17.33, 23.41, 25.53, 26.50, 16.67, 23.75, 2...
## $ perimeter_worst         <dbl> 184.60, 158.80, 152.50, 98.87, 152.20, 103....
```

```
## $ area_worst              <dbl> 2019.0, 1956.0, 1709.0, 567.7, 1575.0, 741....
## $ smoothness_worst        <dbl> 0.1622, 0.1238, 0.1444, 0.2098, 0.1374, 0.1...
## $ compactness_worst       <dbl> 0.6656, 0.1866, 0.4245, 0.8663, 0.2050, 0.5...
## $ concavity_worst         <dbl> 0.71190, 0.24160, 0.45040, 0.68690, 0.40000...
## $ concave.points_worst    <dbl> 0.26540, 0.18600, 0.24300, 0.25750, 0.16250...
## $ symmetry_worst          <dbl> 0.4601, 0.2750, 0.3613, 0.6638, 0.2364, 0.3...
## $ fractal_dimension_worst <dbl> 0.11890, 0.08902, 0.08758, 0.17300, 0.07678...
## $ X                       <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
```

Make diagnosis a factor and remove last column 'X'

```
FNA$diagnosis <- as.factor(FNA$diagnosis)
FNA <- FNA %>% dplyr::select(-X)
glimpse(FNA)
```

```
## Rows: 569
## Columns: 32
## $ id                      <int> 842302, 842517, 84300903, 84348301, 8435840...
## $ diagnosis               <fct> M, M, M, M, M, M, M, M, M, M, M, M, M, M, M...
## $ radius_mean             <dbl> 17.990, 20.570, 19.690, 11.420, 20.290, 12....
## $ texture_mean            <dbl> 10.38, 17.77, 21.25, 20.38, 14.34, 15.70, 1...
## $ perimeter_mean          <dbl> 122.80, 132.90, 130.00, 77.58, 135.10, 82.5...
## $ area_mean               <dbl> 1001.0, 1326.0, 1203.0, 386.1, 1297.0, 477....
## $ smoothness_mean         <dbl> 0.11840, 0.08474, 0.10960, 0.14250, 0.10030...
## $ compactness_mean        <dbl> 0.27760, 0.07864, 0.15990, 0.28390, 0.13280...
## $ concavity_mean          <dbl> 0.30010, 0.08690, 0.19740, 0.24140, 0.19800...
## $ concave.points_mean     <dbl> 0.14710, 0.07017, 0.12790, 0.10520, 0.10430...
## $ symmetry_mean           <dbl> 0.2419, 0.1812, 0.2069, 0.2597, 0.1809, 0.2...
## $ fractal_dimension_mean  <dbl> 0.07871, 0.05667, 0.05999, 0.09744, 0.05883...
## $ radius_se               <dbl> 1.0950, 0.5435, 0.7456, 0.4956, 0.7572, 0.3...
## $ texture_se              <dbl> 0.9053, 0.7339, 0.7869, 1.1560, 0.7813, 0.8...
## $ perimeter_se            <dbl> 8.589, 3.398, 4.585, 3.445, 5.438, 2.217, 3...
## $ area_se                 <dbl> 153.40, 74.08, 94.03, 27.23, 94.44, 27.19, ...
## $ smoothness_se           <dbl> 0.006399, 0.005225, 0.006150, 0.009110, 0.0...
## $ compactness_se          <dbl> 0.049040, 0.013080, 0.040060, 0.074580, 0.0...
## $ concavity_se            <dbl> 0.05373, 0.01860, 0.03832, 0.05661, 0.05688...
## $ concave.points_se       <dbl> 0.015870, 0.013400, 0.020580, 0.018670, 0.0...
## $ symmetry_se             <dbl> 0.03003, 0.01389, 0.02250, 0.05963, 0.01756...
## $ fractal_dimension_se    <dbl> 0.006193, 0.003532, 0.004571, 0.009208, 0.0...
## $ radius_worst            <dbl> 25.38, 24.99, 23.57, 14.91, 22.54, 15.47, 2...
## $ texture_worst           <dbl> 17.33, 23.41, 25.53, 26.50, 16.67, 23.75, 2...
## $ perimeter_worst         <dbl> 184.60, 158.80, 152.50, 98.87, 152.20, 103....
## $ area_worst              <dbl> 2019.0, 1956.0, 1709.0, 567.7, 1575.0, 741....
## $ smoothness_worst        <dbl> 0.1622, 0.1238, 0.1444, 0.2098, 0.1374, 0.1...
## $ compactness_worst       <dbl> 0.6656, 0.1866, 0.4245, 0.8663, 0.2050, 0.5...
## $ concavity_worst         <dbl> 0.71190, 0.24160, 0.45040, 0.68690, 0.40000...
## $ concave.points_worst    <dbl> 0.26540, 0.18600, 0.24300, 0.25750, 0.16250...
## $ symmetry_worst          <dbl> 0.4601, 0.2750, 0.3613, 0.6638, 0.2364, 0.3...
## $ fractal_dimension_worst <dbl> 0.11890, 0.08902, 0.08758, 0.17300, 0.07678...
```

Check to see difference between benign versus malignant diagnosis.

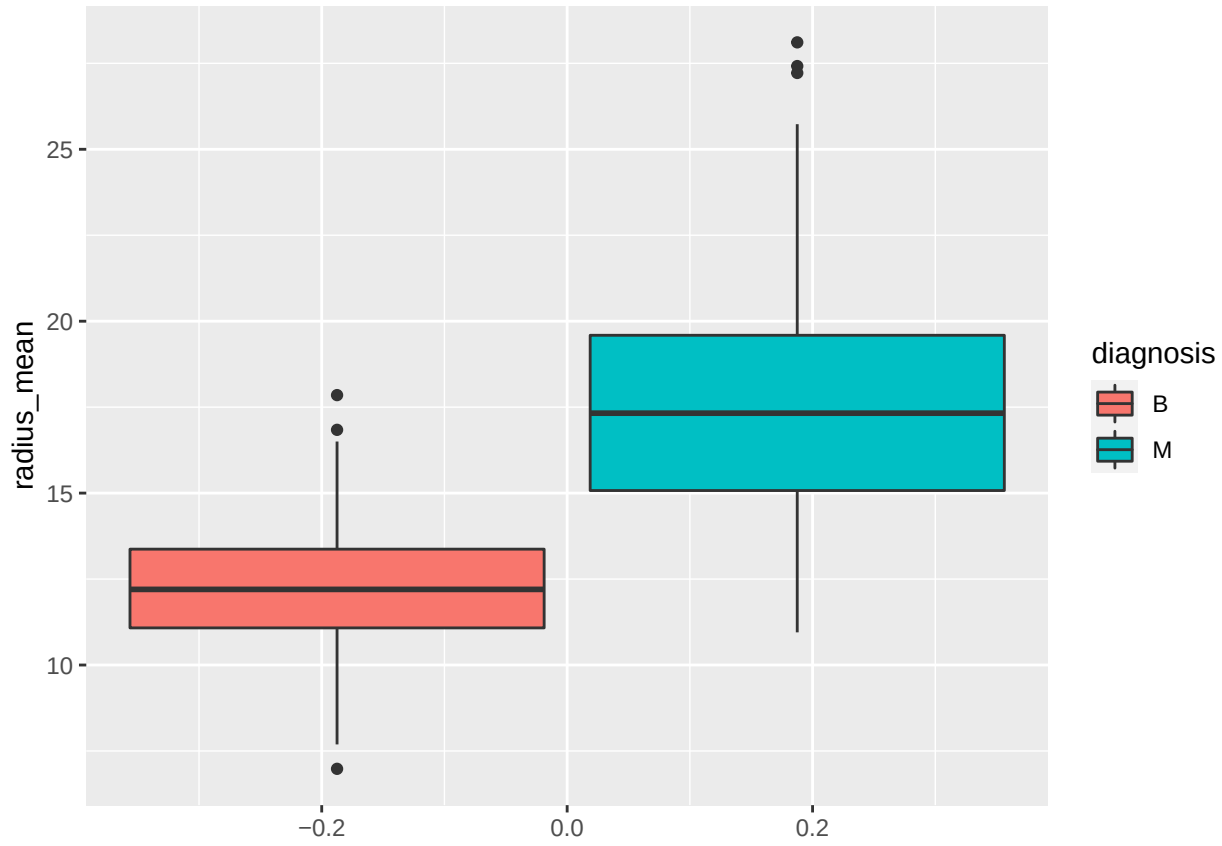
```
table(FNA$diagnosis)
```

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

EDA

Comparison of radius_mean

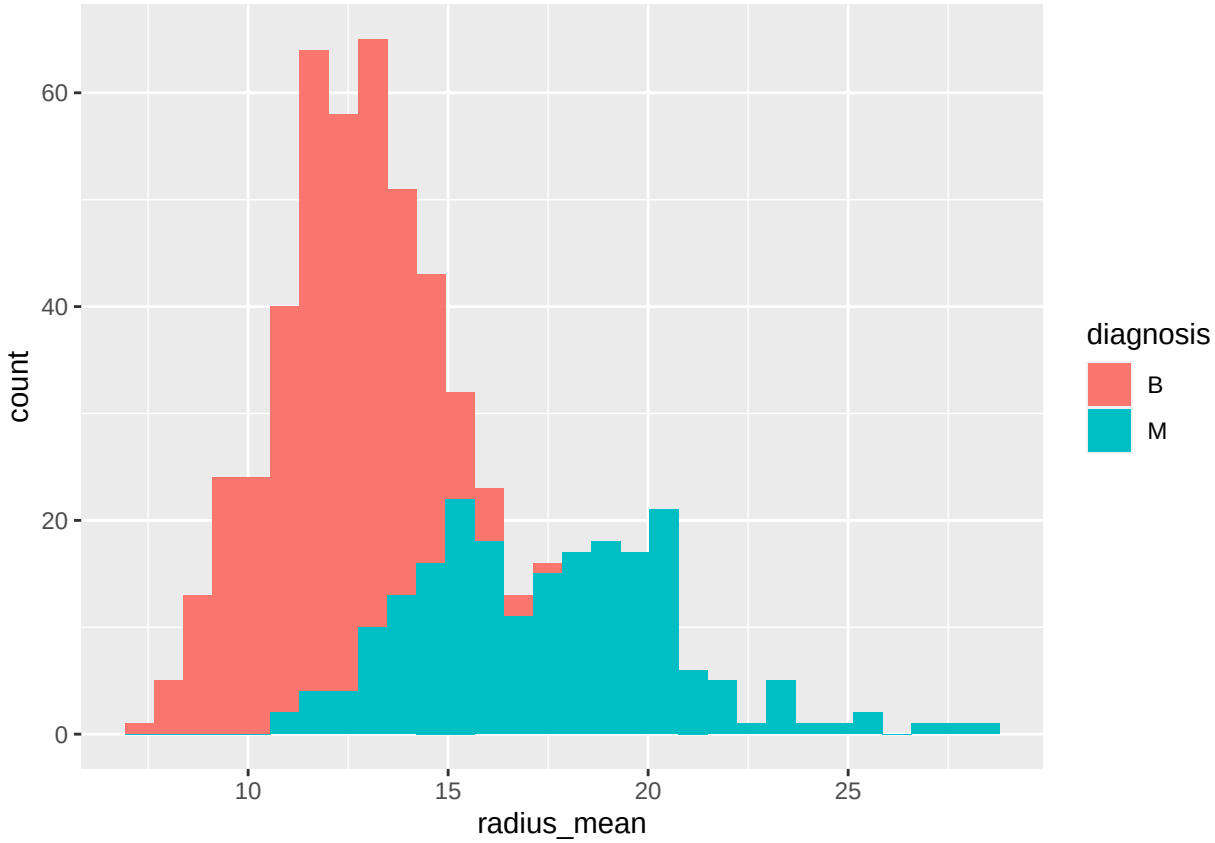
```
ggplot(FNA, aes(x=radius_mean, fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of radius_mean

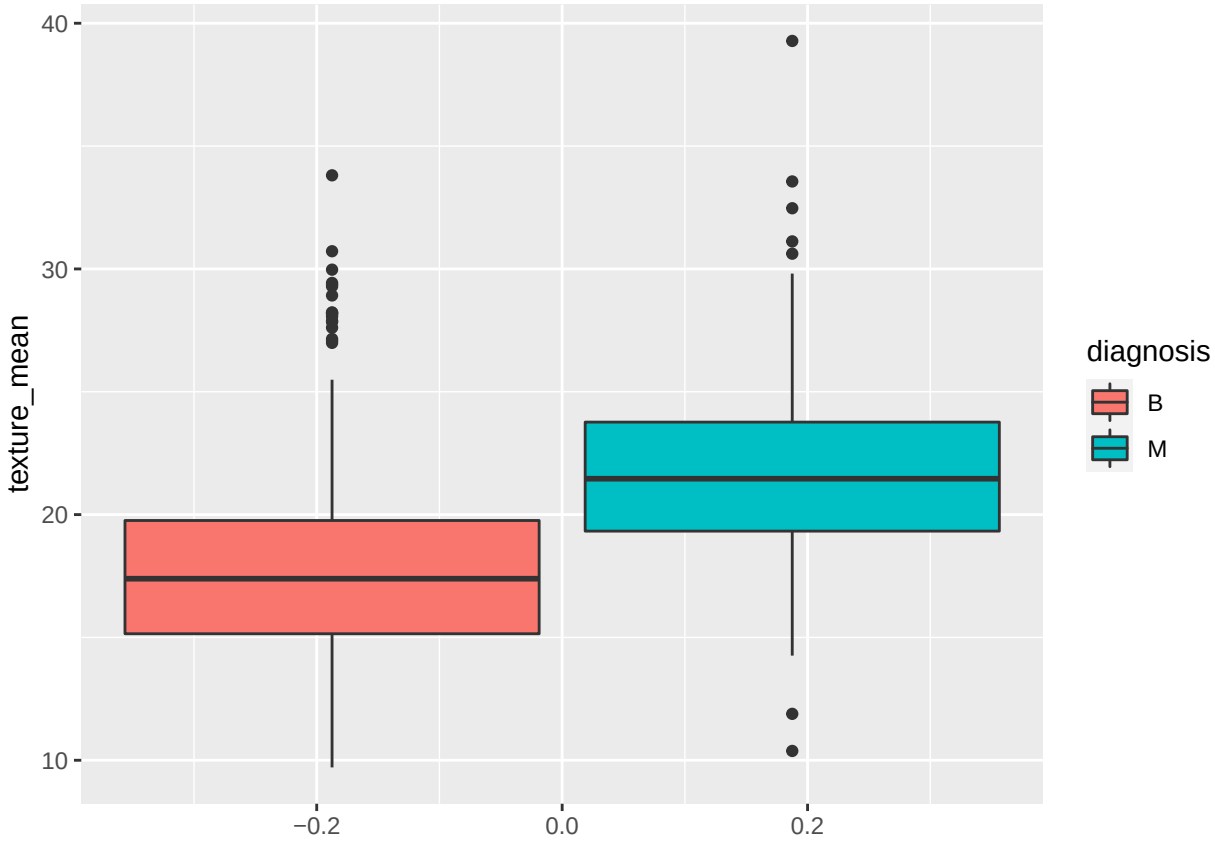
```
ggplot(FNA, aes(x=radius_mean, fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comparison of texture_mean

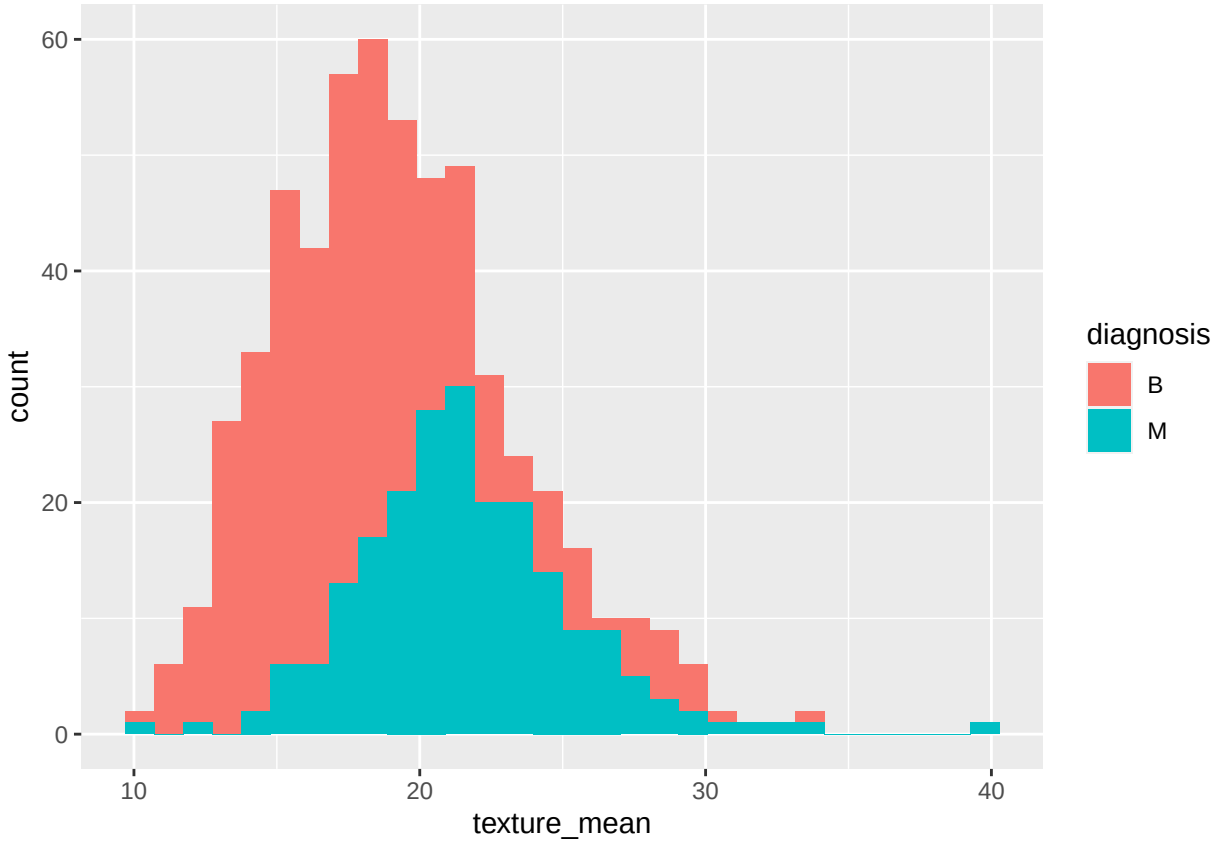
```
ggplot(FNA, aes(x=texture_mean, fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of texture_mean

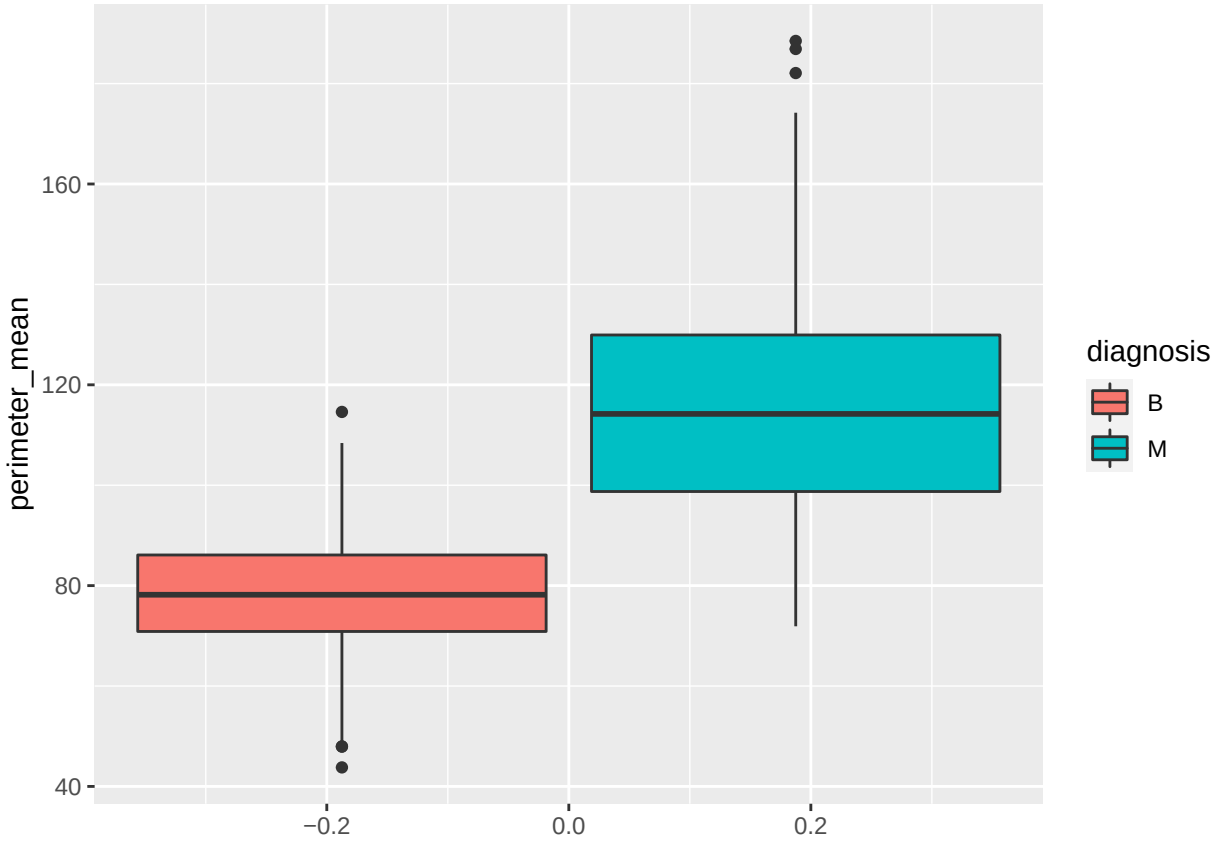
```
ggplot(FNA, aes(x=texture_mean, fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comparison of perimeter_mean

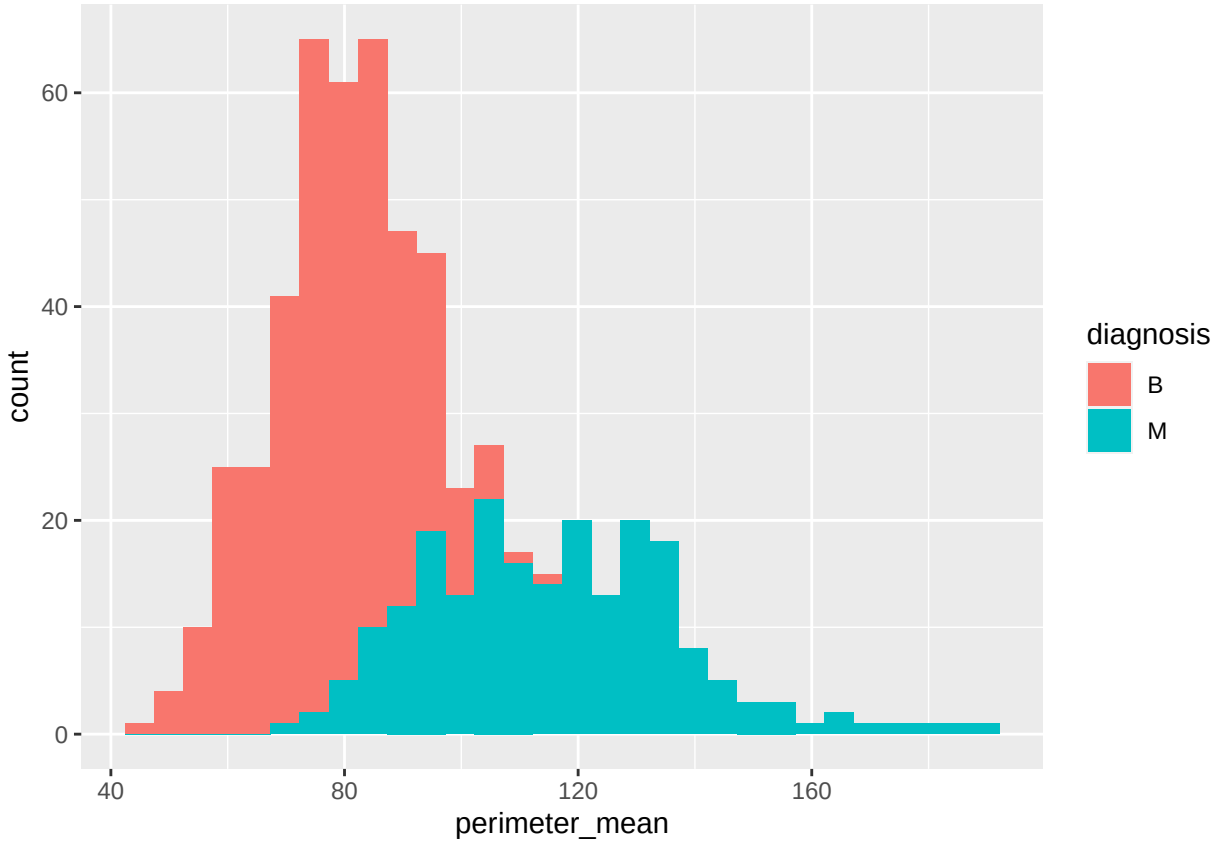
```
ggplot(FNA, aes(x=perimeter_mean, fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of perimeter_mean

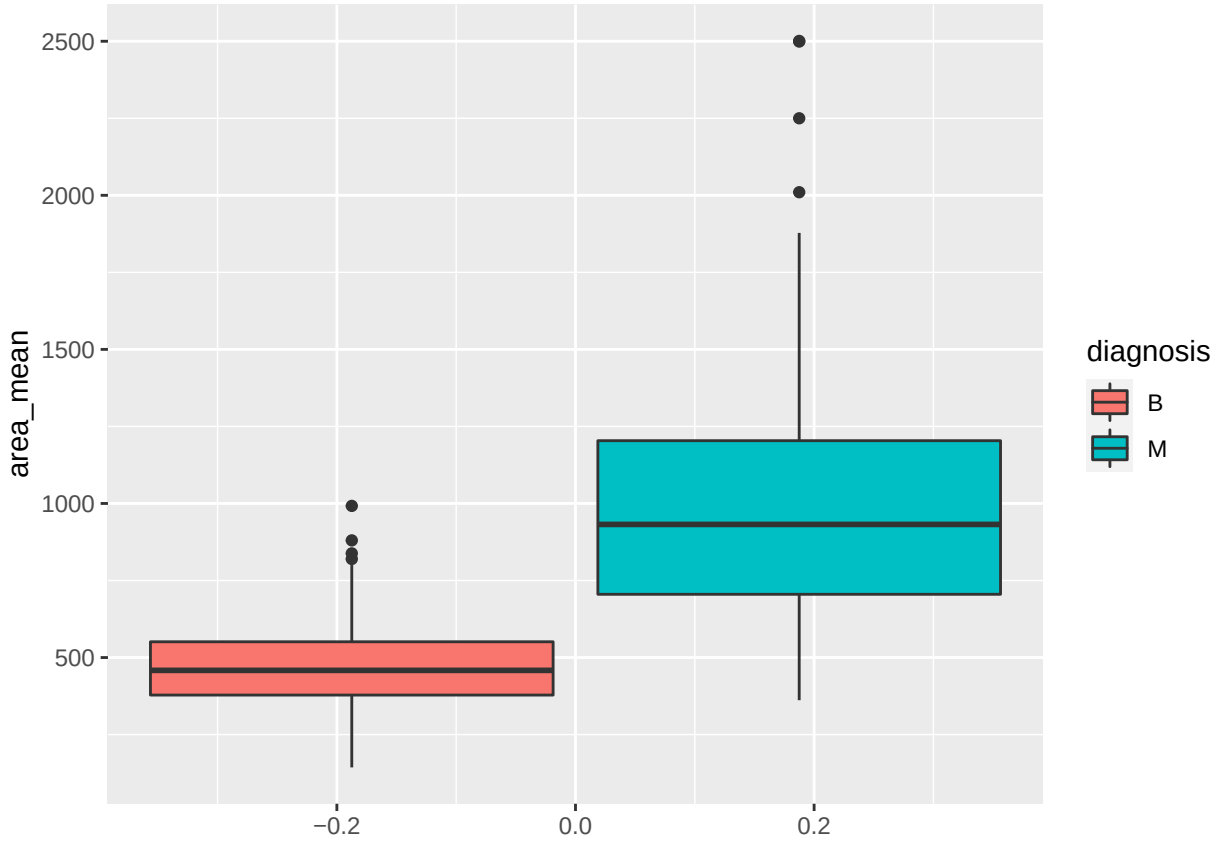
```
ggplot(FNA, aes(x=perimeter_mean, fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comparison of area_mean

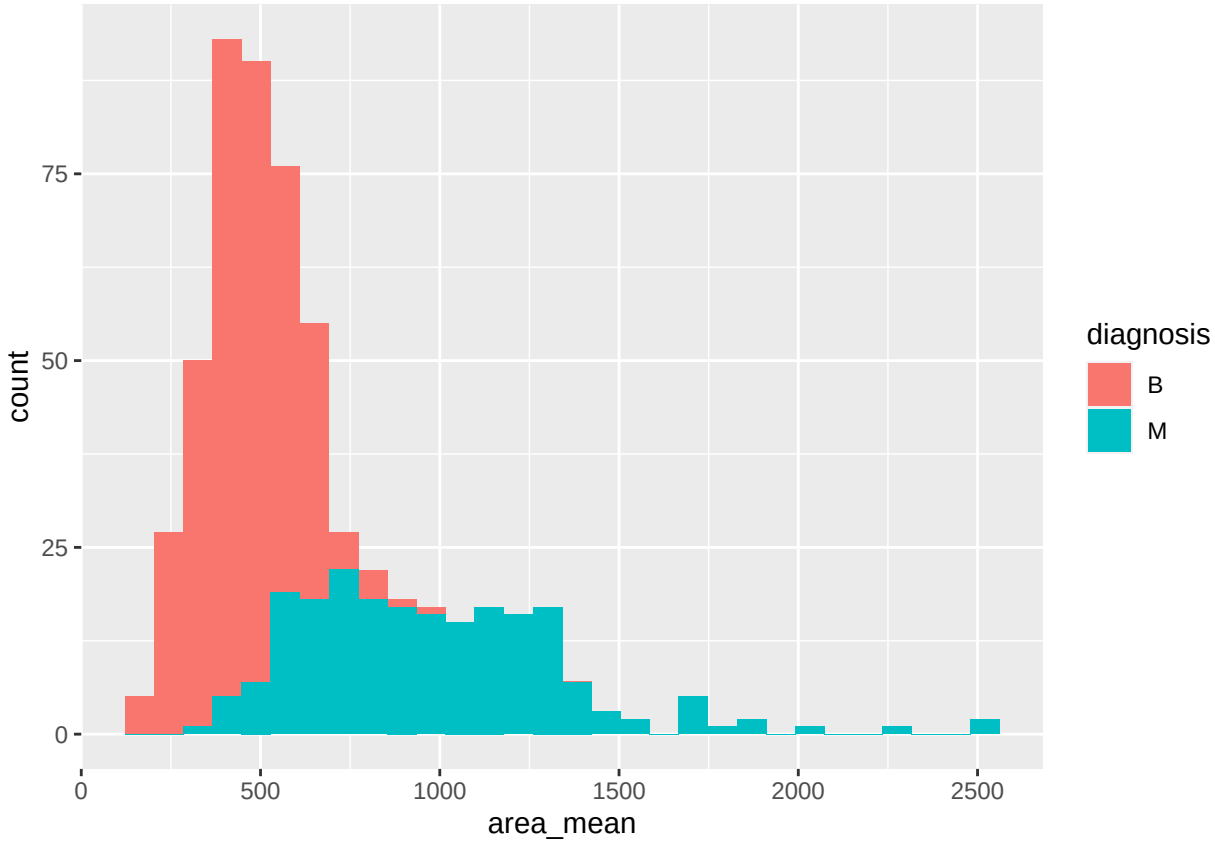
```
ggplot(FNA, aes(x=area_mean, fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of area_mean

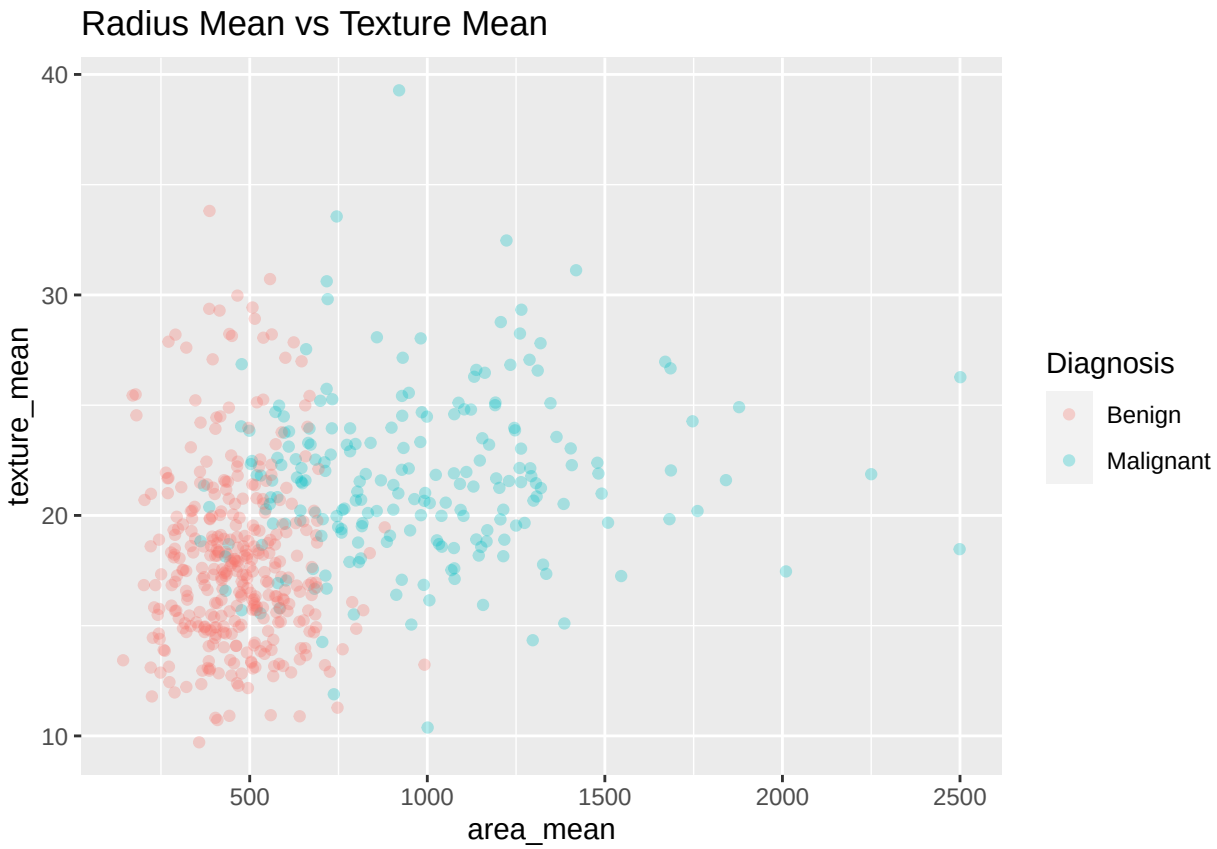
```
ggplot(FNA, aes(x=area_mean, fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Scatter plot of radius_mean versus texture_mean

```
radius_texture <- ggplot(FNA, aes(x=area_mean, y=texture_mean, color=factor(diagnosis, labels = c("Benig
    geom_point(alpha = 0.3)+
  ggtitle("Radius Mean vs Texture Mean")+ labs(color = "Diagnosis")
radius_texture
```

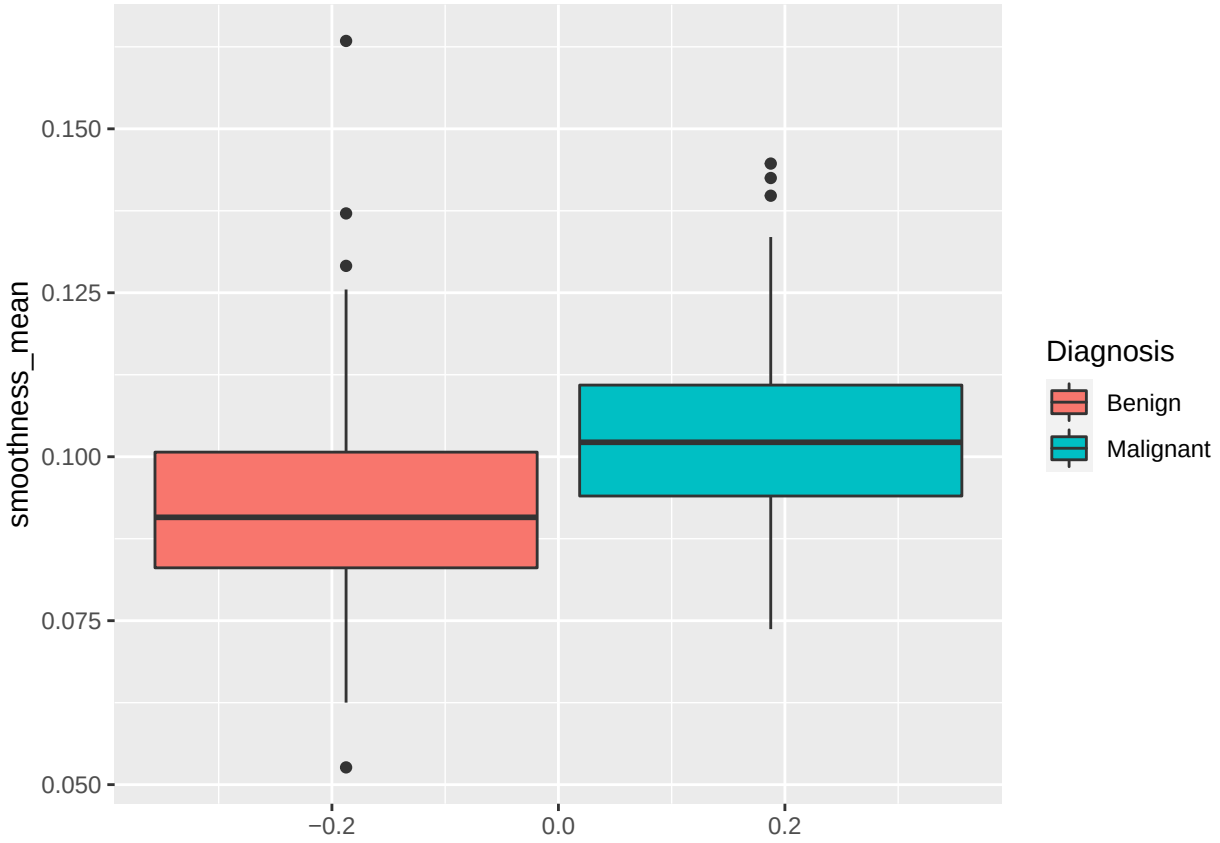


Save image

```
ggsave(filename = "radius_v_texture.jpg", width = 8, height = 5)
```

Comparison of smoothness_mean

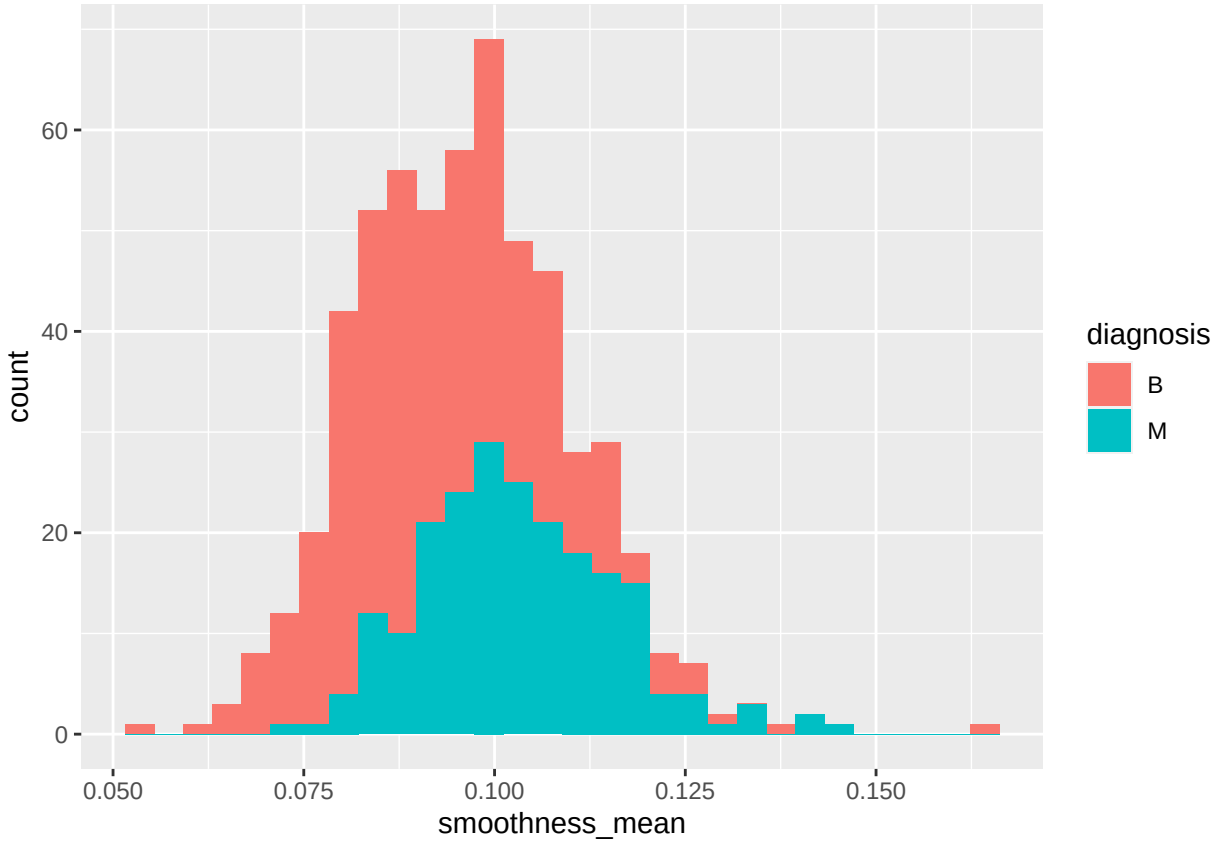
```
ggplot(FNA, aes(x=smoothness_mean, fill=diagnosis)) +
    geom_boxplot()+ coord_flip()+
  scale_fill_discrete(name = "Diagnosis", labels = c("Benign", "Malignant"))
```



Histogram of smoothness_mean

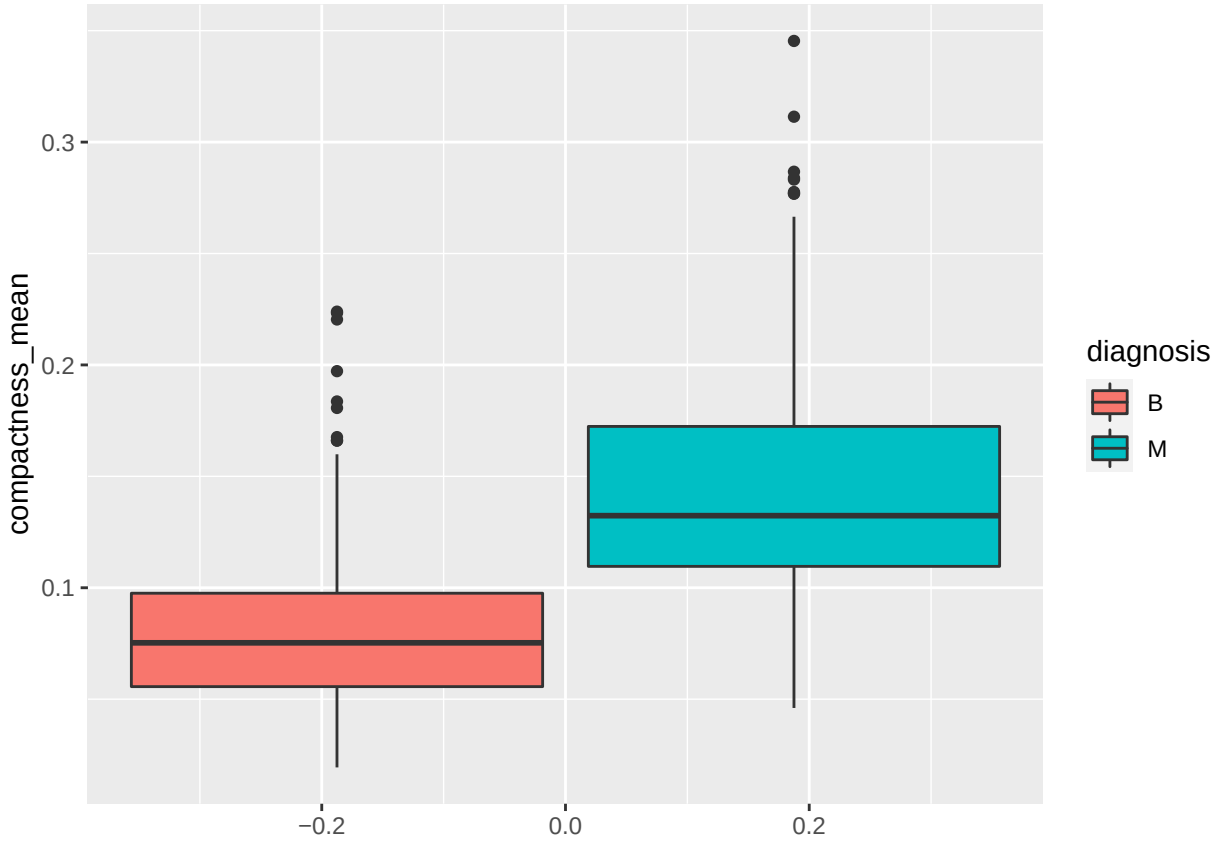
```
ggplot(FNA, aes(x=smoothness_mean, fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comparison of compactness_mean

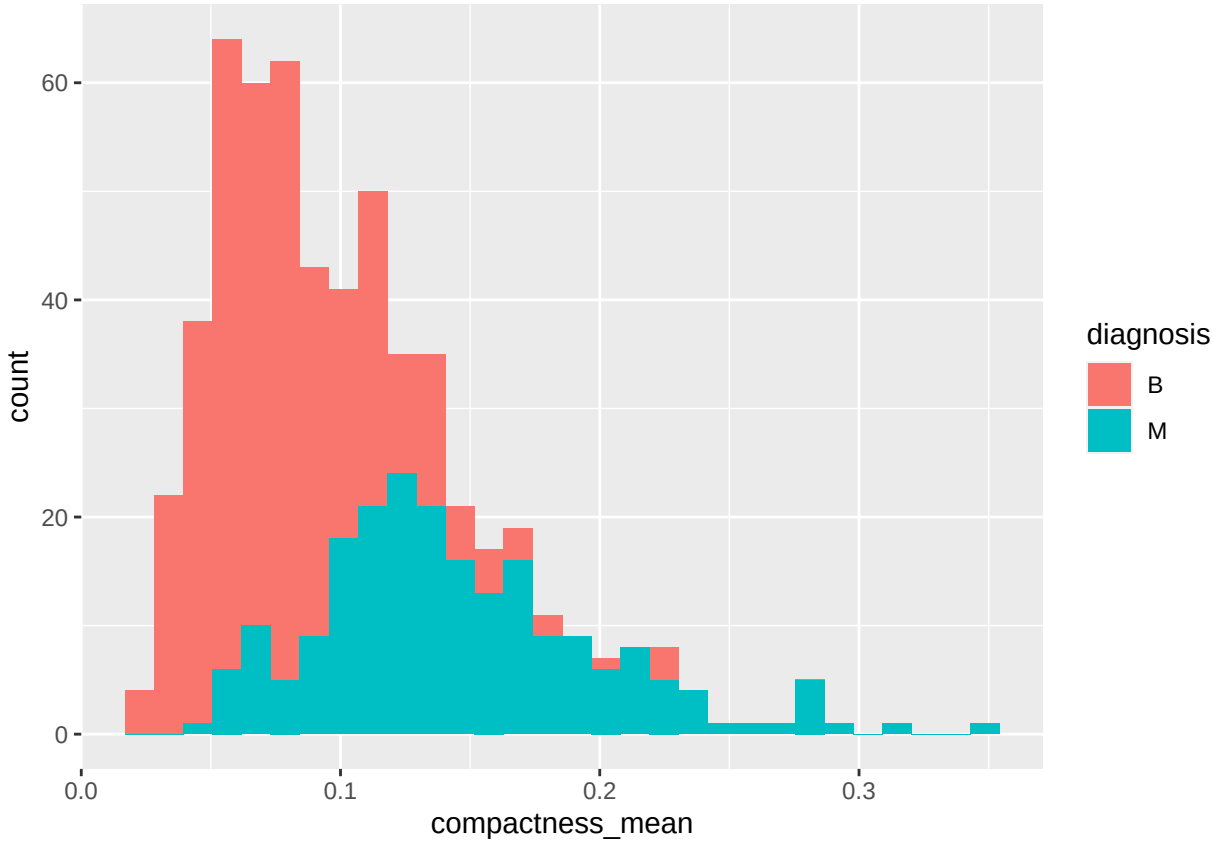
```
ggplot(FNA, aes(x=compactness_mean, fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of compactness_mean

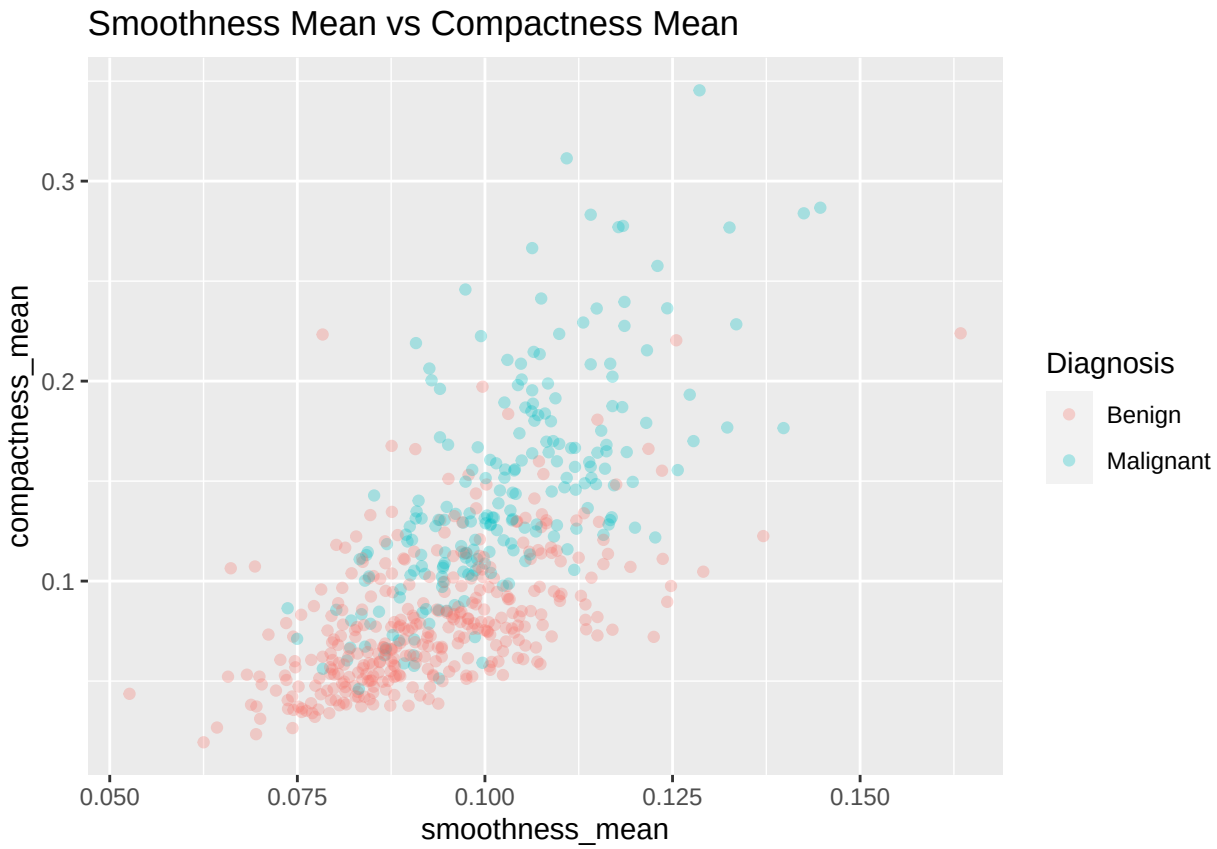
```
ggplot(FNA, aes(x=compactness_mean, fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comparison of Smoothness and Compactness

```
smooth_compact <- ggplot(FNA, aes(x=smoothness_mean, y=compactness_mean, color=factor(diagnosis, labels
    geom_point(alpha = 0.3)+
  ggtitle("Smoothness Mean vs Compactness Mean")+ labs(color = "Diagnosis")
smooth_compact
```

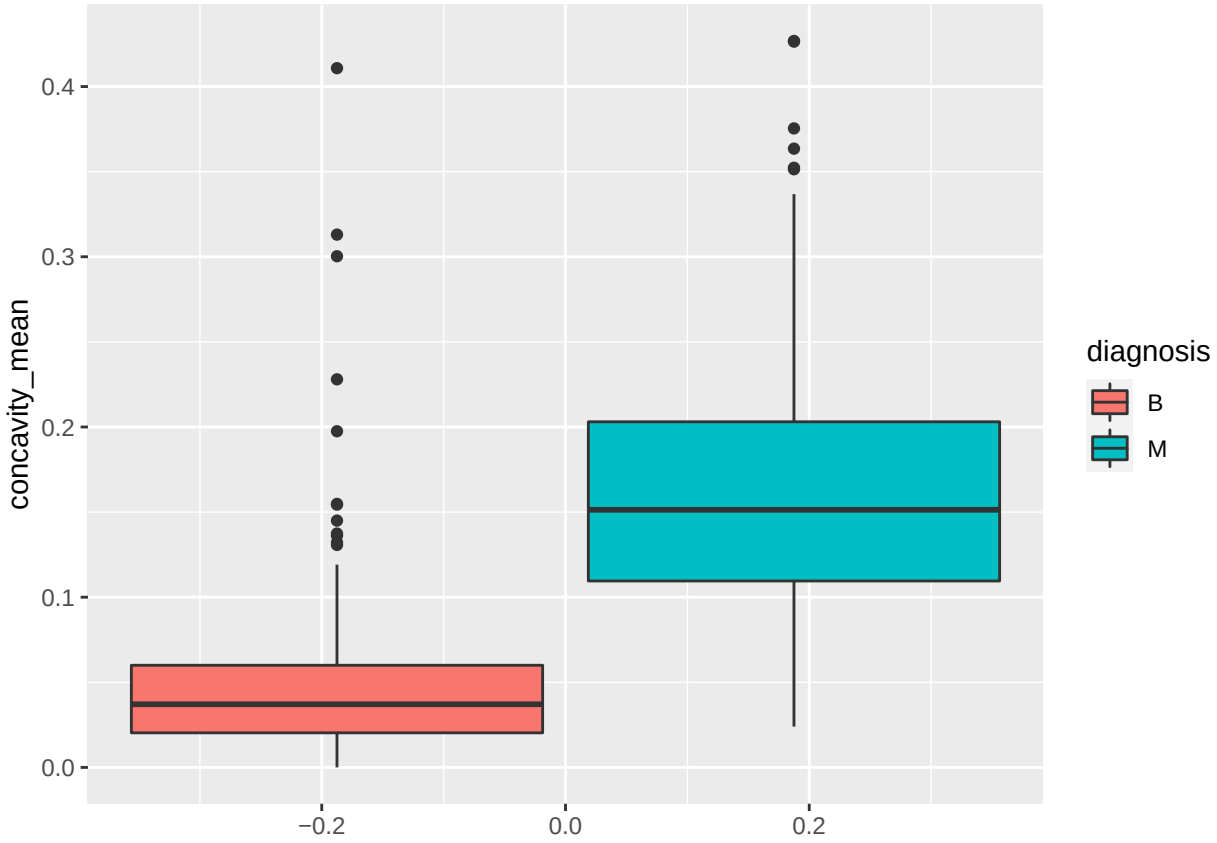


Save image

```
ggsave(plot = smooth_compact, filename = "smoothness_v_compactness.jpg", width = 8, height = 5)
```

Comparison of concavity_mean

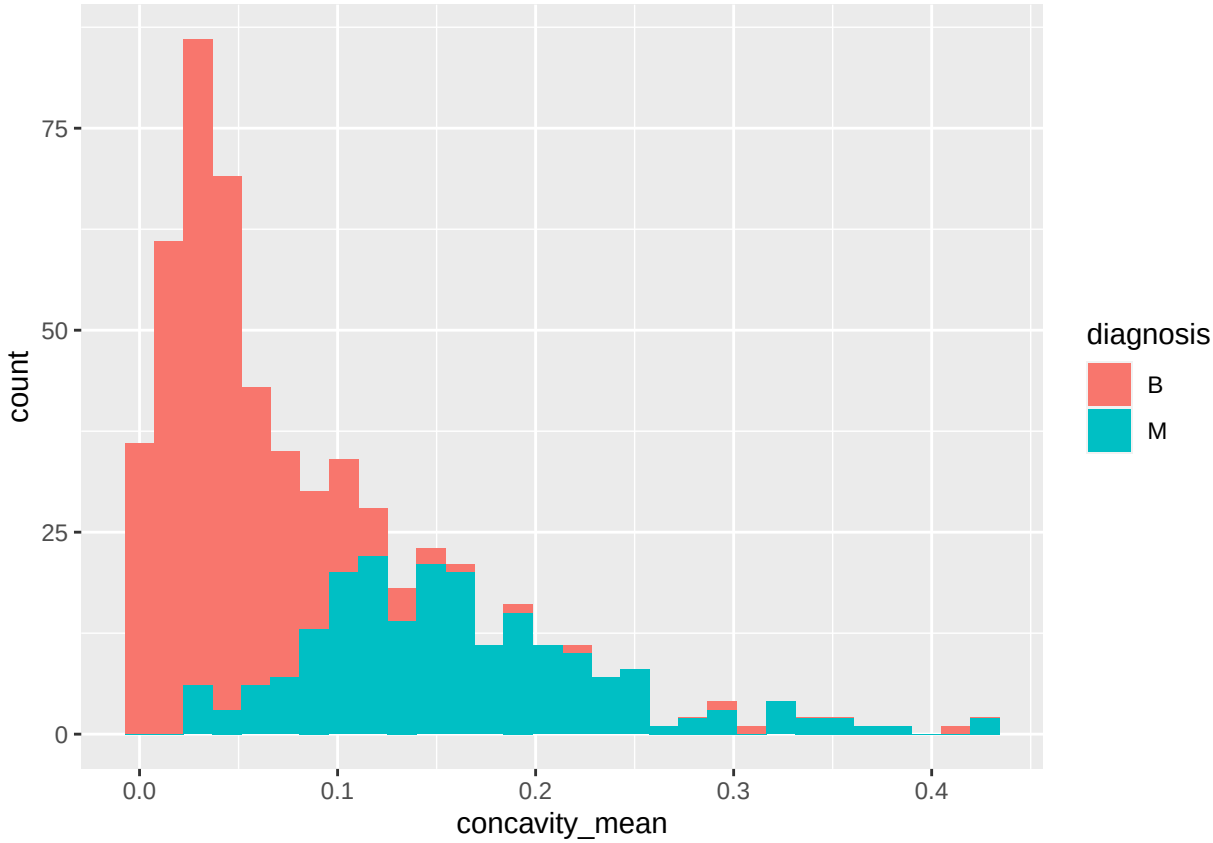
```
ggplot(FNA, aes(x=concavity_mean  , fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of concavity_mean

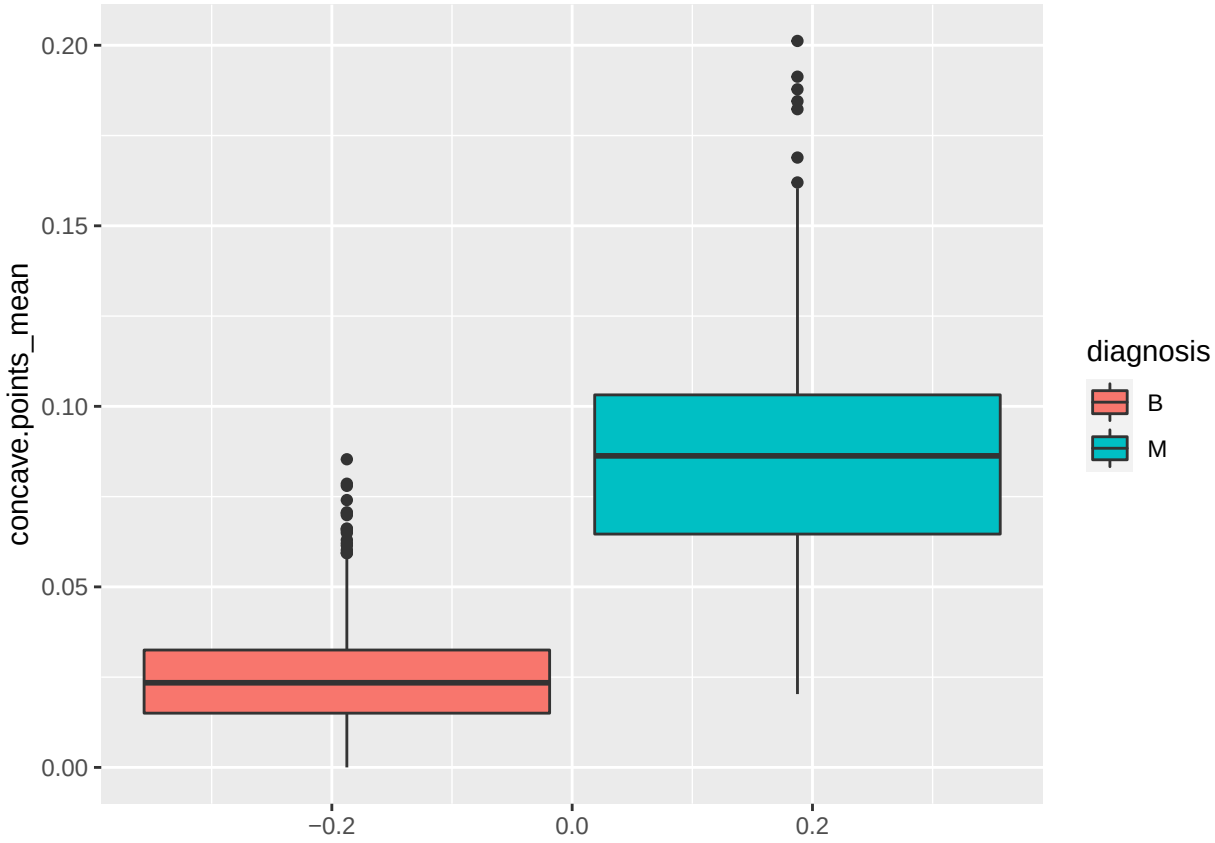
```
ggplot(FNA, aes(x=concavity_mean   , fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comparison of concave.points_mean

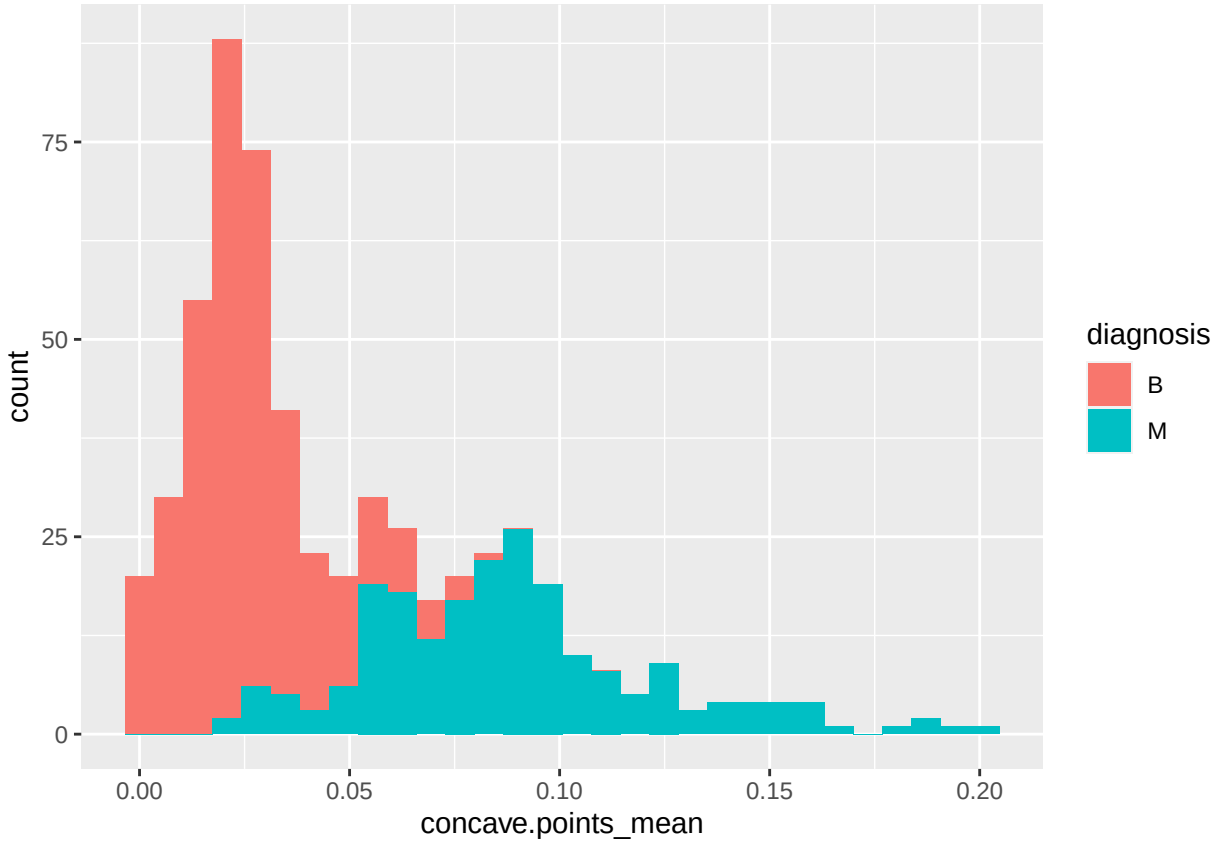
```
ggplot(FNA, aes(x=concave.points_mean   , fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of concave.points_mean

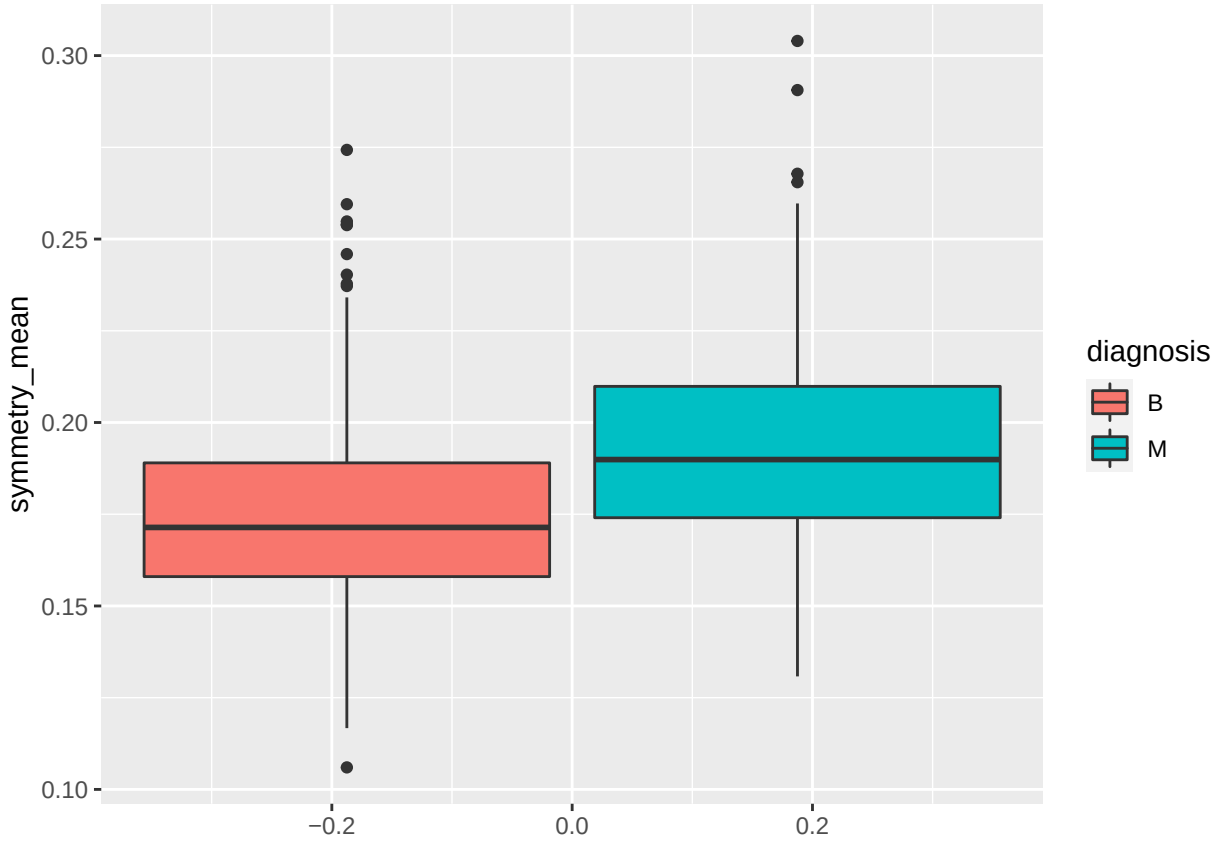
```
ggplot(FNA, aes(x=concave.points_mean   , fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comparison of symmetry_mean

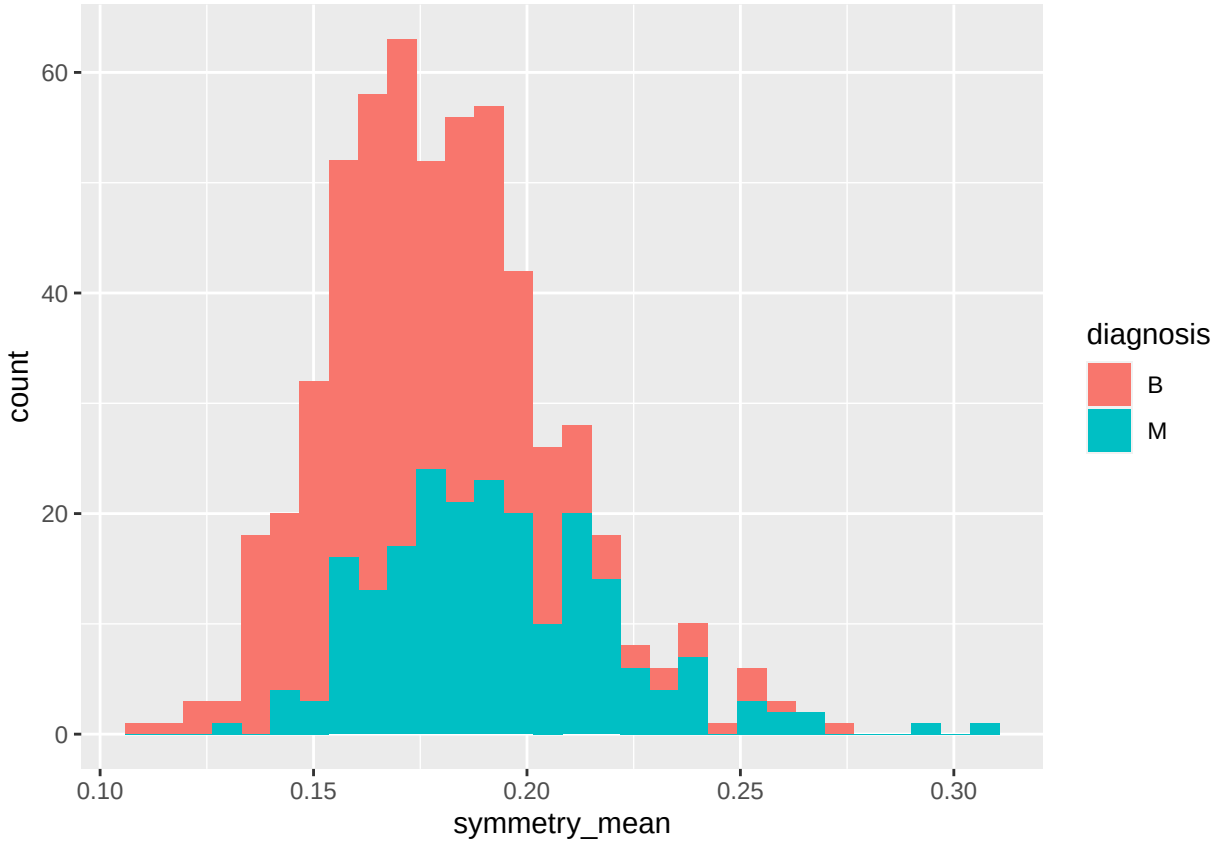
```
ggplot(FNA, aes(x=symmetry_mean    , fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of symmetry_mean

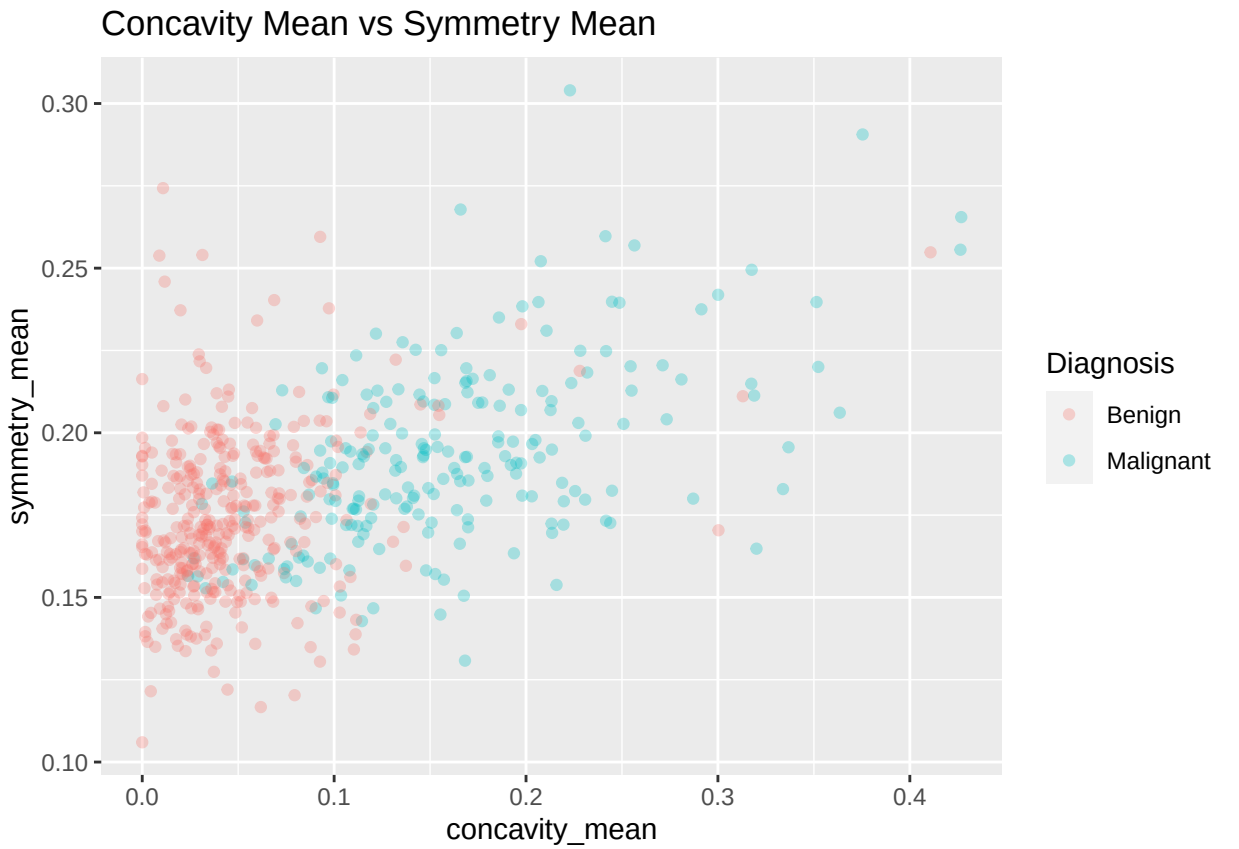
```
ggplot(FNA, aes(x=symmetry_mean    , fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Comparison of Concavity and Symmetry

```
concavity_symmetry <- ggplot(FNA, aes(x=concavity_mean, y=symmetry_mean, color=factor(diagnosis, labels
    geom_point(alpha = 0.3)+
  ggtitle("Concavity Mean vs Symmetry Mean")+ labs(color = "Diagnosis")
concavity_symmetry
```

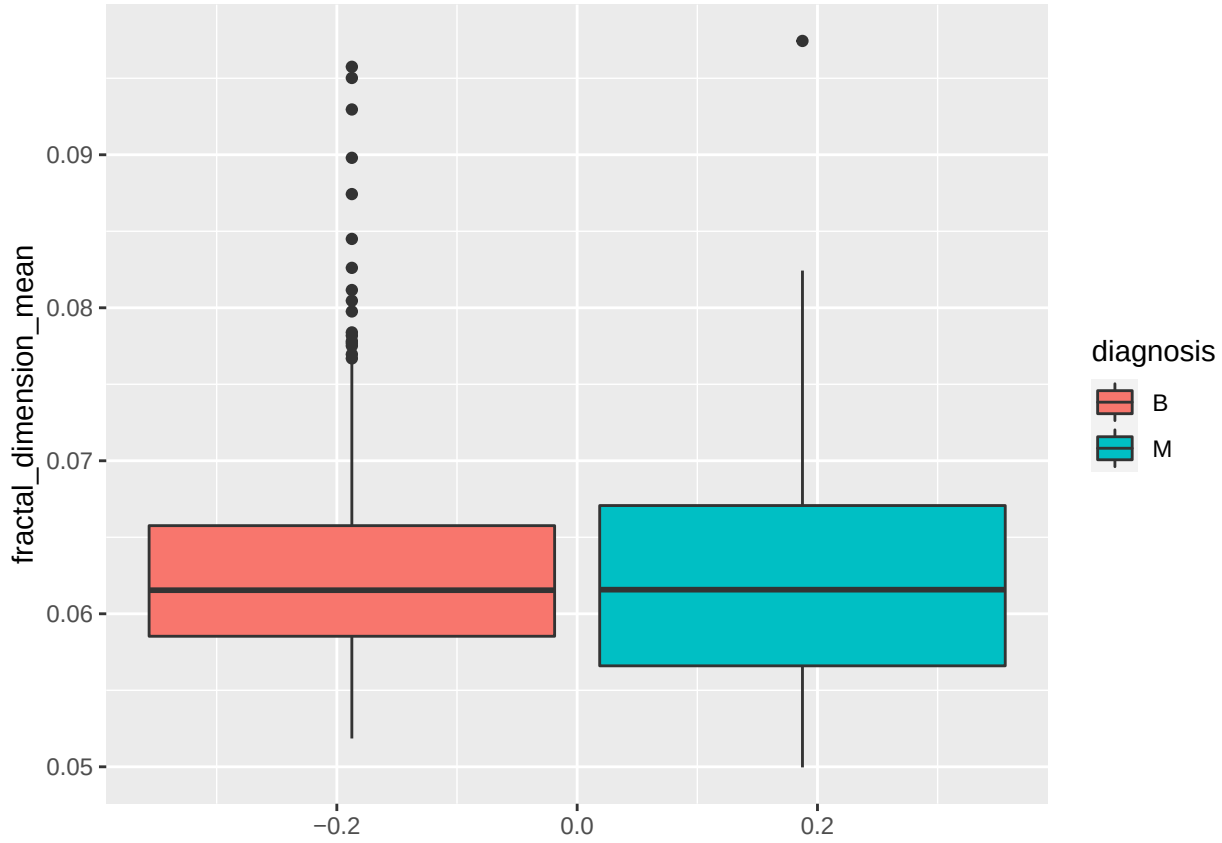


Save image

```
ggsave(plot = concavity_symmetry, filename = "concavity_symmetry.jpg", width = 8, height = 5)
```

Comparison of fractal_dimension_mean

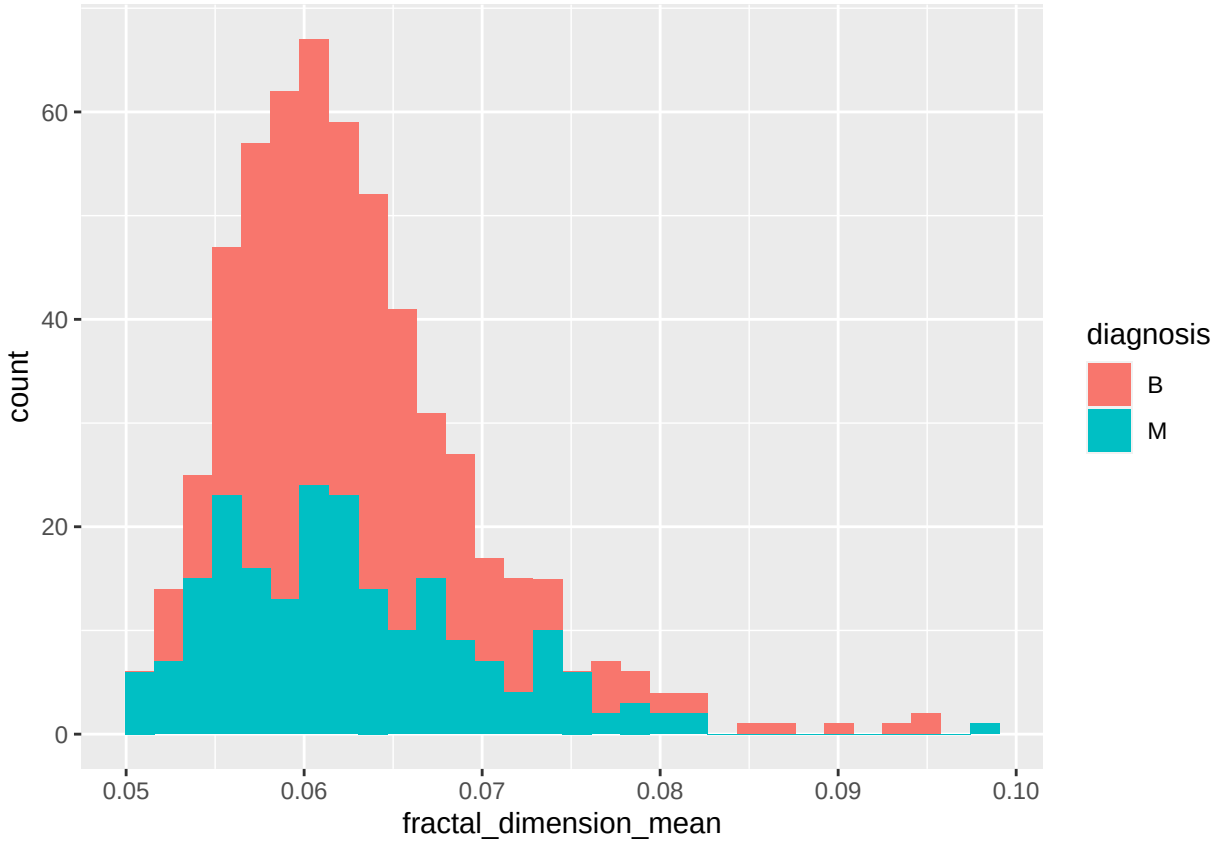
```
ggplot(FNA, aes(x=fractal_dimension_mean     , fill=diagnosis)) +
    geom_boxplot()+ coord_flip()
```



Histogram of fractal_dimension_mean

```
ggplot(FNA, aes(x=fractal_dimension_mean    , fill=diagnosis)) +
    geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



split the data into test and training data

The `id` will be removed from the data set since it is not a relevant predictor

```
FNA_trim <- FNA %>% dplyr::select(-id)
```

Seed is set to 1842. The number of rows are counted and assigned to the variable `n`. The test index is created by randomly sampling 20% of the number of rows. The test index is then used to subset the data to create the test set and the inverse of the test index is used to subset the data to create the training data.

```
set.seed(1842)
n <- nrow(FNA_trim)
test_idx <- sample.int(n, size=(n*.2))
test_FNA <- FNA_trim[test_idx,]
train_FNA <- FNA_trim[-test_idx,]
glimpse(train_FNA)
```

```
## Rows: 456
## Columns: 31
## $ diagnosis              <fct> M, M, M, M, M, M, M, M, M, M, M, M, M, M, M...
## $ radius_mean            <dbl> 17.990, 20.570, 19.690, 11.420, 12.450, 18....
## $ texture_mean           <dbl> 10.38, 17.77, 21.25, 20.38, 15.70, 19.98, 2...
## $ perimeter_mean         <dbl> 122.80, 132.90, 130.00, 77.58, 82.57, 119.6...
## $ area_mean              <dbl> 1001.0, 1326.0, 1203.0, 386.1, 477.1, 1040....
```

```
## $ smoothness_mean         <dbl> 0.11840, 0.08474, 0.10960, 0.14250, 0.12780...
## $ compactness_mean        <dbl> 0.27760, 0.07864, 0.15990, 0.28390, 0.17000...
## $ concavity_mean          <dbl> 0.30010, 0.08690, 0.19740, 0.24140, 0.15780...
## $ concave.points_mean     <dbl> 0.14710, 0.07017, 0.12790, 0.10520, 0.08089...
## $ symmetry_mean           <dbl> 0.2419, 0.1812, 0.2069, 0.2597, 0.2087, 0.1...
## $ fractal_dimension_mean  <dbl> 0.07871, 0.05667, 0.05999, 0.09744, 0.07613...
## $ radius_se               <dbl> 1.0950, 0.5435, 0.7456, 0.4956, 0.3345, 0.4...
## $ texture_se              <dbl> 0.9053, 0.7339, 0.7869, 1.1560, 0.8902, 0.7...
## $ perimeter_se            <dbl> 8.589, 3.398, 4.585, 3.445, 2.217, 3.180, 2...
## $ area_se                 <dbl> 153.40, 74.08, 94.03, 27.23, 27.19, 53.91, ...
## $ smoothness_se           <dbl> 0.006399, 0.005225, 0.006150, 0.009110, 0.0...
## $ compactness_se          <dbl> 0.049040, 0.013080, 0.040060, 0.074580, 0.0...
## $ concavity_se            <dbl> 0.05373, 0.01860, 0.03832, 0.05661, 0.03672...
## $ concave.points_se       <dbl> 0.015870, 0.013400, 0.020580, 0.018670, 0.0...
## $ symmetry_se             <dbl> 0.03003, 0.01389, 0.02250, 0.05963, 0.02165...
## $ fractal_dimension_se    <dbl> 0.006193, 0.003532, 0.004571, 0.009208, 0.0...
## $ radius_worst            <dbl> 25.38, 24.99, 23.57, 14.91, 15.47, 22.88, 1...
## $ texture_worst           <dbl> 17.33, 23.41, 25.53, 26.50, 23.75, 27.66, 3...
## $ perimeter_worst         <dbl> 184.60, 158.80, 152.50, 98.87, 103.40, 153....
## $ area_worst              <dbl> 2019.0, 1956.0, 1709.0, 567.7, 741.6, 1606....
## $ smoothness_worst        <dbl> 0.1622, 0.1238, 0.1444, 0.2098, 0.1791, 0.1...
## $ compactness_worst       <dbl> 0.6656, 0.1866, 0.4245, 0.8663, 0.5249, 0.2...
## $ concavity_worst         <dbl> 0.71190, 0.24160, 0.45040, 0.68690, 0.53550...
## $ concave.points_worst    <dbl> 0.26540, 0.18600, 0.24300, 0.25750, 0.17410...
## $ symmetry_worst          <dbl> 0.4601, 0.2750, 0.3613, 0.6638, 0.3985, 0.3...
## $ fractal_dimension_worst <dbl> 0.11890, 0.08902, 0.08758, 0.17300, 0.12440...
```

Decision Tree

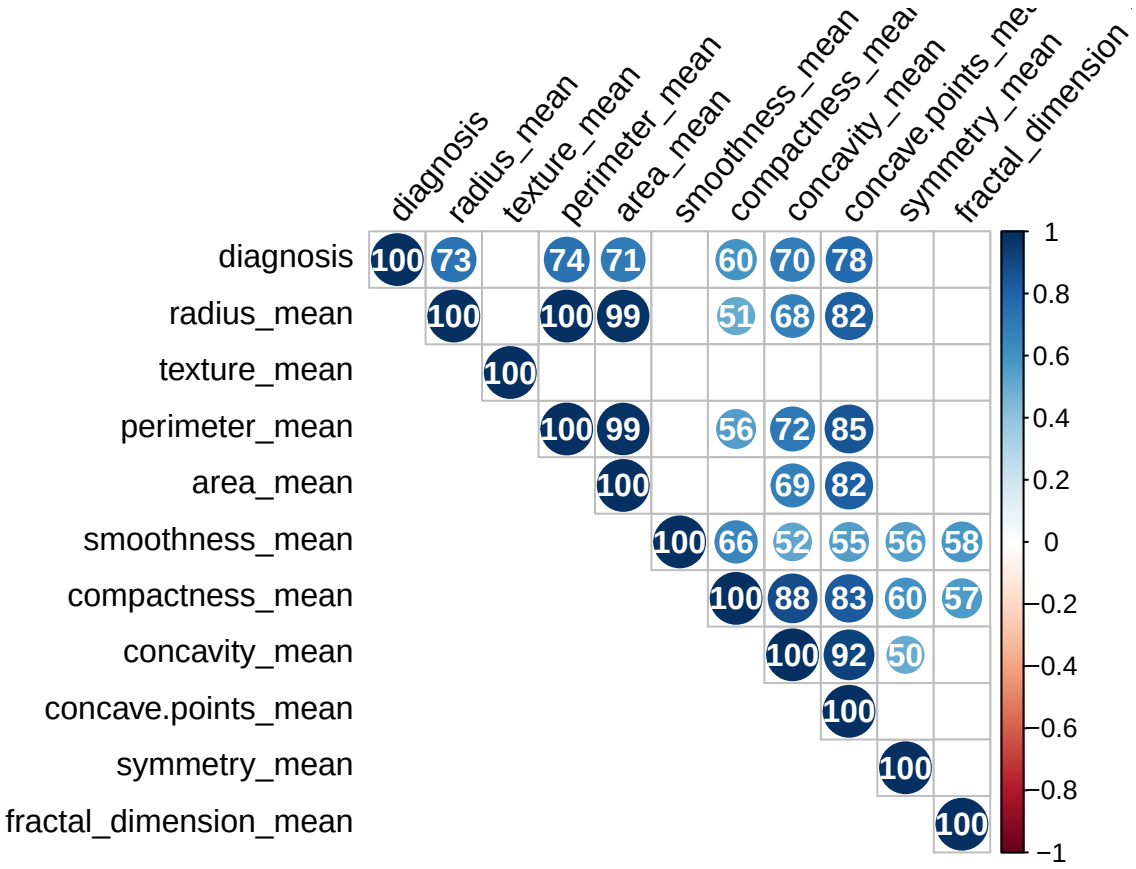
```
library(corrplot)
```

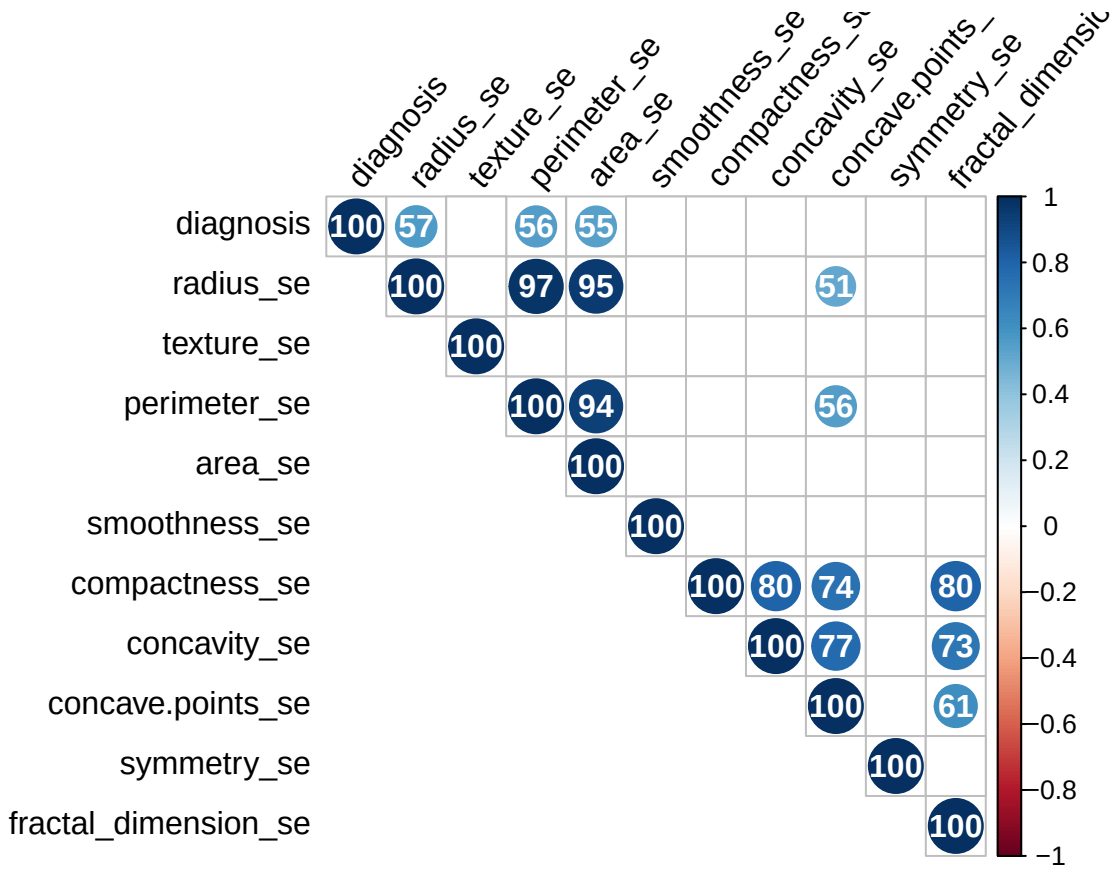
```
## Warning: package 'corrplot' was built under R version 4.0.3
```

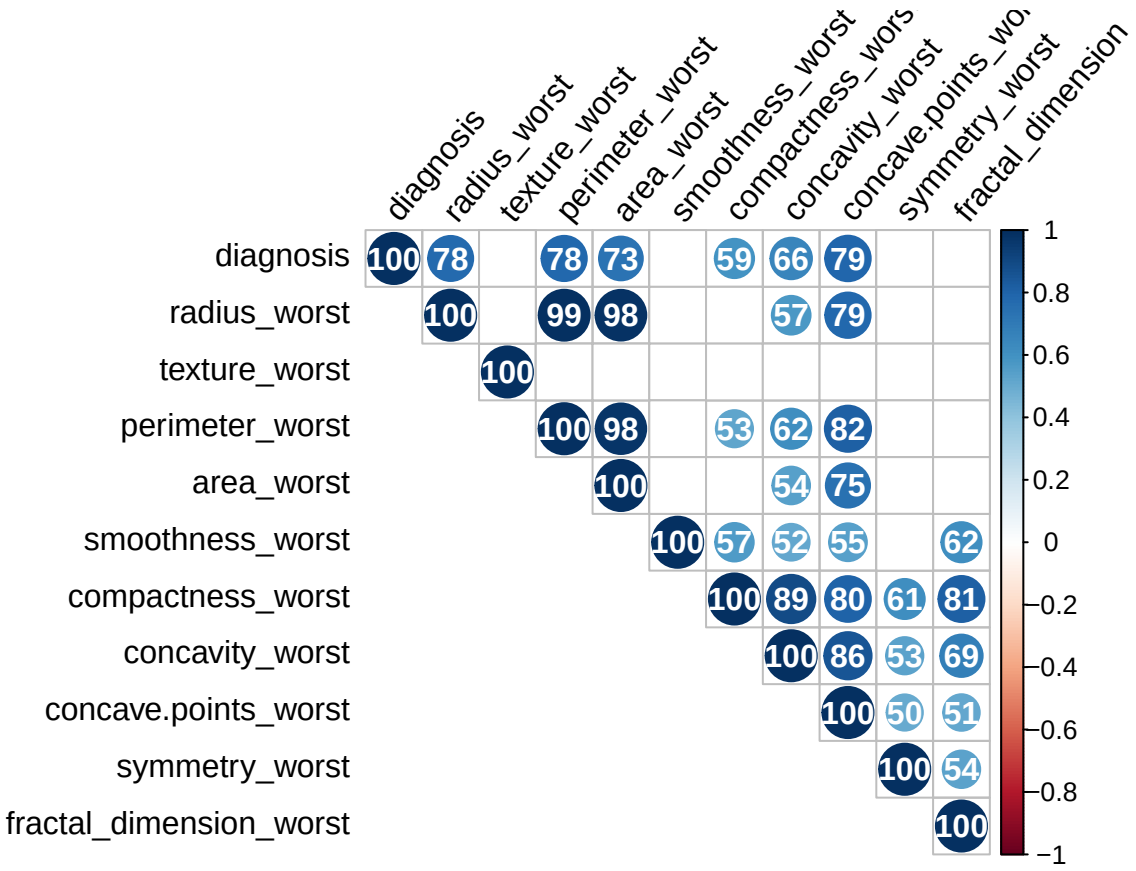
```
## corrplot 0.84 loaded
```

```
#Looking at some corr plots
#Given that the data is a set of metrics viewed in three statistical perspectives, mean, se, and worst,



#Load and manipulate data for corrplots
FNAplots <- read.csv("FNA_cancer.csv", header = T)



#Converting diagnosis response from factor to binary to allow for corr calc
FNAplots$diagnosis <- ifelse(FNAplots$diagnosis=="M",1,0)
```

```r
#Breaking out the predictor columns into 3 subcategories and adding the diagnosis column
FNAmeans <- dplyr::select(FNAplots,contains("mean")) %>% mutate(diagnosis = FNAplots$diagnosis) %>% rel



FNAse <- dplyr::select(FNAplots,contains("_se")) %>% mutate(diagnosis = FNAplots$diagnosis) %>% relocat



FNAworst <- dplyr::select(FNAplots,contains("worst")) %>% mutate(diagnosis = FNAplots$diagnosis) %>% re



#create correlation matricies



fnaCorPlot <- function(df){

  cor_fna <- cor(df)
  #plot cor matrix
  corrplot(cor_fna,  method="circle", tl.pos="lt", type="upper",
  tl.col="black", tl.cex=1, tl.srt=50,
           addCoef.col="white", addCoefasPercent = TRUE,
           p.mat = 1-abs(cor_fna), sig.level=0.50, insig = "blank")
  }



lapply(list(FNAmeans,FNAse,FNAworst),fnaCorPlot)
```


diagnosis
radius_mean
texture_mean
perimeter_mean
area_mean
smoothness_mean
compactness_mean
concavity_mean
concave.points_mean
symmetry_mean
fractal_dimension_mean
100 73 74 71 60 70 78
100 100 99 51 68 82
100
100 99 56 72 85
100 69 82
100 66 52 55 56 58
100 88 83 60 57
100 92 50
100
100
100
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1


diagnosis
radius_se
texture_se
perimeter_se
area_se
smoothness_se
compactness_se
concavity_se
concave.points_se
symmetry_se
fractal_dimension_se
100 57 56 55
100 97 95 51
100
100 94 56
100
100
100 80 74 80
100 77 73
100 61
100
100
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1




```
## [[1]]
##                          diagnosis radius_mean texture_mean perimeter_mean
## diagnosis                1.0000000   0.7300285   0.41518530      0.7426355
## radius_mean              0.7300285   1.0000000   0.32378189      0.9978553
## texture_mean             0.4151853   0.3237819   1.00000000      0.3295331
## perimeter_mean           0.7426355   0.9978553   0.32953306      1.0000000
## area_mean                0.7089838   0.9873572   0.32108570      0.9865068
## smoothness_mean          0.3585600   0.1705812  -0.02338852      0.2072782
## compactness_mean         0.5965337   0.5061236   0.23670222      0.5569362
## concavity_mean           0.6963597   0.6767636   0.30241783      0.7161357
## concave.points_mean      0.7766138   0.8225285   0.29346405      0.8509770
## symmetry_mean            0.3304986   0.1477412   0.07140098      0.1830272
## fractal_dimension_mean -0.0128376  -0.3116308  -0.07643718     -0.2614769
##                          area_mean smoothness_mean compactness_mean
## diagnosis                0.7089838      0.35855997        0.5965337
## radius_mean              0.9873572      0.17058119        0.5061236
## texture_mean             0.3210857     -0.02338852        0.2367022
## perimeter_mean           0.9865068      0.20727816        0.5569362
## area_mean                1.0000000      0.17702838        0.4985017
## smoothness_mean          0.1770284      1.00000000        0.6591232
## compactness_mean         0.4985017      0.65912322        1.0000000
## concavity_mean           0.6859828      0.52198377        0.8831207
## concave.points_mean      0.8232689      0.55369517        0.8311350
## symmetry_mean            0.1512931      0.55777479        0.6026410
## fractal_dimension_mean -0.2831098      0.58479200        0.5653687
##                         concavity_mean concave.points_mean symmetry_mean
```

```
## diagnosis                   0.6963597           0.7766138    0.33049855
## radius_mean                 0.6767636           0.8225285    0.14774124
## texture_mean                0.3024178           0.2934641    0.07140098
## perimeter_mean              0.7161357           0.8509770    0.18302721
## area_mean                   0.6859828           0.8232689    0.15129308
## smoothness_mean             0.5219838           0.5536952    0.55777479
## compactness_mean            0.8831207           0.8311350    0.60264105
## concavity_mean              1.0000000           0.9213910    0.50066662
## concave.points_mean         0.9213910           1.0000000    0.46249739
## symmetry_mean               0.5006666           0.4624974    1.00000000
## fractal_dimension_mean      0.3367834           0.1669174    0.47992133
##                        fractal_dimension_mean
## diagnosis                         -0.01283760
## radius_mean                       -0.31163083
## texture_mean                      -0.07643718
## perimeter_mean                    -0.26147691
## area_mean                         -0.28310981
## smoothness_mean                    0.58479200
## compactness_mean                   0.56536866
## concavity_mean                     0.33678336
## concave.points_mean                0.16691738
## symmetry_mean                      0.47992133
## fractal_dimension_mean             1.00000000
##
## [[2]]
##                          diagnosis radius_se   texture_se perimeter_se
## diagnosis              1.000000000 0.5671338 -0.008303333    0.5561407
## radius_se              0.567133821 1.0000000  0.213247337    0.9727937
## texture_se            -0.008303333 0.2132473  1.000000000    0.2231707
## perimeter_se           0.556140703 0.9727937  0.223170729    1.0000000
## area_se                0.548235940 0.9518301  0.111567247    0.9376554
## smoothness_se         -0.067016011 0.1645142  0.397242853    0.1510753
## compactness_se         0.292999244 0.3560646  0.231699699    0.4163224
## concavity_se           0.253729766 0.3323575  0.194998464    0.3624816
## concave.points_se      0.408042333 0.5133464  0.230283400    0.5562641
## symmetry_se           -0.006521756 0.2405674  0.411620680    0.2664871
## fractal_dimension_se   0.077972417 0.2277535  0.279722748    0.2441428
##                           area_se smoothness_se compactness_se concavity_se
## diagnosis              0.54823594   -0.06701601      0.2929992    0.2537298
## radius_se              0.95183011    0.16451422      0.3560646    0.3323575
## texture_se             0.11156725    0.39724285      0.2316997    0.1949985
## perimeter_se           0.93765541    0.15107533      0.4163224    0.3624816
## area_se                1.00000000    0.07515034      0.2848401    0.2708947
## smoothness_se          0.07515034    1.00000000      0.3366961    0.2686848
## compactness_se         0.28484006    0.33669608      1.0000000    0.8012683
## concavity_se           0.27089473    0.26868476      0.8012683    1.0000000
## concave.points_se      0.41572957    0.32842950      0.7440827    0.7718040
## symmetry_se            0.13410898    0.41350613      0.3947128    0.3094286
## fractal_dimension_se 0.12707090    0.42737421      0.8032688    0.7273722
##                       concave.points_se  symmetry_se fractal_dimension_se
## diagnosis                    0.4080423 -0.006521756           0.07797242
## radius_se                    0.5133464  0.240567362           0.22775353
## texture_se                   0.2302834  0.411620680           0.27972275
## perimeter_se                 0.5562641  0.266487092           0.24414277
```

```
## area_se                       0.4157296  0.134108980           0.12707090
## smoothness_se                 0.3284295  0.413506125           0.42737421
## compactness_se                0.7440827  0.394712835           0.80326882
## concavity_se                  0.7718040  0.309428578           0.72737218
## concave.points_se             1.0000000  0.312780223           0.61104414
## symmetry_se                   0.3127802  1.000000000           0.36907808
## fractal_dimension_se          0.6110441  0.369078083           1.00000000
##
## [[3]]
##                          diagnosis radius_worst texture_worst perimeter_worst
## diagnosis                1.0000000   0.77645378     0.4569028       0.7829141
## radius_worst             0.7764538   1.00000000     0.3599208       0.9937079
## texture_worst            0.4569028   0.35992075     1.0000000       0.3650982
## perimeter_worst          0.7829141   0.99370792     0.3650982       1.0000000
## area_worst               0.7338250   0.98401456     0.3458423       0.9775781
## smoothness_worst         0.4214649   0.21657443     0.2254294       0.2367746
## compactness_worst        0.5909982   0.47582004     0.3608323       0.5294077
## concavity_worst          0.6596102   0.57397471     0.3683656       0.6183441
## concave.points_worst     0.7935660   0.78742385     0.3597546       0.8163221
## symmetry_worst           0.4162943   0.24352920     0.2330275       0.2694928
## fractal_dimension_worst 0.3238722   0.09349198     0.2191224       0.1389569
##                          area_worst smoothness_worst compactness_worst
## diagnosis                0.73382503        0.4214649         0.5909982
## radius_worst             0.98401456        0.2165744         0.4758200
## texture_worst            0.34584228        0.2254294         0.3608323
## perimeter_worst          0.97757809        0.2367746         0.5294077
## area_worst               1.00000000        0.2091453         0.4382963
## smoothness_worst         0.20914533        1.0000000         0.5681865
## compactness_worst        0.43829628        0.5681865         1.0000000
## concavity_worst          0.54333053        0.5185233         0.8922609
## concave.points_worst     0.74741880        0.5476909         0.8010804
## symmetry_worst           0.20914551        0.4938383         0.6144405
## fractal_dimension_worst 0.07964703        0.6176242         0.8104549
##                          concavity_worst concave.points_worst symmetry_worst
## diagnosis                      0.6596102            0.7935660      0.4162943
## radius_worst                   0.5739747            0.7874239      0.2435292
## texture_worst                  0.3683656            0.3597546      0.2330275
## perimeter_worst                0.6183441            0.8163221      0.2694928
## area_worst                     0.5433305            0.7474188      0.2091455
## smoothness_worst               0.5185233            0.5476909      0.4938383
## compactness_worst              0.8922609            0.8010804      0.6144405
## concavity_worst                1.0000000            0.8554339      0.5325197
## concave.points_worst           0.8554339            1.0000000      0.5025285
## symmetry_worst                 0.5325197            0.5025285      1.0000000
## fractal_dimension_worst        0.6865109            0.5111141      0.5378482
##                          fractal_dimension_worst
## diagnosis                             0.32387219
## radius_worst                          0.09349198
## texture_worst                         0.21912243
## perimeter_worst                       0.13895686
## area_worst                            0.07964703
## smoothness_worst                      0.61762419
## compactness_worst                     0.81045486
## concavity_worst                       0.68651092
```

```
## concave.points_worst                0.51111415
## symmetry_worst                      0.53784821
## fractal_dimension_worst             1.00000000
```

```
#Using some linear model analysis on the subset data to look for predictors



modFNAmeans <- lm(diagnosis~.,data=FNAmeans)
modFNAse <- lm(diagnosis~.,data=FNAse)
modFNAworst <- lm(diagnosis~.,data=FNAworst)



summary(modFNAmeans)
```

```
##
## Call:
## lm(formula = diagnosis ~ ., data = FNAmeans)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6654 -0.1908 -0.0387  0.1806  0.8223
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -2.0520842  0.4166539  -4.925 1.11e-06 ***
## radius_mean             0.4900123  0.1312459   3.734 0.000208 ***
## texture_mean            0.0219732  0.0029228   7.518 2.23e-13 ***
## perimeter_mean         -0.0549747  0.0210018  -2.618 0.009095 **
## area_mean              -0.0009548  0.0002460  -3.881 0.000116 ***
## smoothness_mean         1.9408621  1.4107971   1.376 0.169460
## compactness_mean        0.0972608  1.0390787   0.094 0.925458
## concavity_mean          0.8097675  0.4953986   1.635 0.102702
## concave.points_mean     6.4310115  1.3855810   4.641 4.32e-06 ***
## symmetry_mean           1.0119000  0.5612933   1.803 0.071959 .
## fractal_dimension_mean -0.1192924  4.1578258  -0.029 0.977121
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.275 on 558 degrees of freedom
## Multiple R-squared:  0.6828, Adjusted R-squared:  0.6771
## F-statistic: 120.1 on 10 and 558 DF,  p-value: < 2.2e-16
```

```
summary(modFNAse)
```

```
##
## Call:
## lm(formula = diagnosis ~ ., data = FNAse)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -1.3878 -0.2614 -0.1039  0.2660  0.8530
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           1.472e-01  5.298e-02   2.778 0.005651 **
## radius_se             1.639e+00  2.946e-01   5.563 4.13e-08 ***
## texture_se           -3.779e-02  3.268e-02  -1.156 0.248048
## perimeter_se         -1.004e-01  3.845e-02  -2.610 0.009285 **
## area_se              -7.737e-04  1.305e-03  -0.593 0.553499
## smoothness_se        -2.262e+01  6.268e+00  -3.608 0.000336 ***
## compactness_se        1.067e+01  1.926e+00   5.542 4.62e-08 ***
## concavity_se         -1.194e+00  1.011e+00  -1.180 0.238338
## concave.points_se     1.900e+01  4.857e+00   3.912 0.000103 ***
## symmetry_se          -7.432e+00  2.293e+00  -3.241 0.001263 **
## fractal_dimension_se -5.788e+01  1.103e+01  -5.248 2.19e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.365 on 558 degrees of freedom
## Multiple R-squared:  0.441,  Adjusted R-squared:  0.431
## F-statistic: 44.02 on 10 and 558 DF,  p-value: < 2.2e-16
```

```
summary(modFNAworst)
```

```
##
## Call:
## lm(formula = diagnosis ~ ., data = FNAworst)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.56183 -0.18361 -0.02997  0.16305  1.00391
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -2.1742254  0.1946235 -11.171  < 2e-16 ***
## radius_worst             0.1271604  0.0253233   5.021 6.91e-07 ***
## texture_worst            0.0116868  0.0018968   6.161 1.38e-09 ***
## perimeter_worst         -0.0008583  0.0036411  -0.236  0.81373
## area_worst              -0.0005950  0.0001135  -5.243 2.24e-07 ***
## smoothness_worst         2.2087865  0.6759233   3.268  0.00115 **
## compactness_worst       -0.5944836  0.2249069  -2.643  0.00844 **
## concavity_worst          0.2538413  0.1329883   1.909  0.05681 .
## concave.points_worst     1.6807266  0.5225852   3.216  0.00137 **
## symmetry_worst           0.6100787  0.2222176   2.745  0.00624 **
## fractal_dimension_worst  2.5497367  1.3377504   1.906  0.05717 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.249 on 558 degrees of freedom
## Multiple R-squared:  0.7399, Adjusted R-squared:  0.7352
## F-statistic: 158.7 on 10 and 558 DF,  p-value: < 2.2e-16
```

Of the subset groups, the "worst" subset has the best R2 at 74. All three groups have a significant F test

p-value.

```
library(rms)
```

```
## Warning: package 'rms' was built under R version 4.0.4
```

```
## Loading required package: Hmisc
```

```
## Warning: package 'Hmisc' was built under R version 4.0.4
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: Formula
```

```
## Warning: package 'Formula' was built under R version 4.0.3
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
## Loading required package: SparseM
```

```
## Warning: package 'SparseM' was built under R version 4.0.3
```

```
##
## Attaching package: 'SparseM'
```

```
## The following object is masked from 'package:base':
##
##     backsolve
```

```r
#Checking each subset with fastbw



fbw <- function(df){

  ols.mod <- ols(diagnosis ~ ., data = df)
  #Perform p-value based selection using fastbw() function
  fastbw(ols.mod, rule = "p", sls = 0.05)


}



lapply(list(FNAmeans,FNAse,FNAworst),fbw)
```

```
## [[1]]
##
##  Deleted                Chi-Sq d.f. P      Residual d.f. P      AIC   R2
##  fractal_dimension_mean 0.00   1    0.9771 0.00     1    0.9771 -2.00 0.683
##  compactness_mean       0.01   1    0.9245 0.01     2    0.9951 -3.99 0.683
##  smoothness_mean        2.24   1    0.1343 2.25     3    0.5218 -3.75 0.681
##  concavity_mean         1.97   1    0.1606 4.22     4    0.3770 -3.78 0.680
##  symmetry_mean          4.83   1    0.0280 9.05     5    0.1071 -0.95 0.678
##
## Approximate Estimates after Deleting Factors
##
##                          Coef      S.E. Wald Z         P
## Intercept           -1.538645 0.1668871 -9.220 0.000e+00
## radius_mean          0.373726 0.0812297  4.601 4.208e-06
## texture_mean         0.021679 0.0028558  7.591 3.175e-14
## perimeter_mean      -0.039122 0.0124980 -3.130 1.746e-03
## area_mean           -0.000974 0.0002086 -4.669 3.021e-06
## concave.points_mean  9.182604 0.8972126 10.235 0.000e+00
##
## Factors in Final Model
##
## [1] radius_mean         texture_mean        perimeter_mean
## [4] area_mean           concave.points_mean
##
## [[2]]
##
##  Deleted      Chi-Sq d.f. P      Residual d.f. P      AIC   R2
##  area_se      0.35   1    0.5533 0.35     1    0.5533 -1.65 0.441
##  texture_se   1.13   1    0.2875 1.48     2    0.4764 -2.52 0.440
##  concavity_se 1.69   1    0.1938 3.17     3    0.3660 -2.83 0.438
##
## Approximate Estimates after Deleting Factors
##
##                         Coef     S.E. Wald Z         P
## Intercept             0.1493  0.04997  2.988 2.806e-03
## radius_se             1.5272  0.25444  6.002 1.945e-09
```

```
## perimeter_se          -0.1031  0.03651 -2.824 4.744e-03
## smoothness_se        -23.1088  6.02322 -3.837 1.247e-04
## compactness_se         9.9635  1.81311  5.495 3.902e-08
## concave.points_se     17.0410  4.12392  4.132 3.593e-05
## symmetry_se           -7.7804  2.16058 -3.601 3.169e-04
## fractal_dimension_se -61.2282 10.55742 -5.800 6.650e-09
##
## Factors in Final Model
##
## [1] radius_se            perimeter_se         smoothness_se
## [4] compactness_se       concave.points_se    symmetry_se
## [7] fractal_dimension_se
##
## [[3]]
##
##  Deleted         Chi-Sq d.f. P      Residual d.f. P      AIC   R2
##  perimeter_worst 0.06   1    0.8136 0.06     1    0.8136 -1.94 0.740
##  concavity_worst 3.63   1    0.0569 3.68     2    0.1587 -0.32 0.738
##
## Approximate Estimates after Deleting Factors
##
##                              Coef      S.E.  Wald Z         P
## Intercept               -2.1699025 0.1878526 -11.551 0.000e+00
## radius_worst             0.1189136 0.0147940   8.038 8.882e-16
## texture_worst            0.0120057 0.0018876   6.360 2.013e-10
## area_worst              -0.0005774 0.0001104  -5.230 1.696e-07
## smoothness_worst         2.0488494 0.6619589   3.095 1.967e-03
## compactness_worst       -0.4474573 0.1889594  -2.368 1.788e-02
## concave.points_worst     2.0764539 0.4672298   4.444 8.823e-06
## symmetry_worst           0.5958575 0.2212429   2.693 7.076e-03
## fractal_dimension_worst  2.8559367 1.2999911   2.197 2.803e-02
##
## Factors in Final Model
##
## [1] radius_worst            texture_worst           area_worst
## [4] smoothness_worst        compactness_worst       concave.points_worst
## [7] symmetry_worst          fractal_dimension_worst
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
#Evaluate automated selection across 3 subcategories with AIC - running against trained lm's



aic <- function (mod){
```

```
  mod_result <- stepAIC(mod,trace=FALSE)
  return (mod_result$anova)

}
```

```
lapply(list(modFNAmeans,modFNAse,modFNAworst),aic)
```

```
## [[1]]
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## diagnosis ~ radius_mean + texture_mean + perimeter_mean + area_mean +
##     smoothness_mean + compactness_mean + concavity_mean + concave.points_mean +
##     symmetry_mean + fractal_dimension_mean
##
## Final Model:
## diagnosis ~ radius_mean + texture_mean + perimeter_mean + area_mean +
##     smoothness_mean + concavity_mean + concave.points_mean +
##     symmetry_mean
##
##
##                       Step Df    Deviance Resid. Df Resid. Dev       AIC
## 1                                                558   42.19628 -1458.281
## 2 - fractal_dimension_mean  1 6.22491e-05       559   42.19634 -1460.280
## 3       - compactness_mean  1 6.79722e-04       560   42.19702 -1462.271
##
## [[2]]
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## diagnosis ~ radius_se + texture_se + perimeter_se + area_se +
##     smoothness_se + compactness_se + concavity_se + concave.points_se +
##     symmetry_se + fractal_dimension_se
##
## Final Model:
## diagnosis ~ radius_se + perimeter_se + smoothness_se + compactness_se +
##     concave.points_se + symmetry_se + fractal_dimension_se
##
##
##             Step Df   Deviance Resid. Df Resid. Dev       AIC
## 1                                    558   74.35249 -1135.951
## 2      - area_se  1 0.04683815       559   74.39933 -1137.593
## 3   - texture_se  1 0.15074491       560   74.55007 -1138.441
## 4 - concavity_se  1 0.22498469       561   74.77506 -1138.726
##
```

```
## [[3]]
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## diagnosis ~ radius_worst + texture_worst + perimeter_worst +
##     area_worst + smoothness_worst + compactness_worst + concavity_worst +
##     concave.points_worst + symmetry_worst + fractal_dimension_worst
##
## Final Model:
## diagnosis ~ radius_worst + texture_worst + area_worst + smoothness_worst +
##     compactness_worst + concavity_worst + concave.points_worst +
##     symmetry_worst + fractal_dimension_worst
##
##
##                Step Df    Deviance Resid. Df Resid. Dev       AIC
## 1                                        558   34.59533 -1571.292
## 2 - perimeter_worst  1 0.003445163       559   34.59877 -1573.235
```

```
#Re-load, format, and scrub data, then re-subset for dt analysis (returning response to factors). This
```

```
FNAdt <- read.csv("FNA_cancer.csv", header = T)
```

```
FNAdt$diagnosis <- as.factor(FNAdt$diagnosis)
FNAdt <- FNAdt %>% dplyr::select(-X)
```

```
FNA_trim <- FNAdt %>% dplyr::select(-id)
```

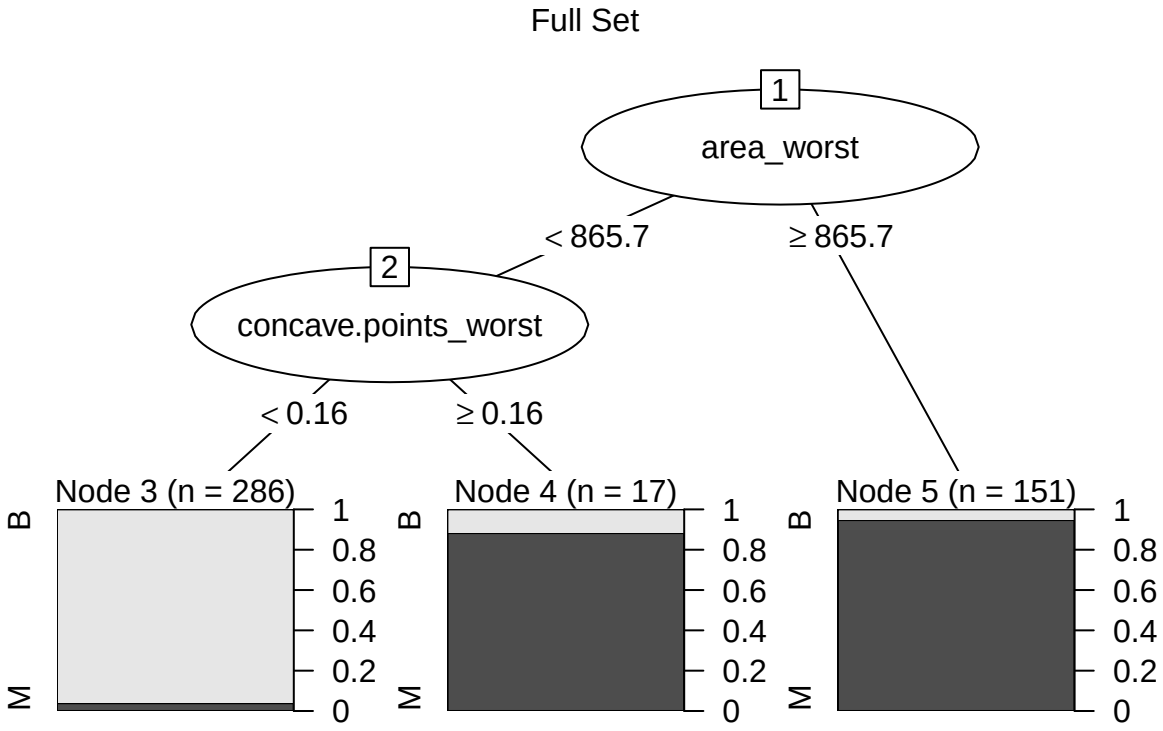
```
#Breaking out the predictor columns into 3 subcategories and adding the diagnosis column
FNAmeans <- dplyr::select(FNA_trim,contains("mean")) %>% mutate(diagnosis = FNA_trim$diagnosis) %>% rel
```

```
FNAse <- dplyr::select(FNA_trim,contains("_se")) %>% mutate(diagnosis = FNA_trim$diagnosis) %>% relocat
```

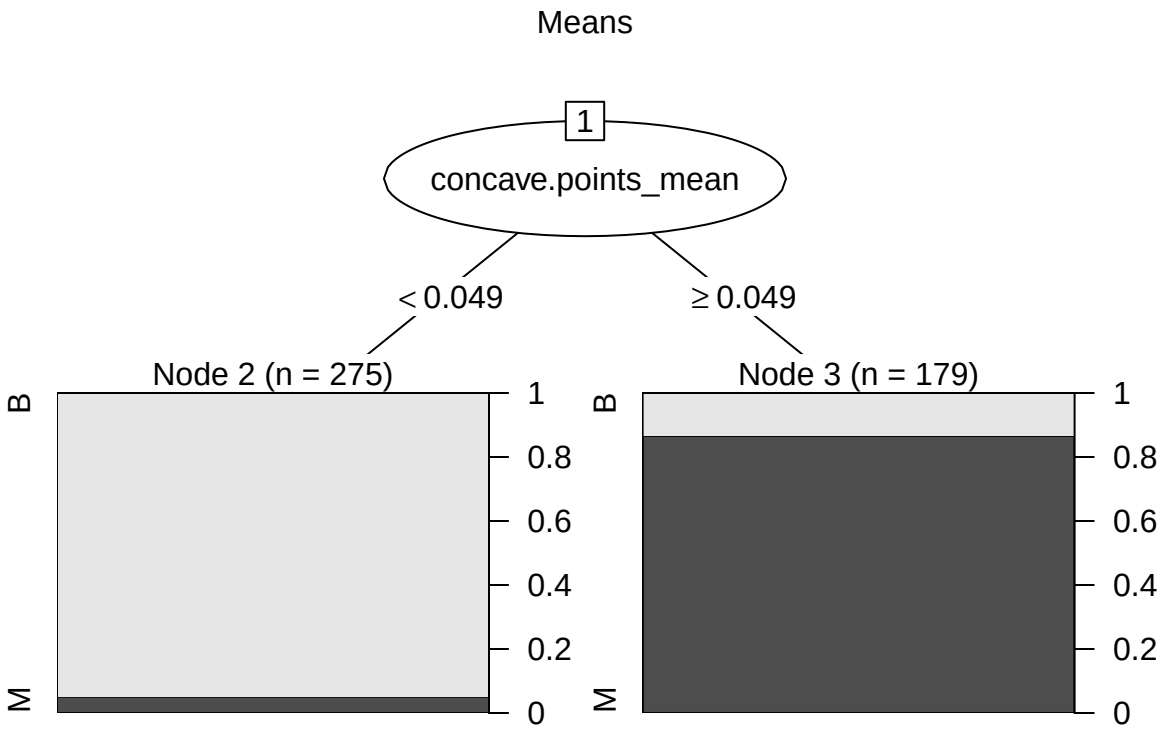
```
FNAworst <- dplyr::select(FNA_trim,contains("worst")) %>% mutate(diagnosis = FNA_trim$diagnosis) %>% re
```

```r
#Create splits of full dataset using caret
set.seed(1842)
test_index <- createDataPartition(FNA_trim$diagnosis,p=0.2,list = F)
train_FNA <- FNA_trim[-test_index,]
test_FNA <- FNA_trim[test_index,]



#Create splits of subsets using caret
set.seed(1842)
n <- nrow(FNAmeans)
test_idx <- createDataPartition(FNAmeans$diagnosis,p=0.2,list = F)
test_FNAmeans <- FNAmeans[test_idx,]
train_FNAmeans <- FNAmeans[-test_idx,]
test_FNAse <- FNAse[test_idx,]
train_FNAse <- FNAse[-test_idx,]
test_FNAworst <- FNAworst[test_idx,]
train_FNAworst <- FNAworst[-test_idx,]
```

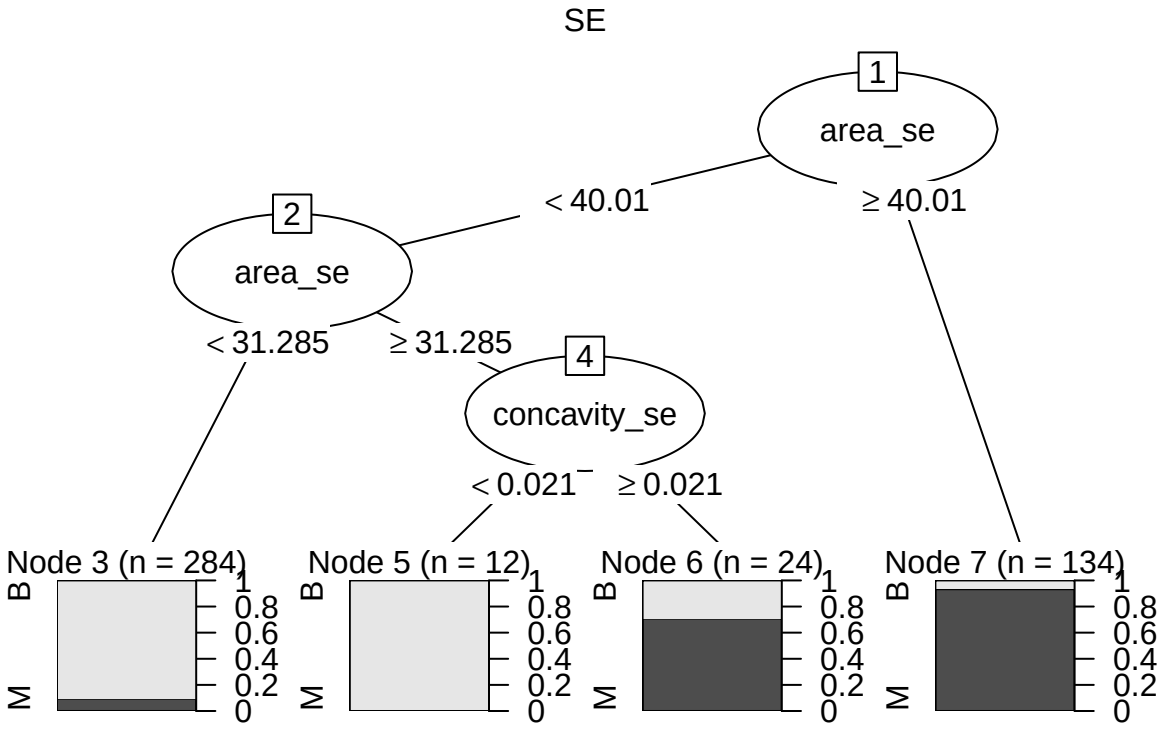
```r
#Look at the subset data with some optimized dt's, using caret and cross validation



#means
fnameans_caret <- train(diagnosis~., data=train_FNAmeans, method="rpart", trControl = trainControl(metho



#se
fnase_caret <- train(diagnosis~., data=train_FNAse, method="rpart", trControl = trainControl(method = "



#worst
fnaworst_caret <- train(diagnosis~., data=train_FNAworst, method="rpart", trControl = trainControl(metho




#fit a tree - using caret and full dataset
fnaFullTree_caret <- train(diagnosis~., data=train_FNA, method="rpart", trControl = trainControl(method




#All summaries
plot(as.party(fnaFullTree_caret$finalModel),main="Full Set")
```



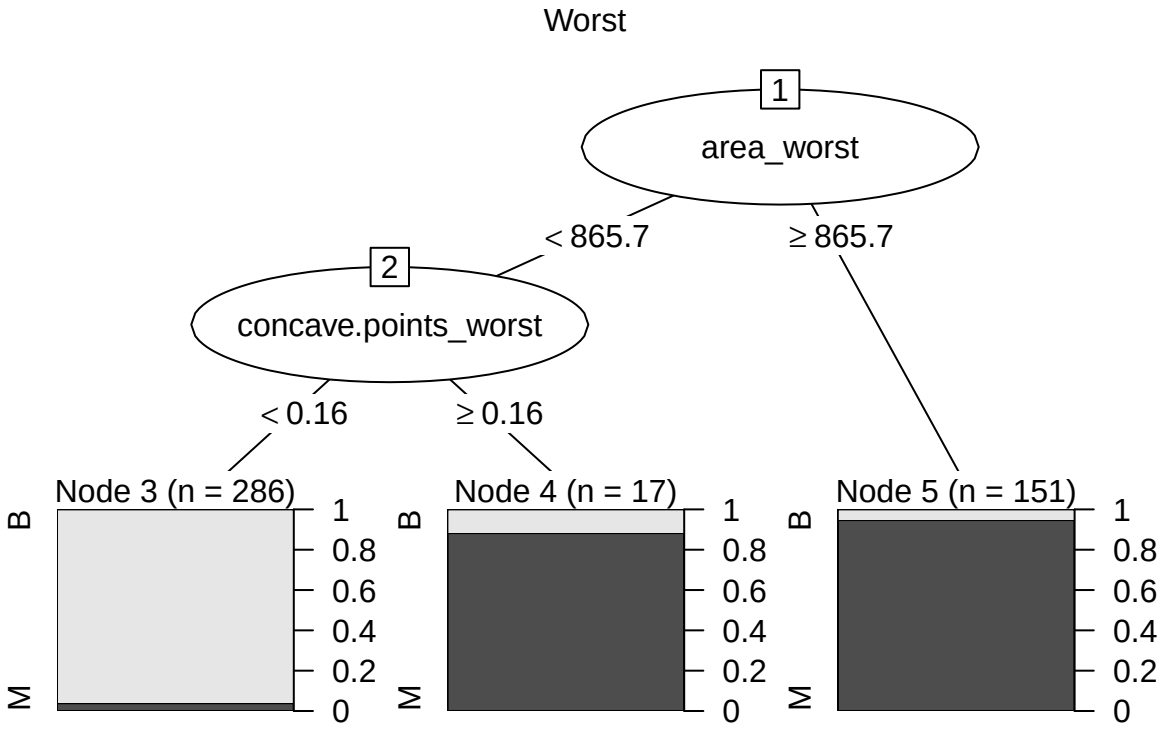

```
plot(as.party(fnameans_caret$finalModel),main="Means")
```




```
plot(as.party(fnase_caret$finalModel),main="SE")
```




```
plot(as.party(fnaworst_caret$finalModel),main="Worst")
```



The "Worst" and "Full" models optimize to the same nodes. Both settle on area_worst and concave.points_worst.

```
#Evaluate each of the dt's
fnaFullTree_caret
```

```
## CART
##
## 454 samples
##  30 predictor
##   2 classes: 'B', 'M'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 409, 408, 409, 409, 409, 409, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.02366864  0.9228986  0.8345286
##   0.07692308  0.9075845  0.8006534
##   0.79881657  0.7816425  0.4481444
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.02366864.
```

```
fnameans_caret
```

```
## CART
##
## 454 samples
##  10 predictor
##   2 classes: 'B', 'M'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 409, 408, 409, 409, 410, 409, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.02958580  0.8965086  0.7794809
##   0.03846154  0.8986825  0.7846520
##   0.77514793  0.7748199  0.4427292
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.03846154.
```

```
fnase_caret
```

```
## CART
##
## 454 samples
##  10 predictor
##   2 classes: 'B', 'M'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 408, 409, 409, 408, 409, 408, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.02071006  0.8811989  0.7416514
##   0.02958580  0.8657400  0.7023811
##   0.68639053  0.7840711  0.4613050
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.02071006.
```

```
fnaworst_caret
```

```
## CART
##
## 454 samples
##  10 predictor
##   2 classes: 'B', 'M'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
```

```
## Summary of sample sizes: 409, 408, 408, 408, 409, 409, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.02366864  0.9209662  0.8316738
##   0.07692308  0.8921739  0.7678456
##   0.79881657  0.7787923  0.4395550
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.02366864.
```

The full model and worst model optimized to the same nodes. I'll look at confusion matrices for both.

```
#FULL
print("####FULL MODEL CONFUSION MATRIX####")
```

```
## [1] "####FULL MODEL CONFUSION MATRIX####"
```

```
#Predict on test data
rpartEval_pred1 <- predict(fnaFullTree_caret$finalModel,newdata = test_FNA,type="class")
#make the confusion matrix
confusionMatrix(rpartEval_pred1, test_FNA$diagnosis)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 69  5
##          M  3 38
##
##                Accuracy : 0.9304
##                  95% CI : (0.8675, 0.9695)
##     No Information Rate : 0.6261
##     P-Value [Acc > NIR] : 4.471e-14
##
##                   Kappa : 0.85
##
##  Mcnemar's Test P-Value : 0.7237
##
##             Sensitivity : 0.9583
##             Specificity : 0.8837
##          Pos Pred Value : 0.9324
##          Neg Pred Value : 0.9268
##              Prevalence : 0.6261
##          Detection Rate : 0.6000
##    Detection Prevalence : 0.6435
##       Balanced Accuracy : 0.9210
##
##        'Positive' Class : B
##
```

```r
#Worst
print("####Worst MODEL CONFUSION MATRIX####")
```

```
## [1] "####Worst MODEL CONFUSION MATRIX####"
```

```r
#Predict on test data
rpartEval_pred2 <- predict(fnaworst_caret$finalModel,newdata = test_FNAworst,type="class")
#make the confusion matrix
confusionMatrix(rpartEval_pred2, test_FNAworst$diagnosis)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 69  5
##          M  3 38
##
##                Accuracy : 0.9304
##                  95% CI : (0.8675, 0.9695)
##     No Information Rate : 0.6261
##     P-Value [Acc > NIR] : 4.471e-14
##
##                   Kappa : 0.85
##
##  Mcnemar's Test P-Value : 0.7237
##
##             Sensitivity : 0.9583
##             Specificity : 0.8837
##          Pos Pred Value : 0.9324
##          Neg Pred Value : 0.9268
##              Prevalence : 0.6261
##          Detection Rate : 0.6000
##    Detection Prevalence : 0.6435
##       Balanced Accuracy : 0.9210
##
##        'Positive' Class : B
##
```

```r
#se
print("####se MODEL CONFUSION MATRIX####")
```

```
## [1] "####se MODEL CONFUSION MATRIX####"
```

```r
#Predict on test data
rpartEval_pred3 <- predict(fnase_caret$finalModel,newdata = test_FNAse,type="class")
#make the confusion matrix
confusionMatrix(rpartEval_pred3, test_FNAse$diagnosis)
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction  B  M
##          B 67  9
##          M  5 34
##
##                Accuracy : 0.8783
##                  95% CI : (0.8042, 0.9318)
##     No Information Rate : 0.6261
##     P-Value [Acc > NIR] : 1.383e-09
##
##                   Kappa : 0.735
##
##  Mcnemar's Test P-Value : 0.4227
##
##             Sensitivity : 0.9306
##             Specificity : 0.7907
##          Pos Pred Value : 0.8816
##          Neg Pred Value : 0.8718
##              Prevalence : 0.6261
##          Detection Rate : 0.5826
##    Detection Prevalence : 0.6609
##       Balanced Accuracy : 0.8606
##
##        'Positive' Class : B
##
```

```
#means
print("####means MODEL CONFUSION MATRIX####")
```

```
## [1] "####means MODEL CONFUSION MATRIX####"
```

```
#Predict on test data
rpartEval_pred4 <- predict(fnameans_caret$finalModel,newdata = test_FNAmeans,type="class")
#make the confusion matrix
confusionMatrix(rpartEval_pred4, test_FNAmeans$diagnosis)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 65  4
##          M  7 39
##
##                Accuracy : 0.9043
##                  95% CI : (0.8353, 0.9513)
##     No Information Rate : 0.6261
##     P-Value [Acc > NIR] : 1.215e-11
##
##                   Kappa : 0.7985
##
##  Mcnemar's Test P-Value : 0.5465
##
##             Sensitivity : 0.9028
##             Specificity : 0.9070
```

```
##          Pos Pred Value : 0.9420
##          Neg Pred Value : 0.8478
##              Prevalence : 0.6261
##          Detection Rate : 0.5652
##    Detection Prevalence : 0.6000
##       Balanced Accuracy : 0.9049
##
##        'Positive' Class : B
##
```

Unsurprisingly, the full and worst models are effectively identical, given how caret optimized them. They are also the most powerful decision tree models, yielding 93% accuracy and 96% sensitivity.

```
library(neuralnet)
```

```
## Warning: package 'neuralnet' was built under R version 4.0.4
```

```
##
## Attaching package: 'neuralnet'
```

```
## The following object is masked from 'package:dplyr':
##
##     compute
```

#ROC Plot of the decision trees #Calculate predictions using probabilities fnafull_pred_prob <- predict(fnaFullTree_caret$finalModel, newdata = $test_FNA$, type = "prob")$fnameans_pred_prob$ <- $-predict(fnameans_caret$finalModel,newdata = test_FNAmeans,type="prob")$ fnase_pred_prob <- predict(fnase_caret$finalModel,newdata = test_FNAse,type="prob")

#Provide ROC with predictions and truth for analysis roc_fnafullpreds <- prediction(fnafull_pred_prob[,2],test_FNA$diagnc$ $-prediction(fnameans_pred_prob[,2], test_FNAmeans$diagnosis)$ roc_fnasepreds <- prediction(fnase_pred_prob[,2],test_FNA

#Calculate true and false positive rates from ROC analysis roc_fnafullPerf1 <- performance(roc_fnafullpreds,"tpr","fpr") roc_fnameansPerf1 <- performance(roc_fnameanspreds,"tpr","fpr") roc_fnasePerf1 <- performance(roc_fnasepreds,"tpr","f

#Plot all three models using this ROC analysis, along with the a/b line as reference plot(roc_fnafullPerf1, col="blue") plot(roc_fnameansPerf1,add=T, col="red") plot(roc_fnasePerf1,add=T, col="orange") abline(a=0,b=1)

legend("bottomright", legend=c("FULL/Worst", "Means", "SE", "Random Chance"), col=c("blue", "red", "orange", "black"), lty=1:1)

The Full tree which reduces to area_worst and concave.points_worst nodes is the best decision tree model evaluated. It minimizes FPR and maximizes TPR.

Create a bagging prediction.

```
bagging <- randomForest(diagnosis ~., data=train_FNA, mtry=30, ntree=500)
bagging
```

```
##
## Call:
##  randomForest(formula = diagnosis ~ ., data = train_FNA, mtry = 30,      ntree = 500)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 30
##
##         OOB estimate of  error rate: 3.52%
## Confusion matrix:
##     B   M class.error
## B 277   8  0.02807018
## M   8 161  0.04733728
```

Predict the diagnosis using bagging and the test data, and then view results in a confusion matrix.

```
bagging_prediction <- predict(bagging, newdata=test_FNA[-1], "class")
confusionMatrix(bagging_prediction, test_FNA$diagnosis, positive="M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 71  4
##          M  1 39
##
##                Accuracy : 0.9565
##                  95% CI : (0.9015, 0.9857)
##     No Information Rate : 0.6261
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9058
##
##  Mcnemar's Test P-Value : 0.3711
##
##             Sensitivity : 0.9070
##             Specificity : 0.9861
##          Pos Pred Value : 0.9750
##          Neg Pred Value : 0.9467
##              Prevalence : 0.3739
##          Detection Rate : 0.3391
##    Detection Prevalence : 0.3478
##       Balanced Accuracy : 0.9465
##
##        'Positive' Class : M
##
```

Create the random forest.

```
rf4 <- randomForest(diagnosis ~., data=train_FNA, mtry=4, ntree=1000)
rf4
```

```
##
## Call:
```

```
##  randomForest(formula = diagnosis ~ ., data = train_FNA, mtry = 4,      ntree = 1000)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 3.74%
## Confusion matrix:
##     B   M class.error
## B 278   7   0.0245614
## M  10 159   0.0591716
```

Predict the diagnosis using random forest and the test data, and then view results in a confusion matrix.

```
rf_prediction <- predict(rf4, newdata=test_FNA[-1], "class")
confusionMatrix(rf_prediction, test_FNA$diagnosis, positive="M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 70  4
##          M  2 39
##
##                Accuracy : 0.9478
##                  95% CI : (0.8899, 0.9806)
##     No Information Rate : 0.6261
##     P-Value [Acc > NIR] : 5.754e-16
##
##                   Kappa : 0.8875
##
##  Mcnemar's Test P-Value : 0.6831
##
##             Sensitivity : 0.9070
##             Specificity : 0.9722
##          Pos Pred Value : 0.9512
##          Neg Pred Value : 0.9459
##              Prevalence : 0.3739
##          Detection Rate : 0.3391
##    Detection Prevalence : 0.3565
##       Balanced Accuracy : 0.9396
##
##        'Positive' Class : M
##
```

```
rf <- randomForest(diagnosis ~., data=train_FNA, mtry=6, ntree=1000)
rf
```

```
##
## Call:
##  randomForest(formula = diagnosis ~ ., data = train_FNA, mtry = 6,      ntree = 1000)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 6
```

```
##
##         OOB estimate of  error rate: 3.74%
## Confusion matrix:
##     B   M class.error
## B 277   8  0.02807018
## M   9 160  0.05325444
```

Predict the diagnosis using random forest and the test data, and then view results in a confusion matrix.

```
rf_prediction <- predict(rf, newdata=test_FNA[-1], "class")
confusionMatrix(rf_prediction, test_FNA$diagnosis, positive="M")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 71  4
##          M  1 39
##
##                Accuracy : 0.9565
##                  95% CI : (0.9015, 0.9857)
##     No Information Rate : 0.6261
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9058
##
##  Mcnemar's Test P-Value : 0.3711
##
##             Sensitivity : 0.9070
##             Specificity : 0.9861
##          Pos Pred Value : 0.9750
##          Neg Pred Value : 0.9467
##              Prevalence : 0.3739
##          Detection Rate : 0.3391
##    Detection Prevalence : 0.3478
##       Balanced Accuracy : 0.9465
##
##        'Positive' Class : M
##
```

```
library(MASS)
library(rms)
```

```
#checking to see what the most influential predictors are
ols.fna <- ols(diagnosis ~., data=train_FNA)
```

```
## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
## Warning in Ops.factor(Y, r): '-' not meaningful for factors
```

```
## Warning in mean.default(Y): argument is not numeric or logical: returning NA
```

```
## Warning in Ops.factor(Y, mean(Y)): '-' not meaningful for factors

## Warning in Ops.factor(Y, fit$residuals): '-' not meaningful for factors
```

```
#Perform p-value based selection using fastbw() function
fastbw(ols.fna, rule = "p", sls = 0.05)
```

```
##
##  Deleted                Chi-Sq d.f. P      Residual d.f. P      AIC    R2
##  texture_mean           0.00   1    0.9704  0.00     1   0.9704  -2.00 NA
##  compactness_se         0.01   1    0.9318  0.01     2   0.9957  -3.99 NA
##  area_mean              0.03   1    0.8583  0.04     3   0.9979  -5.96 NA
##  smoothness_mean        0.05   1    0.8185  0.09     4   0.9989  -7.91 NA
##  concave.points_worst   0.05   1    0.8209  0.14     5   0.9996  -9.86 NA
##  symmetry_se            0.07   1    0.7968  0.21     6   0.9998 -11.79 NA
##  symmetry_mean          0.04   1    0.8490  0.25     7   0.9999 -13.75 NA
##  texture_se             0.09   1    0.7670  0.33     8   1.0000 -15.67 NA
##  perimeter_se           0.11   1    0.7354  0.45     9   1.0000 -17.55 NA
##  fractal_dimension_mean 0.08   1    0.7819  0.53    10   1.0000 -19.47 NA
##  fractal_dimension_se   0.41   1    0.5241  0.93    11   1.0000 -21.07 NA
##  concave.points_mean    0.66   1    0.4158  1.59    12   0.9998 -22.41 NA
##  compactness_worst      0.61   1    0.4348  2.20    13   0.9996 -23.80 NA
##  perimeter_worst        0.57   1    0.4504  2.77    14   0.9994 -25.23 NA
##  perimeter_mean         0.45   1    0.5044  3.22    15   0.9994 -26.78 NA
##  radius_mean            0.06   1    0.8138  3.27    16   0.9997 -28.73 NA
##  smoothness_worst       2.46   1    0.1169  5.73    17   0.9948 -28.27 NA
##  area_se                5.23   1    0.0221 10.97    18   0.8957 -25.03 NA
##  concavity_worst        5.55   1    0.0184 16.52    19   0.6223 -21.48 NA
##  radius_se              4.48   1    0.0342 21.00    20   0.3970 -19.00 NA
##
## Approximate Estimates after Deleting Factors
##
##                              Coef      S.E. Wald Z         P
## Intercept                -1.3745512 0.1719503 -7.994 1.332e-15
## compactness_mean         -3.2683809 0.6346765 -5.150 2.609e-07
## concavity_mean            3.1979936 0.5064457  6.315 2.709e-10
## smoothness_se            18.2732109 4.5175326  4.045 5.233e-05
## concavity_se             -4.7560059 0.7874697 -6.040 1.545e-09
## concave.points_se        14.4891152 3.5771810  4.050 5.112e-05
## radius_worst              0.1400899 0.0147726  9.483 0.000e+00
## texture_worst             0.0109612 0.0020407  5.371 7.815e-08
## area_worst               -0.0007566 0.0001151 -6.571 4.989e-11
## symmetry_worst            1.2122421 0.2324337  5.215 1.834e-07
## fractal_dimension_worst   4.9059089 1.0666210  4.599 4.235e-06
##
## Factors in Final Model
##
##  [1] compactness_mean        concavity_mean          smoothness_se
##  [4] concavity_se            concave.points_se       radius_worst
##  [7] texture_worst           area_worst              symmetry_worst
## [10] fractal_dimension_worst
```

```r
set.seed(1842)
FNA_knn <- FNA_trim
FNA_knn$diagnosis <- as.factor(ifelse(FNA_knn$diagnosis=="M",1,0))
n <- nrow(FNA_knn)
test_idx <- sample.int(n, size=(n*.2))
test_FNA_knn <- FNA_knn[test_idx,]
train_FNA_knn <- FNA_knn[-test_idx,]
glimpse(train_FNA_knn)
```

```
## Rows: 456
## Columns: 31
## $ diagnosis               <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ radius_mean             <dbl> 17.990, 20.570, 19.690, 11.420, 12.450, 18....
## $ texture_mean            <dbl> 10.38, 17.77, 21.25, 20.38, 15.70, 19.98, 2...
## $ perimeter_mean          <dbl> 122.80, 132.90, 130.00, 77.58, 82.57, 119.6...
## $ area_mean               <dbl> 1001.0, 1326.0, 1203.0, 386.1, 477.1, 1040....
## $ smoothness_mean         <dbl> 0.11840, 0.08474, 0.10960, 0.14250, 0.12780...
## $ compactness_mean        <dbl> 0.27760, 0.07864, 0.15990, 0.28390, 0.17000...
## $ concavity_mean          <dbl> 0.30010, 0.08690, 0.19740, 0.24140, 0.15780...
## $ concave.points_mean     <dbl> 0.14710, 0.07017, 0.12790, 0.10520, 0.08089...
## $ symmetry_mean           <dbl> 0.2419, 0.1812, 0.2069, 0.2597, 0.2087, 0.1...
## $ fractal_dimension_mean  <dbl> 0.07871, 0.05667, 0.05999, 0.09744, 0.07613...
## $ radius_se               <dbl> 1.0950, 0.5435, 0.7456, 0.4956, 0.3345, 0.4...
## $ texture_se              <dbl> 0.9053, 0.7339, 0.7869, 1.1560, 0.8902, 0.7...
## $ perimeter_se            <dbl> 8.589, 3.398, 4.585, 3.445, 2.217, 3.180, 2...
## $ area_se                 <dbl> 153.40, 74.08, 94.03, 27.23, 27.19, 53.91, ...
## $ smoothness_se           <dbl> 0.006399, 0.005225, 0.006150, 0.009110, 0.0...
## $ compactness_se          <dbl> 0.049040, 0.013080, 0.040060, 0.074580, 0.0...
## $ concavity_se            <dbl> 0.05373, 0.01860, 0.03832, 0.05661, 0.03672...
## $ concave.points_se       <dbl> 0.015870, 0.013400, 0.020580, 0.018670, 0.0...
## $ symmetry_se             <dbl> 0.03003, 0.01389, 0.02250, 0.05963, 0.02165...
## $ fractal_dimension_se    <dbl> 0.006193, 0.003532, 0.004571, 0.009208, 0.0...
## $ radius_worst            <dbl> 25.38, 24.99, 23.57, 14.91, 15.47, 22.88, 1...
## $ texture_worst           <dbl> 17.33, 23.41, 25.53, 26.50, 23.75, 27.66, 3...
## $ perimeter_worst         <dbl> 184.60, 158.80, 152.50, 98.87, 103.40, 153....
## $ area_worst              <dbl> 2019.0, 1956.0, 1709.0, 567.7, 741.6, 1606....
## $ smoothness_worst        <dbl> 0.1622, 0.1238, 0.1444, 0.2098, 0.1791, 0.1...
## $ compactness_worst       <dbl> 0.6656, 0.1866, 0.4245, 0.8663, 0.5249, 0.2...
## $ concavity_worst         <dbl> 0.71190, 0.24160, 0.45040, 0.68690, 0.53550...
## $ concave.points_worst    <dbl> 0.26540, 0.18600, 0.24300, 0.25750, 0.17410...
## $ symmetry_worst          <dbl> 0.4601, 0.2750, 0.3613, 0.6638, 0.3985, 0.3...
## $ fractal_dimension_worst <dbl> 0.11890, 0.08902, 0.08758, 0.17300, 0.12440...
```

```r
library(caret)

rescale_x <- function(x) {
  (x-min(x))/(max(x)-min(x))
}
pred_knn<-knn(sapply(train_FNA_knn %>% dplyr::select(-diagnosis), rescale_x),sapply(test_FNA_knn %>% dpl
 , train_FNA_knn$diagnosis)
confusionMatrix(pred_knn, test_FNA_knn$diagnosis)
```

```
## Confusion Matrix and Statistics
```

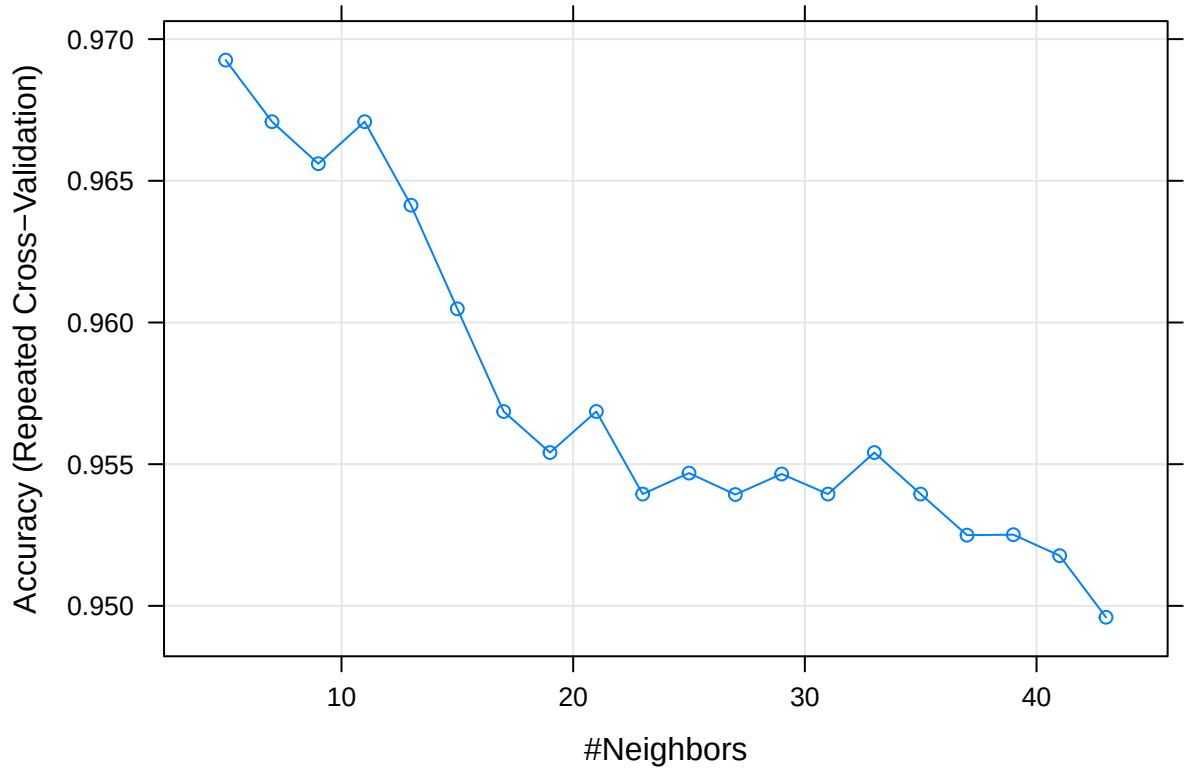
```
##
##          Reference
## Prediction  0  1
##          0 69  9
##          1  1 34
##
##                Accuracy : 0.9115
##                  95% CI : (0.8433, 0.9567)
##     No Information Rate : 0.6195
##     P-Value [Acc > NIR] : 1.767e-12
##
##                   Kappa : 0.8053
##
##  Mcnemar's Test P-Value : 0.02686
##
##             Sensitivity : 0.9857
##             Specificity : 0.7907
##          Pos Pred Value : 0.8846
##          Neg Pred Value : 0.9714
##              Prevalence : 0.6195
##          Detection Rate : 0.6106
##    Detection Prevalence : 0.6903
##       Balanced Accuracy : 0.8882
##
##        'Positive' Class : 0
##
```

```
#Using Caret more
train_proc_knn <- preProcess(x=train_FNA_knn %>% dplyr::select(-diagnosis), method=c("center", "scale"))
test_proc_knn <- preProcess(x=test_FNA_knn %>% dplyr::select(-diagnosis), method=c("center", "scale"))

set.seed(1842)
ctrl <- trainControl(method="repeatedcv",repeats = 3) #,classProbs=TRUE,summaryFunction = twoClassSumma
knnFit <- train(diagnosis ~ ., data = train_FNA_knn, method = "knn", trControl = ctrl, preProcess = c("
knnFit
```

```
## k-Nearest Neighbors
##
## 456 samples
##  30 predictor
##   2 classes: '0', '1'
##
## Pre-processing: centered (30), scaled (30)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 410, 410, 410, 410, 411, 410, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.9692593  0.9328621
##    7  0.9670853  0.9281792
##    9  0.9656039  0.9249048
##   11  0.9670853  0.9281179
##   13  0.9641385  0.9214650
##   15  0.9604831  0.9128932
```

```
##    17  0.9568599  0.9045859
##    19  0.9554106  0.9015159
##    21  0.9568599  0.9045859
##    23  0.9539452  0.8983461
##    25  0.9546860  0.8999024
##    27  0.9539291  0.8983662
##    29  0.9546538  0.8999788
##    31  0.9539452  0.8983451
##    33  0.9554106  0.9015517
##    35  0.9539452  0.8983457
##    37  0.9524960  0.8951187
##    39  0.9525121  0.8950995
##    41  0.9517713  0.8935054
##    43  0.9495974  0.8887802
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

```
plot(knnFit)
```




```
knnPredict <- predict(knnFit,newdata = test_FNA_knn)
#Get the confusion matrix to see accuracy value and other parameter values
confusionMatrix(knnPredict, test_FNA_knn$diagnosis)
```

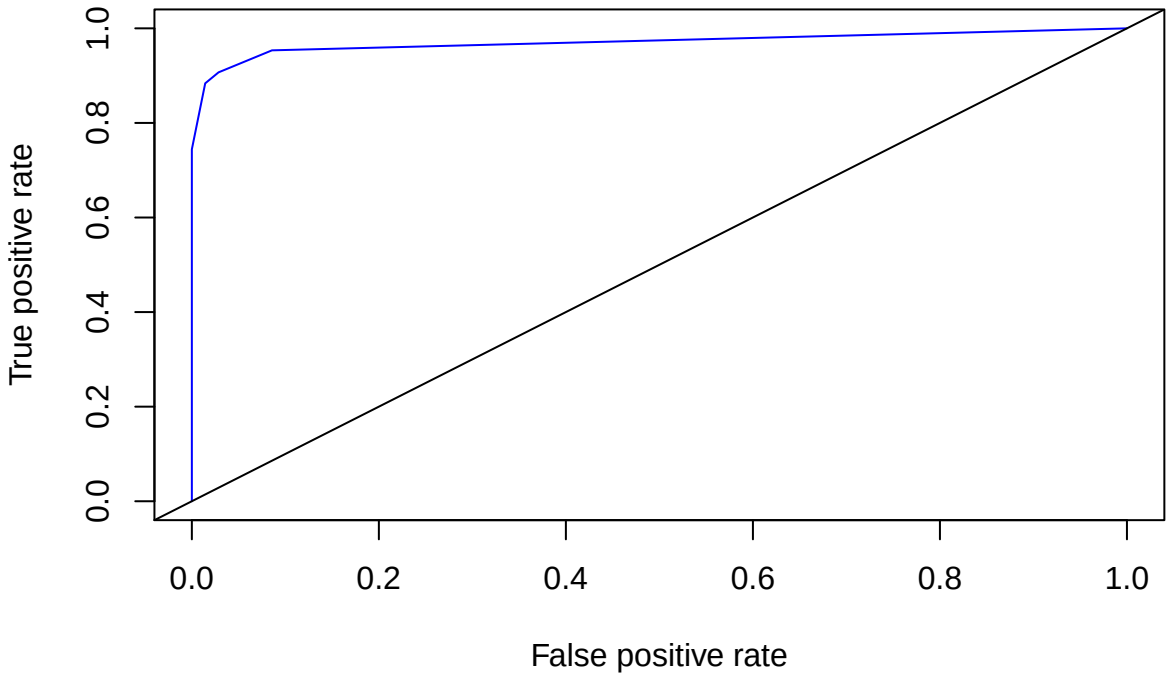
```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction  0  1
##          0 69  5
##          1  1 38
##
##                Accuracy : 0.9469
##                  95% CI : (0.888, 0.9803)
##     No Information Rate : 0.6195
##     P-Value [Acc > NIR] : 4.693e-16
##
##                   Kappa : 0.8853
##
##  Mcnemar's Test P-Value : 0.2207
##
##             Sensitivity : 0.9857
##             Specificity : 0.8837
##          Pos Pred Value : 0.9324
##          Neg Pred Value : 0.9744
##              Prevalence : 0.6195
##          Detection Rate : 0.6106
##    Detection Prevalence : 0.6549
##       Balanced Accuracy : 0.9347
##
##        'Positive' Class : 0
##
```

```r
require(ROCR)
```

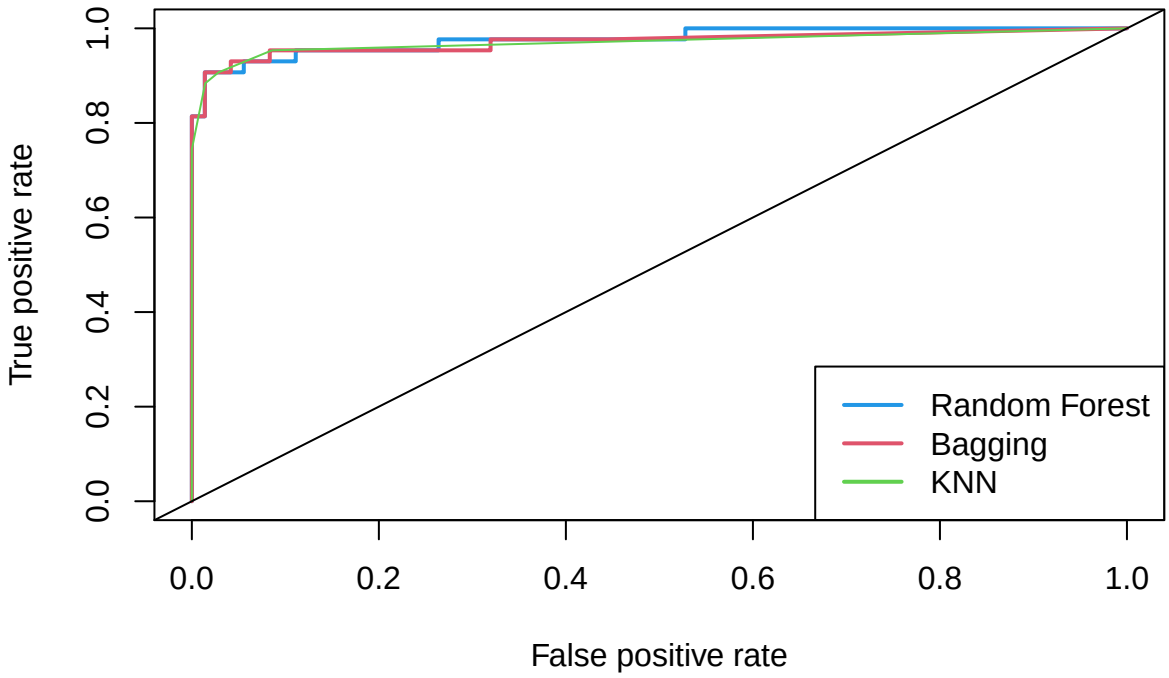
```
## Loading required package: ROCR

## Warning: package 'ROCR' was built under R version 4.0.4

##
## Attaching package: 'ROCR'

## The following object is masked from 'package:neuralnet':
##
##     prediction
```

```r
knnPredict <- predict(knnFit,newdata = test_FNA_knn , type="prob")
#Provide ROC with predictions and truth for analysis
roc_knn_pred <- prediction(knnPredict[,2],test_FNA_knn$diagnosis)


#Calculate true and false positive rates from ROC analysis
roc_knn_perf <- performance(roc_knn_pred,"tpr","fpr")

#Plot all three models using this ROC analysis, along with the a/b line as reference
plot(roc_knn_perf, col="blue") >
abline(a=0,b=1)
```




```
## logical(0)
```

```
library(ROCR)
predrf <- predict(rf4, newdata =  test_FNA, "prob")
predbag <- predict(bagging, newdata =  test_FNA, "prob")
knnPredict <- predict(knnFit,newdata = test_FNA_knn , type="prob")

roc_predsrf <- prediction(predrf[,2],test_FNA$diagnosis)
roc_predsbag <- prediction(predbag[,2],test_FNA$diagnosis)
roc_knn_pred <- prediction(knnPredict[,2],test_FNA_knn$diagnosis)

roc_perfrf <- performance(roc_predsrf,"tpr","fpr")
roc_perfbag <- performance(roc_predsbag,"tpr","fpr")
roc_knn_perf <- performance(roc_knn_pred,"tpr","fpr")

plot(roc_perfrf, col=4, lwd = 2)
plot(roc_perfbag, col=2, lwd = 2, add = T)
plot(roc_knn_perf, col=3, add = T)
abline(a=0,b=1)
legend(x="bottomright",
       legend = c("Random Forest","Bagging","KNN"),
       col = c(4,2,3),
       lwd = 2)
```


True positive rate
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0
Random Forest
Bagging
KNN