Physiological Characteristics of T2DM in Pima Indians

Nirupa Gadi

June/2020

Purpose

The purpose of this study is to develop a regression model that can predict incidence of Type II Diabetes Mellitus (T2DM) in a population of Pima Indians. Four different regression approaches will be attempted to yield an optimised accuracy with sensitivity and specificity measures.

Background

Over thousands of epidemiological studies, researchers and healthcare providers have gathered tremendous information on the healthcare disparities that affect Native Americans. Previous research has shown that this population unfortunately suffers from excessive burden of disease and lower life expectancies. These occurences are perhaps due to high rates of poverty, systemic racism, and many other forms of health discrimination based on factors of intersectionality.

Sources have shown that Native Americans can suffer up to three times the rate of Type II Diabetes Mellitus (T2DM) compared to Caucasian and coloured Americans. This high incidence of disease can be problematic, as proper management of T2DM relies on education and strict self-management. If poorly managed, T2DM may lead to many other adverse health outcomes such as severe heart disease or COVID-19 mortality.

To limit the rates of these diseases, prevention on a primary and secondary level is preferred to life-long treatment. In order to do this, healthcare providers must have robust methods of predicting which patients would likely have the disease. To anticipate risk of diabetes, development of a robust model of regression is necessary.

This study is focused on analysis of the `Diabetes` dataset downloaded to a personal computer from Kaggle at: https://www.kaggle.com/uciml/pima-indians-diabetes-database (https://www.kaggle.com/uciml/pima-indians-diabetes-database) The dataset `Diabetes` relates frequencies and statistics of physiological measurements on women over the age of 21 belonging to the Pima Native American tribe residing of Arizona.

Exploratory Data Analysis

Setup

Below is information to detail the operating system.

```
#Check computer details#
version
```

```
##                _
## platform       x86_64-apple-darwin17.0
## arch           x86_64
## os             darwin17.0
## system         x86_64, darwin17.0
## status
## major          4
## minor          0.0
## year           2020
## month          04
## day            24
## svn rev        78286
## language       R
## version.string R version 4.0.0 (2020-04-24)
## nickname       Arbor Day
```

The dataset has been downloaded from Kaggle and stored on a personal computer, where it will be pushed into R.

```
#Download the dataset from an excel sheet on computer#
if(!require(readxl)) install.packages("readxl")
```

```
## Loading required package: readxl
```

```
library(readxl)
file.exists("/Users/nerp/Desktop/diabetes.xlsx")
```

```
## [1] TRUE
```

```
pima<- read_xlsx("/Users/nerp/Desktop/diabetes.xlsx")
```

After upload, the initial structure of the `pima` dataset should be studied.

```
#Check the first 6 rows of the provided dataset#
head(pima)
```

```
## # A tibble: 6 x 9
##   Pregnancies Glucose BloodPressure SkinThickness Insulin   BMI DiabetesPedigre…
##         <dbl>   <dbl>         <dbl>         <dbl>   <dbl> <dbl>            <dbl>
## 1           6     148            72            35       0  33.6            0.627
## 2           1      85            66            29       0  26.6            0.351
## 3           8     183            64             0       0  23.3            0.672
## 4           1      89            66            23      94  28.1            0.167
## 5           0     137            40            35     168  43.1            2.29
## 6           5     116            74             0       0  25.6            0.201
## # … with 2 more variables: Age <dbl>, Outcome <dbl>
```

`pima` appears to be in tidy format, meaning that each variable forms a column, each observation is a row, and the observational unit forms a table. Next, the parameters within the dataset will be defined.

```
#See the overall structure of the dataset#
str(pima)
```

```
## tibble [768 × 9] (S3: tbl_df/tbl/data.frame)
##  $ Pregnancies             : num [1:768] 6 1 8 1 0 5 3 10 2 8 ...
##  $ Glucose                 : num [1:768] 148 85 183 89 137 116 78 115 197 125 ...
##  $ BloodPressure           : num [1:768] 72 66 64 66 40 74 50 0 70 96 ...
##  $ SkinThickness           : num [1:768] 35 29 0 23 35 0 32 0 45 0 ...
##  $ Insulin                 : num [1:768] 0 0 0 94 168 0 88 0 543 0 ...
##  $ BMI                     : num [1:768] 33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.
5 0 ...
##  $ DiabetesPedigreeFunction: num [1:768] 0.627 0.351 0.672 0.167 2.288 ...
##  $ Age                     : num [1:768] 50 31 32 21 33 30 26 29 53 54 ...
##  $ Outcome                 : num [1:768] 1 0 1 0 1 0 1 0 1 1 ...
```

It appears that `pima` has 768 observations of 9 variables, all of which are in numeric format. `Pregnancies` describes the gravidity of each patient in the dataset `Glucose` describes the mg of glucose per every dL of blood in the patient `BloodPressure` describes the diastolic blood pressure of each patient `SkinThickness` measures the epidermal, dermal, and subcutaneous layers of brachial skin for each patient `Insulin` describes the mIU of insulin protein per litre of blood after 2 hours of fasting `BMI` describes the standardised body mass index of each patient. `DiabetesPedigreeFunction` describes the result of an unlisted function that calculates the genetic influence of T2DM in each patient. `Age` describes the rounded age in years of each patient `Outcome` denotes `0` for non-diabetic and `1` for diabetic for each patient

Next, missing values are quite common in real-life datasets, so it is imperative to inspect the set for this.

```
#Find the proportion of the dataset that is NA#
sum(is.na(pima))
```

```
## [1] 0
```

```
sum(is.na(pima))/(ncol(pima)*nrow(pima))
```

```
## [1] 0
```

There are no missing values in the `pima` dataset, which makes up `0` % of the total set. This is very impressive for clinical data analysis, as medical records tend to have missing data.

Now that the data has been cleaned up, it is ready for some visualisation analysis.

Visualisation

Visualisation is a key step before modelling. Generating plots enables the data scientist to have an overall understand of trends in the data, faciliating the analysis.

```
#Download commonly needed packages for visualisation#
if (!require(plyr)) install.packages("plyr")
```

```
## Loading required package: plyr
```

```
if (!require(ggrepel)) install.packages("ggrepel")
```

```
## Loading required package: ggrepel
```

```
## Loading required package: ggplot2
```

```
if (!require(gridExtra)) install.packages("gridExtra")
```

```
## Loading required package: gridExtra
```

```
library(plyr)
library(dplyr)
```

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

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

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

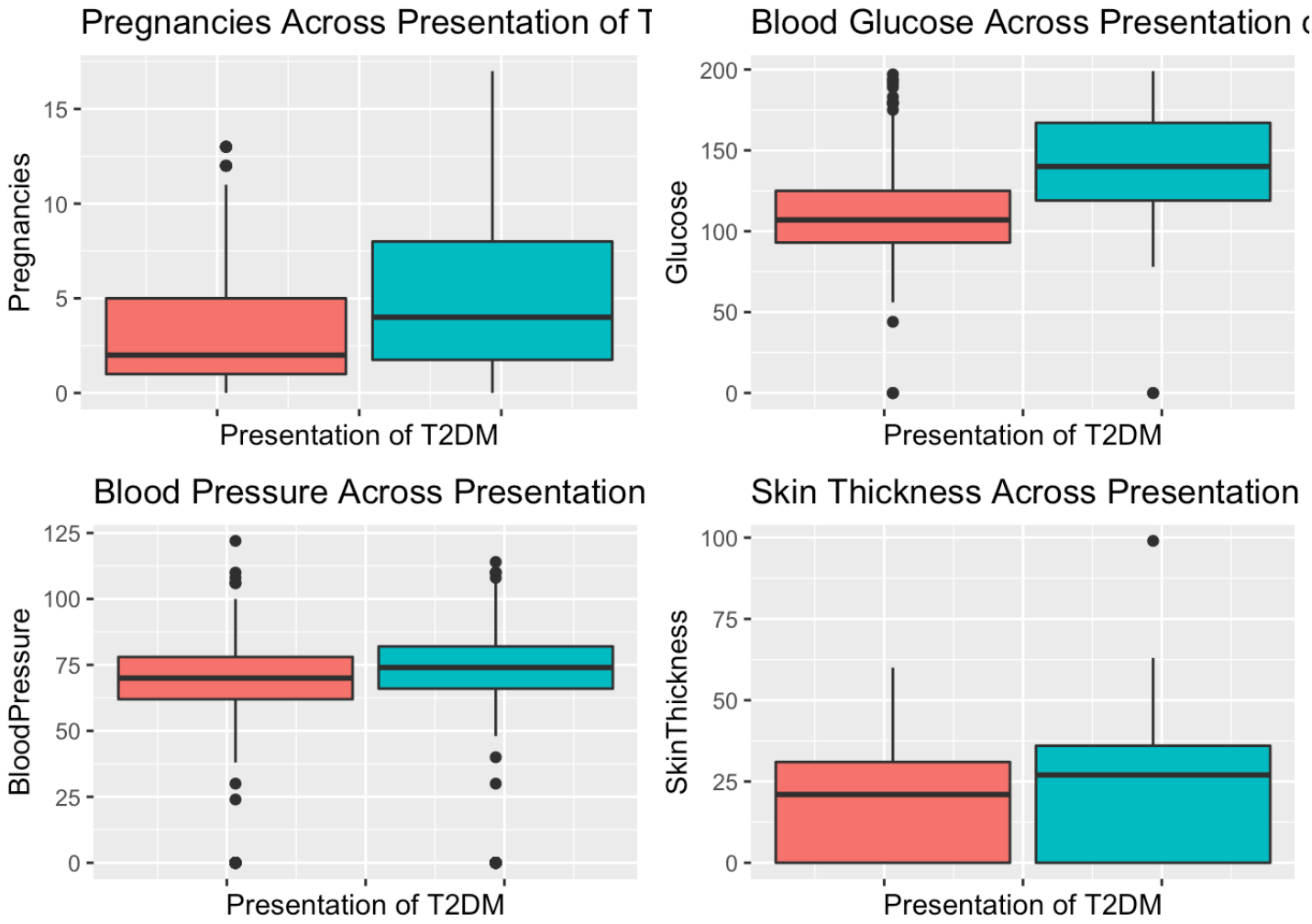
```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
library(gridExtra)
```

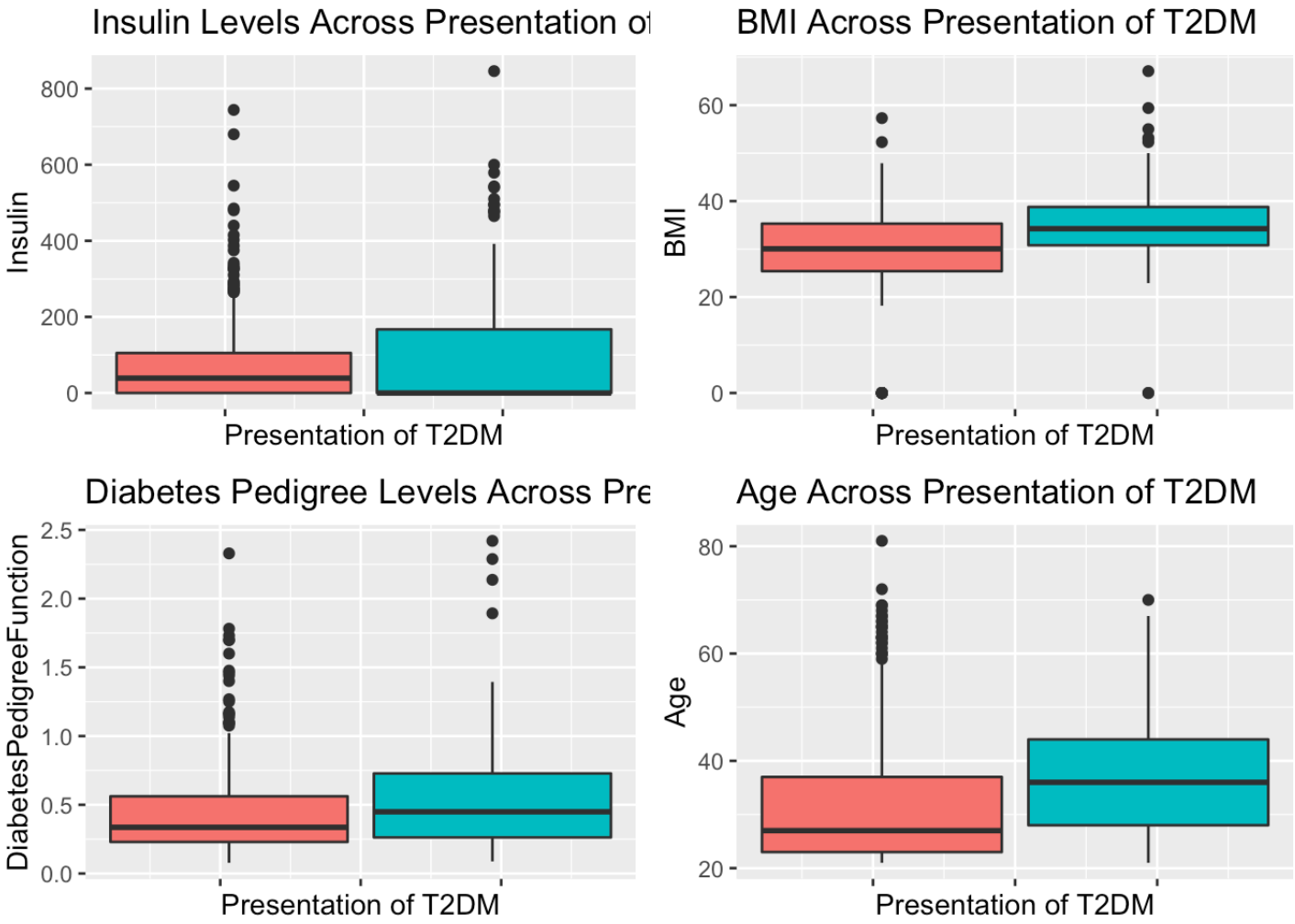
There are 8 parameters that could be possibly correlated with T2DM in the dataset. In this section, each prospective parameter will be stratified by T2DM status to observe differences in the distribution.

```
#Convert outcome to binary factor class#
pima$Outcome <- as.factor(pima$Outcome)

#Box and whisker plot for pregnancies stratified by T2DM status#
p1<- pima %>% group_by(Outcome) %>% ggplot(aes(y=Pregnancies)) + geom_boxplot(aes(fil
l=Outcome)) + labs(title="Pregnancies Across Presentation of T2DM", xlab= "Presentati
on of T2DM", ylab = "Pregnancies") + theme(axis.text.x=element_blank()) + theme(legen
d.position = "none")

#Box and whisker plot for blood glucose stratified by T2DM status#
p2<- pima %>% group_by(Outcome) %>% ggplot(aes(y=Glucose)) + geom_boxplot(aes(fill=Ou
tcome)) + labs(title="Blood Glucose Across Presentation of T2DM", xlab= "Presentation
of T2DM", ylab = "Blood Glucose") + theme(axis.text.x=element_blank()) + theme(legend
.position = "none")

#Box and whisker plot for BP stratified by T2DM status#
p3<- pima %>% group_by(Outcome) %>% ggplot(aes(y=BloodPressure)) + geom_boxplot(aes(f
ill=Outcome)) + labs(title="Blood Pressure Across Presentation of T2DM", xlab= "Prese
ntation of T2DM", ylab = "Blood Pressure") + theme(axis.text.x=element_blank()) + the
me(legend.position = "none")

#Box and whisker plot for skin thickness stratified by T2DM status#
p4<- pima %>% group_by(Outcome) %>% ggplot(aes(y=SkinThickness)) + geom_boxplot(aes(f
ill=Outcome)) + labs(title="Skin Thickness Across Presentation of T2DM", xlab= "Prese
ntation of T2DM", ylab = "Skin Thickness") + theme(axis.text.x=element_blank()) + the
me(legend.position = "none")

#arrange the plots in a 2x2#
grid.arrange(p1,p2,p3,p4,ncol=2)
```



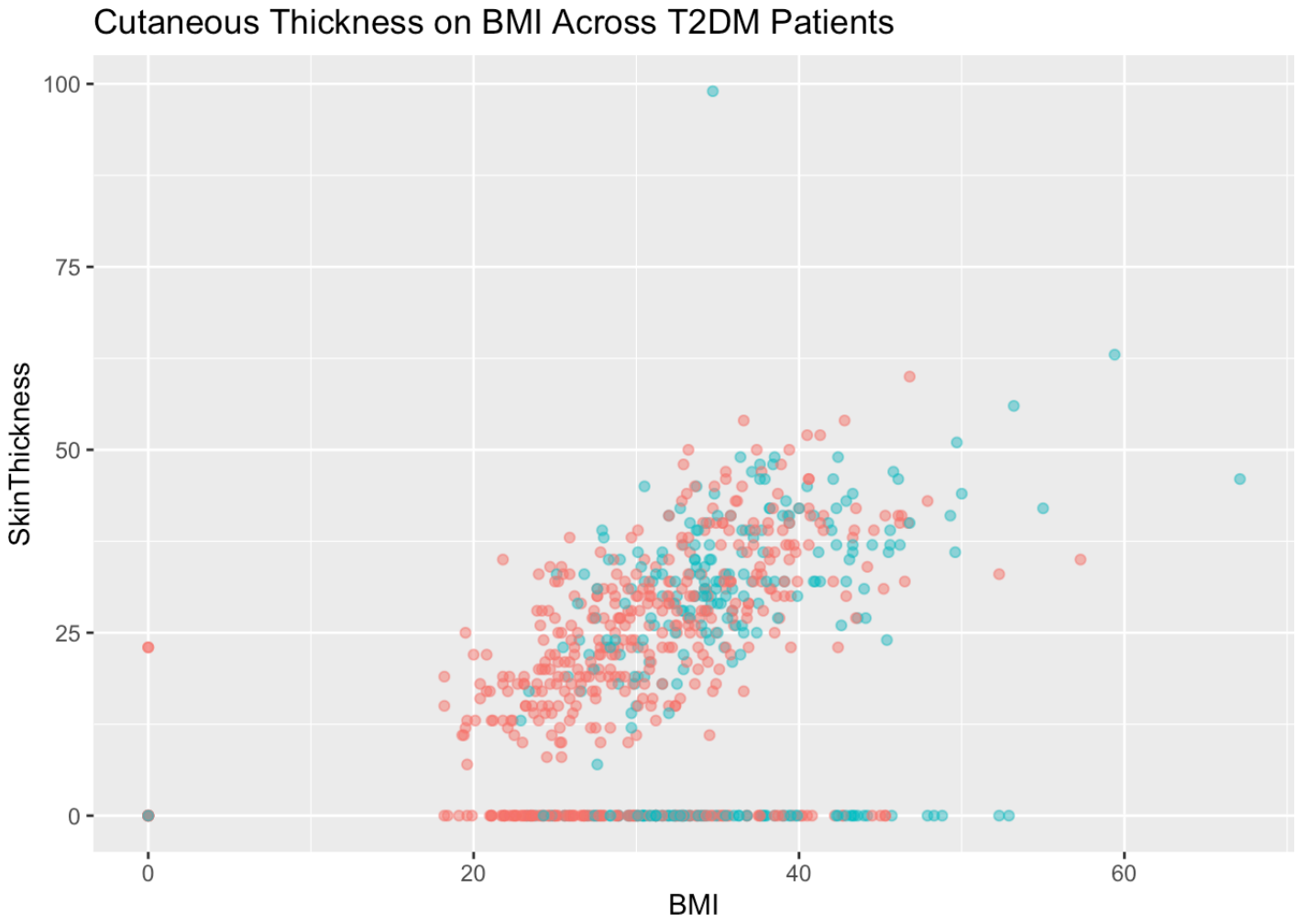
From looking at the distribution of these parameters stratified by T2DM status, it appears that women who have had more pregnancies have a higher mean incidence of T2DM compared to women with fewer pregnancies. Additionally, the mean plasma glucose levels in those with T2DM is higher than in healthy patients. These two parameters may prove to be useful predictors in the logistic regression. Blood pressure and skin thickness have also been plotted as box and whisker plots, but these don't appear to have a distinct difference across T2DM status.

```
#Box and whisker plot for insulin levels stratified by T2DM status#
p5<- pima %>% group_by(Outcome) %>% ggplot(aes(y=Insulin)) + geom_boxplot(aes(fill=Ou
tcome)) + labs(title="Insulin Levels Across Presentation of T2DM", xlab= "Presentatio
n of T2DM", ylab = "Insulin Levels") + theme(axis.text.x=element_blank()) + theme(leg
end.position = "none")

#Box and whisker plot for BMI stratified by T2DM status#
p6<- pima %>% group_by(Outcome) %>% ggplot(aes(y=BMI)) + geom_boxplot(aes(fill=Outcom
e)) + labs(title="BMI Across Presentation of T2DM", xlab= "Presentation of T2DM", yla
b = "BMI") + theme(axis.text.x=element_blank()) + theme(legend.position = "none")

#Box and whisker plot for pedigree levels stratified by T2DM status#
p7<- pima %>% group_by(Outcome) %>% ggplot(aes(y=DiabetesPedigreeFunction)) + geom_bo
xplot(aes(fill=Outcome)) + labs(title="Diabetes Pedigree Levels Across Presentation o
f T2DM", xlab= "Presentation of T2DM", ylab = "Pedigree Levels") + theme(axis.text.x=
element_blank()) + theme(legend.position = "none")

#Box and whisker plot for ages stratified by T2DM status#
p8<- pima %>% group_by(Outcome) %>% ggplot(aes(y=Age)) + geom_boxplot(aes(fill=Outcom
e)) + labs(title="Age Across Presentation of T2DM", xlab= "Presentation of T2DM", yla
b = "Age") + theme(axis.text.x=element_blank()) + theme(legend.position = "none")

#arrange the plots in a 2x2#
grid.arrange(p5,p6,p7,p8,ncol=2)
```



The latter set of parameters appear to have some differences as well. BMI and and age more clearly have a relationship that can be seen across the diabetes strata, but insulin levels and diabetes pedigree appear to be obsure. Ultimately, all of the parameters analysed will have to initially be put into a logistic regression to see which variables are statistically significant in their relationship with T2DM status. Not only should the variables be stratified by the outcome of interest, but there is high potential for confounding variables in this dataset. For example, the parameter `SkinThickness` is a measure of fat accumulation while BMI is also a common measure for load of the body. If these two variables have a high absolute correlation (>0.5), then they should be excluded from the logistic analysis.

```
#scatterplot for Skin Thickness on BMI across T2DM status#
pima %>% group_by(Outcome) %>% ggplot() + geom_point(aes(x=BMI, y=SkinThickness, colo
ur=Outcome), alpha=0.5) + theme(legend.position = "none") + ggtitle("Cutaneous Thickn
ess on BMI Across T2DM Patients")
```

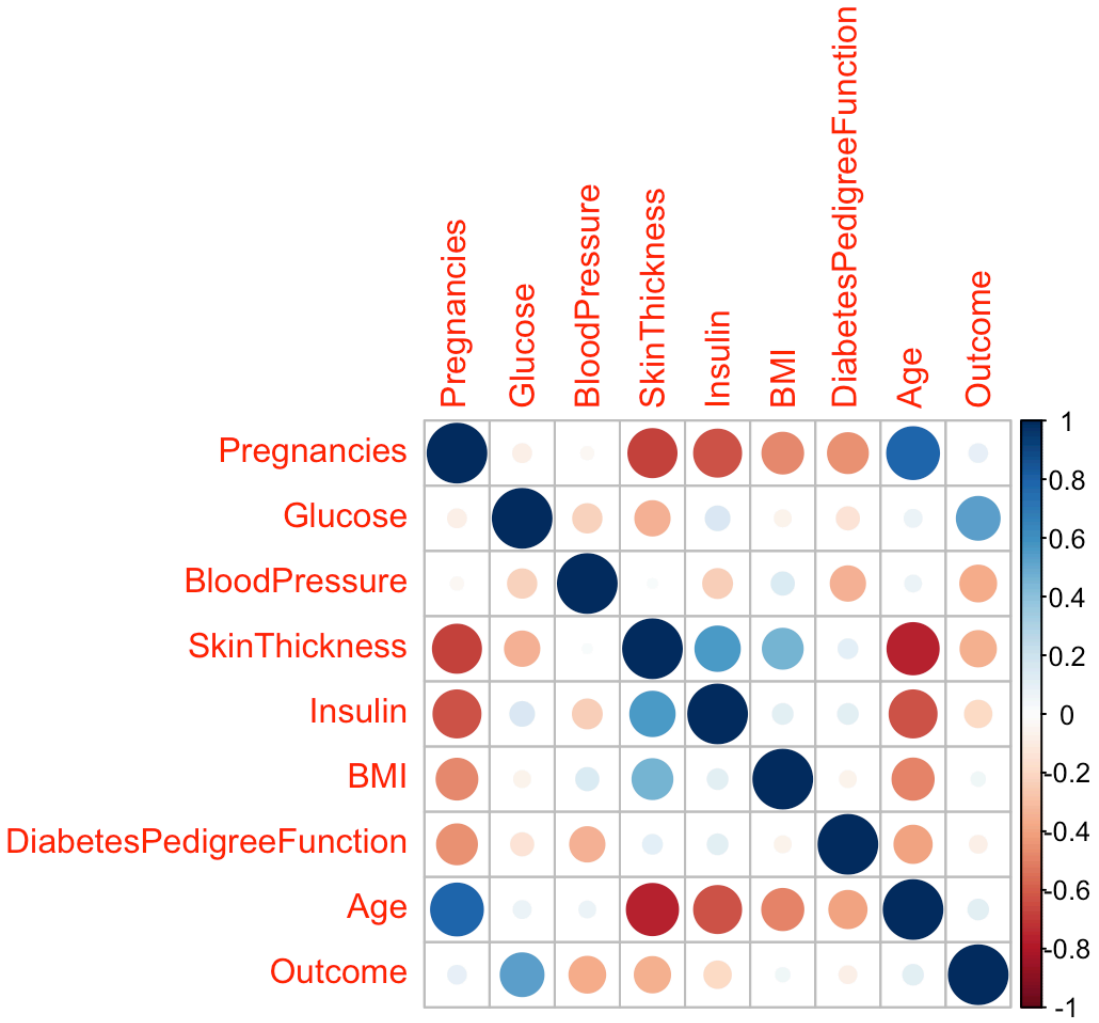



```
#correlation plot between all variables in `pima`"
if(!require(corrplot)) install.packages("corrplot")
```

```
## Loading required package: corrplot
```

```
## corrplot 0.84 loaded
```

```
library(corrplot)
par( mfrow = c(1,1) )
pima$Outcome<- as.numeric(pima$Outcome)
cor1 <- cor(pima, method = c("pearson"))
cor2 <- round(cor(cor1),2)
corrplot::corrplot(cor2, method = "circle")
```



It appears that although the variables do have a correlation, none of them have an absolute value above 0.5, making all of them suitable for incorporation into the logistic regression.

Modeling

The data should now be split into training and test sets. The training set will be used to develop the regressions, and the test set will be to evaluate the accuracy of each regression. This will avoid overfitting the model to a dataset. The training and test sets will be split into an 80:20 divide, as the dataset has less than 800 observations. This 80:20 split will create a strong balance between parameter estimates and performance statistics.

```
if(!require(caret)) install.packages("caret")
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
library(caret)

#Attempt to convert the variables to class of factor#
pima[sapply(pima, is.character)] <- lapply(pima[sapply(pima, is.character)], as.facto
r)

#Split both datasets into an 80:20 split using a seed#
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
sample_size<- floor(0.2*nrow(pima))
test_index <- sample(seq_len(nrow(pima)), size=sample_size)
pima_train<- pima[-test_index,]
pima_test<- pima[test_index,]
```

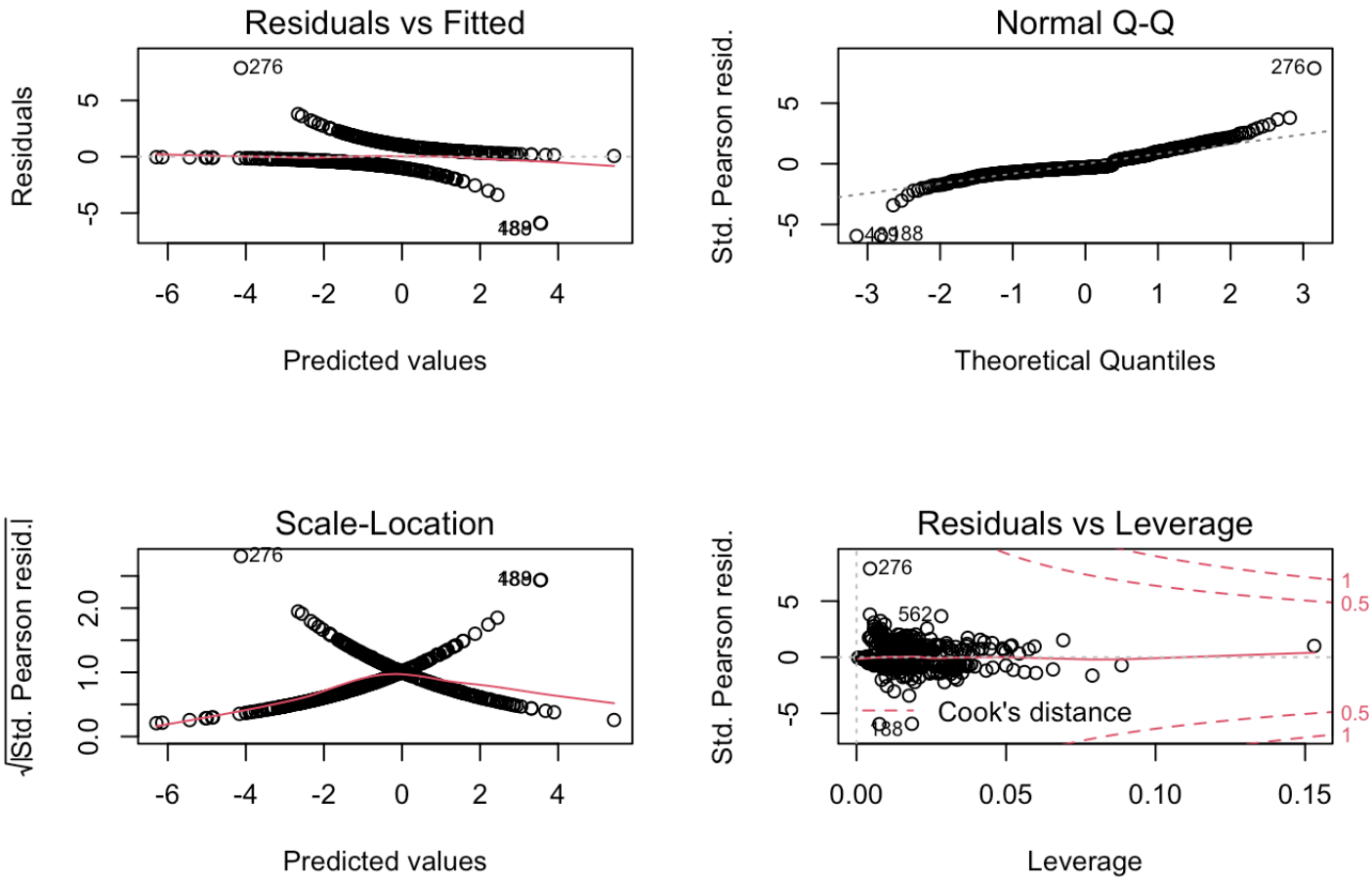
General Logistic Regression

Logistic regressions are models similar to linear regressions, but they are specialised for discrete rather than continuous outcomes. This makes them a perfect starting point for this study of classification algorithms, namely the binomial outcomes “Diabetic” or “non-Diabetic.”

```
#Set the plots to a 2x2#
par(mfrow = c(2,2))
#Calculate the deviance residuals, coefficients, and significances#
pima_train$Outcome<- as.factor(pima_train$Outcome)
reg_glm <- glm(Outcome ~ ., data = pima_train, family = binomial(link = "logit"))
summary(reg_glm)
```

```
##
## Call:
## glm(formula = Outcome ~ ., family = binomial(link = "logit"),
##     data = pima_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6787  -0.7260  -0.4227   0.7112   2.8785
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -8.6134123  0.8176411 -10.534  < 2e-16 ***
## Pregnancies              0.1501737  0.0354522   4.236 2.28e-05 ***
## Glucose                  0.0347289  0.0042125   8.244  < 2e-16 ***
## BloodPressure           -0.0107631  0.0059293  -1.815 0.069489 .
## SkinThickness            0.0003224  0.0076690   0.042 0.966468
## Insulin                 -0.0013166  0.0010076  -1.307 0.191297
## BMI                      0.0945281  0.0172164   5.491 4.01e-08 ***
## DiabetesPedigreeFunction  1.1605340  0.3428217   3.385 0.000711 ***
## Age                      0.0083561  0.0103591   0.807 0.419871
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 797.28  on 614  degrees of freedom
## Residual deviance: 577.59  on 606  degrees of freedom
## AIC: 595.59
##
## Number of Fisher Scoring iterations: 5
```

```
plot(reg_glm)
```

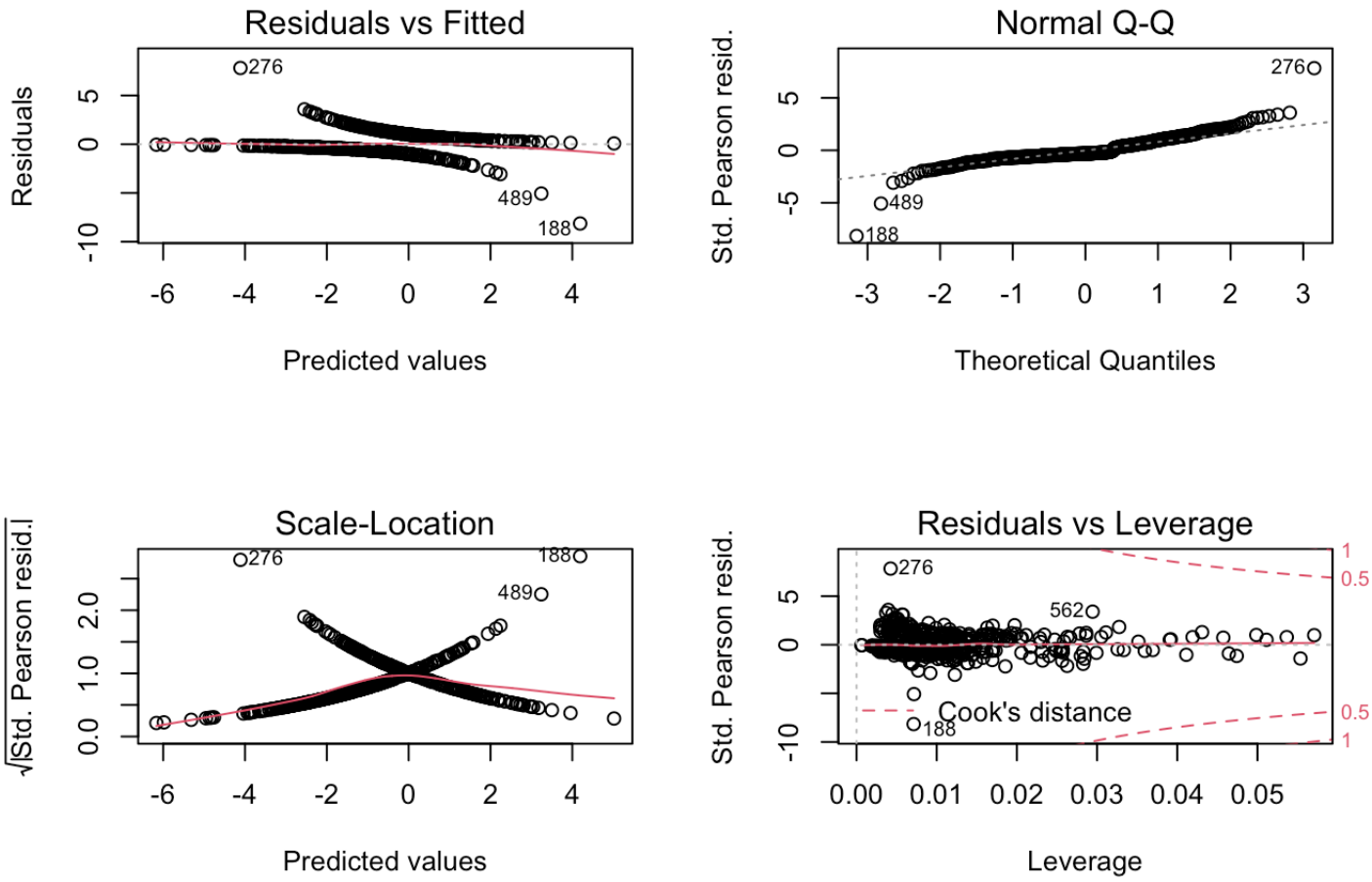


Another reason why the general logistic regression is much appreciated is because it shows which variables are significant in their relationship to T2DM outcome. From the table and plots above, it appears that regression intercept, `Pregnancies`, `Glucose`, `BMI`, and `DiabetesPedigreeFunction` are statistically significant to an alpha of almost 0. The variable `BloodPressure` is significant at the alpha=0.05. All of these parameters will be selected to continue to the optimised regression.

```
#Optimised logistic regression with significant parameters#
par(mfrow = c(2,2))
reg_glm2 <- glm(Outcome~ Pregnancies+Glucose+BloodPressure+BMI + DiabetesPedigreeFunc
tion, data=pima_train, family=binomial(link= "logit"))
summary(reg_glm2)
```

```
##
## Call:
## glm(formula = Outcome ~ Pregnancies + Glucose + BloodPressure +
##     BMI + DiabetesPedigreeFunction, family = binomial(link = "logit"),
##     data = pima_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9005  -0.7323  -0.4186   0.7021   2.8735
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -8.24126    0.77246 -10.669  < 2e-16 ***
## Pregnancies                0.16883    0.03088   5.467 4.59e-08 ***
## Glucose                    0.03348    0.00383   8.742  < 2e-16 ***
## BloodPressure             -0.01004    0.00575  -1.747  0.08068 .
## BMI                        0.09050    0.01611   5.617 1.95e-08 ***
## DiabetesPedigreeFunction   1.09233    0.33688   3.242  0.00119 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 797.28  on 614  degrees of freedom
## Residual deviance: 580.52  on 609  degrees of freedom
## AIC: 592.52
##
## Number of Fisher Scoring iterations: 5
```

```
plot(reg_glm2)
```



In order to predict a binary outcome, bounds must be set. The regression continues in order to create a confusion matrix.

```
pima_test$Outcome<- as.factor(pima_test$Outcome)
pred_glm <- predict(reg_glm2,pima_test, type = "response")
pred_glm <- ifelse(pred_glm <= 0.5, 1, 2)
pred_glm<- as.factor(pred_glm)
cm_glm<- confusionMatrix(pred_glm,pima_test$Outcome)
cm_glm
```

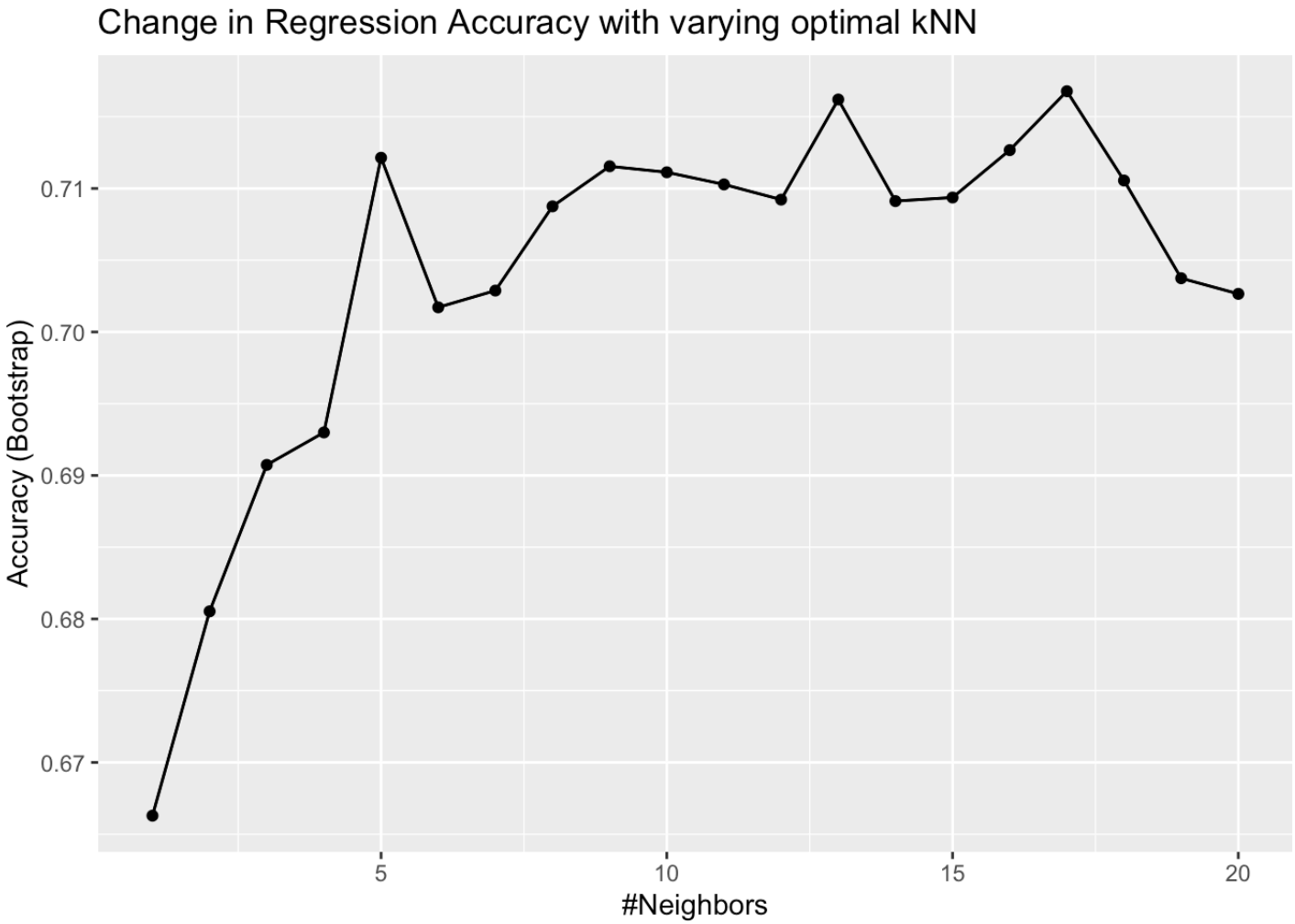
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 88 16
##          2 13 36
##
##                Accuracy : 0.8105
##                  95% CI : (0.7393, 0.8692)
##     No Information Rate : 0.6601
##     P-Value [Acc > NIR] : 2.948e-05
##
##                   Kappa : 0.5716
##
##  Mcnemar's Test P-Value : 0.7103
##
##             Sensitivity : 0.8713
##             Specificity : 0.6923
##          Pos Pred Value : 0.8462
##          Neg Pred Value : 0.7347
##              Prevalence : 0.6601
##          Detection Rate : 0.5752
##    Detection Prevalence : 0.6797
##       Balanced Accuracy : 0.7818
##
##        'Positive' Class : 1
##
```

The confusion matrix generated with the general logistic regression has a balanced accuracy of 0.7818, which is actually quite impressive for an initial trial. The sensitivity of the model is high at 0.8713. However, the specificity leaves room to be desired at 0.6923.

k-Nearest Neighbours Regression

The next model to be attempted will be the k-Nearest Neighbours algorithm. This model operates on the principle that test datapoints are similar to their “neighbouring” datapoints. These `k` values are used to develop the prediction. Compared to the general logistic regression, the kNN approach is a non-parametric model that is more supportive of non-linear models.

```
#Develop a model to test the accuracy on the number of neighbours#
reg_knn <- train(Outcome ~ ., data= pima_test, method = "knn", tuneGrid = data.frame(
k = seq(1,20,1)))
reg_knn %>% ggplot()+geom_line(aes(x=k, y=Accuracy)) + labs(title= "Change in Regress
ion Accuracy with varying optimal kNN")
```



According to the table and graph, the optimal number of neighbours for the model to search for is `k` =15. This value will be used in the prediction.

```
pred_knn <- predict(reg_knn, pima_test, type="raw")
cm_knn<- confusionMatrix(pred_knn,pima_test$Outcome)
cm_knn
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 93 26
##          2  8 26
##
##                Accuracy : 0.7778
##                  95% CI : (0.7036, 0.8409)
##     No Information Rate : 0.6601
##     P-Value [Acc > NIR] : 0.001032
##
##                   Kappa : 0.4594
##
##  Mcnemar's Test P-Value : 0.003551
##
##             Sensitivity : 0.9208
##             Specificity : 0.5000
##          Pos Pred Value : 0.7815
##          Neg Pred Value : 0.7647
##              Prevalence : 0.6601
##          Detection Rate : 0.6078
##    Detection Prevalence : 0.7778
##       Balanced Accuracy : 0.7104
##
##        'Positive' Class : 1
##
```

The kNN regression is shown to actually have a lower balanced accuracy compared to the generalised model, at 0.7247. Its sensitivity is slightly improved at 0.9109. However, the specificity is quite low at 0.5285.

Random Forest Regression

This type of regression is a little more advanced, based on the concept of creating several “decision trees”. This involves a branching equation to derive the output, making random forest models theoretically quite accurate, but very time-intensive.

```
#Develop a random forest regression model to produce a confusion matrix#
reg_rf <- train(Outcome ~ ., method = "rf", data = pima_train)
pred_rf <- predict(reg_rf, pima_test)
pima_test$Outcome<- as.factor(pima_test$Outcome)
cm_rf<- confusionMatrix(pred_rf,pima_test$Outcome)
cm_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 82 18
##          2 19 34
##
##                Accuracy : 0.7582
##                  95% CI : (0.6824, 0.8237)
##     No Information Rate : 0.6601
##     P-Value [Acc > NIR] : 0.005655
##
##                   Kappa : 0.4636
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.8119
##             Specificity : 0.6538
##          Pos Pred Value : 0.8200
##          Neg Pred Value : 0.6415
##              Prevalence : 0.6601
##          Detection Rate : 0.5359
##    Detection Prevalence : 0.6536
##       Balanced Accuracy : 0.7329
##
##        'Positive' Class : 1
##
```

The random forest model shows a similar conclusion to the kNN regression. The balanced accuracy is 0.7183, the sensitivity is 0.8020, and the specificity is low at 0.6346. Its values, along with the kNN values, are inferior to the initial logistic regression.

XGBoost Regression

Finally, the novel eXtreme Gradient Boosting software algorithm will be analysed. This machine-learning approach has gained several awards and traction on the web, and is the algorithm of choice for data science competitions on Kaggle. Its parallel tree boosting method is incredibly flexible and more time-efficient than the Random Forest algorithm.

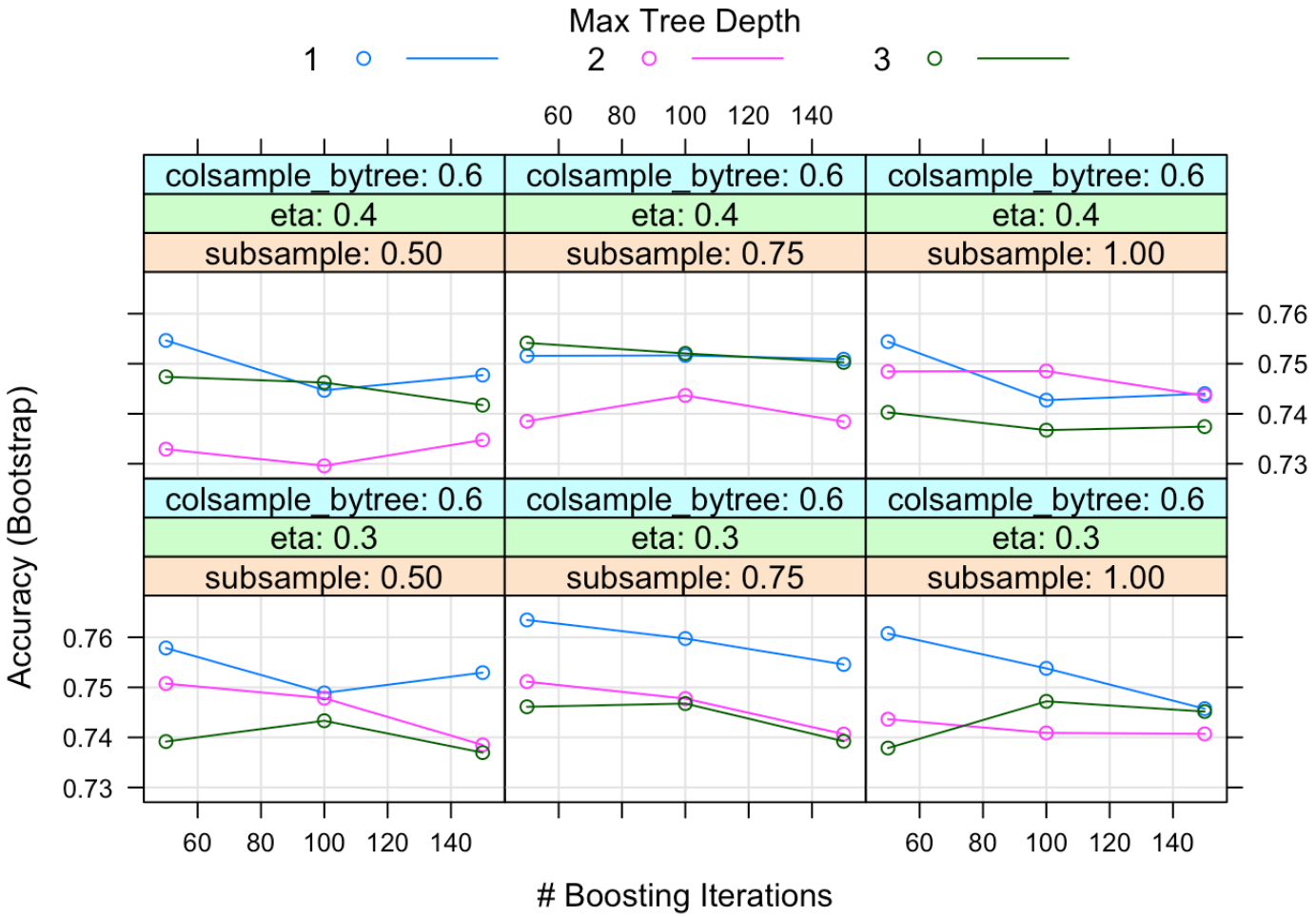
```
# Develop an XGB regression model to produce a confusion matrix#
if(!require(xgboost)) install.packages("xgboost")
```

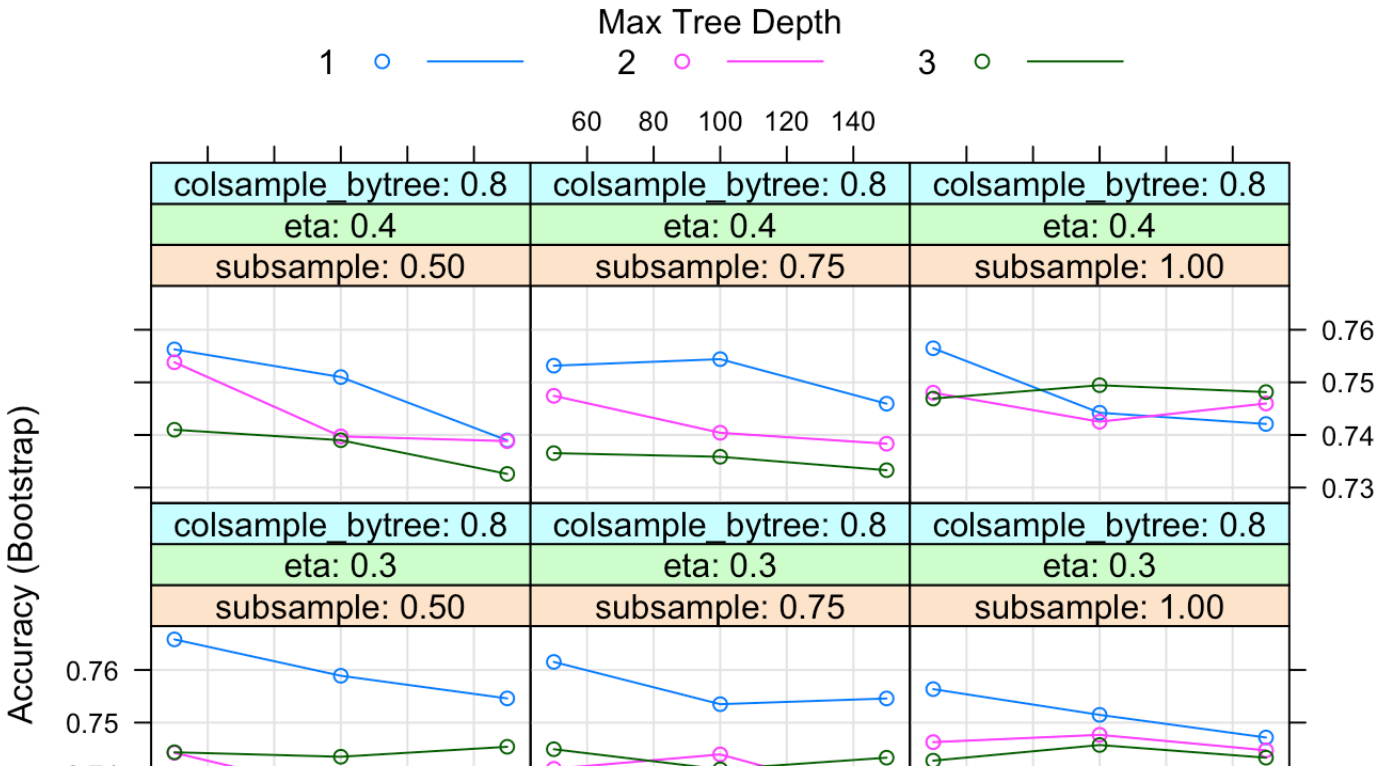
```
## Loading required package: xgboost
```

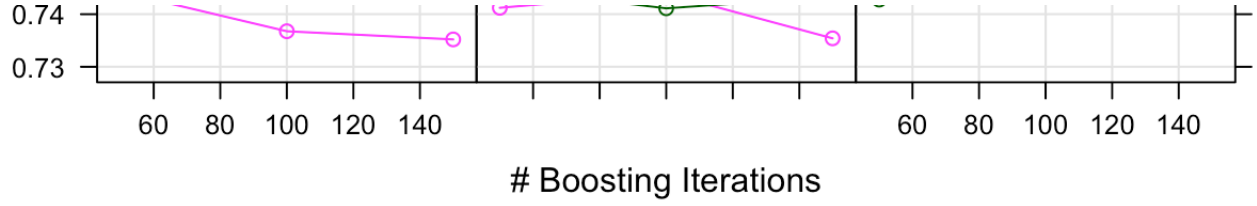
```
##
## Attaching package: 'xgboost'
```

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

```
par(mfrow = c(2,1))
reg_xgb <- train(Outcome ~ ., method = "xgbTree", data = pima_test)
plot(reg_xgb)
```


Max Tree Depth
1
2
3
colsample_bytree: 0.6
eta: 0.4
subsample: 0.50
subsample: 0.75
subsample: 1.00
eta: 0.3
Accuracy (Bootstrap)
Boosting Iterations
60
80
100
120
140
0.73
0.74
0.75
0.76


Max Tree Depth
1
2
3
colsample_bytree: 0.8
eta: 0.4
subsample: 0.50
subsample: 0.75
subsample: 1.00
eta: 0.3
Accuracy (Bootstrap)
60
80
100
120
140
0.73
0.74
0.75
0.76




```
pred_xgb <- predict(reg_xgb, pima_test)
cm_xgb<- confusionMatrix(pred_xgb,pima_test$Outcome)
cm_xgb
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 96 14
##          2  5 38
##
##                Accuracy : 0.8758
##                  95% CI : (0.8129, 0.9236)
##     No Information Rate : 0.6601
##     P-Value [Acc > NIR] : 9.515e-10
##
##                   Kappa : 0.7111
##
##  Mcnemar's Test P-Value : 0.06646
##
##             Sensitivity : 0.9505
##             Specificity : 0.7308
##          Pos Pred Value : 0.8727
##          Neg Pred Value : 0.8837
##              Prevalence : 0.6601
##          Detection Rate : 0.6275
##    Detection Prevalence : 0.7190
##       Balanced Accuracy : 0.8406
##
##        'Positive' Class : 1
##
```

It appears that the XGBoost model was the superior approach compared to the three previously tested regression approaches. Its balanced accuracy is impressive at 0.8403, and its specificity is very high at 0.9307. Specificity has been shown to be lower in this modelling study, but this approach had the highest value at 0.7500.

Discussion

```
# Create a knitr table of the regression approaches and their values#
cm<- data.frame(c("General Logistic Model", "k Nearest Neighbours", "Random Forest",
"XGBoost"), c(cm_glm$byClass[1], cm_knn$byClass[1], cm_rf$byClass[1], cm_xgb$byClass[
1]), c(cm_glm$byClass[2], cm_knn$byClass[2], cm_rf$byClass[2], cm_xgb$byClass[2]), c(
cm_glm$byClass[11], cm_knn$byClass[11], cm_rf$byClass[11], cm_xgb$byClass[11]))
cm<- as_tibble(cm)
colnames(cm) <- c("Regression", "Sensitivity", "Specificity", "Balanced Accuracy")
cm %>% knitr::kable()
```

Regression	Sensitivity	Specificity	Balanced Accuracy
General Logistic Model	0.8712871	0.6923077	0.7817974
k Nearest Neighbours	0.9207921	0.5000000	0.7103960
Random Forest	0.8118812	0.6538462	0.7328637
XGBoost	0.9504950	0.7307692	0.8406321

The analysis of the four regression approaches shows that the XGBoost was the most superior, with the highest sensitivity, specificity, and balanced accuracy. The general linear model has the next highest balanced accuracy, followed by the k Nearest Neighbours and the Random Forest Approaches. To further improve the analysis of this study, it would be recommended to increase the diversity of logistic regression algorithms tested. The `caret` package that was needed in this study holds a plethora of different approaches that may be suitable for this investigation.

The results of this study are very timely, as being able to accurately predict which patients have high risk of being positive for T2DM is vital in robust prevention and treatment mechanisms. As the current health situation of the United States is unsteady due to the COVID-19 pandemic, this importance is only magnified for populations that face disproportionate gaps in health. This study will hopefully bring forward a greater call for investigation into disease incidence and comorbidities in a variety of Native American tribes, attempting to bridge long-overdue disparities.