# Classfication Model For Diabete Case Prediction

Ziyi Yan, Weiyu Wang

## About the project

The Pima Indians, primarily located in Arizona, United States, have been a focal point to numerous diabetes research projects due to the high prevalence of diabetes. Research shows that there is significant prove on the Pima Indians is the exploration of genetic factors contributing to diabetes. Studies have revealed that certain genetic variants are more prevalent in the Pima Indian population, which may increase their risk of developing diabetes.

This project examines diabetes prediction about the onset of diabetes based on diagnostic measures. The Dataset is from the National Institute of Diabetes and Digestive and Kidney Diseases. This dataset provides cross-sectional data of 2000 individual females of Pima Indian Heritage, who are at least 21 years old, with and without diabetes, and includes the mentioned possible contributing factors to diabetes. The dataset includes a total of 9 variables, which 8 are predictor variables: Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, and Age and 1 target variable: Outcome. This dataset will allow us to explore several different possible factors for diabetes and determine if these variables are sufficient to predict if a person has diabetes or not. If not, is it possible to develop a model that accurately predicts diabetes by integrating various health indicators from the dataset?

## About the dataset

### Who is Pima Indians?

"The Pima (or Akimel O'odham, also spelled Akimel O'otham,"River People", formerly known as Pima) are a group of Native Americans living in an area consisting of what is now central and southern Arizona. The majority population of the surviving two bands of the Akimel O'odham are based in two reservations: the Keli Akimel O'otham on the Gila River Indian Community (GRIC) and the On'k Akimel O'odham on the Salt River Pima-Maricopa Indian Community (SRPMIC)." Wikipedia[1]

### What is diabetes?

According to NIH, "Diabetes is a disease that occurs when your blood glucose, also called blood sugar, is too high. Blood glucose is your main source of energy and comes from the food you eat. Insulin, a hormone made by the pancreas, helps glucose from food get into your cells to be used for energy. Sometimes your body doesn't make enough—or any—insulin or doesn't use insulin well. Glucose then stays in your blood and doesn't reach your cells.

Over time, having too much glucose in your blood can cause health problems. Although diabetes has no cure, you can take steps to manage your diabetes and stay healthy.

Sometimes people call diabetes "a touch of sugar" or "borderline diabetes." These terms suggest that someone doesn't really have diabetes or has a less serious case, but every case of diabetes is serious.

### What are the different types of diabetes?

The most common types of diabetes are type 1, type 2, and gestational diabetes.

**Type 1 diabetes** If you have type 1 diabetes, your body does not make insulin. Your immune system attacks and destroys the cells in your pancreas that make insulin. Type 1 diabetes is usually diagnosed in

children and young adults, although it can appear at any age. People with type 1 diabetes need to take insulin every day to stay alive.

**Type 2 diabetes** If you have type 2 diabetes, your body does not make or use insulin well. You can develop type 2 diabetes at any age, even during childhood. However, this type of diabetes occurs most often in middle-aged and older people. Type 2 is the most common type of diabetes.

**Gestational diabetes**

Gestational diabetes develops in some women when they are pregnant. Most of the time, this type of diabetes goes away after the baby is born. However, if you've had gestational diabetes, you have a greater chance of developing type 2 diabetes later in life. Sometimes diabetes diagnosed during pregnancy is actually type 2 diabetes.

**Other types of diabetes**

Less common types include monogenic diabetes, which is an inherited form of diabetes, and cystic fibrosis-related diabetes."[2]

## Detail of the dataset

**Diabetes Data Set**

Source:

Diabetes Data Set (kaggle.com)

https://www.kaggle.com/datasets/vikasukani/diabetes-data-set/data

**Columns/ Variables**

*Pregnancies*: Number of times pregnant

*Glucose*: Plasma glucose concentration, 2 hours in an oral glucose tolerance test

*Blood Pressure*: Diastolic blood pressure (mm Hg)

*Skin Thickness*: Triceps skin fold thickness (mm)

*Insulin*: 2-Hour seruminsulin (mu U/ml)

*BMI*: Body mass index (weight in kg/(height in m)^2)

*Diabetes Pedigree Function*: Diabetes pedigree function

*Age*: Age (years)

*Outcome*: Class variable (0 or 1) 268 of 768 are 1, the others are 0

## Goal

As we all know, diabetes is still a lifelong disease with no cure. That means if a patient is diagnosed with diabetes, it is theoretically impossible to reverse it. Our goal is to try to accomplish the following by building a machine learning predictive model:

· The model can accurately predict whether an individual has diabetes or not.

· he model can tap into which risk factors best predict diabetes risk.

· We can use a subset of risk factors to accurately predict whether an individual has diabete s.

· We can use a screen for several important diabetes-causing characteristics and then combine them to create a short question that accurately predicts whether someone is likely to have diabetes or whether they are at high risk for diabetes.

## Statistic Method

### Ploting

Histograms plot

Scatter plot

Box plot

### Modeling

GLM

GLIM with probit

Random Forest

### Assessing tools

AUC score

ROC score

## Code

### 1. Data Preprocessing

```
#Load the dataset
data = read.csv('/Users/ziyiyan/Downloads/diabetes-dataset.csv')
```

```
#This will show the top a few samples of the dataset, but we will not print here
head(data)
```

```
#Attach some packages
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
## 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
#Checking for NULL values
which(is.na(data))
```

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

Here we do not have null values in the dataset, but we do have some 0 values.

**Handling missing or zero values for columns where zero is not even exist in real-word which could only have one reason for zero - missing data**

*Glucose*: A zero value for glucose is biologically implausible, as glucose is always present in the blood.

*BloodPressure*: A zero blood pressure would mean no blood circulation, which is not possible for a living individual.

*SkinThickness*: Zero skin thickness is not realistic in a physiological context.

*Insulin*: A zero value might indicate a missing measurement rather than an actual absence of insulin.

*BMI*: A Body Mass Index of zero is impossible for a living person.

```r
# It is medically impossible for the following variables to have 0 value
columns_with_zeros <- c("Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI")
data[columns_with_zeros] <- lapply(data[columns_with_zeros], function(x) replace(x, x == 0, NA))
```

```r
# Hence we replace them with the median value of that particular column
for (col in columns_with_zeros) {
  data[[col]] <- ifelse(is.na(data[[col]]), median(data[[col]], na.rm = TRUE), data[[col]])
}
```

We observe that the minimum value of some columns is 0, which is medically impossible. Hence in the data cleaning process, we replace them with median value depending on the distribution.

### 2. Data Visualization

```r
# #This will show some numerical summary of the dataset, but we will not print here
summary(data)
```

```r
## #This will show the correlation matrix of each variable, but we will not print here
correlation_matrix <- cor(data)
print(correlation_matrix)
```

**Histograms plot** An histogram is an accurate graphical representation of the distribution of numerical data. It takes as input one numerical variable only. The variable is cut into several bins, and the number of observation per bin is represented by the height of the bar.

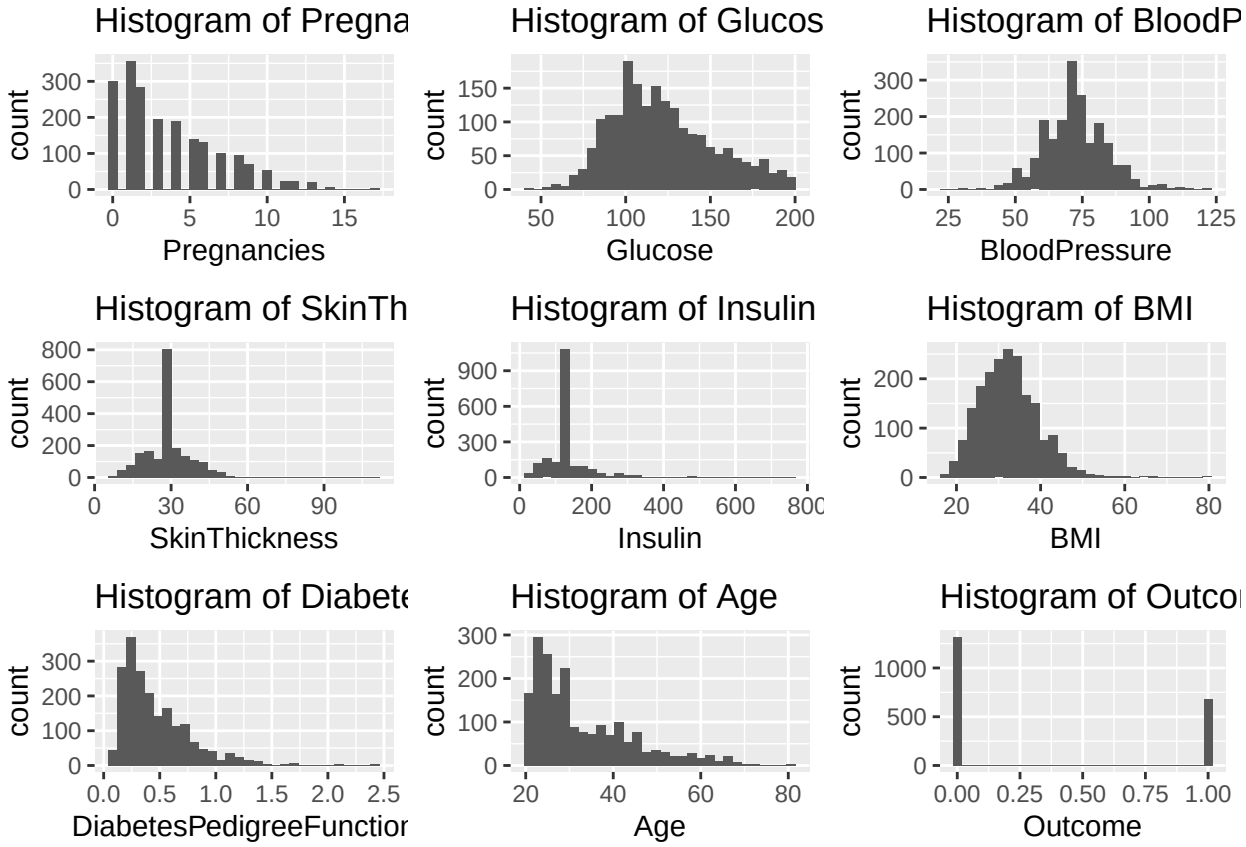
```r
library(ggplot2)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

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

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

## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

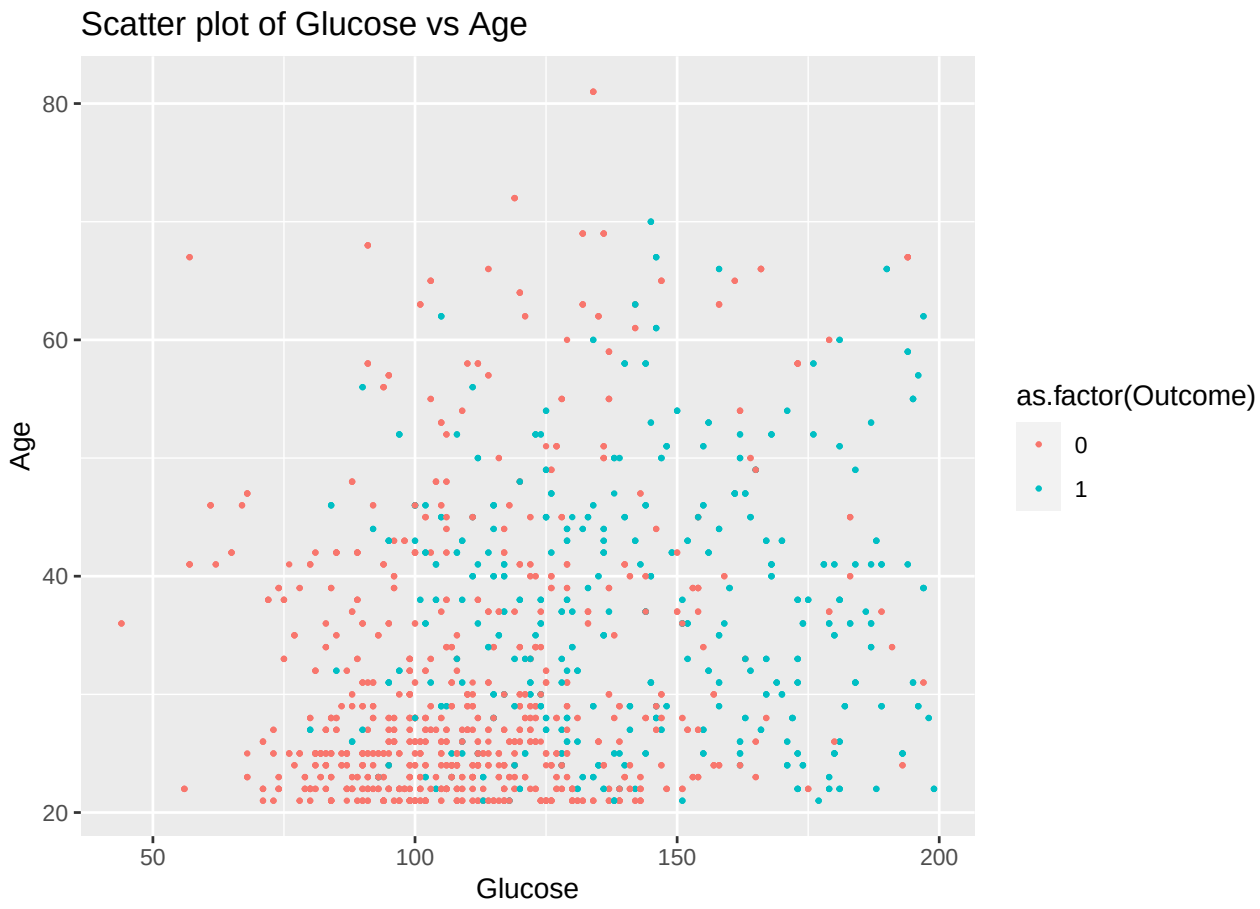



**From the histograms above**:

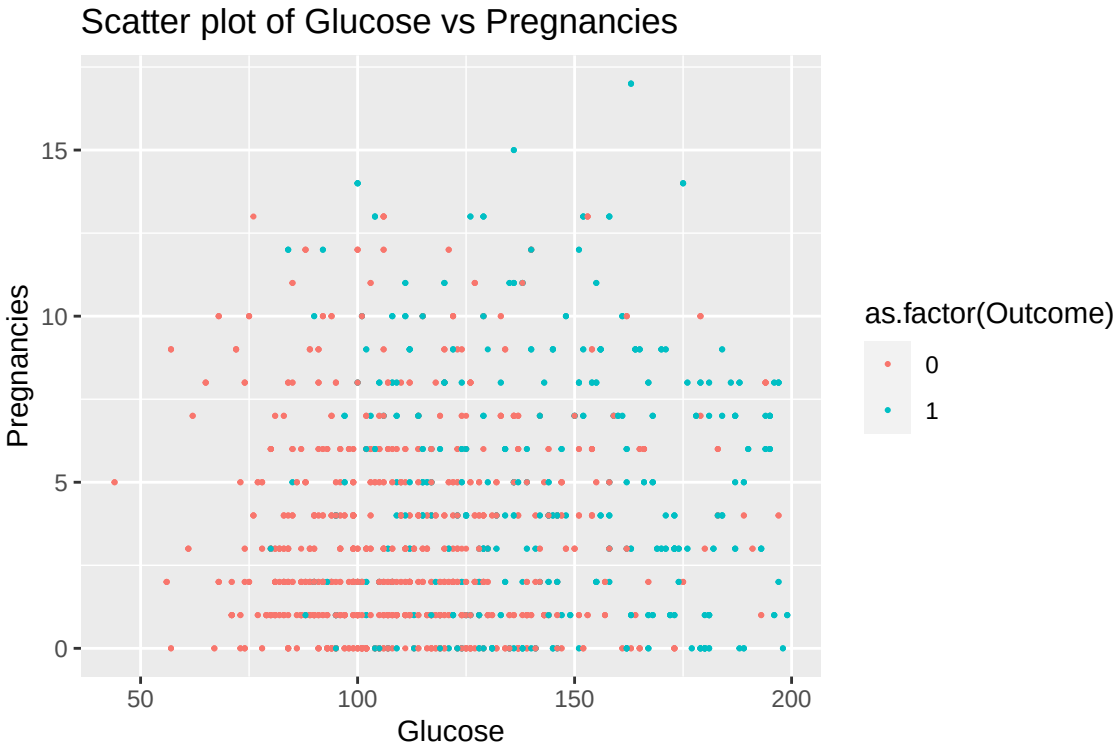
The distribution of *Pregnancies* shows that most values are clustered at the lower end, indicating that a significant portion of individuals in the dataset have few or no pregnancies. The *Glucose*, *BloodPressure*, *SkinThickness*, *Insulin*, and *BMI* variables display somewhat normal distributions, though with varying degrees of skewness. This is expected as these are continuous physiological measurements. DiabetesPedigreeFunction shows a right-skewed distribution, suggesting that higher values are less frequent. The Age distribution is relatively spread out, indicating a diverse age range in the dataset. For the *Outcome*, the dataset seems imbalanced with more non-diabetic cases (0) than diabetic cases (1).

**Scatter plot** A Scatter plot displays in 2x2 with 2 variables. Each dot on the plot represents an sample observation. The plot is mainly use to study the relationship between the variables. It is common to provide even more information using colors or shapes (to show groups, or a third variable). It is also possible to map another variable to the size of each dot, what makes a bubble plot. If you have many dots and struggle with overplotting, consider using 2D density plot.

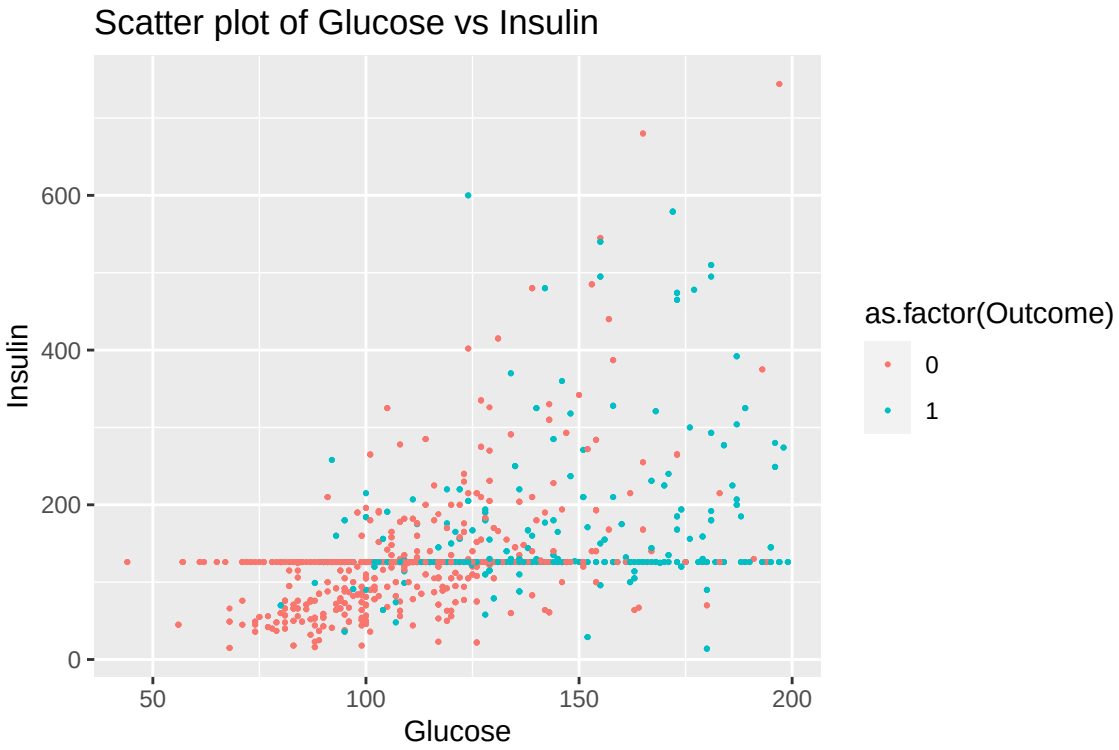
```
#Scatter plot of Glucose vs Age
scatter <- ggplot(data, aes(x = Glucose, y = Age, color = as.factor(Outcome))) + geom_point(size=0.5) +
  labs(title = "Scatter plot of Glucose vs Age")
print(scatter)
```




In the scatter plot of *Glucose vs Age*, there is a noticeable clustering of diabetic cases (Outcome = 1) at higher glucose levels regardless of age. This aligns with the medical understanding that higher glucose levels are a significant indicator of diabetes.

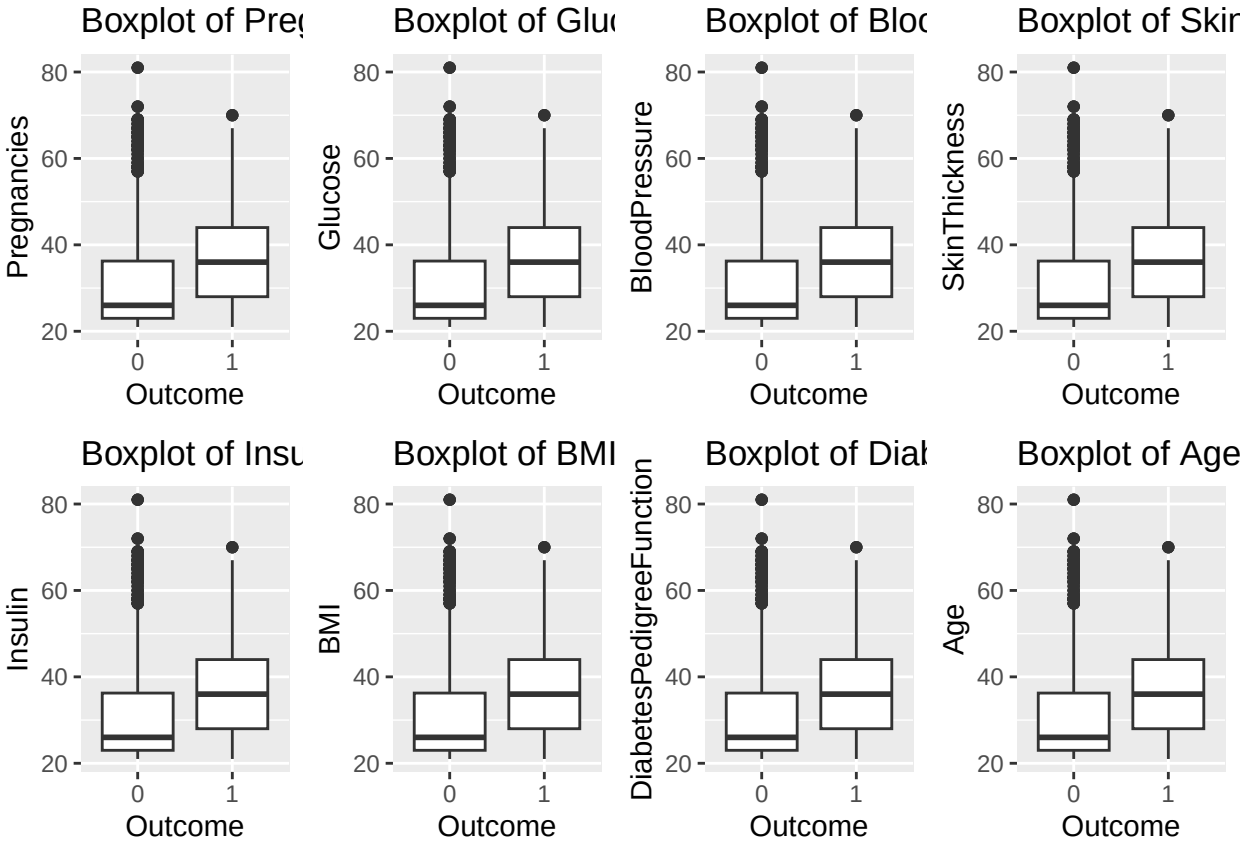



The *Glucose vs Pregnancies* scatter plot shows that higher glucose levels are associated with diabetes across different numbers of pregnancies. However, there doesn't seem to be a clear relationship between the number of pregnancies and glucose levels.




In the *Glucose vs Insulin* plot, diabetic cases are more prevalent at higher levels of both glucose and insulin. This is expected as insulin levels tend to increase in response to high glucose levels, especially in the case of insulin resistance, a common feature of type 2 diabetes.

**Box plot** Box Plot is a box shape that displays the data distribution through quartiles.




From the box plot above:

*Pregnancies*: There's a noticeable difference in the distribution of pregnancies between diabetic and non-diabetic individuals. Diabetic individuals tend to have a higher number of pregnancies on average.

*Glucose*: A clear distinction is visible, with higher glucose levels associated with diabetic individuals. This is consistent with the medical understanding of diabetes.

*Blood Pressure*: While there is some overlap, diabetic individuals tend to have slightly higher blood pressure. However, the difference is not as pronounced as with glucose.

*Skin Thickness*: Diabetic individuals generally have slightly thicker skin. The variance is also larger among diabetic individuals.

*Insulin*: Insulin levels are generally higher in diabetic individuals, with a broader spread in values indicating greater variability.

*BMI*: A higher BMI is observed in diabetic individuals. This suggests a possible link between obesity and diabetes, which is well-established in medical literature.

*Diabetes Pedigree Function*: There's a slight difference in the distribution of this variable, with diabetic individuals having higher values. This metric indicates the genetic influence on the individual's likelihood of developing diabetes.

*Age*: Age distribution indicates that diabetes is more prevalent in older individuals. The median age of diabetic individuals is higher than that of non-diabetic individuals.

### 3. Train test split

The train test split is a useful technique for evaluating the performance of machine learning algorithm.

Train Data: Used to fit the machine learning model.

Test Data: Used to evaluate the fit machine learning model.

```
train_columns <- sapply(data, is.numeric)
# Exclude the 'Outcome' column from scaling
train_columns["Outcome"] <- FALSE
# Scale the numeric columns except for 'Outcome'
data[train_columns] <- scale(data[train_columns])
```

Here in this project, we will use train: 80% and test: 20%.

```
set.seed(123)
train.prop <- 0.80
strats <- data$Outcome
rr <- split(1:length(strats), strats)
idx <- sort(as.numeric(unlist(sapply(rr,
        function(x) sample(x, length(x)*train.prop)))))
data.train <- data[idx, ]
data.test <- data[-idx, ]
```

### 4. Modeling - GLM

#### What is GLM?

The GLM stands for Generalized Linear Model. It is very common used and seen in R because of the flexible generalization of ordinary linear regression that allows for outcome variables that have error distribution models other than a normal distribution. GLM is used to model the relationship between a outcome variable and other predictor variables.

#### What can we know from the GLM?

**Coefficients**: the change in the log odds of the outcome per unit change in the predictor

**Standard Errors**: the statistical accuracy of the coefficients

**z-value**:coefficient / standard error

**P-value**: used to determine if the statistical is significant

```
# This is a model with all variables
full.logit <- glm(Outcome ~ . ,data = data.train,
                  family = binomial(link = "logit"))
summary(full.logit)
```

#### Modeling

```
##
## Call:
## glm(formula = Outcome ~ ., family = binomial(link = "logit"),
##     data = data.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0123  -0.7210  -0.4294   0.7562   2.1527
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)              -0.88074    0.06620 -13.305  < 2e-16 ***
## Pregnancies               0.37878    0.07351   5.153 2.57e-07 ***
## Glucose                   1.08667    0.07860  13.826  < 2e-16 ***
## BloodPressure            -0.01714    0.06976  -0.246  0.80593
## SkinThickness             0.05832    0.07728   0.755  0.45043
## Insulin                  -0.05726    0.06426  -0.891  0.37285
## BMI                       0.45365    0.07954   5.704 1.17e-08 ***
## DiabetesPedigreeFunction  0.21038    0.06636   3.170  0.00152 **
## Age                       0.17099    0.07505   2.278  0.02270 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2054.4  on 1598  degrees of freedom
## Residual deviance: 1528.8  on 1590  degrees of freedom
## AIC: 1546.8
##
## Number of Fisher Scoring iterations: 4
```

The output gives the GLM regression coefficients along with their standard errors, z-test statistics, and p-values.

For example, the estimated coefficient for *BloodPressure* is -0.01714 with a SE of 0.06976. This means that the decrease in expected log count $\log \lambda_i$ for one unit increase in *BloodPressure* is 0.01714. The p-value for *BloodPressure* is 0.80593, where we used to have alpha set as 0.05. The p-value for *BloodPressure* is greater than 0.05, hence we will say that it is not considered statistically significant. But *DiabetesPedigreeFunction* and *Age* both have a p-value that is less than 0.05, these are the only two variables considered as statistically significant.

The output of the model also shows values for null deviance and residual deviance.The residual deviance tells us how well the response variable can be predicted by a model with the intercept and predictor variables.

The model shows: Null deviance: 2054.4 on 1598 degrees of freedom Residual deviance: 1528.8 on 1590 degrees of freedom

```
# This is a model with no varibles,it is used to test the significance of the predictors in the full mo
null.logit <- glm(Outcome ~ 1, data = data.train,
                  family = binomial(link = "logit"))
summary(null.logit)
```

```
# This is a stepwise logit model, it uses both backward and forward selection to choose predictors for
both.logit <- step(null.logit, list(lower = formula(null.logit),
                                    upper = formula(full.logit)),
                   direction = "both", trace = 0, data = data.train)
formula(both.logit)
```

```
## Outcome ~ Glucose + Pregnancies + BMI + DiabetesPedigreeFunction +
##     Age
```

```
summary(both.logit)
```

```
##
## Call:
## glm(formula = Outcome ~ Glucose + Pregnancies + BMI + DiabetesPedigreeFunction +
##     Age, family = binomial(link = "logit"), data = data.train)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -3.0475  -0.7211  -0.4325   0.7547   2.1509
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -0.87911    0.06596 -13.328  < 2e-16 ***
## Glucose                    1.06326    0.07268  14.630  < 2e-16 ***
## Pregnancies                0.37671    0.07317   5.148 2.63e-07 ***
## BMI                        0.46962    0.06766   6.941 3.89e-12 ***
## DiabetesPedigreeFunction   0.20904    0.06613   3.161  0.00157 **
## Age                        0.17577    0.07227   2.432  0.01501 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2054.4  on 1598  degrees of freedom
## Residual deviance: 1530.2  on 1593  degrees of freedom
## AIC: 1542.2
##
## Number of Fisher Scoring iterations: 4
```

**Assessing accuracy What is AUC?**

AUC measures the entire two-dimensional area below the entire ROC curve from (0,0) to (1,1). It provides an aggregated measure of the model's performance over all possible classification thresholds. An AUC score of 1 indicates a perfect model, while a score of 0.5 suggests no discriminative ability, equivalent to random guessing.

**What is ROC?**

ROC curves are graphs that illustrate the diagnostic ability of binary classifiers in recognizing changes in thresholds

We calculate and contrast the confusion matrices using the following code. When the predicted probability exceeds a certain threshold, we categorize the predicted response as 1; otherwise, it is categorized as 0. In these matrices, the rows represent the predictions, where 'FALSE' indicates a prediction of 0, and 'TRUE' indicates a prediction of 1.

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
#Assess train data accuracy
pred.both <- predict(both.logit, newdata = data.train, type = "response")
```

```r
(table.both <- table(pred.both > 0.5, data.train$Outcome))
```

```
##
##           0   1
##   FALSE 935 257
##   TRUE  117 290
```

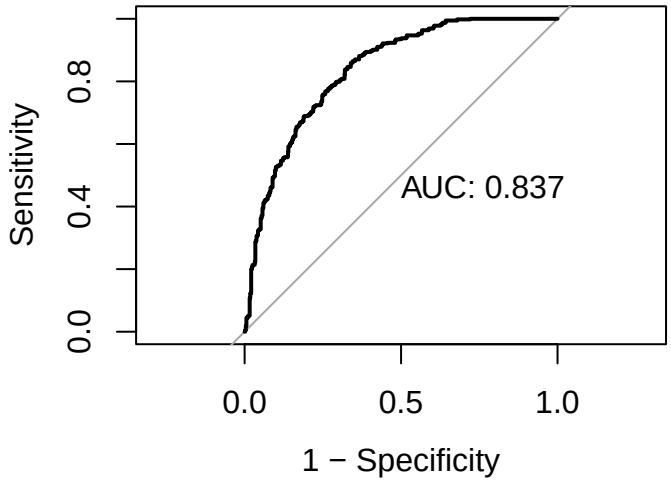
```
(accuracy.both <- round((sum(diag(table.both))/sum(table.both))*100, 3))
```

```
## [1] 76.61
```

```
roc.both <- roc(data.train$Outcome ~ pred.both, plot = TRUE,
                legacy.axes = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
#Assess test data accuracy
pred.both.test <- predict(both.logit, newdata = data.test, type = "response")
```

```
(table.both.test <- table(pred.both.test > 0.5, data.test$Outcome))
```

```
##
##           0   1
##   FALSE 239  60
##   TRUE   25  77
```

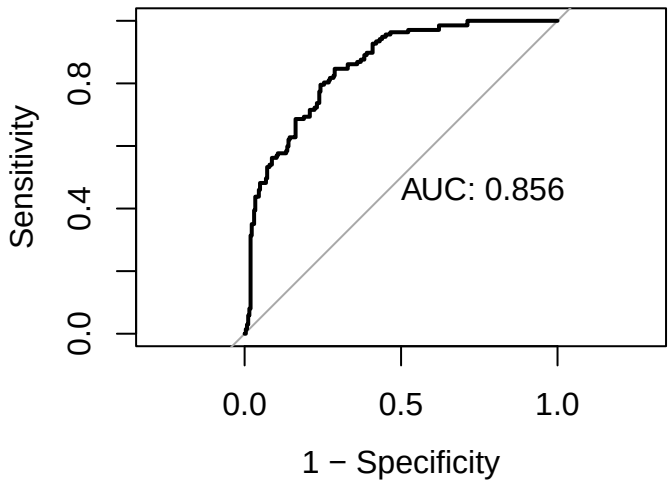
```
(accuracy.both.test <- round((sum(diag(table.both.test))/sum(table.both.test))*100, 3))
```

```
## [1] 78.803
```

```
roc.both.test <- roc(data.test$Outcome ~ pred.both.test, plot = TRUE,
                legacy.axes = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




**Modeling with interactions** We fit a model that incorporates second-order interactions among all predictor variables. We perform a stepwise selection process to identify the most significant variables, then proceed to make predictions using the optimal set of predictors identified through this selection process.

```
#Set the all logit formula
variables <- train_columns
interaction.var <-
  combn(variables, 2,
    FUN = function(x)
      paste(x, collapse = ":")
  )
formula.inter <-
  as.formula(paste("Outcome ~ . +", paste(interaction.var, collapse = "+")))
all.logit <- glm(formula = formula.inter,
      family = binomial(link = "logit"),
      data = data.train)
```

```
summary(all.logit )
```

```
##
## Call:
## glm(formula = formula.inter, family = binomial(link = "logit"),
##     data = data.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1821  -1.0408  -0.6805   0.5380   1.7157
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## Pregnancies                0.370675   0.072717   5.098 3.44e-07 ***
## Glucose                    1.067215   0.079299  13.458  < 2e-16 ***
## BloodPressure              0.002295   0.065328   0.035   0.9720
## SkinThickness             -0.004903   0.072827  -0.067   0.9463
## Insulin                   -0.093022   0.066586  -1.397   0.1624
## BMI                        0.416501   0.076906   5.416 6.11e-08 ***
## DiabetesPedigreeFunction   0.223023   0.064074   3.481   0.0005 ***
## Age                        0.121807   0.073603   1.655   0.0979 .
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2216.7  on 1599  degrees of freedom
## Residual deviance: 1729.1  on 1591  degrees of freedom
## AIC: 1745.1
##
## Number of Fisher Scoring iterations: 4
```

```
#Assess train data accuracy
pred.all <- predict(all.logit, newdata = data.train, type = "response")
```

```
(table.all <- table(pred.all > 0.5, data.train$Outcome))
```

```
##
##           0   1
##   FALSE 754 118
##   TRUE  298 429
```

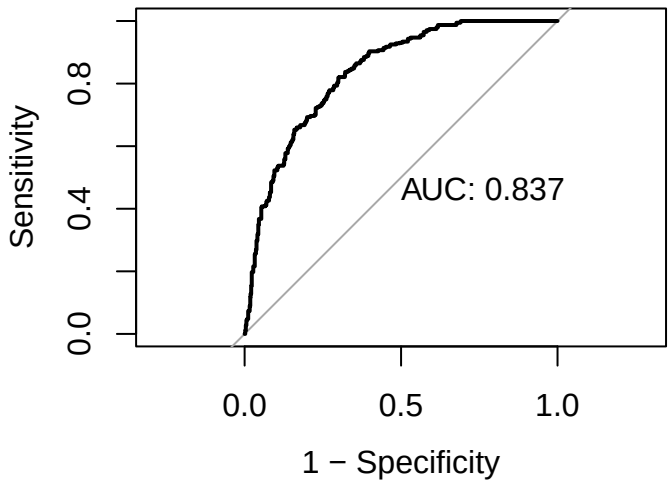
```
(accuracy.all <- round((sum(diag(table.all))/sum(table.all))*100, 3))
```

```
## [1] 73.984
```

```
roc.all <- roc(data.train$Outcome ~ pred.all, plot = TRUE,
               legacy.axes = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
#Assess test data accuracy

pred.all.test <- predict(all.logit, newdata = data.test, type = "response")
```

```
(table.all.test <- table(pred.all.test > 0.5, data.test$Outcome))
```

```
##
##           0   1
##   FALSE 198  28
##   TRUE   66 109
```

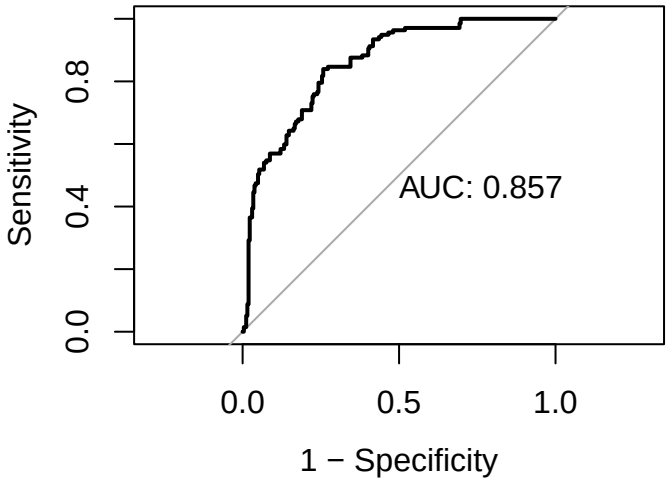
```
(accuracy.all.test <- round((sum(diag(table.all.test))/sum(table.all.test))*100, 3))
```

```
## [1] 76.559
```

```
roc.all.test <- roc(data.test$Outcome ~ pred.all.test, plot = TRUE,
                    legacy.axes = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




**Result** Over all, with all the test above, we can see that stepwise model perform with the best result, with about 79% precision score and 86% AUC score.

Here is the formula: Outcome ~ Glucose + Pregnancies + BMI + DiabetesPedigreeFunction + Age

### 5. Modeling - GLM with Probit

**What's the differences between GLM and GLM probit model?**

The difference between GLM and GLM with a probit link function lies primarily in the choice of the link function used within the GLM framework.

The interpretation and analysis of the probit link are similar to the GLM model.

```
full.probit <- glm(Outcome ~ . ,data = data.train ,
                   family = binomial(link = "probit"))
summary(full.probit)
```

**Modeling**

```
##
## Call:
## glm(formula = Outcome ~ ., family = binomial(link = "probit"),
##     data = data.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1480  -0.7306  -0.4122   0.7945   2.1782
##
```

```
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -0.537775   0.037663 -14.279  < 2e-16 ***
## Pregnancies                0.221363   0.042742   5.179 2.23e-07 ***
## Glucose                    0.639027   0.044196  14.459  < 2e-16 ***
## BloodPressure             -0.000253   0.040574  -0.006  0.99502
## SkinThickness              0.049599   0.044480   1.115  0.26481
## Insulin                   -0.034714   0.037897  -0.916  0.35966
## BMI                        0.241116   0.045790   5.266 1.40e-07 ***
## DiabetesPedigreeFunction   0.109130   0.038101   2.864  0.00418 **
## Age                        0.099016   0.044238   2.238  0.02521 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2054.4  on 1598  degrees of freedom
## Residual deviance: 1527.7  on 1590  degrees of freedom
## AIC: 1545.7
##
## Number of Fisher Scoring iterations: 5
```

With the same example variable *BloodPressure*, the estimated coefficient for *BloodPressure* is -0.000253 with a SE of 0.040574. This means that the decrease in expected log count $\log \lambda_i$ for one unit increase in *BloodPressure* is 0.000253. The p-value for *BloodPressure* is 0.99502, where we used to have alpha set as 0.05. The p-value for *BloodPressure* is greater than 0.05, hence we will say that it is not considered statistically significant. But *DiabetesPedigreeFunction* and *Age* both have a p-value that is less than 0.05, these are the only two variables considered as statistically significant.

The output of the model also shows values for null deviance and residual deviance.The residual deviance tells us how well the response variable can be predicted by a model with the intercept and predictor variables.

The model shows: Null deviance: 2054.4 on 1598 degrees of freedom Residual deviance: 1527.8 on 1590 degrees of freedom

```
null.probit <- glm(Outcome ~ 1, data = data.train,
                   family = binomial(link = "probit"))
summary(null.probit)
```

```
both.probit <- step(null.probit, list(lower = formula(null.probit),
                                       upper = formula(full.probit)),
                    direction = "both", trace = 0, data = data.train)
formula(both.probit)
```

```
## Outcome ~ Glucose + Pregnancies + BMI + DiabetesPedigreeFunction +
##     Age
```

```
summary(both.probit)
```

```
##
## Call:
## glm(formula = Outcome ~ Glucose + Pregnancies + BMI + DiabetesPedigreeFunction +
##     Age, family = binomial(link = "probit"), data = data.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2596  -0.7218  -0.4211   0.7916   2.1777
```

```
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -0.53616    0.03752 -14.291  < 2e-16 ***
## Glucose                    0.62648    0.04078  15.364  < 2e-16 ***
## Pregnancies                0.22160    0.04258   5.205 1.94e-07 ***
## BMI                        0.25838    0.03888   6.645 3.03e-11 ***
## DiabetesPedigreeFunction   0.10640    0.03786   2.810  0.00495 **
## Age                        0.10403    0.04259   2.443  0.01458 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2054.4  on 1598  degrees of freedom
## Residual deviance: 1529.9  on 1593  degrees of freedom
## AIC: 1541.9
##
## Number of Fisher Scoring iterations: 5
```

### Assessing accuracy

```
#Assess train data accuracy
pred.both.probit <- predict(both.probit, newdata = data.train, type = "response")
```

```
(table.both.probit <- table(pred.both.probit > 0.5, data.train$Outcome))
```

### Train data

```
##
##           0   1
##   FALSE 940 257
##   TRUE  112 290
```

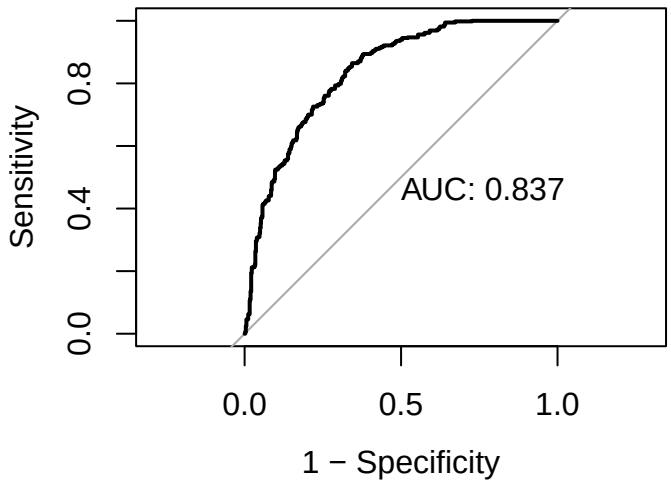
```
(accuracy.both.probit <- round((sum(diag(table.both.probit))/sum(table.both.probit))*100, 3))
```

```
## [1] 76.923
```

```
roc.both.probit <- roc(data.train$Outcome ~ pred.both.probit, plot = TRUE,
                legacy.axes = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
#Assess test data accuracy
pred.both.probit.test <- predict(both.probit, newdata = data.test, type = "response")
```

```
(table.both.probit.test <- table(pred.both.probit.test > 0.5, data.test$Outcome))
```

**Test data**

```
##
##           0   1
##   FALSE 239  60
##   TRUE   25  77
```

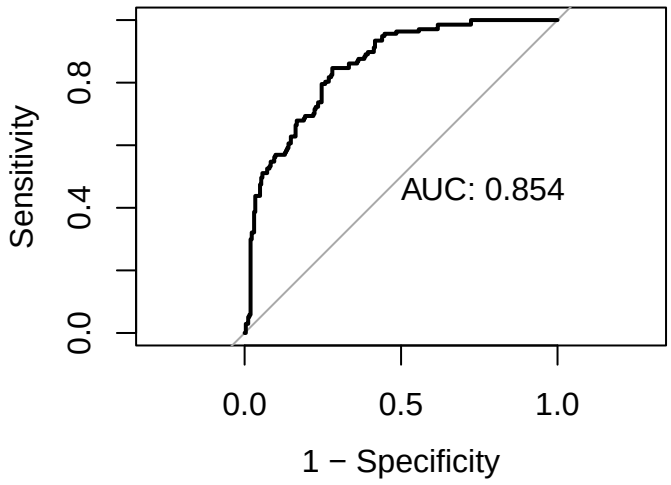
```
(accuracy.both.probit.test <- round((sum(diag(table.both.probit.test))/sum(table.both.probit.test))*100
```

```
## [1] 78.803
```

```
roc.both.probit.test <- roc(data.test$Outcome ~ pred.both.probit.test, plot = TRUE,
                legacy.axes = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```



**Result** With all tests above, the performance of GLM probit model after stepwise selection is very similar to GLM logit model, with both 78% precision score and slightly lower 85% AUC score on test data. Both GLM model, with the probit and logit link function, did not overfitting the train data which is a good sign.

### 6. Modeling - Random Forest

#### What is random forest model?

Random forest is a a powerful machine learning technique that combines the predictions from multiple machine learning algorithms to make more accurate predictions than any individual model.

```
# Train the Random Forest model with selected features
rf_model <- randomForest(Outcome ~ Glucose + Pregnancies + BMI + DiabetesPedigreeFunction + Age,
                         data = data.train,
                         ntree = 500, # Number of trees to grow. This is a default value.
                         importance = TRUE) # Calculate variable importance
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values.  Are you sure you want to do regression?
```

```
# Summarize the model
print(rf_model)
```

```
##
## Call:
##  randomForest(formula = Outcome ~ Glucose + Pregnancies + BMI +      DiabetesPedigreeFunction + Age,
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 0.04267357
##                     % Var explained: 81.04
```

#### Assessing accuracy

```
#Assess train data accuracy
pred.rf <- predict(rf_model, newdata = data.train, type = "response")
```

```
(table.rf <- table(pred.rf > 0.5, data.train$Outcome))
```

#### Train data

```
##
##            0    1
##   FALSE 1052    2
##   TRUE     0  545
```

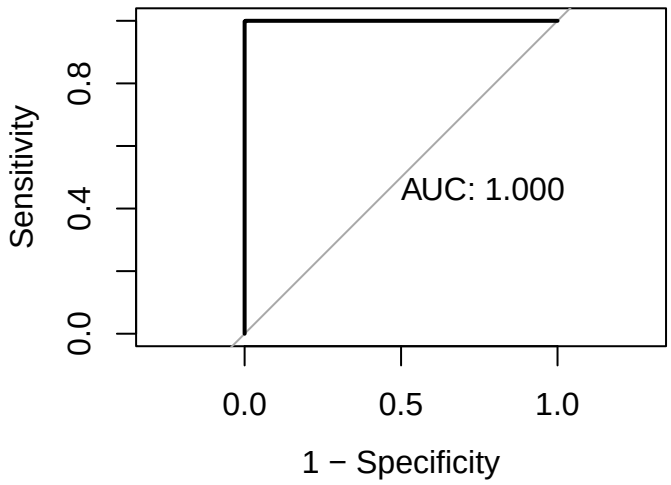
```
(accuracy.rf <- round((sum(diag(table.rf))/sum(table.rf))*100, 3))
```

```
## [1] 99.875
```

```
roc.rf <- roc(data.train$Outcome ~ pred.rf, plot = TRUE,
              legacy.axes = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
#Assess test data accuracy
pred.rf.test <- predict(rf_model, newdata = data.test, type = "response")
```

```
(table.rf.test <- table(pred.rf.test > 0.5, data.test$Outcome))
```

**Test data**

```
##
##           0   1
##   FALSE 262   6
##   TRUE    2 131
```

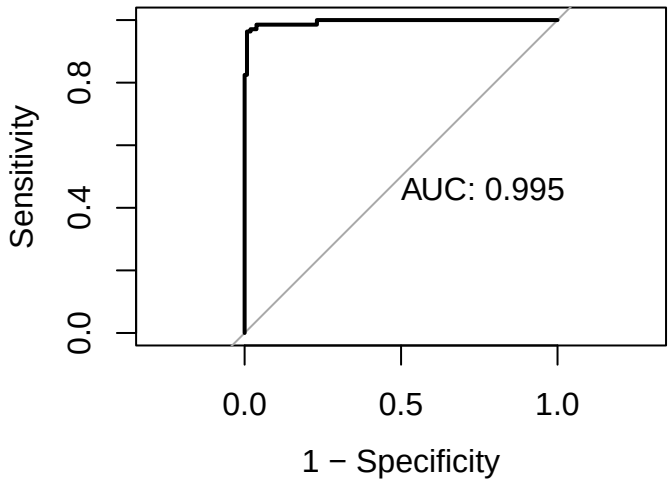
```
(accuracy.rf.test <- round((sum(diag(table.rf.test))/sum(table.rf.test))*100, 3))
```

```
## [1] 98.005
```

```
roc.rf.test <- roc(data.test$Outcome ~ pred.rf.test, plot = TRUE,
                   legacy.axes = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```



**Result** The random forest model with feature selected by stepwise have the best performance. with 98% accuracy on test data, not overfitting at all. Thus in conclusion, random forest can be considered as a desired model for predicting diabetes outcomes in our dataset.

## Conclusion

This comprehensive analysis of the Pima Indian Diabetes dataset has provided valuable insights into the factors influencing diabetes and the effectiveness of various machine learning models in predicting diabetes outcomes. Through proper data preprocessing, including handling of biologically implausible zero values and median imputation, we ensured the integrity and reliability of our dataset for analysis.

The EDA revealed key insights, such as the significant impact of glucose levels, BMI, and age on diabetes outcomes, highlighting their potential as critical predictors in diabetes risk assessment. Our histogram and scatter plot analyses underscored the relationships between these variables and diabetes, reinforcing their importance in predictive modeling.

The application of Generalized Linear Models (GLM) with both logit and probit links provided a robust statistical framework for understanding the influence of individual predictors. The models, especially after stepwise selection, showed notable inprove in predictive power, with the logit model slightly outperforming the probit model in terms of precision and AUC scores.

However, the most significant finding of our study was the superior performance of the Random Forest model. The model, with features selected through stepwise logistic regression, demonstrated remarkable accuracy, outperforming the GLM models in both training and testing phases. Normally, random forest would have high risk of over-fitting, But With an impressive 98% accuracy on the test data, it stood out as the most effective model for predicting diabetes outcomes in the Pima Indian dataset.

In conclusion, our analysis highlights the critical factors associated with diabetes and showcases the potential of machine learning in enhancing the predictive capability of diabetes outcomes. The Random Forest model, with its high accuracy given a mostly Balanced dataset. For future work would include test the model with more data, fine tune the hyperparameter, gather more data metrics, and use more evaluation metrics like F-1 score for model selection.

## Reference

[1]Akimel O'odham From Wikipedia, the free encyclopedia, last edited on 5 December 2023, at 16:16 (UTC).

https://en.wikipedia.org/wiki/Akimel_O%27odham

[2]What Is Diabetes? Last Reviewed April 2023, National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK)

https://www.niddk.nih.gov/health-information/diabetes/overview/what-is-diabetes?utm_source=copy&utm_medium=sharing+button&utm_content=%2Fhealth-information%2Fdiabetes%2Foverview%2Fwhat-is-diabetes