Cancer\_detection\_model

Vijay Mudivedu

2018-09-27

#install.packages("knitr")  
#install.packages("tidyverse")  
#install.packages("ggbiplot")  
#install.packages("reshape2")  
#install.packages("randomForest")  
#install.packages("kernlab")  
#install.packages("glmnet")  
#install.packages("caret")  
#install.packages(gridExtra)  
#install.packages("psych")  
library(knitr)  
library(tidyverse)

## ── Attaching packages ────────

## ✔ ggplot2 3.0.0 ✔ purrr 0.2.4  
## ✔ tibble 1.4.2 ✔ dplyr 0.7.4  
## ✔ tidyr 0.8.0 ✔ stringr 1.3.0  
## ✔ readr 1.1.1 ✔ forcats 0.3.0

## ── Conflicts ─────────────────  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(reshape2)

##   
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':  
##   
## smiths

library(ggbiplot)

## Loading required package: plyr

## -------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.  
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:  
## library(plyr); library(dplyr)

## -------------------------------------------------------------------------

##   
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following object is masked from 'package:purrr':  
##   
## compact

## Loading required package: scales

##   
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':  
##   
## discard

## The following object is masked from 'package:readr':  
##   
## col\_factor

## Loading required package: grid

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

library(kernlab)

##   
## Attaching package: 'kernlab'

## The following object is masked from 'package:scales':  
##   
## alpha

## The following object is masked from 'package:purrr':  
##   
## cross

## The following object is masked from 'package:ggplot2':  
##   
## alpha

library(glmnet)

## Loading required package: Matrix

##   
## Attaching package: 'Matrix'

## The following object is masked from 'package:tidyr':  
##   
## expand

## Loading required package: foreach

##   
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':  
##   
## accumulate, when

## Loaded glmnet 2.0-16

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

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

library(gridExtra)

##   
## Attaching package: 'gridExtra'

## The following object is masked from 'package:randomForest':  
##   
## combine

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

library(psych)

##   
## Attaching package: 'psych'

## The following object is masked from 'package:kernlab':  
##   
## alpha

## The following object is masked from 'package:randomForest':  
##   
## outlier

## The following objects are masked from 'package:scales':  
##   
## alpha, rescale

## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

## Business Understanding

Features 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.

A diagnostics team is developing a model to classify a tumor as either “benign” or “malignant”.

1. ID number
2. Diagnosis (M = malignant, B = benign) Ten real-valued features are computed for each cell nucleus:
3. radius (mean of distances from center to points on the perimeter)
4. texture (standard deviation of gray-scale values)
5. perimeterj
6. area
7. smoothness (local variation in radius lengths)
8. compactness (perimeter^2 / area - 1.0)
9. concavity (severity of concave portions of the contour)
10. concave points (number of concave portions of the contour)
11. symmetry
12. fractal dimension (“coastline approximation” - 1)

The mean, standard error, and “worst” or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius. All feature values are recoded with four significant digits.

### Objective of the exercise

1. Conduct an Exploratory Data Analysis and explain your findings.
2. Develop at least three classification models using different machine learning techniques.
3. Explain in detail which model is better and why?
4. Develop principal components for all independent variables, develop the ML models on principal components. Compare these models with previous models. What are the advantages (if any) of using PCA in classification model building?

# ——————

# Data Understanding

# ——————

#install.packages("xlsx")  
#install.packages("rJava")  
setwd("/Users/vmudivedu/Google Drive/\_OneDrive\_Atimi\_Software/Upgrad/case study/Cancer")  
cancer\_df\_raw <- read.csv(file = "~/Google Drive/\_OneDrive\_Atimi\_Software/Upgrad/case study/Cancer/Worksheet in CaseStudy\_Cancer\_1.csv",header = T,check.names = T,stringsAsFactors = T)  
head(cancer\_df\_raw)

## ID B.M radius texture perimeter area smoothness compactness  
## 1 842302 M 17.99 10.38 122.80 1001.0 0.11840 0.27760  
## 2 842517 M 20.57 17.77 132.90 1326.0 0.08474 0.07864  
## 3 84300903 M 19.69 21.25 130.00 1203.0 0.10960 0.15990  
## 4 84348301 M 11.42 20.38 77.58 386.1 0.14250 0.28390  
## 5 84358402 M 20.29 14.34 135.10 1297.0 0.10030 0.13280  
## 6 843786 M 12.45 15.70 82.57 477.1 0.12780 0.17000  
## concavity concave.points Symmetry fractal.dimension SE.radius texture.SE  
## 1 0.3001 0.14710 0.2419 0.07871 1.0950 0.9053  
## 2 0.0869 0.07017 0.1812 0.05667 0.5435 0.7339  
## 3 0.1974 0.12790 0.2069 0.05999 0.7456 0.7869  
## 4 0.2414 0.10520 0.2597 0.09744 0.4956 1.1560  
## 5 0.1980 0.10430 0.1809 0.05883 0.7572 0.7813  
## 6 0.1578 0.08089 0.2087 0.07613 0.3345 0.8902  
## perimeter.SE area.SE smoothness.SE compactness.SE concavity.SE  
## 1 8.589 153.40 0.006399 0.04904 0.05373  
## 2 3.398 74.08 0.005225 0.01308 0.01860  
## 3 4.585 94.03 0.006150 0.04006 0.03832  
## 4 3.445 27.23 0.009110 0.07458 0.05661  
## 5 5.438 94.44 0.011490 0.02461 0.05688  
## 6 2.217 27.19 0.007510 0.03345 0.03672  
## concave.points.SE Symmetry.SE fractal.dimension.SE radius.W texture.W  
## 1 0.01587 0.03003 0.006193 25.38 17.33  
## 2 0.01340 0.01389 0.003532 24.99 23.41  
## 3 0.02058 0.02250 0.004571 23.57 25.53  
## 4 0.01867 0.05963 0.009208 14.91 26.50  
## 5 0.01885 0.01756 0.005115 22.54 16.67  
## 6 0.01137 0.02165 0.005082 15.47 23.75  
## perimeter.W area.W smoothness.W compactness.W concavity.W  
## 1 184.60 2019.0 0.1622 0.6656 0.7119  
## 2 158.80 1956.0 0.1238 0.1866 0.2416  
## 3 152.50 1709.0 0.1444 0.4245 0.4504  
## 4 98.87 567.7 0.2098 0.8663 0.6869  
## 5 152.20 1575.0 0.1374 0.2050 0.4000  
## 6 103.40 741.6 0.1791 0.5249 0.5355  
## concave.points.W Symmetry.W fractal.dimension.W  
## 1 0.2654 0.4601 0.11890  
## 2 0.1860 0.2750 0.08902  
## 3 0.2430 0.3613 0.08758  
## 4 0.2575 0.6638 0.17300  
## 5 0.1625 0.2364 0.07678  
## 6 0.1741 0.3985 0.12440

### Data Preparation

* Check for:
* outliers
* NAs
* Duplicates
* invalid values

names(cancer\_df\_raw)

## [1] "ID" "B.M" "radius"   
## [4] "texture" "perimeter" "area"   
## [7] "smoothness" "compactness" "concavity"   
## [10] "concave.points" "Symmetry" "fractal.dimension"   
## [13] "SE.radius" "texture.SE" "perimeter.SE"   
## [16] "area.SE" "smoothness.SE" "compactness.SE"   
## [19] "concavity.SE" "concave.points.SE" "Symmetry.SE"   
## [22] "fractal.dimension.SE" "radius.W" "texture.W"   
## [25] "perimeter.W" "area.W" "smoothness.W"   
## [28] "compactness.W" "concavity.W" "concave.points.W"   
## [31] "Symmetry.W" "fractal.dimension.W"

sum(duplicated(cancer\_df\_raw$ID)) # no duplicates found

## [1] 0

apply(cancer\_df\_raw, MARGIN = 2, FUN = function(x) sum(is.na(x))) # no NAs found in the dataset

## ID B.M radius   
## 0 0 0   
## texture perimeter area   
## 0 0 0   
## smoothness compactness concavity   
## 0 0 0   
## concave.points Symmetry fractal.dimension   
## 0 0 0   
## SE.radius texture.SE perimeter.SE   
## 0 0 0   
## area.SE smoothness.SE compactness.SE   
## 0 0 0   
## concavity.SE concave.points.SE Symmetry.SE   
## 0 0 0   
## fractal.dimension.SE radius.W texture.W   
## 0 0 0   
## perimeter.W area.W smoothness.W   
## 0 0 0   
## compactness.W concavity.W concave.points.W   
## 0 0 0   
## Symmetry.W fractal.dimension.W   
## 0 0

* *Normalizing the Parameters*

normalize\_func <- function(x)  
{  
 return((x-min(x))/(max(x)-min(x)))  
}  
  
cancer\_df <- data.frame( label = cancer\_df\_raw$B.M ,apply(cancer\_df\_raw[,-c(1,2)], MARGIN = 2, FUN = function(x) normalize\_func(x)))

# Shortening the headers  
names(cancer\_df)[which(names(cancer\_df) %in% c("fractal.dimension","fractal.dimension.SE","fractal.dimension.W"))] <-   
 c("fractal.dim","fractal.dim.SE","fractal.dim.W")

* Checking the distribution of the diagnosis Class label

table(cancer\_df$label)

##   
## B M   
## 357 212

paste(levels(cancer\_df$label),"-",round(prop.table(table(cancer\_df$label))\*100,2),"%")

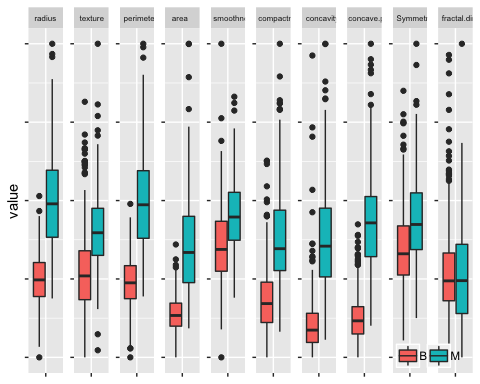
## [1] "B - 62.74 %" "M - 37.26 %"

* B = 357, 62.74% percent of the cases are benign
* M = 212, 37.26% percent of the cases are Malignant
* *Creating three groups of variable for analysis of mean, se and worst parameter of Breast Cancer dataset*

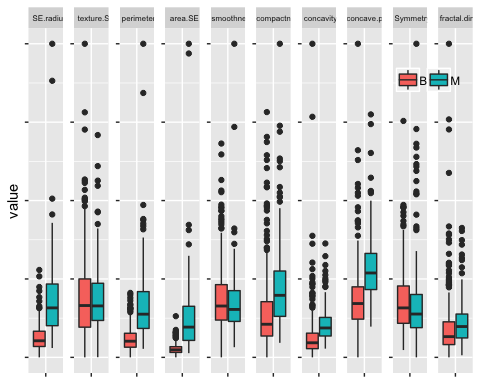
var\_1 <- c("label","radius","texture","perimeter","area","smoothness","compactness","concavity","concave.points","Symmetry","fractal.dim")  
var\_2 <- c("label","SE.radius","texture.SE","perimeter.SE","area.SE","smoothness.SE","compactness.SE","concavity.SE","concave.points.SE","Symmetry.SE",  
 "fractal.dim.SE")   
var\_3 <- c("label","radius.W", "texture.W","perimeter.W","area.W","smoothness.W","compactness.W","concavity.W","concave.points.W","Symmetry.W","fractal.dim.W")

*Exploratory Data Analysis*

plot1 <- ggplot(data = melt(cancer\_df[var\_1],id.vars = "label"),aes(variable,value,fill = label)) + geom\_boxplot() +   
 facet\_wrap(facets = variable ~.,scales = "free",nrow = 1) +   
 theme(legend.position = c(0.9,0.05),legend.background = element\_blank(),legend.title = element\_blank(),legend.direction = "horizontal",  
 axis.title.x = element\_blank(),axis.text.x = element\_blank(), axis.text.y = element\_blank(),  
 strip.text = element\_text(size = 6,hjust = 0.15))  
  
plot1

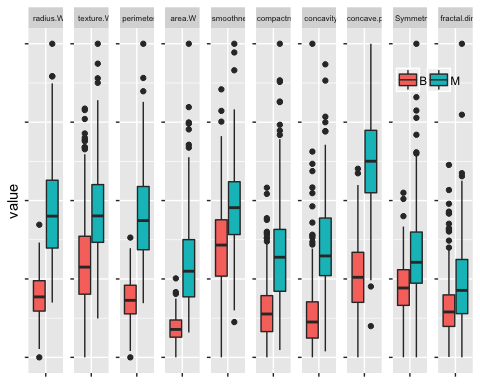
 \* Green Malignant Tumor, M and Red Records \* *Observations*: Parameters of Cancer cells are observed to be larger in values compared to the Benign Cells + Excepting the Fractal.dimesions means of all parameters is higher for a Malignant Breast Cancer cells

plot2 <- ggplot(data = melt(cancer\_df[var\_2],id.vars = "label"),aes(variable,value,fill = label)) + geom\_boxplot() +   
 facet\_wrap(facets = variable ~.,scales = "free",nrow = 1) +   
 theme(legend.position = c(0.9,0.85),legend.background = element\_blank(),legend.title = element\_blank(),legend.direction = "horizontal",  
 axis.title.x = element\_blank(),axis.text.x = element\_blank(), axis.text.y = element\_blank(),  
 strip.text = element\_text(size = 6,hjust = 0.15))  
  
plot2



* Standard Error of most parameters for the diagnosed Malignant Breast Cancer cells is higher compared to Benign cells
* Standard error of texture of cells is similar for Malignant and Benign Breast Cancer cells

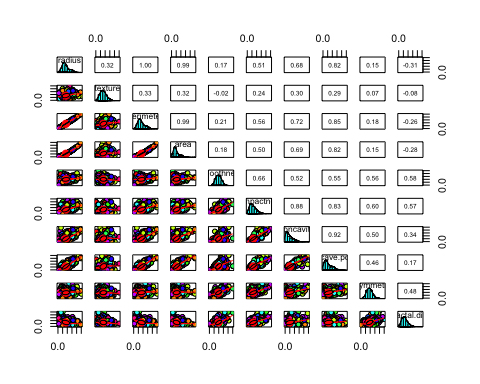
plot3 <- ggplot(data = melt(cancer\_df[var\_3],id.vars = "label"),aes(variable,value,fill = label)) + geom\_boxplot() +   
 facet\_wrap(facets = variable ~.,scales = "free",nrow = 1) +   
 theme(legend.position = c(0.9,0.85),legend.background = element\_blank(),legend.title = element\_blank(),legend.direction = "horizontal",  
 axis.title.x = element\_blank(),axis.text.x = element\_blank(), axis.text.y = element\_blank(),  
 strip.text = element\_text(size = 6,hjust = 0.15))  
  
 plot3



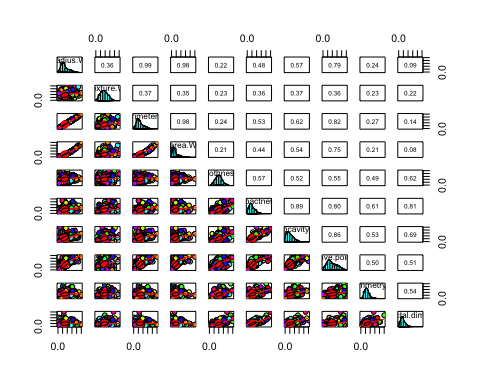
* It can be concluded from the boxplots that
* Malignant tumors have larger radius, texture, perimeter, area, compactnes, concavity, concavity points, fact

*Checking the correlation between the different sets of variables*

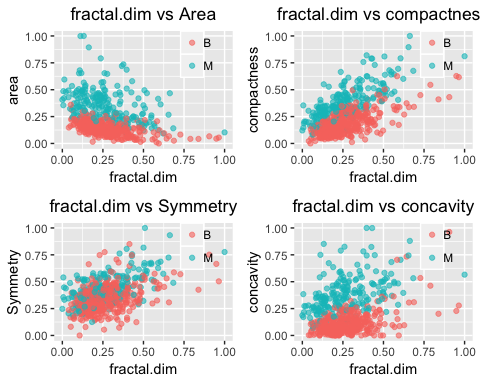
pairs.panels(cancer\_df[var\_1[-1]],smooth = TRUE,ellipses = TRUE,pch = 21,bg = rainbow(n = length(var\_1)-1))



pairs.panels(cancer\_df[var\_3[-1]],smooth = TRUE,ellipses = TRUE,pch = 21,bg = rainbow(n = length(var\_1)-1))

 \* Fractal dimesions of and Area, Smoothness, compactness

p1 <- ggplot(cancer\_df,aes(x = fractal.dim,y = area,col = label)) + geom\_point(alpha = 0.6) + labs(title = "fractal.dim vs Area") +   
 theme(plot.title = element\_text(hjust = 0.5),legend.position = c(0.8,0.8),legend.background = element\_blank(),legend.title = element\_blank())  
  
p2 <- ggplot(cancer\_df,aes(x = fractal.dim,y = compactness,col = label)) + geom\_point(alpha = 0.6) + labs(title = "fractal.dim vs compactness") +   
 theme(plot.title = element\_text(hjust = 0.5),legend.position = c(0.8,0.8),legend.background = element\_blank(),legend.title = element\_blank())  
  
p3 <- ggplot(cancer\_df,aes(x = fractal.dim,y = Symmetry,col = label)) + geom\_point(alpha = 0.6) + labs(title = "fractal.dim vs Symmetry") +   
 theme(plot.title = element\_text(hjust = 0.5),legend.position = c(0.8,0.8),legend.background = element\_blank(),legend.title = element\_blank())  
  
p4 <- ggplot(cancer\_df,aes( x= fractal.dim,y = concavity,col = label)) + geom\_point(alpha = 0.6) + labs(title = "fractal.dim vs concavity") +   
 theme(plot.title = element\_text(hjust = 0.5),legend.position = c(0.8,0.8),legend.background = element\_blank(),legend.title = element\_blank())  
  
gridExtra::grid.arrange(p1,p2,p3,p4)



* As it can be seen several of the parameters are exhibit strong correlation with rest of the other parameters
* Fractal dimensions of Malignant Cancer Cells have
* higher values of area, symmetricity, concavity, and compactness against fractal.dimesions are clear indicators of Malignant cells
* inverse relationship with area
* direct relationship with compactness, concavity,Compactness
* Assuming Benign = 0 no cancer case, Malignant = 1 as the case with positive class of breast cancer B = 0, M = 1

diagnosis <- factor(ifelse(test = cancer\_df$label == "M",yes = 1,no = 0))  
cancer\_df$label <- diagnosis

*Sampling the dataset and splitting the dataset into train and test*

set.seed(100)  
indices\_can <- sample(1:nrow(cancer\_df),size = 0.7\*nrow(cancer\_df))  
train\_df\_cancer <- cancer\_df[indices\_can,]  
test\_df\_cancer <- cancer\_df[-indices\_can,]

*checking the spread of class label*

table(train\_df\_cancer$label)

##   
## 0 1   
## 252 146

prop.table(table(train\_df\_cancer$label))\*100 # this is an imbalanced dataset where benign cases are higher than the Malignant tumor cases

##   
## 0 1   
## 63.31658 36.68342

# (b) Develop at least three classification models using different machine learning techniques.

# ——————-

## Model Building

# ——————-

### Using logistic regression

train\_control\_logit <- trainControl(method = "repeatedcv",repeats = 5,number = 5,search = "random",allowParallel = TRUE)  
model\_logit <- caret::train(label~.,data = train\_df\_cancer[var\_2],method = "glm",trControl = train\_control\_logit)

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred  
  
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

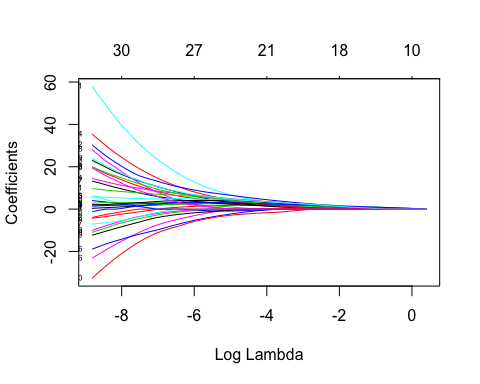
-Variables in the dataset have a perfect correlation between them, thus using regularization is needed in order to penalize the coefficients that exhibit high degree of collinearlity and leading to overfitting

set.seed(100)  
z\_predictors <- setdiff(names(train\_df\_cancer),"label")  
#model\_lasso <- glmnet::glmnet(x = z\_predictors,y = label,,family = "binomial",alpha = 1,lambda = seq(0,0.5,length.out = 50))  
model\_glmnet <- caret::train(x = train\_df\_cancer[,z\_predictors],y = train\_df\_cancer[,c("label")],method = "glmnet",  
 trControl = train\_control\_logit)  
  
model\_glmnet

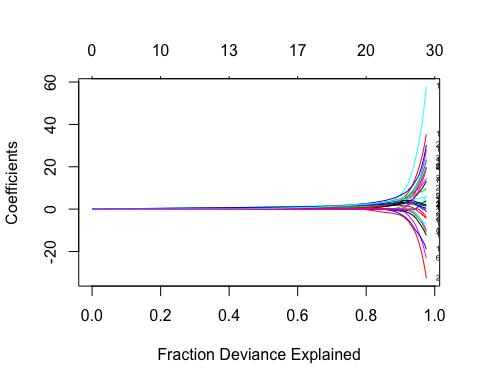
## glmnet   
##   
## 398 samples  
## 30 predictor  
## 2 classes: '0', '1'   
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold, repeated 5 times)   
## Summary of sample sizes: 317, 319, 319, 319, 318, 318, ...   
## Resampling results across tuning parameters:  
##   
## alpha lambda Accuracy Kappa   
## 0.05638315 1.47553079 0.8628268 0.6781687  
## 0.25767250 0.06657575 0.9683540 0.9306031  
## 0.55232243 0.07636310 0.9547587 0.8998455  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were alpha = 0.2576725 and lambda  
## = 0.06657575.

* With k-fold cross validation, 319 samples were used
* most important predictors

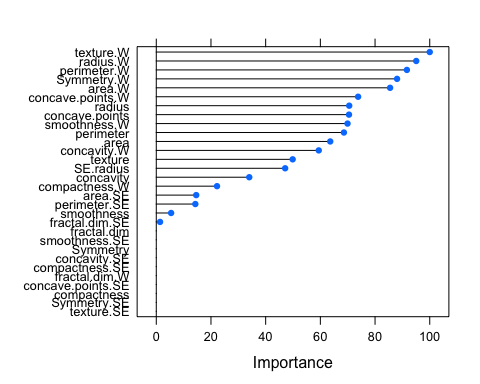
plot(model\_glmnet$finalModel,label = T,xvar = "lambda")



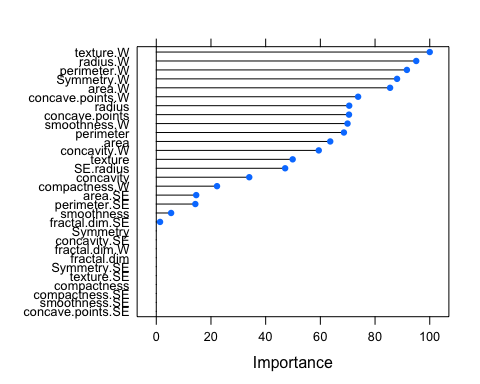
plot(model\_glmnet$finalModel,label = T,xvar = "dev")



plot(varImp(model\_glmnet))

 - 80% of the variablility in the dataset is explained by the 11 variables in the dataset and the remaining variables have grown steeply leading to high collinearity and overfitting

plot(varImp(model\_glmnet))



* The 11 variables are show below by the glmfit

varImp(model\_glmnet)

## glmnet variable importance  
##   
## only 20 most important variables shown (out of 30)  
##   
## Overall  
## texture.W 100.000  
## radius.W 95.055  
## perimeter.W 91.642  
## Symmetry.W 88.047  
## area.W 85.501  
## concave.points.W 73.804  
## radius 70.552  
## concave.points 70.472  
## smoothness.W 69.938  
## perimeter 68.611  
## area 63.613  
## concavity.W 59.414  
## texture 49.874  
## SE.radius 47.123  
## concavity 34.002  
## compactness.W 22.208  
## area.SE 14.600  
## perimeter.SE 14.284  
## smoothness 5.424  
## fractal.dim.SE 1.379

* The best tuning parameters of ElasticNet Regression are:

model\_glmnet$bestTune

## alpha lambda  
## 2 0.2576725 0.06657575

# predicting the glmnet parameters

predict\_glmfit <- predict(object = model\_glmnet,newdata = test\_df\_cancer[z\_predictors])#,type = "response")  
confusionMatrix(factor(predict\_glmfit),test\_df\_cancer$label,positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 104 4  
## 1 1 62  
##   
## Accuracy : 0.9708   
## 95% CI : (0.9331, 0.9904)  
## No Information Rate : 0.614   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9378   
## Mcnemar's Test P-Value : 0.3711   
##   
## Sensitivity : 0.9394   
## Specificity : 0.9905   
## Pos Pred Value : 0.9841   
## Neg Pred Value : 0.9630   
## Prevalence : 0.3860   
## Detection Rate : 0.3626   
## Detection Prevalence : 0.3684   
## Balanced Accuracy : 0.9649   
##   
## 'Positive' Class : 1   
##

*Comments*: Miss-rate is: 3. specifies that 3 records though were

### Using Support Vector Machines

train\_control\_svm <- trainControl(method = "repeatedcv",  
 repeats = 5,  
 number = 5,  
 search = "grid"  
 )  
  
tuning\_grid\_svm <- expand.grid(.sigma = seq(0,0.05,0.01),.C = seq(0.1,3,0.5))   
# Sigma = non-linearity control parameter  
# C = cost function paratmeter to number of misclassifcations control in SVM  
  
model\_svm <- ksvm(label ~.,  
 kernel = "polydot", # Radial Basis Function Kernel, Polydot kernels are superior  
 data = train\_df\_cancer,  
 trControl = trn\_cntrol\_svm,  
 tuneGrid = tuning\_grid\_svm,  
 metric = "auc"  
 )

## Setting default kernel parameters

model\_svm

## Support Vector Machine object of class "ksvm"   
##   
## SV type: C-svc (classification)   
## parameter : cost C = 1   
##   
## Polynomial kernel function.   
## Hyperparameters : degree = 1 scale = 1 offset = 1   
##   
## Number of Support Vectors : 28   
##   
## Objective Function Value : -16.1825   
## Training error : 0.007538

# predicting the parameters  
predicted\_svm <- predict(model\_svm,test\_df\_cancer[z\_predictors],type = "response")  
# confusion matrix to check the accuracy of the model  
confusionMatrix(data = predicted\_svm,reference = test\_df\_cancer$label,positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 101 2  
## 1 4 64  
##   
## Accuracy : 0.9649   
## 95% CI : (0.9252, 0.987)  
## No Information Rate : 0.614   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9264   
## Mcnemar's Test P-Value : 0.6831   
##   
## Sensitivity : 0.9697   
## Specificity : 0.9619   
## Pos Pred Value : 0.9412   
## Neg Pred Value : 0.9806   
## Prevalence : 0.3860   
## Detection Rate : 0.3743   
## Detection Prevalence : 0.3977   
## Balanced Accuracy : 0.9658   
##   
## 'Positive' Class : 1   
##

### Using Random Forest

# Train control  
train\_control\_rf <- trainControl(method = "repeatedcv",  
 repeats = 5,  
 number = 5,  
 search = "grid",  
 sampling = "smote",  
 allowParallel = TRUE)  
  
# Tuning grid parameters of Random Forest  
tuning\_grid\_rf <- expand.grid(.mtry = round(sqrt(ncol(train\_df\_cancer))),ntree = seq(100,500,100))   
  
model\_rf <- randomForest(label ~.,  
 data = train\_df\_cancer,  
 #method = "rf",  
 trControl = train\_control\_rf,  
 tuneGrid = tuning\_grid\_rf,  
 metric = "auc")  
  
model\_rf$importance

## MeanDecreaseGini  
## radius 7.3953101  
## texture 3.1419139  
## perimeter 10.9513108  
## area 8.3137987  
## smoothness 1.0772928  
## compactness 1.8511822  
## concavity 7.4471696  
## concave.points 16.9051658  
## Symmetry 0.7149775  
## fractal.dim 0.5276583  
## SE.radius 1.9118863  
## texture.SE 0.7953776  
## perimeter.SE 2.6020721  
## area.SE 7.1423376  
## smoothness.SE 0.6583737  
## compactness.SE 0.8444405  
## concavity.SE 1.1100036  
## concave.points.SE 0.8458098  
## Symmetry.SE 0.6723726  
## fractal.dim.SE 0.8097716  
## radius.W 18.6198145  
## texture.W 3.5228826  
## perimeter.W 25.8924795  
## area.W 24.1959641  
## smoothness.W 2.0870272  
## compactness.W 2.9855770  
## concavity.W 8.3902906  
## concave.points.W 19.7043292  
## Symmetry.W 2.3192125  
## fractal.dim.W 0.8539361

* Evaluating the svm model on the test\_data

predict\_rf <- stats::predict(object = model\_rf,test\_df\_cancer[z\_predictors])  
  
# Confusion Matrix  
confusionMatrix(predict\_rf,test\_df\_cancer$label,positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 100 5  
## 1 5 61  
##   
## Accuracy : 0.9415   
## 95% CI : (0.8951, 0.9716)  
## No Information Rate : 0.614   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.8766   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9242   
## Specificity : 0.9524   
## Pos Pred Value : 0.9242   
## Neg Pred Value : 0.9524   
## Prevalence : 0.3860   
## Detection Rate : 0.3567   
## Detection Prevalence : 0.3860   
## Balanced Accuracy : 0.9383   
##   
## 'Positive' Class : 1   
##

# Using XGBoost

# preparing the trainingControl   
trn\_cntrol\_xgb <- caret::trainControl(method = "repeatedcv",  
 number = 5,  
 repeats = 3,  
 #summaryFunction = twoClassSummary, # Use AUC to pick the best model  
 allowParallel = TRUE)  
  
# and tuning parapmeters for the xgb tree  
xgb.grid <- expand.grid(eta = seq(0.1,0.31, 0.1),  
 nrounds = c(50, 75, 100),  
 max\_depth = 3:5, # 4  
 min\_child\_weight = c(2.0, 2.25), #2   
 colsample\_bytree = c(0.3, 0.4, 0.5), # 3  
 gamma = 0, #1  
 subsample = 1) # 1  
  
# Modelling the dataset using the algorithm  
model\_xgb <- caret::train(label ~ .,   
 data = train\_df\_cancer,  
 method = "xgbTree",  
 trControl = trn\_cntrol\_xgb,  
 tuneGrid = xgb.grid)  
  
#  
print(model\_xgb$finalModel)

## ##### xgb.Booster  
## raw: 36.6 Kb   
## call:  
## xgboost::xgb.train(params = list(eta = param$eta, max\_depth = param$max\_depth,   
## gamma = param$gamma, colsample\_bytree = param$colsample\_bytree,   
## min\_child\_weight = param$min\_child\_weight, subsample = param$subsample),   
## data = x, nrounds = param$nrounds, objective = "binary:logistic")  
## params (as set within xgb.train):  
## eta = "0.3", max\_depth = "4", gamma = "0", colsample\_bytree = "0.4", min\_child\_weight = "2.25", subsample = "1", objective = "binary:logistic", silent = "1"  
## xgb.attributes:  
## niter  
## callbacks:  
## cb.print.evaluation(period = print\_every\_n)  
## # of features: 30   
## niter: 100  
## nfeatures : 30   
## xNames : radius texture perimeter area smoothness compactness concavity concave.points Symmetry fractal.dim SE.radius texture.SE perimeter.SE area.SE smoothness.SE compactness.SE concavity.SE concave.points.SE Symmetry.SE fractal.dim.SE radius.W texture.W perimeter.W area.W smoothness.W compactness.W concavity.W concave.points.W Symmetry.W fractal.dim.W   
## problemType : Classification   
## tuneValue :  
## nrounds max\_depth eta gamma colsample\_bytree min\_child\_weight  
## 138 100 4 0.3 0 0.4 2.25  
## subsample  
## 138 1  
## obsLevels : 0 1   
## param :  
## list()

model\_xgb$bestTune

## nrounds max\_depth eta gamma colsample\_bytree min\_child\_weight  
## 138 100 4 0.3 0 0.4 2.25  
## subsample  
## 138 1

### extremeGradient Boosing prediction

# predicting the val  
predict\_xgb <- predict(object = model\_xgb,test\_df\_cancer[z\_predictors])  
  
# confusion matrix  
confusionMatrix(data = predict\_xgb,reference = test\_df\_cancer$label,positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 102 2  
## 1 3 64  
##   
## Accuracy : 0.9708   
## 95% CI : (0.9331, 0.9904)  
## No Information Rate : 0.614   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9385   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9697   
## Specificity : 0.9714   
## Pos Pred Value : 0.9552   
## Neg Pred Value : 0.9808   
## Prevalence : 0.3860   
## Detection Rate : 0.3743   
## Detection Prevalence : 0.3918   
## Balanced Accuracy : 0.9706   
##   
## 'Positive' Class : 1   
##

# ——————————

# (c) Explain in detail which model is better and why?

# ——————————

* Comparing different sets of models glmnet,svm,randomforest, and XgBoostTree; the glmnet model is the best model from the available set of data Advantages:

1. glmnet data, has best specificity = 1, and sensitivity = 0.98 and balanced accuracy = 0.977,
2. glmnet explains the variablility between the coefficients and clearly orders the variables of importance
3. glmnet regularizes the coeficients controlling the overfitting problem associated with the inclusion of large number of predictors
4. with large datasets the model converges faster and more higher accuracy
5. Although the model performs well on the dataset, it suffers with hyperparameter tuning of alpha and lambda part of regularization.

* Next best model in the list the Support Vector machines. Undoubtedly SVM outperforms the glmnet in terms of numbers, however, it suffers from some of the major drawbacks. Some of them are:

1. Since the dataset is smaller svm has converged faster, with large datasets support vector machines takes larger durations to converge
2. with kernel implementation the abstract behavior of the svm is difficult to analyse the output.

# ————————-

# (d) Develop principal components for all independent variables, develop the ML models on principal components. Compare these models with previous models. What are the advantages (if any) of using PCA in classification model building?

# ————————-

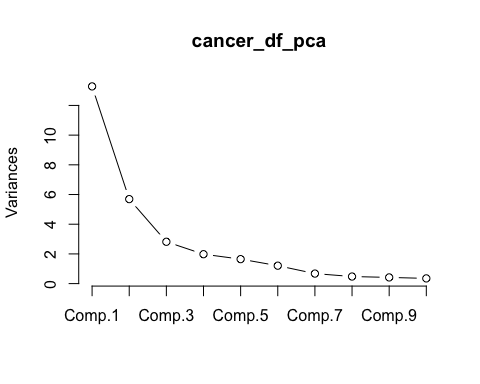
### principal component analysis

cancer\_df\_pca <- princomp(cancer\_df[,-1],cor = TRUE,scores = TRUE)  
summary(cancer\_df\_pca)

## Importance of components:  
## Comp.1 Comp.2 Comp.3 Comp.4  
## Standard deviation 3.6443940 2.3856560 1.67867477 1.40735229  
## Proportion of Variance 0.4427203 0.1897118 0.09393163 0.06602135  
## Cumulative Proportion 0.4427203 0.6324321 0.72636371 0.79238506  
## Comp.5 Comp.6 Comp.7 Comp.8  
## Standard deviation 1.28402903 1.09879780 0.82171778 0.69037464  
## Proportion of Variance 0.05495768 0.04024522 0.02250734 0.01588724  
## Cumulative Proportion 0.84734274 0.88758796 0.91009530 0.92598254  
## Comp.9 Comp.10 Comp.11 Comp.12  
## Standard deviation 0.64567392 0.59219377 0.54213992 0.511039500  
## Proportion of Variance 0.01389649 0.01168978 0.00979719 0.008705379  
## Cumulative Proportion 0.93987903 0.95156881 0.96136600 0.970071383  
## Comp.13 Comp.14 Comp.15 Comp.16  
## Standard deviation 0.49128148 0.396244525 0.306814219 0.282600072  
## Proportion of Variance 0.00804525 0.005233657 0.003137832 0.002662093  
## Cumulative Proportion 0.97811663 0.983350291 0.986488123 0.989150216  
## Comp.17 Comp.18 Comp.19 Comp.20  
## Standard deviation 0.243719178 0.229387845 0.222435590 0.176520261  
## Proportion of Variance 0.001979968 0.001753959 0.001649253 0.001038647  
## Cumulative Proportion 0.991130184 0.992884143 0.994533397 0.995572043  
## Comp.21 Comp.22 Comp.23 Comp.24  
## Standard deviation 0.1731268145 0.1656484305 0.1560155049 0.1343689213  
## Proportion of Variance 0.0009990965 0.0009146468 0.0008113613 0.0006018336  
## Cumulative Proportion 0.9965711397 0.9974857865 0.9982971477 0.9988989813  
## Comp.25 Comp.26 Comp.27 Comp.28  
## Standard deviation 0.1244237573 0.090430304 0.0830690308 3.986650e-02  
## Proportion of Variance 0.0005160424 0.000272588 0.0002300155 5.297793e-05  
## Cumulative Proportion 0.9994150237 0.999687612 0.9999176271 9.999706e-01  
## Comp.29 Comp.30  
## Standard deviation 0.0273642668 1.153451e-02  
## Proportion of Variance 0.0000249601 4.434827e-06  
## Cumulative Proportion 0.9999955652 1.000000e+00

* From the screeplot it can be inferred that the after Comp.7, the increses in the variances is not signifcant enough.
* Thus Top 8 components explain 93% of the variance among the total 30 components.

screeplot(cancer\_df\_pca,type = "l")



ggbiplot(pcobj = cancer\_df\_pca,groups = cancer\_df$label,ellipse = TRUE,obs.scale = 2,var.scale = 2,circle = T,ellipse.prob = 0.68) +  
 scale\_color\_discrete(name = "") + theme(legend.position = c(0.8,0.1),legend.background = element\_blank())



* 68% of the probability is explained by the red ellipse and this explains the portion of the principal component with Benign cases
* 32% of the probability is explained by the green ellipse records
* It can be inferred that PC1 and fractal dimensions, symmetry, smoothness, compactness have a negatiave correalation while positive correlation with the rest of the other quantitative parameters, area, perimeter, radius, etc

# predicting the training and test datasets using PCA

train\_df\_pca <- data.frame(predict(cancer\_df\_pca,train\_df\_cancer[,-1]),label = train\_df\_cancer[,1])  
test\_df\_pca <- data.frame(predict(cancer\_df\_pca,test\_df\_cancer[,-1]),label = test\_df\_cancer[,1])

# Model Building

Using logistic regression

train\_control\_logit <- trainControl(method = "repeatedcv",repeats = 5,number = 5,search = "random",allowParallel = TRUE)  
model\_logit\_pca <- caret::train(label~ Comp.1,data = train\_df\_pca,method = "glm",trControl = train\_control\_logit)  
  
z\_predictors\_pca <- setdiff(names(train\_df\_pca),"label")  
pred\_logit\_pca <- predict(model\_logit\_pca,test\_df\_pca[z\_predictors\_pca])  
confusionMatrix(pred\_logit\_pca,test\_df\_pca$label,positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 100 7  
## 1 5 59  
##   
## Accuracy : 0.9298   
## 95% CI : (0.8806, 0.9632)  
## No Information Rate : 0.614   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.8511   
## Mcnemar's Test P-Value : 0.7728   
##   
## Sensitivity : 0.8939   
## Specificity : 0.9524   
## Pos Pred Value : 0.9219   
## Neg Pred Value : 0.9346   
## Prevalence : 0.3860   
## Detection Rate : 0.3450   
## Detection Prevalence : 0.3743   
## Balanced Accuracy : 0.9232   
##   
## 'Positive' Class : 1   
##

* model has predicted low specificity, yet the model converged with considering the Comp.1
* The model Component.1 was wieghted values values of each variable that expains the contribution of each predictor to the model accuracy.

### Using random forest model with PCA comonents

train\_control\_rf <- trainControl(method = "repeatedcv",  
 repeats = 5,  
 number = 5,  
 search = "grid",  
 sampling = "smote",  
 allowParallel = TRUE)  
  
# Tuning grid parameters of Random Forest  
tuning\_grid\_rf <- expand.grid(.mtry = round(sqrt(ncol(train\_df\_cancer))),ntree = seq(100,500,100))   
  
model\_rf\_pca <- randomForest(label ~ Comp.1+Comp.2+Comp.3+Comp.4+Comp.5+Comp.6+Comp.7,  
 data = train\_df\_pca,  
 #method = "rf",  
 trControl = train\_control\_rf,  
 tuneGrid = tuning\_grid\_rf,  
 metric = "auc")  
  
pred\_rf\_pca <- predict(object = model\_rf\_pca,test\_df\_pca[z\_predictors\_pca])  
confusionMatrix(pred\_rf\_pca,test\_df\_pca$label,positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 101 3  
## 1 4 63  
##   
## Accuracy : 0.9591   
## 95% CI : (0.9175, 0.9834)  
## No Information Rate : 0.614   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9139   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9545   
## Specificity : 0.9619   
## Pos Pred Value : 0.9403   
## Neg Pred Value : 0.9712   
## Prevalence : 0.3860   
## Detection Rate : 0.3684   
## Detection Prevalence : 0.3918   
## Balanced Accuracy : 0.9582   
##   
## 'Positive' Class : 1   
##

* with Random Forest there is no significant improvement in the Accuracy, however, there are more misclassifications in the predicted values.
* with several components

# Using support vector machines with pca components

train\_control\_svm <- trainControl(method = "repeatedcv",  
 repeats = 5,  
 number = 5,  
 search = "grid"  
 )  
  
tuning\_grid\_svm <- expand.grid(.sigma = seq(0,0.05,0.01),.C = seq(0.1,3,0.5))   
# Sigma = non-linearity control parameter  
# C = cost function paratmeter to number of misclassifcations control in SVM  
  
model\_svm\_pca <- ksvm(label ~ Comp.1+Comp.2+Comp.3+Comp.4+Comp.5,  
 kernel = "polydot", # Radial Basis Function Kernel, Polydot kernels are superior  
 data = train\_df\_pca,  
 trControl = trn\_cntrol\_svm,  
 tuneGrid = tuning\_grid\_svm,  
 metric = "auc"  
 )

## Setting default kernel parameters

model\_svm

## Support Vector Machine object of class "ksvm"   
##   
## SV type: C-svc (classification)   
## parameter : cost C = 1   
##   
## Polynomial kernel function.   
## Hyperparameters : degree = 1 scale = 1 offset = 1   
##   
## Number of Support Vectors : 28   
##   
## Objective Function Value : -16.1825   
## Training error : 0.007538

pred\_svm\_pca <- predict(model\_svm\_pca,test\_df\_pca[z\_predictors\_pca])  
confusionMatrix(pred\_svm\_pca,test\_df\_pca$label,positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 100 3  
## 1 5 63  
##   
## Accuracy : 0.9532   
## 95% CI : (0.9099, 0.9796)  
## No Information Rate : 0.614   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9019   
## Mcnemar's Test P-Value : 0.7237   
##   
## Sensitivity : 0.9545   
## Specificity : 0.9524   
## Pos Pred Value : 0.9265   
## Neg Pred Value : 0.9709   
## Prevalence : 0.3860   
## Detection Rate : 0.3684   
## Detection Prevalence : 0.3977   
## Balanced Accuracy : 0.9535   
##   
## 'Positive' Class : 1   
##

# There are more misclassications with the less number of components, with an optimum till Comp.6. With more data the problem of overfitting can be further reduced and model can be further tuned to reach an optimum accuracy.

# Thus a PCA model is simpler compared to the models studied earlier, and robust with optimacy accuracy, although not as better as models without PCA. PCA weeds out the problem of overfitting. Further it is robust to changes to precision errors and modification in variable coefficients and predictors themselves.