Introduction to Data Science and Artificial Intelligence
PS0002 Project AY20/21 Sem 2

Name	Matriculation Number	Section
Chua Han Yun Corlyss	U1940555J	Prediction
Pravenisa D/O B Viswambharan	U1940985B	
Evelyn Chong Jia Ling	U1940642B	Classification
Moo Zi Hao	U1940860K	

**Note all figures in report are labelled in numerical order while figures in appendix are labelled in alphabetical order*

1) Introduction

In 2019, about 9.3% of the global population was diagnosed with diabetes[1]. Research has shown that the percentage of diabetic patients will continue to rise to 11% by 2045[2]. This is a growing concern as diabetes lowers the quality of life due to potential diabetic-related complications[3]. Hence, this project aims to predict a patient's glucose level, to determine prominent factors affecting plasma glucose concentration and predict whether one gets diabetes when given the other data present in the dataset and what is the most significant contributing factor. We will be using the dataset "PimaIndiansDiabetes2" provided in mlbench for our report.

We will be focusing on 2 machine learning (ML) techniques, prediction and classification but not clustering as class labels are already pre-defined and assigned in the dataset. For predicting a patient's glucose, we will be evaluating the root mean square error (RMSE) to determine its accuracy.

Here are our study questions:

1. Which are the most prominent factors that affect plasma glucose concentration in Pima Indian Women?
2. Which is the most effective prediction method for predicting the plasma glucose concentration in Pima Indian Women?
3. Which is the most important variable that contributes to gestational diabetes amongst Pima Indian Women?
4. Which is the most effective classification method for predicting whether a Pima Indian Woman has Gestational diabetes or not?

2.1) Data Description

In PimaIndiansDiabetes2, the variables are: (i) Pregnant: Number of times pregnant, (ii) Glucose: Plasma glucose concentration (glucose tolerance test), (iii) Pressure: Diastolic blood pressure (mm Hg), (iv) Triceps: Triceps skin fold thickness (mm), (v) Insulin: 2-Hour serum insulin (mu U/ml), (vi) Mass: Body mass index, (vii) pedigree: Diabetes pedigree function, (viii) Age: Age (years), (ix) Diabetes: Class variable (test for diabetes) with a total of 768 observations.

Since there are some missing values in the dataset, we remove all the missing values and are left with 392 observations. From PimaIndiansDiabetes2, only diabetes is a categorical variable while the rest are continuous variables. Hence for prediction, we decided to exclude diabetes as it is not continuous and we identify glucose as the outcome variable. As for classification we choose diabetes as the outcome variable. We then replace the variable "Diabetes" which consists of "pos" and "neg" with the variable "Result" which consists of "0" and "1" to classify whether a patient has diabetes or not.

2.2) Analysis and Visualisation
From Figure 1, the boxplot shows the glucose level among two groups: diabetic and non-diabetic people. In terms of glucose level, diabetic patients have a higher median, 1st and 3rd quartile as compared to non-diabetic people.

To determine whether there is any distinctive correlation between any variables, we plot the matrix scatter plot using library psych to visualise the relationships between any pair of 8 continuous variables. From Figure A in appendix, both the numerical measure and visualisation of the relationship between any pair of variables show that glucose and insulin are moderately linearly related with the second highest Pearson's coefficient of 0.58, followed by the relationship between glucose and age with 0.34 for Pearson's coefficient. As a result, we choose to focus on glucose and identify significant variables affecting it. Also, we wish to classify patients according to their diabetic conditions.

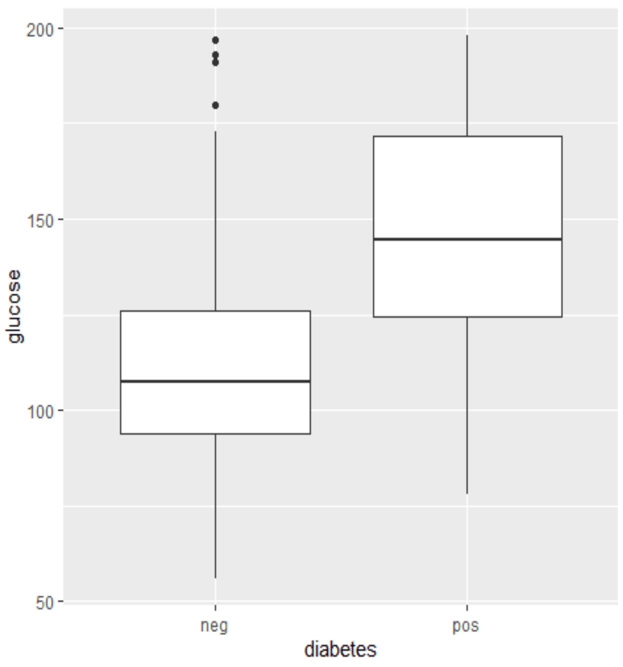

Figure 1: Boxplot of Glucose against Diabetes

3) Methodology

3.1) Splitting test sets and training sets
We split the data into 80% training and 20% test sets according to the commonly used splitting rule. The training set is used for creating a predictive model and the test set is used for evaluation of the model.

3.2) Prediction
Prediction is a basic ML technique that applies the trained dataset to a new data whose output is the predicted continuous outcome value.

3.2.1) K-nearest neighbors (kNN) Regression – Prediction
kNN regression is a simple algorithm popularly used for regression prediction. In the model, the RMSE value was given as 21.3. As the data points are scattered around with some points lying relatively close to the 45-degree line shown in Figure B, it is insufficient to indicate a good prediction performance.

3.2.2) Linear Regression – Prediction
Firstly, the summary statistics was used to determine which predictor has the greatest impact on the mode1. As shown in Figure C, insulin and age have a significant impact on the patient's glucose levels. The adjusted R-sq = 0.385, which shows that the proposed linear model can explain 38.5% of the variation in outcome glucose. The RMSE value is 21.0049.

To help us understand and improve the linear regression model, we plot diagnostic plots and focus on the residual plot. From Figure D, it shows a clear quadratic pattern in the residual which implies that the model has room for improvement. Next, we remove the outlier 585 and added second order terms of insulin and age as they are highly related to glucose with $|r| \geq 0.3$ in the correlation coefficient matrix (Figure E). As the RMSE reduces to 18.7 and the points in the residual plots are scattered randomly (Figure F), this implies that the fitting of the model has been improved.

For our project, linear regression has a lower RMSE value when compared to kNN regression. Therefore, it has a better predicting performance. This may be because kNN regression is not a good prediction for a dataset with higher dimensions. Another limitation of kNN is that it has no explicit model, so we are unable to interpret the effects of predictor variables on the outcome - glucose.

3.3) Classification
Classification is another supervised ML technique, which specifies the class to which data elements belong to. Similar to regression, classification constructs a succinct model to predict the value of the outcome variable using the predictor or independent variables. Here, we use all the other variables in the dataset as predictors to classify patients into 2 categories, either diabetic or non-diabetic.

3.3.1) Logistic Regression – Classification
Logistic regression is used to model a binary outcome variable, in our case- diabetes, by modelling the log odds of the outcome as a linear combination of predictor variables. Also, from the output shown in Figure 2, we can tell that "pedigree" is the most significant variable causing diabetes with simple calculations and interpretation as follows: For every one unit increase in 'pedigree', the ln(odds) of having diabetes increases by 0.7259, ie. the odds of being diabetic increases by $e^{0.7259}$ = 2.07 times.

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -9.4446976  1.3343964  -7.078  1.46e-12 ***
pregnant     0.0853357  0.0599309   1.424   0.15447
glucose      0.0376512  0.0062978   5.978   2.25e-09 ***
pressure    -0.0105065  0.0132776  -0.791   0.42877
triceps      0.0032515  0.0190242   0.171   0.86429
insulin     -0.0008402  0.0013773  -0.610   0.54181
mass         0.0892265  0.0297019   3.004   0.00266 **
pedigree     0.7258681  0.4611507   1.574   0.11548
age          0.0339550  0.0201199   1.688   0.09148
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

Figure 2: Output for Logistic Regression Classification

3.3.2) kNN – Classification
Another method for classification would be the kNN method whereby different k-values would give different classification results. Hence, we find an optimal k-value when using this method. According to Figure G, we can see that the best k-value would be k = 24.

3.3.3) SVM – Classification
SVM classification is another method that can be used. SVM is used to define a hyperplane which acts like a decision boundary that best separates the data into two classes. After comparing the results of the 2 kernel types we used, the linear kernel gave a more accurate classification.

Using the confusion matrix, we calculate the accuracy, recall, false positive and false negative for each model, as shown in Table 1. From the table, we deduce that kNN is the most accurate as it has the highest accuracy and recall (tied with SVM, radial kernel), and lowest false positive and false negative (tied with SVM, radial kernel) percentage among the 3 models.

$$Accuracy/\% = \frac{TP + TN}{\mathrm{TP + FP + FN + TN}} \times 100\%$$

$$Recall/\% = \frac{TP}{\mathrm{TP + FN}} \times 100\%$$

$$False\ Positive/\% = \frac{FP}{\mathrm{TP + FP + FN + TN}} \times 100\%$$

$$False\ Negative/\% = \frac{FN}{\mathrm{TP + FP + FN + TN}} \times 100\%$$

Where TP = true positive, FP = false positive, FN = false negative and TN = true negative

Classification Method	**Logistic Regression Classification**	**kNN Classification**	**SVM Classification**	
			Linear Kernel	**Radial Kernel**
Accuracy / %	78.5	82.3	79.7	78.5
Recall / %	56.5	60.9	56.5	60.9
False positive / %	8.86	6.33	7.59	10.1
False negative / %	12.7	11.4	12.7	11.4

Table 1: Summary of classification results

4) Conclusion

Using linear regression, our findings showed that insulin and age are the most prominent factors affecting plasma glucose concentration. After comparing, linear regression has a lower RMSE value than kNN regression. Therefore, linear regression has a better predictor model. To further improve our model, we can normalize all the data in the dataset and use it for prediction.

As for classification, we see that pedigree is the most significant variable causing diabetes from the logistic regression model. As we compare the different methods for classification, we conclude that the most accurate method for classifying the data into diabetic vs. non-diabetic would be the kNN classification method. However, there are limitations for this method. Firstly, the kNN method is computationally expensive and sensitive to the scale of data and irrelevant features. For further improvement of the kNN method, we could remove outliers from our dataset as the accuracy for classification would be compromised.

5) Appendix

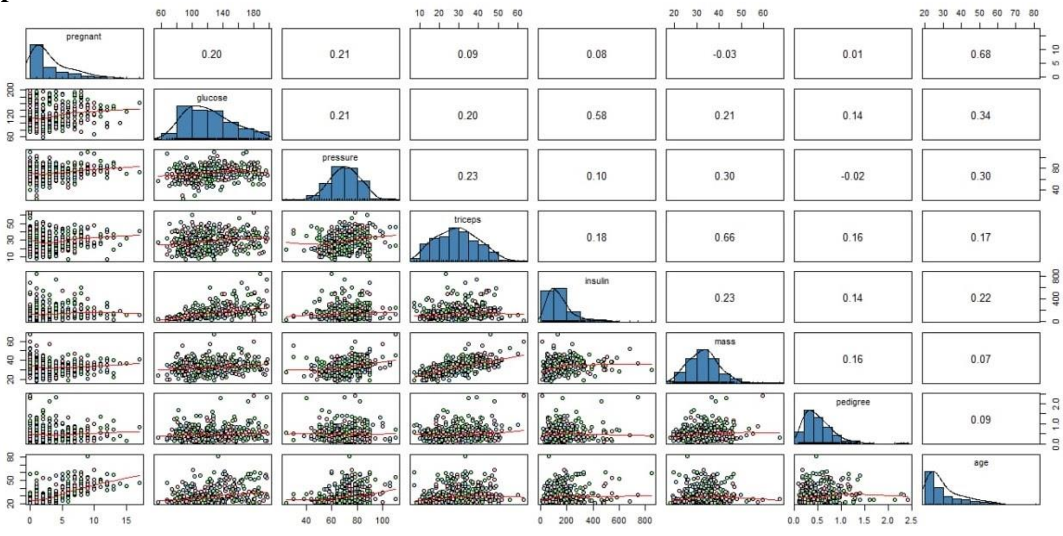

Figure A: Matrix scatter plot of continuous variables in PimaIndiansDiabetes2

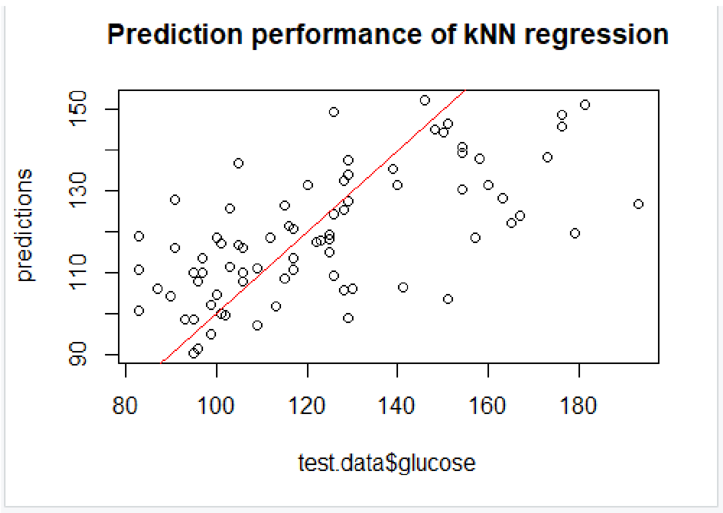

Figure B: Prediction Performance for kNN

```
(Intercept) 56.07990    9.82195   5.710   2.69e-08 ***
pregnant    -0.29917    0.58883   -0.508   0.61177
pressure     0.24395    0.12643   1.930   0.05459
triceps      0.11390    0.18038   0.631   0.52822
insulin      0.12991    0.01186   10.956  < 2e-16 ***
mass         0.10469    0.27444   0.381    0.70312
pedigree     5.73643    4.06349   1.412    0.15906
age          0.63763    0.19591   3.255    0.00126 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 24.88 on 305 degrees of freedom
Multiple R-squared:  0.399,       Adjusted R-squared:  0.3852
F-statistic: 28.92 on 7 and 305 DF,  p-value: < 2.2e-16
```

Figure C: Estimated Coefficient for Linear Regression

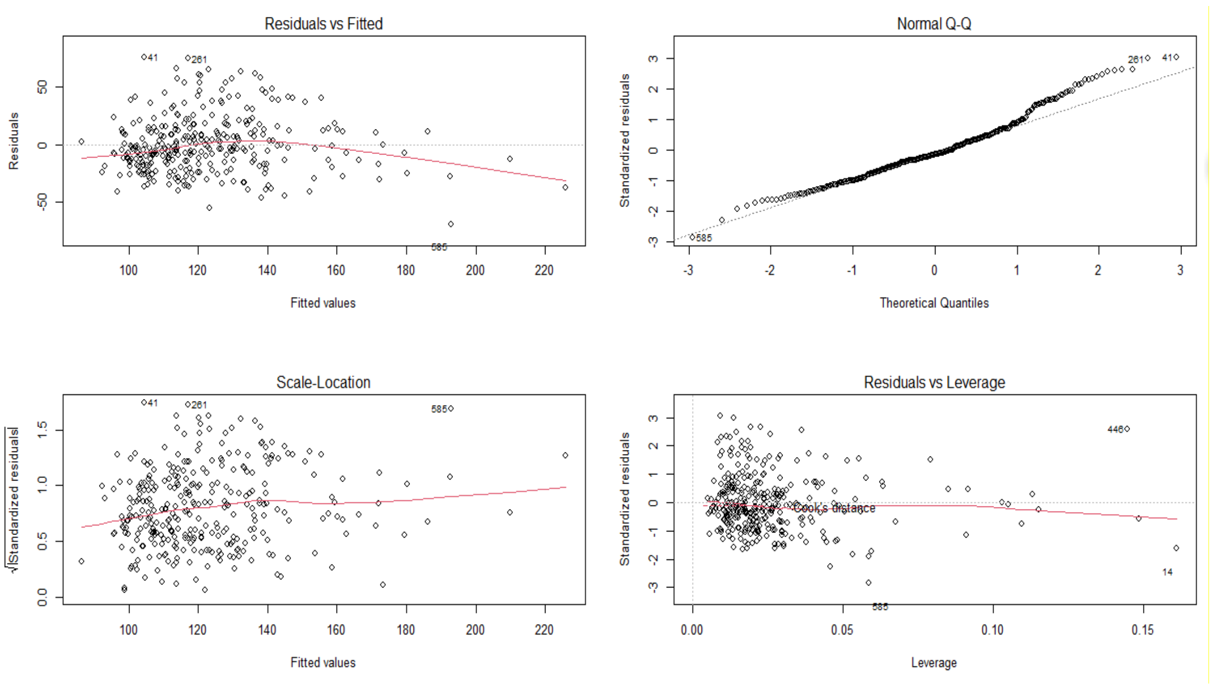

Figure D: Diagnostic plots before outlier is removed

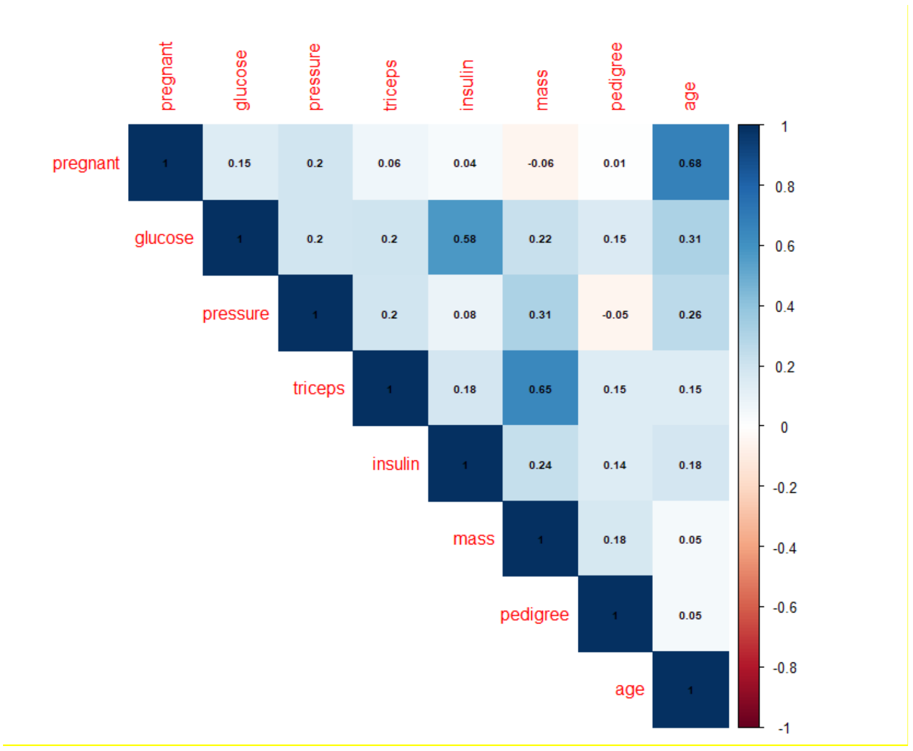

Figure E: Corrplot

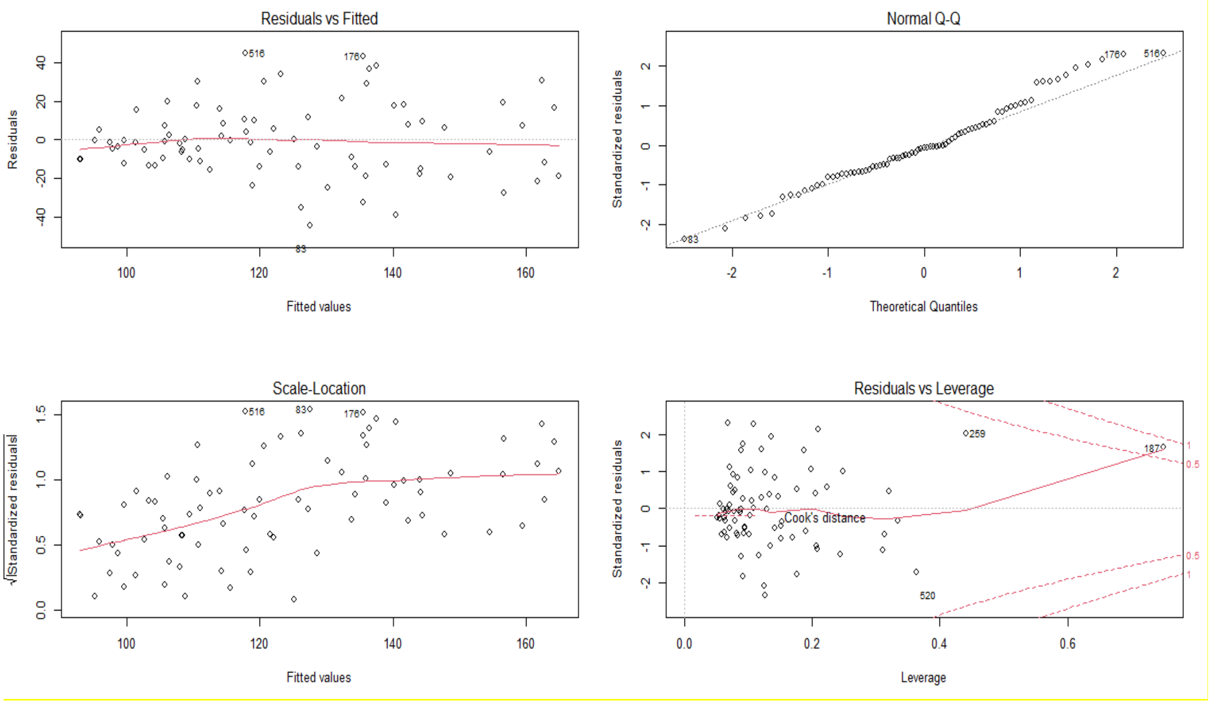

Figure F: Diagnostic plots after outlier is removed

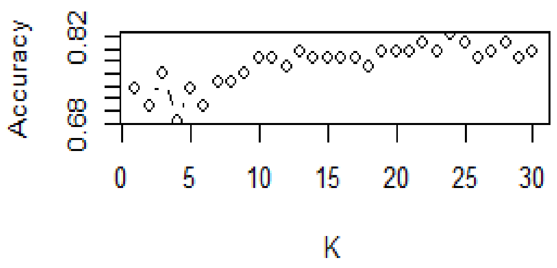

Figure G: Accuracy plot to find optimal value of k

R Code

```
install.packages("dplyr")
library(dplyr)
install.packages("mlbench")
library(mlbench)
install.packages("ggplot2")
library(ggplot2)
install.packages("corrplot")
library(corrplot)
install.packages("caret")
library(caret)
install.packages("class")
library(class)
install.packages("e1071")
library(e1071)

data(PimaIndiansDiabetes2)
PimaIndiansDiabetes2
```

#prepare data - Since some of the data provided is incomplete, we have removed those incomplete cases to form a complete data set.

```
PI <-PimaIndiansDiabetes2[complete.cases(PimaIndiansDiabetes2),]
PI1 = PI[-9]
```

Boxplot for dataset

```
PI%>%ggplot(aes(x=diabetes, y=glucose)) + geom_boxplot()
```

Matrix scatter plot

```
pairs.panels(PI[,-10],
        method = "pearson", # correlation method
        hist.col = "steelblue",
        pch = 21, bg = c("pink", "light green", "light blue"),
        density = TRUE, # show density plots
        ellipses = FALSE # show correlation ellipses
)
```

Knn - prediction

```
#Split the data into training and test sets
set.seed(100)
training.idx <- sample(1: nrow(PI1), nrow(PI1)*0.8)
train.data <- PI1[training.idx, ]
test.data <- PI1[-training.idx, ]

# Fit the model on the training set. what number to put instead of 10? result or diabetes
library(caret)
set.seed(101)
model.knn <- train(
  glucose~., data = train.data, method = "knn",
  trControl = trainControl("cv", number = 10),
  preProcess = c("center","scale"),
  tuneLength = 10
)

# Best tuning parameter k that minimize the RMSE
model.knn$bestTune
#[k=11]

# Make predictions based on the test data
predictions<-predict(model.knn, test.data)
head(predictions)

# Compute the prediction error RMSE
RMSE(predictions, test.data$glucose)
#[output= 21.34912]
```

Prediction : Linear Regression

```
set.seed(100)
training.idx <- sample(1: nrow(PI1), nrow(PI1)*0.8)
train.data <- PI1[training.idx, ]
test.data <- PI1[-training.idx, ]

#Prediction : Linear Regression
```

```
lmodel<- lm(glucose ~., data = train.data)
summary(lmodel)

RMSE(predictions, test.data$glucose)
#rmse = 21.0049

#create multiple plots on the same page
par(mfrow=c(2,2))
plot(lmodel)

#Visualize the correlation between the outcome medv and each predictor
corrplot(cor(train.data), type="upper", method="color",addCoef.col = "black",number.cex = 0.6)

#remove the outlier
PI2<-PI1[-c(585),]
set.seed(100)
training1.idx <- sample(1: nrow(PI2), size=nrow(PI2)*0.8)
train.data1 <- PI2[-training.idx,]
test.data1 <- PI2[-training.idx, ]
#second order term is added
lmodel1=lm(glucose
~I(insulin^2)+I(age^2)+I(insulin*age)+pregnant+pressure+triceps+insulin+mass+pedigree+age, data
= train.data1)
summary(lmodel1)
predictions1 <- predict(lmodel1, test.data1)
RMSE(predictions1, test.data1$glucose)
par(mfrow=c(2,2))
plot(lmodel1)
#18.74496
```

Logistic - classification

```
PI_numeric=PI
PI_numeric[,1:8]<- sapply(PI[,1:8], as.numeric)
PI_numeric
PI_numeric<-PI_numeric%>%mutate(result=factor(ifelse(diabetes=="pos", 1,0)))%>%select(-
diabetes)

set.seed(100)
training.idx <- sample(1: nrow(PI_numeric), size=nrow(PI_numeric)*0.8)
train.data <-PI_numeric[training.idx, ]
test.data <- PI_numeric[-training.idx, ]
mlogit <- glm(result ~., data = train.data, family = "binomial")
summary(mlogit)

Pred.p <-predict(mlogit, newdata =test.data, type = "response")
result_pred_num <-ifelse(Pred.p > 0.5, 1, 0)
result_pred <-factor(result_pred_num, levels=c(0, 1))
mean(result_pred ==test.data$result )
tab <-table(result_pred,test.data$result)
tab
```

Knn - classification

```
nor <-function(x) { (x -min(x))/(max(x)-min(x)) }
PI_numeric[,1:8] <- sapply(PI_numeric[,1:8], nor)
```

```
set.seed(100)
training.idx <- sample(1: nrow(PI_numeric), size=nrow(PI_numeric)*0.8)
train.data <-PI_numeric[training.idx, ]
test.data <- PI_numeric[-training.idx, ]

library(class)
set.seed(101)
knn1<-knn(train.data[,1:8], test.data[,1:8], cl=train.data$result, k=2)
mean(knn1 ==test.data$result)
table(knn1,test.data$result)

#try for a better k
ac<-rep(0, 30)
for(i in 1:30){
  set.seed(101)
  knn.i<-knn(train.data[,1:8], test.data[,1:8], cl=train.data$result, k=i)
  ac[i]<-mean(knn.i ==test.data$result)
  cat("k=", i, " accuracy=", ac[i], "\n")
}
plot(ac, type="b", xlab="K",ylab="Accuracy")

set.seed(101)
knn2<-knn(train.data[,1:8], test.data[,1:8], cl=train.data$result, k=24)
mean(knn2 ==test.data$result)
table(knn2,test.data$result)
```

SVM

```
#split the data
set.seed(100)
training.idx <- sample(1: nrow(PI_numeric), size=nrow(PI_numeric)*0.8)
train.data <-PI_numeric[training.idx, ]
test.data <- PI_numeric[-training.idx, ]

#svm classification
library(e1071)
set.seed(100)
m.svm<-svm(result~., data = train.data, kernel = "linear")
summary(m.svm)

#predict newdata in test set
pred.svm <- predict(m.svm, newdata=test.data[,1:8])

#evaluate classification performance and check accuracy
table(pred.svm, test.data$result)
mean(pred.svm ==test.data$result)

#improve the model
set.seed(100)
m.svm.tune<-tune.svm(result~., data=train.data, kernel="radial", cost=10^(-1:2), gamma=c(.1,.5,1,2))
summary(m.svm.tune)

#visualize results of parameter tuning
plot(m.svm.tune)
```

```
#confusion matrix and accuracy
best.svm = m.svm.tune$best.model
pred.svm.tune = predict(best.svm, newdata=test.data[,1:8])
table(pred.svm.tune, test.data$result)
mean(pred.svm.tune ==test.data$result)

#In this case, the linear kernel gives a better classification as compared to the radial kernel. This could
be due to the way the data is scattered. Hence, we  see that SVM with linear kernel classifies over
79% of the patients correctly into diabetic and non-diabetic.
```

6) References

[1] https://www.diabetesresearchclinicalpractice.com/article/S0168-8227(19)31230-6/fulltext

[2] https://www.statista.com/statistics/271464/percentage-of-diabetics-worldwide/#:~:text=Diabetics%20prevalence%20worldwide%202019%20and%202045&text=Around%209.3%20percent%20of%20the,chronic%20high%20blood%20sugar%20levels

[3] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5394731/