Cancer Classification

```r
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```r
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(keep)
library(glmnet)
```

```
## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-16
```

```r
library(NeuralNetTools)
library(randomForest)
```

```
## 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
```

```r
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 3.5.2

##
## Attaching package: 'tidyr'

## The following object is masked from 'package:Matrix':
##
```

```
##     expand
```

```
set.seed(1101)
```

read data

The breast cancer data consists of 30 features , they are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. n the 3-dimensional space is that described in: [K. P. Bennett and O. L. Mangasarian: “Robust Linear Programming Discrimination of Two Linearly Inseparable Sets”, Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server: ftp ftp.cs.wisc.edu cd math-prog/cpo-dataset/machine-learn/WDBC/

Also can be found on UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29

The target variable is diagnosis, tumor being malignant or benign. These 30 features are measures of the tumor such as radius, size, perimeter etc

```
bcancer <- read.csv("data.csv")
table(bcancer$diagnosis)
```

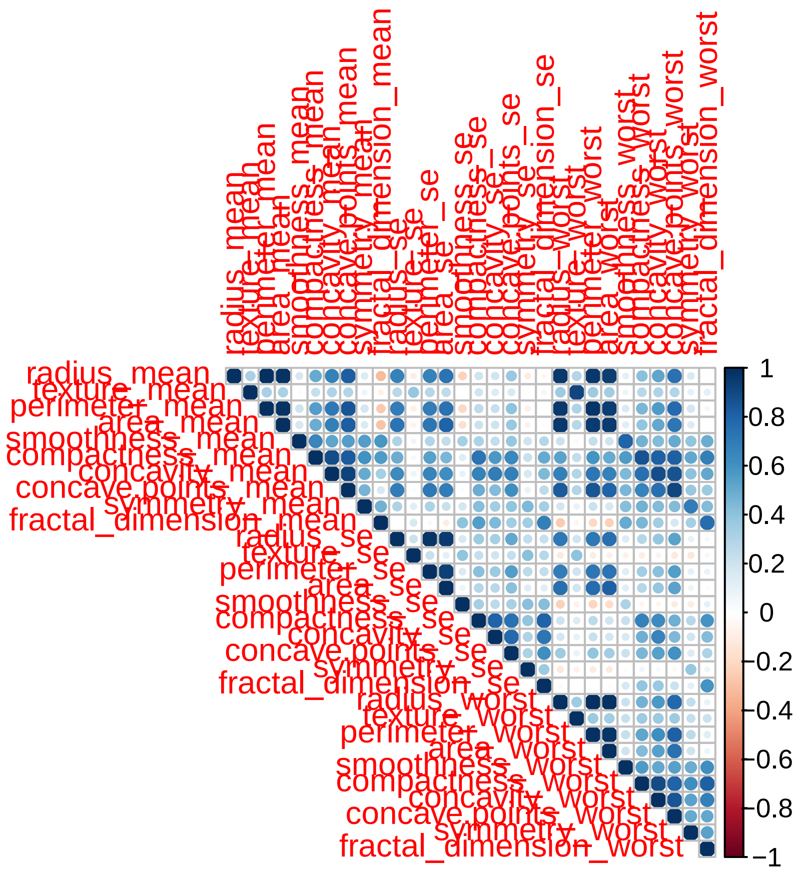
```
##
##   B   M
## 357 212
```

The objective of the analysis is to predict the the diagnosis of each patient id using these 30 features. I will use a classification model to identify the diagnosis.

DATA EXPLORATION

There are no missing values in the data and the distribution of the target variable is 63% of benign cancer and 37% of malignant cancer cells.

```
corMatrix <- cor(bcancer[,3:32])
corrplot(corMatrix , tl.cex = 1, addrect = 8 , type = "upper")
```

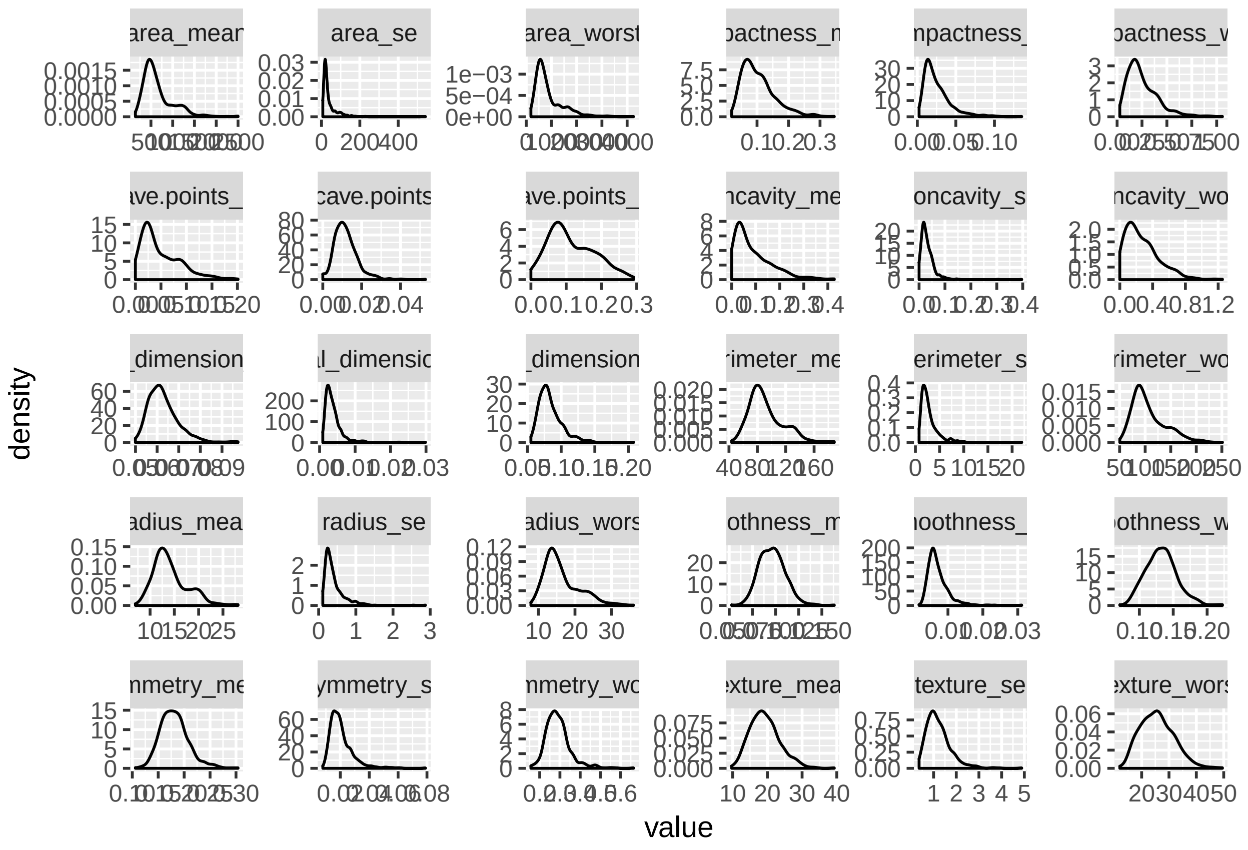


```r
summary(bcancer[,3:32])
```

```
##   radius_mean      texture_mean   perimeter_mean     area_mean
##  Min.   : 6.981   Min.   : 9.71   Min.   : 43.79   Min.   : 143.5
##  1st Qu.:11.700   1st Qu.:16.17   1st Qu.: 75.17   1st Qu.: 420.3
##  Median :13.370   Median :18.84   Median : 86.24   Median : 551.1
##  Mean   :14.127   Mean   :19.29   Mean   : 91.97   Mean   : 654.9
##  3rd Qu.:15.780   3rd Qu.:21.80   3rd Qu.:104.10   3rd Qu.: 782.7
##  Max.   :28.110   Max.   :39.28   Max.   :188.50   Max.   :2501.0
##  smoothness_mean   compactness_mean  concavity_mean    concave.points_mean
##  Min.   :0.05263   Min.   :0.01938   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.08637   1st Qu.:0.06492   1st Qu.:0.02956   1st Qu.:0.02031
##  Median :0.09587   Median :0.09263   Median :0.06154   Median :0.03350
##  Mean   :0.09636   Mean   :0.10434   Mean   :0.08880   Mean   :0.04892
##  3rd Qu.:0.10530   3rd Qu.:0.13040   3rd Qu.:0.13070   3rd Qu.:0.07400
##  Max.   :0.16340   Max.   :0.34540   Max.   :0.42680   Max.   :0.20120
##  symmetry_mean    fractal_dimension_mean   radius_se        texture_se
##  Min.   :0.1060   Min.   :0.04996        Min.   :0.1115   Min.   :0.3602
##  1st Qu.:0.1619   1st Qu.:0.05770        1st Qu.:0.2324   1st Qu.:0.8339
##  Median :0.1792   Median :0.06154        Median :0.3242   Median :1.1080
##  Mean   :0.1812   Mean   :0.06280        Mean   :0.4052   Mean   :1.2169
##  3rd Qu.:0.1957   3rd Qu.:0.06612        3rd Qu.:0.4789   3rd Qu.:1.4740
##  Max.   :0.3040   Max.   :0.09744        Max.   :2.8730   Max.   :4.8850
##   perimeter_se       area_se        smoothness_se       compactness_se
##  Min.   : 0.757   Min.   :  6.802   Min.   :0.001713   Min.   :0.002252
##  1st Qu.: 1.606   1st Qu.: 17.850   1st Qu.:0.005169   1st Qu.:0.013080
##  Median : 2.287   Median : 24.530   Median :0.006380   Median :0.020450
##  Mean   : 2.866   Mean   : 40.337   Mean   :0.007041   Mean   :0.025478
##  3rd Qu.: 3.357   3rd Qu.: 45.190   3rd Qu.:0.008146   3rd Qu.:0.032450
##  Max.   :21.980   Max.   :542.200   Max.   :0.031130   Max.   :0.135400
##   concavity_se     concave.points_se   symmetry_se
##  Min.   :0.00000   Min.   :0.000000   Min.   :0.007882
##  1st Qu.:0.01509   1st Qu.:0.007638   1st Qu.:0.015160
##  Median :0.02589   Median :0.010930   Median :0.018730
##  Mean   :0.03189   Mean   :0.011796   Mean   :0.020542
##  3rd Qu.:0.04205   3rd Qu.:0.014710   3rd Qu.:0.023480
##  Max.   :0.39600   Max.   :0.052790   Max.   :0.078950
##  fractal_dimension_se  radius_worst   texture_worst   perimeter_worst
##  Min.   :0.0008948    Min.   : 7.93   Min.   :12.02   Min.   : 50.41
##  1st Qu.:0.0022480    1st Qu.:13.01   1st Qu.:21.08   1st Qu.: 84.11
##  Median :0.0031870    Median :14.97   Median :25.41   Median : 97.66
##  Mean   :0.0037949    Mean   :16.27   Mean   :25.68   Mean   :107.26
##  3rd Qu.:0.0045580    3rd Qu.:18.79   3rd Qu.:29.72   3rd Qu.:125.40
##  Max.   :0.0298400    Max.   :36.04   Max.   :49.54   Max.   :251.20
##    area_worst     smoothness_worst  compactness_worst concavity_worst
##  Min.   : 185.2   Min.   :0.07117   Min.   :0.02729   Min.   :0.0000
##  1st Qu.: 515.3   1st Qu.:0.11660   1st Qu.:0.14720   1st Qu.:0.1145
##  Median : 686.5   Median :0.13130   Median :0.21190   Median :0.2267
##  Mean   : 880.6   Mean   :0.13237   Mean   :0.25427   Mean   :0.2722
##  3rd Qu.:1084.0   3rd Qu.:0.14600   3rd Qu.:0.33910   3rd Qu.:0.3829
##  Max.   :4254.0   Max.   :0.22260   Max.   :1.05800   Max.   :1.2520
##  concave.points_worst symmetry_worst   fractal_dimension_worst
##  Min.   :0.00000      Min.   :0.1565   Min.   :0.05504
##  1st Qu.:0.06493      1st Qu.:0.2504   1st Qu.:0.07146
##  Median :0.09993      Median :0.2822   Median :0.08004
##  Mean   :0.11461      Mean   :0.2901   Mean   :0.08395
##  3rd Qu.:0.16140      3rd Qu.:0.3179   3rd Qu.:0.09208
##  Max.   :0.29100      Max.   :0.6638   Max.   :0.20750
```

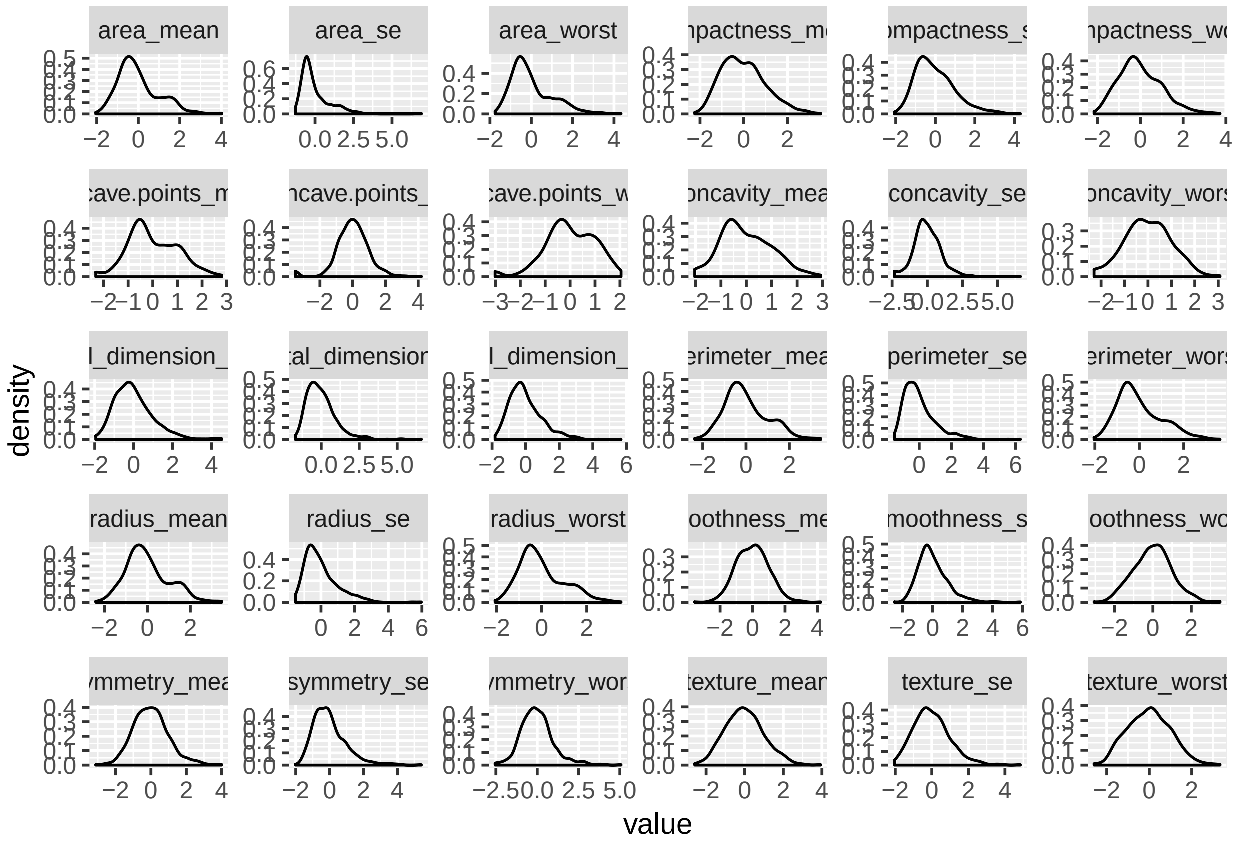
Many features in the data are highly correlated. For e.g. radius mean and radius worst. This could cause multicolinearity in our models. Looking at the univariate plot of the data, we can see that the variables are very skewed . Most variables are right skewed and have varying range and scales. The variance in some of the area_se variable is quite high.

```
bcancer[,3:32] %>%
  #keep(is.numeric) %>%                    # Keep only numeric columns
  gather() %>%                             # Convert to key-value pairs
  ggplot(aes(value)) +                     # Plot the values
  facet_wrap(~ key, scales = "free") +   # In separate panels
  geom_density()
```



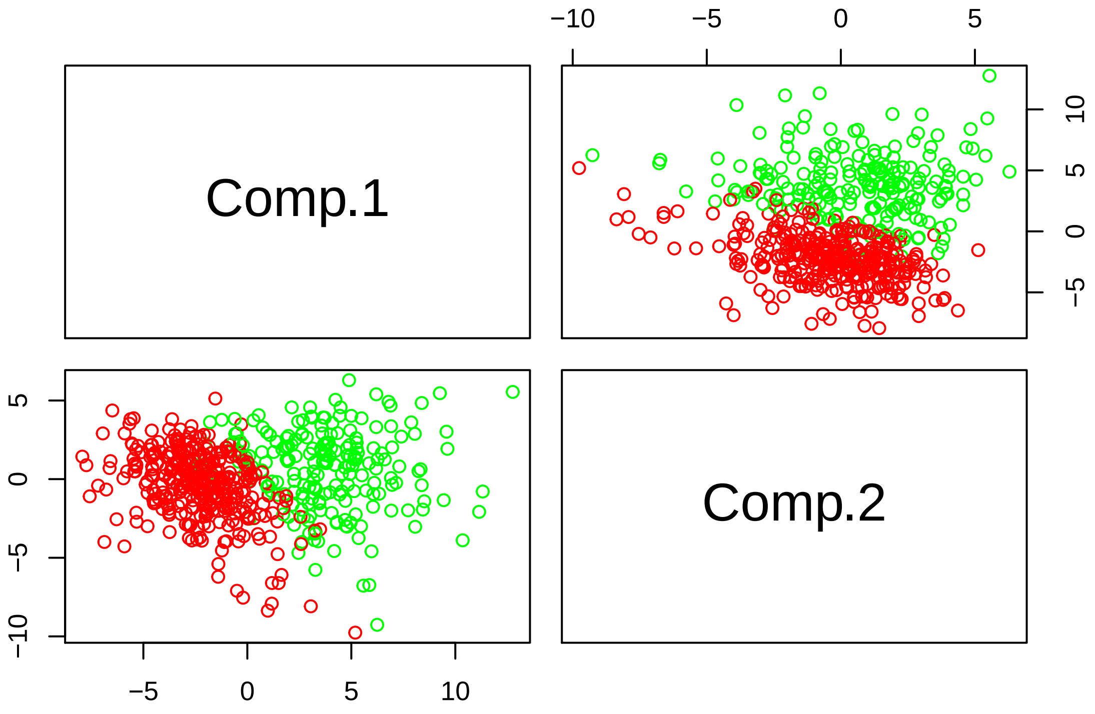
As these distributions are skewed and vary largely on the scale I will transform the data. Using log transformation will cause errors as some values are 0 leading to undefined cases. Hence, I use a square root transformation and then rescale the data.

```
sdata <- bcancer
sdata[,3:32] <- sqrt(sdata[,3:32])
sdata[,3:32] <-  scale(sdata[,3:32])

sdata[,3:32] %>%
  #keep(is.numeric) %>%                     # Keep only numeric columns
  gather() %>%                             # Convert to key-value pairs
  ggplot(aes(value)) +                     # Plot the values
  facet_wrap(~ key, scales = "free") +   # In separate panels
  geom_density()
```



Now that the data is centered we could try different models on our data set. As there are many features in the data , to reduce the dimentionality of the data I will run a PCA on the scaled and transformed data.

```r
pca <- princomp(sdata[,3:32])
pca_scores <- pca$scores
#pca$loadings

pairs(pca_scores[,1:2] , col = c("red" , "green")[sdata$diagnosis])
```



From the factor loading we can see that none of the components contribute heavily into the classification and only 7% variance is explained by the first 2 components. There are some overlap regions between the two classes which would be difficult to classify. Its hard to say from the princomp to decide which component will classify the data correctly, I chose to not go ahead with pca.

```r
s <- sdata[,2:32]
inTrain <- createDataPartition(y=s$diagnosis, p=0.7, list=FALSE)
training <- s[inTrain,]
testing <- s[-inTrain,]
```

SVM - Linear

```r
train_control <- trainControl(method="repeatedcv", number=10, repeats=20)
s <- sdata[,2:32]
svmLinear <- train(diagnosis~., data= training, trControl=train_control, method="svmLinear")


fit_svmLinear <- predict(svmLinear , newdata = testing)
cm_svmL <- confusionMatrix(fit_svmLinear , testing$diagnosis)
cm_svmL
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 107   7
##          M   0  56
##
##                Accuracy : 0.9588
##                  95% CI : (0.917, 0.9833)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : < 2e-16
##
```

```
##                   Kappa : 0.9097
##  Mcnemar's Test P-Value : 0.02334
##
##             Sensitivity : 1.0000
##             Specificity : 0.8889
##          Pos Pred Value : 0.9386
##          Neg Pred Value : 1.0000
##              Prevalence : 0.6294
##          Detection Rate : 0.6294
##    Detection Prevalence : 0.6706
##       Balanced Accuracy : 0.9444
##
##        'Positive' Class : B
##
```

SVM - Radial

We will perform a 10 fold cross validation on the test data and do an out of sample testing for each model.

```
svmRadial <- train(diagnosis~., data= training, trControl=train_control, method="svmRadial")


fit_svmRadial <- predict(svmRadial , newdata = testing)
cm_svmR <- confusionMatrix(fit_svmRadial , testing$diagnosis)
cm_svmR
```

```
## 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 : 1.0000
##             Specificity : 0.9206
##          Pos Pred Value : 0.9554
##          Neg Pred Value : 1.0000
##              Prevalence : 0.6294
##          Detection Rate : 0.6294
##    Detection Prevalence : 0.6588
##       Balanced Accuracy : 0.9603
##
##        'Positive' Class : B
##
```

```
svmRadial$results$Accuracy
```

```
## [1] 0.9670545 0.9744583 0.9798365
```

```
svmRadial$results$AccuracySD
```

```
## [1] 0.02731376 0.02417758 0.02079046
```

SVM does a good job in predicting the classes with only 3 data points misclassified in the Radial SVM model. The insample accuracy of the model is also quite high with the standard deviation of 0.02. We can say the accuracy estimate of SVM is strong. In this problem, the false negatives are of high importance. It is crucial to detect the tumor so that the patients get treatments. Hence, we will try to reduce the false negatives, i.e those cases that are originally malignant but classified as benign.

```
knn <- train(diagnosis~., data= training, trControl=train_control, method="knn")
fit_knn <- predict(knn , newdata = testing)
CM_KNN <- confusionMatrix(fit_knn , testing$diagnosis)
CM_KNN
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 105   5
##          M   2  58
##
##                Accuracy : 0.9588
##                  95% CI : (0.917, 0.9833)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9109
##  Mcnemar's Test P-Value : 0.4497
##
##             Sensitivity : 0.9813
##             Specificity : 0.9206
##          Pos Pred Value : 0.9545
##          Neg Pred Value : 0.9667
##              Prevalence : 0.6294
##          Detection Rate : 0.6176
##    Detection Prevalence : 0.6471
##       Balanced Accuracy : 0.9510
##
##        'Positive' Class : B
##
```

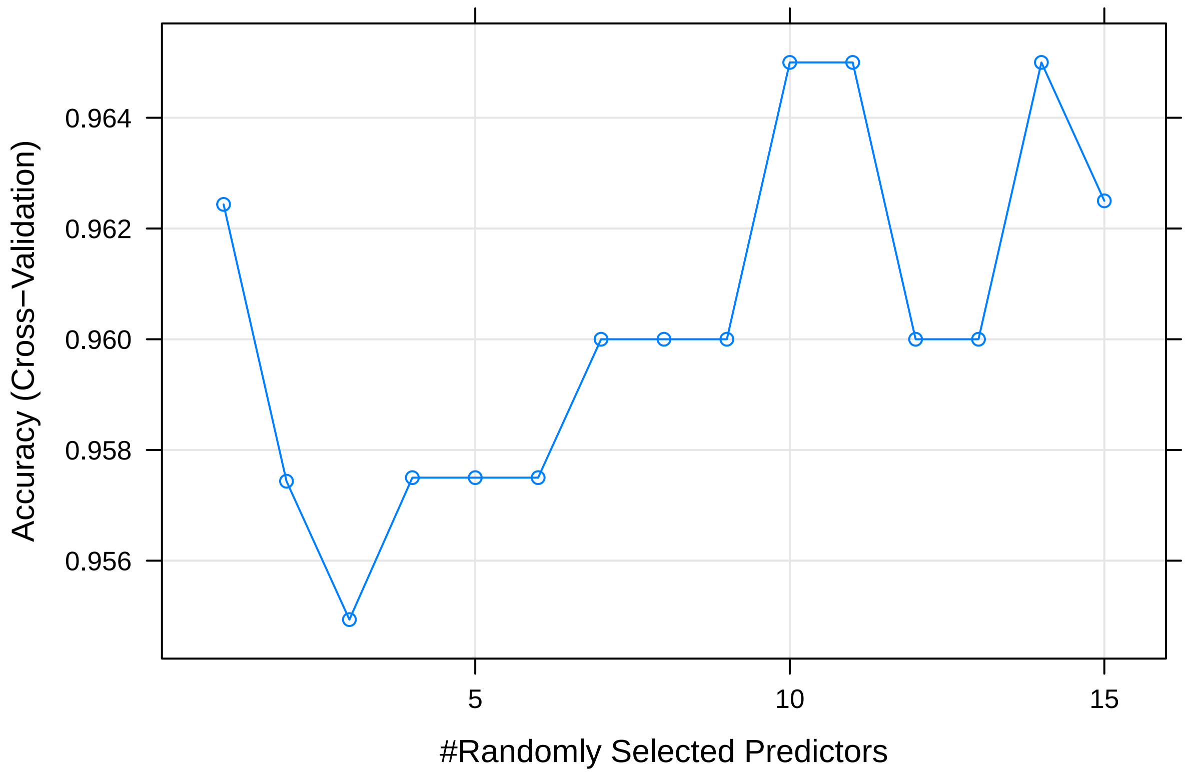
KNN also has high accuracy but again there are more false negatives , which is not desirable.

We will try thr random forest model with grid search approach.

```
tuneGrid <- expand.grid(.mtry = c(1:15) )
trControl <- trainControl(method = "cv", number = 10, search = "grid")
rf <- train(diagnosis~., data= training ,method = "rf", metric = "Accuracy",  trControl =trControl,tune
pred_rf <-predict(rf, testing)
CM_RF <- confusionMatrix(pred_rf , testing$diagnosis)
CM_RF
```

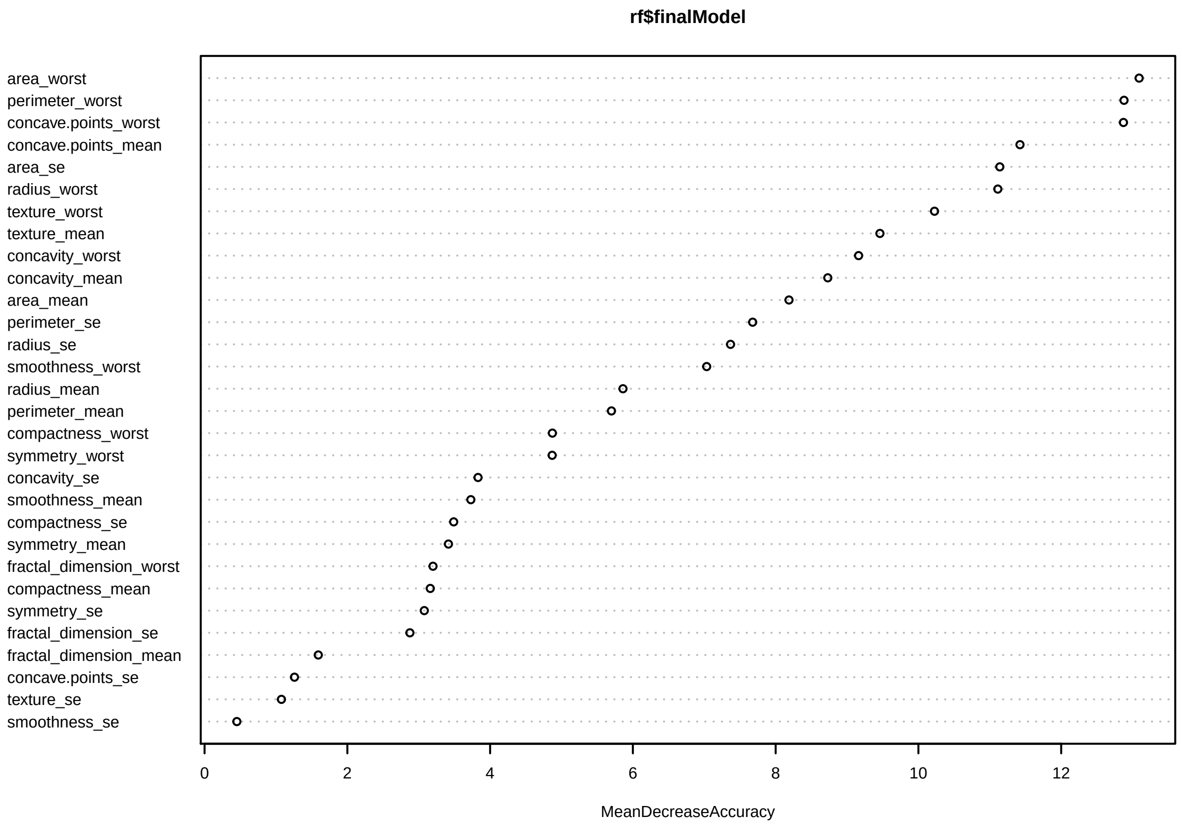
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##           B 104   4
##           M   3  59
##
##                 Accuracy : 0.9588
##                   95% CI : (0.917, 0.9833)
##      No Information Rate : 0.6294
##      P-Value [Acc > NIR] : <2e-16
##
##                    Kappa : 0.9114
##   Mcnemar's Test P-Value : 1
##
##              Sensitivity : 0.9720
##              Specificity : 0.9365
##           Pos Pred Value : 0.9630
##           Neg Pred Value : 0.9516
##               Prevalence : 0.6294
##           Detection Rate : 0.6118
##     Detection Prevalence : 0.6353
##        Balanced Accuracy : 0.9542
##
##         'Positive' Class : B
##
```

```r
plot(rf)
```



The results of random forest are more promising with just 2 false negatives. There are still 2 malignant cases that are classified as benign. We also have increased number of false positive in this model. I tried out different mtry for grid search and we can see that the maximum accuracy is obtained when 7 features are randomly sampled for each evaluvation which alos close to sqrt(30)

```r
varImpPlot(rf$finalModel,type=1  ,cex=.5)
```



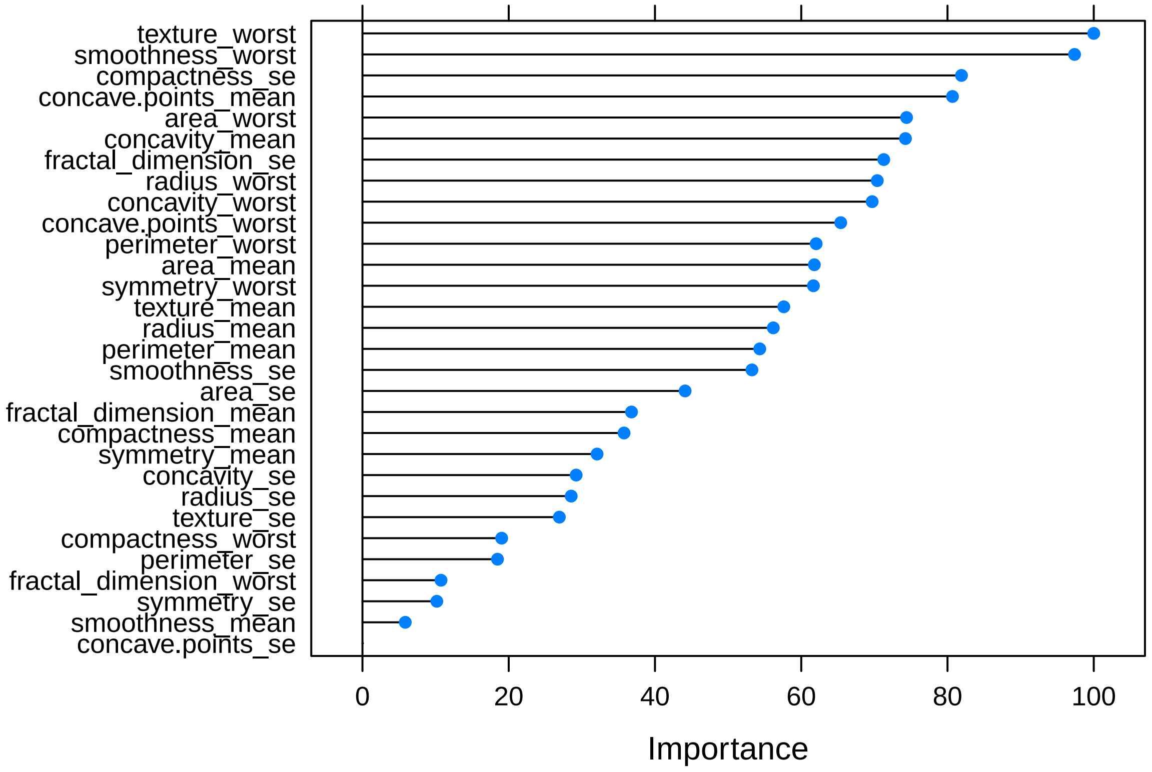
We also look at the variable importance to understand the contribution of each variable in the classification process. To try an improve our model I will next try a neural network.

```r
set.seed(1101)
tuneGrid <- expand.grid(.size = c(1:6), .decay=c(0,2.5e-2,5e-2,7.5e-2,1e-1,1e-2) )
nnet <- capture.output(nn <- caret::train(diagnosis~., data= training, method = "nnet", metric = "Accura


fit_nn <- predict(nn , testing)
t <- confusionMatrix(fit_nn , testing$diagnosis)
t
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   B   M
##          B 106   4
##          M   1  59
##
##                Accuracy : 0.9706
##                  95% CI : (0.9327, 0.9904)
##     No Information Rate : 0.6294
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9363
##  Mcnemar's Test P-Value : 0.3711
##
```

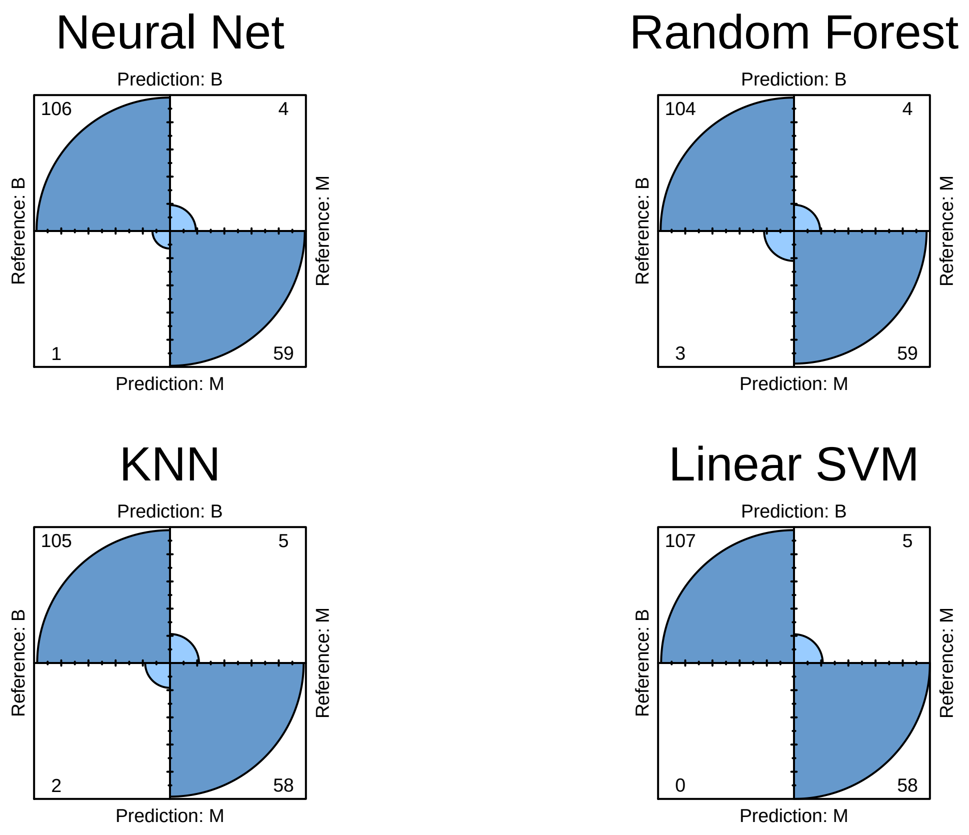
```
##             Sensitivity : 0.9907
##             Specificity : 0.9365
##          Pos Pred Value : 0.9636
##          Neg Pred Value : 0.9833
##              Prevalence : 0.6294
##          Detection Rate : 0.6235
##    Detection Prevalence : 0.6471
##       Balanced Accuracy : 0.9636
##
##        'Positive' Class : B
##
```

```
nnet_vatimpt <- varImp(nn)
plot(nnet_vatimpt)
```



The neural network model also gives high accuracy and the missclassified rate is also low. Both the models random forest and Neural Network work well for this data. But I would like to go ahead with NNet model as it is less computational expensive compared to random forest.

```
par(mfrow=c(2,2))

fourfoldplot(t$table ,conf.level = 0, margin = 1 , main = "Neural Net")
fourfoldplot(CM_RF$table ,conf.level = 0, margin = 1 , main = "Random Forest")
fourfoldplot(CM_KNN$table,conf.level = 0, margin = 1 , main = "KNN")
fourfoldplot(cm_svmR$table ,conf.level = 0, margin = 1 , main = "Linear SVM")
```

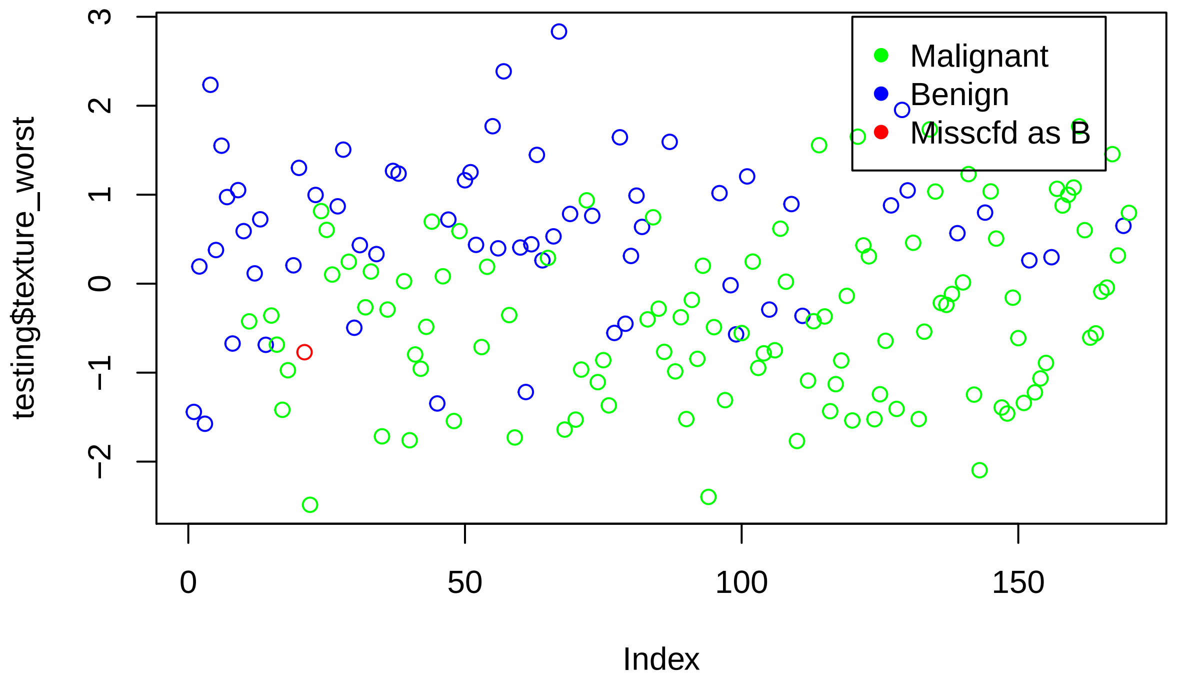


Analysing the missclassified data to understand the error in our model. Through bivariate plots it is difficult to understand the error. Some expertise on the data and knowledge of the subject would probably help us understand the problem and build a finer model to detect the cancer.

```
misclassified <- testing[which(fit_nn != testing[,1]), ]
misclassified
```

```
##     diagnosis radius_mean texture_mean perimeter_mean  area_mean
## 74          M -0.03606509  -0.81124238   -0.002097492 -0.1028500
## 136         M -0.34756777   0.77621175   -0.386399045 -0.3688263
## 216         M -0.01828376  -0.51937386    0.020676442 -0.1200526
## 264         M  0.48438093   0.07632073    0.399369535  0.4355538
## 364         B  0.72920285  -0.18375347    0.665137478  0.6600813
##     smoothness_mean compactness_mean concavity_mean concave.points_mean
## 74       0.34321220        0.5685772     0.08907512           0.2517173
## 136     -0.38574209       -0.9443617    -0.38190215          -0.4400026
## 216      0.47551577        0.9760872     0.35911332           0.3832229
## 264     -1.31440991       -0.9834196    -0.47216534          -0.3910957
## 364      0.07195321       -0.2893660    -0.19160176           0.1918038
##     symmetry_mean fractal_dimension_mean  radius_se  texture_se
## 74     -0.5288348              0.4396747 -0.4519290 -1.24250505
## 136    -0.8301097             -0.2883112 -0.6796914  0.41341433
## 216     1.0865208              0.9318799 -0.5710834  0.06242961
## 264    -0.9814880             -1.2353503 -0.7189902 -0.33678578
## 364    -1.1916831             -1.0022117 -0.1535941  0.51973648
##     perimeter_se     area_se smoothness_se compactness_se concavity_se
## 74    -0.4568406 -0.39407448    -0.8500412     -0.1549129   -0.4492374
## 136   -0.8507038 -0.54233489     0.2635577     -0.8319450   -0.1635229
## 216   -0.4745039 -0.42129666    -0.3204383      0.6742002    0.4767663
## 264   -0.7860712 -0.44260427    -1.8147117     -1.1144742    -0.7071624
## 364   -0.1856965 -0.01308313     0.1759969     -0.3407554    -0.4104797
##     concave.points_se symmetry_se fractal_dimension_se radius_worst
## 74         -0.2837356 -1.18118889           -0.1858153   0.13602251
## 136        -0.3340067 -0.48928137           -0.4569881  -0.32625021
## 216         0.5142158 -0.06704247            0.4863016  -0.04254602
## 264        -1.0875033 -1.56060319           -1.3042620   0.41857230
## 364        -0.0933569 -0.41892717           -0.8366042   0.46393764
##     texture_worst perimeter_worst   area_worst smoothness_worst
## 74    -0.77038345      0.16919269  0.007198285        0.4207874
## 136    1.23708790     -0.41966855 -0.339458820        0.4548136
## 216    0.26229439     -0.01534565 -0.124366960        0.6277119
## 264    0.98965394      0.33983417  0.354132653       -1.0636702
## 364    0.02225315      0.37885383  0.392226809        0.1057293
##     compactness_worst concavity_worst concave.points_worst symmetry_worst
## 74          0.7683540      0.23384268            0.4739518     -0.4901866
## 136        -0.6292397     -0.05728276           -0.1483584     -0.0708075
## 216         1.2179557      0.97257653            0.8000677      1.2132903
## 264        -0.3916455     -0.01493833           -0.2678673     -0.3237052
## 364        -0.4962290     -0.33978416           -0.1804370     -0.8454749
##     fractal_dimension_worst
## 74                1.1103462
## 136              -0.1430000
## 216               1.2626932
## 264              -0.9138259
## 364              -1.1508780
```

```
plot(testing$texture_worst, col=ifelse(rownames(testing) %in% rownames(misclassified[1,])
, 'red', c("green" , "blue")[testing$diagnosis]), lower.panel = NULL)
legend(120, 3, legend = c("Malignant" , "Benign" , "Misscfd as B") , col = c( "green" , "blue" ,"red")
```


● Malignant
● Benign
● Misscfd as B
testing$texture_worst
Index