Modelling Medical Cost of Diabetic Patients

Adeetya Vikrama Tantia
Roll No: MS14033

A dissertation submitted for the partial fulfilment
of BS-MS dual degree in Science

Under the guidance of

Dr. N G Prasad and Prof. Kanchan Jain

April 2019

Indian Institute of Science Education and Research, Mohali
Sector - 81, SAS Nagar, Mohali 140306, Punjab, India

Contents

Certificate of Examination

This is to certify that the dissertation titled **"Modelling Medical Cost of Diabetic Patients"** submitted by **Adeetya Vikrama Tantia** (Reg. No. MS14033) for the partial fulfillment of BS-MS dual degree programme of the Institute, has been examined by the thesis committee duly appointed by the Institute. The committee finds the work done by the candidate satisfactory and recommends that the report be accepted.

Prof. Kanchan Jain
(Co-Supervisor)

Prof. Somdatta Sinha

Dr. Rhitoban Raychaudhary

Dr. N G Prasad
(Supervisor)

Dated: 26.04.2019

Declaration

Declaration

The work presented in this dissertation has been carried out by me under the guidance of Dr. N G Prasad and Prof. Kanchan Jain, Panjab University at the Indian Institute of Science Education and Research Mohali.

This work has not been submitted in part or in full for a degree, a diploma,or a fellowship to any other university or institute. Whenever contributions of others are involved, every effort is made to indicate this clearly, with due acknowledgement of collaborative research and discussions. This thesis is a bonafide record of original work done by me and all sources listed within have been detailed in the bibliography.

Adeetya Vikrama Tantia
(Candidate)

Dated: April 26, 2019

In my capacity as the supervisor of the candidate's project work, I certify that the above statements by the candidate are true to the best of my knowledge.

Dr. N G Prasad
(Supervisor)

Prof. Kanchan Jain
(Co-Supervisor)

Acknowledgement

I am extremely grateful to my parents, who have supported me in this endeavour thoroughly.
I would like to thank my thesis supervisors Dr. N G Prasad and Prof. Kanchan Jain, without whose help and supervision, this thesis would have never been possible.
My thanks are also due to Dr. Arindam Chanda of GD Hospital and Diabetes Institute, Kolkata who gave us access to the patients, and to the staff of the Diabetology Department for their co-operation.

Adeetya Vikrama Tantia
MS14033
IISER Mohali.

List of Figures

List of Tables

If—

Rudyard Kipling

If you can keep your head when all about you
Are losing theirs and blaming it on you,
If you can trust yourself when all men doubt you,
But make allowance for their doubting too;
If you can wait and not be tired by waiting,
Or being lied about, don’t deal in lies,
Or being hated, don’t give way to hating,
And yet don’t look too good, nor talk too wise:

If you can dream—and not make dreams your master;
If you can think—and not make thoughts your aim;
If you can meet with Triumph and Disaster
And treat those two impostors just the same;
If you can bear to hear the truth you’ve spoken
Twisted by knaves to make a trap for fools,
Or watch the things you gave your life to, broken,
And stoop and build ’em up with worn-out tools:

If you can make one heap of all your winnings
And risk it on one turn of pitch-and-toss,
And lose, and start again at your beginnings
And never breathe a word about your loss;
If you can force your heart and nerve and sinew
To serve your turn long after they are gone,
And so hold on when there is nothing in you
Except the Will which says to them: ‘Hold on!’

If you can talk with crowds and keep your virtue,
Or walk with Kings—nor lose the common touch,
If neither foes nor loving friends can hurt you,
If all men count with you, but none too much;
If you can fill the unforgiving minute
With sixty seconds’ worth of distance run,
Yours is the Earth and everything that’s in it,
And—which is more—you’ll be a Man, my son!

Introduction

Never doubt that a small group of thoughtful,
committed citizens can change the world;
indeed, it's the only thing that ever has.
-Margaret Mead

The International Diabetes Foundation estimates 72.9 million Indians to be currently suffering from diabetes, with this number set to increase to 134.3 million by 2045.[21] This makes India the country with the second highest number of adults living with diabetes. But, mean healthcare expenditure on diabetes per person in 2017 was only ID 426,[21] far behind countries other countries.

The International Diabetes Foundation's 2045 conservative projections, assuming mean per capita expenditure and diabetes prevalence rate remain constant, estimate global cost of Diabetes to increase to USD 776 Billion, which represents a 7% growth.

Insurance exists to protect oneself against increasing and unforseen costs. Existing health insurance plans were unable to appropriately cover expenses of diabetes.

Only recently have specific insurance plans for Diabetes sprung up but all seem to use age as a proxy to classify patients into premium bands and then offer adjustments based on medical state.

It is believed that doing so is convenient, but a more equitable solution exists which would not only help patients by appropriately identifying their costs, but would also help insurance companies make health classes in their diabetes insurance policy using medical indicators as well as age.

An existing dataset is used to identify important indicators of diabetes using various Machine Learning Classification techniques.

Machine Learning Classification Models would help us identify these indicators using variable importance.

Collected data from GD Hospital & Diabetes Institute in Kolkata, is used to create a Generalized Additive Model (GAM) that links these indicators of diabetes to the annual expenditure of the patient.

GAM Models have been previously used to model new pricing systems and thus were chosen due to their flexibility and wider range of applicability.

Clustering algorithms were subseuently cluster the patients into different health classes, based on annual spending but categorized via medical attributes.

Part I

Identification of Indicators of Diabetes

Chapter 1

Machine Learning Classification Algorithms

Machine Learning Classification algorithms such as Logistic Regression, K- Nearest Neighbours, Support Vector Machines, Naive Bayes, Decision Tree and Random Forest are used to classify our dataset. The VarImp function is used to see how important each variable is in classifying the dataset.

All the algorithms used are Supervised Machine Learning Algorithms, i.e. these algorithms require a training set of data which contains not only the attributes, X but also the correct class, Y. These algorithms use this training set of data to shape the model in the required fashion and are then able to classify the test set data.

1.1 Logistic Regression[1]

1.1.1 Introduction

Logistic Regression was developed by D. R. Cox in 1958[22] as a statistical method to find the relation between independent variables and a target binary variable.

In the model, dependent variable prediction is given by a summation of products of the independent variable and a coefficient. The value of the coefficient is a measure of the effect of the independent variable on the dependent variable, adjusted for all other independent variables.

Thus the model helps us predict the dependent variable for new values of the independent variables and helps explain the relative contribution of each independent variable.

1.1.2 The Model

For a model with x_i's being the independent variables and y being the binary target variable, the logit model can be written as -

$$logit(E(y)) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{1.1}$$

where $logit(E(y))$ is nothing but $log(\frac{E(y)}{1-E(y)})$. This log transformation is necessary to avoid values of x that will give y values not between 0 and 1.[23]
Thus equation (1.1) can be transformed into -

$$E(y) = \frac{e^{\beta_0+\beta_1 x_1+\beta_2 x_2}}{1 + e^{\beta_0+\beta_1 x_1+\beta_2 x_2}} \tag{1.2}$$

Equation (1.2) ensures that the values produced are between 0 and 1, to represent the probability of y being equal to 1.

1.1.3 Assumptions & Requirements

- Binary Logistic regression may only be used for a binary dependent variable.
- As the model estimates the probability of an event occurring $(P(Y = "Occured"))$, the dependent variable must be coded accordingly.
- The model should not be over fitted with more than required and/or nonsensical variables.
- Logistic regression requires each record to be independent. The model should not exhibit multicollinearity i.e. independent variables must not be linear functions of other independent variables.
- Logistic regression requires that the independent variables be linearly related to the log odds of the event to be modelled.
- Logistic regression needs larger sample sizes as the Maximum Likelihood Estimates method is less powerful than the Ordinary Least Squares method, used to estimate unknown parameters.

- Error terms need not be multivariate normally distributed–but multivariate normality provides stabler solutions.

- Variance of Error terms may be heteroscedastic for different levels of independent variables.

- Logistic regression is able to handle both continuous data and discrete data as independent variables.

1.1.4 Fitting the Model

Logistic Regression model fitting is based on the Maximum Likelihood Method. So for each observation with independent variables, X_i and target variable y_i, we can let $E(y_i) = p(X_i)$. Therefore the likelihood for n observations can be written as -

$$L(\beta) = \prod_{i=1}^{n} p(x_i)^{y_i}(1 - p(x_i))^{1-y_i} \tag{1.3}$$

Using the maximum likelihood method, we can get parameter estimates as well as variances for each parameter in the model.

1.2 K-Nearest Neighbours[2]

1.2.1 Introduction

The K-Nearest Neighbours algorithm is a non-parametric classifier that classifies new data on the basis of the most frequently represented class in the K-nearest neighbours of the new point.

If two or more such samples exist of K-Nearest neighbours, the sample with the minimum average distance to the new point is chosen.

As $K \to \infty$, the K-Nearest neighbours algorithm becomes the Bayes optimal decision rule.[24]

1.2.2 Algorithm

The K-Nearest Neighbours algorithm was proposed by Cover[25] & Hart[26] in 1968.

Due to the ease and efficiency of the Euclidean distance measure, K-NN classifiers usually use Euclidean distances.[27] Other measures such as Taxicab distance and Cosine distance are also available.

When a new data point to be classified is provided to the K-NN algorithm, it calculates the K closest points to that new point in the n-dimensional feature space (where n is the number of independent variables). It finds the the dependent variable value that is the most represented in the K neighbours and assigns that value to the new point.

K-NN being a lazy learning algorithm, i.e. it doesn't have a true learning period and classifies the new point by actively using the training set at time of classification, is computationally intensive.

The most effective values of K are in the range of 30-45.[28]

1.3 Support Vector Machines[3]

1.3.1 Introduction

Support Vector Machines separate the classes by a hyperplane defined by a normal vector and a bias term.

The most favourable separating hyperplane would be one that would maximizes the margin, i.e. the distance between the hyperplane and the nearest points of both classes.

Kernel functions alongwith SVM can be used so as to make non-linear decision boundaries. This allows for much more precise decision functions, as real - world data is usually non-linearly separable. The kernel function, maps the original non-linear observations into a higher-dimensional space in which they might become separable, and then the SVM algorithm is applied in this new higher dimensional space.

SVMs were originally designed for binary target variables, but using a one-against-one and one-against-all approach, they can be extended for multiple target class classification.

1.3.2 The Model

The hyperplane can be specified by its normal vector, $\mathbf{w}$ and its bias term,b .

The kernel function is given by k and associated with the non-linear mapping function Φ.

Then the formula becomes -

$$\mathbf{w}.\Phi(x) + b = 0 \tag{1.4}$$

which will yield the decision function -

$$f(x) = y^* = sgn(\langle \mathbf{w}.\Phi(x) + b \rangle) \tag{1.5}$$

The sgn function here is the sign function which gives a value of $+1$ if the value is > 0 and -1 otherwise. If $y^* = 1$ then x belongs to the corresponding class and if $y^* = -1$, then it does not.

1.4 Naïve Bayes[4]

1.4.1 Introduction

The Naïve Bayes classifier works on the Bayes' Theorem of posterior probability. It is called Naïve due to its strong independence assumption, that each variable's effect is independent of the other.
It is extremely fast and can be run quite well on small datasets as well but its strong independence assumptions make it unsuitable for a lot of different natural models.

1.4.2 The Model

We have n independent attributes given by $x_1, x_2, ...x_n$ and let the target variable have m classes given by $c_1, c_2, ..., c_m$.
Then to classify a new data point, represented by $\mathbf{X}$, we need to find the maximum $P(c_i|\mathbf{X})$.
This is obtained via Bayes' Theorem as -

$$P(c_i|\mathbf{X}) = \frac{P(\mathbf{X}|c_i)P(c_i)}{P(\mathbf{X})} \tag{1.6}$$

As $P(X)$ is just a normalizing factor and independent of class, it can be ignored.
From the independence assumption and given $\mathbf{X} = \{x_1, x_2, ..., x_n\}$, we get -

$$P(\mathbf{X}|c_i) = \prod_{k=1}^{n} P(x_k|c_i) \tag{1.7}$$

Also, to calculate $P(c_i)$, we simply divide occurrences of c_i seen in the training data(of say N records), divided by the total data points, i.e.

$$P(c_i) = \frac{\sum_{k=1}^{N} 1_{c=c_i}}{N} \tag{1.8}$$

Here $1_{c=c_i}$ is the indicator function taking a value of 1 if $c = c_i$ and 0 otherwise.

So, when a new data point is presented to the algorithm, it calculates the probability of the data point being in each class given its attributes using equation (1.6). It then finds the maximum of these probabilities and classifies the new data point into that class.

1.4.3 Smoothing[5]

The Naïve Bayes classifier in this form is susceptible to incorrect classification if it encounters an unseen value of an independent variable, as the probability $P(x_k|c_i)$ in such a case is always 0. To solve this, we turn to smoothing.

We specifically use Laplace smoothing[29] which adds a psedo-count, α in every probability estimate as follows -

$$P(\hat{x_k}|c_i) = \frac{|x_k| + \alpha}{N + \alpha n} \tag{1.9}$$

By doing so, no value of x_k has zero probability.

1.5 Decision Trees[6]

1.5.1 Introduction

Decision trees divide the feature space into disjoint cells. Each disjoint cell would contain atleast one point from the training set. The disjoint cell is classified into a particular class, if that class has maximum representation in that cell.

Then, once a new data point is to be classified, it can be plotted on the feature space to classify it.

In decision trees, we start from the top (root) node and then follow the branches as per the feature criteria to get to branch nodes. We will reach, in the end, the leaf node that doesn't split any further and will be classified based on the class most represented in that leaf node in the training set.

1.5.2 Algorithm[7]

Given a training sample, we use a set of non-negative integer valued weights, $\mathbf{w} = (w_1, w_2, ...w_n)$ where n is the number of data points in the training sample.

Each node of the tree is defined by a vector of weights which have non-zero elements when the corresponding observations are elements of the node and zero if they're not.

For $j = 1, 2, ...m$ there are m (Number of features) partial hypotheses given by

$$H_0^j : D(\mathbf{Y}|X_j) = D(Y)$$

where $D(Y|X)$ is the conditional distribution of Y given X. The global null hypothesis is thus given by $H_0 = \cap_{j=1}^m H_0^j$. These null hypotheses essentially say that the m covariates and the response variable are independent. When we cannot reject this hypothesis at a pre-specified α level, our algorithm should stop as if the covariates and response variable are independent, there is no point in making further splits.

When we do reject this global null hypothesis, we subsequently choose the covariate X_j that has the strongest association with Y.

In the feature space of $\mathcal{X}_j$, we then choose a set $A^* \subset \mathcal{X}_j$ to split $\mathcal{X}_j$ into two parts -

$$A^* \text{ \& } \mathcal{X}_j \backslash A^*$$

We use weights, $\mathbf{w}_{right}$ and $\mathbf{w}_{left}$ given by

$$\mathbf{w}_{right,i} = w_i I(X_{ji} \in A^*) \tag{1.10}$$

$$\mathbf{w}_{left,i} = w_i I(X_{ji} \notin A^*) \tag{1.11}$$

for all $i = 1, 2, ..., n$ where $I(\cdot)$ is the indicator function.

We repeat these steps until we can no longer reject the global null hypothesis.

1.6 Random Forest

1.6.1 Introduction[8]

Random Forest is an ensemble-learning model which trains multiple classifiers and then combines the results via a voting process.

Boosting[30] is another ensemble training model,which uses iterative retraining, in which incorrectly classified data points are given increased weightage as the iterations progress.[31] Bagging,[32] another model type trains multiple classifiers on bootstrapped samples from the training set. Bootstrapped samples are smaller subsets of the original data sampled with replacement multiple times to calculate each boostrapped sample's required statistic. This reduces the variance of the classification.

Boosting is much more computationally intensive and slower than bagging but, it is considerably more accurate than bagging. Boosting can reduce both the variance and the bias of the classification. But it also has costs - it is slow, prone to overtrain the model and can be sensitive to noise.[33]

Random Forests use a better method of bootstrapping and show accuracy comparable to boosting models, but without the drawbacks of boosting.[34] They are even less computationally intensive.

1.6.2 Algorithm

Random Forest algorithm trains multiple Decision Trees,[35] each trained on bootstrapped samples of the training data, and chooses from a randomly chosen subset of the input variables to determine a split (for each node).

By limiting number of variables used to decide a split, computational complexity and correlation between trees are reduced. Trees in the Random Forest are not pruned, which could reduce the computational load even more.

For the classification, each tree casts a vote and the majority of votes decides the category of the new input variable.

1.7 AdaBoost Classification Trees[9]

1.7.1 Introduction

AdaBoost[30,36] uses boosting, a method which uses weights for each training set record and updates them to a higher value for the next classification iteration if they are misclassified in the previous one. Once the training is complete, the classifiers are combined into one, powerful classifier, which is highly accurate on the training set. It thus, shows an extremely high accuracy.[37,38]

1.7.2 The Algorithm

Let the training set,D_n be given by -

$$D_n = \{(X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n)\} \tag{1.12}$$

Here, Y takes values of -1 or 1. A weight, $w_b(i)$ is assigned to each observation, X_i. At the start of the algorithm, this is taken to be $\frac{1}{n}$. This is the weight that will be updated after every step.

A basic classifier,$C_b(X_i)$ is built on D_n^b. The error of the classifier is given by ϵ_b and calculated as-

$$\epsilon_b = \sum_{i=1}^{n} w_b(i)\xi_b(i) \tag{1.13}$$

Here,

$$\xi_b = \begin{cases} 0 & C_b(x_i) = y_i \\ 1 & C_b(x_i) \neq y_i \end{cases} \tag{1.14}$$

The updated weights for the $b + 1^{th}$ classifier would be calculated as -

$$w_{b+1}(i) = w_b(i).e^{\alpha_b \xi_b(i)} \tag{1.15}$$

Here,

$$\alpha_b = \ln(\frac{1 - \epsilon_b}{\epsilon_b}) \tag{1.16}$$

These new weights are subsequently normalized.

If the error of the classifier is small, the weight will be increased more than if the error was larger. This is because more importance is given to the few mistakes made when the classifier achieves a high level of accuracy. α is interpreted as a learning rate.

This process is repeated for $b = 1, 2, 3..., B$. The final ensemble-classifier is built via a linear combination of all the other classifiers, weighted by α_b,

$$C(x) = sgn(\sum_{b=1}^{B} \alpha_b C_b(x)) \tag{1.17}$$

The sgn function here is the sign function which gives a value of $+1$ if the value is > 0 and -1 otherwise.

1.8 eXtreme Gradient Boosting - Linear[10]

eXtreme Gradient Boosting or XGBoost is relatively new but very popular ensemble-classifier.It can use either tree based models or linear models as its base model.
The model initializes by fitting a simple classifier to the data. It then computes the gradient of the loss function and fits a function to this gradient.
A new model is thus generated using the original model and the function, fit to the gradient of the loss function. This new model will have a lower error than that of the original model.
After being run for n iterations, the final model is expected to be much better at classifying the problem.

Chapter 2

Machine Learning Tools and Performance Measures

2.1 Cross Validation

Cross Validation is a method of getting better parameter estimates of a model when data is limited.

K-fold Cross Validation splits the dataset, $\mathcal{Q}$ into K mutually exclusive subsets $\mathcal{Q}_1, \mathcal{Q}_2, ..., \mathcal{Q}_K$ of equal size.

The algorithm is then both, trained and tested K times; for each time, $t \in \{1, 2, 3..., k\}$, it is trained on the set $\mathcal{Q} \backslash \mathcal{Q}_t$ and then it is tested on $\mathcal{Q}_t$. The estimate of accuracy is given by the total number of correct classifications divided by number of records in the training dataset.[39]

The estimates of parameters can therefore also be taken from each of these k estimates, usually resulting in better estimates.

2.2 Predictive Accuracy[1]

2.2.1 Confusion Matrix

We can use a Confusion Matrix to find the predictive accuracy of the model. We select a cutoff, usually 0.5. All predicted values $>$ the cutoff are classified as 1 and similarly all predicted values $<$ cutoff are classified as 0. Then we make a 2x2 table that has on one axis the observed values and on the other, the predicted values. The Confusion Matrix will be similar to -

		Predicted	
		1	0
Observed	1	a	b
	0	c	d

Table 2.1: Confusion Matrix

2.2.2 Accuracy & Balanced Accuracy

In the confusion matrix, if the model is a good fit, the values of a(True Positives) & d(True Negatives) will be high while b(False Negatives) & c(False Positives) will be low.

Accuracy is given by $\frac{a+d}{a+b+c+d}$.

Balanced Accuracy is a more accurate measure of accuracy when the test set is not balanced in terms of number of instances of each class. Balanced accuracy is calculated as the average of the proportion of correct classifications of each class. Thus, balanced accuracy is given by $\frac{1}{2}(\frac{a}{a+b} + \frac{d}{d+c})$.

2.2.3 Sensitivity

Sensitivity measures the percentage of actual positive instances correctly identified as such. It is therefore also known as the True Positive Rate.

It thus quantifies how well the classifier avoids false negatives.

Therefore sensitivity is given by $\frac{a}{a+b}$.

2.2.4 Specificity

Specificity measures the percentage of actual negative instances that are correctly identified as such.It is therefore also known as the True Negative Rate.

It thus quantifies how well the classifier avoids false positives.

Therefore specificity is given by $\frac{d}{d+c}$.

2.2.5 RoC (Receiver operating characteristic) Curves

We also examine the complete range of cutoff values from 0 to 1. For every possible cutoff value, a 2x2 table is made. Plotting the pairs of sensitivity($\frac{a}{a+b}$) and $1-$specificity($\frac{d}{c+d}$) on a scatter plot gives us an ROC curve.

Area Under the Curve(AUC)

The AUC is the area under the ROC curve. It provides a measure of fit of the model.[40] The AUC can vary from 0.5, where it has no predictive ability, to 1.0, where it has perfect predictive ability. The higher the AUC the better the predictability of the model. Points above the diagonal in the ROC space represent good classification results, whereas points below it, represent poor results (worse than random).

2.2.6 Cohen's Kappa[11]

Cohen's Kapppa compares the Observed Accuracy of the model with the Expected Accuracy(random chance).

Observed Accuracy is simply given by accuracy, $\frac{a+d}{a+b+c+d}$.

Expected Accuracy is given by multiplying the marginal frequency of a class from the observed values, by the marginal frequency of a class from the predicted values, and divided by the total number of instances, and then summing this value across all classes and dividing by the total number of instances again. So in our confusion matrix,

$$EA = (\frac{(a+c)(a+b)}{a+b+c+d} + \frac{(b+d)(c+d)}{a+b+c+d})\frac{1}{a+b+c+d} \tag{2.1}$$

Kappa is then calculated using the following formula -

$$\kappa = \frac{OA - EA}{1 - EA} \tag{2.2}$$

There is no universally agreed-upon way to interpret this statistic.

Landis & Koch,[41] providing no evidence, stated values < 0 as being poor, $0-0.20$ as slight, $0.21-0.40$ as fair, $0.41-0.60$ as moderate, $0.61-0.80$ as substantial, and $0.81-1$ as almost perfect.

Subsequently, Fleiss'[42] published equally arbitrary guidelines of > 0.75 as excellent, $0.40-0.75$ as good, and < 0.40 as poor.

2.2.7 No Information Rate

The No Information Rate(NIR) is the accuracy from a model that has no other information provided to it other than the prevalence of the classes, in the training set. Given only this information, this model would always choose the class that is in the majority and its accuracy would be equal to the prevalence of that class.

Thus, if our model's accuracy is lower than the NIR, that means that our model is doing a worse job than the NIR model which chooses the majority class irrespective of the values of the independent variables.

Thus, accuracy of a model should always be compared with the NIR so as to get a better idea of how much better or worse our model is actually doing.

2.3 VarImp

The varImp function[43] calculates the importance of each variable for the classifiers. The function scales the importance from 0 to 100, to provide a relative measure.

For Linear models, it returns the absolute value of the t-statistic for each model's parameter.

For Random Forests, for each tree, the accuracy is calculated on the out-of-bag portion. It then repeats this after permuting each predictor variable. The difference between these two values is averaged over all trees and then normalized via the standard error.

For AdaBoost Classification Trees, the importance is summed over each boosting iteration using the approach of the single tree model.

For other models, it conducts an ROC curve analysis for each variable. The area under the curve is then used as a measure of variable importance.

Chapter 3

Data and Preliminary Analysis

3.1 Provenance

The original dataset had been collected by the National Institute of Diabetes and Digestive and Kidney Diseases between 1965 and 1969.[44] A total of 2917 half and full blooded Pima Indians were examined.

The subject was said to be diabetic according to WHO guidelines,[45] i.e. , if the 2 hour post-load plasma glucose was at least 200mg/dl (11.1 mmol/l) at any examination or if the Indian Health Service Hospital serving the community found a glucose concentration of at least 200 mg/dl during the course of routine medical care.[46]

We use a trimmed dataset obtained which filtered out entries based on the following criteria[12] -

1. The subject is female.

2. The subject is atleast 21 years of age.

3. Only subjects which had a non-diabetic Glucose Tolerance Test($<$ 200mg/dl following ingestion of 75gm of Carbohydrate solution) and met either of the two following criteria were included.

   (a) Diabetes was diagnosed within 5 years of the examination

   (b) A Glucose Tolerance Test done $>$ 5 years later did not reveal diabetes.

4. If diabetes occurred within 1 year of the examination, that case was removed. Of the excluded examination, 75% had Diabetes diagnosed within 6 months.

This resulted in the trimming of the dataset from 2917 records to 768 records. Further, after removing missing values, the dataset is trimmed down to 392 observations.

3.2 Parameters

There are a total of 8 independent variables. The final column marked ”Outcome” is a class variable with 1s and 0s depicting whether the subject developed diabetes ultimately or not.The independent variables are-

1. Age (in years)
2. Body Mass Index ($= \frac{\text{Weight in kg}}{(\text{Height in m})^2}$)
3. 2-Hr Serum Insulin(μIU/ml)
4. Triceps Skin Fold Thickness(mm)
5. Diastolic Blood Pressure(mmHg)
6. Plasma Glucose Concentration at 2 Hours in an Oral Glucose Tolerance Test (OGTT) (mg/dl)
7. Number of times pregnant
8. Diabetes Pedigree Function

Patients are given a 75gm Glucose solution and their plasma glucose concentration and serum insulin levels are noted 2 hours later. The OGTT is meant to diagnose Type 2 Diabetes, while the serum insulin provides a measure of risk of developing diabetes.[47]
Triceps skin fold thickness is a measure of innate obesity.[48]
The number of pregnancies can increase the risk of development of Type 2 Diabetes, particularly if they suffered from gestational diabetes.[49]

3.2.1 Diabetes Pedigree Function[12]

The Diabetes Pedigree Function aims to distill the family history of diabetes mellitus of the subject into a numerical value. It uses information from parents, grandparents, full and half siblings, full and half aunts and uncles, and first cousins.

It gives a measure of the expected genetic influence of affected and unaffected relatives on the subject's eventual diabetes risk. It is given by

$$DPF = \frac{\sum_t K_i(88 - ADM_i) + 20}{\sum_j K_j(ALC_j - 14) + 50} \tag{3.1}$$

Here,

i ranges over all relatives, who developed diabetes by examination date

j ranges over all relatives, who did not developed diabetes by examination date

K_x percentage of genes shared with relative
$= 0.500$ when relative is parent or full sibling
$= 0.250$ when relative is half sibling, grandparent, aunt or uncle
$= 0.125$ when relative is a half aunt, half uncle or first cousin

ADM_i age of relative when diabetes was diagnosed

ACL_j age of relative at last non-diabetic examination

88 Constant representing maximum age at which subject's relatives developed diabetes

14 Constant representing minimum age at which subject's relatives developed diabetes

$20, 50$ Chosen so that
A subject with no relatives would have a DPF value slightly lower than average
The DPF value would decrease relatively slowly as young relatives free of Diabetes joined the database
The DPF value would increase relatively quickly as known relatives developed Diabetes

The value of the DPF increases as the number of relatives who developed diabetes increases, as the age at which those relatives developed diabetes decreases, and as the percentage of genes that they share with the subject increases.

Also the value of the DPF decreases as the number of relatives who never developed diabetes increases, as their ages at their last examination increase, and as the percent of genes that they share with the subject increases.

3.3 Preliminary Analysis

Code for creating the visuals is in Appendix - Section A.[16,17]

3.3.1 Age

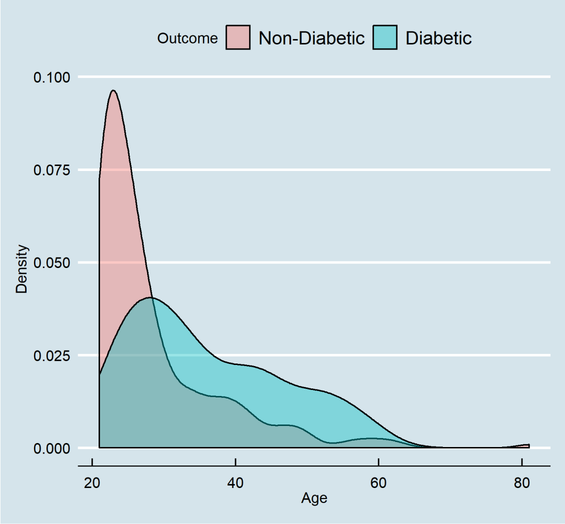


Figure 3.1: Density Plot of Age

Statistic	Value		
	Non-Diabetics	Diabetics	Overall
Mean	28.34	35.93	30.86
Std. Dev.	8.98	10.63	10.20
1^{st} Quantile	22	27.25	23
Median	25	33	27
3^{rd} Quantile	30	43	36
Min	21	21	21
Max	81	60	81

Table 3.1: Summary Statistics of Age

3.3.2 Body Mass Index

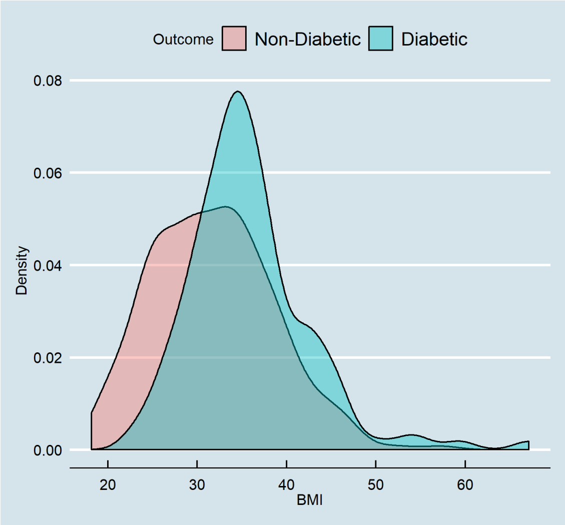


Figure 3.2: Density Plot of Body Mass Index

Statistic	Value		
	Non-Diabetics	Diabetics	Overall
Mean	31.75	35.77	33.08
Std. Dev.	6.79	6.73	7.02
1^{st} Quantile	26.125	31.6	28.4
Median	31.25	34.6	33.2
3^{rd} Quantile	36.1	38.35	37.1
Min	18.2	22.9	18.2
Max	57.3	67.1	67.1

Table 3.2: Summary Statistics of BMI

3.3.3 2-Hr Serum Insulin

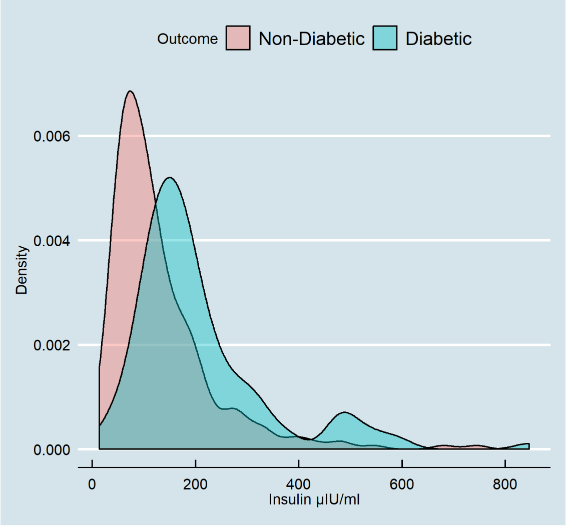


Figure 3.3: Density Plot of 2-Hr Serum Insulin

Statistic	Value		
	Non-Diabetics	Diabetics	Overall
Mean	130.85	206.84	156.05
Std. Dev.	102.62	132.69	118.84
1^{st} Quantile	66	127.5	76.75
Median	105	169.5	125.5
3^{rd} Quantile	163.75	239.25	190
Min	15	14	14
Max	744	846	846

Table 3.3: Summary Statistics of Insulin

3.3.4 Triceps Skin Fold Thickness

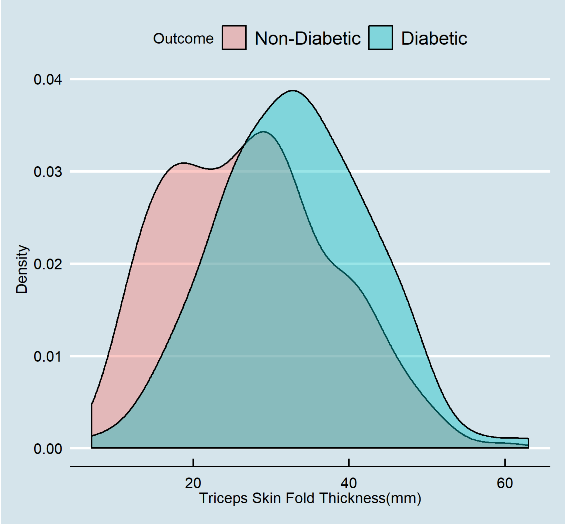


Figure 3.4: Density Plot of Triceps Skin Fold Thickness

Statistic	Value		
	Non-Diabetics	Diabetics	Overall
Mean	27.25	32.96	29.14
Std. Dev.	10.43	9.64	10.51
1^{st} Quantile	18.25	26	21
Median	27	33	29
3^{rd} Quantile	34	39.75	37
Min	7	7	7
Max	60	63	63

Table 3.4: Summary Statistics of Skin Thickness

3.3.5 Diastolic Blood Pressure

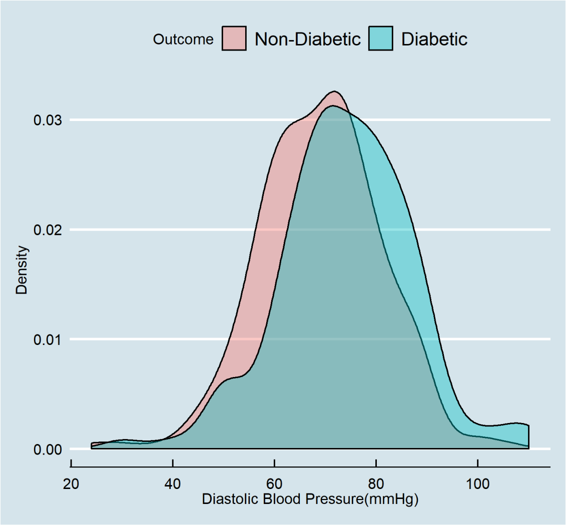


Figure 3.5: Density Plot of Diastolic Blood Pressure

Statistic	Value		
	Non-Diabetics	Diabetics	Overall
Mean	68.96	74.07	70.66
Std. Dev.	11.89	13.02	12.49
1^{st} Quantile	60	66.5	62
Median	70	74	70
3^{rd} Quantile	76	82	78
Min	24	30	24
Max	106	110	110

Table 3.5: Summary Statistics of Blood Pressure

3.3.6 Plasma Glucose Conc. at 2Hrs in OGTT

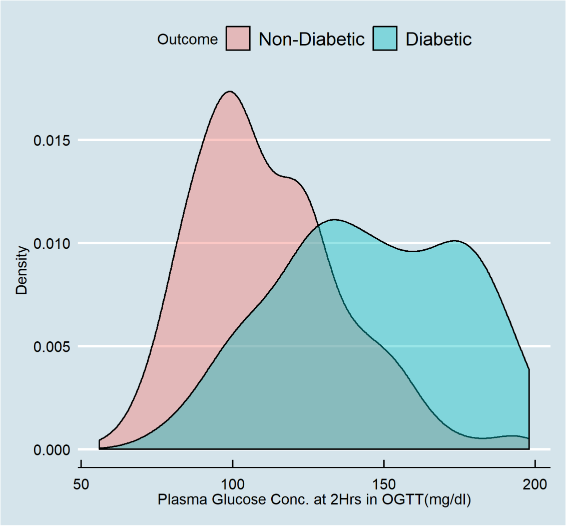


Figure 3.6: Density Plot of Plasma Glucose Conc. at 2Hrs in OGTT

Statistic	Value		
	Non-Diabetics	Diabetics	Overall
Mean	111.43	145.19	122.62
Std. Dev.	24.64	29.83	30.86
1^{st} Quantile	94	124.25	99
Median	107.5	144.5	119
3^{rd} Quantile	126	171.75	143
Min	56	78	56
Max	197	198	198

Table 3.6: Summary Statistics of Glucose

3.3.7 Times Pregnant

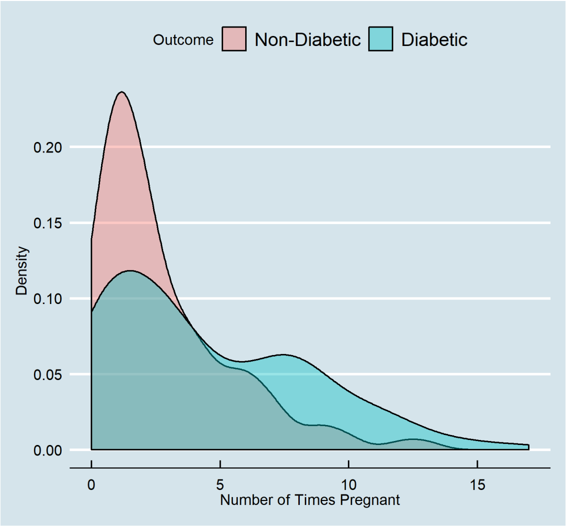


Figure 3.7: Density Plot of Times Pregnant

Statistic	Value		
	Non-Diabetics	Diabetics	Overall
Mean	2.72	4.46	3.3
Std. Dev.	2.61	3.91	3.21
1^{st} Quantile	1	1	1
Median	2	3	2
3^{rd} Quantile	4	7	5
Min	0	0	0
Max	13	17	17

Table 3.7: Summary Statistics of Pregnancies

3.3.8 Diabetes Pedigree Function

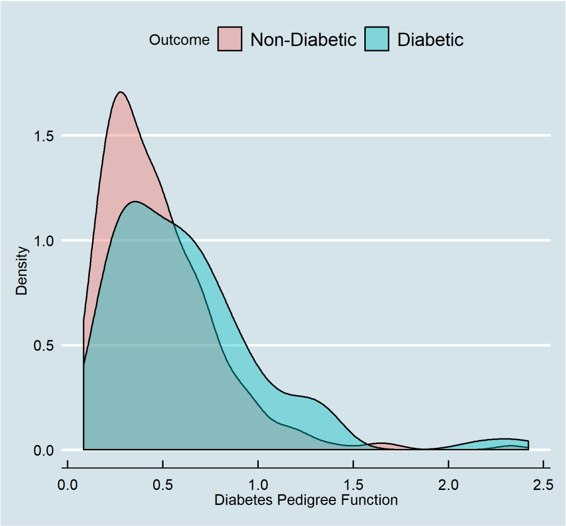


Figure 3.8: Density Plot of Diabetes Pedigree Function

Statistic	Value		
	Non-Diabetics	Diabetics	Overall
Mean	0.47	0.62	0.52
Std. Dev.	0.29	0.40	0.34
1^{st} Quantile	0.261	0.329	0.269
Median	0.413	0.546	0.449
3^{rd} Quantile	0.624	0.786	0.687
Min	0.085	0.127	0.085
Max	2.329	2.42	2.42

Table 3.8: Summary Statistics of Diabetes Pedigree Function

3.3.9 Outcome

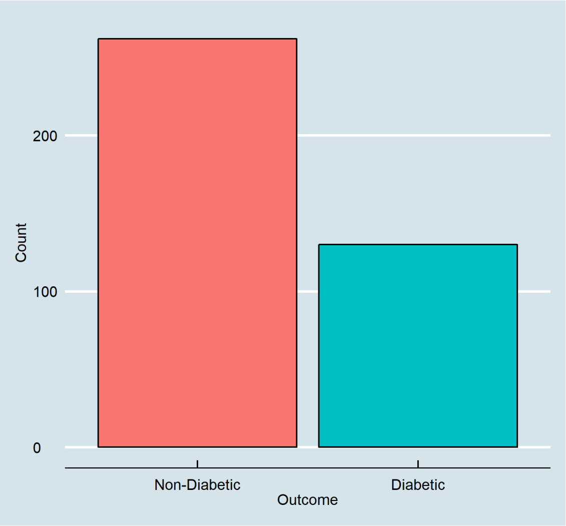


Figure 3.9: Bar Plot of Outcome

Class	Count
Non-Diabetics	262
Diabetics	130

Table 3.9: Summary Statistics of Outcome

3.4 Correlations[13]

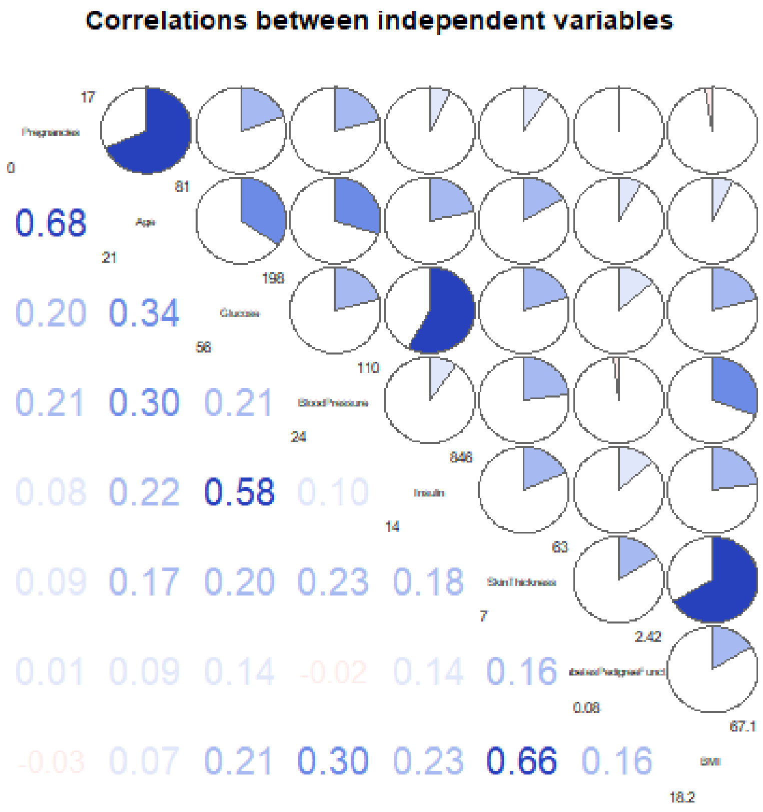


Figure 3.10: Correlation between Independent Variables

The only three significant correlations are between Age and Pregnancies, Glucose level at 2 hrs in OGTT and Insulin levels at the same time and Triceps Skin Fold Thickness and Body Mass Index. All of these are easily explained. The more the age, the more chances for the subject to get pregnant. At 2 hours into an OGTT, diabetics would tend to have elevated glucose and insulin levels whereas no-diabetics would have lower levels for both. Triceps Skin Fold Thickness and Body Mass Index are both essentially measures of obesity.

Chapter 4

Results

All coding has been done in the statistical software, R.[50]

Some basic packages we use are caTools[51] for splitting the dataset into training and test sets, pROC[52] for producing ROC curves and calculating the Area Under the Curve(AUC).

We use the caret[43] package throughout to train models using its trainControl function and to find variable importance using its varImp function

All codes are listed in Appendix - Section A.

4.1 Regular Classification Models

4.1.1 Logistic Regression

Fitting a binomial Generalized Linear Model i.e. Logistic Regression Model to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.833
95% CI	$(0.7319, 0.9082)$
No-Information Rate	0.7051
P-Value [Acc >NIR]	0.0068
Kappa	0.6139
Sensitivity	0.8545
Specificity	0.7826
Balanced Accuracy	0.8186
AUC	0.8839

Table 4.1: Summary Results of Logistic Regression

		Predicted	
		1	0
Observed	1	18	5
	0	8	47

Table 4.2: Confusion Matrix of Logistic Regression

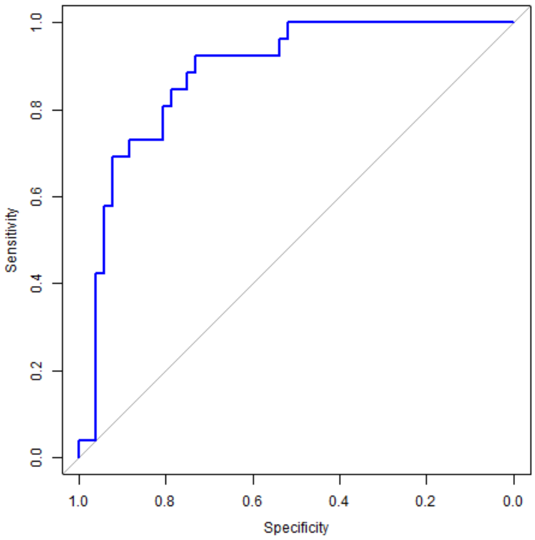


Figure 4.1: ROC for Logistic Regression

Variable	Importance
Glucose	100.000
DPF	64.144
BMI	39.075
Age	27.252
Pregnancies	15.581
Skin Thickness	12.333
Insulin	1.427
Blood Pressure	0.000

Table 4.3: Variable Importance for Logistic Regression

4.1.2 K-Nearest Neighbours

Fitting a K-Nearest Neighbours Model to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.7949
95% CI	$(0.6884, 0.878)$
No-Information Rate	0.7692
P-Value [Acc >NIR]	0.35177
Kappa	0.5
Sensitivity	0.8000
Specificity	0.7778
Balanced Accuracy	0.7889
AUC	0.848

Table 4.4: Summary Results of K-Nearest Neighbours

		Predicted	
		1	0
Observed	1	14	4
	0	12	48

Table 4.5: Confusion Matrix of K-Nearest Neighbours

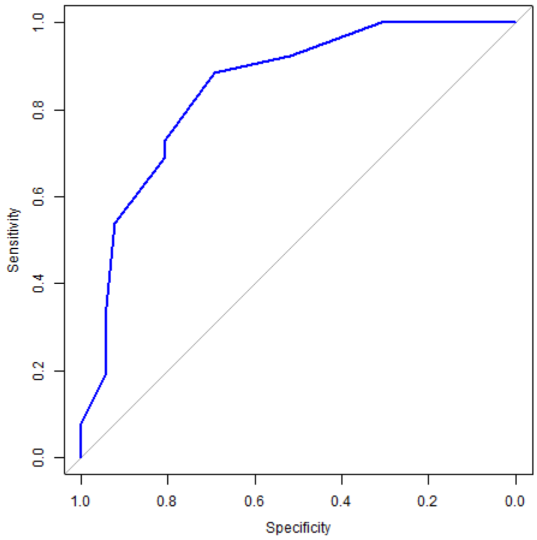


Figure 4.2: ROC for K-Nearest Neighbours

Variable	Importance
Glucose	100.000
Age	74.030
Insulin	66.133
Skin Thickness	37.034
BMI	32.888
DPF	26.669
Pregnancies	5.316
Blood Pressure	0.000

Table 4.6: Variable Importance for K-Nearest Neighbours

4.1.3 Support Vector Machines

We try out two basic types of kernels[53] - linear and radial. The polynomial kernel and the linear kernel had given the same result.

Linear Kernel

Fitting a Support Vector Machines Model with a Linear Kernel to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.8462
95% CI	$(0.7467, 0.9179)$
No-Information Rate	0.7179
P-Value [Acc >NIR]	0.0061
Kappa	0.64
Sensitivity	0.8571
Specificity	0.8182
Balanced Accuracy	0.8377
AUC	0.8824

Table 4.7: Summary Results of SVM-Linear Kernel

		Predicted	
		1	0
Observed	1	18	4
	0	8	48

Table 4.8: Confusion Matrix of SVM-Linear Kernel

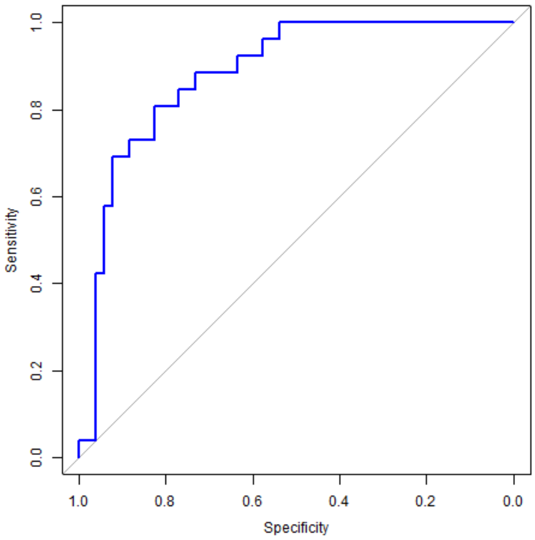


Figure 4.3: ROC for SVM-Linear Kernel

Variable	Importance
Glucose	100.000
Age	74.030
Insulin	66.133
Skin Thickness	37.034
BMI	32.888
DPF	26.669
Pregnancies	5.316
Blood Pressure	0.000

Table 4.9: Variable Importance for SVM-Linear Kernel

Radial Kernel

Fitting a Support Vector Machines Model with a Radial Kernel to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.8333
95% CI	$(0.7319, 0.9082)$
No-Information Rate	0.6795
P-Value [Acc >NIR]	0.0016
Kappa	0.6214
Sensitivity	0.8679
Specificity	0.7600
Balanced Accuracy	0.8140
AUC	0.8898

Table 4.10: Summary Results of SVM-Radial Kernel

		Predicted	
		1	0
Observed	1	19	6
	0	7	46

Table 4.11: Confusion Matrix of SVM-Radial Kernel

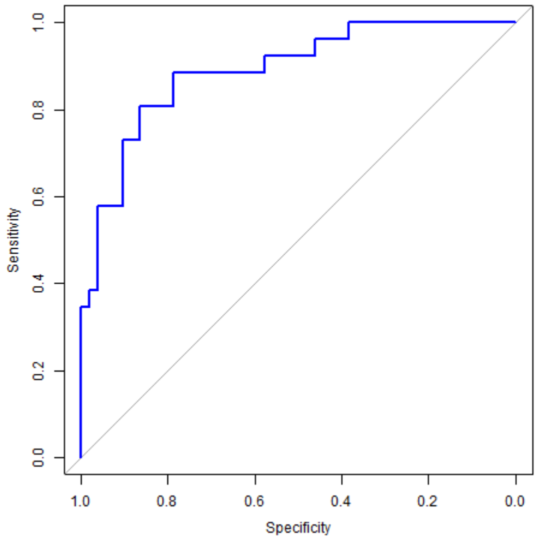


Figure 4.4: ROC for SVM-Radial Kernel

Variable	Importance
Glucose	100.000
Age	74.030
Insulin	66.133
Skin Thickness	37.034
BMI	32.888
DPF	26.669
Pregnancies	5.316
Blood Pressure	0.000

Table 4.12: Variable Importance for SVM-Radial Kernel

4.1.4 Naïve Bayes

Fitting a Naïve Bayes Model[54] to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.7564
95% CI	$(0.646, 0.8465)$
No-Information Rate	0.6538
P-Value [Acc >NIR]	0.03449
Kappa	0.4571
Sensitivity	0.8235
Specificity	0.6266
Balanced Accuracy	0.7266
AUC	0.8476

Table 4.13: Summary Results of Naïve Bayes

		Predicted	
		1	0
Observed	1	17	10
	0	9	42

Table 4.14: Confusion Matrix of Naïve Bayes

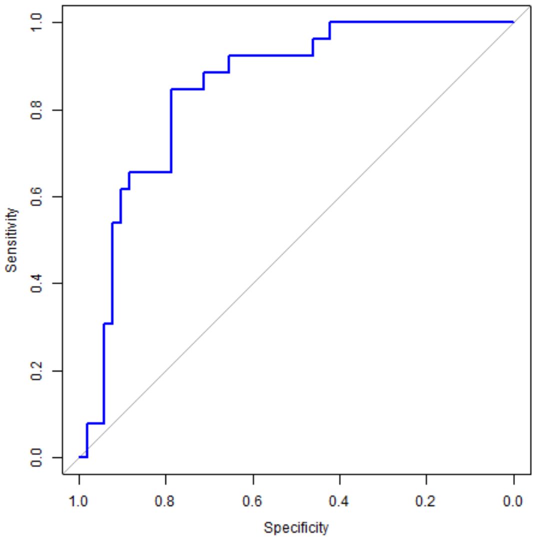


Figure 4.5: ROC for Naïve Bayes

Variable	Importance
Glucose	100.000
Age	74.030
Insulin	66.133
Skin Thickness	37.034
BMI	32.888
DPF	26.669
Pregnancies	5.316
Blood Pressure	0.000

Table 4.15: Variable Importance for Naïve Bayes

4.1.5 Decision Tree

Fitting a Decision Tree Model[55] to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.7308
95% CI	$(0.6124, 0.825)$
No-Information Rate	0.6538
P-Value [Acc >NIR]	0.09351
Kappa	0.4
Sensitivity	0.8039
Specificity	0.5926
Balanced Accuracy	0.6983
AUC	0.8129

Table 4.16: Summary Results of Decision Tree

		Predicted	
		1	0
Observed	1	16	11
	0	10	41

Table 4.17: Confusion Matrix of Decision Tree

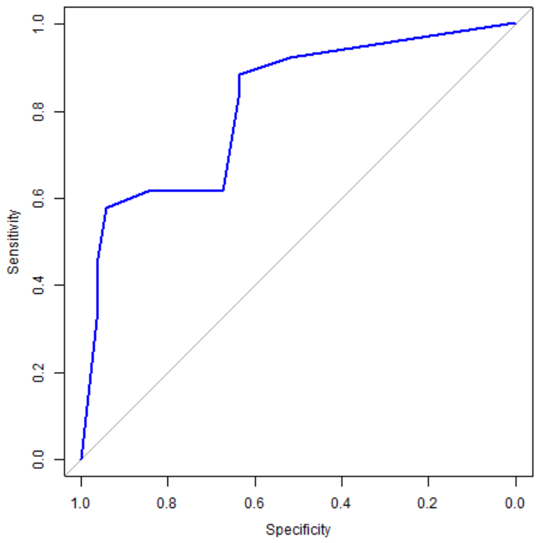


Figure 4.6: ROC for Decision Tree

Variable	Importance
Glucose	100.000
Age	74.030
Insulin	66.133
Skin Thickness	37.034
BMI	32.888
DPF	26.669
Pregnancies	5.316
Blood Pressure	0.000

Table 4.18: Variable Importance for Decision Tree

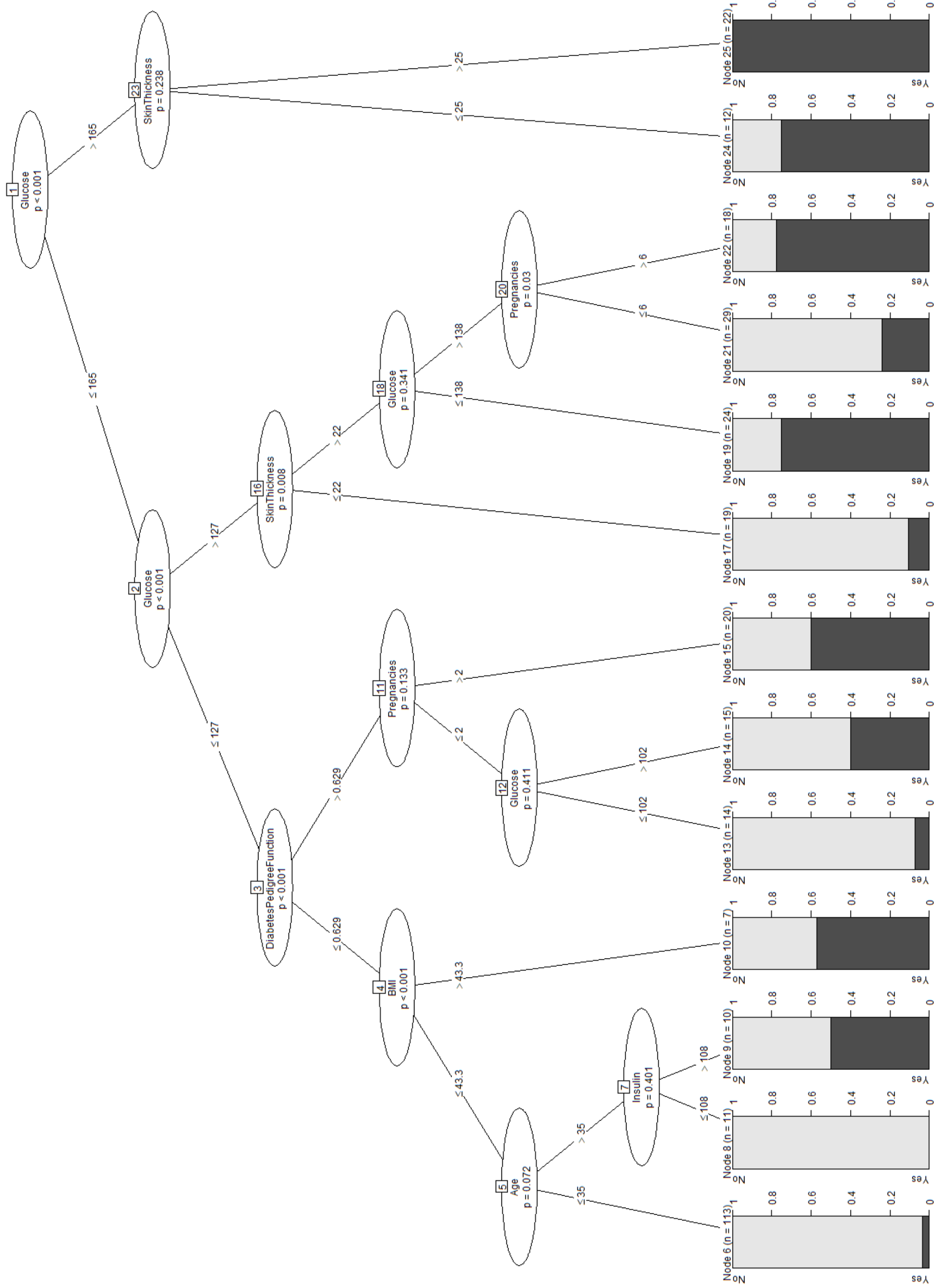


Figure 4.7: Decision Tree

4.1.6 Random Forest

Fitting a Random Forest Model[56] to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.8333
95% CI	$(0.7319, 0.9082)$
No-Information Rate	0.7051
P-Value [Acc >NIR]	0.006897
Kappa	0.6139
Sensitivity	0.8545
Specificity	0.7826
Balanced Accuracy	0.8186
AUC	0.8754

Table 4.19: Summary Results of Random Forest

		Predicted	
		1	0
Observed	1	18	5
	0	8	47

Table 4.20: Confusion Matrix of Random Forest

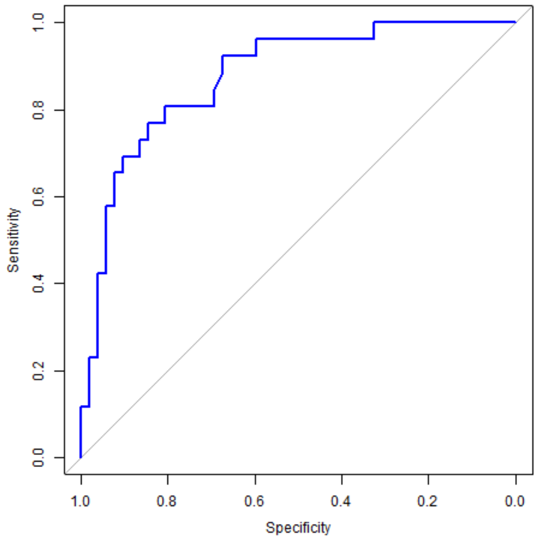


Figure 4.8: ROC for Random Forest

Variable	Importance
Glucose	100.000
Age	48.953
Insulin	47.647
DPF	38.110
BMI	26.654
Skin Thickness	10.856
Pregnancies	0.665
Blood Pressure	0.000

Table 4.21: Variable Importance for Random Forest

4.2 Boosted Models

4.2.1 AdaBoost Classification Trees

Fitting a Bagged AdaBoost Model[57] to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.7821
95% CI	(0.6741, 0.8676)
No-Information Rate	0.6538
P-Value [Acc >NIR]	0.0099
Kappa	0.5143
Sensitivity	0.8431
Specificity	0.6667
Balanced Accuracy	0.7549
AUC	0.8536

Table 4.22: Summary Results of AdaBoost Classification Trees

		Predicted	
		1	0
Observed	1	18	9
	0	8	43

Table 4.23: Confusion Matrix of AdaBoost Classification Trees

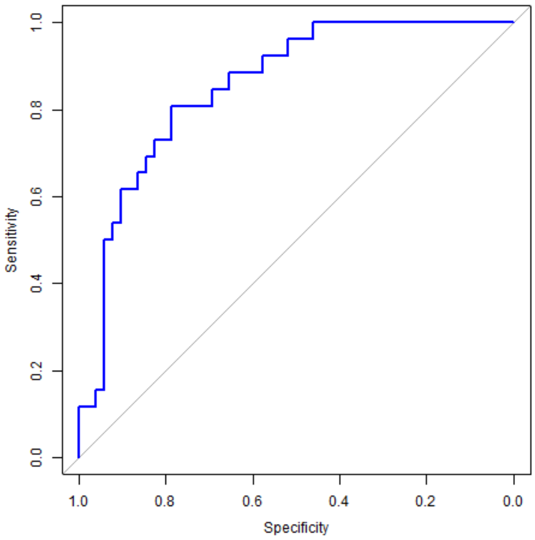


Figure 4.9: ROC for AdaBoost Classification Trees

Variable	Importance
Glucose	100.000
Age	74.030
Insulin	66.133
Skin Thickness	37.034
BMI	32.888
DPF	26.669
Pregnancies	5.316
Blood Pressure	0.000

Table 4.24: Variable Importance for AdaBoost Classification Trees

4.2.2 eXtreme Gradient Boosting-Linear

Fitting an eXtreme Gradient Boosting-Linear Model[58] to the training data containing 314 observations(80%), we get the following results-

Metric	Value
Accuracy	0.8462
95% CI	(0.7467, 0.9179)
No-Information Rate	0.6923
P-Value [Acc >NIR]	0.001457
Kappa	0.6471
Sensitivity	0.8704
Specificity	0.7917
Balanced Accuracy	0.8310
AUC	0.8632

Table 4.25: Summary Results of eXtreme Gradient Boosting-Linear

		Predicted	
		1	0
Observed	1	19	5
	0	7	47

Table 4.26: Confusion Matrix of eXtreme Gradient Boosting-Linear

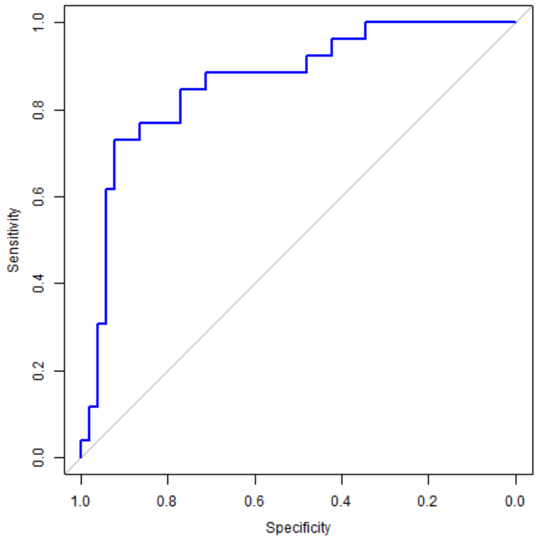


Figure 4.10: ROC for eXtreme Gradient Boosting-Linear

Variable	Importance
Glucose	100.000
Age	34.405
BMI	29.444
DPF	22.032
Insulin	19.187
Blood Pressure	6.383
Skin Thickness	4.385
Pregnancies	0.000

Table 4.27: Variable Importance for eXtreme Gradient Boosting-Linear

4.3 Comparison of Models

4.3.1 In terms of Accuracy

In terms of accuracy, comparing all models together we get the following table.

The best values for each column are coloured in green.

Model	Accuracy	Kappa	Sensitivity	Specificity	Balanced Accuracy	AUC
Logistic	0.833	0.6139	0.8545	0.786	0.8186	0.8839
K-NN	0.7949	0.5	0.8	0.7778	0.7889	0.848
SVM-L	0.8462	0.64	0.8571	0.8182	0.8377	0.8824
SVM-R	0.8333	0.6214	0.8679	0.76	0.814	0.8898
Naïve Bayes	0.7564	0.4571	0.8235	0.6266	0.7266	0.8476
Decision Tree	0.7308	0.4	0.8039	0.5926	0.6983	0.8129
Random Forest	0.8333	0.6139	0.8545	0.7826	0.8186	0.8754
AdaBoost	0.7821	0.5143	0.8431	0.6667	0.7549	0.8536
XGB-Linear	0.8462	0.6471	0.8704	0.7917	0.831	0.8632

Table 4.28: Comparison of Models in terms of Accuracy

Thus, the best models were given by eXtreme Gradient Boosting - Linear and Support Vector Machines - Linear.

While Support Vector Machines - Radial(AUC = 0.8898) has a slight advantage(0.0074) in the AUC metric over the closest other value depicted by Support Vector Machines - Linear (AUC = 0.8824), it fails by a larger margin in other metrics, and is thus not chosen as one of the best models in terms of accuracy for the classification.

4.3.2 In terms of Variable Importance

Model	Age	BMI	Insulin	Skin Thickness	Blood Pressure	Glucose	Pregnancies	DPF
Logistic	27.25	39.07	1.42	12.33	0.00	100.0	15.58	64.14
K-NN	74.03	32.88	66.13	37.03	0.00	100.0	5.31	26.66
SVM-L	74.03	32.88	66.13	37.03	0.00	100.0	5.31	26.66
SVM-R	74.03	32.88	66.13	37.03	0.00	100.0	5.31	26.66
Naïve Bayes	74.03	32.88	66.13	37.03	0.00	100.0	5.31	26.66
Decision Tree	74.03	32.88	66.13	37.03	0.00	100.0	5.31	26.66
Random Forest	48.95	26.65	47.64	10.85	0.00	100.0	0.66	38.11
Ada-Boost	74.03	32.88	66.13	37.03	0.00	100.0	5.31	26.66
XGB-Linear	34.40	29.44	19.18	4.38	6.38	100.0	0.00	22.03

Table 4.29: Comparison of Models in terms of Variable Importance

Therefore, we see that Glucose is the most important classification criteria in all models. Age appears to be a consensus second while Insulin is third. BMI and Skin Thickness appear to fight for the fourth position. Diastolic Blood Pressure seems to be the most inconsequential.

Part II

Linking Diabetes' Indicators to Associated Medical Cost

Chapter 5

Generalized Linear Models and Generalized Additive Models

5.1 Generalized Linear Models(GLMs)[14]

5.1.1 Introduction

GLMs[59] are a general set of models that can be used to assess & quantify the relationship between a dependent variable and a set of independent variables. GLMs differ from ordinary linear regression modelling in two aspects -

- The distribution of the dependent variable is chosen to be from an exponential family.
- A transformation of the mean of the dependent variable is linearly related to the independent variables.

If the distribution of the dependent variable is from the exponential family, it allows the dependent variable to be heteroskedastic i.e. the variance is allowed to vary with the mean which varies with the independent variables.

5.1.2 The Model

If the dependent variable is y, the GLM is given by -

$$f(y) = c(y, \phi)e^{\frac{y\theta - a(\theta)}{\phi}} \tag{5.1}$$

$$g(\mu) = \mathbf{X}^T\beta \tag{5.2}$$

One can write popular probability distributions in the exponential form as given -

Distribution	θ	a(θ)	ϕ	E(y)	V(μ) = $\frac{Var(y)}{\phi}$
$\mathbf{B}$(n,p)	$\ln\frac{p}{1-p}$	$n\ln 1+e^{\theta}$	1	np	$np(1-p)$
$\mathbf{P}(\mu)$	$\ln\mu$	e^{θ}	1	μ	μ
$\mathbf{N}(\mu,\sigma^2)$	μ	$\frac{1}{2}\theta^2$	σ^2	μ	1
$\mathbf{G}(\mu,\nu)$	$-\frac{1}{\mu}$	$-\ln -\theta$	$\frac{1}{\nu}$	μ	μ^2
$\mathbf{IG}(\mu,\sigma^2)$	$-\frac{1}{2\mu^2}$	$-\sqrt{-2\theta}$	σ^2	μ	μ^3
$\mathbf{NB}(\mu,\kappa)$	$\ln\frac{\kappa\mu}{1+\kappa\mu}$	$-\frac{1}{\kappa}\ln 1-\kappa e^{\theta}$	1	μ	$\mu(1+\kappa\mu)$

Table 5.1: Exponential Forms of Popular Distributions

Equation (5.1) describes the distribution of the dependent variable in the exponential family canonical form. Equation(5.2) describes the transformation of the mean to be linearly related to the independent variables in $\mathbf{X}$.

The form of $a(\theta)$ determines the exact distribution of the exponentially distributed dependent variable.

The form of the link function, $g(\mu)$ describes how the mean of the dependent variable is linked to the independent variables. g needs to be a monotonic and differentiable function, such as a log function or square root.

Observations of y are assumed to be independent.

These equations work in the following fashion, given $\mathbf{X}$, one can determine μ from $g(\mu)$. Then one can determine θ via $\dot{a}(\theta) = \mu$. And now, given θ, y can be determined.

The word ”linear” in GLM refers to the linearity of β and not $\mathbf{X}$. Therefore it is known as linear, because the coefficients of the model are linear.

5.1.3 Procedure of Generalized Linear Modelling

- A distribution $f(y)$ and $a(\theta)$ is chosen as in (3.1). This distribution chosen is customized to the situation under consideration.

- A link function, $g(\mu)$ is chosen. To simplify matters, one may choose the ”canonical” link function corresponding to the different types of dependent variable distributions, $f(y)$.

- The independent variables, $\mathbf{X}$ are then chosen, in terms of which $g(\mu)$ is to be modelled.

- The model is fit to our training set data by estimating β and ϕ. The fitting is done using Maximum Likelihood Estimation.

- Prediction of the dependent variable values for our test set data is done and residuals are checked.

5.1.4 The Link function

Canonical link functions are given in the table below.

The link function is canonical if $g(\mu) = \theta = \mathbf{X}^T\beta$ corresponding to $a(\theta)$.

Link function	$g(\mu)$	Canonical Link for
identity	μ	Normal Distribution
log	$\ln \mu$	Poisson Distribution
power	μ^p	Gamma(p=-1) Inverse Gaussian(p=-2)
square root	$\sqrt{\mu}$	
logit	$\ln \frac{\mu}{1-\mu}$	binomial

Table 5.2: Link Functions for Popular Distributions

5.1.5 Maximum Likelihood Estimation

The MLE for β and ϕ can be derived by maximizing the log-likelihood function given by -

$$l(\beta, \phi) = \sum_{i=1}^{n} \ln f(y_i; \beta, \phi) = \sum_{i=1}^{n} \{\ln c(y_i, \phi) + \frac{y_i\theta_i - a(\theta_i)}{\phi}\} \tag{5.3}$$

which again assumes independent exponential family responses, y.

To find the maximum, Equation (5.3) is differentiated with respect to the parameters and then the resulting equation is set to zero.

5.1.6 Assessing Fit of the Model

The best possible fit is obtained when the model is saturated, with the number of parameters equal to the number of observations. The saturated log-likelihood is

$$\check{l} = \sum_{i=1}^{n} \{\ln c(y_i, \phi) + \frac{y_i\check{\theta}_i - a(\check{\theta}_i)}{\phi}\} \tag{5.4}$$

which is also the maximum possible log-likelihood value for y, given $a(\theta)$.

The value obtained from (5.4) is compared to $\hat{l}$, which is the maximum of the log-likelihood value based on y and the given independent variables.

Deviance Δ is defined as the distance between the saturated model and fitted model, given by -

$$\Delta \equiv 2(\check{l} - \hat{l}) \tag{5.5}$$

Therefore, a large deviance indicates a poor fit.

The size of Δ is assessed relative to the χ^2_{n-p} distribution.[60]

5.2 Generalized Additive Models(GAMs)[15]

5.2.1 Introduction

GAMs[61] extends GLMs by including a sum of smooth functions of the covariates. The general model structure for i observations, is given by

$$g(\mu_i) = \mathbf{X}_i^T\theta + f_1(x_{1i}) + f_2(x_{2i}) + f_3(x_{3i}, x_{4i}) + ... \tag{5.6}$$

where $\mu_i = E(Y_i)$ and Y_i's follow a distribution belonging to the Exponential Family.

Here, Y_i is the dependent variable, $\mathbf{X}_i^T$ is a row vector representing strictly parametric independent variables, θ represents the corresponding parameter vector and the f_j's are smooth functions of the smoothed independent variables.

Thus, the model provides considerable flexibility but the flexibility has a cost of two problems - We need to represent the smooth functions in some way and choose how smooth they should be.

For simplicity, we consider only a simple model with upto 2 univariate smooth components.

5.2.2 Univariate Smooth Functions

Let us consider a model with only one smooth function of a covariate -

$$y_i = f(x_i) + \epsilon_i \tag{5.7}$$

where, y_i is the dependent variable, x_i is the independent variable, f is the smooth function and ϵ_i are independent and identically distributed $\mathbf{N}(0, \sigma^2)$ random variables. We assume that x_i lies in the interval $[0, 1]$

5.2.3 Regression Splines

For model to be linear, a basis (space of functions) containing the f is chosen such that the i^{th} basis function is $b_i(x)$ (assumed to be known), $i = 1, 2....q$. Then,

$$f(x) = \sum_{i=1}^{q} b_i(x)\beta_i \tag{5.8}$$

for some unknown parameter β_i

5.2.4 Cubic Splines

A cubic spline is a curve made by joining sections of a cubic polynomial joined so that the resulting function is continuous and has continuous first and second derivatives.

Points of joining are known as knots, which must be chosen. Mostly, the knots are chosen to be at evenly spaced points in the range of x values.

5.2.5 Controlling Smoothing

To control smoothing in the model, the basis dimension is kept constant, at a size larger than is believed to be required so that the smoothing can be controlled by adding a penalty to the least squares fitting objective. So instead of minimizing

$$\parallel y - \mathbf{X}\beta \parallel^2 \tag{5.9}$$

we minimize

$$\parallel y - \mathbf{X}\beta \parallel^2 + \lambda \int_0^1 [f''(x)]^2 dx \tag{5.10}$$

where the second term, representing the integrated square of the second derivative penalizes models that are too wobbly. This trade off between model fitting and model smoothing is determined by the value of the smoothing parameter, λ. A straight line is obtained if $\lambda \to \infty$ and an unpenalized regression spline estimate is obtained if $\lambda = 0$.

Because f is linear in the parameters, β_i, one can write the penalty as a quadratic equation in β

$$\int_0^1 [f''(x)]^2 dx = \beta^{\mathbf{T}} \mathbf{S} \beta \tag{5.11}$$

where S is a matrix of known coefficients. So our problem is now to minimize

$$\parallel y - \mathbf{X}\beta \parallel^2 + \lambda \beta^{\mathbf{T}} \mathbf{S} \beta \tag{5.12}$$

with respect to β.

It can be shown that minimizing (5.11), results in

$$\hat{\beta} = (\mathbf{X}^{\mathbf{T}}\mathbf{X} + \lambda \mathbf{S})^{-1} \mathbf{X}^{\mathbf{T}} y \tag{5.13}$$

Chapter 6

Data and Preliminary Analysis

6.1 Provenance

The data were collected in two stretches from 26^{th} December 2018 to 2^{nd} January 2019 and from 18^{th} March 2019 to 23^{rd} March 2019. The data was collected at the Diabetology department at GD Hospital & Diabetes Institute located at 139 A, Lenin Sarani, Bowbazar, Kolkata, West Bengal-700013. The data was collected via face-to-face interviews with the patients during the OPD hours between 10am and 1pm from Monday to Saturday.

6.2 Survey Methodology

We did not have access to trained medical personnel for the exercise and thus had to rely on test reports and other measures which could be recorded without medical training.

Eye complications of diabetes such as Retinopathy, Glaucoma, Cataracts and Blindness can lead to high costs of care.[62] Similrly, kidney complications of diabetes such as Renal insufficiency or Kidney failure can lead to high costs of care or even death.[62]

Diabetic patients often have non-healing wounds[63] due to neuropathy, vascular problems or other complications. These can eventually lead to infections, gangrene and even result in amputation.

All patients were asked to indicate if they had any eye complications, kidney complications or any non-healing wounds.

Based on our own classification analysis, Age and number of pregnancies data was also recorded.

In terms of glucose tests, determination of glycated haemoglobin and fasting plasma glucose concentrations alone is an acceptable alternative to measuring glucose concentration two hours after challenge with 75 g glucose for the diagnosis of diabetes.[64] Thus, HbA1c (Glycated Haemogobin) levels along with both Fasting Plasma Glucose concentration and Post-Prandial Glucose concentrations were recorded.

HDL and LDL Cholesterol levels are also recorded so as to asses the cardiovascular status of the patients.

Sex of the patient is recorded as a further segmentation of the dataset.

Serum Creatinine levels, Albumin/Creatinine ratio were recorded which indicate severity of kidney disease, if any.

Additionally, details of Alanine Transaminase (ALT), Aspartate Aminotransferase (AST), Alkaline Phosphatase (ALP), Albumin/Globulin ratio and Gamma GT were recorded- all of which are indicators of potential liver disease.

If the patient was currently prescribed insulin, this was recorded. The patients were enquired as to their annual expenditure on diabetes and if any recent major hospital spending had been made by them, alongwith insurance information on the same.

The survey sheet is provided in Appendix - Section D

6.3 Collected Data

Being a low-cost clinic, GD Hospital & Diabetes Institute attracts diabetic patients in the lower socio-economic strata. These patients usually do not have detailed medical tests such as Liver function tests and kidney tests done and therefore do not possess that data.

Thus the variables relating to Liver Function test and Kidney function tests were removed.

Additionally, people do not possess diabetes insurance and therefore those questions had to be disregarded in the final analysis as well. No patients with non-healing wounds were encountered and therefore the variable was removed as it cannot be incorporated into any model.

Finally, the data of the following variables -

1. Eye Complications (Y/N)

2. Kidney Complications (Y/N)

3. Age (in years as of 1^{st} Jan 2019)
4. Sex (M/F)
5. Height (in cms)
6. Weight (in kgs)
7. Body Mass Index
8. Number of Pregnancies
9. HbA1C level (in % terms)
10. Fasting Plasma Glucose Concentration (mg/dl)
11. Post-Prandial Glucose Concentration (mg/dl)
12. Blood Pressure Systolic
13. Blood Pressure Diastolic
14. HDL Cholesterol (mg/dl)
15. LDL Cholesterol (mg/dl)
16. Insulin Prescribed (Y/N)
17. Annual Spending on Diabetes (in INR)

A total of 44 records were collected but 3 records were discarded due to having more than 3 fields missing.

6.4 Missing Data

No. of records	No. of Missing fields	Missing Fields
16	NIL	-
6	ONE	HbA1C level in 3 records HDL Cholesterol in 2 records FPGC in 1 record
11	TWO	Systolic and Diastolic BP in 1 record HDL and LDL Cholesterol in 10 records
8	THREE	HbA1C, HDL and LDL Levels in 6 records PP Glucose, HDL and LDL levels in 2 records

Table 6.1: Details of Missing Data

6.4.1 Dealing with Missing Data

The missing data in the 41 records were estimated using a Random Forest[56] Regression algorithm trained using 10-fold cross validation.

The R[50] code for this is listed in Appendix - Section B.

6.5 Preliminary Analysis[16–18]

The t-test carried out in the following tables is testing for significant differences in the parameter values grouped by Sex.

6.5.1 Eye Complications

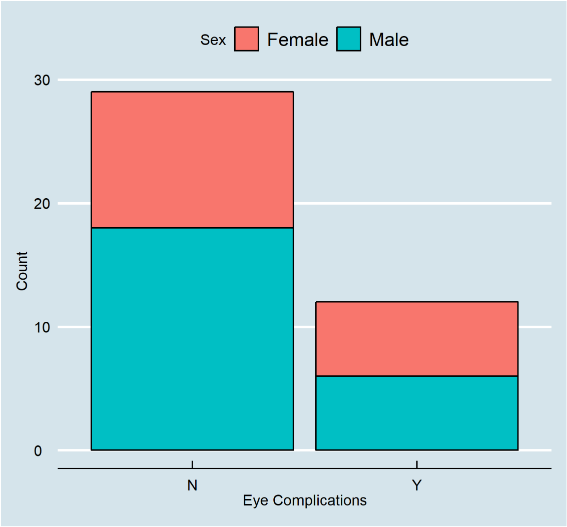
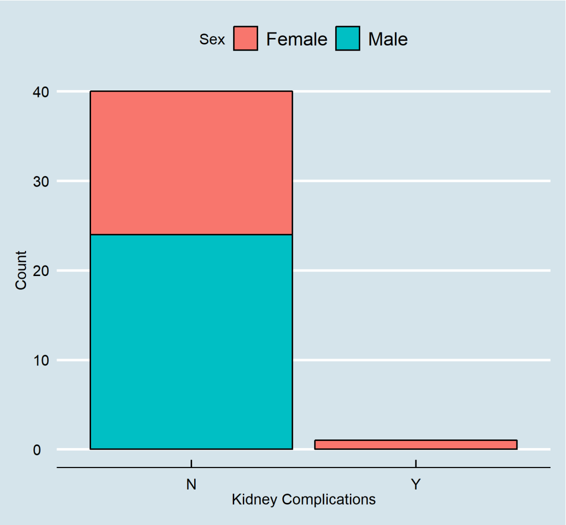


Figure 6.1: Bar Plot of Eye Complications

Statistic	Value		
	Males	Females	Overall
Yes	6	6	12
No	18	11	29

Table 6.2: Summary Statistics of Eye Complications

6.5.2 Kidney Complications

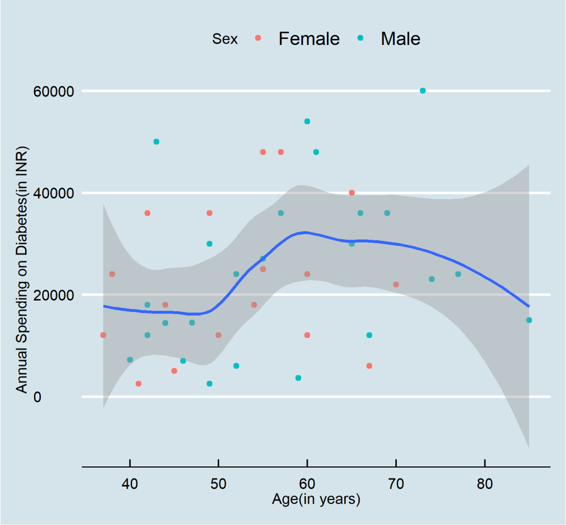


Statistic	Value		
	Males	Females	Overall
Yes	0	1	1
No	24	16	40

Figure 6.2: Bar Plot of Kidney Complications

Table 6.3: Summary Statistics of Kidney Complications

6.5.3 Age

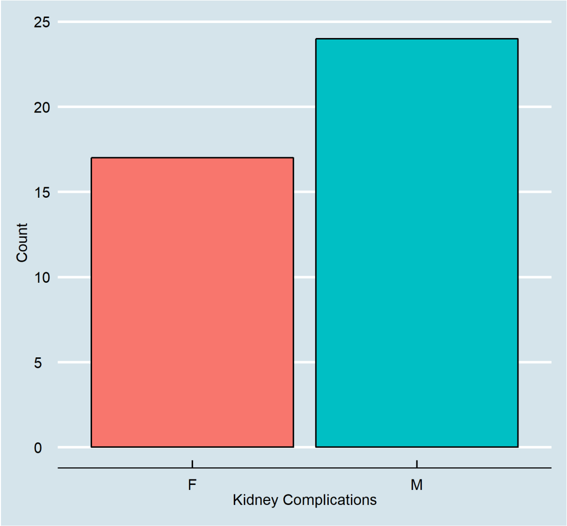


Statistic	Value		
	Males	Females	Overall
Mean	57.25	52.29	55.19
Std. Dev.	12.60	10.18	11.78
1^{st} Quantile	46.75	44	45
Median	56	54	55
3^{rd} Quantile	66.25	60	65
Min	40	37	37
Max	85	70	85
	t-value	df	p-value
by Sex	1.38	38.23	0.1727

Figure 6.3: Scatter Plot of Age

Table 6.4: Summary Statistics of Age

6.5.4 Sex

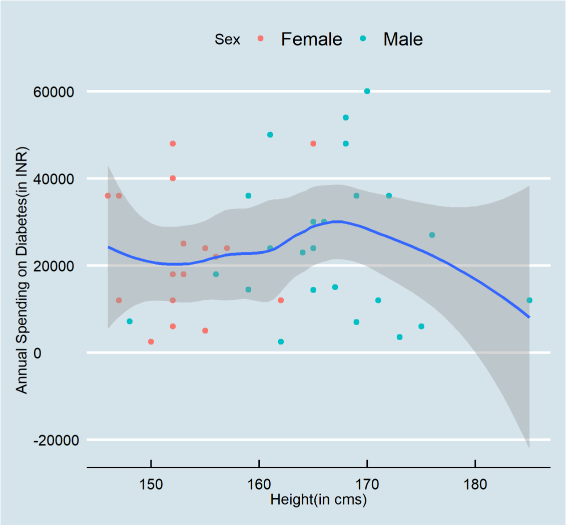


Statistic	Value		
	Males	Females	Overall
Count	24	17	41

Figure 6.4: Bar Plot of Sex

Table 6.5: Summary Statistics of Sex

6.5.5 Height

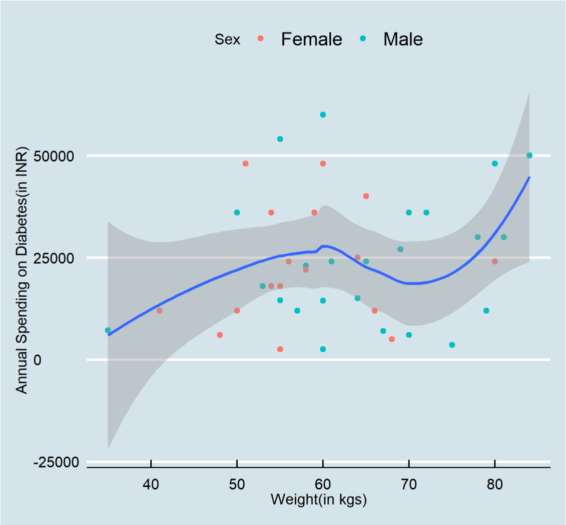


Statistic	Value		
	Males	Females	Overall
Mean	166.41	153.29	160.97
Std. Dev.	4.94	7.47	9.20
1^{st} Quantile	161.75	152	153
Median	166.5	152	161
3^{rd} Quantile	170.25	155	168
Min	148	146	146
Max	185	165	165
	t-value	df	p-value
by Sex	6.75	38.87	$4.66\text{x}10^{-8}$

Figure 6.5: Scatter Plot of Height

Table 6.6: Summary Statistics of Height

6.5.6 Weight

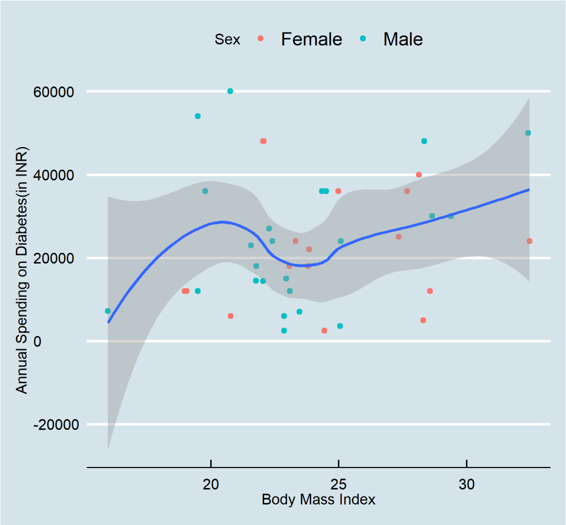


Statistic	Value		
	Males	Females	Overall
Mean	64.91	57.88	62
Std. Dev.	11.60	8.99	11.05
1^{st} Quantile	57.75	54	55
Median	64.5	56	60
3^{rd} Quantile	72.75	64	69
Min	35	41	35
Max	80	84	84
	t-value	df	p-value
by Sex	2.18	38.62	0.0350

Figure 6.6: Scatter Plot of Weight

Table 6.7: Summary Statistics of Weight

6.5.7 Body Mass Index



Statistic	Value		
	Males	Females	Overall
Mean	23.34	24.63	23.88
Std. Dev.	3.58	3.67	3.63
1^{st} Quantile	21.70	22.07	21.77
Median	22.85	23.83	23.08
3^{rd} Quantile	24.64	27.67	25.07
Min	15.97	18.97	15.97
Max	32.40	32.45	32.45
	t-value	df	p-value
by Sex	-1.19	34.093	0.2706

Figure 6.7: Scatter Plot of Body Mass Index

Table 6.8: Summary Statistics of Body Mass Index

6.5.8 Number of Pregnancies

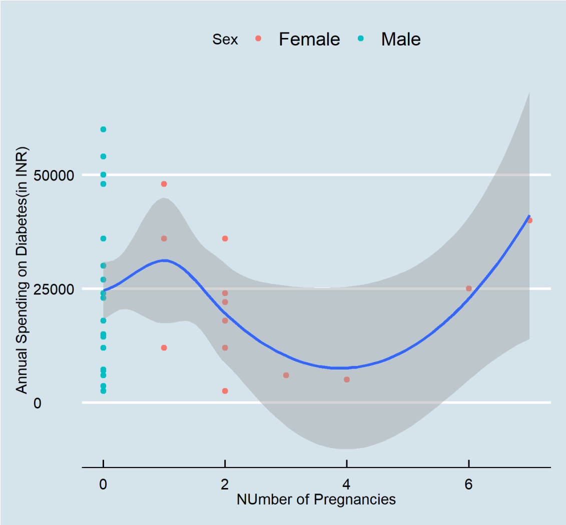


Figure 6.8: Bar Plot of Number of Pregnancies

Statistic	Value
	Females
Mean	2.41
Std. Dev.	1.73
1^{st} Quantile	1
Median	2
3^{rd} Quantile	2
Min	1
Max	7

Table 6.9: Summary Statistics of Number of Pregnancies

6.5.9 HbA1C level

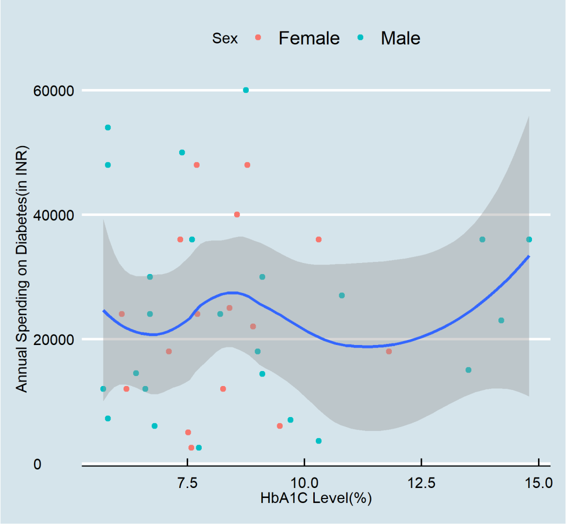
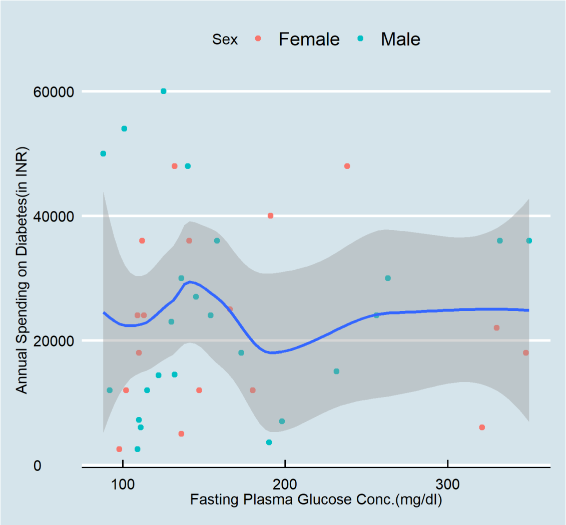


Figure 6.9: Scatter Plot of HbA1C Level

Statistic	Value		
	Males	Females	Overall
Mean	8.76	8.08	8.48
Std. Dev.	2.83	1.53	2.38
1^{st} Quantile	6.67	7.35	6.7
Median	7.97	7.71	7.74
3^{rd} Quantile	9.85	8.78	9.1
Min	5.7	5.7	5.7
Max	14.8	11.8	14.8
	t-value	df	p-value
by Sex	0.98	36.90	0.3312

Table 6.10: Summary Statistics of HbA1C Level

6.5.10 Fasting Plasma Glucose Concentration

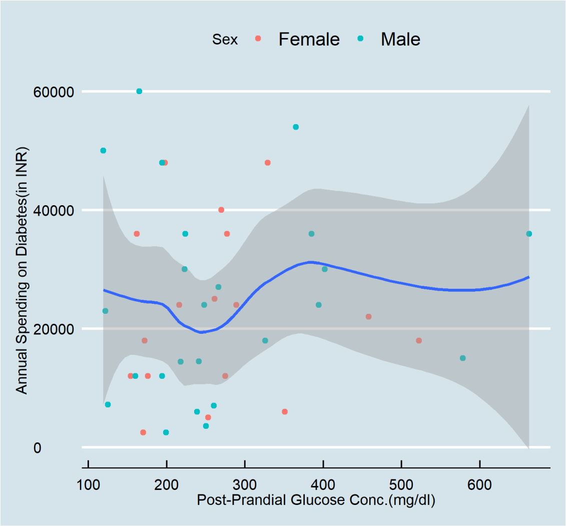


Statistic	Value		
	Males	Females	Overall
Mean	165.06	174.94	169.15
Std. Dev.	72.51	83.87	76.56
1^{st} Quantile	114	112	112
Median	138	141	140
3^{rd} Quantile	192	191	191
Min	88	98	88
Max	350	348	350
	t-value	df	p-value
by Sex	−0.39	31.31	0.6973

Figure 6.10: Scatter Plot of Fasting Plasma Glucose Conc.

Table 6.11: Summary Statistics of Fasting Plasma Glucose Conc.

6.5.11 Post-Prandial Glucose Concentration



Statistic	Value		
	Males	Females	Overall
Mean	273.34	266.64	270.56
Std. Dev.	135.53	103.42	121.87
1^{st} Quantile	194	176	194
Median	240	261	248
3^{rd} Quantile	335.75	289	326
Min	119	154	119
Max	663	522	663
	t-value	df	p-value
by Sex	0.17	38.73	0.8586

Figure 6.11: Scatter Plot of Post-Prandial Glucose Conc.

Table 6.12: Summary Statistics of Post-Prandial Glucose Conc.

6.5.12 Blood Pressure Systolic

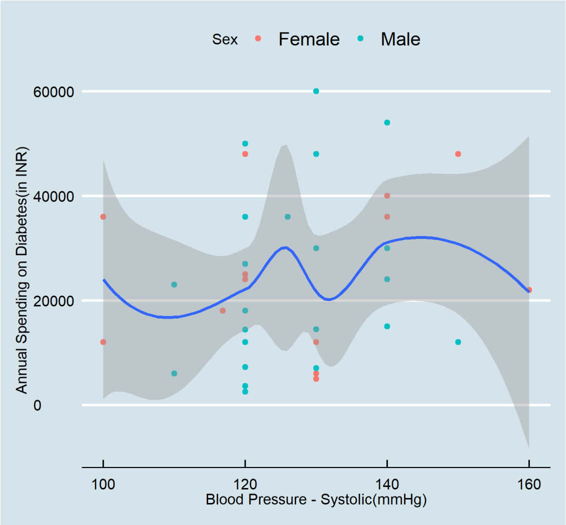


Figure 6.12: Scatter Plot of Systolic Blood Pressure

Statistic	Value		
	Males	Females	Overall
Mean	126.91	126.28	126.65
Std. Dev.	10.40	15.48	12.58
1^{st} Quantile	120	120	120
Median	123	120	120
3^{rd} Quantile	132.5	130	130
Min	110	100	100
Max	150	160	160
	t-value	df	p-value
by Sex	0.14	26.00	0.8846

Table 6.13: Summary Statistics of Systolic Blood Pressure

6.5.13 Blood Pressure Diastolic

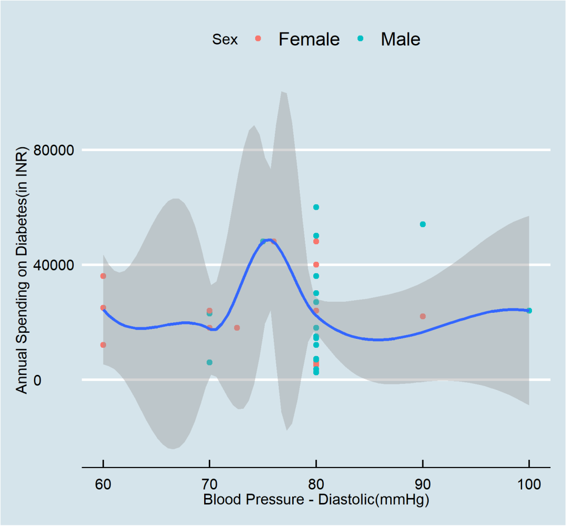


Figure 6.13: Scatter Plot of Diastolic Blood Pressure

Statistic	Value		
	Males	Females	Overall
Mean	80.20	75.25	78.13
Std. Dev.	5.61	8.59	7.37
1^{st} Quantile	80	70	80
Median	80	70	80
3^{rd} Quantile	80	80	80
Min	70	60	60
Max	100	90	100
	t-value	df	p-value
by Sex	2.10	25.50	0.04

Table 6.14: Summary Statistics of Diastolic Blood Pressure

6.5.14 HDL Cholesterol

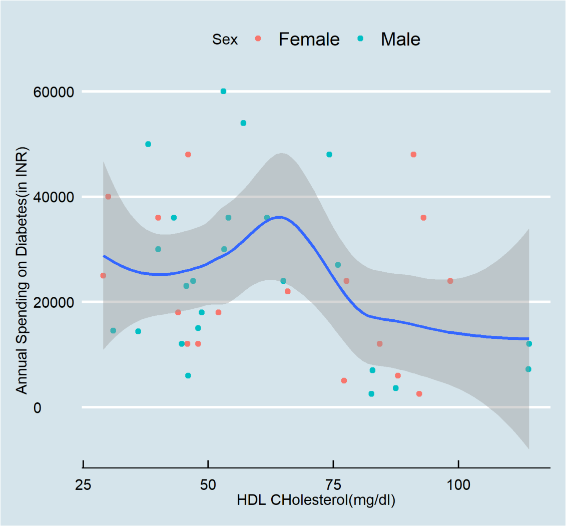


Figure 6.14: Scatter Plot of HDL Cholesterol

Statistic	Value		
	Males	Females	Overall
Mean	60.12	64.83	62.08
Std. Dev.	22.76	24.10	23.14
1^{st} Quantile	45.39	45.8	45.62
Median	53.09	65.84	53.18
3^{rd} Quantile	74.61	87.90	82.59
Min	31	29	29
Max	114.14	98.37	114.14
	t-value	df	p-value
by Sex	-0.63	33.41	0.5324

Table 6.15: Summary Statistics of HDL Cholesterol

6.5.15 LDL Cholesterol

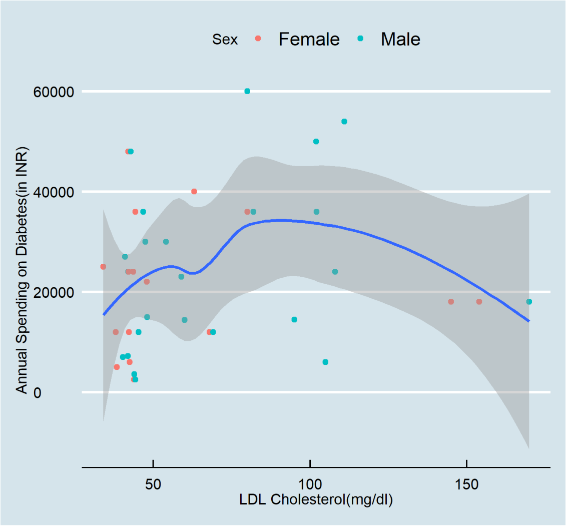


Figure 6.15: Scatter Plot of LDL Cholesterol

Statistic	Value		
	Males	Females	Overall
Mean	70.03	59.49	65.66
Std. Dev.	33.14	35.92	34.29
1^{st} Quantile	44.19	42.15	42.49
Median	56.5	43.58	47.84
3^{rd} Quantile	96.75	63	80
Min	40.32	34	34
Max	170	154	170
	t-value	df	p-value
by Sex	0.95	32.81	0.3462

Table 6.16: Summary Statistics of LDL Cholesterol

6.5.16 Insulin Prescribed

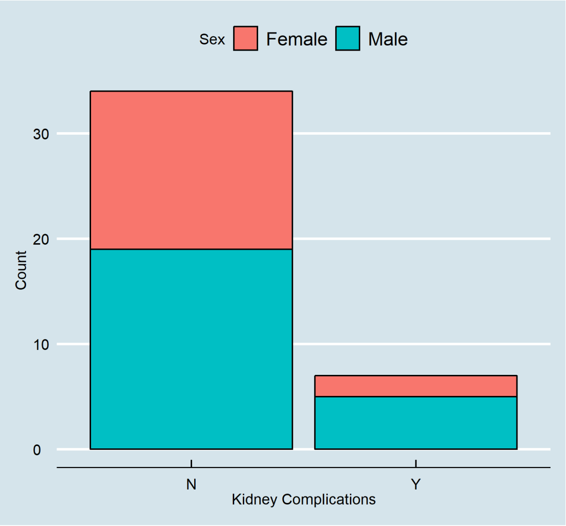


Figure 6.16: Bar Plot of Insulin Prescribed

Statistic	Value		
	Males	Females	Overall
Yes	5	2	7
No	19	15	34

Table 6.17: Summary Statistics of Insulin Prescribed

6.5.17 Annual Spending on Diabetes

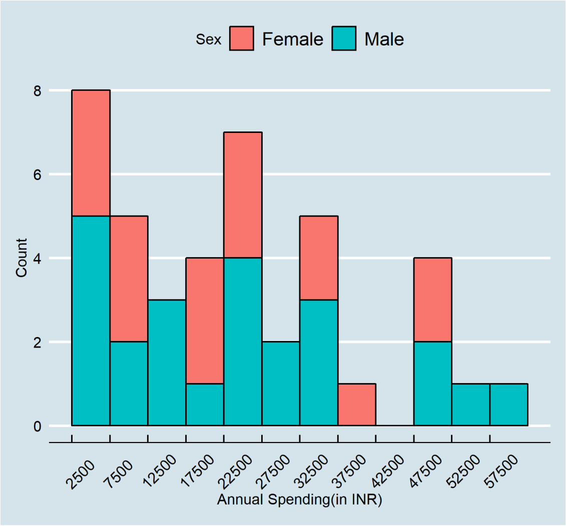


Figure 6.17: Histogram of Annual Spending

Statistic	Value		
	Males	Females	Overall
Mean	24591.67	22852.94	23870.73
Std. Dev.	16569.56	14426.28	15552.03
1^{st} Quantile	12000	12000	12000
Median	23500	22000	23000
3^{rd} Quantile	36000	36000	36000
Min	2500	2500	2500
Max	60000	48000	60000
	t-value	df	p-value
by Sex	0.35	37.24	0.7729

Table 6.18: Summary Statistics of Annual Spending

6.6 Correlations[13]

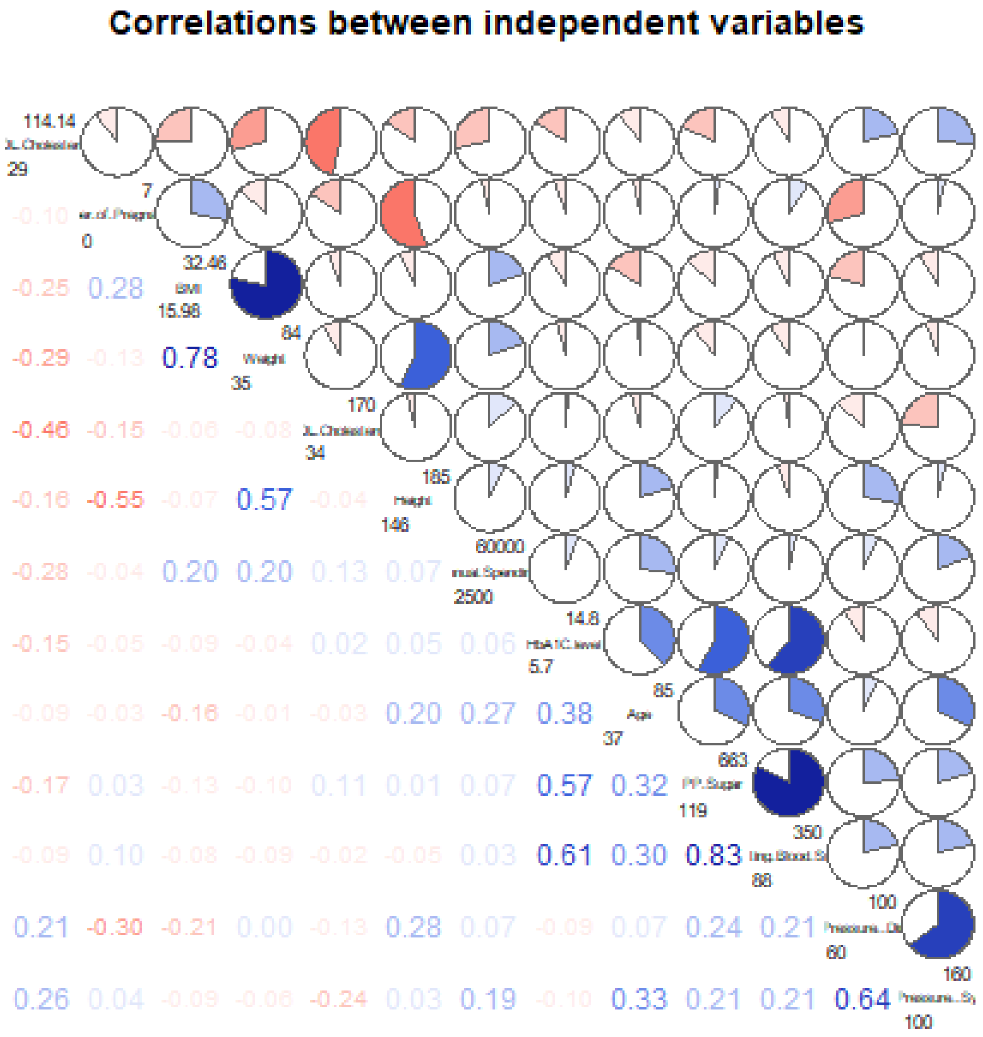


Figure 6.18: Correlation between Independent Variables

On the basis of Fig 6.18, it is concluded that there exists

a) a positive correlation between

   (i) Weight and Body Mass Index;

   (ii) Weight and Height;

   (iii) Post-Prandial Glucose Concentration and Fasting Plasma Glucose Concentration;

   (iv) Diastolic Blood Pressure and Systolic Blood Pressure;

(v) HbA1c Level and Post-Prandial Glucose Concentration;

(vi) HbA1c Level and Fasting Plasma Glucose Concentration.

b) a negative correlation between

(i) Number of Pregnancies and Height;

(ii) HDL Cholesterol and LDL Cholesterol.

Chapter 7

Results

7.1 Generalized Linear Model

A 10-fold cross validation[43] Generalized Linear Model with a split ratio[51] of 80% in the training data is run. Various diagnostic curves as well as prediction and fitted value curves[16,17] are plotted.

The models has been fitted to a Gaussian family with identity as link function. The response variable is taken to be the natural logarithm in order to avoid negative predictions.

The code is provided in Section B of the Appendix.

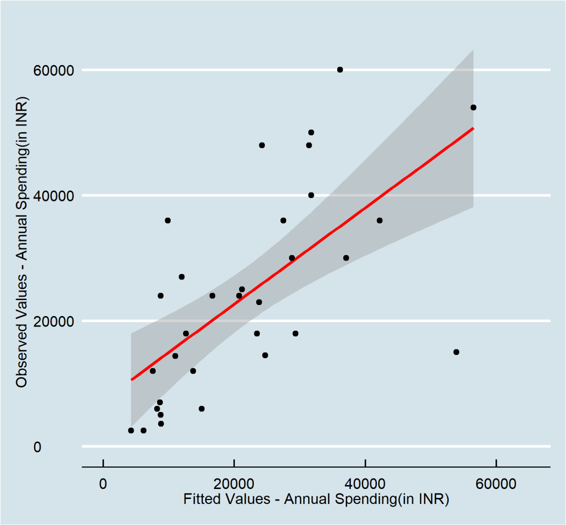


Figure 7.1: Fitted Values vs Observed Values

Variable	Statistic		
	Estimate	Std Error	p-value
(Intercept)	-54.24	44.73	0.242
Eye Complications	0.50	0.40	0.229
Age	0.024	0.019	0.219
Sex	-0.67	0.78	0.403
Height	0.37	0.28	0.209
Weight	-0.48	0.36	0.197
BMI	1.34	0.93	0.169
Pregnancies	-0.40	0.19	0.055
HbA1C level	-0.06	0.13	0.614
FPGC	0.001	0.005	0.787
PPGC	-0.0001	0.002	0.949
BP(D)	-0.039	0.04	0.336
BP(S)	0.045	0.02	0.058
HDL	-0.014	0.01	0.171
LDL	0.005	0.005	0.362
Insulin	0.075	0.63	0.906
Statistic		Value	df
Null Dev.		24.591	31
Residual Dev.		10.71	16
Pseudo-R^2		0.564	

Table 7.1: Summary Statistics of Gaussian GLM

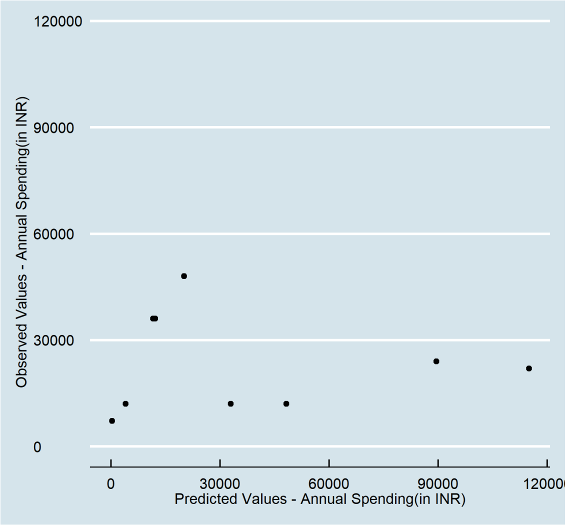


Figure 7.2: Predicted Values vs Observed Values

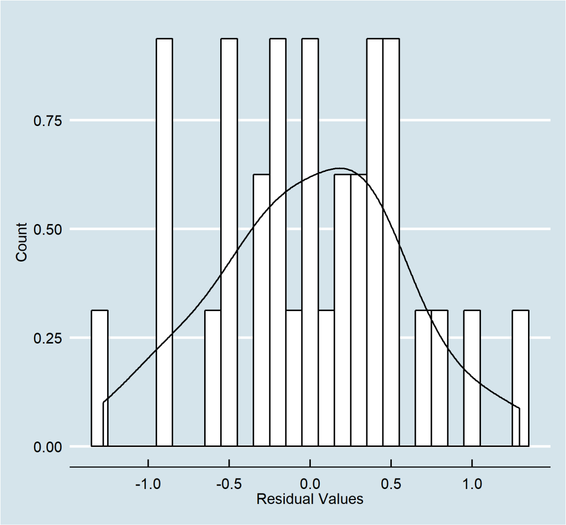


Figure 7.3: Histogram of Residuals

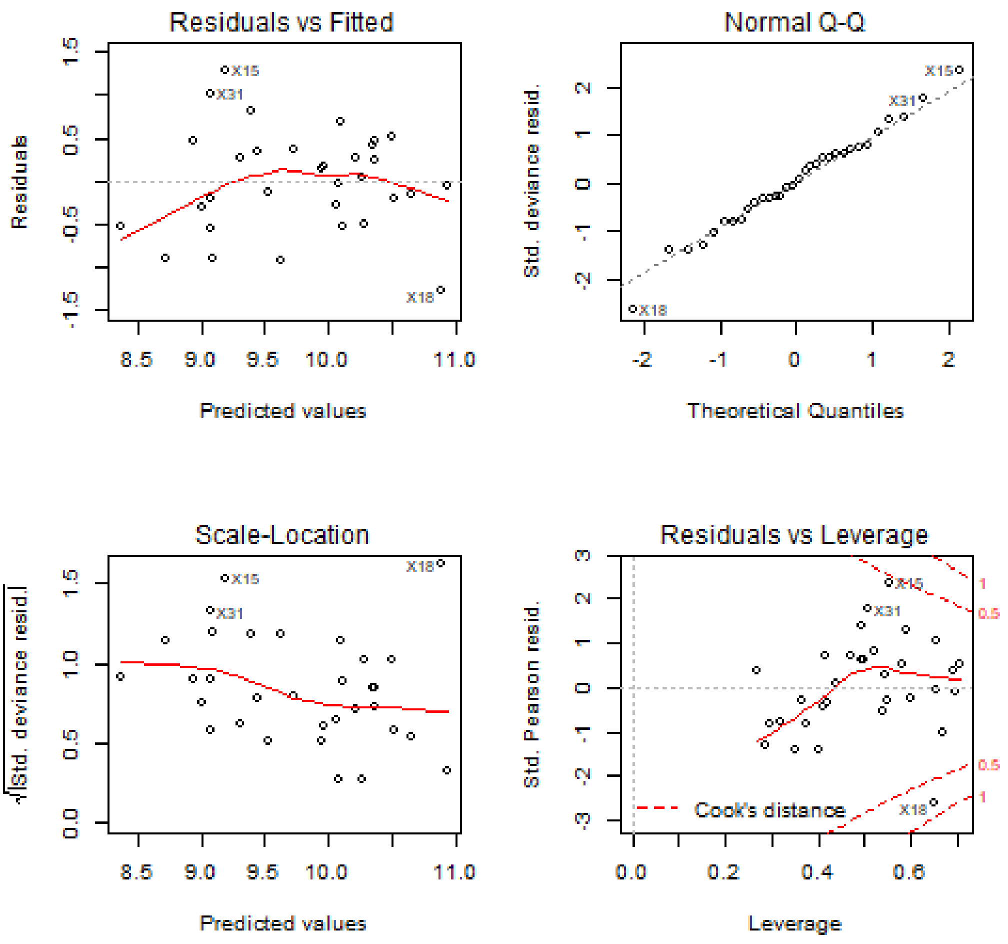


Figure 7.4: Diagnostic Graphs for GLM

7.2 Generalized Additive Models

A Generalized Additive Model[65] with a split ratio[51] of 80% in the training data is run. Various diagnostic curves as well as prediction and fitted value curves[16,17] are plotted.

The models are fit to a Gaussian family. The natural logarithm of Annual Spending is used as the response variable in order to avoid negative predictions.

The first model run includes a smoothing term for all our continuous variables and parametric forms for all other categorical data.

If the estimated degrees of freedom for any smoothed variable is 1.00, the smoothing term of that variable is removed and it is added as a parametric variable in the model instead.This step is repeated until all remaining smoothed variables have their estimated degrees of freedom > 1. Parametric variables which had very high p-values, i.e. > 0.7 were also removed as they served no purpose.

- 6 variables (HbA1C level, Fasting Plasma Glucose Conc., PP Glucose Conc., Systolic Blood Pressure. HDL Cholesterol and LDL Cholesterol) were removed from smoothed terms to parametric form in model 2.
- 3 variables(Fasting Plasma Glucose Conc., HbA1C level and PP Glucose Conc.) were removed completely from the model in model 3.
- Subsequently the Eye Complications parameter is removed in model 4, followed by removal of HDL Cholesterol in model 5.
- With enough data points now not involved in estimating smoothing parameters, the number of degrees of freedom of Age are increased in model 6.
- Sex is removed as a variable in model 7.
- Finally, Number of Pregnancies is moved from a smoothed term to a parametric term and is retained there as it was significant but had 1.0 estimated degrees of freedom. This was Model 8 our final model.

7.2.1 Initial Model (Model 1)

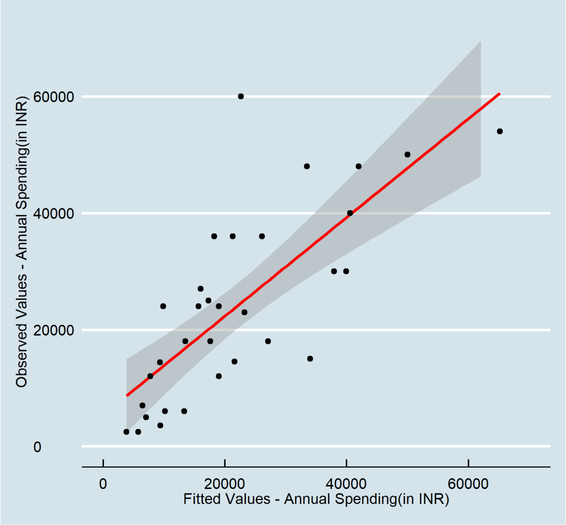


Figure 7.5: Fitted Values vs Observed Values

Variable	Statistic		
	Estimate	Std. Dev.	p-value
(Intercept)	10.32	0.56	3.6e-12
Eye Complications	0.30	0.36	0.419
Sex	-1.11	0.85	0.209
Insulin	0.41	0.58	0.482
	λ	EDF	p-value
s(Age)	0.825	1.59	0.288
s(BMI)	0.480	1.78	0.029
s(Preg)	1.01	1.33	0.142
s(HbA1C)	209522	1.00	0.995
s(FPGC)	161524	1.00	0.667
s(PP)	117650	1.00	0.700
s(BP-D)	5.96	1.17	0.596
s(BP-S)	290822	1.00	0.066
s(HDL)	255081	1.00	0.248
s(LDL)	154677	1.00	0.243
Statistic			Value
Adj. R^2			0.37
Dev. Explained			67.3%

Table 7.2: Summary Statistics of Gaussian GAM Model 1

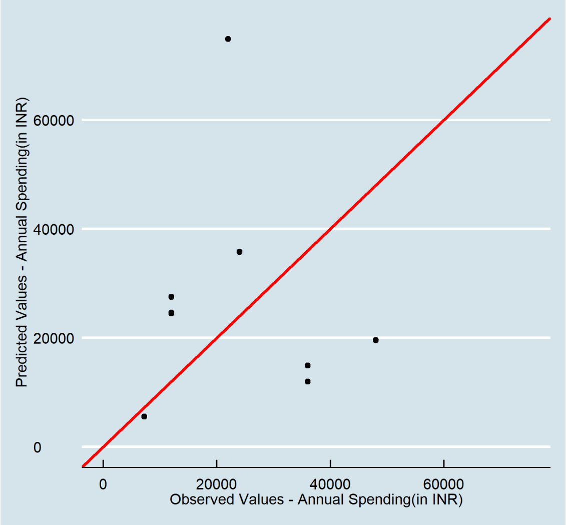


Figure 7.6: Predicted Values vs Observed Values

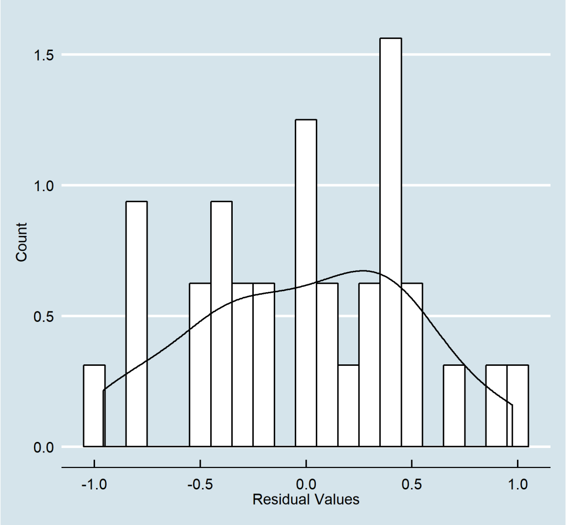


Figure 7.7: Histogram of Residuals

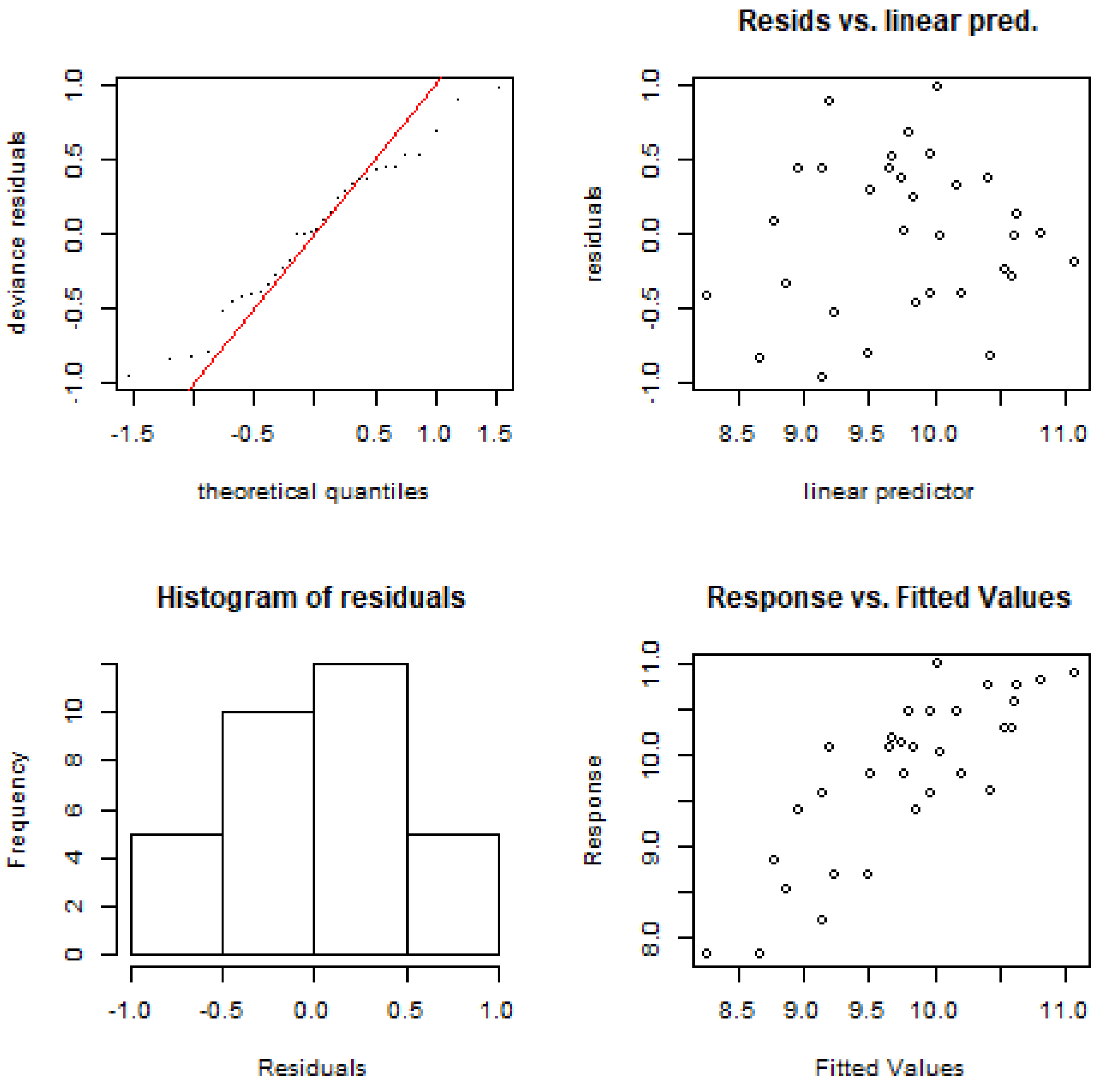


Figure 7.8: Diagnostic Plots for GAM Model 1

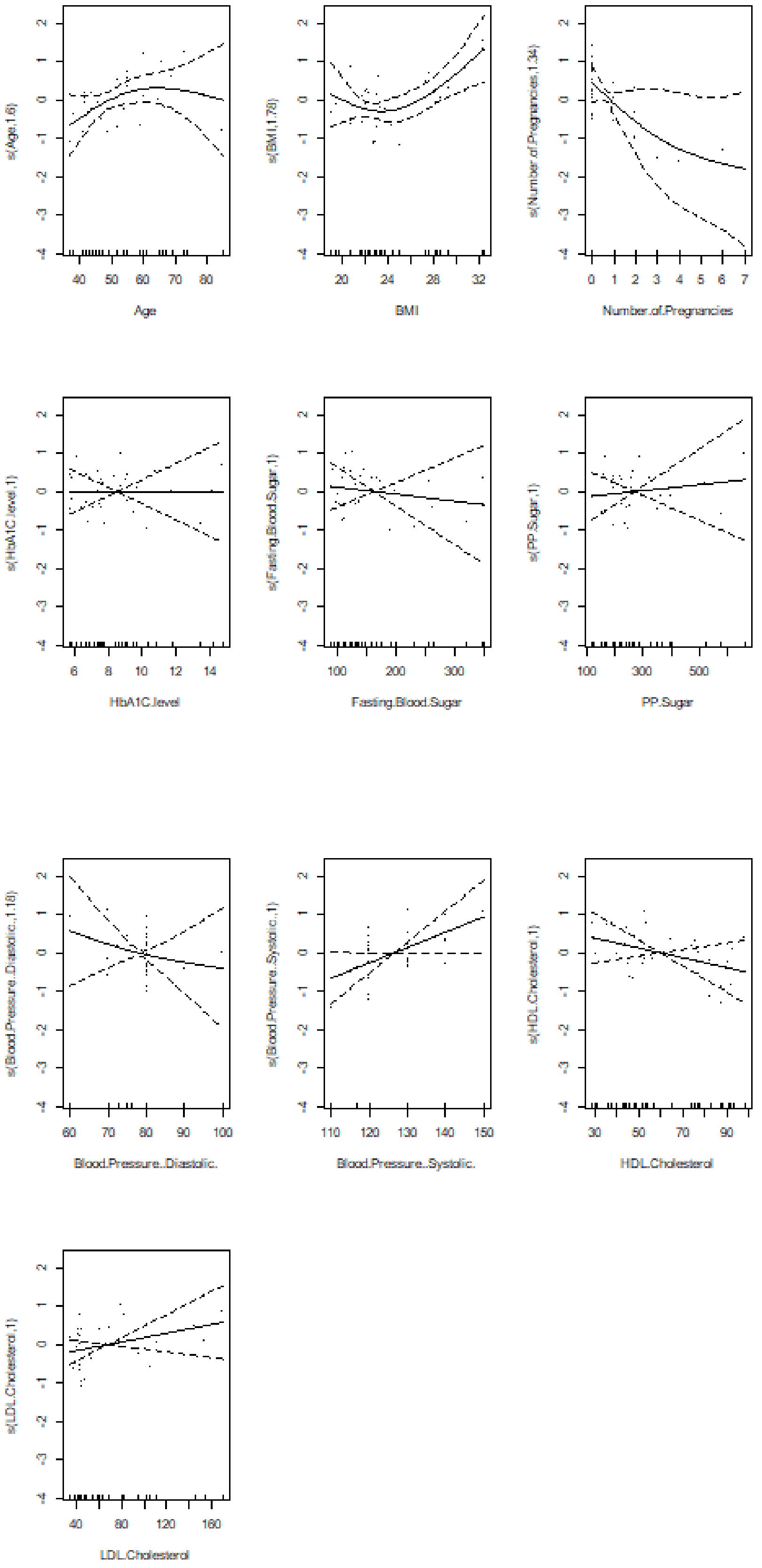


Figure 7.9: Smoothed functions of all Variables

7.2.2 Final Model(Model 8)

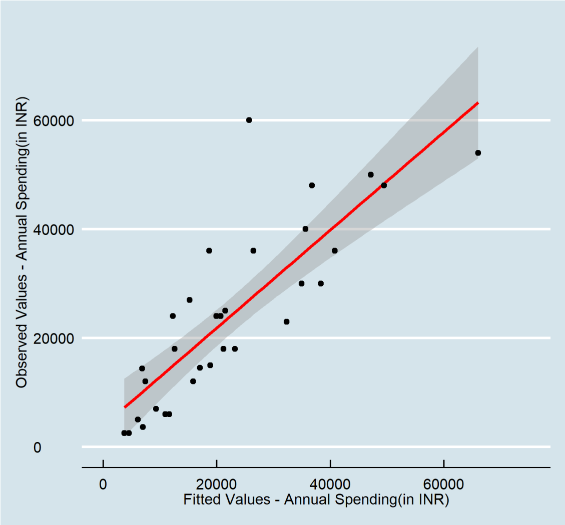


Figure 7.10: Fitted Values vs Observed Values

Variable	Statistic		
	Estimate	Std. Dev.	p-value
(Intercept)	0.45	1.88	0.81
Pregnancies	-0.23	0.07	0.005
BP - S	0.06	0.01	0.000
LDL	0.009	0.002	0.005
Insulin	0.99	0.36	0.013
	λ	EDF	p-value
s(Age)	16.79	3.507	0.074
s(BMI)	14.84	2.387	0.010
s(BP-D)	0.17	1.897	0.004
Statistic			Value
Adj. R^2			0.641
Dev. Explained			77.8%

Table 7.3: Summary Statistics of Gaussian GAM Model 8

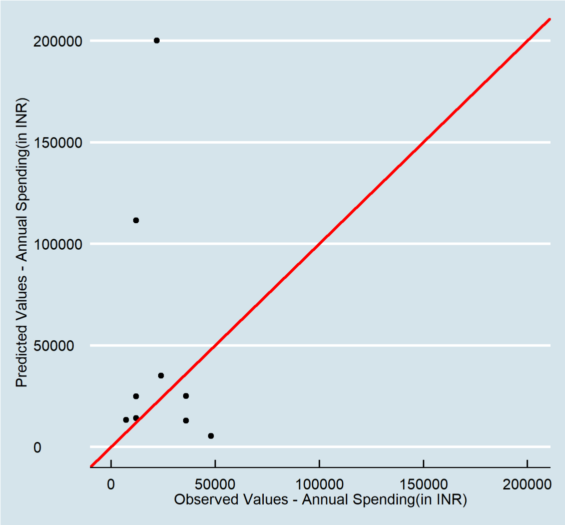


Figure 7.11: Predicted Values vs Observed Values

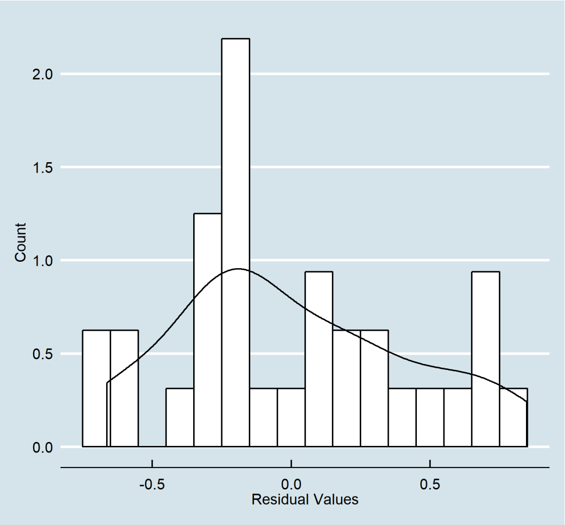


Figure 7.12: Histogram of Residuals

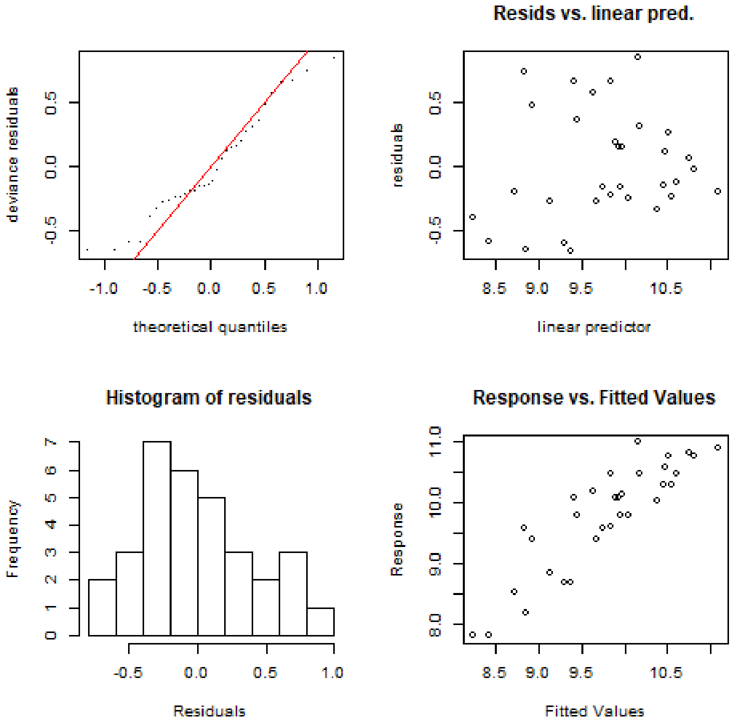


Figure 7.13: Diagnostic Plots for GAM Model 8

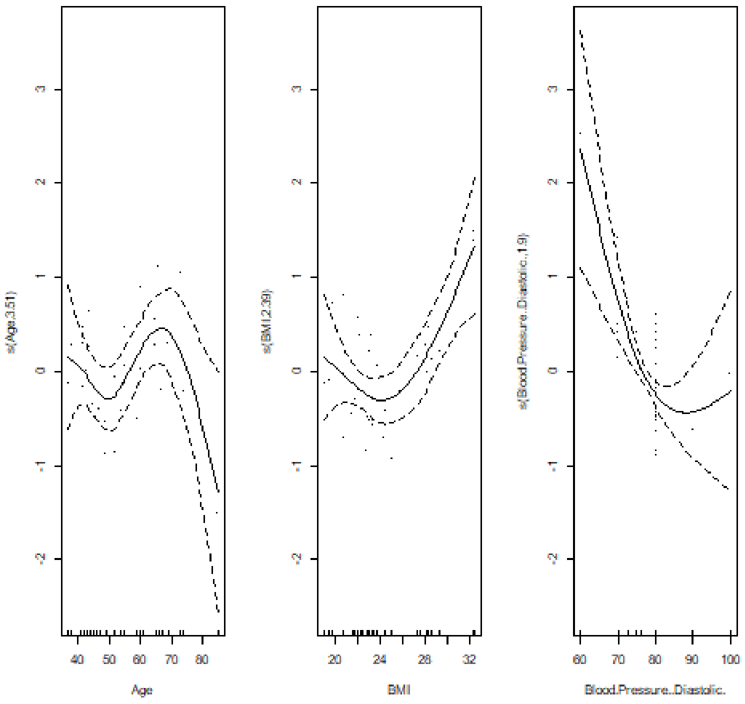


Figure 7.14: Smoothed functions of all Variables

The interaction of the smoothed functions of Age, BMI and Diastolic Blood Pressure are shown here.

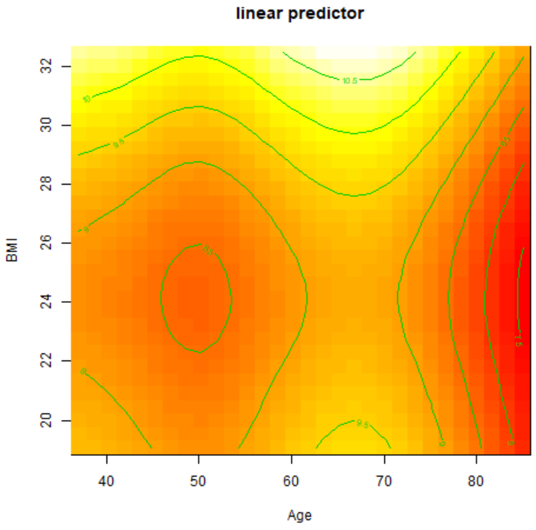


Figure 7.15: Interaction between smoothed Age and BMI terms

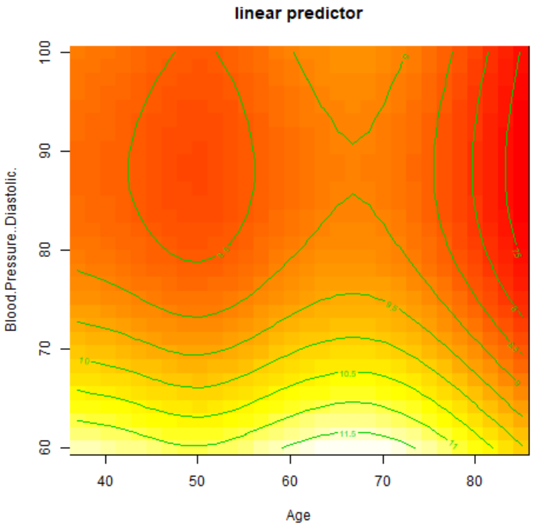


Figure 7.16: Interaction between smoothed Age and Diastolic Blood Pressure terms

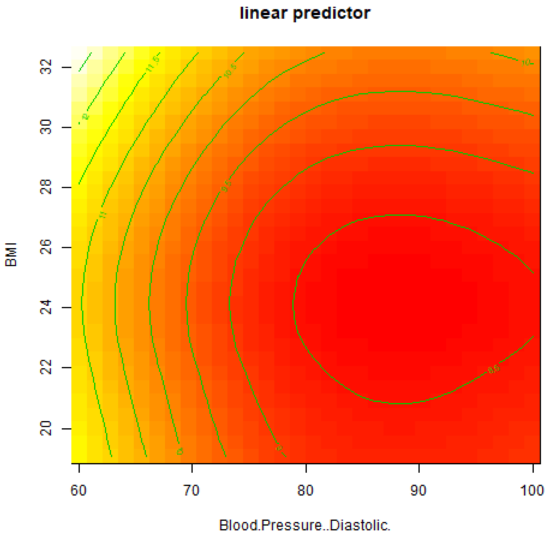


Figure 7.17: Interaction between smoothed BMI and Diastolic Blood Pressure terms

7.3 Conclusion

The histogram of residuals by the Generalized Linear Model (GLM) (Figure 7.4) is mostly normal, but Predicted values are not in line with the Observed data points (Figure 7.3). The Diagnostic curves (Figure 7.5) show that the residuals of the fitted/predicted values are overall around zero, standard deviance residuals are mostly normal, the Scale-Location graph as well shows a mostly straight line.

We see a huge jump in p-values, adjusted R^2 and Deviance Explained values between Model 1 and Model 8 of the Generalized Additive Models (GAMs). Apart from 2 values, the predicted values are close to the observed values. We see that the smoothed functions of all three smoothed functions - Age, BMI and Diastolic Blood Pressure are non-linear.

Also, overall GAM fits better to the data than GLM. It is seen that

1. there is a negative relationship between
   - (i) the number of pregnancies has an inverse relationship to the annual spending.
2. there is a positive relationship between
   - (i) Systolic Blood Pressure and Annual Spending (probably functioning as an indicator cardiovascular disease)
   - (ii) LDL Cholesterol and Annual Spending
   - (iii) Insulin Prescription and Annual Spending

The smoothed functions of BMI indicate that both high and low BMIs add more to the annual spending.
Similarly for Diastolic Blood Pressure, low values cause a higher prediction of annual spending than the average values. The curve also tilts upward as the diastolic blood pressure increases.
For Age, we see an increasing trend in spending once age crosses 50, which only reverses when the age crosses 68.

Part III

Annual Spending Clustering based on Medical Indicators

Chapter 8

Machine Learning Clustering Algorithms

We divide our dataset into different classes so as to create Health Groups and provide an insurance premium band to the customers. This is done via clustering algorithms discussed below:

8.1 K-Means Clustering[19]

8.1.1 Introduction

K-means clustering[66] is a partitional clustering algorithm that uses the Squared Error criterion. It is one of the simplest algorithms that employ the squared error criterion.[67] Partitional algorithms are best suited for large sets, where dendrograms are computationally expensive. But, with such partitional algorithms comes the problem of choosing the number of desired output clusters. This problem is solved by the Modified Hubert's Γ (MH) Statistic.[68] The algorithm is run multiple times with different starting states and the best criterion value is then used as the output cluster.

8.1.2 The Algorithm

The squared error criterion for clustering is given by

$$e^2 = \sum_{j=1}^{K} \sum_{i=1}^{n_j} \parallel x_i^{(j)} - c_j \parallel^2 \tag{8.1}$$

where $x_i^{(j)}$ is the i^{th} pattern that belongs to the j^{th} cluster and c_j is the centroid of the j^{th} cluster.

Steps

- Given a particular K, the algorithm chooses K cluster centers at random points inside the hypervolume of observations.
- The algorithm then assigns each observation to its closest cluster center.
- The cluster centers are then recomputed using the current cluster memberships in an effort to minimize the squared error criterion.
- When the decrease in the squared error criterion falls below a certain threshold, the convergence criteria is met and the algorithm stops.

8.1.3 Elbow method

The Elbow method creates a graph of the number of clusters vs the Weighted Sum-of-Squares. The optimum number of clusters would be the one that would be closest point to the origin.

8.1.4 NbClust

The NbClust[69,70] method uses 30 different indices to score the optimum number of clusters. The number of clusters is then chosen based on majority vote of the 30 indices.

8.2 Hierarchical Clustering[20]

Hierarchical clustering is sequentially agglomerative i.e. it merges clusters at every step until only 1 cluster remains. This generates a strictly nested hierarchy of n partitions (n = number of observations). We can then select a clustering level that represents the specific number of clusters of interest.

Methods for agglomeration are minimum variance method,[71] complete and single-link methods[72] and non-parametric U statistic.[73]

The other method of hierarchical clustering is the divisive method, a top-down approach where all observations are in one cluster and are subsequently broken until there are n clusters. This method is based on minimizing the within cluster error sum of squares.[74]

Chapter 9

Results

9.1 Finding the correct number of Clusters

The annual spending data is clustered using the hierarchical clustering and K-means clustering methods. A decision tree using the medical variables as independent variables and the cluster number as the dependent variable is also run. The resulting decision tree is divided based on medical variables and has end nodes as the cluster number, which are mostly homogenous classes of annual spending.

Both the elbow method and the NbClust majority rule recommend 3 clusters. Thus our annual spending data is divided into 3 clusters.

The Code[16,17,69,70,75–77] is in Appendix - Section C.

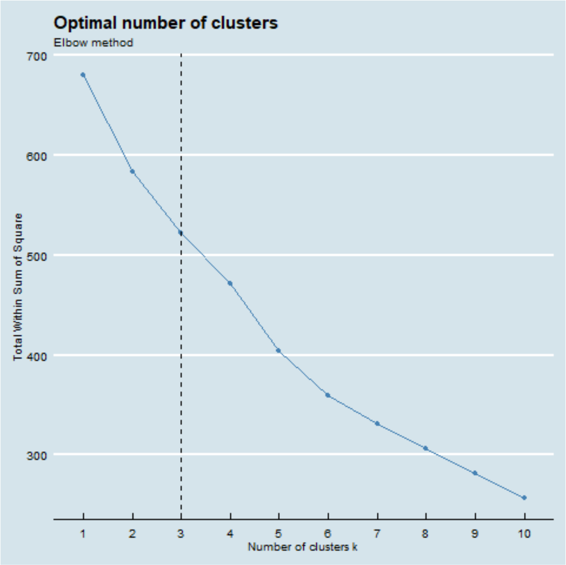
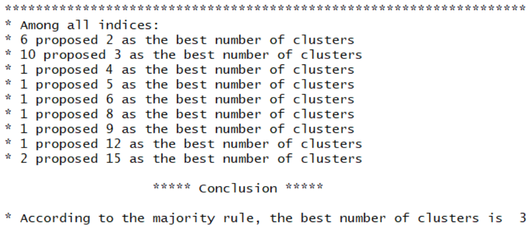


Figure 9.1: Elbow Method



```
**********************************************************************
* Among all indices:
* 6 proposed 2 as the best number of clusters
* 10 proposed 3 as the best number of clusters
* 1 proposed 4 as the best number of clusters
* 1 proposed 5 as the best number of clusters
* 1 proposed 6 as the best number of clusters
* 1 proposed 8 as the best number of clusters
* 1 proposed 9 as the best number of clusters
* 1 proposed 12 as the best number of clusters
* 2 proposed 15 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  3
```

Figure 9.2: NbClust Results

9.2 Clustering Results

The annual spending thus clustered is represented as -

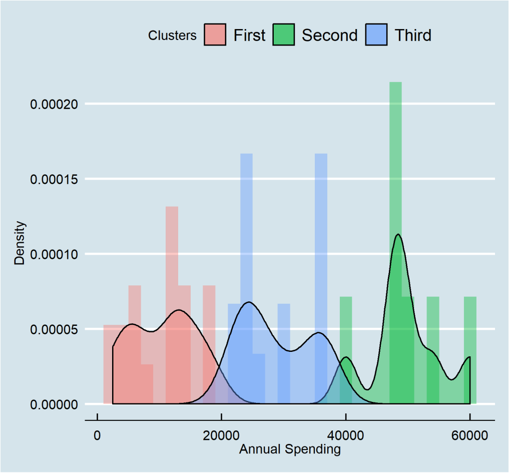


Figure 9.3: Clustered Annual Spending

9.3 Decision Tree-Clustering Results

From our decision tree, we get the following -

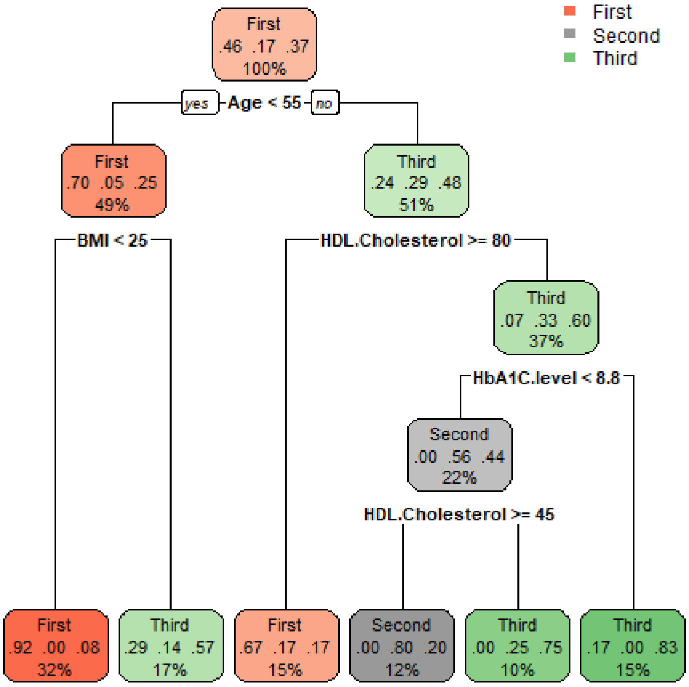


Figure 9.4: Final Decision Tree

9.4 Conclusions

For Annual Spending, three clusters are obtained. The major medical indicators that place patients in particular categories are Age, BMI, HDL Cholesterol and HbA1C Level.
One can easily see that when Age< 65 and BMI< 25, people are relegated into the lowest annual spending cluster.
Better medical indicators lead to the decision tree classifying one into a lower mean annual spending cluster.

Chapter 10

Conclusions

In conclusion, from the first part of our thesis, it can be obtained that indicators including Plasma Glucose Concentration 2hrs into an OGTT, Age, Body Mass Index, Triceps Skin Fold Thickness and Diastolic Blood Pressure have the most predictive power in terms of predicting onset of diabetes.
Thus, a non-diabetic should make sure that these quantities are kept in check so as to minimize risk of diabetes.

In the second part of our thesis Annual spending on diabetes is succesfully linked to medical indicators of diabetes. The most important medical indicators here turn out to be Number of Pregnancies, Systolic and Diastolic Blood Pressure, LDL Cholesterol, Prescription of Insulin, Age and BMI. All variables except Number of Pregnancies are positively correlated with annual spending, indicating that to keep costs down, one should control their BMI, Diastolic and Systolic Blood Pressure and LDL Cholesterol.

The third part of our thesis shows that Annual Spending data can be clustered into three distinct patches. After running a decision tree through the same, based on medical indicators, better health is usually a sign of lower annual spending. The main deciding features here used were Age, BMI, HDL Cholesterol and HbA1C level values.
Thus, Glucose measuring quantities, Fasting Plasma Glucose Concentration, Post-Prandial Glucose Concentration and HbA1C levels are not correlated with annual spending but HbA1C levels make a hyperplane that divides our data well in terms of annual spending clusters.

Chapter 11

Future Work

Seeing as the thesis has been plagued with lack of data, we recommend a long-term in-depth study of $18 - 24$ months, where monthly patient data is noted along with monthly spending.

Doing so will result in a time-series dataset via which we can find causes of spikes and lulls in spending based on medical data.

One can include variables such as the Triceps Skin Fold thickness, Liver,Kidney and cardiovascular disease indicators as well as listing other diabetic problems such as podiatric problems.

Spending itself can be broken down by drug, procedure or physician visits etc. to give a clearer picture.

With such a study, we believe a much more concrete link can be established between Spending on diabetes and patients' medical indicators which would allow widespread diabetes' insurance penetration by offering patients competitive and affordable plans.

Bibliography

[1] Park HA. An introduction to logistic regression: From basic concepts to interpretation with particular attention to nursing domain. *Journal of Korean Academy of Nursing*, 43(2):154–164, 2013.

[2] Aman Kataria and M D Singh. A review of data classification using k-nearest neighbour algorithm. *International Journal of Emerging Technology and Advanced Engineering*, 3(6):354–360, 2013.

[3] Ortiz et. al. A review of classification problems and algorithms in renewable energy applications. *Energies*, 9(8):607, 2016.

[4] Mostafa Langarizadeh and Fateme Moghbeli. Applying naive bayesian networks to disease prediction: a systematic review. *Acta Informatica Medica (2016): 364.*, 24.5:364–369, 2016.

[5] Gurneet Kaur and Er. Neelam Oberai. A review article on naive bayes classifies with various smoothing techniques. *International Journal of Computer Science and Mobile Computing*, 3(10):864–868, 2014.

[6] SR Safavian and D Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[7] Torsten Hothorn, Kurt Hornik, and Achim Zeileis. ctree: Conditional inference trees. *The Comprehensive R Archive Network*, pages 1–34, 2015.

[8] PO Gislason, JA Benidiktsson, and JR Sveinsson. Random forests for land cover classification. *Pattern Recognition Letters*, 27:294–300, 2006.

[9] E Alfaro, N Garcia, M Gámez, and D Elizondo. Bankruptcy forecasting: An empirical comparison of adaboost and neural networks. *Decision Support Systems*, 45(1):110–122, 2008.

[10] T Chen and C Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKKD International conference on knowledge discovery and data mining*, pages 785–794, 2016.

[11] J Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.

[12] JW Smith, JE Everhart, WC Dickson, WC Knowler, and RS Johannes. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Annual Symposium on Computer Application in Medical Care*, pages 261–265. American Medical Informatics Association, 1988.

[13] Kevin Wright. *corrgram: Plot a Correlogram*, 2017. R package version 1.12.

[14] P Jong and G Heller. *Generalized Linear Models for Insurance Data*. Cambridge University Press, 2008.

[15] SN Wood. *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC, 2017.

[16] JB Arnold. *ggthemes: Extra Themes, Scales and Geoms for 'ggplot2'*, 2019. R package version 4.1.0.

[17] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.

[18] Hadley Wickham, Romain Francois, Lionel Henry, and Kirill Müller. *dplyr: A Grammar of Data Manipulation*, 2017. R package version 0.7.4.

[19] AK Jain, MN Murty, and PJ Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[20] GW Milligan and MC Cooper. Methodology review: Clustering methods. *Applied Psychological Measurement*, 11:329–354, 1987.

[21] International Diabetes Federation. Diabetes Atlas 2017. Technical report, International Diabetes Federation, 2017.

[22] D R Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, 20, No. 2:215–242, 1958.

[23] Chao-Ying Joanne Peng, Kuk Lida Lee, and Gary M Ingersoll. An introduction to logistic regression analysis and reporting. *The Journal of Educational Research*, 96:3–14, 2002.

[24] RO Duda and PE Hart. *Pattern Classification and Scene Analysis*. New York: John Wiley & Sons, 1973.

[25] T Cover. Estimation by the nearest neighbor rule. *IEEE Transactions on Information Theory*, 14.1:50–55, 1968.

[26] Peter Hart. The condensed nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory*, 14.3:515–516, 1968.

[27] BV Dasarathy. *Nearest Neighbor (NN) Norms,NN Pattern Classification Techniques*. IEEE Computer Society Press, 1990.

[28] Y Yang and X Liu. Re-examination of text categorization methods. *In SIGIR 1999*, pages 42–29, 1999.

[29] Alfons Juan and Hermann Ney. Reversing and smoothing the multinomial naive bayes text classifier. *PRIS*, pages 200–212, 2002.

[30] Y Freund and RE Schapire. Experiments with a new boosting algorithm. *Machine Learning. Proceedings of the Thirteenth International Conference*, pages 148–156, 1996.

[31] RE Schapire. A brief introduction to boosting. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1401–1406, 1999.

[32] L Breiman. Bagging predictors. *Technical Report No. 421, Department of Statistics, University of California, Berkeley.*, 1994.

[33] GJ Briem, JA Benediktsson, and JR Sveinsson. Multiple classifiers applied to multisource remote sensing data. *IEEE Trans. Geosci. Remote Sens*, 40:2291–2299, 2002.

[34] L Breiman. Random forests. *Mach. Learn.*, 40:5–32, 2001.

[35] L Breiman, JH Friedman, RA Olshen, and CJ Stone. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[36] Y Freund and RE Schapire. A decision theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

[37] E Bauer and R Kohavi. An empirical comparison of voting classification algorithm: bagging, boosting and variants. *Machine Learning*, 36:105–142, 1999.

[38] J Friedman, T Hastie, and R Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 38(2):391–393, 2000.

[39] R Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. *International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.

[40] V Bewick, L Cheek, and J Ball. Statistics review 13:receiver operating characteristic curves. *Critical Care (London, England)*, 8(6):508–512, 2004.

[41] JR Landis and GG Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.

[42] JL Fleiss. *Statistical methods for rates and proportions*. John Wiley:New York, 1981.

[43] M Kuhn et al. *caret: Classification and Regression Training*, 2018. R package version 6.0-80.

[44] PH Bennett, TA Burch, and M Miller. Diabetes mellitus in american(pima) indians. *The Lancet*, 298(7716):125–128, 1971.

[45] World Health Organization Technical Report Series. Geneva. Report of a study group: Diabetes mellitus. Technical Report 727, World Health Organization, 1985.

[46] WC Knowler, PH Bennett, RF Hamman, and M Milier. Diabetes incidence and prevalence in pima indians: a 19-fold greater incidence than in rochester, minnesota. *American Journal of Epidemiology*, 108:497–505, 1978.

[47] JR Kraft. Detection of diabetes mellitus in situ (occult diabetes). *Lab med*, 6.2:10–22, 1975.

[48] MJ Albrink and JW Meigs. The relationship between serum triglycerides and skinfold thickness in obese subjects. *Annals of the New York Academy of Sciences*, 131.1:673–683, 1965.

[49] American Diabetes Association. Gestational diabetes mellitus. *Diabetes Care*, 27(Suppl 1):s88–s90, 2004.

[50] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013.

[51] Jarek Tuszynski. *caTools: Tools: moving window statistics, GIF, Base64, ROC AUC, etc.*, 2014. R package version 1.17.1.

[52] X Robin et al. proc: an open-source package for r and s+ to analyze and compare roc curves. *BMC Bioinformatics*, 12:77, 2011.

[53] A Karatzoglou, A Smola, K Hornik, and A Zeileis. kernlab – an S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9):1–20, 2004.

[54] C Weihs, U Ligges, K Luebke, and N Raabe. klar analyzing german business cycles. In D. Baier, R. Decker, and L. Schmidt-Thieme, editors, *Data Analysis and Decision Support*, pages 335–343, Berlin, 2005. Springer-Verlag.

[55] T Hothorn, K Hornik, and A Zeileis. Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3):651–674, 2006.

[56] A Liaw and M Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.

[57] S Chatterjee. *fastAdaboost: A Fast Implementatio of AdaBoost*, 2016.

[58] T Chen et al. *xgboost: Extreme Gradient Boosting*, 2018. R package version 0.71.2.

[59] JA Nelder and RWM Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135.3:370–384, 1972.

[60] AJ Dobson and A Barnett. *An Introduction to Generalized Linear Models*. Chapman & Hall/CRC, 2008.

[61] TJ Hastie and RJ Tibshirani. *Generalized Additive Models*. Chapman & Hall/CRC, 1990.

[62] R Williams, L Van Gaal, and C Lucioni. Assessing the impact of complications on the costs of type ii diabetes. *Diabetologia*, 45.1:S13–S17, 2002.

[63] V Falanga. Wound healing and its impairment in the diabetic foot. *The Lancet*, 366(9498):1736–1743, 2005.

[64] DR McCane et al. Comparison of tests for glycated haemoglobin and fasting and two hour plasma glucose concentrations as diagnostic methods for diabetes. *BMJ*, 308.6940:1323–1328, 1994.

[65] S. N. Wood. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society (B)*, 73(1):3–36, 2011.

[66] JA Hartigan. *Clustering Algorithms*. Wiley New York, 1975.

[67] J McQueen. Some methods for classification and analysis of multivariate observations. In *In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[68] RC Dubes. How many clusters are best?-an experiment. *Pattern Recognition*, 20.6:645–663, 1987.

[69] Alboukadel Kassambara and Fabian Mundt. *factoextra: Extract and Visualize the Results of Multivariate Data Analyses*, 2017. R package version 1.0.5.

[70] Malika Charrad, Nadia Ghazzali, Véronique Boiteau, and Azam Niknafs. NbClust: An R package for determining the relevant number of clusters in a data set. *Journal of Statistical Software*, 61(6):1–36, 2014.

[71] JH Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58:236–244, 1963.

[72] SC Johnson. Hierarchical clustering schemes. *Psychometrika*, 32:241–254, 1967.

[73] RG D'Andrade. U-statistic hierarchical clustering. *Psychometrika*, 43:59–67, 1978.

[74] AWF Edwards and L Cavalli-Sforza. A method for cluster analysis. *Biometrics*, 35:169–181, 1965.

[75] Tal Galili. dendextend: an r package for visualizing, adjusting, and comparing trees of hierarchical clustering. *Bioinformatics*, 2015.

[76] Terry Therneau, Beth Atkinson, and Brian Ripley. *rpart: Recursive Partitioning and Regression Trees*, 2017. R package version 4.1-11.

[77] Stephen Milborrow. *rpart.plot: Plot 'rpart' Models: An Enhanced Version of 'plot.rpart'*, 2018. R package version 3.0.5.

Appendices

Appendix A

Section A

A.1 Preliminary Analysis

```
df <- read.csv("diabetes.csv")

df <- df[!(df$BloodPressure==0),]
df <- df[!(df$SkinThickness==0),]
df <- df[!(df$Glucose==0),]
df <- df[!(df$BMI==0),]
df <- df[!(df$Insulin==0),]

df$Outcome[df$Outcome == "0"] <- "Non-Diabetic"
df$Outcome[df$Outcome == "1"] <- "Diabetic"
df$Outcome <- as.factor(df$Outcome)
df$Outcome <- factor(df$Outcome, levels = c("Non-Diabetic","Diabetic"))

library(ggplot2)
library(ggthemes)
library(corrgram)

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


ggplot() +geom_density(aes(x = df$Age, fill= df$Outcome), alpha = 0.4) +
  theme_economist()+ theme(legend.position="top")+ xlab("Age")+ylab("Density")+
  scale_fill_discrete(name = "Outcome", labels =c("Non-Diabetic","
    Diabetic"))
ggsave("Age.png")

mean(df$Age)
aggregate(df$Age~df$Outcome, FUN=mean)

sd(df$Age)
aggregate(df$Age~df$Outcome, FUN=sd)

quantile(df$Age)
aggregate(df$Age~df$Outcome, FUN=quantile)


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


ggplot() +geom_density(aes(x = df$BMI, fill= df$Outcome), alpha = 0.4) +
  theme_economist()+ theme(legend.position="top")+ xlab("BMI")+ylab("
    Density")+
  scale_fill_discrete(name = "Outcome", labels =c("Non-Diabetic","
    Diabetic"))
ggsave("BMI.png")

mean(df$BMI)
aggregate(df$BMI~df$Outcome, FUN=mean)

sd(df$BMI)
aggregate(df$BMI~df$Outcome, FUN=sd)

quantile(df$BMI)
aggregate(df$BMI~df$Outcome, FUN=quantile)


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


ggplot() +geom_density(aes(x = df$Insulin, fill= df$Outcome), alpha =
    0.4) +
  theme_economist()+ theme(legend.position="top")+ xlab("Insulin \
     U003BCIU/ml")+ylab("Density")+
   scale_fill_discrete(name = "Outcome", labels =c("Non-Diabetic","
     Diabetic"))
ggsave("Insulin.png")

mean(df$Insulin)
aggregate(df$Insulin~df$Outcome, FUN=mean)

sd(df$Insulin)
aggregate(df$Insulin~df$Outcome, FUN=sd)


quantile(df$Insulin)
aggregate(df$Insulin~df$Outcome, FUN=quantile)


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

ggplot() +geom_density(aes(x = df$SkinThickness, fill= df$Outcome),
     alpha = 0.4) +
   theme_economist()+ theme(legend.position="top")+ xlab("Triceps Skin
     Fold Thickness(mm)")+ylab("Density")+
   scale_fill_discrete(name = "Outcome", labels =c("Non-Diabetic","
     Diabetic"))
ggsave("Skin.png")

mean(df$SkinThickness)
aggregate(df$SkinThickness~df$Outcome, FUN=mean)

sd(df$SkinThickness)
aggregate(df$SkinThickness~df$Outcome, FUN=sd)

quantile(df$SkinThickness)
aggregate(df$SkinThickness~df$Outcome, FUN=quantile)

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

```

```
ggplot() +geom_density(aes(x = df$BloodPressure, fill= df$Outcome), alpha = 0.4) +
  theme_economist()+ theme(legend.position=”top”)+ xlab(”Diastolic Blood Pressure(mmHg)”)+ylab(”Density”)+
  scale_fill_discrete(name = ”Outcome”, labels =c(”Non−Diabetic”,”Diabetic”))
ggsave(”BP.png”)

mean(df$BloodPressure)
aggregate(df$BloodPressure~df$Outcome, FUN=mean)

sd(df$BloodPressure)
aggregate(df$BloodPressure~df$Outcome, FUN=sd)

quantile(df$BloodPressure)
aggregate(df$BloodPressure~df$Outcome, FUN=quantile)

#——————————————————————————————————————————————

ggplot() +geom_density(aes(x = df$Glucose, fill= df$Outcome), alpha = 0.4) +
  theme_economist()+ theme(legend.position=”top”)+ xlab(”Plasma Glucose Conc. at 2Hrs in OGTT(mg/dl)”)+
  ylab(”Density”)+scale_fill_discrete(name = ”Outcome”, labels =c(”Non−Diabetic”,”Diabetic”))
ggsave(”Glucose.png”)

mean(df$Glucose)
aggregate(df$Glucose~df$Outcome, FUN=mean)

sd(df$Glucose)
aggregate(df$Glucose~df$Outcome, FUN=sd)

quantile(df$Glucose)
aggregate(df$Glucose~df$Outcome, FUN=quantile)


#——————————————————————————————————————————————
```

```

ggplot() +geom_density(aes(x = df$Pregnancies, fill= df$Outcome), alpha = 0.4) +
  theme_economist()+ theme(legend.position=”top”)+ xlab(”Number of Times Pregnant”)+ylab(”Density”)+
  scale_fill_discrete(name = ”Outcome”, labels =c(”Non–Diabetic”,”Diabetic”))
ggsave(”Preg.png”)

mean(df$Pregnancies)
aggregate(df$Pregnancies~df$Outcome, FUN=mean)

sd(df$Pregnancies)
aggregate(df$Pregnancies~df$Outcome, FUN=sd)

quantile(df$Pregnancies)
aggregate(df$Pregnancies~df$Outcome, FUN=quantile)


#————————————————————————————————————————————————————————————————————
ggplot() +geom_density(aes(x = df$DiabetesPedigreeFunction, fill= df$Outcome), alpha = 0.4) +
  theme_economist()+ theme(legend.position=”top”)+ xlab(”Diabetes Pedigree Function”)+ylab(”Density”)+
  scale_fill_discrete(name = ”Outcome”, labels =c(”Non–Diabetic”,”Diabetic”))
ggsave(”DPF.png”)

mean(df$DiabetesPedigreeFunction)
aggregate(df$DiabetesPedigreeFunction ~df$Outcome, FUN=mean)

sd(df$DiabetesPedigreeFunction)
aggregate(df$DiabetesPedigreeFunction~df$Outcome, FUN=sd)

quantile(df$DiabetesPedigreeFunction)
aggregate(df$DiabetesPedigreeFunction ~df$Outcome, FUN=quantile)

```

```

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

ggplot() +geom_bar(aes(df$Outcome, fill = df$Outcome), color = ”black”)+
  ylab(”Count”) + xlab(”Outcome”)+ theme_economist()+
  scale_fill_discrete(name = ”Outcome”, labels =c(”Non−Diabetic”,”
    Diabetic”))+ theme(legend.position=”none”)
ggsave(”Outcome.png”)

length(df$Outcome[df$Outcome==”Diabetic”])
length(df$Outcome[df$Outcome==”Non−Diabetic”])

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

png(filename=”Corr.png”)
corrgram(df, order=TRUE,
          main=”Correlations between independent variables”,
          lower.panel=panel.cor, upper.panel=panel.pie,
          diag.panel=panel.minmax, text.panel=panel.txt)
dev.off()
```

A.2 Logistic Regression

```
library(pROC)
library(caret)
library(caTools)
df = read.csv(’diabetes.csv’)  #importing dataset

#Removing missing data
df <− df[!(df$BloodPressure==0),]
df <− df[!(df$SkinThickness==0),]
df <− df[!(df$Glucose==0),]
df <− df[!(df$BMI==0),]
df <− df[!(df$Insulin==0),]

set.seed(06061968)

```

```
#Categorical Data
df$Outcome <– as.factor(df$Outcome)
df$Outcome <– factor(df$Outcome,labels = c(”No”,”Yes”))

#Splitting data into training and test sets.

split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)

#Fitting the Model
fitControl <– trainControl(method = ”cv”,number =10, summaryFunction= twoClassSummary,
                              classProbs=T, savePredictions = T)

lreg<–train(Outcome~.,data=training_set,method=”glm”,family=binomial(), trControl=fitControl)


#Making Predictions on test set
pred <–  predict(lreg,newdata = test_set,type=”prob”)

pred2 <–  predict(lreg,newdata = test_set,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome,pred2)

#ROC Curve
rocCurve.lreg <– roc(test_set$Outcome,pred[,”Yes”])

png(”ROC–Lreg.png”)
plot(rocCurve.lreg,col=c(4))
dev.off()

#AUC metric
auc(rocCurve.lreg)

#varImp
varImp(lreg)
```

A.3 K-Nearest Neighbours

```r
library(pROC)
library(caret)
library(caTools)
df = read.csv(’diabetes.csv’)  #importing dataset

#Removing missing data
df <- df[!(df$BloodPressure==0),]
df <- df[!(df$SkinThickness==0),]
df <- df[!(df$Glucose==0),]
df <- df[!(df$BMI==0),]
df <- df[!(df$Insulin==0),]

set.seed(06061968)

#Categorical Data
df$Outcome <- as.factor(df$Outcome)
df$Outcome <- factor(df$Outcome, labels = c(”No”,”Yes”))

#Splitting data into training and test sets.
split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)

fitControl <- trainControl(method = ”cv”,number =10, summaryFunction= twoClassSummary,
                             classProbs=T, savePredictions = T)
knnFit <- train(Outcome ˜ ., data = training_set, method = ”knn”, trControl = ctrl,
                  preProcess = c(”center”,”scale”))

#Making Predictions on test set
pred <-  predict(knnFit,newdata = test_set,type=”prob”)

pred2 <-  predict(knnFit,newdata = test_set,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome,pred2)
```

```
#ROC Curve
rocCurve.knn <- roc(test_set$Outcome,pred[,”Yes”])

png(”ROC-KNN.png”)
plot(rocCurve.knn,col=c(4))
dev.off()

#AUC metric
auc(rocCurve.knn)

#varImp
varImp(knnFit)
```

A.4 Support Vector Machines

A.4.1 Linear Kernel

```
library(caTools)
library(caret)
library(pROC)
df = read.csv('diabetes.csv')  #importing dataset

#Removing missing data
df <- df[!(df$BloodPressure==0),]
df <- df[!(df$SkinThickness==0),]
df <- df[!(df$Glucose==0),]
df <- df[!(df$BMI==0),]
df <- df[!(df$Insulin==0),]

set.seed(06061968)

#Categorical Data
df$Outcome <- as.factor(df$Outcome)
df$Outcome <- factor(df$Outcome,labels = c(”No”,”Yes”))

#Splitting data into training and test sets.
split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
```

```
test_set = subset(df, split == FALSE)

#Fitting the Model
fitControl <− trainControl(method = ”cv”,number =10, summaryFunction=
    twoClassSummary,
                                  classProbs=T, savePredictions = T)
svmfit_lin <−train(Outcome˜.,data=training_set,method=”svmLinear”,
    preProcess = c(”center”, ”scale”),
                        tuneLength = 10,trControl=fitControl)

#Making Predictions on test set
pred <−  predict(svmfit_lin,newdata = test_set,type=”prob”)

pred2 <−  predict(svmfit_lin,newdata = test_set,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome,pred2)

#ROC Curve
rocCurve.svm_lin <− roc(test_set$Outcome,pred[,”Yes”])

png(”ROC−SVML.png”)
plot(rocCurve.svm_lin,col=c(4))
dev.off()

#AUC metric
auc(rocCurve.svm_lin)

#varImp
varImp(svmfit_lin)
```

A.4.2 Radial Kernel

```
library(pROC)
library(caret)
library(caTools)
df = read.csv('diabetes.csv')  #importing dataset

#Removing missing data
```

```
df <- df[!(df$BloodPressure==0),]
df <- df[!(df$SkinThickness==0),]
df <- df[!(df$Glucose==0),]
df <- df[!(df$BMI==0),]
df <- df[!(df$Insulin==0),]

set.seed(06061968)

#Categorical Data
df$Outcome <- as.factor(df$Outcome)
df$Outcome <- factor(df$Outcome,labels = c(”No”,”Yes”))

#Splitting data into training and test sets.

split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)

#Fitting the Model
fitControl <- trainControl(method = ”cv”,number =10, summaryFunction=twoClassSummary,
                              classProbs=T, savePredictions = T)

svmfit_rad <-train(Outcome~.,data=training_set,method=”svmRadial”,preProcess = c(”center”, ”scale”),
                     trControl=fitControl)


#Making Predictions on test set
pred <-  predict(svmfit_rad,newdata = test_set,type=”prob”)

pred2 <-  predict(svmfit_rad,newdata = test_set,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome,pred2)

#ROC Curve
rocCurve.svm_rad <- roc(test_set$Outcome,pred[,”Yes”])

```

```
png(”ROC−SVMR.png”)
plot(rocCurve.svm_rad, col=c(4))
dev.off()

#AUC metric
auc(rocCurve.svm_rad)

#varImp
varImp(svmfit_rad)
```

A.5 Naïve Bayes

```
library(pROC)
library(caret)
library(caTools)
df = read.csv('diabetes.csv')  #importing dataset

#Removing missing data
df <− df[!(df$BloodPressure==0),]
df <− df[!(df$SkinThickness==0),]
df <− df[!(df$Glucose==0),]
df <− df[!(df$BMI==0),]
df <− df[!(df$Insulin==0),]

set.seed(06061968)

#Categorical Data
df$Outcome <− as.factor(df$Outcome)
df$Outcome <− factor(df$Outcome, labels = c(”No”,”Yes”))

#Splitting data into training and test sets.
split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)

#Fitting the Model
fitControl <− trainControl(method = ”cv”,number =10, summaryFunction=
    twoClassSummary,
                              classProbs=T, savePredictions = T)
```

```
nbfit <-train(Outcome~.,data=training_set,method=”nb”,preProcess = c(”center”, ”scale”),
              trControl=fitControl)

#Making Predictions on test set
pred <-  predict(nbfit,newdata = test_set,type=”prob”)

pred2 <-  predict(nbfit,newdata = test_set,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome,pred2)

#ROC Curve
rocCurve.nb <- roc(test_set$Outcome,pred[,”Yes”])

png(”ROC-NB.png”)
plot(rocCurve.nb,col=c(4))
dev.off()

#AUC metric
auc(rocCurve.nb)

#varImp
varImp(nbfit)
```

A.6 Decision Tree

```
library(pROC)
library(caret)
library(caTools)
library(party)
df = read.csv('diabetes.csv')  #importing dataset

#Removing missing data
df <- df[!(df$BloodPressure==0),]
df <- df[!(df$SkinThickness==0),]
df <- df[!(df$Glucose==0),]
df <- df[!(df$BMI==0),]
df <- df[!(df$Insulin==0),]
```

```

set.seed(06061968)

#Categorical Data
df$Outcome <– as.factor(df$Outcome)
df$Outcome <– factor(df$Outcome,labels = c(”No”,”Yes”))

#Splitting data into training and test sets.
split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)


#Fitting the Model
fitControl <– trainControl(method = ”cv”,number =10, summaryFunction=
    twoClassSummary,
                                  classProbs=T, savePredictions = T)
dtree<–train(Outcome˜ .,data=training_set,method=”ctree”,trControl=
    fitControl)

#Plotting the Decision Tree
png(filename = ”DT.png”, width = 1600, height = 1200)
plot(dtree$finalModel)
dev.off()

#Making Predictions on test set
pred <–  predict(dtree,newdata = test_set,type=”prob”)

pred2 <–  predict(dtree,newdata = test_set,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome,pred2)

#ROC Curve
rocCurve.dtree <– roc(test_set$Outcome,pred[,”Yes”])

png(”ROC–DT.png”)
plot(rocCurve.dtree,col=c(4))
dev.off()
```

```

#AUC metric
auc(rocCurve.dtree)

#varImp
varImp(dtree)
```

A.7 Random Forest

```
library(pROC)
library(caret)
library(caTools)
library(randomForest)
df = read.csv(’diabetes.csv’) #importing dataset

#Removing missing data
df <- df[!(df$BloodPressure==0),]
df <- df[!(df$SkinThickness==0),]
df <- df[!(df$Glucose==0),]
df <- df[!(df$BMI==0),]
df <- df[!(df$Insulin==0),]

set.seed(06061968)

#Categorical Data
df$Outcome <- as.factor(df$Outcome)
df$Outcome <- factor(df$Outcome,labels = c(”No”,”Yes”))

#Splitting data into training and test sets.
split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)

#Fitting the Model
fitControl <- trainControl(method = ”cv”,number =10, summaryFunction=twoClassSummary,
                                  classProbs=T, savePredictions = T)

rfFit <- train(Outcome ~ ., data = training_set, method = ”rf”,
    trControl = fitControl,
                   preProcess = c(”center”,”scale”))


#Making Predictions on test set
pred <-  predict(rfFit,newdata = test_set,type=”prob”)

pred2 <-  predict(rfFit,newdata = test_set,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome,pred2)

#ROC Curve
rocCurve.rf <- roc(test_set$Outcome,pred[,”Yes”])

png(”ROC-RF.png”)
plot(rocCurve.rf,col=c(4))
dev.off()

#AUC metric
auc(rocCurve.rf)

#varImp
varImp(rfFit)
```

A.8 Boosted Models

A.8.1 AdaBoost Classification Trees

```
library(pROC)
library(caret)
library(caTools)
df = read.csv('diabetes.csv')  #importing dataset

#Removing missing data
df <- df[!(df$BloodPressure==0),]
df <- df[!(df$SkinThickness==0),]
df <- df[!(df$Glucose==0),]
df <- df[!(df$BMI==0),]
df <- df[!(df$Insulin==0),]

set.seed(06061968)

#Categorical Data
df$Outcome <- as.factor(df$Outcome)
df$Outcome <- factor(df$Outcome,labels = c(”No”,”Yes”))

#Splitting data into training and test sets.
split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)


#Fitting the Model
fitControl <- trainControl(method = ”cv”, summaryFunction=twoClassSummary,
                              classProbs=T, savePredictions = T)

ab.fit <- train(Outcome~., data = training_set, method = ”adaboost”,
                   trControl = fitControl, metric = ”Accuracy”)

#Making Predictions on test set
pred <-  predict(ab.fit,newdata = test_set,type=”prob”)

pred2 <-  predict(ab.fit,newdata = test_set,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome,pred2)

#ROC Curve
rocCurve.ab <- roc(test_set$Outcome,pred[,”Yes”])

png(”ROC-ab.png”)
plot(rocCurve.ab,col=c(4))
dev.off()

#AUC metric
auc(rocCurve.ab)
```

```

#varImp
varImp(ab.fit)
```

A.8.2 eXtreme Gradient Boosting - Linear

```
library(pROC)
library(caret)
library(caTools)
df = read.csv('diabetes.csv')  #importing dataset

#Removing missing data
df <- df[!(df$BloodPressure==0),]
df <- df[!(df$SkinThickness==0),]
df <- df[!(df$Glucose==0),]
df <- df[!(df$BMI==0),]
df <- df[!(df$Insulin==0),]

set.seed(06061968)

#Categorical Data
df$Outcome <- as.factor(df$Outcome)
df$Outcome <- factor(df$Outcome,labels = c("No","Yes"))

#Splitting data into training and test sets.
split = sample.split(df$Outcome, SplitRatio = 0.75)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)


#Fitting the Model
fitControl <- trainControl(method = "cv", summaryFunction=
    twoClassSummary,
                              classProbs=T, savePredictions = T)

xgbL.fit <- train(Outcome~., data = training_set, method = "xgbLinear",
                  trControl = fitControl, metric = "Accuracy")

#Making Predictions on test set
```

```
pred <-  predict(xgbL.fit ,newdata = test_set ,type=”prob”)

pred2 <-  predict(xgbL.fit ,newdata = test_set ,type=”raw”)

#Confusion Matrix
confusionMatrix(test_set$Outcome ,pred2)

#ROC Curve
rocCurve.xgbL <- roc(test_set$Outcome ,pred[,”Yes”])

png(”ROC-xgbL.png”)
plot(rocCurve.xgbL,col=c(4))
dev.off()

#AUC metric
auc(rocCurve.xgbL)

#varImp
varImp(xgbL.fit)
```

Appendix B

Section B

B.1 Dealing with Missing Data

B.1.1 Records missing 1 variable

Predicting Fasting Plasma Glucose Concentration

```
library(caret)
library(caTools)
library(randomForest)
#loading datasets
df1 <- read.csv(”Complete.csv”)
df2 <- read.csv(”FBS.csv”)

#Removing Kidney Complications here as all are N & removing Annual Spending
df1 <- df1[,-c(2,17)]
df2 <- df2[,-c(2,17)]

#Splitting complete cases to test model
split = sample.split(df1$Fasting.Blood.Sugar, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

```

```
rfFit <- train(Fasting.Blood.Sugar ~ ., data = training_set, method = ”rf”, trControl = fitControl,
                 preProcess = c(”center”,”scale”))

#Making prediction on test set
pred <-  predict(rfFit,newdata = test_set,type=”raw”)
pred
test_set$Fasting.Blood.Sugar


MSE= c()
for (i in c(1:6)){
  a = (pred[i] - test_set$Fasting.Blood.Sugar[i])^2
  MSE = c(MSE,a)
}
sum(MSE)

pred2 = predict(rfFit,newdata = df2)
pred2

df2$Fasting.Blood.Sugar = pred2
write.csv(df2,”predicted.csv”)
```

Predicting HbA1C level

```
library(caret)
library(caTools)
library(randomForest)
#loading datasets
df1 <- read.csv(”Complete.csv”)
df2 <- read.csv(”HbA1c.csv”)

#Removing Kidney Complications here as all are N & removing Annual Spending
df1 <- df1[,-c(2,17)]
df2 <- df2[,-c(2,17)]

#Splitting complete cases to test model
split = sample.split(df1$Fasting.Blood.Sugar, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
```

```
test_set = subset(df1, split == FALSE)

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit <- train(HbA1C.level ~ ., data = training_set, method = ”rf”, trControl = fitControl,
                 preProcess = c(”center”,”scale”))

#Making prediction on test set
pred <-  predict(rfFit,newdata = test_set,type=”raw”)
pred
test_set$HbA1C.level


MSE= c()
for (i in c(1:6)){
  a = (pred[i] - test_set$HbA1C.level[i])^2
  MSE = c(MSE,a)
}
sum(MSE)

pred2 = predict(rfFit,newdata = df2)
pred2

df2$HbA1C.level = pred2
write.csv(df2,”predicted.csv”)
```

Predicting HDL Cholesterol level

```
library(caret)
library(caTools)
library(randomForest)
#loading datasets
df1 <- read.csv(”Complete.csv”)
df2 <- read.csv(”HDL.csv”)

#Removing Kidney Complications here as all are N & removing Annual Spending
```

```
df1 <- df1[,-c(2,17)]
df2 <- df2[,-c(2,17)]

#Splitting complete cases to test model
split = sample.split(df1$Fasting.Blood.Sugar, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit <- train(HDL.Cholesterol ~ ., data = training_set, method = ”rf”, trControl = fitControl,
               preProcess = c(”center”,”scale”))

#Making prediction on test set
pred <-  predict(rfFit,newdata = test_set,type=”raw”)
pred
test_set$HDL.Cholesterol


MSE= c()
for (i in c(1:6)){
  a = (pred[i] - test_set$HDL.Cholesterol[i])^2
  MSE = c(MSE,a)
}
sum(MSE)

pred2 = predict(rfFit,newdata = df2)
pred2

df2$HDL.Cholesterol = pred2
write.csv(df2,”predicted.csv”)
```

B.1.2 Records missing 2 variables

Predicting Blood Pressure Systolic and Diastolic

```
library(caret)
library(caTools)
library(randomForest)
#loading datasets
df1 <- read.csv(”Complete.csv”)
df2 <- read.csv(”BP(S), BP(D).csv”)

df2$Sex <- ”F”
df2$Sex <- as.factor(df2$Sex)

#Removing Kidney Complications here as all are N & removing Annual Spending
df1 <- df1[,-c(2,17)]
df2 <- df2[,-c(2,17)]


#Predicting BP(S)------------------------------------------------------------

#Splitting complete cases to test model
split = sample.split(df1$Blood.Pressure..Systolic., SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove BP(D) from the set
training_set = training_set[,-11]
test_set = test_set[,-11]

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit <- train(Blood.Pressure..Systolic. ~ ., data = training_set, method = ”rf”, trControl = fitControl,
                 preProcess = c(”center”,”scale”))

#Making prediction on test set
pred <-  predict(rfFit,newdata = test_set,type=”raw”)
pred
test_set$Blood.Pressure..Systolic.
```

```


MSE1= c()
for (i in c(1:5)){
  a = (pred[i] − test_set$Blood.Pressure..Systolic.[i])^2
  MSE1 = c(MSE1,a)
}

pred2 = predict(rfFit,newdata = df2)
pred2



#Predicting BP(D)——————————————————————————————

#Splitting complete cases to test model
split = sample.split(df1$Blood.Pressure..Diastolic., SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove BP(S) from the set
training_set = training_set[,−12]
test_set = test_set[,−12]

#Fitting the Model
set.seed(12071804)
fitControl <− trainControl(method = ”cv”,number =10)

rfFit <− train(Blood.Pressure..Diastolic. ˜ ., data = training_set,
    method = ”rf”, trControl = fitControl,
                    preProcess = c(”center”,”scale”))

#Making prediction on test set
pred3 <−  predict(rfFit,newdata = test_set,type=”raw”)
pred3
test_set$Blood.Pressure..Diastolic


MSE2= c()
```

```
for (i in c(1:5)){
  a = (pred3[i] - test_set$Blood.Pressure..Diastolic.[i])^2
  MSE2 = c(MSE2,a)
}

pred4 = predict(rfFit,newdata = df2)
pred4


sum(MSE2)
sum(MSE1)

df2$Blood.Pressure..Diastolic. = pred4
df2$Blood.Pressure..Systolic. = pred2

write.csv(df2,”predicted.csv”)
```

Predicting HDL and LDL Cholesterol

```
library(caret)
library(caTools)
library(randomForest)
#loading datasets
df1 <- read.csv(”Complete.csv”)
df2 <- read.csv(”HDL,LDL.csv”)

#Removing Kidney Complications here as all are N & removing Annual Spending
df1 <- df1[,-c(2,17)]
df2 <- df2[,-c(2,17)]


#Predicting HDL--------------------------------------------------

#Splitting complete cases to test model
split = sample.split(df1$HDL.Cholesterol, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove LDL from the set
```

```
training_set = training_set[,-14]
test_set = test_set[,-14]

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = "cv",number =10)

rfFit <- train(HDL.Cholesterol ~ ., data = training_set, method = "rf", trControl = fitControl,
                 preProcess = c("center","scale"))

#Making prediction on test set
pred <-  predict(rfFit,newdata = test_set,type="raw")
pred
test_set$HDL.Cholesterol


MSE1= c()
for (i in c(1:6)){
  a = (pred[i] - test_set$HDL.Cholesterol[i])^2
  MSE1 = c(MSE1,a)
}

pred2 = predict(rfFit,newdata = df2)
pred2



#Predicting LDL---------------------------------------------------------------

#Splitting complete cases to test model
split = sample.split(df1$LDL.Cholesterol, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove HDL from the set
training_set = training_set[,-13]
test_set = test_set[,-13]

```

```
#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit <- train(LDL.Cholesterol ~ ., data = training_set, method = ”rf”,
    trControl = fitControl,
                preProcess = c(”center”,”scale”))

#Making prediction on test set
pred3 <-  predict(rfFit,newdata = test_set,type=”raw”)
pred3
test_set$LDL.Cholesterol


MSE2= c()
for (i in c(1:6)){
  a = (pred3[i] - test_set$LDL.Cholesterol[i])^2
  MSE2 = c(MSE2,a)
}

pred4 = predict(rfFit,newdata = df2)
pred4


sum(MSE2)
sum(MSE1)

df2$HDL.Cholesterol = pred4
df2$LDL.Cholesterol = pred2

write.csv(df2,”predicted.csv”)
```

B.1.3 Records missing 3 variables

Predicting Post-Prandial Glucose Concentration, HDL and LDL Cholesterol levels

```
library(caret)
library(caTools)
library(randomForest)
```

```
#loading datasets
df1 <- read.csv(”Complete.csv”)
df2 <- read.csv(”PP,HDL,LDL.csv”)

#Removing Kidney Complications here as all are N & removing Annual Spending
df1 <- df1[,-c(2,17)]
df2 <- df2[,-c(2,17)]


#Predicting HDL--------------------------------------------------------

#Splitting complete cases to test model
split = sample.split(df1$HDL.Cholesterol, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove LDL,PP from the set
training_set = training_set[,-c(10,14)]
test_set = test_set[,-c(10,14)]

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit1 <- train(HDL.Cholesterol ~ ., data = training_set, method = ”rf”, trControl = fitControl,
                  preProcess = c(”center”,”scale”))

#Making prediction on test set
pred <-  predict(rfFit1,newdata = test_set,type=”raw”)
pred
test_set$HDL.Cholesterol


MSE1= c()
for (i in c(1:6)){
  a = (pred[i] - test_set$HDL.Cholesterol[i])^2
  MSE1 = c(MSE1,a)
}

pred2 = predict(rfFit1,newdata = df2)
pred2



#Predicting LDL--------------------------------------------------

#Splitting complete cases to test model
split = sample.split(df1$LDL.Cholesterol, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove HDL,PP from the set
training_set = training_set[,-c(10,13)]
test_set = test_set[,-c(10,13)]

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit2 <- train(LDL.Cholesterol ~ ., data = training_set, method = ”rf”,
     trControl = fitControl,
                 preProcess = c(”center”,”scale”))

#Making prediction on test set
pred3 <-  predict(rfFit2,newdata = test_set,type=”raw”)
pred3
test_set$LDL.Cholesterol


MSE2= c()
for (i in c(1:6)){
  a = (pred3[i] - test_set$LDL.Cholesterol[i])^2
  MSE2 = c(MSE2,a)
}

pred4 = predict(rfFit2,newdata = df2)
```

```
pred4

#Predicting PP------------------------------------------------------------

#Splitting complete cases to test model
split = sample.split(df1$PP.Sugar , SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove HDL,LDL from the set
training_set = training_set[,-c(13,14)]
test_set = test_set[,-c(13,14)]

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit3 <- train(PP.Sugar ~ ., data = training_set, method = ”rf”,
    trControl = fitControl,
                   preProcess = c(”center”,”scale”))

#Making prediction on test set
pred5 <-  predict(rfFit3,newdata = test_set,type=”raw”)
pred5
test_set$PP.Sugar


MSE3= c()
for (i in c(1:6)){
  a = (pred3[i] - test_set$PP.Sugar[i])^2
  MSE3 = c(MSE3,a)
}

pred6 = predict(rfFit3,newdata = df2)
pred6

sum(MSE2)
sum(MSE1)
sum(MSE3)
```

```

df2$HDL.Cholesterol = pred4
df2$LDL.Cholesterol = pred2
df2$PP.Sugar = pred6

write.csv(df2,”predicted.csv”)
```

Predicting HbA1C level, HDL and LDL Cholesterol levels

```
library(caret)
library(caTools)
library(randomForest)
#loading datasets
df1 <- read.csv(”Complete.csv”)
df2 <- read.csv(”HbA1c,HDL,LDL.csv”)

#Removing Kidney Complications here as all are N & removing Annual Spending
df1 <- df1[,-c(2,17)]
df2 <- df2[,-c(2,17)]


#Predicting HDL-----------------------------------------------------------

#Splitting complete cases to test model
split = sample.split(df1$HDL.Cholesterol, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove LDL,HbA1c from the set
training_set = training_set[,-c(8,14)]
test_set = test_set[,-c(8,14)]

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit1 <- train(HDL.Cholesterol ˜ ., data = training_set, method = ”rf”, trControl = fitControl,
                preProcess = c(”center”,”scale”))
```

```

#Making prediction on test set
pred <-  predict(rfFit1,newdata = test_set,type=”raw”)
pred
test_set$HDL.Cholesterol


MSE1= c()
for (i in c(1:6)){
  a = (pred[i] - test_set$HDL.Cholesterol[i])^2
  MSE1 = c(MSE1,a)
}

pred2 = predict(rfFit1,newdata = df2)
pred2



#Predicting LDL------------------------------------------------------------

#Splitting complete cases to test model
split = sample.split(df1$LDL.Cholesterol, SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove HDL,HbA1c from the set
training_set = training_set[,-c(8,13)]
test_set = test_set[,-c(8,13)]

#Fitting the Model
set.seed(12071804)
fitControl <- trainControl(method = ”cv”,number =10)

rfFit2 <- train(LDL.Cholesterol ~ ., data = training_set, method = ”rf”,
    trControl = fitControl,
                preProcess = c(”center”,”scale”))

#Making prediction on test set
pred3 <-  predict(rfFit2,newdata = test_set,type=”raw”)
```

```
pred3
test_set$LDL.Cholesterol


MSE2= c()
for (i in c(1:6)){
  a = (pred3[i] − test_set$LDL.Cholesterol[i])^2
  MSE2 = c(MSE2,a)
}

pred4 = predict(rfFit2 ,newdata = df2)
pred4

#Predicting HbA1c——————————————————————————————————

#Splitting complete cases to test model
split = sample.split(df1$HbA1C.level , SplitRatio = 2/3)
training_set = subset(df1, split == TRUE)
test_set = subset(df1, split == FALSE)

#We remove HDL,LDL from the set
training_set = training_set[,−c(13,14)]
test_set = test_set[,−c(13,14)]

#Fitting the Model
set.seed(12071804)
fitControl <− trainControl(method = ”cv”,number =10)

rfFit3 <− train(HbA1C.level ~ ., data = training_set , method = ”rf”,
    trControl = fitControl ,
                  preProcess = c(”center”,”scale”))

#Making prediction on test set
pred5 <−  predict(rfFit3 ,newdata = test_set ,type=”raw”)
pred5
test_set$HbA1C.level


MSE3= c()
```

```
for (i in c(1:6)){
  a = (pred3[i] − test_set$HbA1C.level[i])^2
  MSE3 = c(MSE3,a)
}

pred6 = predict(rfFit3 ,newdata = df2)
pred6

sum(MSE2)
sum(MSE1)
sum(MSE3)

df2$HDL.Cholesterol = pred4
df2$LDL.Cholesterol = pred2
df2$HbA1C.level = pred6

write.csv(df2,”predicted.csv”)
```

B.2 Preliminary Analysis

```
df <− read.csv(”Predicted.csv”)

library(ggplot2)
library(ggthemes)
library(corrgram)
library(dplyr)

#------------------------------------------------------------------------------
ggplot(df, aes(x = df$Age, y = df$Annual.Spending)) +geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”Age(in years)”) + ylab(”Annual Spending on Diabetes(in INR)”)+
  scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”Age.png”)

mean(df$Age)
aggregate(df$Age~df$Sex, FUN=mean)
```

```

sd(df$Age)
aggregate(df$Age˜df$Sex, FUN=sd)

quantile(df$Age)
aggregate(df$Age˜df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,3],df[df$Sex==”F”,3],var.equal = FALSE)

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

ggplot(df, aes(x = df$Height , y = df$Annual.Spending)) +geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”Height(in cms)”) + ylab(”Annual Spending on Diabetes(in INR)”)+
  scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”Height.png”)

mean(df$Height)
aggregate(df$Height˜df$Sex, FUN=mean)

sd(df$Height)
aggregate(df$Height˜df$Sex, FUN=sd)

quantile(df$Height)
aggregate(df$Height˜df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,5],df[df$Sex==”F”,5],var.equal = FALSE)

#--------------------------------------------------------------------------------
ggplot(df, aes(x = df$Weight , y = df$Annual.Spending)) +geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”Weight(in kgs)”) + ylab(”Annual Spending on Diabetes(in INR)”)+
  scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”Weight.png”)

mean(df$Weight)
```

```
aggregate(df$Weight~df$Sex, FUN=mean)

sd(df$Weight)
aggregate(df$Weight~df$Sex, FUN=sd)

quantile(df$Weight)
aggregate(df$Weight~df$Sex, FUN=quantile)

t.test(df[df$Sex=="M",6],df[df$Sex=="F",6],var.equal = FALSE)


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


ggplot(df, aes(x = df$BMI, y = df$Annual.Spending)) +geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab("Body Mass Index") + ylab("Annual Spending on Diabetes(in INR)")+
  scale_colour_discrete(name = "Sex", labels =c("Female","Male"))
ggsave("BMI.png")

mean(df$BMI)
aggregate(df$BMI~df$Sex, FUN=mean)

sd(df$BMI)
aggregate(df$BMI~df$Sex, FUN=sd)

quantile(df$BMI)
aggregate(df$BMI~df$Sex, FUN=quantile)

t.test(df[df$Sex=="M",7],df[df$Sex=="F",7],var.equal = FALSE)

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

ggplot(df, aes(x = df$Number.of.Pregnancies, y = df$Annual.Spending)) + geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab("NUmber of Pregnancies") +ylab("Annual Spending on Diabetes(in INR)")+
  scale_colour_discrete(name = "Sex", labels =c("Female","Male"))
```

```
ggsave(”Preg.png”)

mean(df$Number.of.Pregnancies)
aggregate(df$Number.of.Pregnancies~df$Sex, FUN=mean)

sd(df$Number.of.Pregnancies)
aggregate(df$Number.of.Pregnancies~df$Sex, FUN=sd)

quantile(df$Number.of.Pregnancies)
aggregate(df$Number.of.Pregnancies~df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,8],df[df$Sex==”F”,8],var.equal = FALSE)

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


ggplot(df, aes(x = df$HbA1C.level, y = df$Annual.Spending)) +geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”HbA1C Level(%)”) + ylab(”Annual Spending on Diabetes(in INR)”)+
  scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”HbA1C.png”)

mean(df$HbA1C.level)
aggregate(df$HbA1C.level~df$Sex, FUN=mean)

sd(df$HbA1C.level)
aggregate(df$HbA1C.level~df$Sex, FUN=sd)

quantile(df$HbA1C.level)
aggregate(df$HbA1C.level~df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,9],df[df$Sex==”F”,9],var.equal = FALSE)

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

ggplot(df, aes(x = df$Fasting.Blood.Sugar, y = df$Annual.Spending)) + geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”Fasting Plasma
    Glucose Conc.(mg/dl)”) +
  ylab(”Annual Spending on Diabetes(in INR)”)+scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”FPGC.png”)

mean(df$Fasting.Blood.Sugar)
aggregate(df$Fasting.Blood.Sugar˜df$Sex, FUN=mean)

sd(df$Fasting.Blood.Sugar)
aggregate(df$Fasting.Blood.Sugar˜df$Sex, FUN=sd)

quantile(df$Fasting.Blood.Sugar)
aggregate(df$Fasting.Blood.Sugar˜df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,10],df[df$Sex==”F”,10],var.equal = FALSE)

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

ggplot(df, aes(x = df$PP.Sugar, y = df$Annual.Spending)) +geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”Post-Prandial Glucose Conc.(mg/dl)”) +
  ylab(”Annual Spending on Diabetes(in INR)”)+scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”PPSugar.png”)

mean(df$PP.Sugar)
aggregate(df$PP.Sugar˜df$Sex, FUN=mean)

sd(df$PP.Sugar)
aggregate(df$PP.Sugar˜df$Sex, FUN=sd)

quantile(df$PP.Sugar)
aggregate(df$PP.Sugar˜df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,11],df[df$Sex==”F”,11],var.equal = FALSE)

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

```

```
ggplot(df, aes(x = df$Blood.Pressure..Diastolic., y = df$Annual.Spending)) +geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”Blood Pressure - Diastolic(mmHg)”) +
  ylab(”Annual Spending on Diabetes(in INR)”)+scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”BPD.png”)

mean(df$Blood.Pressure..Diastolic.)
aggregate(df$Blood.Pressure..Diastolic.˜df$Sex, FUN=mean)

sd(df$Blood.Pressure..Diastolic.)
aggregate(df$Blood.Pressure..Diastolic.˜df$Sex, FUN=sd)

quantile(df$Blood.Pressure..Diastolic.)
aggregate(df$Blood.Pressure..Diastolic.˜df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,12],df[df$Sex==”F”,12],var.equal = FALSE)

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

ggplot(df, aes(x = df$Blood.Pressure..Systolic., y = df$Annual.Spending)) +geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”Blood Pressure - Systolic(mmHg)”) +
  ylab(”Annual Spending on Diabetes(in INR)”)+scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”BPS.png”)

mean(df$Blood.Pressure..Systolic.)
aggregate(df$Blood.Pressure..Systolic.˜df$Sex, FUN=mean)

sd(df$Blood.Pressure..Systolic.)
aggregate(df$Blood.Pressure..Systolic.˜df$Sex, FUN=sd)

quantile(df$Blood.Pressure..Systolic.)
aggregate(df$Blood.Pressure..Systolic.˜df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,13],df[df$Sex==”F”,13],var.equal = FALSE)
```

```

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

ggplot(df, aes(x = df$HDL.Cholesterol, y = df$Annual.Spending)) + geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”HDL CHolesterol(mg/dl)”) +ylab(”Annual Spending on Diabetes(in INR)”)+
  scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”HDL.png”)

mean(df$HDL.Cholesterol)
aggregate(df$HDL.Cholesterol˜df$Sex, FUN=mean)

sd(df$HDL.Cholesterol)
aggregate(df$HDL.Cholesterol˜df$Sex, FUN=sd)

quantile(df$HDL.Cholesterol)
aggregate(df$HDL.Cholesterol˜df$Sex, FUN=quantile)

t.test(df[df$Sex==”M”,14],df[df$Sex==”F”,14],var.equal = FALSE)

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

ggplot(df, aes(x = df$LDL.Cholesterol, y = df$Annual.Spending)) + geom_point(aes(colour=df$Sex))+
  geom_smooth(method=loess)+theme_economist()+ xlab(”LDL Cholesterol(mg/dl)”) +ylab(”Annual Spending on Diabetes(in INR)”)+
  scale_colour_discrete(name = ”Sex”, labels =c(”Female”,”Male”))
ggsave(”LDL.png”)

mean(df$LDL.Cholesterol)
aggregate(df$LDL.Cholesterol˜df$Sex, FUN=mean)

sd(df$LDL.Cholesterol)
aggregate(df$LDL.Cholesterol˜df$Sex, FUN=sd)

quantile(df$LDL.Cholesterol)
aggregate(df$LDL.Cholesterol˜df$Sex, FUN=quantile)

```

```
t.test(df[df$Sex=="M",15],df[df$Sex=="F",15],var.equal = FALSE)

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

ggplot(df, aes(x = df$Eye.Complications)) + geom_bar(aes(fill=df$Sex), color="black")+ theme_economist()+xlab("Eye Complications")+
  ylab("Count")+scale_fill_discrete(name = "Sex", labels = c("Female","Male"))
ggsave("Eye.png")

df %>%
  group_by(df$Eye.Complications,df$Sex) %>%
  summarise(no_rows = length(Eye.Complications))

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

ggplot(df, aes(x = df$Kidney.Complications)) + geom_bar(aes(fill=df$Sex),color="black")+ theme_economist()+xlab("Kidney Complications")+
  ylab("Count")+scale_fill_discrete(name = "Sex", labels = c("Female","Male"))
ggsave("Kidney.png")
df %>%
  group_by(df$Kidney.Complications,df$Sex) %>%
  summarise(no_rows = length(Kidney.Complications))

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

ggplot(df, aes(x = df$Sex)) + geom_bar(aes(fill=df$Sex),color="black")+ theme_economist()+xlab("Kidney Complications")+
  ylab("Count")+theme(legend.position = "none")
ggsave("Sex.png")

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

ggplot(df, aes(x = df$Insulin)) + geom_bar(aes(fill=df$Sex),color="black")+ theme_economist()+xlab("Kidney Complications")+
  ylab("Count")+scale_fill_discrete(name = "Sex", labels = c("Female","Male"))
ggsave("Insulin.png")
```

```
df %>%
  group_by(df$Insulin,df$Sex) %>%
  summarise(no_rows = n())

#————————————————————————————————————————————

ggplot(df, aes(x = df$Annual.Spending)) + geom_histogram(aes(fill=df$Sex
    ),binwidth = 5000,color="black")+
  theme_economist()+xlab("Annual Spending(in INR)")+ theme(axis.text.x =
      element_text(angle=45,hjust=0.01))+
  ylab("Count")+scale_fill_discrete(name = "Sex", labels = c("Female","
    Male"))+
  scale_x_continuous(breaks = round(seq(min(df$Annual.Spending), max(
    df$Annual.Spending), by = 5000),1))
ggsave("Spending.png")
mean(df$Annual.Spending)
aggregate(df$Annual.Spending~df$Sex, FUN=mean)

sd(df$Annual.Spending)
aggregate(df$Annual.Spending~df$Sex, FUN=sd)

quantile(df$Annual.Spending)
aggregate(df$Annual.Spending~df$Sex, FUN=quantile)

t.test(df[df$Sex=="M",17],df[df$Sex=="F",17],var.equal = FALSE)

#————————————————————————————————————————————

png(filename="Corr.png")
corrgram(df, order=TRUE,
         main="Correlations between independent variables",
         lower.panel=panel.cor, upper.panel=panel.pie,
         diag.panel=panel.minmax, text.panel=panel.txt)
dev.off()
```

B.3 GAMs and GLMs

GLM

```
library(caTools)
library(caret)
library(ggplot2)
library(ggthemes)

df <- read.csv(”Predicted.csv”)

#Encoding categorical variables
df$Eye.Complications = factor(df$Eye.Complications, label = c(0,1))
df <- df[,-2]
df$Sex = factor(df$Sex, label = c(0,1))
df$Insulin = factor(df$Insulin, label = c(0,1))


#Splitting data into training and test sets.
set.seed(15031933)
split = sample.split(df$Annual.Spending, SplitRatio = 0.8)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)

#The Model (Gaussian)
ctrl <- trainControl(method = ”cv”, number = 10)

model <- train(log(Annual.Spending) ˜ .
                 ,data = training_set, method = ”glm”,family = gaussian(
    link=”identity”),trControl = ctrl)

summary(model)

1-(model$finalModel$deviance/model$finalModel$null.deviance)

png(”GLM1_Gaussian_Diag.png”, width = 1200, height = 1200)
par(mfrow=c(2,2))
plot(model$finalModel)
dev.off()
pred <- predict(model,newdata = test_set)

ggplot() + geom_point(aes(exp(model$finalModel$fitted.values),
    training_set$Annual.Spending)) +theme_economist()+
  xlab(”Fitted Values − Annual Spending(in INR)”)+ylab(”Observed Values − Annual Spending(in INR)”)+ xlim(0,65000)+
  ylim(0,65000)
ggsave(”GLM1_Gaussian_Fit.png”)

ggplot() + geom_point(aes(exp(pred),test_set$Annual.Spending)) + theme_economist()+
  xlab(”Predicted Values − Annual Spending(in INR)”)+ylab(”Observed Values − Annual Spending(in INR)”)+
  xlim(0,65000)+ylim(0,65000)
ggsave(”GLM1_Gaussian_Pred.png”)

ggplot() + geom_histogram(aes(model$finalModel$residuals), fill = ”white”, color=”black”, binwidth = 0.1)+
  theme_economist()+ xlab(”Residual Values”) + ylab(”Count”)
ggsave(”GLM1_Gaussian_Res.png”)
```

GAM

```
library(caTools)
library(mgcv)

df <− read.csv(”Predicted.csv”)
df <− df[,−2]

#Encoding categorical variables
df$Eye.Complications = factor(df$Eye.Complications, label = c(0,1))
df$Sex = factor(df$Sex, label = c(0,1))
df$Insulin = factor(df$Insulin, label = c(0,1))

#Splitting data into training and test sets.
split = sample.split(df$Annual.Spending, SplitRatio = 0.80)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)

#The Model

gamFit1 <− gam(Annual.Spending ˜ s(Age,k=3,fx=F, bs=”cr”) +
                 s(BMI,k=3,fx=F, bs=”cr”) +
                 s(HbA1C.level,k=3,fx=F, bs=”cr”) +
                 s(Fasting.Blood.Sugar,k=3,fx=F, bs=”cr”) +
                 s(PP.Sugar,k=3,fx=F, bs=”cr”)+
                 s(Blood.Pressure..Systolic.,k=3,fx=F, bs=”cr”)+
                 s(HDL.Cholesterol,k=3,fx=F, bs=”cr”)+
                 s(LDL.Cholesterol,k=3,fx=F, bs=”cr”) +
                 Sex +
                 Eye.Complications +
                 Insulin,
               family = gaussian,
               data=df)

gamFit1$sp

par(mfrow=c(2,4)) #to partition the Plotting Window
plot.gam(gamFit1)
dev.off()

predict_gamFit1 <- predict(gamFit1,test_set)
predict_gamFit1
test_set$Annual.Spending

plot(test_set$Annual.Spending, col=’green’, ylim=c(9500,50000), ylab = ”Amount”)
points(predict_gamFit1)

summary(gamFit1)


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

gamFit2 <- gam(Annual.Spending ~ s(Age,k=3,fx=F, bs=”cr”) +
                 s(BMI,k=3,fx=F, bs=”cr”) +
                 HbA1C.level +
                 Fasting.Blood.Sugar +
                 PP.Sugar+
                 Blood.Pressure..Systolic.+ Blood.Pressure..Diastolic.+
                 HDL.Cholesterol+
                 LDL.Cholesterol+
                   Sex +
                   Eye.Complications +
                   Insulin,
                 family = gaussian,
                 data=df)


gamFit2$sp

par(mfrow=c(1,2)) #to partition the Plotting Window
plot.gam(gamFit2)
dev.off()

predict_gamFit2 <- predict(gamFit2,test_set)
predict_gamFit2
test_set$Annual.Spending

plot(df$Annual.Spending, col=’green’, ylim=c(min(df$Annual.Spending),max(df$Annual.Spending)), ylab = ”Amount”)
points(gamFit2$fitted.values)

par(mfrow=c(2,2))
gam.check(gamFit1)

anova(gamFit1, gamFit2,test = ”Chisq”)

summary(gamFit2)
summary(gamFit1)$r.sq

plot(gamFit1, shade = TRUE)
```

Appendix C

Section C

C.1 Clustering

```
library(factoextra)
library(NbClust)
library(ggplot2)
library(ggthemes)
library(dendextend)
library(rpart)
library(rpart.plot)

df <- read.csv(”Data.csv”)

df3 <-sapply(df,as.numeric)
df3 <- scale(df3)
df3 <- as.data.frame(df3)

set.seed(14111889)

# Elbow method
png(”Elbow.png”)
fviz_nbclust(df3[,c(1:17)], kmeans, method = ”wss”) + theme_economist()+
  geom_vline(xintercept = 3, linetype = 2)+
  labs(subtitle = ”Elbow method”)
dev.off()

#30 indices
```

```
NbClust(df3[,c(1:16)], distance = ”euclidean”, method = ”kmeans”)

#Hierarchial Clustering
dist_mat <- dist(df3[,17], method = ’euclidean’)
hclust_avg <- hclust(dist_mat, method = ’average’)
cut_avg <- cutree(hclust_avg, k = 3)

avg_dend_obj <- as.dendrogram(hclust_avg)
avg_col_dend <- color_branches(avg_dend_obj, k = 3)
png(”HC.png”)

ggplot(avg_col_dend) + theme_economist() + xlab(”Cluster Number”) + ylab
    (”Height”)+
  ggtitle(”Three Clusters based on Hierarchial Clustering”)

dev.off()

#K-Means Clustering
scluster <- kmeans(df3[,c(17)],3)
df$SCluster <- scluster$cluster
df$SCluster <- as.factor(df$SCluster)

df4 <- df[,c(1:16)]

df4$Scluster <- df$SCluster
levels(df4$Scluster) <- c(”First”, ”Second”, ”Third”)

#Decision Tree
dtree <- rpart(Scluster~., df4, method=”class”, control = rpart.control(
    minbucket = 4, minsplit = 2))

png(filename = ”DT.png”)
rpart.plot(dtree)
dev.off()


ggplot() + geom_histogram(aes(df$Annual.Spending,..density.., fill =
    df4$Scluster), alpha = 0.4, binwidth = 2000)+
  geom_density(aes(df$Annual.Spending, fill= df4$Scluster), alpha = 0.4)
```

```
    +xlab("Annual Spending")+ylab("Density")+
  scale_fill_discrete(name = "Clusters")+ theme_economist()
ggsave("Annual Spending-Clustered.png")
```

Appendix D

Section D

D.1 Survey Questionnaire

Qualification Questions

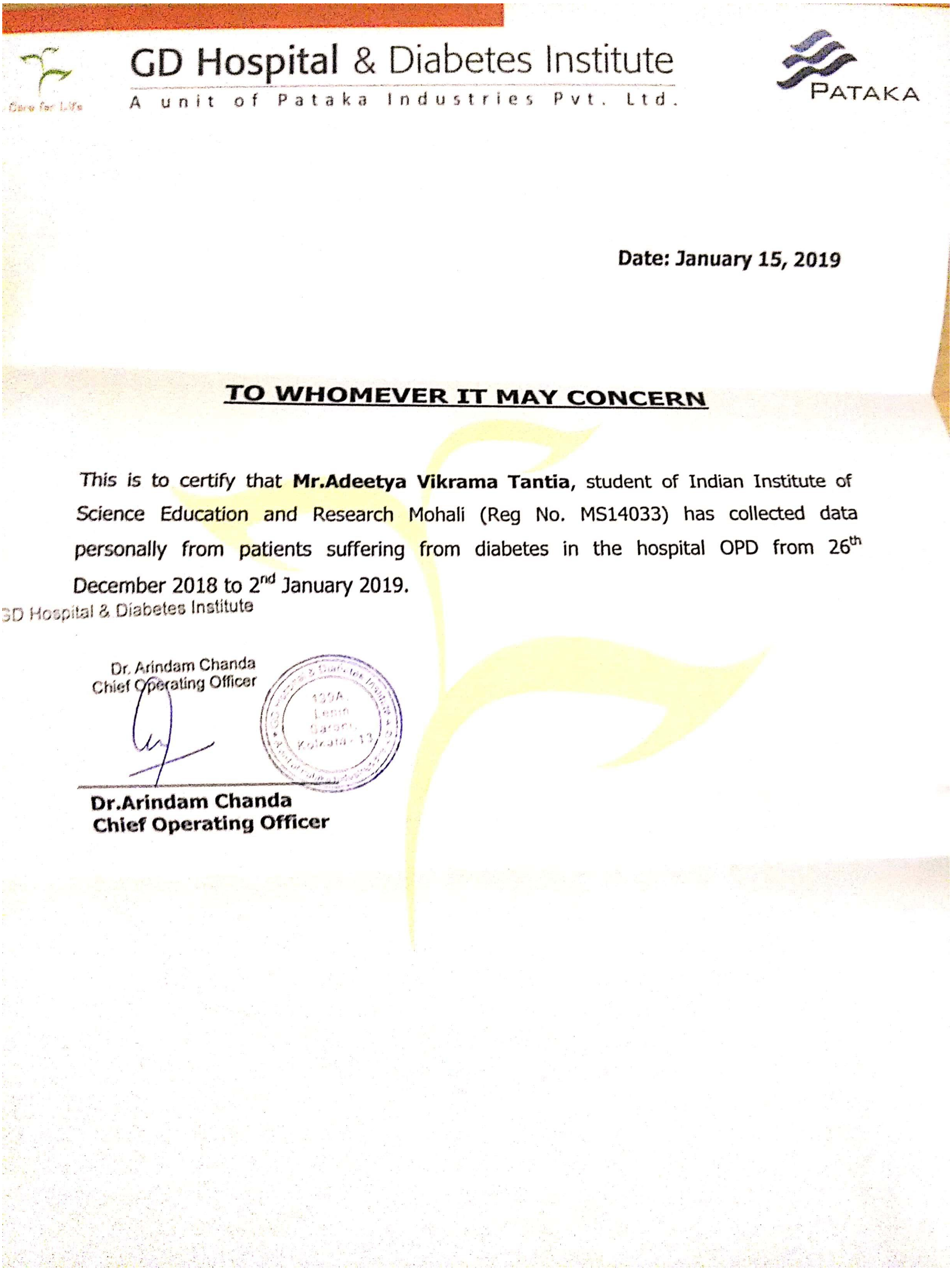
Do you have any complications relating to eyes?	
Do you have any complications relating to kidneys?	
Do you have any non-healing wounds on your body?	
What is your Body Mass Index (BMI)?	
What is your age?	

Part A - Medical Questions

Age (in Years as of 1st Jan 2019)	
Sex	
Height	
Weight	
Number of Pregnancies (for Females)	
Serum Creatinine level	
Albumin/ Creatinine Ratio	
HbA1C level	
Fasting Blood Sugar	
PP Sugar	
Blood Pressure (Diastolic)	
Blood Pressure (Systolic)	
HDL Cholesterol	
LDL Cholesterol	
Alkaline Phosphatase (ALP)	
Albumin/ Globulin Ratio	
Gamma GT	
Alanine Transaminase (ALT)	
Aspartate Aminotransferase (AST)	
Have you been diagnosed with diabetes ever?	
Are you on Insulin?	
Do you have any other diabetes-related complications?	

Part B - Financial Questions (Only to be answered by people who have diabetes)

Do you have a Diabetes specific Health Insurance plan?	
Which diabetes specific insurance plan do you have?	
How much insurance cover do you have?	
What is your current annual premium for this plan?	
How much did you claim from your insurer in health expenses relating to diabetes and complications in the past year?	
How much do you actually spend in health expenses relating to diabetes and complications per annum on average (incl. Lab tests, drugs and consultations)?	
Have you incurred any major expenses such as on surgery, hospital admissions etc. relating to diabetes? If Yes, how much and when?	



GD Hospital & Diabetes Institute

A unit of Pataka Industries Pvt. Ltd.

Care for Life

PATAKA

Date: January 15, 2019

TO WHOMEVER IT MAY CONCERN

This is to certify that **Mr.Adeetya Vikrama Tantia,** student of Indian Institute of Science Education and Research Mohali (Reg No. MS14033) has collected data personally from patients suffering from diabetes in the hospital OPD from 26th December 2018 to 2nd January 2019.

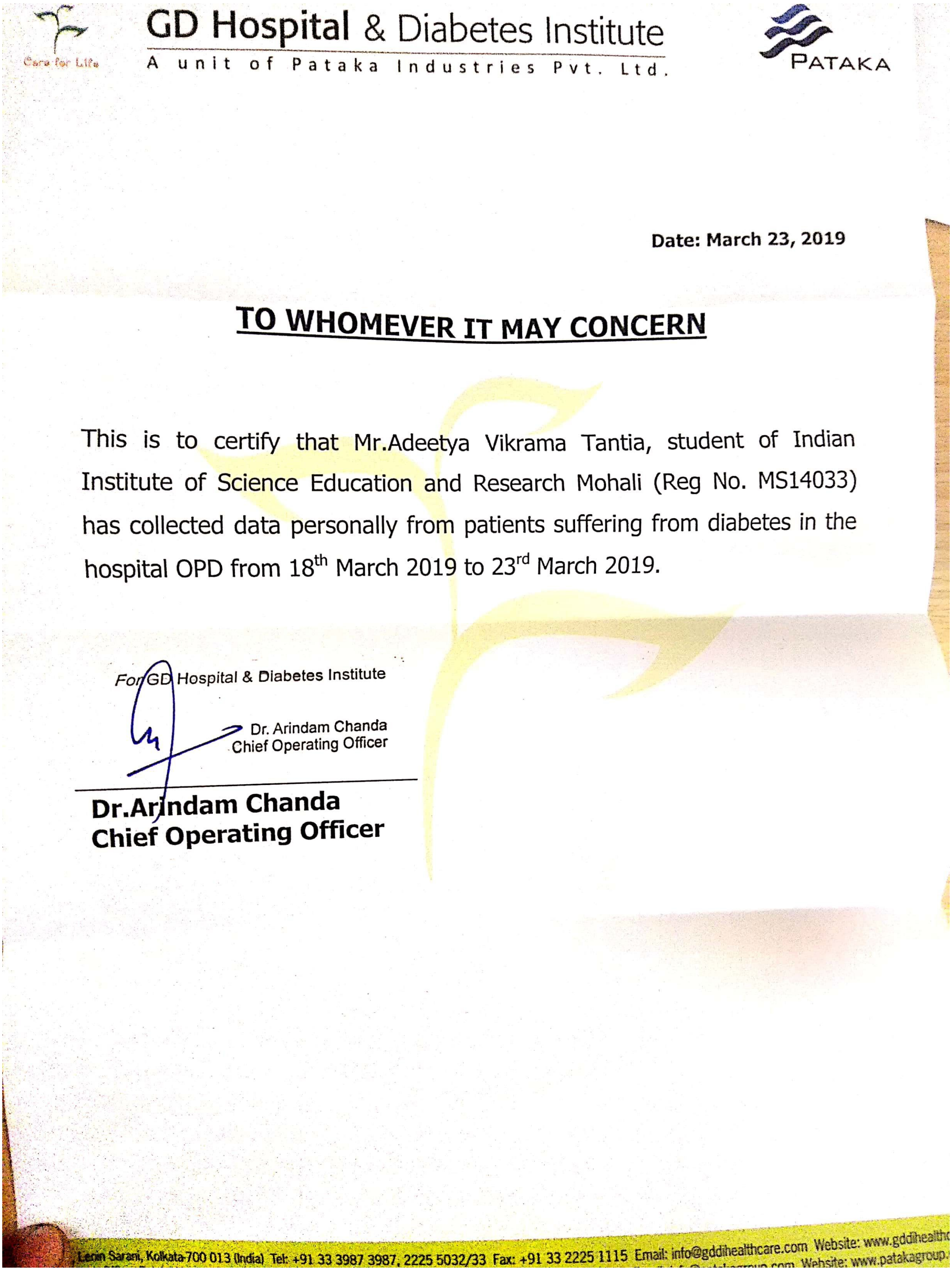
GD Hospital & Diabetes Institute

Dr. Arindam Chanda
Chief Operating Officer

139A, Lenin Sarani, Kolkata - 13

Dr.Arindam Chanda
Chief Operating Officer

Scanned by CamScanner



Care for Life

GD Hospital & Diabetes Institute

A unit of Pataka Industries Pvt. Ltd.

PATAKA

Date: March 23, 2019

TO WHOMEVER IT MAY CONCERN

This is to certify that Mr.Adeetya Vikrama Tantia, student of Indian Institute of Science Education and Research Mohali (Reg No. MS14033) has collected data personally from patients suffering from diabetes in the hospital OPD from 18th March 2019 to 23rd March 2019.

For GD Hospital & Diabetes Institute

Dr. Arindam Chanda
Chief Operating Officer

Dr.Arindam Chanda
Chief Operating Officer

Lenin Sarani, Kolkata-700 013 (India) Tel: +91 33 3987 3987, 2225 5032/33 Fax: +91 33 2225 1115 Email: info@gddihealthcare.com Website: www.gddihealth
com Website: www.patakagroup.

Scanned by CamScanner