Spring 2024

Classification in Supervised Statistical Learning With the New Weighted Newton-Raphson Method

Toma Debnath

CLASSIFICATION IN SUPERVISED STATISTICAL LEARNING WITH THE NEW WEIGHTED NEWTON-RAPHSON METHOD

by

TOMA DEBNATH

(Under the Direction of Divine Wanduku)

ABSTRACT

In this thesis, the Weighted Newton-Raphson Method (WNRM), an innovative optimization technique, is introduced in statistical supervised learning for categorization and applied to a diabetes predictive model, to find maximum likelihood estimates. The iterative optimization method solves nonlinear systems of equations with singular Jacobian matrices and is a modification of the ordinary Newton-Raphson algorithm. The quadratic convergence of the WNRM, and high efficiency for optimizing nonlinear likelihood functions, whenever singularity in the Jacobians occur allow for an easy inclusion to classical categorization and generalized linear models such as the Logistic Regression model in supervised learning. The WNRM is thoroughly investigated in the logistic regression model for both repeated and non-repeated response variables. Furthermore, the method is applied to obtain the best-fitted predictive logistic regression model for diabetes health status that depends on several predictor factors for the patients surveyed in a real-life study.

INDEX WORDS: Logistic regression, Weighted Newton-Raphon method, Optimization, Deviance, Akaike information criterion, Likelihood function

2009 Mathematics Subject Classification: 62-08, 62J12, 68P01

CLASSIFICATION IN SUPERVISED STATISTICAL LEARNING WITH THE NEW WEIGHTED NEWTON-RAPHSON METHOD

by

TOMA DEBNATH

B.S., University of Dhaka, Bangladesh, 2016

M.S., University of Dhaka, Bangladesh, 2017

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

CLASSIFICATION IN SUPERVISED STATISTICAL LEARNING WITH THE NEW WEIGHTED NEWTON-RAPHSON METHOD

by

TOMA DEBNATH

Major Professor:	Divine Wanduku
Committee:	Charles Champ
	Shijun Zheng

Electronic Version Approved:
May 2024

ACKNOWLEDGMENTS

I would like to express my utmost appreciation to my thesis advisor, Dr. Divine Wanduku, for her immense direction, assistance, and motivation during the journey of this research and thesis. In addition to improving my research skills, his instruction additionally developed within me the values of patience and persistence. I express deep gratitude for the opportunity to have Dr. Wanduku be my thesis supervisor.

I express my gratitude to the exceptional instructors and staff at Georgia Southern University, whose combined knowledge and competence have enhanced my academic journey. I would like to extend my sincere gratitude to Dr. Hua Wang and Dr. Yi Hu for their solid guidance, as well as to Dr. Jiehua Zhu, Dr. Yan Wu, Dr. Goran Lesaja, Dr. Emil Iacob and Dr. Scott Kersey for their consistent support and significant contributions to my educational journey. Their commitment to teaching is solid. I wish to extend my utmost gratitude to Dr. Charles Champ and Dr. Shijun Zheng for their generous willingness to serve as members of my thesis committee. Their profound understanding and specialized knowledge have significantly enhanced the standard of my thesis.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 SOME CONCEPTS AND ISSUES IN STATISTICAL LEARNING

Several statistical learning concepts that are essential to understanding the approaches used in this research, especially in relation to logistic regression, are begun with a thorough examination. Logistic regression is usually the most basic method in supervised statistical learning[4, 11] for data classification, and it uses optimization theory to estimate parameters to fit models to data. One particular optimization technique is the Newton-Raphson Method [13, 14, 15], extensively used in computational statistics for optimizing likelihood functions of statistical models to find *maximum likelihood estimates (mle)* of parameters, for example, in *Generalized linear models*, such as, the logistic, Poisson and quantile regression models [3]. The Newton-Raphson iterative optimization approach refines parameter estimates by using the gradient of the log-likelihood function to converge towards the maximum likelihood estimates. Despite the popularity of this iterative method, there are issues, such as, the existence of a singular Jacobian matrix near the mle or the solution of a system of equations, which has led to several important extensions of the Newton-Raphson methods to efficiently find the mle. [16, 17, 18, 19, 20, 5].

Apart from the Newton methods, there are other methods for solving systems of nonlinear equations such as the *tensor methods*[21]. In this study, an extension of the Newton methods called the **Weighted Newton-Raphson Method (WNRM)**[5] is revised, and applied for the first time in a statistical model to find mle for the model. Although, the logistic regression model has been extensively investigated, the issue of a singular Jacobian near the mle's of the model remains an important subject for investigation and poses a huge challenge in classifying the responses of the logistic regression model. While for most nonlinear systems of equations, the issue of the Jacobian is easily resolved by a correctly

selected initial solution [3], this is not always the case for most optimization problems in statistics, and there is a need for more advanced methods for optimizing likelihood functions.

In this study, the new iterative approach called the **Weighted Newton-Raphson Method (WNRM)** [5] is investigated in logistic regression analysis, for both repeated or non-repeated data response structures. The Weighted Newton-Raphson Method utilizes *weights* to overcome obstacles of the singular Jacobian matrix in logistic regression, whenever the ordinary Newton- Raphson method is inefficient. The reliability of parameter estimation is improved by this adaptation through the integration of modifications to the classic Newton-Raphson technique, ensuring consistent convergence even in complicated situations. In the subsequent subsections, some basic terminologies used in statistical learning are revised.

1.1.1 WHAT IS STATISTICAL LEARNING?

Statistical learning [4, 11] is the process of creating models with statistical techniques and algorithms in order to reach conclusions from data. Based on observed data, it involves determining the association between input variables—such as characteristics, predictors, or independent variables—and an output variable—often known to as the response or dependent variable. Preparing and data exploration, model selection, training, assessment, and deployment are all steps in this process. Statistical learning allows the development of exact models for outcome prediction and the comprehension of complicated relationships within datasets by using computer methods and statistical concepts. Estimating the function f has two primary objectives: prediction and inference. The goal of predictive modeling is to determine the connection between input variables X and an output variable Y, which is frequently challenging to measure directly. This relationship is

often written as

$$Y = f(X) + \epsilon, \tag{1.1}$$

where X is fixed and ϵ is the random error with mean zero. That is $E(\epsilon) = 0$.
We use $\hat{Y} = \hat{f}(X)$ to predict Y, where $\hat{f}$ is our estimated function and $\hat{Y}$ is the consequent prediction. This prediction's accuracy relies on two sorts of errors: **reducible and irreducible errors**. The reducible error depends on improving the structure of the component $f(X)$ to more accurately fit the data. The irreducible error depends on the random error component ϵ that occurs with the natural variability in the data. Consider a collection of predictors X and a given estimate $\hat{f}$, such that $\hat{Y} = \hat{f}(X)$. Assume that $\hat{f}$ and X are both fixed, meaning that ϵ is the only source of variability.

$$\begin{aligned} E(Y - \hat{Y})^2 &= E[f(X) + \epsilon - \hat{f}(X)]^2 \\ &= E\left[(f(X) + \epsilon - \hat{f}(X))^2\right] \\ &= \left[f(X) - \hat{f}(X)]^2 + 2[f(X) - \hat{f}(X)\right] E(\epsilon) + Var(\epsilon) \end{aligned}$$

$f(X)$ and $\hat{f}(X)$ are constant, $E(\epsilon) = 0$ and $E(\epsilon^2) = Var(\epsilon)$.

$$E(Y - \hat{Y})^2 = [f(X) - \hat{f}(X)]^2 + Var(\epsilon), \tag{1.2}$$

where $[f(X) - \hat{f}(X)]^2$ is the **reducible error** term, $Var(\epsilon)$ is the variance related to the **random error** term ϵ and $E(Y - \hat{Y})^2$ is the anticipated value of the squared expected difference between the predicted and actual value of Y. The difference between the estimated function $\hat{f}(X)$ and the true underlying function $f(X)$ causes this component of the error.

In the context of supervised learning, reducible error refers to inaccuracies in the model that can be minimized or eliminated through adjustments to the algorithm, refining the features, or increasing the size or quality of the training dataset. For example, a machine learning model is trained to predict the sales of a retail store based on factors like advertising expenditure, seasonality, and competitor activities. If the model fails to account for an

important predictor, such as the impact of local events or holidays, the error resulting from this oversight is reducible. By incorporating additional relevant features into the model, such as data on local events or holiday calendars, the model's predictive accuracy can be improved. On the other hand, **irreducible error** pertains to the inherent noise or randomness present in the data, which cannot be reduced regardless of the sophistication of the model or the size of the dataset. For instance, in predicting stock prices, even with the most advanced machine learning algorithms and extensive financial data, there will always be unpredictable market fluctuations and external events influencing stock movements, constituting irreducible error.

Unlike prediction, inference aims to comprehend the connection between the output variable Y and the input variables $X_1, X_2, ..., X_p$ without specifically generating predictions for Y. We seek to estimate the function f to understand the relationship between the variables. Within this particular framework: 1) The focus is on determining whether predictors are significantly linked to the response variable Y. 2) It is essential to comprehend the nature of the relationship between each predictor and the response variable, determining if it is positive, negative, or more complicated. 3) We will investigate if the relationship between the predictors and the response can be clearly expressed by a linear equation or if a more complicated model is required.

Assume an inference might involve understanding how teaching methods X influence student performance Y on standardized tests. Instead of solely aiming to predict individual students' test scores based on teaching techniques, the focus is on discerning the precise relationship between teaching methods and student achievement. By employing statistical models such as regression analysis, researchers can estimate coefficients to interpret the impact of various teaching strategies on test scores. For instance, if the coefficient for a particular teaching method is positive and statistically significant, it suggests that employing that method tends to lead to higher test scores. This understanding is vital

for educators and policymakers seeking to improve educational practices and outcomes.

1.1.2 SUPERVISED AND UNSUPERVISED LEARNING

Statistical learning problems are often classified as either supervised or unsupervised learning. Supervised learning implies observations, where each predictor measurement x_i, i = 1, 2, ..., n, is linked with a corresponding response measurement y_i. The purpose is to construct a model that can predict future responses effectively or improve the comprehension of the relation between predictors and responses. Classical methods include linear regression and logistic regression, along with contemporary approaches like GAM, boosting, and support vector machines, function within this framework. Unsupervised learning involves situations where only predictor measures x_i are present, without linked response variables y_i. The lack of a response variable causes difficulties, limiting the use of techniques such as linear regression.

It is essential in statistical learning to comprehend the connections between variables or observations. Cluster analysis, commonly referred to as clustering, is a powerful technique for this purpose. Cluster analysis aims to find out if data can be categorized into separate clusters according to their attributes. Market segmentation studies involve observing client attributes such as zip code, income, and buying patterns to discover different customer categories like big spenders and low spenders. Without detailed data on client spending habits, a supervised analysis cannot be conducted. Clustering assists in categorizing clients based on measured data, identifying unique segments that may vary in significant characteristics such as spending patterns.

1.1.3 REGRESSION AND CLASSIFICATION PROBLEMS

Regression is used to predict continuous results, like estimating house prices using factors like as square footage, number of rooms, and location. The purpose here is

to establish the connection between predictors and the continuous response variable via methods such as linear regression, polynomial regression, or sophisticated techniques like random forests. However, classification is concerned with forecasting categorical results, such identifying emails as spam or not, or diagnosing people as having a specific illness or not. Here, creating a model that can classify observations into specified groups according to input attributes is the goal. Logistic regression is a classification technique that predicts the probability of an observation fitting into a specific category by modeling a logistic function to the dataset. It is especially beneficial for addressing binary classification issues, which involve only two possibilities.

There are more classification techniques besides logistic regression. A member of the memory-based learning technique family is the **k-Nearest Neighbors (kNN)** algorithm. kNN is an example of an unsupervised classification technique, as as compared to logistic regression. It is a flexible and intuitive method, especially in situations when the decision boundary is nonlinear or the data distribution is complex, as it allocates newly collected data points to the category that most closely resembles that of their k nearest neighbors in the feature space.

1.1.4 GENERALIZED LINEAR MODELS

Generalized Linear Models (GLMs) in statistical modeling provide a more flexible framework than traditional multiple linear regression models by deviating from their assumptions to handle different sorts of response variables. GLMs differ from typical regression models by accommodating a wider variety of distributions and modifying the assumptions of normality and constant variance. GLMs are capable of handling response variables that do not follow the normal distribution assumption. This is especially beneficial when addressing non-normal response and variance inequality, which are prevalent issues in practical data analysis. Generalized Linear Models (GLMs) offer versatility by

accepting response variables that adhere to various distributions within the exponential family, such as normal, Poisson, binomial, exponential, and gamma distributions.

Furthermore, GLMs combine linear and nonlinear regression models, giving them a versatile tool for empirical modeling and data analysis. The normal-error linear model, a specific example of Generalized Linear Model (GLM), is merely one illustration of its wider range of applications. GLMs, like logistic regression, are used when the response variable has binary outcomes, such as success or failure. Logistic regression surpasses linear regression models by treating the response as qualitative, offering a strong framework for assessing categorical data. Another scenario where this method is useful is when the response variable involves numbers, such as errors in a product unit or infrequent events like Atlantic hurricanes hitting land. GLMs provide customized techniques, like Poisson regression, to properly model count data in certain situations.

Logistic regression is frequently utilized for categorical answer variables having two outcomes, such as success or failure, ”yes” or ”no”, or ”true” or ”false”. Visualize a research project examining the variables that impact a student's chances of university admission, such as their GPA and entrance exam results. Logistic regression can be used to estimate the probability of admission based on GPA and exam score, with the response variable being binary (admitted or not admitted).

Poisson regression is appropriate when the dependent variable indicates the frequency of occurrences happening within a specific time or space period. For example, in a manufacturing environment, a researcher may analyze the correlation between the quantity of errors in a product batch and different manufacturing characteristics. Poisson regression is used to estimate defect counts by considering variables like manufacturing speed, temperature, or raw material quality, to understand the factors affecting defect rates.

1.1.5 Outline of Thesis

The thesis is organized as follows: In Chapter 1, some concepts in statistical learning are discussed, including the definition of statistical learning, the distinction between supervised and unsupervised learning, the exploration of regression and classification problems, and the introduction to generalized linear models. In Chapter 2, a complete description of The Weighted Newton-Raphson Method (WNRM) is provided, and an example for the Weighted Newton-Raphson Method is presented. In Chapter 3, the assumption of the logistic regression model is discussed, and The Method of Maximum Likelihood estimation in the logistic regression model for non-repeated data as well as repeated data is derived. Additionally, the application of the Ordinary Newton Raphson method and Weighted Newton Raphson method is demonstrated. In Chapter 4, the binary classification of diabetes occurrence in the Pima Indians Diabetes Database is examined, including the logistic regression model for the diabetes data and the derivation of the Weighted Newton-Raphson algorithm. The application of the WNRM and the Reweighted Least Square method is explored, along with the interpretation of the Maximum Likelihood Estimates (MLEs) for the regression coefficients. Finally, methodologies for selecting the best predictive logistic regression model, such as Forward Selection and Backward Selection, are discussed and final conclusion is given in Chapter 4.

CHAPTER 2

THE WEIGHTED NEWTON-RAPHSON METHOD (WNRM)

2.1 INTRODUCTION

In this chapter the *ordinary Newton-Raphson algorithm (ONRM)* [5] is revised and a new and more efficient extension of the method called the *Weighted Newton-Raphson Method (WNRM)* [5] is introduced. Recall [5] , the *Newton-Raphson method* is a numerical approach to determine the real roots $\vec{\alpha} \in D$ of the nonlinear functions $F(\vec{x})$, where the function $F : D \subseteq \hat{R}^m \to R^m$, $m > 0$ and the vector

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ x_m \end{pmatrix}, F(\vec{x}) = \begin{pmatrix} f_1(\vec{x}) \\ f_2(\vec{x}) \\ . \\ . \\ f_m(\vec{x}) \end{pmatrix}.$$

That is, the method applies to solving the nonlinear equation $F(\vec{x}) = 0$. At the $(n+1)^{th}$ step in the ONRM, the solution of the system is given by

$$\vec{x}^{(n+1)} = \vec{x}^{(n)} - [J_F(\vec{x}^{(n)})]^{-1} F(\vec{x}^{(n)}), \tag{2.1}$$

where $J_F(\vec{x}^{(n)})$ is the Jacobian matrix at the n^{th} step. The following definition describes the order of convergence of an algorithm.

Definition 1. Let $\{\vec{x}^{(n)}\}_{k\geq 0}$ be a sequence in $\mathbb{R}^n$ convergent to $\vec{\alpha}$. Then, the convergence is said to be

(a) linear, if there exists M, $0 < M < 1$, and k_0 such that

$$\left\|\vec{x}^{(k+1)} - \vec{\alpha}\right\| \leq M \left\|\vec{x}^{(k)} - \vec{\alpha}\right\|, \quad \forall k \geq k_0.$$

(b) of order p, $p \geq 2$, if there exists M, $M > 0$, and k_0 such that

$$\left\|\vec{x}^{(k+1)} - \vec{\alpha}\right\| \leq M \left\|\vec{x}^{(k)} - \vec{\alpha}\right\|^p, \quad \forall k \geq k_0.$$

The following result describes the convergence order of the iteration (2.1). The *quadratic convergence* of the ONRM requires that the Jacobian $J_F(\vec{x}^{(n)})$ is non-singular near the solution, that is, $|J_F(\vec{x})| \neq 0$, near $\vec{\alpha} \in D$. Further recall [5], the order of convergence ρ of the iterations $\{\vec{x}^{(n)}\}_{n\geq 0}$ of an algorithm to the solution $\vec{\alpha}$ is approximated by the following

$$\rho \approx \frac{\ln(\left\|\vec{x}^{(k+1)} - \vec{\alpha}\right\|)/\left\|\vec{x}^{(k)} - \vec{\alpha}\right\|}{\ln(\|\vec{x}^{(k)} - \vec{\alpha}\|)/\left\|\vec{x}^{(k-1)} - \vec{\alpha}\right\|}, \forall k \geq 0. \tag{2.2}$$

In the following, an example is given to show the ONRM.

Example 2.1. *Consider the nonlinear system of equations below. Solve for the vector* $\vec{x} = (x, y, z)$.

$$\begin{aligned} &3x - cos(yz) - 0.5 = 0, \\ &x^2 - 625y^2 - 0.25 = 0, \\ &e^{-xy} + 20z + (10\pi - 3)/3 = 0. \end{aligned} \tag{2.3}$$

Solution 2.2. *Define the following functions*

$$\begin{aligned} f_1 &= 3x - cos(yz) - 0.5, \\ f_2 &= x^2 - 625y^2 - 0.25, \\ f_3 &= e^{-xy} + 20z + (10\pi - 3)/3, \end{aligned}$$

and compute the Jacobian matrix as follows.

$$[J_F(\vec{x})] = \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial z} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial z} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial z} \end{pmatrix} = \begin{pmatrix} 3 & zsin(yz) & ysin(yz) \\ 2x & -1250y & 0 \\ -ye^{-xy} & -xe^{-xy} & 20 \end{pmatrix}.$$

Applying the ONRM in (2.1), it is easy to see that

$$\vec{x}^{k+1} = \vec{x}^k - [J_F(\vec{x}^k)]^{-1} F(\vec{x}^k).$$

Select initial solution $\vec{x}^{(0)} = (0.2, 0.2, -0.2)^T$*; the solution is given by*

$$\hat{x} = 5.000000e - 01, \quad \hat{y} = 7.718786e - 11, \quad \hat{z} = -5.235988e - 01,$$

where the iteration converges after 31 *steps. The R-code for the solution is given in A.1.*

2.2 The Weighted Newton-Raphson Method (WNRM)

The ONRM fails, whenever the Jacobian matrix is singular. The WNRM is an iterative method designed for achieving quadratic convergence in the Newton-Raphson method, whenever the Jacobian $|J_F(\vec{x})| = 0$, near the solution $\vec{\alpha} \in D$.

2.2.1 Derivation of the Weighted Newton-Raphson Method (WNRM)

The WNRM [5] is derived in this section. The following result will be used to derive the WNRM.

Theorem 2.3. *Let* $G(x)$ *be a fixed point function with continuous partial derivatives of order* p *with respect to all components of the vector* $\vec{x} = (x_{j_1}, x_{j_2}, \ldots, x_{j_p})^T$*. The iterative method* $\vec{x}^{(k+1)} = G(\vec{x}^{(k)})$ *is of order* p*, if* $G(\vec{\alpha}) = \vec{\alpha}$*;*

$$\frac{\partial^k g_i(\vec{\alpha})}{\partial x_{j1}, \partial x_{j2}, \ldots, \partial x_{jk}} = 0, \tag{2.4}$$

for all $1 \leq k \leq p-1$*,* $1 \leq i, j_1, \ldots, j_k \leq n$*; and*

$$\frac{\partial^p g_i(\vec{\alpha})}{\partial x_{j1}, \partial x_{j2}, \ldots, \partial x_{jp}} = 0, \tag{2.5}$$

for at least one value of $i, j_1, \ldots, j_p$ *where* $g_i, i = 1, 2, \ldots, n$*, are the component functions of* G*.*

Proof. See [5]. □

The root $\vec{\alpha}$ of $F(\vec{x})$ is obtained by solving the equation $F(\vec{\alpha}) = 0$. Suppose $|J_F(\vec{x})| = 0$ near $\vec{\alpha} \in D$. In the *Ordinary Newton Raphson method*, the fixed point function is given by

$$G(\vec{x}) = \vec{x} - [J_F(\vec{x})]^{-1} F(\vec{x}). \tag{2.6}$$

The *Weighted Newton Raphson method* is derived by replacing the function G by

$$\hat{G}(\vec{x}) = \vec{x} - [J_F(\vec{x})]^{-1} M F(\vec{x}), \tag{2.7}$$

where M is a diagonal matrix used to establish convergence for the failed Newton-Raphson method. The diagonal element of $M = diag(m_1, m_2, \ldots, m_n)$, that is, m_i are called *weights* and given by

$$m_i = \frac{1}{1 - \frac{\partial g_i(\vec{\alpha})}{\partial x_i}}, i = 1, 2, \ldots, n, \tag{2.8}$$

and

$$G(\vec{x}) = (g_1(\vec{x}), g_2(\vec{x}), \ldots, g_n(\vec{x}))^T = \vec{x} - [J_F(\vec{x})]^{-1} F(\vec{x}). \tag{2.9}$$

Given the nonlinear system, $F(\vec{x}) = 0$,

$$F(\vec{x}) = (f_1(\vec{x}), \ldots, f_n(\vec{x}))^T \tag{2.10}$$

where $F(\vec{\alpha}) = 0$ and $|J_F(\vec{\alpha})| = 0$. Consider the auxiliary system,

$$\hat{F}(\vec{x}) = \left(e^{v_1 x_1} f_1(\vec{x})^{\frac{1}{m_1}}, \ldots, e^{v_n x_n} f_n(\vec{x})^{\frac{1}{m_n}} \right) = 0, \tag{2.11}$$

where $m_i > 0, i = 1, 2, \ldots, n, v_i \in \mathbb{R}, i = 1, 2, \ldots, n$ are chosen to obtain $|J_{\hat{F}}.(\vec{\alpha})| \neq 0$. Applying the ONRM in (2.1) to (2.11),

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - \left[J_{\hat{F}}(\vec{x}^{(k)}) \right]^{-1} \hat{F}(\vec{x}^{(k)}), \tag{2.12}$$

where the Jacobian matrix of $\hat{F}(\vec{x})$

$$J_{\hat{F}}(\vec{x}) = diag \left(e^{v_i x_i} \frac{1}{m_i} f_i(\vec{x})^{\frac{1}{m_i} - 1} \right) \times (diag(v_i m_i f_i(\vec{x}))) + J_F(\vec{x}). \tag{2.13}$$

Substituting (2.13) to (2.12) yields,

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - \left[diag(v_i m_i f_i(\vec{x}^{(k)})) + J_F(\vec{x}^{(k)})\right]^{-1} \times diag\left[m_i e^{-v_i x_i} f_i(\vec{x}^{(k)})^{(1-\frac{1}{m_i})}\right] \hat{F}(\vec{x}^{(k)})$$
$$= \vec{x}^{(k)} - \left[diag(v_i m_i f_i(\vec{x}^{(k)})) + J_F(\vec{x}^{(k)})\right]^{-1} diag(m_i)\hat{F}(\vec{x}^{(k)}) \quad (2.14)$$

The iteration (2.14) is the *Generalized Newton Raphson Method(GRN)*. Observe that (2.14) reduces to (2.15), when $v_i = 0, i = 1, 2, \ldots, n$

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - [J_F(\vec{x})]^{-1} diag(m_i) F(\vec{x}^{(k)}). \quad (2.15)$$

The iteration (2.15) is Weighted Newton Raphson Method (WNRM).

Theorem 2.4. *Under the above mentioned conditions, if $f_i(x) \in C^2(D) : D \subseteq R^n, \alpha \in D$ and $x^{(0)}$ is chosen sufficiently close to the solution, then the method defined by (2.15) has quadratic convergence.*

Proof. see [5] □

Algorithm 2.2.1. ***Algorithm for Weighted Newton Method***

Step-1: *Select initial solution $\vec{x}^0$;*

Step-2: *Replace the fixed point function $G(\vec{x}) = \vec{x} - [J_F(\vec{x})]^{-1} F(\vec{x})$ by the function $\hat{G}(x) = x - [J_F(x)]^{-1} MF(x)$, where M is the diagonal matrix with weights m_i.*

Step-3: *Using $\hat{G}(\vec{x}) = \vec{x} - [J_F(\vec{x})]^{-1} MF(\vec{x})$, find*

$$m_i = \frac{1}{1 - \frac{\partial g_i(\alpha)}{\partial x_i}},$$

$$\textit{where} \quad G(\vec{x}) = (g_1(\vec{x}), g_2(\vec{x}), ..., g_n(\vec{x}))^T = \vec{x} - [J_F(\vec{x})]^{-1} F(\vec{x}).$$

Step-4: *Apply the Weighted Newton Raphson Method is given by*

$$\vec{x}^{k+1} = \vec{x}^k - [J_F(\vec{x}^k)]^{-1} diag(m_i) F(\vec{x}^k), \forall k \geq 0,$$

where

$$\vec{x^k} = \begin{pmatrix} x_0^k \\ x_1^k \\ . \\ . \\ x_k^k \end{pmatrix}, \quad [J_F(\vec{x^k})] = \begin{pmatrix} \frac{\partial f_1}{\partial \beta_0} & \frac{\partial f_1}{\partial \beta_1} & .... & \frac{\partial f_k}{\partial \beta_k} \\ \frac{\partial f_2}{\partial \beta_0} & \frac{\partial f_2}{\partial \beta_1} & .... & \frac{\partial f_k}{\partial \beta_k} \\ . & . & . & . \\ . & . & . & . \\ \frac{\partial f_3}{\partial \beta_0} & \frac{\partial f_3}{\partial \beta_1} & .... & \frac{\partial f_k}{\partial \beta_k} \end{pmatrix}, and \quad F(\vec{x^k}) = \begin{pmatrix} f_1(x_0^k, x_1^k, ..., x_2^k) \\ f_2(x_0^k, x_1^k, ..., x_2^k) \\ . \\ . \\ f_k(x_0^k, x_1^k, ..., x_2^k) \end{pmatrix}.$$

Step-5: *If the discrepancy between the estimate in the current iteration $\vec{\beta}^{k+1}$ and the estimate in the previous iteration $\vec{\beta}^k$ is less than a predetermined tolerance level ϵ, stop the procedure and take the current estimate $\vec{\beta}$ as the root.*

Step-6: *Provide the ultimate approximation of the root.*

Apply the WNRM in Example 2.1 to find the solution $\vec{x} = (x, y, z)^T$.

2.2.2 Example for the Weighted-Newton Raphson Method

Example 2.5. *Solve the system below for $\vec{X} = (x, y, z)^T$.*

$$\begin{aligned} &3x - cos(yz) - 0.5 = 0, \\ &x^2 - 625y^2 - 0.25 = 0, \\ &e^{-xy} + 20z + (10\pi - 3)/3 = 0. \end{aligned} \tag{2.16}$$

Solution 2.6. *Define the following functions*

$$\begin{aligned} f_1 &= 3x - cos(yz) - 0.5, \\ f_2 &= x^2 - 625y^2 - 0.25, \\ f_3 &= e^{-xy} + 20z + (10\pi - 3)/3, \end{aligned}$$

$$[J_F(\vec{x})] = \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial z} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial z} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial z} \end{pmatrix} = \begin{pmatrix} 3 & zsin(yz) & ysin(yz) \\ 2x & -1250y & 0 \\ -ye^{-xy} & -xe^{-xy} & 20 \end{pmatrix}.$$

and compute the Jacobian matrix as follows.

$$[J_F(\vec{x})] = \begin{pmatrix} 3 & zsin(yz) & ysin(yz) \\ 2x & -1250y & 0 \\ -ye^{-xy} & -xe^{-xy} & 20 \end{pmatrix}.$$

Fixed point function

$$G(\vec{x}) = \vec{x} - [J_F(x)]^{-1}F(\vec{x}) = (g_1(\vec{x}), g_2(\vec{x}), ..., g_k(\vec{x}))^T.$$

The weights of the diagonal matrix $M = diag(m_i)$

$$m_i = \frac{1}{1 - \left(\frac{\partial g_i(\vec{x})}{\partial x_i}\right)}, M = diag(m_i) = \begin{pmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & 0 \\ 0 & 0 & m_{33} \end{pmatrix}.$$

Applying the Weighted Newton Raphson method in (2.15), it is easy to see that

$$\vec{x}^{k+1} = \vec{x}^k - [J_F(\vec{x}^k)]^{-1}M^k F(\vec{x}^k).$$

Select initial solution $\vec{x}^{(0)} = (0.2, 0.2, -0.2)^T$*; The Solution is*

$$\hat{x} = 5.000000e - 01$$

$$\hat{y} = 2.812312e - 09$$

$$\hat{z} = -5.235988e - 01,$$

where the iteration converges after 31 *steps. The R-code for the solution is given in A.2.*

CHAPTER 3

THE LOGISTIC REGRESSION MODEL

3.1 About the logistic regression model

The starting point for most statistical supervised learning methods for classification is the classical *logistic regression model*. In this chapter the logistic regression model is defined for both repeated and non-repeated response variables. Furthermore, the method of *Maximum Likelihood Estimation* is applied to find *Maximum likelihood estimates (MLE)* for the parameters of the logistic regression model by the iterative method of the Weighted Newton-Raphson Method introduced in Chapter 2.

3.2 Assumptions of the logistic model for non-repeated data

It is assumed that the p regressors $X_1, X_2, \ldots X_p$, where the i^{th} observation in a sample of size n is denoted by $X_{i1}, X_{i2}, \ldots X_{ip}$, $i = 1, 2, \ldots, n$. Also, for each $i = 1, 2, \ldots, n$, the response variable y_i takes on the value either 0 or 1 and y_i is a Bernoulli random variable. The cases y_i and the regressors $X_{i1}, X_{i2}, \ldots X_{ip}$, are related as follows.

$$\begin{aligned} y_i &= \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip} + \epsilon_i, \\ &= (\vec{X}_i)^T \vec{\beta} + \epsilon_i, \quad \forall i = 1, 2, \ldots, n \end{aligned} \tag{3.1}$$

where $(\vec{X}_i)^T = (1, X_{i1}, X_{i2}, \ldots X_{ip})$, and ϵ_i is the error random variable not satisfying the normality conditions as in the multiple linear regression model. In addition, the distribution of y_i, is given as follows.

$$P(y_i = 1) = \pi_i, \quad P(y_i = 0) = 1 - \pi_i \quad \forall i = 1, 2, \ldots, n. \tag{3.2}$$

The error variance is not constant, since from (3.1) $var(\epsilon_i) = var[y_i - (\vec{x}_i)^T\vec{\beta}] = var[y_i] = \sigma^2_{y_i}$, where

$$\begin{aligned}\sigma^2_{yi} &= E(y_i - E(y_i))^2 \\ &= (1-\pi_i)^2 P(y_i = 1) + (0-\pi_i)^2 P(y_i = 0) \\ &= (1-\pi_i)^2\pi_i + \pi_i^2(1-\pi_i) \\ &= \pi_i(1-\pi_i)(1-\pi_i+\pi_i) \\ &= \pi_i(1-\pi_i), \forall i \geq 1, 2, \ldots, n, \end{aligned} \tag{3.3}$$

and $0 \leq E(y_i) = \pi_i \leq 1$. Since the expected value $\mathbb{E}[y_i] = \pi_i \in (0,1)$ and nonlinear, then the logistic growth function is selected given as follows

$$\pi_i = \frac{e^{(\vec{X}_i)^T\vec{\beta}}}{1 + e^{(\vec{X}_i)^T\vec{\beta}}}. \tag{3.4}$$

From (3.4), it follows that the logit function is given by

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = (\vec{X}^i)^T\vec{\beta}. \tag{3.5}$$

Every sample observation has a probability distribution that is based on the Bernoulli distribution, which is

$$P(yi) = (\pi_i)^{y_i}(1-\pi_i)^{1-y_i}, \forall y_i = 0, 1; i = 0, 1, \ldots, n. \tag{3.6}$$

Note that for each $i = 1, 2, \ldots, n$, the random variables y_i are mutually independent Bernoulli random variables i.e. $y_1, y_2, \ldots, y_n$ are independent. Given the sample observations $\{y_1, y_2, \ldots, y_n\}$ of the Bernoulli distribution, the log-likelihood function for the parameter vector $\vec{\beta}$ in Section 3.1 is derived.

$$\begin{aligned} L(\vec{\beta}|y_1, y_2, \ldots, y_n) &= \prod_{i=1}^{n} p(y_i) \\ &= \prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{1-y_i} \end{aligned}$$

Taking logarithm function on both sides,

$$\log L(\vec{\beta}|y_1, y_2, \ldots, y_n) = \log \left[\prod_{i=1}^{n} \pi_i^{y_i} (1-\pi_i)^{1-y_i} \right] \quad = \sum_{i=1}^{n} y_i log(\pi_i) + \sum_{i=1}^{n} (1-y_i) log(1-\pi_i) \tag{3.7}$$

Using equation (3.5) into (3.7),

$$\log L(\vec{\beta}|y_1, y_2, \ldots, y_n) = \sum_{i=1}^{n} y_i (\vec{X}_i)^T \vec{\beta} - \sum_{i=1}^{n} \log(1 + e^{(\vec{X}_i)^T \vec{\beta}}). \tag{3.8}$$

In the next section, the ONRM and WNRM are applied to optimize (3.8) to find MLE for $\vec{\beta}$.

3.3 The Method of Maximum Likelihood estimation in the logistic regression model for non-repeated data

In this section the log-likelihood function (3.8) is optimized by applying both the ONRM and the WNRM. Recall (3.8)

$$\log L(\vec{\beta}|y_1, y_2, \ldots, y_n) = \sum_{i=1}^{n} y_i (\vec{X}_i)^T \vec{\beta} - \sum_{i=1}^{n} \log(1 + e^{(\vec{X}_i)^T \vec{\beta}}) \tag{3.9}$$

Optimizing the function (3.9) consist of solving the system of equations below.

$$\begin{cases} \frac{\partial \log}{\partial \beta_0}(L(\vec{\beta}|y_1, y_2, \ldots, y_p)) = 0 \\ \frac{\partial log}{\partial \beta_1}(L(\vec{\beta}|y_1, y_2, \ldots, y_p)) = 0 \\ . \\ . \\ \frac{\partial \log}{\partial \beta_p}(L(\vec{\beta}|y_1, y_2, \ldots, y_p)) = 0. \end{cases} \tag{3.10}$$

Let $f_j(\vec{\beta}), j = 1, 2, \ldots, p$ be as follows.

$$f_j(\vec{\beta}) = \frac{\partial log L(\vec{\beta}|y_1, \ldots, y_n)}{\partial \beta_j}, j = 0, 1, 2, \ldots, p \tag{3.11}$$

and

$$F(\vec{\beta}) = (f_0(\vec{\beta}), f_1(\vec{\beta}), \ldots, f_p(\vec{\beta}))^T. \tag{3.12}$$

The system (3.10) reduces to

$$F(\vec{\beta}) = 0. \tag{3.13}$$

The ordinary Newton Raphson method is given by,

$$\vec{\beta}^{k+1} = \vec{\beta}^k - [J_F(\vec{\beta}^k)]^{-1} F(\vec{\beta}^k), \tag{3.14}$$

and the WNRM is given by

$$\vec{\beta}^{k+1} = \vec{\beta}^k - [J_{\hat{F}}(\vec{\beta}^k)]^{-1} M F(\vec{\beta}^k), \tag{3.15}$$

where

$$\vec{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ . \\ . \\ \beta_p \end{pmatrix}, [J_F(\vec{\beta})] = \begin{pmatrix} \frac{\partial f_1}{\partial \beta_0} & \frac{\partial f_1}{\partial \beta_1} & \frac{\partial f_1}{\partial \beta_2} \\ \frac{\partial f_2}{\partial \beta_0} & \frac{\partial f_2}{\partial \beta_1} & \frac{\partial f_2}{\partial \beta_2} \\ . & & \\ . & & \\ \frac{\partial f_3}{\partial \beta_0} & \frac{\partial f_3}{\partial \beta_1} & \frac{\partial f_3}{\partial \beta_2} \end{pmatrix}, F(\vec{\beta}) = \begin{pmatrix} f_1(\beta_0, \beta_1, \beta_2) \\ f_2(\beta_0, \beta_1, \beta_2) \\ . \\ . \\ f_3(\beta_0, \beta_1, \beta_2) \end{pmatrix},$$

$$G(\vec{\beta}) = (g_1(\vec{\beta}), ..., g_n(\vec{\beta}))^T = \vec{\beta} - [J_F(\vec{\beta})]^{-1} F(\vec{\beta}), \tag{3.16}$$

$$M = diag(m_i), \quad m_i = \frac{1}{1 - \frac{\partial g_i(\alpha)}{\partial \beta_i}} \tag{3.17}$$

The steps for the ONRM in logistic regression are summarized in the algorithm below.

Algorithm 3.3.1. ***Algorithm for ONRM in logistic regression***

Step-1: *Begin by selecting an initial estimate $\beta_0, \beta_1, \beta_2$ for the root of the function.*

Step-2: *Compute the function value* $F(\vec{\beta}) = (f_0(\vec{\beta}), f_1(\vec{\beta}), f_2(\vec{\beta}))^T$ *and its derivative* $\frac{\partial f(\vec{\beta})}{\partial \beta_j}$, $\forall j = 0, 1, 2$ *to find the Jacobian matrix* $[J_F(\vec{\beta})]$ *at the* k^{th} *iteration* $\vec{\beta}^k$. *Furthermore, apply the following iteration to find the solution at the* $(k+1)^{th}$ *step.*

$$\vec{\beta}^{k+1} = \vec{\beta}^k - [J_F(\vec{\beta})]^{-1} F(\vec{\beta}^k) \tag{3.18}$$

Step-3: *If the discrepancy between the estimate in the current iteration* $\vec{\beta}^{k+1}$ *and the estimate in the previous iteration* $\vec{\beta}^k$ *is less than a predetermined tolerance level* ϵ. *Stop the procedure and take the current estimate* $\vec{\beta}$ *as the root.*

Step-4: *Provide the ultimate approximation of the root.*

Also, the steps for the WNRM in logistic regression based in the iteration in (3.15) are summarized in the algorithm below.

Algorithm 3.3.2. ***Algorithm for WNRM in logistic regression***

Step-1: *Begin by selecting an initial estimate* $\beta_0, \beta_1, \beta_2$ *for the root of the function.*

Step-2: *Compute the function value* $F(\vec{\beta}) = (f_0(\vec{\beta}), f_1(\vec{\beta}), f_2(\vec{\beta}))^T$ *and its derivative* $\frac{\partial f(\vec{\beta})}{\partial \beta_j}$, $\forall j = 0, 1, 2$ *to find the Jacobian matrix* $[J_F(\vec{\beta})]$ *at the* k^{th} *iteration* $\vec{\beta}^k$.

Step-3: *At* $(k+1)^{th}$ *step, define* $\hat{G}(\beta) = \vec{\beta} - [J_F(\vec{\beta})]^{-1} M F(\vec{\beta})$ *applied to system* $F(\vec{\beta}) = 0$, *where* $M = diag(m_i)$,

$$m_i = \frac{1}{1 - \frac{\partial g_i(\alpha)}{\partial \beta_i}}, G(\vec{\beta}) = (g_1(\vec{\beta}), ..., g_n(\vec{\beta}))^T = \vec{\beta} - [J_F(\vec{\beta})]^{-1} F(\vec{\beta}).$$

Step-4: *If the discrepancy between the estimate in the current iteration* $\vec{\beta}^{k+1}$ *and the estimate in the previous iteration* $\vec{\beta}^k$ *is less than a predetermined tolerance level* ϵ, *stop the procedure and take the current estimate* $\vec{\beta}$ *as the root.*

Step-5: *Provide the ultimate approximation of the root.*

In the next section Algorithms 3.3.1 & 3.3.2 are applied to an example for a logistic regression model. It is shown that the ONRM is limited for finding MLE's for the parameter $\vec{\beta}$, while the WNRM is more efficient, whenever the Jacobian matrix is singular.

3.4 EXAMPLE OF MAXIMUM LIKELIHOOD METHOD IN LOGISTIC REGRESSION WITH NON-REPEATED DATA

In this section, an example of a multiple logistic regression model is given; and the ONRM and the WNRM are applied to find MLE for $\vec{\beta}$. Consider the model

$$y_i = \beta_0 + \beta_1 x^2 + \beta_2 e^x + \epsilon_i, \tag{3.19}$$

where $i = 1, 2, \ldots, n$;

$$Var(\epsilon_i) = \sigma_{y_i} = \pi_i(1 - \pi_i),$$

and

$$E(y_i) = \pi_i, \forall i = 1, 2, \ldots, n.$$

The iterative methods are applied to fit the model (3.19) to the data in Table (3.1).

y	x
0	1.5
1	2
0	3.5
1	2.5
1	1
0	1.3

Table 3.1: Bivariate data for the example of non-repeated data logistic regression model.

From (3.5), it is easy to see that,

$$E(y_i) = \frac{e^{\beta_0 + \beta_1 x_i^2 + \beta_2 e_i^x}}{1 + e^{\beta_0 + \beta_1 x_i^2 + \beta_2 e_i^x}}, i = 1, 2, \ldots, n. \tag{3.20}$$

From (3.10), the MLE of $\vec{\beta} = (\beta_0, \beta_1, \beta_2)^T$ is obtained by solving the system (3.13). Observe from (3.7) and (3.10) that the log-likelihood for the model (3.19) is expressed as follows.

$$\log L(y_1, y_2, ..., y_n, \vec{\beta}) = \sum_{i=1}^{n} y_i(\beta_0 + \beta_1 x^2 + \beta_2 e^x) - \sum_{i=1}^{n} \log(\frac{1}{1 + e^{\beta_0 + \beta_1 x^2 + \beta_2 e^x}}). \quad (3.21)$$

Applying (3.11),(3.12) and (3.13) to (3.21), it is easy to see that the system of equations,

$$f_j(\vec{\beta}) = \frac{\partial log L(\vec{\beta}|y_1, y_2, \ldots, y_n)}{\partial \beta_j} = 0, j = 0, 1, 2 \quad (3.22)$$

is written as

$$F(\vec{\beta}) = (f_0(\vec{\beta}), f_1(\vec{\beta}), f_2(\vec{\beta})) = 0, \quad (3.23)$$

where

$$\begin{aligned}
f_0(\beta_0, \beta_1, \beta_2) &= \sum_{i=1}^{n} \left[y_i - (1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^{-1} \right], \\
f_1(\beta_0, \beta_1, \beta_2) &= \sum_{i=1}^{n} \left[y_i - (1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^{-1} \right] (x_i^2), \\
f_2(\beta_0, \beta_1, \beta_2) &= \sum_{i=1}^{n} \left[y_i - (1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^{-1} \right] (e^{x_i}),
\end{aligned} \quad (3.24)$$

and

$$\begin{aligned}
&\frac{\partial f_0}{\partial \beta_0} = -\sum_{i=1}^{n} \left[\frac{e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right], \frac{\partial f_0}{\partial \beta_1} = -\sum_{i=1}^{n} \left[\frac{x_i^2 e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right], \\
&\frac{\partial f_0}{\partial \beta_2} = -\sum_{i=1}^{n} \left[\frac{e^{x_i} e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right], \frac{\partial f_1}{\partial \beta_0} = -\sum_{i=1}^{n} \left[\frac{x_i^2 e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right], \\
&\frac{\partial f_1}{\partial \beta_1} = -\sum_{i=1}^{n} \left[\frac{x_i^4 e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right], \frac{\partial f_1}{\partial \beta_2} = -\sum_{i=1}^{n} \left[\frac{x_i^2 e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}} e^{x_i}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right], \\
&\frac{\partial f_2}{\partial \beta_0} = -\sum_{i=1}^{n} \left[\frac{e^{x_i} e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right], \frac{\partial f_2}{\partial \beta_1} = -\sum_{i=1}^{n} \left[\frac{x_i^2 e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}} e^{x_i}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right], \\
&\frac{\partial f_2}{\partial \beta_2} = -\sum_{i=1}^{n} \left[\frac{e^{2x_i} e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}}}{(1 + e^{-\beta_0 - \beta_1 x_i^2 - \beta_2 e^{x_i}})^2} \right].
\end{aligned} \quad (3.25)$$

3.4.1 APPLICATION OF THE ONRM

Applying the steps of Algorithm 3.3.1 to solve (3.23) for $\vec{\beta}$, the following are obtained.

(1.) In **Step 1**, select $\vec{\beta}^{(0)} = (\beta_0, \beta_1, \beta_2) = (1, 2, 3)$.

(2.) In **Step 2**, compute the function value $F(\vec{\beta}) = (f_0(\vec{\beta}), f_1(\vec{\beta}), f_2(\vec{\beta}))^T$ in (3.23) and its derivative $\frac{\partial f_j(\vec{\beta})}{\partial \beta_j}$, $\forall j = 0, 1, 2$ in (3.25) to find the Jacobian matrix $[J_F(\vec{\beta})]$ at the k^{th} iteration $\vec{\beta}^k$.

(3.) In **Step 3**, apply the ONRM iteration below to find the solution at the $(k+1)^{th}$ step.

$$\vec{\beta}^{k+1} = \vec{\beta}^k - [J_F(\vec{\beta})]^{-1} F(\vec{\beta}^k). \tag{3.26}$$

(4.) In **Step-4:** if the discrepancy between the estimate in the current iteration $\vec{\beta}^{k+1}$ and the estimate in the previous iteration $\vec{\beta}^k$ is less than a predetermined tolerance level ϵ. Stop the procedure and take the current estimate $\vec{\beta}$ as the root.

(5.) **Step-5:** Provide the ultimate approximation of the root.

Applying steps (1.)-(4.) above for the data in Table 3.1 the results of the ORNM are given below for 6 iterations.

iteration	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	Error	MSE
1	2.236146e+12	2.309280e+12	-1.672176e+12	1.672176e+12	NaN
2	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN

Table 3.2: Results for the ONRM.This table shows the estimated 1st iteration for $\vec{\beta} = (\beta_0, \beta_1, \beta_2)^T$ at each iteration $k = 1, 2, \ldots, 6$. The error statistics measures the distance $||\vec{\beta}^{k+1} - \vec{\beta}^k||_{max} = \max_{0 \leq j \leq 2} |\beta_j^{k+1} - \beta_j^k|$, where $||\vec{\beta}^{k+1} - \vec{\beta}^k||_{max} \to 0$, as $k \to \infty$ implies convergence. Subsequent iterations yield 'NaN' values for the estimated coefficients $\hat{\beta}0$, $\hat{\beta}1$, and $\hat{\beta}_2$, indicating the inability to converge towards a solution.

Remark 3.1. *Observe from Table 3.2 that the ONRM does not converge, and a singular Jacobian matrix is obtained after the first iteration. This necessitates a more efficient iterative method for finding MLE's in the logistic regression. The computer code for the solution is given in A.3.*

3.4.2 Application of the WNRM

Applying the steps of Algorithm 3.3.2 to solve (3.23) for $\vec{\beta}$, the following are obtained.

(1.) In **Step 1**, select $\vec{\beta}^{(0)} = (\beta_0, \beta_1, \beta_2) = (1, 2, 3)$.

(2.) In **Step 2**, compute the function value $F(\vec{\beta}) = (f_0(\vec{\beta}), f_1(\vec{\beta}), f_2(\vec{\beta}))^T$ in (3.23) and its derivative $\frac{\partial f_j(\vec{\beta})}{\partial \beta_j}$, $\forall j = 0, 1, 2$ in (3.25) to find the Jacobian matrix $[J_F(\vec{\beta})]$ at the k^{th} iteration $\vec{\beta}^k$.

(3.) In **Step 3**, Define $\hat{G}(\beta) = \vec{\beta} - [J_F(\vec{\beta})]^{-1} MF(\vec{\beta})$ applied to system $F(\vec{\beta}) = 0$, where $M = diag(m_i)$,

$$m_i = \frac{1}{1 - \frac{\partial g_i(\alpha)}{\partial \beta_i}}, G(\vec{\beta}) = (g_1(\vec{\beta}), ..., g_n(\vec{\beta}))^T = \vec{\beta} - [J_F(\vec{\beta})]^{-1} F(\vec{\beta}).$$

(4.) In **Step-4:** if the discrepancy between the estimate in the current iteration $\vec{\beta}^{k+1}$ and the estimate in the previous iteration $\vec{\beta}^k$ is less than a predetermined tolerance level ϵ. Stop the procedure and take the current estimate $\vec{\beta}$ as the root.

(5.) **Step-5:** Provide the ultimate approximation of the root.

Applying steps (1.)-(4.) above for the data in Table 3.1 the results of the WRNM are given below for 6 iterations.

iteration	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	Error	MSE
1	15.29005	16.759264	-7.68669	10.68669	6.34577
2	15.29858	16.76787	-7.69300	0.00853	6.33949
3	17.64353	19.165757	-9.43779	0.00853	5.72467
4	17.70802	19.174725	-9.46479	0.00631	5.69493
5	30.41378	32.94553	-19.20651	0.006301	22.92075
6	21.11058	14.627347	-9.04506	0.006301	0.00000

Table 3.3: Results for the WNRM. This table shows the estimated for $\vec{\beta} = (\beta_0, \beta_1, \beta_2)^T$ at each iteration $k = 1, 2, \ldots, 6$. The error statistics measures the distance $||\vec{\beta}^{k+1} - \vec{\beta}^k||_{max} = \max_{0 \leq j \leq 2} |\beta_j^{k+1} - \beta_j^k|$, where $||\vec{\beta}^{k+1} - \vec{\beta}^k||_{max} \to 0$, as $k \to \infty$ implies convergence. Since norms are equivalent on $\mathbb{R}^m$, the estimated mean square error given by the Euclidean norm $MSE = ||\vec{\beta}^k - \vec{\beta}_{WNRM}||_2^2$ subsequently converges to zero for large iterations k.

Remark 3.2. *Observe from Table 3.3 that the WNRM converges with nonsingular Jacobian matrix is obtained. The computer code for the solution is given in A.4.*

3.5 ASSUMPTIONS FOR THE LOGISTIC REGRESSION MODEL WITH REPEATED DATA

It is assumed that the p regressors $X_1, X_2, \ldots X_p$ consists of $m \geq 1$ levels, where the j^{th} level is denoted by $X_1^j, X_2^j, \ldots X_p^j$, $j = 1, 2, \ldots, m$. Furthermore, at each level j, there are n_j cases denoted by the random variable y_{ij}^j where $i = 1, 2, \ldots, n_j$. Also, for each $j = 1, 2, \ldots, m$, the y_{ij}^j are independent Bernoulli random variables i.e. $y_{1j}^j, y_{2j}^j, \ldots, y_{n_j j}^j$ are independent random variables. The cases y_{ij}^j and the regressors $X_1^j, X_2^j, \ldots X_p^j$, are related as follows.

$$\begin{aligned} y_{ij}^j &= \beta_0 + \beta_1 X_1^j + \beta_2 X_2^j + \cdots + \beta_p X_p^j + \epsilon_i^j, \\ &= (\vec{X}^j)^T \vec{\beta} + \epsilon_i^j, \quad \forall j = 1, 2, \ldots, m; \forall i = 1, 2, \ldots, n_j, \end{aligned} \tag{3.27}$$

where $(\vec{X}^j)^T = (1, X_1^j, X_2^j, \ldots X_p^j)$, and ϵ_i^j is the error random variable not satisfying the normality conditions as in the multiple linear regression model. In addition, the distribution of y_{ij}^j is given as follows.

$$P(y_{ij}^j = 1) = \pi_j, \quad P(y_{ij}^j = 0) = 1 - \pi_j \quad \forall j = 1, 2, \ldots, m; \forall i = 1, 2, \ldots, n_j. \tag{3.28}$$

Similarly as the case for non-repeated logistic regression, since the expected value $\mathbb{E}[y_{ij}^j] = \pi_j \in (0, 1)$, then we select the logistic growth function as follows

$$\pi_j = \frac{e^{(\vec{X}^j)^T \vec{\beta}}}{1 + e^{(\vec{X}^j)^T \vec{\beta}}}. \tag{3.29}$$

That is, from (3.29), it follows that the logit function is given by

$$\log\left(\frac{\pi_j}{1 - \pi_j}\right) = (\vec{X}^j)^T \vec{\beta}. \tag{3.30}$$

Observe from (3.27) that the sum of the Bernoulli random variables has binomial distribution given as follows.

$$y_j^j = \sum_{i=1}^{n_j} y_{ij}^j \sim Binomial(n_j, \pi_j). \tag{3.31}$$

That is,

$$P(y_j^j) = \binom{n_j}{y_j^j}(\pi_j)^{y_j^j}(1-\pi_j)^{n_j-y_j^j}, \forall y_j^j = 0, 1, \ldots, n_j; \forall j. \tag{3.32}$$

Note that for each $j = 1, 2, \ldots, m$, the random variables y_j^j are mutually independent binomial random variables i.e. $y_1^1, y_2^2, \ldots, y_m^m$ are independent.

3.6 The Method of Maximum Likelihood estimation in the logistic regression model for repeated data

Given the sample observations $\{y_1^1, y_2^2, \ldots, y_m^m\}$ of the binomial distribution, the log-likelihood function for the parameter vector $\vec{\beta}$ in Section 3.5 is derived.

$$\begin{aligned} L(\vec{\beta}|y_1^1, y_2^2, \ldots, y_m^m) &= \prod_{j=1}^{m} p(y_j^j); y_{ij}^j \sim Binomial(n_j, \pi_j) \\ &= \prod_{j=1}^{m} \binom{n_j}{y_j^j} \pi_j^{y_j^j}(1-\pi_j)^{n_j-y_j^j}. \end{aligned}$$

Taking logarithm function on both sides,

$$\log L(\vec{\beta}|y_1^1, y_2^2, \ldots, y_m^m) = \log \left[\prod_{j=1}^{m} \binom{n_j}{y_j^j} \pi_j^{y_j^j}(1-\pi_j)^{n_j-y_j^j} \right],$$

where y_j^j are the number of 1's and $n_j - y_j$ are zeros in n_j cases and $\binom{n_j}{y_j^j}$ can be taken as a constant C. The log-likelihood function from the binomial distribution can be written as,

$$\log L(\vec{\beta}|y_1^1, y_2^2, \ldots, y_m^m) = C + \sum_{j=1}^{m} y_j^j log(\pi_j) + \sum_{j=1}^{m} (n_j - y_j^j) log(1-\pi_j).$$

Using equation (3.30),

$$\log L(\vec{\beta}|y_1^1, y_2^2, \ldots, y_m^m) = C + \sum_{j=1}^{m} y_j^j (\vec{X}^j)^T \vec{\beta} - \sum_{j=1}^{m} n_j \log(1 + e^{(\vec{X}^j)^T \vec{\beta}}). \tag{3.33}$$

To optimize the log-likelihood function in (3.33), the *Weighted Newton Raphson Method(WNRM)* defined in Section 2.2 will be applied. In the following, the derivative of the log-likelihood function with respect to the parameter vector $\vec{\beta}$ is given.

3.7 EXAMPLE OF MAXIMUM LIKELIHOOD METHOD IN LOGISTIC REGRESSION WITH REPEATED DATA

The logistic model

$$y = \beta_0 + \beta_1 x^2 + \beta_2 e^x + \epsilon,$$

in (3.19) is reconsidered for the repeated data in Table 3.4, where it is assumed that there are $m = 5$ levels of the regressors (x^2, e^x), where the values of x are given in Table 3.5.

Levels	predictor value x	case 1	case 2	case 3	case 4	case 5	case 6	case 7	case 8	case 9	case 10
1	3.78379	1	1	1	0	1	NA	NA	NA	NA	NA
2	4.637155	1	1	0	0	0	0	0	1	0	1
3	4.586748	0	1	1	0	1	1	0	NA	NA	NA
4	4.915587	1	0	0	0	0	NA	NA	NA	NA	NA
5	4.32147	1	0	1	1	1	1	1	NA	NA	NA

Table 3.4: The table provides a summary of repeated data, presenting the predictor values (x^j) and corresponding binary outcomes across multiple cases.

Denote by $X^1, X^2, \ldots, X^5$, the five levels

$$X^1 = (3.78, e^{3.78}), X^2 = (4.64, e^4.64), X^3 = (4.59, e^4.8), X^4 = (4.92, e^4.92),$$
$$X^5 = (4.32, e^4.32) \tag{3.34}$$

At each level $j = 1, 2, \ldots, 5, X^j \in \{X^1, X^2, X^3, X^4, X^5\}$, Let y_{ij}^j be the i^{th} observation at level X^j. In Table 3.5, observe for $X^3 = (4.59, e^{4.59}), y_{i_3}^3 = 0, 1, 1, 0, 1, 1, 0,$ $\forall i = 1, 2, \ldots, 7$. Furthermore, at level $X^3, y_3^3 = \sum_{i=1}^{n_3} y_{i_3}^3 = 4$. Similarly, the summary of Table 3.4 for the repeated data is obtained in Table 3.5.

Level j	y_j^j	n_j
X^1	$y_1^1 = 4$	5
X^2	4	10
X^3	4	7
X^4	1	5
X^5	6	7

Table 3.5: Summary of a repeated data, detailing success counts y_j^j and corresponding total observations n_j across five levels X^1 through X^5. Each row represents a level, with the success count indicating the number of positive outcomes observed within that level and n_j denoting the total number of observations.

Given the sample for the Binomial random variables $(y_1^1, y_2^2, \ldots, y_m^m)$, where $m = 5$ and $y_j^j, \forall j$ is given in Table$\sim$3.4, the log-likelihood function from (3.21) is given by

$$\log L(\vec{\beta}|y_1^1, \ldots, y_m^m) = C + \sum_{j=1}^{m} y_j^j(\beta_0 + \beta_1 x^2 + \beta_2 e^x) - \sum_{j=1}^{m} n_j \log(1 + e^{\beta_0+\beta_1 x^2+\beta_2 e^x}). \tag{3.35}$$

From (3.35), let

$$f_j(\beta|\vec{\vec{y}}) = \frac{\partial log L(\vec{\beta}|\vec{y})}{\partial \beta_j}, j = 0, 1, 2. \tag{3.36}$$

It is easy to see that

$$f_1(\vec{\beta}|\vec{y}) = \sum_{j=1}^{m} \left[y_j^j - \frac{n_j}{(1 + e^{\beta_0+\beta_1 x^2+\beta_2 e^x})} \right], f_2(\vec{\beta}|\vec{y}) = \sum_{j=1}^{m} \left[y_j^j - \frac{n_j(x^2)}{(1 + e^{\beta_0+\beta_1 x^2+\beta_2 e^x})} \right],$$
$$f_3(\vec{\beta}|\vec{y}) = \sum_{j=1}^{m} \left[y_j^j - \frac{n_j(e^x)}{(1 + e^{\beta_0+\beta_1 x^2+\beta_2 e^x})} \right]. \tag{3.37}$$

Let

$$F(\vec{\beta}) = (f_1(\vec{\beta}|\vec{y}), f_2(\vec{\beta}|\vec{y}), f_3(\vec{\beta}|\vec{y})). \tag{3.38}$$

To solve the system

$$F(\vec{\beta}) = 0, \tag{3.39}$$

we apply the WNRM. The steps of the method and solution are given in the next section.

3.7.1 APPLICATION OF THE WNRM

Applying the steps of Algorithm 3.3.2 to solve (3.39) for $\vec{\beta}$, the following are obtained.

(1.) In **Step 1**, select $\vec{\beta}^{(0)} = (\beta_0, \beta_1, \beta_2) = (0.1, 0.2, 0.3)$.

(2.) In **Step 2**, compute the function value $F(\vec{\beta}) = (f_0(\vec{\beta}), f_1(\vec{\beta}), f_2(\vec{\beta}))^T$ in (3.38) and its derivative $\frac{\partial f_j(\vec{\beta})}{\partial \beta_j}$, $\forall j = 0, 1, 2$ in (3.37) to find the Jacobian matrix $[J_F(\vec{\beta})]$ at the k^{th} iteration $\vec{\beta}^k$.

(3.) In **Step 3**, define $\hat{G}(\beta) = \vec{\beta} - [J_F(\vec{\beta})]^{-1} M F(\vec{\beta})$ applied to system $F(\vec{\beta}) = 0$, where $M = diag(m_j)$, and

$$m_j = \frac{1}{1 - \frac{\partial g_j(\beta)}{\partial \beta_j}}, j = 1, 2, ..., m,$$

$$\begin{aligned} G(\vec{\beta}) &= (g_1(\vec{\beta}), ..., g_n(\vec{\beta}))^T \\ &= \vec{\beta} - [J_F(\vec{\beta})]^{-1} F(\vec{\beta}) \\ &= \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix} - \begin{pmatrix} \frac{\partial f_1}{\partial \beta_0} & \frac{\partial f_1}{\partial \beta_1} & \frac{\partial f_1}{\partial \beta_2} \\ \frac{\partial f_2}{\partial \beta_0} & \frac{\partial f_2}{\partial \beta_1} & \frac{\partial f_2}{\partial \beta_2} \\ \frac{\partial f_3}{\partial \beta_0} & \frac{\partial f_3}{\partial \beta_1} & \frac{\partial f_3}{\partial \beta_2} \end{pmatrix}^{-1} \begin{pmatrix} f_1(\beta_0, \beta_1, \beta_2 | \vec{y}) \\ f_2(\beta_0, \beta_1, \beta_2 | \vec{y}) \\ f_3(\beta_0, \beta_1, \beta_2 | \vec{y}) \end{pmatrix} \end{aligned} \tag{3.40}$$

where the terms of the Jacobian matrix are given by

$$\frac{\partial f_1}{\partial \beta_0} = -\sum_{j=1}^{m} n_j \left[\frac{e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right], \frac{\partial f_2}{\partial \beta_0} = -\sum_{j=1}^{m} n_j \left[\frac{(x^2)e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right],$$

$$\frac{\partial f_1}{\partial \beta_1} = -\sum_{j=1}^{m} n_j \left[\frac{(x^2)e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right], \frac{\partial f_2}{\partial \beta_1} = -\sum_{j=1}^{m} n_j \left[\frac{(x^4)e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right],$$

$$\frac{\partial f_1}{\partial \beta_2} = -\sum_{j=1}^{m} n_j \left[\frac{(e^x)e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right], \frac{\partial f_2}{\partial \beta_2} = -\sum_{j=1}^{m} n_j \left[\frac{(x^2)(e^x)e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right],$$

$$\frac{\partial f_3}{\partial \beta_0} = -\sum_{j=1}^{m} n_j \left[\frac{(e^x)e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right],$$

$$\frac{\partial f_3}{\partial \beta_1} = -\sum_{j=1}^{m} n_j \left[\frac{(x^2)(e^x)e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right],$$

$$\frac{\partial f_3}{\partial \beta_2} = -\sum_{j=1}^{m} n_j \left[\frac{(e^x)^2 e^{-\beta_0-\beta_1 x^2-\beta_2 e^x}}{(1+e^{-\beta_0-\beta_1 x^2-\beta_2 e^x})^2} \right].$$

(4.) In **Step-4:** if the discrepancy between the estimate in the current iteration $\vec{\beta}^{k+1}$ and the estimate in the previous iteration $\vec{\beta}^{k}$ is less than a predetermined tolerance level ϵ. Stop the procedure and take the current estimate $\vec{\beta}$ as the root.

(5.) **Step-5:** Provide the ultimate approximation of the root.

Applying steps (1.)-(4.) above for the repeated data in Table 3.4 the results for the WRNM to find the MLE for $\vec{\beta}$ are given below for 5 iterations.

iteration	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	MSE
1	-34.7136747	4.4465461	-0.2907909	13.698003
2	-35.3266133	4.5206603	-0.3009804	13.080517
3	-41.7527672	5.2971503	-0.4076345	6.606753
4	-44.6413870	5.6413478	-0.4540003	3.697331
5	-48.3129852	6.0730555	-0.5110502	0.000000

Table 3.6: Results for the repeated data of WNRM . This table shows the estimated for $\vec{\beta} = (\beta_0, \beta_1, \beta_2)^T$ at each iteration $k = 1, 2, \ldots, 5$.

Remark 3.3. *Observe from Table 3.6 that the WNRM converges with nonsingular Jacobian matrix obtained. The computer code for the solution is given in A.5.*

CHAPTER 4

BINARY CLASSIFICATION OF DIABETES OCCURRENCE IN THE PIMA INDIANS DIABETES DATABASE

In this chapter, the WNRM is applied to classify the occurrence of of diabetes in the Pima Indians Diabetes data[2]. The data was collected by the "National Institute of Diabetes and Digestive and Kidney Diseases" describing different factors related to diabetes in the *Pima Indian native American* population. The subjects of the study are female patients over the age of 21.

4.1 Description of the Pima Indians Diabetes data

In the data[2] there are eight predictors and one binary response variable denoted by "Outcome" summarized in the Table 4.1 below. Furthermore, the all predictors are quantitative and their ranges are given in the column denoted by "Range" in Table 4.1.

Attributes	**Description**	**Range**
Pregnancies	No. of pregnancies	0–17
Glucose	2 hours of oral glucose tolerance test	0–199
Blood Pressure	Blood pressure in mm Hg	0–122
Skin thickness	Skinfold thickness of triceps (mm)	0–99
Insulin	Two hours of serum insulin (mu U/ml)	0–846
BMI	Body mass index (weight in kg/(height in m)2)	0–67
Diabetes Pedigree Function	Attribute used in diabetes prognosis	0.078–2.4
Age	Age (years)	21–81
Outcome	Class variable (0 or 1)	Y/N

Table 4.1: Predictor variables for the outcome of diabetes

The graphical summaries for the predictor variables are given in Figures 4.1 & 4.2.

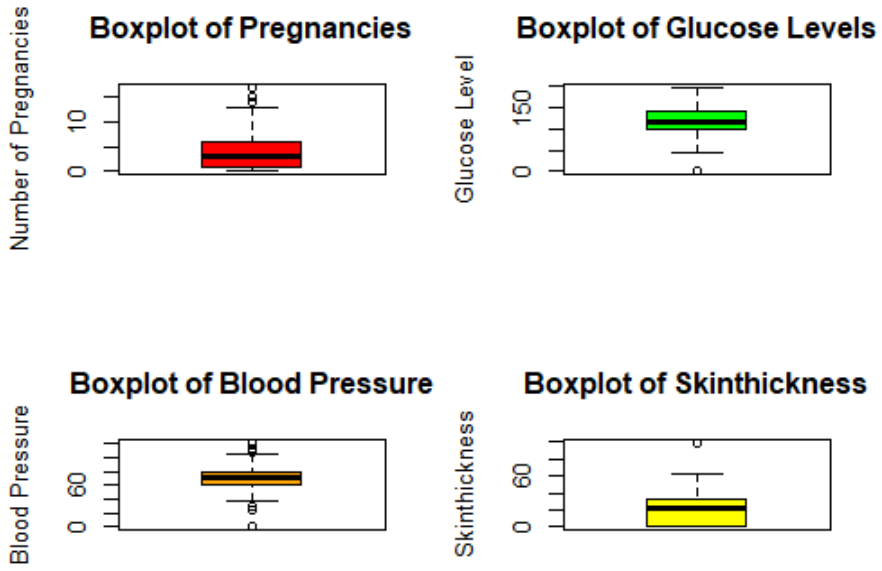

Figure 4.1: The boxplots are summaries for the predictor variables representing “number of pregnancies”, “Blood pressure”, “Glucose level” and “Skin thickness”.

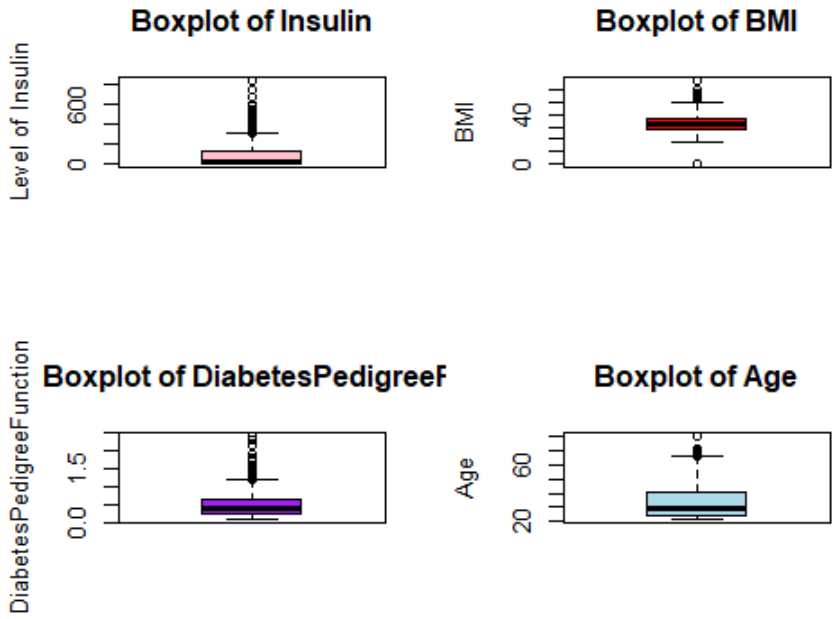

Figure 4.2: The boxplots are summaries for the predictor variables representing "Level of Insulin", "BMI", "Diabetes Pedigree Function" and "Age".

Remark 4.1. *The boxplot visually represents the midpoint, quartiles, and outliers, offering valuable information regarding the extent and dispersion for the predictor variables "pregnancies", "Blood pressure", "Glucose", "Skin thickness", "Insulin", "BMI", "Diabetes Pedigree Function" and "Age".*

4.2 The logistic regression model for the diabetes data and derivation of the Weighted Newton-Raphson algorithm

The logistic model for the diabetes data based on (3.27) is given by

$$\begin{aligned} y_i &= \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_8 X_{i8} + \epsilon_i, \\ &= (\vec{X}_i)^T \vec{\beta} + \epsilon_i, \quad \forall i = 1, 2, \ldots, n, \end{aligned} \tag{4.1}$$

where

$$\vec{\beta} = (\beta_0, \beta_1, \ldots, \beta_8)^T.$$

The WNRM for this example is given as follows.

$$\vec{\beta}^{k+1} = \vec{\beta}^k - [J_F(\vec{x}^k)]^{-1} diag(m_i) F(\vec{\beta}^k), \tag{4.2}$$

where the steps of the algorithm are given in Algorithm 3.3.2 , and the method is used to optimize the log-likelihood function for the model defined subsequently.

Note that in the data[2] there 768 patients, and hence the binary responses are denoted by $\vec{y} = (y_1, y_2, \ldots, y_{768})$ representing whether the patient has diabetes ($y_i = 1$) or not ($y_i = 0$). Also, the eight factors that predict diabetes in the population are given in Table 4.1. Thus, the log-likelihood function based on logistic regression in (3.30) is given by

$$\log L(y_1, y_2, ..., y_{768}, \vec{\beta}) = \sum_{i=1}^{768} y_i [\vec{x}_i^T \vec{\beta}] + \sum_{i=1}^{768} \log \left(\frac{1}{1 + e^{\vec{x}_i^T \vec{\beta}}} \right), \tag{4.3}$$

where

$$\vec{x}_i^T = (1, x_{i1}, x_{i2}, \ldots, x_{i8}), i = 1, 2, \ldots, 768$$
$$\vec{\beta} = (\beta_0, \beta_1, \ldots, \beta_8)^T.$$

The function (4.3) is optimized by solving the system of nine equations

$$f_{j+1}(\vec{\beta}) = \frac{\partial \log L(\vec{y}, \vec{\beta})}{\partial \vec{\beta}_j} = 0, j = 0, 1, 2, \ldots, 8, \tag{4.4}$$

also written as

$$F(f_1(\vec{\beta}), f_2(\vec{\beta}), \ldots, f_9(\vec{\beta})) = 0, \tag{4.5}$$

by applying the WNRM.

4.2.1 APPLICATION OF THE WNRM

Note that the form of the Jacobian matrix is given by

$$J(\vec{\beta}) = \begin{pmatrix} \frac{\partial f_1}{\partial \beta_0} & \frac{\partial f_1}{\partial \beta_1} & ..... & \frac{\partial f_1}{\partial \beta_8} \\ \frac{\partial f_2}{\partial \beta_0} & \frac{\partial f_2}{\partial \beta_1} & ..... & \frac{\partial f_2}{\partial \beta_8} \\ \frac{\partial f_3}{\partial \beta_0} & \frac{\partial f_3}{\partial \beta_1} & ..... & \frac{\partial f_3}{\partial \beta_8} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \frac{\partial f_9}{\partial \beta_0} & \frac{\partial f_9}{\partial \beta_1} & ..... & \frac{\partial f_9}{\partial \beta_8} \end{pmatrix}.$$

From (4.3) and (4.4) it is easy to see that

$$f_j(\vec{\beta}) = \sum_{i=1}^{n} \left[y_i - \frac{e^{\vec{x_i}^T \vec{\beta}}}{(1 + e^{\vec{x_i}^T \vec{\beta}})} \right], j = 1, 2, \ldots, 9. \tag{4.6}$$

Furthermore, note that from (4.6) the following hold.

$$\frac{\partial f_1}{\partial \beta_0} = -\sum_{i=1}^{n} \left[\frac{e^{\vec{x_i}^T \vec{\beta}}}{(1 + e^{\vec{x_i}^T \vec{\beta}})^2} \right]; \tag{4.7}$$

$$\frac{\partial f_1}{\partial \beta_j} = -\sum_{i=1}^{n} \left[\frac{(x_{ij}) e^{\vec{x_i}^T \vec{\beta}}}{(1 + e^{\vec{x_i}^T \vec{\beta}})^2} \right], j = 1, 2, 3, \ldots, 8; \tag{4.8}$$

$$\frac{\partial f_j}{\partial \beta_0} = -\sum_{i=1}^{n} \left[\frac{(x_{ij-1}) e^{\vec{x_i}^T \vec{\beta}}}{(1 + e^{\vec{x_i}^T \vec{\beta}})^2} \right], j = 2, 3, \ldots, 9; \tag{4.9}$$

$$\frac{\partial f_j}{\partial \beta_1} = -\sum_{i=1}^{n} \left[\frac{(x_{i1} x_{ij-1}) e^{\vec{x_i}^T \vec{\beta}}}{(1 + e^{\vec{x_i}^T \vec{\beta}})^2} \right], j = 2, 3, \ldots, 9; \tag{4.10}$$

$$\frac{\partial f_j}{\partial \beta_l} = -\sum_{i=1}^{n} \left[\frac{(x_{il} x_{ij-1}) e^{\vec{x_i}^T \vec{\beta}}}{(1 + e^{\vec{x_i}^T \vec{\beta}})^2} \right], j = 2, 3, \ldots, 9; l = 1, 2 \ldots, 8. \tag{4.11}$$

Also,

$$\frac{\partial f_j}{\partial \beta_2} = -\sum_{i=1}^{n}\left[\frac{(x_{i2}x_{ij-1})e^{\vec{x_i}^T\vec{\beta}}}{(1+e^{\vec{x_i}^T\vec{\beta}})^2}\right], j = 2,3,\ldots,9; \tag{4.12}$$

$$\frac{\partial f_j}{\partial \beta_3} = -\sum_{i=1}^{n}\left[\frac{(x_{i3}x_{ij-1})e^{\vec{x_i}^T\vec{\beta}}}{(1+e^{\vec{x_i}^T\vec{\beta}})^2}\right], j = 2,3,\ldots,9; \tag{4.13}$$

$$\frac{\partial f_j}{\partial \beta_8} = -\sum_{i=1}^{n}\left[\frac{(x_{i8}x_{ij-1})e^{\vec{x_i}^T\vec{\beta}}}{(1+e^{\vec{x_i}^T\vec{\beta}})^2}\right], j = 2,3,\ldots,9. \tag{4.14}$$

Now, setting the fixed point function

$$G(\vec{\beta}) = \vec{\beta} - [J_F(\beta)]^{-1}F(\vec{\beta}) = (g_1(\vec{\beta}), g_2(\vec{\beta}), ..., g_9(\vec{\beta}))^T,$$

the weights of the diagonal matrix $M = diag(m_i)$ are obtained using

$$m_i = \frac{1}{1-\left(\frac{\partial g_i(\vec{\beta})}{\partial \beta_i}\right)},$$

via approximating the derivative using the gradient of g_i over the values of β_i, $\forall i$.
For the selected initial solution $\vec{\beta}^{(0)} = (0.01, 0.02, 0.03, 0.07, 0.04, 0.1, 0.05, 0.06, 0.02)^T$, the MLE is obtained via the WNRM, and the solution of $\vec{\beta}$ for the system (4.5) at each iteration is given in the Table 4.2. Moreover, the convergence of each coefficient β_j of $\vec{\beta}$ to the MLE is exhibited in Figures 4.3& 4.4 .

It.	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$	MSE
1	15.7018673	0.7207873	-0.1519086	0.2483672	-0.5704282	-0.1761688	0.2625451	3.3461076	-0.1398580	8.374106e+00
2	14.7761955	0.6634895	-0.1429595	0.2321221	-0.5314312	-0.1654340	0.2426316	3.1624618	-0.1273688	4.325943e+01
3	10.21810584	0.46278450	-0.09833367	0.15876069	-0.36497026	-0.11429201	0.16838205	2.11127009	-0.08637898	3.872988e+01
4	7.93874120	0.35911382	-0.07574936	0.12146202	-0.28008550	-0.08851523	0.12984334	1.55004888	-0.06451209	2.819186e+01
5	5.89668254	0.26252871	-0.05504916	0.08765787	-0.20190764	-0.06544913	0.09249635	1.06213816	-0.04465416	2.352459e+01
6	4.45153755	0.19251594	-0.04024674	0.06385739	-0.14631978	-0.04952900	0.06434748	0.74378561	-0.02985423	1.938498e+01
7	3.21853763	0.14029728	-0.02835819	0.04605900	-0.10528933	-0.03747026	0.04376092	0.53005845	-0.01782847	1.622250e+01
8	1.851895449	0.104670303	-0.018160393	0.033621043	-0.078124667	-0.027993939	0.035631828	0.417673308	-0.007978441	1.355642e+01
9	-3.714800e-01	8.486107e-02	-8.393908e-03	2.633933e-02	-6.554369e-02	-1.992756e-02	5.626866e-02	4.217423e-01	-8.969541e-05	1.132102e+01
10	-8.957442324	0.122643242	-0.010730962	0.027245318	-0.144730320	-0.015004728	0.375007430	0.327917770	0.004950934	9.113631e+00
11	12.4546557557	0.0040231075	-0.0005874807	0.0251019525	0.0408023944	0.0962392643	-0.6614690830	-0.7299445553	-0.0199102382	6.360850e+00
12	-5.79816632	0.19866922	-0.02121204	0.04442662	-0.08386838	-0.01272730	0.28144967	0.59429721	-0.07314947	3.703046e+01
13	-6.51668948	0.29960483	-0.02219140	0.07129287	-0.10681619	-0.01343488	0.31999545	0.87406400	-0.14239235	5.070423e+00
14	-3.967685118	0.180422495	-0.003334904	0.020379636	-0.049693464	-0.008946607	0.190576694	0.469802695	-0.081277393	4.986331e+00
15	-4.171962887	0.147377661	0.002247174	0.008237185	-0.034776848	-0.006929949	0.162026545	0.451151213	-0.052789558	1.588115e+01
16	5.1825778422	0.1325764386	-0.0096573608	-0.0429537327	-0.0096104542	-0.0004621504	-0.0096204239	0.0348891295	-0.0370879545	1.890119e+00
17	-5.1719825960	0.0661062498	0.0132293293	0.0198563687	0.0013620542	-0.0002064434	0.0332007956	0.5709185517	0.0034239544	2.138039e+00
18	-6.1692912763	0.1002159420	0.0215114439	0.0115973370	0.0053728009	-0.0003915115	0.0354331999	0.5966315746	0.0071004361	6.714345e+00
19	-23.734674048	0.097739460	0.057368699	0.035796185	-0.023586606	-0.002976117	0.322094734	2.074356640	0.056839318	9.018973e-01
20	-7.0991871563	0.1048108264	0.0272130305	0.0027434346	-0.0047700355	-0.0007498842	0.0662698855	0.6918541049	0.0065347607	2.302565e+00
21	-7.5040173002	0.1272051660	0.0300285769	-0.0007733009	-0.0033286322	-0.0008879174	0.0717268625	0.7826353585	0.0046136068	2.992251e-01
22	-9.194551528	0.115646035	0.037755194	-0.012809292	0.002455179	-0.001366140	0.092258025	1.023313323	0.023280618	1.583946e-01
23	-8.311481074	0.123196972	0.034726763	-0.013383206	0.000288166	-0.001163558	0.089841078	0.932409635	0.014221956	6.209808e-02
24	-8.4039595018	0.1232123879	0.0351360568	-0.0133431492	0.0005745193	-0.0011899499	0.0899413722	0.9443912302	0.0148502554	5.500343e-02

Table 4.2: Results for the diabetes data of WNRM . This table shows the estimated for $\vec{\beta} = (\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8)^T$ at each iteration $k = 1, 2, \ldots, 24$. The MLE for $\vec{\beta}$ is given by the results at $k = 24$.

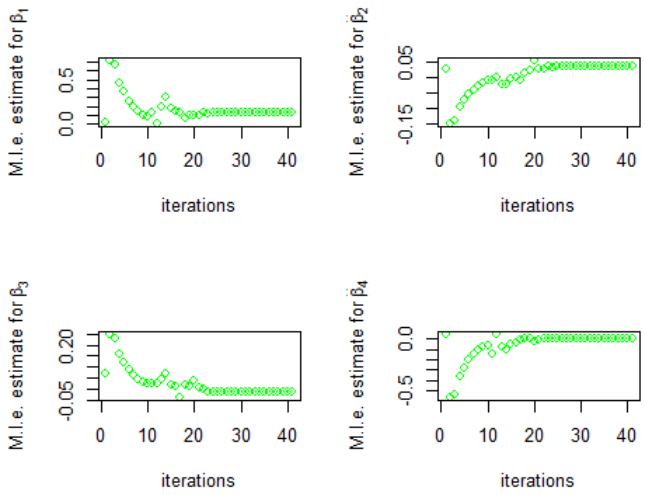

Figure 4.3: Number of interactions until convergence for the Maximum Likelihood Estimation for $\vec{\beta}$

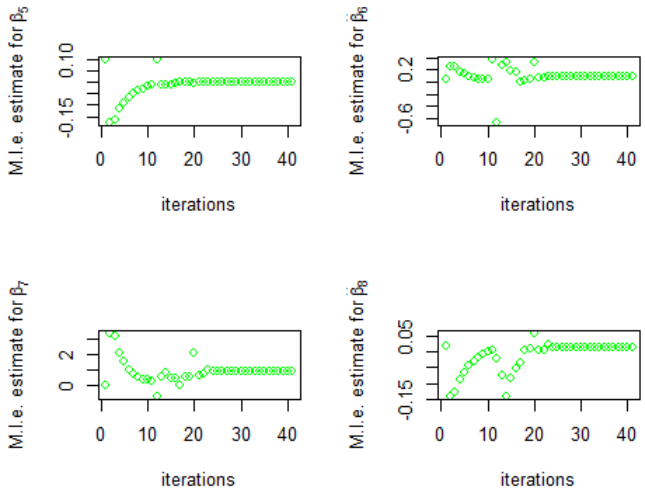

Figure 4.4: Number of interactions until convergence for the Maximum Likelihood Estimation for $\vec{\beta}$

Remark 4.2. *The figure (4.3)-(4.4) is explained by emphasizing the relationship between the quantity of iterations necessary for the Maximum Likelihood Estimation (MLE) algo-*

rithm to reach a solution for the parameter vector $\vec{\beta}$. Each graph represents the number of interactions required until convergence for different datasets or scenarios. Notably, all four graphs demonstrate consistent convergence, reaching stability after approximately 24 *iterations.*

4.2.2 Application of the Reweighted Least Square method

The classification of the response variable in the logistic model (4.1) is also conducted by applying the *Reweighted Least Square method* in the $glm(\cdots)$ function of the R-software [1] for fitting *Generalized Linear Models*. The statistical method is explained in the text[3]. The R-code is given in A.7 and the output is shown in Computer Output 1.

Computer-Output 1. *The output in Table 4.3 is obtained from R, applying the Reweighted Least Square method to find the MLE for the parameter $\vec{\beta}$ in 4.3.*

Coefficients:	Estimate	Std.Error	z value	$Pr(>\|z\|)$
(Intercept)	-8.4046964	0.7166359	-11.728	$< 2 \times 10^{-16}$ ***
Pregnancies	0.1231823	0.0320776	3.840	0.000123 ***
Glucose	0.0351637	0.0037087	9.481	$< 2 \times 10^{-16}$ ***
Blood Pressure	-0.0132955	0.0052336	-2.540	0.011072 *
Skin Thickness	0.0006190	0.0068994	0.090	0.928515
Insulin	-0.0011917	0.0009012	-1.322	0.186065
BMI	0.0897010	0.0150876	5.945	2.76×10^{-09} ***
Diabetes Pedigree F.	0.9451797	0.2991475	3.160	0.001580 **
Age	0.0148690	0.0093348	1.593	0.111192

Table 4.3: The output of a Generalized Linear Model (GLM) analysis for (4.1) using the glm function in R is presented in the table. Every row in the dataset represents a predictor variable, including information on the estimated coefficients, standard errors, z-values, and corresponding p-values.

Remark 4.3.

(1.) Comparing the WNRM in Table 4.2 with the Re-weighted least squares in Table 4.3 methods in the glm() package in R. The results for the MLE are similar. But the Re-weighted least squares methods in the glm() package in R converges after 5 iterations while the Weighted NRM algorithm converges after 24 iterations.

(2.) While the Weighted NRM is more efficient that the ordinary NRM, it appears to less efficient that the Re-weighted least squares algorithm for this particular problem.

We have applied Weighted NRM to fitted the model for non-repeated data and we have extended the method for repeated data in logistic regression model.

4.3 INTERPRETATION OF THE MLE'S FOR THE REGRESSION COEFFICIENTS

In logistic regression, the dependent variable is a dichotomous variable. Dichotomous variables are variables with only two values that is 0 and 1. In the model (4.1), 0 stands for not having diabetes and 1 for having diabetes. The predictor variables- "Pregnancies", "Glucose","Blood Pressure", "Skin-thickness", "Insulin", "BMI", "Diabetes Pedigree Function", "Age" and the dependent variable is "outcome" with values of 0 and 1. In our model, the probability of the i^{th} subject having diabetes is a function of the predictor variables above and given by (3.4) and estimated by

$$\hat{\pi_i} = E[\hat{y}_i] = \left(1 + e^{-(\vec{x}_i)^T\vec{\beta}}\right)^{-1}, i = 1, 2, \ldots, n, \tag{4.15}$$

where $\hat{\vec{\beta}}$ is the MLE of $\vec{\beta}$ obtained in Subsection 4.2.1. Furthermore,

$$\log\left(\frac{\hat{\pi_i}}{1-\hat{\pi_i}}\right) = (\vec{x}_i)^T\hat{\vec{\beta}}. \tag{4.16}$$

To minimize complex notations, the population version of the model (4.1) given below will be used for interpretation.

$$\begin{aligned}
&y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_8 X_8 + \epsilon,\\
&= \vec{x}^T\vec{\beta} + \epsilon, \quad \vec{x}^T = (1, X_1, \ldots, X_8)^T,\\
&\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_8 X_8,\\
&P(y=1) = \pi, \quad P(y=0) = 1-\pi.
\end{aligned} \tag{4.17}$$

The following notation is used to interpret the MLE $\hat{\beta}_j, \forall j = 1, 2, \ldots, 8$ of the j^{th} coefficient β_j of the model (4.17). From (4.17), denote by

$$\begin{aligned}
&\pi_{log+odd}(X_j) = \log\left(ODD(X_j)\right)\\
&= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_j X_j + \ldots + \beta_8 X_8; \forall j = 1, 2, \ldots, 8.
\end{aligned} \tag{4.18}$$

and

$$\pi_{log+odd}(X_j+1) = \log\left(ODD(X_j+1)\right)$$
$$= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_j(X_j+1) + \ldots + \beta_8 X_8; \forall j = 1, 2, \ldots, 8. \quad (4.19)$$

It is easy to see from (4.18) & (4.19) that

$$\pi_{log+odd}(X_j+1) - \pi_{log+odd}(X_j) = \log\left(\frac{ODD(X_j+1)}{ODD(X_j)}\right) = \beta_j, \forall j = 1, 2, \ldots, 8, \quad (4.20)$$

and

$$\frac{ODD(X_j+1)}{ODD(X_j)} = e^{\beta_j}, \forall j = 1, 2, \ldots, 8. \quad (4.21)$$

It is also easy to see from (4.21) that for any positive real number $c > 0$,

$$\frac{ODD(X_j+c)}{ODD(X_j)} = e^{c\beta_j}, \forall j = 1, 2, \ldots, 8. \quad (4.22)$$

Observe that (4.21) describes the odd ratio for increasing the regressor X_j by one unit; and (4.22) describes the odd ratio for increasing the regressor X_j by $c > 0$ units.

(1.) Intercept: From Subsection 4.2.1, the MLE of the intercept is $\hat{\beta}_0 = -8.4046963669 \approx -8.405$. From (4.16), observe that when $\vec{x}_i = (1, 0, 0, \ldots, 0)^T$, the odd of diabetes occurrence is given by

$$\frac{\hat{\pi}_i}{1-\hat{\pi}_i} = e^{\hat{\beta}_0} = e^{-8.4046963669} = 0.00022381374 \quad (4.23)$$

That is, the odd of getting diabetes in the absence of "pregnancies","glucose","blood pressure","skinthickness","insulin","BMI","diabetes predigree function","age" is 0.00022381374. This relatively small odd value suggests it is unlikely to have diabetes in the absence of the predictors.

(2.) Pregnancies: From Subsection 4.2.1, the MLE of $\hat{\beta}_1 = 0.1231822984 \approx 0.123$. From (4.21)-(4.22), the odd ratios

$$\begin{aligned} \frac{ODD(X_1+1)}{ODD(X_1)} &= e^{\hat{\beta_1}} = e^{0.1231822984} = 1.13109059816 \\ \frac{ODD(X_1+c)}{ODD(X_1)} &= e^{c\hat{\beta}_1} = e^{c\times 0.1231822984}. \end{aligned} \tag{4.24}$$

The equation (4.24) suggests that the odd ratio for getting diabetes increases by approximately 1.131, for every additional pregnancy. By increasing the number of pregnancies by $c > 0$ units, the odd ratio further changes by $e^{c\times 0.1231822984}$ units. Also, since the p-value is $0.000123 < 0.05$, this implies that the coefficient β_1 for Pregnancies is statistically significant at the 0.05 significance level. Hence, pregnancy significantly contributes to the occurrence of diabetes.

(3.) Glucose level: From Subsection 4.2.1, the MLE of $\hat{\beta}_2 = 0.0351637146 \approx 0.035$. From (4.21)-(4.22), the odd ratios

$$\begin{aligned} \frac{ODD(X_1+1)}{ODD(X_1)} &= e^{\hat{\beta_2}} = e^{0.0351637146} = 1.03578926875 \\ \frac{ODD(X_1+c)}{ODD(X_1)} &= e^{c\hat{\beta}_2} = e^{c\times 1.03578926875}. \end{aligned} \tag{4.25}$$

The equation (4.24) suggests that the odd ratio for getting diabetes increases by approximately 1.035, for every additional glucose level. By increasing the number of glucose level by $c > 0$ units, the odd ratio further changes by $e^{c\times 0.0351637146}$ units. Also, since the p-value is $2 \times 10^{-16} < 0.05$, this implies that the coefficient β_2 for Glucose is statistically significant at the 0.05 significance level. Hence, glucose significantly contributes to the occurrence of diabetes.

(4.) Blood Pressure: From Subsection 4.2.1, the MLE of $\hat{\beta}_3 = -0.0132955469 \approx -0.013$.

From (4.21)-(4.22), the odd ratios

$$\frac{ODD(X_1+1)}{ODD(X_1)} = e^{\hat{\beta}_3} = e^{-0.0132955469} = 0.98679244847$$
$$\frac{ODD(X_1+c)}{ODD(X_1)} = e^{c\hat{\beta}_3} = e^{c\times 0.98679244847}. \tag{4.26}$$

The equation (4.24) suggests that the odd ratio for getting diabetes increases by approximately 1.035, for every additional blood pressure. By increasing the number of blood pressure by $c > 0$ units, the odd ratio further changes by $e^{c\times -0.0132955469}$ units. Also, since the p-value is $0.011072 < 0.05$, this implies that the coefficient β_2 for Blood Pressure is statistically significant at the 0.05 significance level. Hence, blood pressure significantly contributes to the occurrence of diabetes.

(5.) Skin Thickness: From Subsection 4.2.1, the MLE of $\hat{\beta}_4 = 0.0006189644 \approx -0.0006$. From (4.21)-(4.22), the odd ratios

$$\frac{ODD(X_1+1)}{ODD(X_1)} = e^{\hat{\beta}_4} = e^{0.0006189644} = 1.000619156$$
$$\frac{ODD(X_1+c)}{ODD(X_1)} = e^{c\hat{\beta}_4} = e^{c\times 0.0006189644}. \tag{4.27}$$

The equation (4.24) suggests that the odd ratio for getting diabetes increases by approximately 1.0006, for every additional skin thickness. By increasing the number of blood pressure by $c > 0$ units, the odd ratio further changes by $e^{c\times 0.0006189644}$ units. Also, since the p-value is $0.928515 < 0.05$, this implies that the coefficient β_4 for Skin Thickness is statistically significant at the 0.05 significance level. Hence, skin thickness significantly contributes to the occurrence of diabetes.

(6.) Insulin: From Subsection 4.2.1, the MLE of $\hat{\beta}_5 = 0.0006189644 \approx -0.0006$. From (4.21)-(4.22), the odd ratios

$$\frac{ODD(X_1+1)}{ODD(X_1)} = e^{\hat{\beta}_5} = e^{0.0006189644} = 1.000619156$$
$$\frac{ODD(X_1+c)}{ODD(X_1)} = e^{c\hat{\beta}_5} = e^{c\times 0.0006189644}. \tag{4.28}$$

The equation (4.24) suggests that the odd ratio for getting diabetes increases by approximately 1.0006, for every additional insulin. By increasing the number of insulin by $c > 0$ units, the odd ratio further changes by $e^{c\times 0.0006189644}$ units. Also, since the p-value is $0.186065 > 0.05$, this implies that the coefficient β_5 for Insulin is statistically insignificant at the 0.05 significance level. Hence, insulin does not significantly contributes to the occurrence of diabetes.

(7.) BMI: From Subsection 4.2.1, the MLE of $\hat{\beta}_6 = 0.0006189644 \approx -0.0006$. From (4.21)-(4.22), the odd ratios

$$\begin{aligned}\frac{ODD(X_1+1)}{ODD(X_1)} &= e^{\hat{\beta_6}} = e^{0.0897009700} = 1.09384714168\\ \frac{ODD(X_1+c)}{ODD(X_1)} &= e^{c\hat{\beta}_6} = e^{c\times 0.0897009700}.\end{aligned} \tag{4.29}$$

The equation (4.24) suggests that the odd ratio for getting diabetes increases by approximately 1.0006, for every additional BMI. By increasing the number of blood pressure by $c > 0$ units, the odd ratio further changes by $e^{c\times 0.0897009700}$ units. Also, since the p-value is $2.76e-09 < 0.05$, this implies that the coefficient β_6 for BMI is statistically significant at the 0.05 significance level. Hence, BMI significantly contributes to the occurrence of diabetes.

(8.) Diabetes Pedigree Function: From Subsection 4.2.1, the MLE of $\hat{\beta}_7 = 0.9451797406 \approx 0.9452$. From (4.21)-(4.22), the odd ratios

$$\begin{aligned}\frac{ODD(X_1+1)}{ODD(X_1)} &= e^{\hat{\beta_7}} = e^{0.9451797406} = 2.57327585917\\ \frac{ODD(X_1+c)}{ODD(X_1)} &= e^{c\hat{\beta}_7} = e^{c\times 0.9451797406}.\end{aligned} \tag{4.30}$$

The equation (4.24) suggests that the odd ratio for getting diabetes increases by approximately 2.5733, for every additional Diabetes Pedigree Function. By increasing the number of blood pressure by $c > 0$ units, the odd ratio further changes by

$e^{c\times 0.9451797406}$ units. Also, since the p-value is $0.001580 < 0.05$, this implies that the coefficient β_7 for Diabetes Pedigree Function is statistically significant at the 0.05 significance level. Hence, Diabetes Pedigree Function significantly contributes to the occurrence of diabetes.

(9.) Age: From Subsection 4.2.1, the MLE of $\hat{\beta}_8 = 0.9451797406 \approx 0.9452$. From (4.21)-(4.22), the odd ratios

$$\frac{ODD(X_1+1)}{ODD(X_1)} = e^{\hat{\beta}_8} = e^{0.0148690047} = 1.01498009828$$
$$\frac{ODD(X_1+c)}{ODD(X_1)} = e^{c\hat{\beta}_8} = e^{c\times 1.01498009828}. \tag{4.31}$$

The equation (4.24) suggests that the odd ratio for getting diabetes increases by approximately 2.5733, for every additional Age. By increasing the number of blood pressure by $c > 0$ units, the odd ratio further changes by $e^{c\times 0.0148690047}$ units. Also, since the p-value is $0.111192 > 0.05$, this implies that the coefficient β_8 for Age is statistically insignificant at the 0.05 significance level. Hence, Age is not significantly contributes to the occurrence of diabetes.

4.4 SELECTING THE BEST PREDICTIVE LOGISTIC REGRESSION MODEL

Model performance and complexity must be balanced in order to select the best predictive model. A more detailed model with many predictor variables might be able to identify subtle patterns in the data, but it runs at risk of over-fitting, which happens when the model catches up noise in the training set instead of actual underlying relationships. Conversely, while a simpler model with fewer predictor variables could be easier to comprehend and more flexible to new data, it might also lose some prediction accuracy. Numerous methods for selecting models have been developed, each with benefits and drawbacks of their own. Among these methods, the most popular ones for choosing a subset of predictor variables that best enhance the model's predictive ability are forward selection and backward

elimination. In forward and backward selection method, some statistical goodness of fit statistics applied include: *Akaike information criterion (AIC); deviance*; and *p-value* [3]. We define these statistics in the following.

Definition 4.4.1. ***Residual Sum of Squares (RSS):***

In linear regression, RSS (Residual Sum of Squares) quantifies the difference between the actual values of the dependent variable and the values estimated by the model. It is calculated as the sum of the squared differences between the observed values y and the predicted values y. That is,

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

RSS quantifies the overall fit of the model to the data. Smaller variations between actual and predicted values are indicated by lower RSS values, which suggest a better fit.

Definition 4.4.2. ***Deviance:***

Deviance in logistic regression is a measure that assesses the degree of compatibility between the observed data and the model's predictions, similar to the residual sum of squares in linear regression. Deviance quantifies the difference between the fitted model and the saturated model derived from the logistic regression model. It is calculated as twice the difference in log-likelihood between the null model (with only the intercept) and the fitted model:

Definition 4.4.3. ***Null Deviance:***

The deviation occurs when the model involves only the intercept term. The null deviance is the difference between $-2logL$ for the saturated model and $-2logL$ for the intercept-only model.

Definition 4.4.4. ***Residual Deviance:***

Residual deviance is the difference between $-2logL$ for the saturated model and $-2logL$ for the currently fit model.

Definition 4.4.5. ***(Akaike information criterion) AIC:***

AIC is a measure that evaluates the quality of a model by considering both its ability to fit the data and its level of complexity. It imposes penalties on models with a higher number of parameters. AIC is calculated as 2k - 2log(L), where k represents the quantity of parameters in the model, while L denotes the likelihood of the data given the model. Smaller AIC values indicate better models, with a more optimal trade-off between accuracy and complexity. In logistic regression, AIC is used to analyze multiple models and determine the one that achieves the optimal trade-off between model fit and complexity.

Definition 4.4.6. ***Fisher Scoring Iterations:*** *The reported number of Fisher scoring iterations in the output signifies the total iterations needed for the method to achieve convergence and estimate the model's parameters.*

In the following subsections, the statistical goodness of fit statistics are applied in the forward and backward selection methods. Since the results of the WNRM are compartible with the Re-weighted Least Squares method via the glm [1] function in R, the software R is used in the rest of the analysis below.

4.4.1 Forward Selection

The forward selection method consists of the following steps.

Algorithm 4.4.1. ***Forward selection method***

*(1.) In **Step 1** Forward selection starts with an initial model that does not include any predictors, and thereafter incorporates one predictor variable at each iteration, based on the values of AIC and deviation.*

*(2.) In **Step 2** The predictor variable that contributes the most to improving the model fit is included in the model at each phase.*

*(3.) In **Step 3** The method continues until the model's fit is no longer significantly enhanced by the inclusion of an additional predictor.*

*(4.) In **Step 4** The Akaike Information Criterion (AIC) and deviance are utilized to assess the adequacy of the model during the forward selection procedure. Following the addition of each predictor, the model's fitness is assessed using AIC and deviation. The balance between model fit and complexity is determined by the AIC, where lower AIC values indicate models that fit better while also taking into account the complexity of the model. Similarly, deviation measures the extent of disparity between the observed and projected values, with lower deviance values indicating a more accurate fit of the model.*

*(5.) In **Step 5** If an additional regressor does not yield a significant improvement in the model's performance, it should not be included in the final model. Moreover, the relevance of each additional predictor variable is assessed by considering its p-value. A predictor is considered to have a substantial impact on the model fit if its p-value is below the set significance level of 0.05. Conversely, a predictor will be excluded from the final model if its p-value exceeds the significance level, suggesting that it does not contribute significantly to improving the model's fit.*

Applying Algorithm 4.4.1 to model (4.1), the following results are obtained.

1. In **Step 1**, the output for the logistic null model, that is, the model with only intercept is given below.

Coefficients	Estimate	Std. Error	z value	$Pr(>\|z\|)$
(Intercept)	-0.62362	0.07571	-8.237	$< 2 \times 10^{-16}$***

Table 4.4: Results of the initial step of forward selection only with intercept term.

2. In Step 2, the predictors are added into the null model in Step 1 on a one by one basis and the predictor that gives the lowest AIC is added to the model for Step 3. From Table 4.5 observe that the lowest AIC is obtained for "Glucose". Thus, glucose is addded to the model moving forward to step 3.

coefficients	Df	Deviance	AIC
+ Glucose	1	808.72	812.72
+ BMI	1	920.71	924.71
+ Age	1	950.72	954.72
+ Pregnancies	1	956.21	960.21
+ Diabetes Pedigree Function	1	970.86	974.86
+ Insulin	1	980.81	984.81
+ Skin Thickness	1	989.19	993.19
+ Blood Pressure	1	990.13	994.13
$< none >$		993.48	995.48

Table 4.5: Results of the 2nd step of forward selection. Each row indicates the addition of a predictor variable to the model, along with the corresponding degrees of freedom (Df), deviance, and Akaike Information Criterion (AIC) values.

(3.) In step 3, with "Glucose" already added to the model from Step 2, the next predictor that gives the lowest AIC at the current Step 3 will be added to the model. The output for this step is given in Table 4.6. Observe from Table 4.6 that "BMI" is the predictor with the lowest AIC. Hence, moving to step 4, "BMI" and "Glucose" will be added to the model.

coefficients	Df	Deviance	AIC
+ BMI	1	771.40	777.40
+ Pregnancies	1	784.95	790.95
+ Diabetes Pedigree Function	1	796.99	802.99
+ Age	1	797.36	803.36
$< none >$		808.72	812.72
+ Skin Thickness	1	807.07	813.07
+ Insulin	1	807.77	813.77
+ Blood Pressure	1	808.59	814.59

Table 4.6: Results of the 3rd step of forward selection along with the corresponding Df, deviance, and AIC values

(4.) In step 4, with "Glucose" and "BMI" already added to the model from Step 3, the next predictor that gives the lowest AIC at the current Step 4 will be added to the model. The output for this step is given in Table 4.7. Observe from Table 4.7 that "Pregnancies" is the predictor with the lowest AIC. Hence, moving to step 5, "BMI", "Glucose" and "Pregnancies" will be added to the model.

coefficients	Df	Deviance	AIC
+ Pregnancies	1	744.12	752.12
+ Age	1	755.68	763.68
+ Diabetes Pedigree Function	1	762.87	770.87
+ Insulin	1	767.79	775.79
+ Blood Pressure	1	769.07	777.07
$< none >$		771.40	777.40
+ Skin Thickness	1	770.20	778.20

Table 4.7: Results of the 4th step of forward selection along with the corresponding Df, deviance, and AIC values

(5.) In step 5, with "BMI", "Glucose" and "Pregnancies" already added to the model from Step 4, the next predictor that gives the lowest AIC at the current Step 5 will be added to the model. The output for this step is given in Table 4.8. Observe from Table 4.8 that "Diabetes Pedigree Function" is the predictor with the lowest AIC. Hence, moving to step 6, "BMI", "Glucose", "Pregnancies" and "Diabetes Pedigree Function" will be added to the model.

coefficients	Df	Deviance	AIC
+ Diabetes Pedigree Function	1	734.31	744.31
+ Blood Pressure	1	738.43	748.43
+ Age	1	742.10	752.10
$< none >$		744.12	752.12
+ Insulin	1	742.43	752.43
+ Skin Thickness	1	743.60	753.60

Table 4.8: Results of the 5th step of forward selection along with the corresponding Df, deviance, and AIC values

(6.) In step 6, with "BMI", "Glucose", "Pregnancies" and "Diabetes Pedigree Function" already added to the model from Step 5, the next predictor that gives the lowest AIC at the current Step 6 will be added to the model. The output for this step is given in Table 4.9. Observe from Table 4.9 that "Blood Pressure" is the predictor with the lowest AIC. Hence, moving to step 7, "BMI", "Glucose", "Pregnancies", "Diabetes Pedigree Function" and "Blood Pressure" will be added to the model.

coefficients	Df	Deviance	AIC
+ Blood Pressure	1	728.56	740.56
+ Insulin	1	731.51	743.51
$< none >$		734.31	744.31
+ Age	1	732.51	744.51
+ Skin Thickness	1	733.06	745.06

Table 4.9: Results of the 6th step of forward selection along with the corresponding Df, deviance, and AIC values

(7.) In step 7, with “BMI”, “Glucose”, “Pregnancies”, “Diabetes Pedigree Function” and “Blood Pressure” already added to the model from Step 6, the next predictor that gives the lowest AIC at the current Step 7 will be added to the model. The output for this step is given in Table 4.10. Observe from Table 4.10 that ”Age” is the predictor with the lowest AIC. Hence, moving to step 7, “BMI”, “Glucose”, “Pregnancies”, “Diabetes Pedigree Function”, “Blood Pressure” and “Age” will be added to the model.

coefficients	Df	Deviance	AIC
+ Age	1	725.46	739.46
+ Insulin	1	725.97	739.97
$< none >$		728.56	740.56
+ Skin Thickness	1	728.00	742.00

Table 4.10: Results of the 7th step of forward selection along with the corresponding Df, deviance, and AIC values

(8.) In step 8, with “BMI”, “Glucose”, “Pregnancies”, “Diabetes Pedigree Function” and “Blood Pressure” already added to the model from Step 7, the next predictor that gives the lowest AIC at the current Step 8 will be added to the model. The output for this step is given in Table 4.11. Observe from Table 4.11 that “Insulin” is the predictor with the lowest AIC. Hence, moving to step 9, “BMI”, “Glucose”, “Pregnancies”, “Diabetes Pedigree Function”, “Blood Pressure”, “Age” and “Insulin” will be added to the model.

coefficient	Df	Deviance	AIC
+ Insulin	1	723.45	739.45
$< none >$		725.46	739.46
+ Skin Thickness	1	725.19	741.19

Table 4.11: Results of the 8th step of forward selection along with the corresponding Df, deviance, and AIC values

(9.) In step 9, with "BMI", "Glucose", "Pregnancies", "Diabetes Pedigree Function", "Blood Pressure", "Age" and "Insulin" already added to the model from Step 8, the next predictor that gives the lowest AIC at the current Step 9 will be added to the model. The output for this step is given in Table 4.12. Observe from Table 4.12 that Skin Thickness is not yield a significant improvement in the model's performance, it will not be included in the final model.

coefficient	Df	Deviance	AIC
$< none >$		723.45	739.45
+ Skin Thickness	1	723.45	741.45

Table 4.12: Results of the 9th step of forward selection along with the corresponding Df, deviance, and AIC values

The final model for this step is given in Table 4.13.

Coefficients	Estimate	std.Error	z value	$Pr(>\|z\|)$
(Intercept)	-8.4051362	0.7167033	-11.727	$< 2 \times 10^{-16}$ ***
Pregnancies	0.1231724	0.0320688	3.841	0.000123 ***
Glucose	0.0351123	0.0036625	9.587	$< 2 \times 10^{-16}$ ***
Blood Pressure	-0.0132136	0.0051537	-2.564	0.010350 *
Insulin	-0.0011570	0.0008142	-1.421	0.155275
BMI	0.0900886	0.0144619	6.229	4.68×10^{-10} ***
Diabetes Pedigree Function	0.9475954	0.2980063	3.180	0.001474 **
Age	0.0147888	0.0092897	1.592	0.111393

Table 4.13: Results of the final step of forward selection including information on the estimated coefficients, standard errors, z-values, and corresponding p-values.

The R-code for forward selection method given in A.8

Remark 4.4.

Interpretation: Applying Forward Selection method, we can observe that the p-values for "Pregnancies","Glucose","Blood Pressure","BMI" and "Diabetes Pedigree Function" are lower than the significance criteria of 0.05, so we choose to include these variables in the final model. However, Since the p-values for "Age", and "Insulin" are higher than the significance criteria of 0.05, we choose not to include these variables in the final model. This implies that, based on forward selection and analysis of p-values, the variables "Pregnancies," "Glucose," "Blood Pressure," "BMI," and "Diabetes Pedigree Function" have a significant degree of association with the response variable while the predictors "Insulin," "Age," and "Skin Thickness"have a insignificant degree of association with the response variable.

4.4.2 BACKWARD SELECTION

Algorithm 4.4.2. ***Backward Selection method***

*(1.) In **Step 1** Backward elimination is a process that systematically eliminates predictor variables one by one, using the values of AIC and deviation as criteria. The process commences with a comprehensive model that includes all predictors.*

*(2.) In **Step 2** At each phase, the predictor variable that has the least contribution to the model fit is identified and removed from the model.*

*(3.) In **Step 3** The method continues until the model's fit is significantly enhanced by deleting predictors.*

*(4.) In **Step 4** The Akaike Information Criterion (AIC) and deviance are used to assess the adequacy of the model throughout the backward selection procedure. We eliminate the regressor that resulted in the most significant decrease in AIC and also exhibits a statistically significant reduction in AIC relative to the other predictor variable model. Furthermore, deviance is computed and in logistic regression, deviation is used as an indicator of the quality of fit; smaller deviance values indicate superior fits and the variable with the greatest p-value is eliminated from the model.*

Remark 4.5. *Backward selection is a method that begins with the whole model and gradually simplifies it by eliminating predictors, therefore exploring the range of useful models. In summary, The model selection criteria used for both forward and backward selection strategies are AIC (Akaike Information Criterion) and deviation.In our study, we included regressor variables for blood pressure, glucose, BMI, diabetes pedigree function, and pregnancies. By analyzing the data, we were able to create an optimal model. The p-values for Age and Insulin were found to be 0.111393 and 0.155275, respectively, which is above the*

significance level of 0.05. This indicates that the coefficients for Age and Insulin are not statistically significant, meaning that they do not make any meaningful contributions to the model.

Applying Algorithm 4.4.2 to the model (4.1) the following results are obtained.

(1.) In Step 1, the backward selection begins with the complete model that includes all predictor variables ("Pregnancies", "Glucose", "Blood Pressure", "Skin Thickness", "Insulin", "BMI", "Diabetes Pedigree Function", and "Age"). In the subsequent steps, the predictors will be eliminated on a one by one basis based on reduced AIC. At the current step 1, the output for the logistic model with all predictors is given in Table 4.14. Note that the starting AIC for the model in Table 4.14 is 741.45. Hence, moving to the next steps, the predictors that result in a lower AIC will remain in the model, while those that give a higher AIC value are eliminated.

Coefficients	Estimate	Std. Error	z value	$Pr(>\|z\|)$
(Intercept)	-8.4046964	0.7166359	-11.728	$< 2 \times 10^{-16}$ ***
Pregnancies	0.1231823	0.0320776	3.840	0.000123 ***
Glucose	0.0351637	0.0037087	9.481	$< 2 \times 10^{-16}$ ***
Blood Pressure	-0.0132955	0.0052336	-2.540	0.011072 *
Skin Thickness	0.0006190	0.0068994	0.090	0.928515
Insulin	-0.0011917	0.0009012	-1.322	0.186065
BMI	0.0897010	0.0150876	5.945	2.76×10^{-09} ***
DiabetesPedigreeFunction	0.9451797	0.2991475	3.160	0.001580 **
Age	0.0148690	0.0093348	1.593	0.111192

Table 4.14: Full model for Backward Selection including information on the estimated coefficients, standard errors, z-values, and corresponding p-values.

(2.) By removing the regressors one at a time, observe in Table 4.15 that the lowest AIC is obtained when "Skin Thickness" removed from the model in Step 1. Hence, moving to Step 3, "Skin Thickness" will no longer be a part of the model for the next selection.

coefficient	Df	Deviance	AIC
- SkinThickness	1	723.45	739.45
- Insulin	1	725.19	741.19
$< none >$		723.45	741.45
- Age	1	725.97	741.97
- BloodPressure	1	729.99	745.99
- DiabetesPedigreeFunction	1	733.78	749.78
- Pregnancies	1	738.68	754.68
- BMI	1	764.22	780.22
- Glucose	1	838.37	854.37

Table 4.15: Results of the 2nd step of backward selection along with the corresponding Df, deviance, and AIC values

(3.) From the remaining predictors: ("Pregnancies", "Glucose", "Blood Pressure", "Insulin", "BMI", "Diabetes Pedigree Function", and "Age") from Step 2, additional predictors are removed on a one by one basis, and the predictors that give the lowest AIC at this level will remain the model, while the predictor that results in a higher AIC is removed. Observe from Table 4.16 that the lowest AIC (739.45) is obtained by no longer removing any predictors from the model.

coefficient	Df	Deviance	AIC
$< none >$		723.45	739.45
- Insulin	1	725.46	739.46
- Age	1	725.97	739.97
- Blood Pressure	1	730.13	744.13
- Diabetes Pedigree Function	1	733.92	747.92
- Pregnancies	1	738.69	752.69
- BMI	1	768.77	782.77
- Glucose	1	840.87	854.87

Table 4.16: Results of the 3rd step of backward selection along with the corresponding Df, deviance, and AIC values

(4.) In the final step, the explicit logistic regression model with only the predictors ("Pregnancies", "Glucose", "Blood Pressure", "Insulin", "BMI", "Diabetes Pedigree Function", and "Age") and their p-values are shown in Table 4.18.

Coefficients	Estimate	Std. Error	z value	$Pr(>\|z\|)$
(Intercept)	-8.4051362	0.7167033	-11.727	$< 2 \times 10^{-16}$ ***
Pregnancies	0.1231724	0.0320688	3.841	0.000123 ***
Glucose	0.0351123	0.0036625	9.587	$< 2 \times 10^{-16}$ ***
Blood Pressure	-0.0132136	0.0051537	-2.564	0.010350 *
Insulin	-0.0011570	0.0008142	-1.421	0.155275
BMI	0.0900886	0.0144619	6.229	4.68×10^{-10} ***
Diabetes Pedigree Function	0.9475954	0.2980063	3.180	0.001474 **
Age	0.0147888	0.0092897	1.592	0.111393

Table 4.17: Results of the final step of backward selection including information on the estimated coefficients, standard errors, z-values, and corresponding p-values.

The R-code for the forward method is given in A.9.

Remark 4.6.

Interpretation: Applying Backward Selection method, we can observe that the p-values for "Pregnancies","Glucose","Blood Pressure","BMI" and "Diabetes Pedigree Function" are lower than the significance criteria of 0.05, so we choose to include these variables in the final model. However, Since the p-values for "Age", and "Insulin" are higher than the significance criteria of 0.05, we choose not to include these variables in the final model. This implies that, based on backward selection and analysis of p-values, the variables "Pregnancies," "Glucose," "Blood Pressure," "BMI," and "Diabetes Pedigree Function" have a significant degree of association with the response variable while the predictors "Insulin," "Age," and "Skin Thickness"have a insignificant degree of association with the response variable.

4.4.3 OPTIMUM MODEL

- *The Optimum model of our model (4.1) is given below:*

Coefficients	Estimate	Std. Error	z value	$Pr(>\|z\|)$
Intercept	-7.954952	0.675823	-11.771	$< 2 \times 10^{-16}$ ***
Pregnancies	0.153492	0.027835	5.514	3.5×10^{-08} ***
Glucose	0.034658	0.003394	10.213	$< 2 \times 10^{-16}$ ***
Blood Pressure	-0.012007	0.005031	-2.387	0.01700 *
BMI	0.084832	0.014125	6.006	1.9×10^{-09} ***
Diabetes Pedigree Function.	0.910628	0.294027	3.097	0.00195 **

Table 4.18: Results of the optimum model with estimated values, std. Error, Z value and P value.

4.5 THESIS CONCLUSION

This paper presents the development and evaluation of a new Weighted Newton Raphson Method (WNRM) for the analysis of diabetic data. The model, which includes a Jacobian matrix and a unique weighting scheme, enables the application of Maximum Likelihood Estimation (MLE) in logistic regression for data that is non repeated. Furthermore, an overview of the WNRM algorithm and its practical implementation has been provided. In addition, the model incorporates both forward and backward selection approaches, and employs p-value analysis to determine the most suitable model fit. In addition, a thorough comparison has been provided between the traditional Newton Raphson Method (NRM) and WNRM, focusing specifically on their suitability for dealing with repetitive data.

REFERENCES

[1] Generalized linear models in R, https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/glm, March 2024.

[2] Kaggle, https://www.kaggle.com/datasets/kandij/diabetes-dataset, Feb 2024.

[3] J. Neter, M. H. Kutner, C. J. Natchtsheim, W. Wasserman, *Applied Linear Statistical Models*, Irwin, Chicago, 1996.

[4] G. James, D. Witten, T. Hastie, R. Tibshirani, *An Introduction to Statistical Learning with Applications in R*, Springer, 2014.

[5] J. L. Hueso, E. Martínez, J. R. Torregrosa, Modified Newton's method for systems of nonlinear equations with singular Jacobian, *Journal of Computational and Applied Mathematics*, 224 (2009), 77-83.

[6] S. Weerakoon, T.G.I. Fernando, *A variant of Newton's method with accelerated third-order convergence*, *Applied Mathematics Letters*, 13(8) (2000), 87-93.

[7] J.F. Traub, *Iterative Methods for the Solution of Equations*, *Chelsea Publishing Company*, New York, 312 1982.

[8] D. C. Montgomery, E. A. Peck, G. G. Vining, *Introduction to Linear Regression Analysis*, p. cm. – *John Wiley & Sons*, 821 2021.

[9] X.Wu, *Note on the improvement of Newton's method for systems of nonlinear equations*, *Applied Mathematics and Computation*, 189 (2007), 1476-1479.

[10] A. Beck *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB*, *Society for Industrial and Applied Mathematics*,1st edition, 2014.

[11] T. Hastie, R. Tibshirani, J. Friedman *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*, *Springer Series in Statistics*, Second Edition, 2009.

[12] B. Abraham, J. Ledolter *Introduction to Regression Modeling*, *John Wiley & Sons*, United States, 1976.

[13] A. Kharab, R. B. Guenther *An introduction to Numerical Methods, A MATLAB Approach*, *CRC press*, Fifth Edition, 2023.

[14] W. Cheney, D. Kincaid *Numerical Mathematics and Computing*, *International Thomson Publishing*, 1998.

[15] R.L.Burden, J.D.Faires *Numerical Analysis*, *Brooks Cole*, (2001): 574.

[16] M. W. Yusuf, L. W. June and M. A. Hassan *Jacobian-Free Diagonal Newton's Method for Solving Nonlinear Systems with Singular Jacobian*, *Malaysian Journal of Mathematical Sciences*, 5(2) (2011), 241-255.

[17] M. Y. Waziri., W. J. Leong, M. A. Hassan and M. Monsi *An Efficient Solver for Systems of Nonlinear Equations with Singular Jacobian via Diagonal Updating*, *Applied Mathematical Sciences*, 4 (2010), 3403 - 3412.

[18] M. W. Yusuf, I. Saidu. *A Comparative Study Of Diagonal Updating Newton Methods For Systems Of Nonlinear Equations With Singular Jacobian*, *ARPN Journal of Engineering and Applied Sciences*, 5 (2010), 39-47.

[19] S. Han. *Estimation and inference with a (nearly) singular Jacobian*, *Quantitative Economics*, 10 (2019), 1019–1068.

[20] H. Okawa., K. Fujisawa., Y. Yamamoto., R. Hirai., N. Yasutake., H. Nagakura., S. Yamada. *The W4 method: A new multi-dimensional root-finding scheme for nonlinear systems of equations*, *Applied Numerical Mathematics*, 183 (2023), 157-172.

[21] Schnabel, Robert B., and Paul D. Frank. *Tensor Methods for Nonlinear Equations*, *SIAM Journal on Numerical Analysis*., vol. 21, no. 5, 1984, pp. 815–43. JSTOR, `http://www.jstor.org/stable/2156931`, Accessed 27 Mar. 2024.

Appendix A

THE WEIGHTED NEWTON RAPHSON METHOD (WNRM)

A.1 R-CODE FOR EXAMPLE 2.1

The R code for Example 2.1 is given below.

Computer-Code A.1.

```
x = c(0.2,0.2,-0.2)
F = function(x) {
  f1 = sum(3 * x[1] - cos(x[2] * x[3]) - 0.5)
  f2 = sum((x[1])^2 - 625 * (x[2]^2) - 0.25)
  f3 = sum(exp(-x[1] * x[2]) + 20 * x[3] + (10 * pi - 3) / 3)
  res = c(f1,f2,f3)
  return(res)
}
#Jacobian matrix
J = function(x) {
  res = matrix(c(
    3, sum(x[3] * sin(x[2] * x[3])), sum(x[2] * sin(x[2] * x[3])),
    sum(2 * x[1]), sum(-1250 * x[2]), 0,
    sum(-x[2] * exp(-x[1] * x[2])), sum(-x[1] * exp(-x[1] * x[2])), 20)
    ,nrow=3, byrow=T)
  return(res)
}
x_old = c(0, 0, 0)
x.vector = x
#F(x)
```

```
#J(x)
error = c()
for (i in 1:31) {
  error = c(error, sum(abs(F(x))))
  x_new = x - solve(J(x)) %*% F(x)
  x = x_new
  #print(beta)
  x.vector = c(x.vector, x)
}
x
```

A.2 R CODE FOR EXAMPLE 2.5

The R code for Example 2.5 is given below.

Computer-Code A.2.

```
x <- c(0.2, 0.2, -0.2)
F = function(x) {
  f1 = sum(3 * x[1] - cos(x[2] * x[3]) - 0.5)
  f2 = sum((x[1])^2 - 625 * (x[2]^2) - 0.25)
  f3 = sum(exp(-x[1] * x[2]) + 20 * x[3] + (10 * pi - 3) / 3)
  res = c(f1,f2,f3)
  return(res)
}
#Jacobian matrix
J = function(x) {
  res = matrix(c(
```

```
  3, sum(x[3] * sin(x[2] * x[3])), sum(x[2] * sin(x[2] * x[3])),
  sum(2 * x[1]), sum(-1250 * x[2]), 0,
  sum(-x[2] * exp(-x[1] * x[2])), sum(-x[1] * exp(-x[1] * x[2])), 20)
  ,nrow=3, byrow=T)
return(res)
}
g_old = c(0, 0, 0)
x_old = c(0, 0, 0)
x.vector = x
error = c()
for (i in 1:33) {
  g = x - (solve(J(x))) %*% F(x)
  error = c(error, sum(abs(F(x))))
  m = rep(0, length(x))
  for (i in 1:length(x)) {
    if (abs(x[i] - x_old[i]) < 0.00001) {
      m[i] = 1 / (1 - 0)
    } else {
      m[i] = 1 / (1 - (g[i] - g_old[i]) / (x[i] - x_old[i]))
    }
  }
  m = diag(m)
  x_new = x - solve(J(x)) %*% m %*% F(x)
  x_old = x
  x = x_new
  g_old = g
```

```
  x.vector = c(x.vector, x)
}
print(x_new)
```

A.3 R CODE FOR EXAMPLE 3.19

The R code for Example 3.19 is given below.

Computer-Code A.3.

```
#Enter the data points
x=c(1.5,2,3.5,2.5,1,1.3)
y=c(0,1,0,1,1,0)

#Enter the function f1,f2,f3
F = function(beta, x, y) {
  f1 = sum(y - 1/(1 + exp(-beta[1] - beta[2] * x ^ 2
  - beta[3] * exp(x))))
  f2 = sum((y - 1/(1 + exp(-beta[1] - beta[2] * x ^ 2
  - beta[3] * exp(x))))*x^2)
  f3 = sum((y - 1/(1 + exp(-beta[1] - beta[2] * x ^ 2
  - beta[3] * exp(x))))*exp(x))
  res = c(f1,f2,f3)
  return(res)
}
#Jacobian matrix
J = function(beta, x) {
  coef = exp(- beta[1] - beta[2] * x ^ 2
```

```
  - beta[3] * exp(x))
  coef = -(coef / (1 + coef) ^ 2)
  res = matrix(c(
    sum(coef), sum(coef * x^2), sum(coef * exp(x)),
    sum(coef * x^2), sum(coef * x^4), sum(coef * x^2 * exp(x)),
    sum(coef * exp(x)), sum(coef * x^2 * exp(x)), sum(coef * exp(2*x))
  ), nrow=3, byrow=T)
  return(res)
}
beta = c(1, 2, 3)
error = c()
for (i in 1:6) {
  error = c(error, sum(abs(F(beta, x, y))))
  beta_new = beta - solve(J(beta, x)) %*% F(beta, x, y)
  beta = beta_new
  print(beta)
}
```

A.4 R CODE FOR EXAMPLE 3.19

The R code for Example 3.19 is given below.

Computer-Code A.4.

```
%%This is the R-code this example
x=c(1.5,2,3.5,2.5,1,1.3)
y=c(0,1,0,1,1,0)
```

```
F = function(beta, x, y) {
  f1 = sum(y - 1/(1 + exp(-beta[1] - beta[2] * x ^ 2
  - beta[3] * exp(x))))
  f2 = sum((y - 1/(1 + exp(-beta[1] - beta[2] * x ^ 2
  - beta[3] * exp(x))))*x^2)
  f3 = sum((y - 1/(1 + exp(-beta[1] - beta[2] * x ^ 2
  - beta[3] * exp(x))))*exp(x))
  res = c(f1,f2,f3)
  return(res)
}
J = function(beta, x) {
  coef = exp(- beta[1] - beta[2] * x ^ 2 - beta[3] * exp(x))
  coef = -(coef / (1 + coef) ^ 2)
  res = matrix(c(
    sum(coef), sum(coef * x^2), sum(coef * exp(x)),
    sum(coef * x^2), sum(coef * x^4), sum(coef * x^2 * exp(x)),
    sum(coef * exp(x)), sum(coef * x^2 * exp(x)), sum(coef * exp(2*x))
  ), nrow=3, byrow=T)
  return(res)
}
g_old = c(0, 0, 0)
beta_old = c(0, 0, 0)
beta = c(1, 2, 3)
error = c()
for (i in 1:6) {
  g = beta - (solve(J(beta, x))) %*% F(beta, x, y)
```

```
  error = c(error, sum(abs(F(beta, x, y))))
  m = 1 / (1 - (g - g_old) / (beta - beta_old))
  m = diag(m[,1])
  beta_new = beta - solve(J(beta, x)) %*% m %*% F(beta, x, y)
  beta_old = beta
  beta = beta_new
  g_old = g
  print(beta)
}
```

A.5 R CODE FOR EXAMPLE 3.19

The R code for Example 3.19 is given below.

Computer-Code A.5.

```
%%This is the R-code this example
data=data.frame(repeatedlogisic_sample2)
data_subset <- data[-1, ]
x <- c(data_subset$Levels)
y <- as.vector(rowSums(data_subset[, c(3:12)], na.rm = TRUE))
data_subset1 <- data_subset[, -c(1, 2)]
n <- as.vector(rowSums(!is.na(data_subset1)))
#Enter the function f1,f2,f3
F = function(beta, x, y,n) {
  f1 = sum(y- n/(1 + exp(-beta[1] - beta[2] * (x^2)
  - beta[3] * exp(x))))
  f2 = sum((y - n/(1 + exp(-beta[1] - beta[2] * (x)^2
```

```
  - beta[3] * exp(x))))*(x^2))
  f3 = sum(y - n/(1 + exp(-beta[1] - beta[2] * (x)^2
  - beta[3] * exp(x)))*exp(x))
  res = c(f1,f2,f3)
  return(res)
}
#Jacobian matrix
J = function(beta, x,n) {
  coef = exp(- beta[1] - beta[2] * (x)^2 - beta[3] * exp(x))
  coef = -n * (coef / (1 + coef) ^ 2)
  res = matrix(c(
    sum(coef), sum(coef * (x)^2), sum(coef * exp(x)),
    sum(coef * (x)^2), sum(coef * (x)^4), sum(coef * (x)^2 * exp(x)),
    sum(coef * exp(x)), sum(coef * exp(x) * (x^2)), sum(coef * exp(x)^2)
  ), nrow=3, byrow=T)
  return(res)
}
g_old = c(0, 0, 0)
beta_old = c(0, 0, 0)
beta = c(0.1,0.2,0.3)
m <- length(x)
beta.vector = beta
%F(beta,x,y,n)
%J(beta,x,n)
error = c()
for (j in 1:m) {
```

```
  g = beta - (solve(J(beta,x,n))) %*% F(beta,x,y,n)
  error = c(error, sum(abs(F(beta,x,y,n))))
  mi = rep(0, length(beta))
  for (j in 1:length(beta)) {
    if (abs(beta[j] - beta_old[j]) < 0.000001) {
      mi[j] = 1 / (1 - 0)
    } else {
      mi[j] = 1 / (1 - (g[j] - g_old[j]) / (beta[j] - beta_old[j]))
    }
  }
  mi = diag(mi)
  beta_new = beta - solve(J(beta,x,n)) %*% mi %*% F(beta,x,y,n)
  beta_old = beta
  beta = beta_new
  g_old = g
  beta.vector = c(beta.vector, beta)
}
print(error)
print(beta_new)
```

A.6 R CODE FOR EXAMPLE 4.1

The R code for Example 4.1 is given below.

Computer-Code A.6.

```
library(readr)
diabetes_data <- read_csv("diabetes_data.csv",
```

```
    col_types = cols(Pregnancies = col_number(),
    Glucose = col_number(), BloodPressure = col_number(),
    SkinThickness = col_number(), Insulin = col_number(),
    BMI = col_number(), DiabetesPedigreeFunction = col_number(),
    Age = col_number(), Outcome = col_number()))
View(diabetes_data)
#Enter the function f1,f2,f3,f4,f5,f6,f7,f8
F = function(beta) {
  f1 = sum(diabetes_data$Outcome - 1/(1 + exp(-beta[1]
  - beta[2] * (diabetes_data$Pregnancies)
  - beta[3] * (diabetes_data$Glucose)
  - beta[4] * (diabetes_data$BloodPressure)
  - beta[5] * (diabetes_data$SkinThickness)
  -beta[6] * (diabetes_data$Insulin)
  - beta[7] * (diabetes_data$BMI)
  - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
  - beta[9] * (diabetes_data$Age))))

  f2 = sum((diabetes_data$Outcome - 1/(1 + exp(-beta[1]
  - beta[2] * (diabetes_data$Pregnancies)
  - beta[3] * (diabetes_data$Glucose)
  - beta[4] * (diabetes_data$BloodPressure)
  - beta[5] *(diabetes_data$SkinThickness)
 -beta[6] * (diabetes_data$Insulin)
 - beta[7] * (diabetes_data$BMI)
 - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
```

```
- beta[9] * (diabetes_data$Age))))*(diabetes_data$Pregnancies))

f3 = sum((diabetes_data$Outcome - 1/(1 + exp(-beta[1]
- beta[2] * (diabetes_data$Pregnancies)
      - beta[3] * (diabetes_data$Glucose)
      - beta[4] * (diabetes_data$BloodPressure)
      - beta[5] * (diabetes_data$SkinThickness)
      -beta[6] * (diabetes_data$Insulin)
      - beta[7] * (diabetes_data$BMI) - beta[8] * (diabetes_data$Diabete
      - beta[9] * (diabetes_data$Age))))*(diabetes_data$Glucose))

f4 = sum((diabetes_data$Outcome - 1/(1 + exp(-beta[1]
- beta[2] * (diabetes_data$Pregnancies)
      - beta[3] * (diabetes_data$Glucose)
      - beta[4] * (diabetes_data$BloodPressure)
      - beta[5] * (diabetes_data$SkinThickness)
      -beta[6] * (diabetes_data$Insulin)
      - beta[7] * (diabetes_data$BMI)
      - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
      - beta[9] * (diabetes_data$Age))))*(diabetes_data$BloodPressure))

f5 = sum((diabetes_data$Outcome - 1/(1 + exp(-beta[1]
- beta[2] * (diabetes_data$Pregnancies)
      - beta[3] * (diabetes_data$Glucose)
      - beta[4] * (diabetes_data$BloodPressure)
      - beta[5] * (diabetes_data$SkinThickness)
```

```
      -beta[6] * (diabetes_data$Insulin)
      - beta[7] * (diabetes_data$BMI)
      - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
      - beta[9] * (diabetes_data$Age))))*(diabetes_data$SkinThickness))

f6 = sum((diabetes_data$Outcome - 1/(1 + exp(-beta[1]
- beta[2] * (diabetes_data$Pregnancies)
      - beta[3] * (diabetes_data$Glucose)
      - beta[4] * (diabetes_data$BloodPressure)
      - beta[5] * (diabetes_data$SkinThickness)
      -beta[6] * (diabetes_data$Insulin)
      - beta[7] * (diabetes_data$BMI)
      - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
      - beta[9] * (diabetes_data$Age))))*(diabetes_data$Insulin))

f7 = sum((diabetes_data$Outcome
- 1/(1 + exp(-beta[1] - beta[2] * (diabetes_data$Pregnancies)
      - beta[3] * (diabetes_data$Glucose)
      - beta[4] * (diabetes_data$BloodPressure)
      - beta[5] * (diabetes_data$SkinThickness)
      - beta[6] * (diabetes_data$Insulin)
      - beta[7] * (diabetes_data$BMI)
      - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
      - beta[9] * (diabetes_data$Age))))*(diabetes_data$BMI))

f8 = sum((diabetes_data$Outcome - 1/(1 + exp(-beta[1]
```

```
  - beta[2] * (diabetes_data$Pregnancies)
        - beta[3] * (diabetes_data$Glucose)
        - beta[4] * (diabetes_data$BloodPressure)
        - beta[5] * (diabetes_data$SkinThickness)
        - beta[6] * (diabetes_data$Insulin)
        - beta[7] * (diabetes_data$BMI)
        - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
        - beta[9] * (diabetes_data$Age))))*(diabetes_data$DiabetesPedigree

  f9 = sum((diabetes_data$Outcome - 1/(1 + exp(-beta[1]
  - beta[2] * (diabetes_data$Pregnancies)
        - beta[3] * (diabetes_data$Glucose)
        - beta[4] * (diabetes_data$BloodPressure)
        - beta[5] * (diabetes_data$SkinThickness)
        -beta[6] * (diabetes_data$Insulin)
        - beta[7] * (diabetes_data$BMI)
        - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
        - beta[9] * (diabetes_data$Age))))*(diabetes_data$Age))
  res = c(f1,f2,f3,f4,f5,f6,f7,f8,f9)
  return(res)
}
#Jacobian matrix
J = function(beta) {
  coef = exp(- beta[1] - beta[2] * (diabetes_data$Pregnancies)
  - beta[3] * (diabetes_data$Glucose)
        -  beta[4] * (diabetes_data$BloodPressure)
```

```
    - beta[5] * (diabetes_data$SkinThickness)
    - beta [6] * (diabetes_data$Insulin)
    - beta[7] * (diabetes_data$BMI)
    - beta[8] * (diabetes_data$DiabetesPedigreeFunction)
    - beta[9] * (diabetes_data$Age))

coef = -(coef / (1 + coef) ^ 2)
res = matrix(c(
  sum(coef), sum(coef * (diabetes_data$Pregnancies)), sum(coef * (diab

  sum(coef * (diabetes_data$Pregnancies)), sum(coef * (diabetes_data$P

  sum(coef * (diabetes_data$Glucose)), sum(coef * (diabetes_data$Pregn

  sum(coef * (diabetes_data$BloodPressure)), sum(coef * (diabetes_data$

  sum(coef * (diabetes_data$SkinThickness)), sum(coef * (diabetes_data$

  sum(coef * (diabetes_data$Insulin)), sum(coef * (diabetes_data$Pregn

  sum(coef * (diabetes_data$BMI)), sum(coef * (diabetes_data$Pregnanci

  sum(coef * (diabetes_data$DiabetesPedigreeFunction)), sum(coef * (di

  sum(coef * (diabetes_data$Age)), sum(coef * (diabetes_data$Pregnanci
), nrow=9, byrow=T)
```

```
  return(res)
}
g_old = c(0, 0, 0, 0, 0, 0, 0, 0, 0)
beta_old = c(0, 0, 0, 0, 0, 0, 0, 0, 0)
beta = c(0.01, 0.02, 0.03, 0.07, 0.04, 0.1, 0.05, 0.06, 0.02)
error = c()
for (i in 1:40) {
  g = beta - (solve(J(beta))) %*% F(beta)
  error = c(error, sum(abs(F(beta))))
  m = rep(0, length(beta))
  for (i in 1:length(beta)) {
    if (abs(beta[i] - beta_old[i]) < 0.000001) {
      m[i] = 1 / (1 - 0)
    } else {
      m[i] = 1 / (1 - (g[i] - g_old[i]) / (beta[i] - beta_old[i]))
    }
  }
  m = diag(m)
  beta_new = beta - solve(J(beta)) %*% m %*% F(beta)
  beta_old = beta
  beta = beta_new
  g_old = g
}
print(error)
print(beta_new)
```

A.7 R CODE FOR EXAMPLE 4.1

The R code for Example 4.1 is given below.

```
library(readr)
diabetes_data <- read_csv("diabetes_data.csv",
                          col_types = cols(Pregnancies = col_number(),
    Glucose = col_number(), BloodPressure = col_number(),
    SkinThickness = col_number(), Insulin = col_number(),
    BMI = col_number(), DiabetesPedigreeFunction = col_number(),
    Age = col_number(), Outcome = col_number()))
View(diabetes_data)

diabetes_data$Outcome <- as.factor(diabetes_data$Outcome)
log_m <- glm(Outcome ~ Pregnancies+Glucose+BloodPressure+SkinThickness
             +Insulin+BMI+DiabetesPedigreeFunction+Age,
             data=diabetes_data, family = binomial(link = "logit"))
summary(log_m)
```

A.8 R CODE FOR EXAMPLE 4.1

The R code for Model 4.1 is given below.

Computer-Code A.7.

```
library(readr)
diabetes_data <- read_csv("diabetes_data.csv",
                    col_types = cols(Pregnancies = col_number(),
                    Glucose = col_number(),
                    BloodPressure = col_number(),
                    SkinThickness = col_number(),
```

```
                    Insulin = col_number(),
                    BMI = col_number(),
                    DiabetesPedigreeFunction = col_number(),
                    Age = col_number(), Outcome = col_number()))
#View(diabetes_data)
diabetes_data
full_model <- glm(Outcome ~ ., data = diabetes_data, family =
binomial(link = "logit"))
summary(full_model)
# Forward selection (start with no regressor)
# Step 1
f1 <- glm(Outcome ~ 1, data = diabetes_data,
family = binomial(link = "logit"))
summary(f1)
# Step 2
step_result <- step(f1, direction = "forward", scope =
formula(full_model))
step_result
```

A.9 R CODE FOR EXAMPLE 3.19

The R code for Model 3.19 is given below.

Computer-Code A.8.

```
library(readr)
diabetes_data <- read_csv("diabetes_data.csv",
        col_types = cols(Pregnancies = col_number(),
```

```
        Glucose = col_number(), BloodPressure = col_number(),
        SkinThickness = col_number(), Insulin = col_number(),
        BMI = col_number(), DiabetesPedigreeFunction = col_number(),
        Age = col_number(), Outcome = col_number()))
View(diabetes_data)
full_model <- glm(Outcome ~ ., data = diabetes_data, family = binomial(li
summary(full_model)
# Forward selection (start with no regressor)
# Step 1
f1 <- glm(Outcome ~ 1, data = diabetes_data,
family = binomial(link = "logit"))
summary(f1)
# Step 2
step_result <- step(f1, direction = "forward", scope = formula(full_mode
step_result
# Backward selection (start with all regressor)
step_result <- step(full_model, direction = "backward")
step_result
# Step 3 (Final model after forward & backward selection)
final_model <- glm(Outcome ~ Pregnancies + Glucose +
BloodPressure + Insulin +
        BMI + DiabetesPedigreeFunction + Age,
        data = diabetes_data, family = binomial(link = "logit"))
summary(final_model)
step_result
```

Appendix B

APPLICATION OF THE WNRM TO DIABETES DATA

B.1 COMPUTER CODE FOR THE WEIGHTED NEWTON-RAPHSON ALGORITHM IN THE DIABETES DATA

The following is R-code for numerically solving the iterative equation (A.6).

B.2 COMPUTER CODE FOR THE REWEIGHTED LEAST SQUARES ALGORITHM IN THE "GLM" FUNCTION IN R FOR THE DIABETES DATA

(A.7) The output of Reweighted Least Squares algorithm:

Call: $glm(formula = Outcome \sim Pregnancies + Glucose + BloodPressure + SkinThickness + Insulin + BMI + DiabetesPedigreeF. + Age, family = binomial(link = "logit"), data = diabetes_data)$

	Coefficients:	Estimate	Std.Error	z value	$Pr(>\|z\|)$
	(Intercept)	-8.4046964	0.7166359	-11.728	$< 2e-16$ ***
1	Pregnancies	0.1231823	0.0320776	3.840	0.000123 ***
2	Glucose	0.0351637	0.0037087	9.481	$< 2e-16$ ***
3	Blood Pressure	-0.0132955	0.0052336	-2.540	0.011072 *
4	Skin Thickness	0.0006190	0.0068994	0.090	0.928515
5	Insulin	-0.0011917	0.0009012	-1.322	0.186065
6	BMI	0.0897010	0.0150876	5.945	2.76e-09 ***
7	Diabetes Pedigree F.	0.9451797	0.2991475	3.160	0.001580 **
8	Age	0.0148690	0.0093348	1.593	0.111192

Table B.1: Results for the Reweighted Least Squares including information with on the estimated coefficients, standard errors, z-values, and corresponding p-values

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 (Dispersion parameter for binomial family taken to be 1)

Null deviance: 993.48 on 767 degrees of freedom

Residual deviance: 723.45 on 759 degrees of freedom

AIC: 741.45

Number of Fisher Scoring iterations: 5