2018

Preference Probability Based on Ranks - A New Approach Using Logistic Regression with Zero Intercept

Oluwagbenga David Agboola
agboola@marshall.edu

PREFERENCE PROBABILITY BASED ON RANKS - A NEW APPROACH USING LOGISTIC REGRESSION WITH ZERO INTERCEPT

A thesis submitted to
the Graduate College of
Marshall University
In partial fulfillment of
the requirements for the degree of
Master of Arts
in
Mathematics
by
Agboola Oluwagbenga David
Approved by
Dr. Laura Adkins, Committee Chairperson
Dr. Scott Sarra
Dr. Raid Al-Aqtash

MARSHALL UNIVERSITY
MAY 2018

APPROVAL OF THESIS

We, the faculty supervising the work of Agboola Oluwagbenga David, affirm that the thesis, *Preference Probability Based on Ranks - A New Approach Using Logistic Regression with Zero Intercept*, meets the high academic standards for original scholarship and creative work established by the Department of Mathematics and the College of Science. This work also conforms to the editorial standards of our discipline and the Graduate College of Marshall University. With our signatures, we approve the manuscript for publication.

Laura Adkins 5-1-2018

Dr. Laura Adkins, Department of Mathematics Committee Chairperson Date

Scott Sarra 5-1-2018

Dr. Scott Sarra, Department of Mathematics Committee Member Date

Raid Aqtash 5-2-2018

Dr. Raid Al-Aqtash, Department of Mathematics Committee Member Date

ACKNOWLEDGEMENTS

First and foremost, I give thanks to Jesus Christ my Lord and Saviour for giving me the wisdom and grace to complete and defend this paper.

I appreciate my parents - Mr. & Mrs. D.O. Agboola, the family of Mr. Olufunso Fasooto, and my siblings - Damilola, Tobiloba and Tomisin for their constant prayers and support. I would not have gone this far without you all.

I sincerely appreciate Dr. Laura Adkins for her relentless help in completing this paper and for being the committee chairperson. I am humbled by her unwavering passion to always guide and give suggestions throughout the course of this project. You are the best supervisor I could ever wish for. Thank you, Dr. Scott Sarra for working with me when writing the Python codes and for your feedback on my several drafts especially on chapter 3. I am grateful for Dr. Raid Al-Aqtash for agreeing to be a member of my committee and for his support. I thank the information technology department for helping with dual booting on my PC.

I appreciate my soccer pal, Kevin Joo for his input in translating the work of Jung, Hong and Lee (2010) [5] from Korean to English. Dr. Carl Mummert is awesome, a very supportive advisor. I appreciate your help while using LaTex, for helping with the formatting of this paper and for being such an amazing teacher.

Thank you Dr. Bonita Lawrence for being like a mother to me. It was a privilege to work with you in the only publicly accessible Differential Analyzer lab in the country. I have learned a lot from you via the travels, how you hold on to your dreams and how you expect everyone to be treated fairly. The memories shared will always be in my heart. I appreciate you, ma.

Sincere greetings to Dr. Alfred Akinsete and his family, for their regular support throughout my stay in Marshall. I appreciate Dr. Michael Otunuga and his family for being there for me. I say thank you to the family of Larry and Randi Butcher for making me a part of their family. I do not take that for granted. My heart blesses Rob Ely and BCM for being a true ambassador of Christ here on earth.

I am grateful for the professors who taught me, those I worked with, and the entire faculty members of Mathematics Department. I also recognize Eyoel Berhane, Chad Lott, my colleagues, and other graduate assistants in the department.

To my friends Damilola Okunrounmu, Damilola Amosun, Ayobami Bakare, Adi Vershima and Corey Tornes, you are friends that stick closer than a brother. And to Elizabeth Otunuga, Boluwatife Babajide and LaShonda McDowell, I say thank you for your kindness.

Time and space will fail me to list out all my friends who have been there for me throughout my stay at Marshall. To you all, I say thank you!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Many probability models have been proposed to describe rankings. One of these is the Bradley-Terry model, which is based on observed pairwise preferences. For this study, we reverse the case and propose a new approach for estimating pairwise preference probabilities based on observed rankings. The new approach uses logistic regression with zero intercept as the statistical model that fits this situation. In order to implement the model, we first estimate the parameter using maximum likelihood estimation. Then we evaluate this estimation using numerical approximation procedures. We consider three such procedures: bisection method, Newton-Raphson method, and improved Newton's method. Using simulated data, we compare the three procedures based on the number of iterations required for convergence, as well as CPU time. We identify the improved Newton's method as the fastest of the three methods.

CHAPTER 1

INTRODUCTION

For this study, we propose a statistical model that can estimate or predict preference probability of a pairwise comparison using known rankings; that is, we are estimating the probability that one item is preferred to another given each item's rank. Let us say that we have two equally ranked brands B_α and B_β; in this case either brand should have equal chances of being chosen. That is, the probability either brand's product is selected by some consumer is 0.5, so both brands have equal chances of being selected by some consumer. According to Holland and Wessells (1998) [4], predicting preference can be based on some specific attributes of the goods or product. But for this study, we do not intend to go into details as to other factors that could influence the consumer's preference except for the given rank.

Another instance can be found in the sporting context. If we have two teams A_α and A_β that are ranked first in their respective leagues, it is logical to say that both teams would have equal chances of winning. Thus the probability that team A_α beats team A_β (and vice-versa) is 0.5. We know that there are various factors that can influence the chances of a team winning a game, some of which were considered by Willoughby (2002) [16] with respect to Canadian Football. However, we take the position that the ranks incorporate all these factors and, thus, we intend to base the preference probability on these ranks alone. In fact, we will base preference probabilities on only the difference between the two ranks. Another example is racetrack betting as discussed by Lo, Bacon-Shone and Busche (1995) [9]. Further examples can be found in tennis, soccer, and other team sports.

Considering the historical use of ranking models in paired comparisons, we observe that the most popular of such models is the Bradley-Terry model. This probability model is used to predict the outcome of a pairwise comparison. As discussed by Jong-June and Yongdai (2014) [7], the Bradley-Terry model has been used to estimate ranking probabilities which are based on observed preferences.

Now, we intend to reverse this concept by estimating preference probabilities which are based on observed rankings. Then we consider a situation in which it is possible to apply this concept, stating all the properties our proposed model should satisfy. We would explore a particular regression method - logistic regression - and show that a special case in which the intercept is zero satisfies all the required properties. Our choice of logistic regression model is based on its use for analyzing data with categorical or binary outcome.

In the following chapters, we will do a review of the Bradley-Terry model and logistic regression model. Then we will identify all the required properties for the situation considered and then prove that logistic regression model with zero intercept satisfies all of the required properties. Then we apply the method of maximum likelihood estimation to fit our model. Afterwards, we will discuss and compare three numerical approximation procedures we have chosen to estimate the parameter β_1, identifying the fastest and most efficient of the three procedures. We illustrate this by simulating data to compare these numerical procedures, by fixing the values of β_1 and x but with varying values of y. We explore the distribution of the estimator for small samples. And finally, we conclude by stating our findings, and some limitations encountered that lead to future work.

CHAPTER 2

PREFERENCE PROBABILITY USING LOGISTIC REGRESSION

In this section, we review the application of the Bradley-Terry model, study logistic regression, fit the special case of zero intercept and examine how it applies to this study.

2.1 The Bradley-Terry Model

The Bradley-Terry model, though studied in the 1920s by Ernst Zermelo, is named after R.A Bradley and M.E Terry who presented the model in 1952 in their paper titled "Rank analysis of incomplete block designs: I. The method of paired comparisons" [1]. Bradley-Terry model is one of several models used in the analysis of categorical or dichotomous data and can be viewed as a special case of generalized linear models. The Bradley-Terry model is for paired comparison or for analyzing pairwise preference data. For any pair of entities u and v, selected from some sample, the probability that u ranks higher than v can be estimated using this model.

Now, let us consider a situation where k entities are in pairwise comparisons to one another, given their order of preference: $\tau_1, \cdots, \tau_k$. For this model, we have the condition that every $\tau_i \geq 0$ with

$$\sum_{i=1}^{k} \tau_i = 1$$

(which implies that there is no chance for a tie), so that it is expressed as:

$$P(u \text{ ranks higher than } v) = \frac{\tau_u}{\tau_u + \tau_v}.$$

where τ_u and τ_v are positive real-valued score functions assigned to u and v respectively. In using this model, entities from the sample are considered to have true ratings (or preferences); thus, the estimated ranking is based on these preferences.

We note that for mutually independent events, the probability $p_{uv} = P(u \text{ beats } v)$ satisfies the logit model (and removing ties):

$$\log \frac{p_{uv}}{1 - p_{uv}} = \psi_u - \psi_v,$$

where $\psi_u = \log \tau_u$, as expressed by Bradley and Terry (1952) [1].

According to Jong-June and Yongdai (2014) [7], it was explained that the popularity of the Bradley-Terry model is gained not only due to its easy computation but also because of how it exhibits some nice asymptotic properties when the model is misspecified. Model misspecification generally means that there is an omission of relevant variables or inclusion of irrelevant variables. We also realized that this model can be constructed to fit a dataset simply by constructing an appropriate matrix with response vector for some binomial regression model; we could do this from scratch for a single dataset. Another approach is to construct functions or quantities that make the data more specified and in a nice form. In addition to these, the Bradley-Terry model can address some specific questions, such as, what is the estimated value of the probability that u beats v? In illustrating these ideas, Drakos (1995) [15] was able to fit a Bradley-Terry model to the results for the eastern division of the American league for the 1987 baseball season. Conclusively, the Bradley-Terry model is used for estimating ranking probabilities of a finite number of items by pairwise comparison, which is based on a known order of preference. Some of the real-world applications of the Bradley-Terry model are:

1. Ranking documents based on relevance for any given query by information retrieval, Jong-June and Yongdai (2014) [7].

2. Quantification of the influence of statistical journals, Stigler(1994) [14].

3. Prediction of the results of FIFA 2010 South Africa World Cup, Hong, Jung and Lee (2010) [5].

4. Transmission/disequilibrium test in genetics, Sham and Curtis (1995) [13], amongst others.

2.2 Logistic Regression

Logistic regression is one of the regression methods that have been largely used for any data analysis involving the description of the relationship between a response variable and one or more explanatory variables. The logistic regression model has been the standard method of analysis for several years, and put to use in many fields for this type of case. Just like every other regression method, the goal of the analysis using logistic regression is to find the best fitting and most reasonable model to describe the relationship between a response and a set of predictor variables. It is also worthy of note that logistic regression has an outcome response that is dichotomous or binary. This largely differentiates logistic regression from linear regression and other regression methods, although it follows the same general principles as used in linear regression. Also, many distribution functions have been suggested for use in the analysis of categorical response variables. But the logistic distribution has been popular because it is very flexible and can be easily used, and it tends to give a meaningful interpretation as described by Hosmer and Lemeshow (1989) [6].

Before we discuss logistic regression, let us formally state the familiar linear regression model. Let Y and X be response and predictor variables respectively.A simple linear model is expressed as

$$E(Y|X) = \mu = \beta_0 + \beta_1 X,$$

with real-valued constants β_0 and β_1.

The simple linear model is a special case of the generalized linear model, which is given by

$$f\Big(E(Y|X)\Big) = \beta_0 + \beta_1 X \Rightarrow f(\mu) = \beta_0 + \beta_1 X,$$

where f is the link function.

Logistic regression is another special case of the generalized linear model, used for a categorical or binary outcome. We will consider the fixed effects case. So, let our predictor assume a fixed numerical value x, and let the random response variable Y be a categorical outcome "Yes/No" or 0/1. Suppose that the distribution of Y given x is Bernoulli(p), where p depends on the value of x. The probability function for Y is then given by

$$P(y) = p^y (1-p)^{1-y}. \tag{2.1}$$

This implies that $P(Y = 1) = p$ and $P(Y = 0) = 1 - p$.

By exploring its graph, we could see that a linear model is bad for this case because we get values of p that are less than 0 or greater than 1 against varying values of x. So we replace p with the odds quantity

$$\left(\frac{p}{1-p}\right).$$

This is to guarantee we always have a positive number. Taking the natural log gives,

$$\ln\left(\frac{p}{1-p}\right),$$

which is called the logit function. Now this guarantees that it no longer has to be a positive number, but can be any real number. Hence, the model for logistic regression is

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x. \tag{2.2}$$

By solving for p in equation 2.2, we end up with the logistic function

$$p = \frac{e^{(\beta_0+\beta_1 x)}}{1+e^{(\beta_0+\beta_1 x)}}. \tag{2.3}$$

By exploring the graph of the logistic function, we can see that $0 \leq p \leq 1$ which makes sense for this kind of regression. And β_0 determines the location, while β_1 determines direction (increasing/decreasing) and steepness.

2.3 Fitting the Logistic Regression Model with Zero Intercept

Let us suppose that we want to develop a model that can predict the outcome probabilities of an inter-league competition in which both leagues have n teams. For such a contest, each team will have a known rank within its league. Now, suppose that these ranks are given by i and j, and let p_{ij} denote the probability that the team with rank i defeats the team with rank j. Then, a model for the probabilities must satisfy some definite properties:

Property 1: $p_{ij} + p_{ji} = 1$ for all $i, j \in \{1, 2, \cdots, n\}$.

This must be true because, provided that ties are not allowed, the events that team i wins and that team j wins are complementary events.

Since the two competing teams are in different leagues, there exists the possibility that they could have the same rank such that $i = j$. It then follows from *Property 1* that:

Property 2: $p_{ii} = \dfrac{1}{2}$ for all $i \in \{1, 2, \cdots, n\}$.

We propose that such model is given by the logistic regression model with zero intercept, where $x = \text{rank}_A$ - rank_B. From 2.3, if $\beta_0 = 0$, then the probability that Team A beats Team B is given by

$$p(x) = \frac{e^{(\beta_1 x)}}{1 + e^{(\beta_1 x)}}. \tag{2.4}$$

Now suppose that Team A and Team B are equally ranked in their respective leagues. Then $x = \text{rank}_A$ - $\text{rank}_B = 0$, implying that

$$P(A \text{ beats } B) = p(0) = \frac{e^0}{1 + e^0} = \frac{1}{2}.$$

The graph will be symmetric about the point $(0, \frac{1}{2})$ and follows that when $x = 0, p(0) = \frac{1}{2}$. Thus this model satisfies Property 1.

Also, this model satisfies Property 2, since

$$\begin{aligned}
P(A \text{ beats } B) + P(B \text{ beats } A) &= p(x) + p(-x) \\
&= \frac{e^{(\beta_1 x)}}{1 + e^{(\beta_1 x)}} + \frac{e^{(-\beta_1 x)}}{1 + e^{(-\beta_1 x)}} \\
&= \frac{e^{(\beta_1 x)}}{1 + e^{(\beta_1 x)}} + \frac{e^{(-\beta_1 x)}}{(1 + e^{(-\beta_1 x)})} \cdot \frac{e^{(\beta_1 x)}}{e^{(\beta_1 x)}} \\
&= \frac{e^{(\beta_1 x)}}{1 + e^{(\beta_1 x)}} + \frac{1}{1 + e^{(\beta_1 x)}} \\
&= \frac{1 + e^{(\beta_1 x)}}{1 + e^{(\beta_1 x)}} = 1.
\end{aligned}$$

Since all the required properties are satisfied, the logistic regression model with zero intercept is a good choice to model the probability that Team A beats Team B in an inter-league tournament. We will next proceed to fitting this model.

We merge equation 2.1 and equation 2.4 into

$$P(y) = \left(\frac{e^{\beta_1 x}}{1 + e^{\beta_1 x}}\right)^y \cdot \left(\frac{1}{1 + e^{\beta_1 x}}\right)^{1-y} = \frac{e^{\beta_1 x y}}{1 + e^{\beta_1 x}}. \tag{2.5}$$

So given a set of data with a finite number of independent observations, to fit the logistic regression model with zero intercept in equation 2.5 to these data requires us to estimate the value of the unknown parameters β_1. Least squares is the most often used method for estimating the unknown parameters in linear regression. But this method cannot be applied to a model with dichotomous response variable because it forces the estimators to lose the desirable statistical properties, as demonstrated by Hosmer and Lemeshow (1989) [6]. So we invoke the method of maximum likelihood following the approach of Hosmer and Lemeshow (1989) [6]. In a more general sense, this method produces values for the unknown parameters that maximize the probability of getting the observed or given set of data. The likelihood function basically demonstrates the probability of the given data as a function of the unknown parameters and it is first constructed before applying the method of maximum likelihood. The values that maximize this function are the maximum likelihood estimators of the parameters. And these resulting estimators are described to agree most closely with the given data. So, we find these values from the logistic regression model with zero intercept.

Since we assumed to have independent observations, the likelihood function of equation 2.5 is:

$$L(\beta_1; Y_1, Y_2, \cdots, Y_n) = \prod_{i=1}^{n} \left(\frac{e^{\beta_1 x_i y_i}}{1 + e^{\beta_1 x_i}} \right) \tag{2.6}$$

By principle, the method of maximum likelihood estimation requires that we use an estimate of β_1 whose value maximizes equation 2.6. It is mathematically easier to work with the log function of equation 2.6 as argued by Hosmer and Lemeshow (1989) [6]. Taking the natural log of both sides of equation 2.6 produces

$$\begin{aligned}
\ln L(\beta_1; Y_i) &= \ln \prod_{i=1}^{n} \left(\frac{(e^{\beta_1 x_i})^{y_i}}{1 + e^{\beta_1 x_i}} \right), \\
&= \sum_{i=1}^{n} \ln(e^{\beta_1 x_i})^{y_i} - \sum_{i=1}^{n} \ln(1 + e^{\beta_1 x_i}), \\
&= \sum_{i=1}^{n} y_i(\beta_1 x_i) - \sum_{i=1}^{n} \ln(1 + e^{\beta_1 x_i}).
\end{aligned}$$

So, letting $l(\beta_1; Y_i) = \ln L(\beta_1; Y_i)$, we have

$$l(\beta_1; Y_i) = \beta_1 \sum_{i=1}^{n} x_i \cdot y_i - \sum_{i=1}^{n} \ln(1 + e^{\beta_1 x_i}). \tag{2.7}$$

Now, to find the value of β_1 that maximizes $L(\beta_1; Y_i)$ we differentiate equation 2.7 with respect to

β_1 and set the result to zero. Taking derivatives yields

$$\begin{aligned}\frac{\partial l(\beta_1;Y_i)}{\partial \beta_1} &= \sum_{i=1}^{n} x_i \cdot y_i - \sum_{i=1}^{n}\left(\frac{x_i e^{\beta_1 x_i}}{1+e^{\beta_1 x_i}}\right)\\ &= \sum_{i=1}^{n} x_i \cdot y_i - \sum_{i=1}^{n} x_i\left(\frac{e^{\beta_1 x_i}}{1+e^{\beta_1 x_i}}\right)\\ &= \sum_{i=1}^{n} x_i\left(y_i - \frac{e^{\beta_1 x_i}}{1+e^{\beta_1 x_i}}\right).\end{aligned}$$

Now, setting the derivative to zero gives

$$\sum_{i=1}^{n} x_i\left(y_i - \frac{e^{\beta_1 x_i}}{1+e^{\beta_1 x_i}}\right) = 0. \tag{2.8}$$

Unfortunately, the form of equation 2.8 makes it unfeasible to solve directly. So, we will need to apply some numerical approximation procedure to estimate its solution. The value of β_1 from equation 2.8 is denoted by $\hat{\beta}_1$ and it is called the maximum likelihood estimate of β_1. This represents the predicted or fitted value for the logistic regression model with zero intercept.

CHAPTER 3

NUMERICAL APPROXIMATION PROCEDURES

In this section, we study, implement, and compare the bisection method, Newton's method and improved-Newton's method as applied to get the estimate of β_1 in equation 2.8.

3.1 Bisection Method

Sauer (2012) [12] discussed that the bisection method uses the intuitive concept of bracketing the root, which is done first to ensure that a root exists for an equation. This method converges to only one root of the equation and does not give any clue whether there are additional solution(s) for the equation or how to find them, as shown by Levy (2010) [8], and by Gerald and Wheatley (2004) [3]. Hence, it is said to be linearly convergent.

Sauer (2012) [12] showed with an example that the bisection method is one of the *linearly convergent* methods, by observing the solutions from the point they begin to converge. That is, it was observed that the number of accurate decimal places increases by one for each iteration.

Definition 1. *Linear Convergence:*

An iterative method is said to satisfy linear convergence at rate R if

$$\lim_{j \to \infty} \frac{\Psi_{j+1}}{\Psi_j} = R < 1,$$

where Ψ_j is the error at iteration j.

Sarra (2018) [11] made mention of the fact that many other numerical approximation procedures also share significant characteristics with the bisection method. This idea is summarized in a corollary of the Intermediate Value Theorem(IVT). (See Appendix B.) The general algorithm for the bisection method developed by Sauer (2012) [12] can be used to find solutions of equations manually. Some applications of this method are also described by Sauer (2012) [12], and by Gerald and Wheatley (2004) [3] using MATLAB. So, starting with some closed interval $[u, v]$ of a function $f(x)$, after m number of iterations, the interval $[u_m, v_m]$ will have a length of

$$\frac{(v-u)}{2^m}.$$

The best estimate of a solution is obtained by selecting the midpoint;

$$z_c = \frac{u+v}{2}.$$

Hence, the error of the solution at the m^{th} iteration (Ψ_m) and function evaluation (Λ) proposed by Sauer (2012) [12] and by Sarra (2018) [11] now becomes

$$\Psi_e = |z_c - a| \leq \frac{v-u}{2^{m+1}} \text{ and } \Lambda = m+2,$$

where z_c is the value of the midpoint, and a is the solution of $f(x)$. To assess the efficiency of the bisection method, Sauer (2012) [12] suggested that we can measure how much accuracy is obtained for each Λ. And each Λ reduces the uncertainty in the solution by some number divisible by 2. So, we know a root is accurate within some k decimal places if the value of Ψ is below $0.5 * 10^{-k}$. It is worthy of note that the desired level of accuracy for the solution decides how many iterations that should be carried out when solving by hand. But when using computer programs, Sauer (2012) [12] suggested we define the *stopping criteria* or *tolerance (tol)*, which sets a limit to the number of possible correct digits. According to Sarra (2018) [11], we can achieve that by setting

$$m \geq \log_2 \Big(\frac{v-u}{tol}\Big).$$

This formula only exists for the bisection method; other methods require other criteria, and the importance of this *stopping criteria* is described by Sarra (2018) [11]. Levy (2010) [8] made us understand that the bisection method would always converge to a root and described how to identify how close we get to a solution after some number of iterations. Sarra (2018) [11] proposed the bisection algorithm in Python as shown in Figure 3.1.

Let us explore this method with the following example:

```
# import RootFinders as RF
from math import cos, log, ceil, fabs, copysign
# ------------------------------------------------------------------
def bisection(f,u,v,tol=1e-9):
        fu, fv = f(u), f(v)
        N = int(ceil(log((v-u)/tol)/log(2.0)))
             # the number of iterations
        b = []                  # an empty list
        for i in range(N):
                w = (v + u)/2.0
                b.append(w)
                fw = f(w)
                if fu*fw < 0:         # b* is in [u,w]
                        v, fv = w, fw
                else:                 # x* is in [w,v]
                        u, fu = w, fw
        b.append((u+v)/2)  # new midpoint is the best estimate
        return b
```

Figure 3.1: Python code for the bisection method.

Example 1. *Find the approximate root of* $xe^x - 1$.

From Figure 3.3, we get an approximate root of 0.567143290390959 *after* 37 *iterations in about* 0.03 *seconds.*

3.2 Newton's Method

This is also referred to as the Newton-Raphson method. Sauer (2012) [12] stated that it is known to converge more quickly than the bisection method and other linearly convergent methods. Sarra (2018) [11] argued that Newton's method can be extended easily to higher dimensions. These and many more are the reasons why it is popular and widely used. Sarra (2018) [11] mentioned that this method resulted from Newton's solution to Kepler's equation B.1 and argued that Newton's method is also a fixed point iteration, but a clever choice of the iteration function results in its quicker convergence. We understand from Levy (2010) [8] that Newton's method does not always converge, for example $f(x) = tan^{-1}x$. Gerald and Wheatley (2004) [3] argued that Newton's method might converge to a different solution or diverge completely if the initial guess is not quite close enough to the solution or root.

For any function $f(a)$, to find its root using Newton's method, we start with an initial guess a_0,

```python
# ----------------------------------------------------------------------
def newton(f,fp,b0,tol=1e-10,maxIt=50):
        iter = 0
        ba = []          # empty list
        ba.append(b0)    # add initial guess to list
        b = b0
        db = 100                    # increment
        fb = f(b)                   # residual
        fpb = fp(b)
        while abs(db)>tol and abs(fb)>tol and iter<=maxIt:
                db = -fb/fpb
                b += db
                fb = f(b)
                fpb = fp(b)
                ba.append(b)
                iter += 1
        return ba
# ----------------------------------------------------------------------
def newtonImproved(f,fp,b0,tol=1e-10,maxIt=50):     # 3rd order
        db, fb = 100, 100     # for the first iteration
        iter = 0
        ba = []                     # an empty list
        ba.append(b0)
        b = b0
        while fabs(db)>tol and fabs(fb)>tol and iter<=maxIt:
                fb = f(b)
                fpb = fp(b)
                db = -fb/fpb
                fb2 = f(b + db)
                      # an extra function evaluation
                db2 = -(fb+fb2)/fpb       # extra division
                b += db2
                ba.append(b)
                iter += 1
        return ba
```

Figure 3.2: Python code for two Newton approximation procedures.

```
import RootFinders as RF
import math
from pylab import *
def f(x): return x*math.exp(x) - 1
xStar = 0.567143                              # reference solution
a, b = -2, 2                      # search on interval [a,b]
tol = 1e-10                       # tolerance
x = RF.bisection(f,a,b,tol)
print('The approximate root is {:1.15f}'.format(x[-1]))
```

Figure 3.3: Finding the root of $xe^x - 1$ with the bisection method.

and then draw a line of tangent at a_0 to f, following the works of Sauer (2012) [12] and of Gerald and Wheatley (2004) [3]. So, invoking the equation of a tangent line formula, given the point $(a_0, f(a_0))$ and slope $(f'(a_0))$, we have

$$b - f(a_0) = f'(a_0)(a - a_0),$$

which is also a first order Taylor polynomial, according to Sarra (2018) [11]. Now, we set $b = 0$ to get the *x-intercept*, which is the point where the tangent line intercepts with the *x-axis*, to get:

$$-f(a_0) = f'(a_0)(a - a_0) \Rightarrow (a - a_0) = \frac{-f(a_0)}{f'(a_0)} \Rightarrow a = a_0 - \frac{f(a_0)}{f'(a_0)}$$

which is the algebraic formula for Newton's method. Repeating this procedure to solve for each $a_i, i \geq 1$ results in having the following iterative formula.

Letting a_0 be the initial guess,

$$a_{i+1} = a_i - \frac{f(a_i)}{f'(a_i)} \text{ for } i \geq 0. \tag{3.1}$$

Error at the m^{th} iteration is defined as: $e_m = a_m - c$, where c be a root of $f(a)$. Sauer (2012) [12] described with an example to show that Newton's method is one of the *quadratically convergent* methods, by observing the solutions from the point they begin to converge. That is, it was observed that the number of accurate decimal places in a_i doubles approximately on each iteration.

```
# run -t newtonExample2.py
import RootFinders as RF
import math
from pylab import *
def f(x): return x*math.exp(x) - 1
def fp(x): return x*math.exp(x) + math.exp(x)
xStar, x0 = 0.567143, 0.3
x = RF.newton(f,fp,x0)
print('The approximate root is {:1.15f}'.format(x[-1]))
```

Figure 3.4: Using Newton's method to find the root of $xe^x - 1$.

Definition 2. *Quadratic Convergence:*

An iterative method is said to satisfy quadratic convergence if

$$S = \lim_{j \to \infty} \frac{\Psi_{j+1}}{\Psi_j^2} < \infty,$$

where Ψ_j *is the error at iteration* j*.*

Further discussions on other properties the Newton's method exhibits such as *quadratically convergent*, *linearly convergent* and how Newton's method relates to other methods, are discussed by Sauer (2012) [12], Levy (2010) [8], Gerald and Wheatley (2004) [3], and Sarra (2018) [11]. Gerald and Wheatley (2004) [3] described the general algorithm for Newton's method. Sarra (2018) [11] developed Newton's method in Python as shown in Figure 3.2.

Let us explore this method with the following example:

Example 2. *Find the approximate root of* $xe^x - 1$*.*

From Figure 3.4, using an initial guess of 0.3*, we get an approximate root of* 0.567143290409784 *after* 6 *iterations in about* 0.02 *seconds.*

3.3 Improved Newton's Method

Yao (2014) [17] proposed a concept of accelerating an iterative method for solving algebraic equation by adding a simple term to the method having an order k convergence rate and increase the order of convergence to $(2k - 1)$. The order of convergence as defined by Yao (2014) [17], is simply a measure of how quickly an iterative method converges to the actual solution or root. This method was demonstrated by Yao (2014) [17] using an easy algebraic equation of 5^{th} order convergence but eventually used 4 function values on each iteration. When this method is applied to Newton's method with quadratic convergence, it can attain a cubic convergence, which gives us the improved Newton's method.

Definition 3. *Cubic Convergence:*

An iterative method is said to satisfy cubic convergence if

$$T = \lim_{j \to \infty} \frac{\Psi_{j+1}}{\Psi_j^3} < \infty,$$

where Ψ_j is the error at iteration j.

This method is valid for equation 2.8 since it is a system of equation that can be solved by Newton's method. We know that obtaining the root of $f(a)$ using the Newton's method requires we solve

$$f(a_i) + f'(a_i) \cdot e_i = 0. \tag{3.2}$$

This is *second order convergent* and follows from equation 3.1, where $e_i = a_{i+1} - a_i$, e_i being the error at each iteration i, $i \geq 0$. Hence, to find the root of $f(a)$ using the improved Newton's method, we add the term $f(a_i + e_i)$ to Newton's iterative formula in equation 3.2 and solve again to get

$$f(a_i + e_i) + f(a_i) + f'(a_i) \cdot \lambda_i = 0,$$

where $a_{i+1} = a_i + \lambda_i$. Adding the term improves the formula from being *second order convergent* to *third order convergent* as illustrated by Yao (2014) [17]. Sarra (2018) [11] developed the improved Newton's method in Python as shown in Figure 3.2.

Let us explore this method with the following example:

Example 3. *Find the approximate root of $xe^x - 1$.*

From Figure 3.5, using an initial guess of 0.3*, we get an approximate root of* 0.567143290409784 *after* 5 *iterations in about* 0.02 *seconds.*

We observe from the convergence plot in Figure 3.6 as coded in B.1 that the improved Newton's method is the fastest of the three methods, converging after 5 iterations and taking only about 0.01 seconds.

```
# run -t impNewtonExample2.py
import RootFinders as RF
import math
from pylab import *
tol, maxIt = 1e-16, 50
def f(x): return x*math.exp(x) - 1
def fp(x): return x*math.exp(x) + math.exp(x)
xStar, x0 = 0.567143, 0.3 # to avoid float division by zero
x = RF.newtonImproved(f,fp,x0,tol,maxIt)
print('The approximate root is {:1.15f}'.format(x[-1]))
```

Figure 3.5: Using improved Newton's method to find the root of $xe^x - 1$

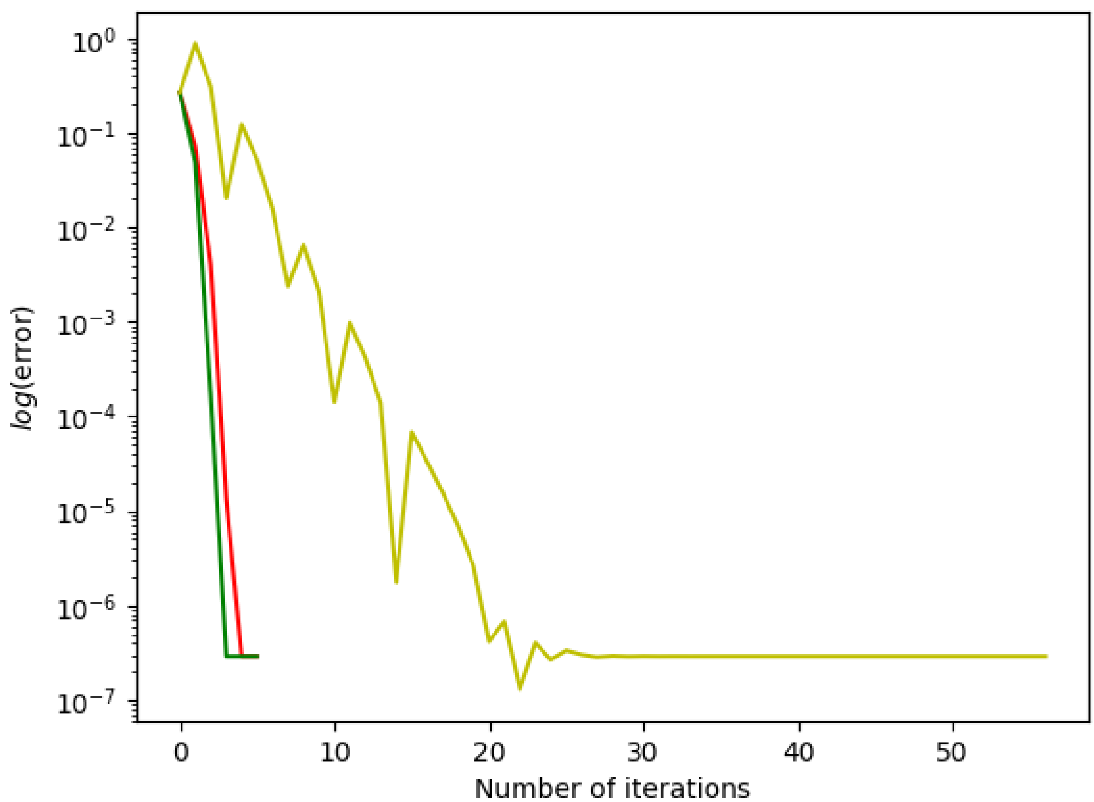


Figure 3.6: Convergence plot of the three approximation procedures.
Plot of log(error) against number of iterations for bisection (yellow), Newton's (red), and improved Newton's (green) method.

Other numerical approximation procedures such as fixed point iteration, secant method, Muller's method, etc., were discussed by Sauer (2012) [12], Levy (2010) [8], Gerald and Wheatley (2004) [3], and Sarra (2018) [11].

CHAPTER 4

SIMULATION STUDY

In this section we simulate data that fits equation 2.8 to test the goodness of the estimator, compare the three numerical approximation procedures - bisection, Newton's and improved Newton's methods and to examine if the estimator has a normal distribution. We fix the values of β_1 and x for varying values of y. Using Python, the code for equation 2.8 is shown in figure 4.1.

For this simulation, we consider an inter-league tournament involving six teams from two different leagues. The rank of each team within its league is given, so we calculate x, which is the difference in ranks for all 36 possible pairings as shown in Table 4.1.

Ranks	**1**	**2**	**3**	**4**	**5**	**6**
1	0	-1	-2	-3	-4	-5
2	1	0	-1	-2	-3	-4
3	2	1	0	-1	-2	-3
4	3	2	1	0	-1	-2
5	4	3	2	1	0	-1
6	5	4	3	2	1	0

Table 4.1: Difference in ranks in a 6-team inter-league competition

4.1 Simulation Process

With fixed value of β_1 and fixed values of x shown above, we simulate the values of y using Python. (See Figure 4.2.) Our choice of β_1 is dependent on the steepness of the logistic regression curve with zero intercept. (See Figure B.2.) The simulation process is as follows:

1. We set the value of $\beta_1 = 0.5$ and set the values of x to range from -5 to 5, which follows from the result in Table 4.1.

2. We evaluate the probability, p for each value of x using the logistic regression with zero intercept model.

3. Using the result from step 2 above, we generate random binomial values of y for $n = 1$, which guarantees a Bernoulli distribution.

```
# import function as F
import math
import numpy as np
x = input('Enter values of x: ')
x_i = x.split()          # convert input to an arrray
x_i = [float(a) for a in x_i]
y = input('Enter values of y: ')
y_i = y.split()
y_i = [float(a) for a in y_i]
CP = np.dot(x_i, y_i)    # dot product
def func(b):
        S = 0
        for a in x_i:
                exp = ((a*(math.exp((b)*a)))/(1+math.exp((b)*a)))
                S +=  exp
                fb = CP - S
        return fb
```

Figure 4.1: Python code for logistic regression model with zero intercept.
This code is for Equation 2.8.

4. We estimate the value of β_1 by computing equation 2.8 with the fixed values of x and values of y gotten from step 3.

5. We record the value of the estimate, number of iterations taken to converge, and system time, for each numerical approximation procedure.

6. We repeat steps $3 - 5$, 50 times.

7. We repeat steps $1 - 6$, for $\beta_1 = 1$.

4.2 Simulation Output

When $\beta_1 = 0.5$, we observe that the mean estimate is approximately 0.5 which justifies the goodness of the estimator. (See Table 5.1.) We also found that the average iteration required by bisection, Newton's and improved Newton's method to converge are 36, 6 and 5 respectively. (See Table 4.2.) And we found that the average system time it took for bisection, Newton's and improved Newton's method to run are 0.025, 0.012 and 0.009 seconds respectively.

When $\beta_1 = 1$, we observe that the mean estimate is approximately 1 which justifies the goodness of the estimator. (See Table 5.2.) We also found that the average number of iterations

```
#ipython /home/agboola/Documents/Python/sim_trial.py \
         > /home/agboola/Documents/Python/sim_data1.txt
import random
import math
import numpy as np
values = list(range(1,7,1)) #sequence of number from 1 to 6
values1 = list(range(1,7,1))
for a in values:
for b in values1:
x = b-a
print(x)
p = (math.exp (0.5*x))/(1+math.exp(0.5*x))
    # probability by log. reg. with zero intercept
y = np.random.binomial(1,p,1)
print (y)
```

Figure 4.2: Python code for simulating values of y.

required by bisection, Newton's and improved Newton's method to converge are 36, 7 and 6 respectively. (See Table 4.3.) And we found that the average system time it took for bisection, Newton's and improved Newton's method to run are 0.052, 0.035 and 0.026 seconds respectively.

Repetitions	Bisection	Iterations	System Time (secs)	Newton	Iterations	System Time (secs)	Imp. Newton	Iterations	System Time (secs)
1	1.0958479217	36	0.044	1.0958479217	7	0.024	1.0958479217	7	0.006
2	0.4006000966	36	0.028	0.4006000966	6	0.007	0.4006000966	5	0.007
3	0.3167543972	36	0.023	0.3167543972	5	0.013	0.3167543972	5	0.011
4	0.3711641013	36	0.040	0.3711641013	6	0.014	0.3711641013	5	0.010
5	0.4006000966	36	0.029	0.4006000966	6	0.011	0.4006000966	5	0.007
6	0.3167543972	36	0.023	0.3167543972	5	0.013	0.3167543972	5	0.011
7	0.3711641013	36	0.040	0.3711641013	6	0.014	0.3711641013	5	0.010
8	0.431843944	36	0.019	0.431843944	6	0.010	0.431843944	5	0.010
9	0.1566115235	36	0.021	0.1566115235	5	0.009	0.1566115235	4	0.008
10	0.267054218	36	0.024	0.267054218	5	0.010	0.267054218	5	0.008
11	0.5398301016	36	0.022	0.5398301016	6	0.015	0.5398301016	5	0.007
12	0.7393644407	36	0.032	0.7393644407	7	0.013	0.7393644407	6	0.012
13	0.2913944001	36	0.030	0.2913944001	5	0.009	0.2913944001	5	0.007
14	0.5821608908	36	0.021	0.5821608908	6	0.016	0.5821608908	6	0.010
15	0.431843944	36	0.019	0.431843944	6	0.010	0.431843944	5	0.010
16	0.6807404625	36	0.044	0.6807404625	7	0.010	0.6807404625	6	0.004
17	0.3167543972	36	0.023	0.3167543972	5	0.013	0.3167543972	5	0.011
18	0.4651994093	36	0.025	0.4651994093	6	0.011	0.4651994093	5	0.010
19	0.3711641013	36	0.040	0.3711641013	6	0.014	0.3711641013	5	0.010
20	0.5398301016	36	0.022	0.5398301016	6	0.015	0.5398301016	5	0.007
21	0.3167543972	36	0.023	0.3167543972	5	0.013	0.3167543972	5	0.011
22	0.4006000966	36	0.028	0.4006000966	6	0.007	0.4006000966	5	0.007
23	0.3711641013	36	0.040	0.3711641013	6	0.014	0.3711641013	5	0.010
24	1.4547919446	36	0.026	1.4547919446	8	0.019	1.4547919446	7	0.010
25	0.6807404625	36	0.023	0.6807404625	7	0.016	0.6807404625	6	0.010
26	0.5010396703	36	0.015	0.5010396703	6	0.010	0.5010396703	5	0.007
27	0.3432853248	36	0.020	0.3432853248	5	0.010	0.3432853248	5	0.007
28	0.6287949561	36	0.026	0.6287949561	6	0.010	0.6287949561	6	0.013
29	0.6807404625	36	0.023	0.6807404625	7	0.016	0.6807404625	6	0.010
30	0.9793127114	36	0.010	0.9793127114	7	0.004	0.9793127114	6	0.006
31	0.267054218	36	0.029	0.267054218	5	0.006	0.267054218	5	0.008
32	0.431843944	36	0.009	0.431843944	6	0.012	0.431843944	5	0.011
33	0.3167543972	36	0.023	0.3167543972	5	0.013	0.3167543972	5	0.011
34	0.1566115235	36	0.013	0.1566115235	5	0.007	0.1566115235	4	0.012
35	0.6807404625	36	0.023	0.6807404625	7	0.016	0.6807404625	6	0.010
36	0.3432853248	36	0.020	0.3432853248	5	0.010	0.3432853248	5	0.007
37	0.4651994093	36	0.025	0.4651994093	6	0.011	0.4651994093	5	0.010
38	0.3711641013	36	0.040	0.3711641013	6	0.014	0.3711641013	5	0.010
39	0.3432853248	36	0.020	0.3432853248	5	0.010	0.3432853248	5	0.007
40	0.3432853248	36	0.020	0.3432853248	5	0.010	0.3432853248	5	0.007
41	0.5398301016	36	0.022	0.5398301016	6	0.015	0.5398301016	5	0.007
42	0.1775167515	36	0.017	0.1775167515	5	0.014	0.1775167515	5	0.002
43	0.9793127114	36	0.010	0.9793127114	7	0.004	0.9793127114	6	0.006
44	0.4651994093	36	0.025	0.4651994093	6	0.011	0.4651994093	5	0.010
45	0.8851557161	36	0.013	0.8851557161	7	0.009	0.8851557161	6	0.017
46	0.7393644407	36	0.032	0.7393644407	7	0.013	0.7393644407	6	0.012
47	0.267054218	36	0.024	0.267054218	5	0.010	0.267054218	5	0.008
48	0.431843944	36	0.019	0.431843944	6	0.010	0.431843944	5	0.010
49	0.5010396703	36	0.015	0.5010396703	6	0.010	0.5010396703	5	0.007
50	0.3711641013	36	0.040	0.3711641013	6	0.014	0.3711641013	5	0.010
Average time			0.025			0.012			0.009
Average iterations		36			6			5	

Table 4.2: Table of values when $\beta_1 = 0.5$.
Showing the approximate estimate, number of iterations, and system time for bisection, Newton's and improved Newton's method for $\beta_1 = 0.5$.

Repetitions	Bisection	Iterations	System Time (secs)	Newton	Iterations	System Time (secs)	Imp. Newton	Iterations	System Time (secs)
1	1.2466254558	36	0.18	1.2466254558	7	0.04	1.2466254558	7	0.04
2	1.0958479217	36	0.03	1.0958479217	7	0.02	1.0958479217	7	0.02
3	0.8851557161	36	0.03	0.8851557161	7	0.06	0.8851557161	6	0.03
4	1.4547919446	36	0.05	1.4547919446	8	0.03	1.4547919446	7	0.03
5	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
6	1.7740830356	36	0.03	1.7740830356	8	0.03	1.7740830356	7	0.02
7	0.7393644407	36	0.06	0.7393644407	7	0.02	0.7393644407	6	0.03
8	0.8851557161	36	0.02	0.8851557161	7	0.02	0.8851557161	6	0.04
9	0.6807404625	36	0.03	0.6807404625	7	0.05	0.6807404625	6	0.01
10	1.0958479217	36	0.03	1.0958479217	7	0.02	1.0958479217	7	0.02
11	0.5398301016	36	0.02	0.5398301016	6	0.03	0.5398301016	5	0.02
12	2.3786204181	36	0.07	2.3786204181	9	0.04	2.3786204181	8	0.01
13	2.3786204181	36	0.07	2.3786204181	9	0.04	2.3786204181	8	0.01
14	0.8851557161	36	0.03	0.8851557161	7	0.06	0.8851557161	6	0.03
15	0.8065782598	36	0.04	0.8065782598	7	0.02	0.8065782598	6	0.02
16	1.2466254558	36	0.18	1.2466254558	7	0.04	1.2466254558	7	0.04
17	1.4547919446	36	0.05	1.4547919446	8	0.03	1.4547919446	7	0.03
18	0.7393644407	36	0.06	0.7393644407	7	0.02	0.7393644407	6	0.03
19	0.8065782598	36	0.04	0.8065782598	7	0.02	0.8065782598	6	0.02
20	2	36	0.06	2.3786204181	9	0.03	2.3786204181	8	0.01
21	1.2466254558	36	0.18	1.2466254558	7	0.04	1.2466254558	7	0.04
22	0.5398301016	36	0.02	0.5398301016	6	0.03	0.5398301016	5	0.02
23	1.0958479217	36	0.03	1.0958479217	7	0.02	1.0958479217	7	0.02
24	0.6807404625	36	0.03	0.6807404625	7	0.05	0.6807404625	6	0.01
25	0.8851557161	36	0.03	0.8851557161	7	0.06	0.8851557161	6	0.03
26	0.5398301016	36	0.02	0.5398301016	6	0.03	0.5398301016	5	0.02
27	1.0958479217	36	0.03	1.0958479217	7	0.02	1.0958479217	7	0.02
28	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
29	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
30	0.6287949561	36	0.07	0.6287949561	6	0.06	0.6287949561	6	0.03
31	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
32	0.7393644407	36	0.06	0.7393644407	7	0.02	0.7393644407	6	0.03
33	0.5821608908	36	0.02	0.5821608908	6	0.02	0.5821608908	6	0.02
34	0.7393644407	36	0.06	0.7393644407	7	0.02	0.7393644407	6	0.03
35	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
36	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
37	0.8851557161	36	0.03	0.8851557161	7	0.06	0.8851557161	6	0.03
38	0.5010396703	36	0.02	0.5010396703	6	0.02	0.5010396703	5	0.03
39	0.5010396703	36	0.02	0.5010396703	6	0.02	0.5010396703	5	0.03
40	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
41	1.7740830356	36	0.03	1.7740830356	8	0.03	1.7740830356	7	0.02
42	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
43	0.8851557161	36	0.03	0.8851557161	7	0.06	0.8851557161	6	0.03
44	1.2466254558	36	0.18	1.2466254558	7	0.04	1.2466254558	7	0.04
45	1.4547919446	36	0.05	1.4547919446	8	0.03	1.4547919446	7	0.03
46	1.0958479217	36	0.03	1.0958479217	7	0.02	1.0958479217	7	0.02
47	1.2466254558	36	0.18	1.2466254558	7	0.04	1.2466254558	7	0.04
48	1.0958479217	36	0.03	1.0958479217	7	0.02	1.0958479217	7	0.02
49	0.6807404625	36	0.03	0.6807404625	7	0.05	0.6807404625	6	0.01
50	0.9793127114	36	0.04	0.9793127114	7	0.04	0.9793127114	6	0.03
Average time			0.052			0.035			0.026
Average iterations		36			7			6	

Table 4.3: Table of values when $\beta_1 = 1$.
Showing the approximate estimate, number of iterations, and system time for bisection, Newton's and improved Newton's method for $\beta_1 = 1$.

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this section we summarize our results and identify some limitations that were experienced during this study which lead to future work.

5.1 Conclusion

In statistics, a new model or approach for estimation is considered valid after justifying the goodness of the estimator and goodness of the model. This study was able to justify the goodness of the estimator, which followed from the results in Chapter 4. (See Tables 5.1 and 5.2.) We also used the simulation process to examine whether the estimator has a normal distribution. But, when $\beta_1 = 0.5$, we found evidence that the estimator is not normal using the Kolmogorov-Smirnov test. (See Table 5.1.) This was also complemented by the Q-Q plot and histogram. (See Table 5.1.) We also observed that the improved Newton's method is the fastest of the three numerical approximation procedures, taking an average of 5 iterations and approximately 0.009 seconds, as shown in Table 4.2. Also, when $\beta_1 = 1$, we observed that the estimator is not normal using the Kolmogorov-Smirnov test. (See Table 5.2.) This was also complemented by the Q-Q plot and histogram. (See Table 5.2.) And similarly, we observed that the improved Newton's method is the fastest of the three numerical approximation procedures, taking an average of 6 iterations and approximately 0.026 seconds, as shown in Table 4.3.

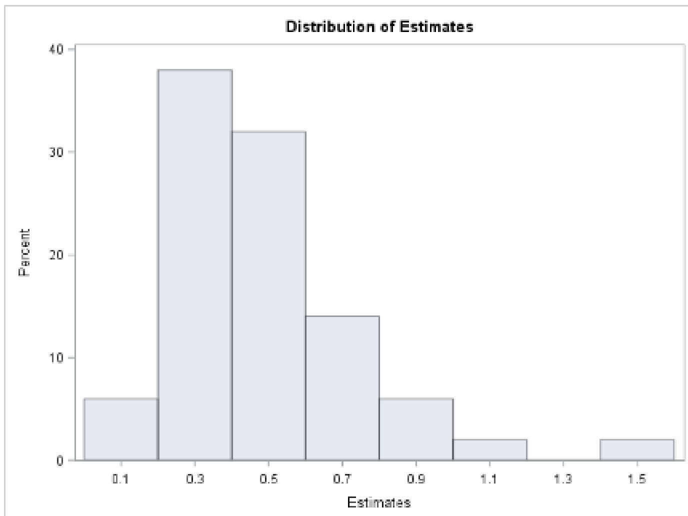
It should be noted that we would not be using the Mean Square Error for logistic regression as it is being used for linear regression. Michael A. Nielsen (2015) [10] gave a thorough mathematical explanation in his book, but it was summarized from a Bayesian perspective in [2] that it is "because our prediction function is non-linear (due to sigmoid transform). Squaring this prediction as we do in MSE results in a non-convex function with many local minimums."

5.2 Future work

1. **Implementation of the model:** This study focused on proposing logistic regression with zero intercept as a new approach to estimating preference probability based on ranks. As mentioned earlier, we are yet to determine the goodness of the model as applied to real-life

Moments			
N	50	Sum Weights	50
Mean	0.49041213	Sum Observations	24.5206063
Std Deviation	0.25180839	Variance	0.06340747
Skewness	1.70134476	Kurtosis	3.70697733
Uncorrected SS	15.1321684	Corrected SS	3.10696582
Coeff Variation	51.3462816	Std Error Mean	0.03561108

Tests for Normality				
Test	Statistic		p Value	
Shapiro-Wilk	W	0.851746	Pr < W	<0.0001
Kolmogorov-Smirnov	D	0.179878	Pr > D	<0.0100
Cramer-von Mises	W-Sq	0.407574	Pr > W-Sq	<0.0050
Anderson-Darling	A-Sq	2.269327	Pr > A-Sq	<0.0050



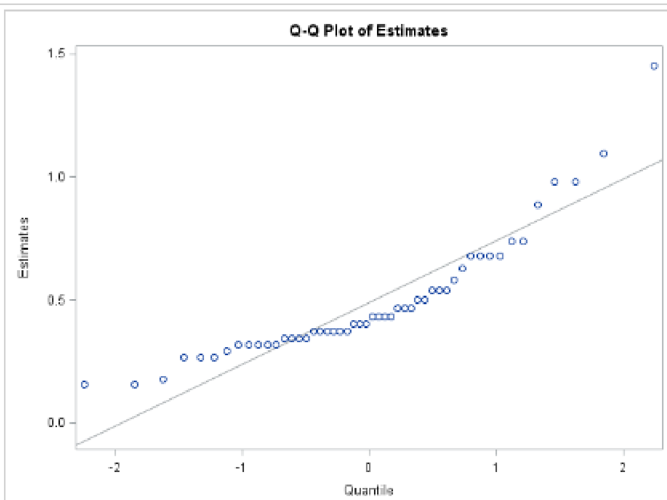
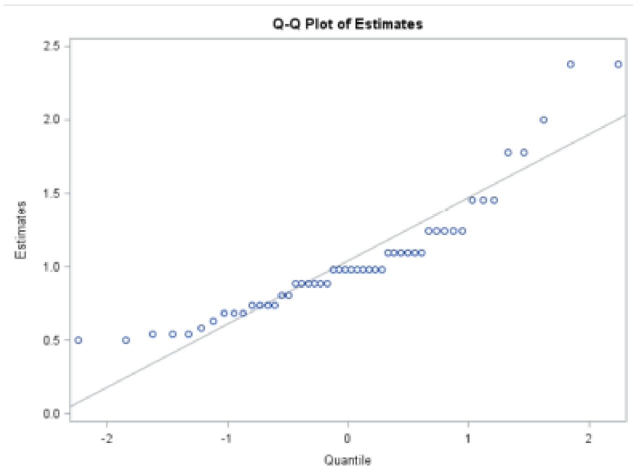


Table 5.1: Table of measures and graphs for estimates when $\beta_1 = 0.5$.
Showing measures of moments, tests for normality and graphs of Q-Q plot and histogram of estimates when $\beta_1 = 0.5$.

Moments			
N	50	Sum Weights	50
Mean	1.04096215	Sum Observations	52.0481074
Std Deviation	0.43038522	Variance	0.18523144
Skewness	1.48125177	Kurtosis	2.4880909
Uncorrected SS	63.25645	Corrected SS	9.07634033
Coeff Variation	41.344944	Std Error Mean	0.06086566

Tests for Normality				
Test	Statistic		p Value	
Shapiro-Wilk	W	0.865274	Pr < W	<0.0001
Kolmogorov-Smirnov	D	0.189262	Pr > D	<0.0100
Cramer-von Mises	W-Sq	0.31372	Pr > W-Sq	<0.0050
Anderson-Darling	A-Sq	1.914957	Pr > A-Sq	<0.0050



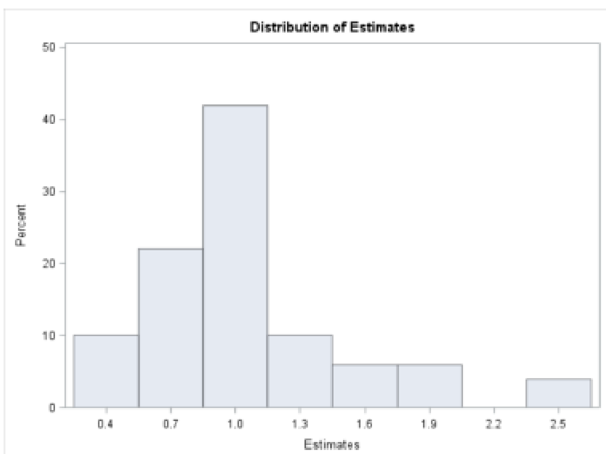


Table 5.2: Table of measures and graphs for estimates when $\beta_1 = 1$.
Showing measures of moments, tests for normality and graphs of Q-Q plot and histogram of estimates when $\beta_1 = 1$.

data. We tried simulating data, hoping it could mimic a real-life scenario and give us a true representation of the model's effectiveness but we have no idea whether reality would conform to our model, and we did not have enough time to dig for real data and implement. This presents an intriguing area for future research.

2. **Comparing varying ranks with preference probability:** The preference probability for equal ranks is most obvious in this study, which is one of the properties our statistical model satisfies and one of the reasons why the model is considered in this study. But it does not mention or compare the probability when we have equal or varying values of x with varying ranks. For instance, $P(\text{Rank1} > \text{Rank4})$ and $P(\text{Rank3} > \text{Rank6})$ or vice-versa. This is open to further work to see what interesting things happen in the above cases.

3. **Structure of Python module for the model:** The Python code for equation 2.8 only permits the user to input corresponding values of x and y. Though it supports the "copy" and "paste" option, but formatting the data before copying could be painful when dealing with very large files. This is open to further work if we need to compare numerical approximation procedures, since we already have statistical packages and software that estimates logistic regression using Newton's method, as found in SAS.

4. **Unequal number of teams:** We assumed we should have equal numbers of teams for the inter-league competition, which might not always be the case in reality. Hence, further work could consider how to format the model to suit this special case.

5. **Additional predictors:** We could explore other factors that can influence the preference probability asides from known ranking, such as number of wins prior to the game. We can achieve that by introducing additional predictors to our model.

6. **Possibility of ties:** We could also explore ways that allow the possibility of ties.

REFERENCES

[1] R. A. Bradley and M. E. Terry, *Rank analysis of incomplete block designs: I. The method of paired comparisons*, Biometrika Trust **39** (1952), 324–345.

[2] Brendan Fortuner, *Machine learning cheatsheet: Logistic regression*, revision 016f83bb, online book, 2017.

[3] C. F. Gerald and P. O. Wheatley, *Applied Numerical Analysis*, Pearson Education, 2004.

[4] D. Holland and R. Wessells, *Predicting consumer preferences for fresh salmon: The influence of safety inspection and production method attributes*, Agricultural and Resource Economics Review **27** (1998), no. 1, 1–14.

[5] C. Hong, M. Jung, and J. Lee, *Prediction model analysis of 2010 South Africa world cup*, Journal of the Korean Data and Information Science Society **21** (2010), no. 6, 1137–1146.

[6] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression*, John Wiley and Sons, 1989.

[7] J. Jong-June and K. Yongdai, *Revisiting the Bradley-Terry model and its application to information retrieval*, Journal of the Korean Data and Information Science Society **24** (2013), no. 5, 1089–1099.

[8] D. Levy, *Introduction to Numerical Analysis*, lecture notes, 2010.

[9] V. S. Y. Lo, J. Bacon-Shone, and K. Busche, *The application of ranking probability models to racetrack betting*, Management Science **41** (1995), no. 6, 1048–1059.

[10] M. A. Nielsen, *Neural Networks and Deep Learning*, online book, 2015.

[11] S. Sarra, *Applied Numerical Analysis*, lecture notes, 2018.

[12] T. Sauer, *Numerical Analysis*, Pearson Education, 2012.

[13] P.C. Sham and D. Curtis, *An extended transmission/disequilibrium test (tdt) for multiallele marker loci*, Ann Hum Genet **59** (1995), 323–336.

[14] S. Stigler, *Citation patterns in the journals of statistics and probability*, Statist. Sci. **9** (1994), 94–108.

[15] Luke Tierney, *Generalized Linear Models in Lisp-Stat*, lecture notes, 1997.

[16] K. A. Willoughby, *Winning games in canadian football: A logistic regression analysis*, College Mathematics Journal **33** (2002), 215–220.

[17] J. Yao, *An easy method to accelerate an iterative algebraic equation solver*, Journal of Computational Physics **267** (2014), 139–145.

APPENDIX A

LETTER FROM INSTITUTIONAL RESEARCH BOARD

MARSHALL UNIVERSITY®

www.marshall.edu

Office of Research Integrity

March 13, 2018

Oluwagbenga David Agboola
2520 4th Avenue
Huntington, WV 25703

Dear Mr. Agboola:

This letter is in response to the submitted thesis abstract entitled *"Preference Probability Based on Ranks – A New Approach Using Logistic Regression with Zero Intercept."* After assessing the abstract, it has been deemed not to be human subject research and therefore exempt from oversight of the Marshall University Institutional Review Board (IRB). The Code of Federal Regulations (45CFR46) has set forth the criteria utilized in making this determination. Since the information in this study does not involve human subjects as defined in the above referenced instruction, it is not considered human subject research. If there are any changes to the abstract you provided then you would need to resubmit that information to the Office of Research Integrity for review and a determination.

I appreciate your willingness to submit the abstract for determination. Please feel free to contact the Office of Research Integrity if you have any questions regarding future protocols that may require IRB review.

Sincerely,

Bruce F. Day, ThD, CIP
Director

WE ARE... MARSHALL.

One John Marshall Drive • Huntington, West Virginia 25755 • Tel 304/696-4303
A State University of West Virginia • An Affirmative Action/Equal Opportunity Employer

APPENDIX B

RELEVANT RESULTS

The concept of the bisection method is rooted in the Intermediate Value Theorem (IVT) from Calculus 1. The bracketing the root idea is summarized in the corollary of the IVT.

Theorem 1 (Intermediate Value Theorem)**.** *Let f be a continuous function on a closed interval $[u, v]$. If M is some number between $f(u)$ and $f(v)$, then there exists a number n in (u, v) such that $f(n) = M$.*

Corollary 1.1. *Let f be a continuous function on a closed interval $[u, v]$ so that it satisfies $f(u)f(v) < 0$. Then there exists a number n satisfying $u < n < v$ and $f(n) = 0$ which implies that f has a root between u and v.*

The Kepler's equation is defined as

$$\theta - \gamma \sin(\theta) - k = 0, \tag{B.1}$$

where θ is the eccentric anomaly, k is the mean anomaly, and γ is the eccentricity.

```
# run Example.py
from pylab import *
import math
import RootFinders as RF
tol, maxIt = 1e-16, 50
a, b = -2, 2.6                    # search on interval [a,b]
def f(x): return x*math.exp(x) - 1
def fp(x): return x*math.exp(x) + math.exp(x)    # f'(x)

xStar = 0.567143          # exact solution
x0 = 0.3                  # initial guess

x = RF.bisection(f,a,b,tol)
x = array(x)
eB = abs(xStar - x)

x = RF.newton(f,fp,x0,tol,maxIt)
x = array(x)
e = abs(xStar - x)

x = RF.newtonImproved(f,fp,x0,tol,maxIt)
x = array(x)
eN = abs(xStar - x)

rho = (log10(eB[-1]) - log10(eB[-2]))/(log10(eB[-2]) - log10(eB[-3]))
print('Bisection {:1.15f}'.format(x[-1]))

rho = (log10(e[-2]) - log10(e[-3]))/(log10(e[-3]) - log10(e[-4]))
print('Newton {:1.8f}'.format(x[-1]))

rho = (log10(eN[-3]) - log10(eN[-4]))/(log10(eN[-4]) - log10(eN[-5]))
print('improved Newton {:1.8f}'.format(x[-1]))

semilogy(range(0,len(e)),e,'r',range(0,len(eN)),eN,'g',
         range(0,len(eB)),eB,'y')
ylabel('$log$(error)'); xlabel('Number of iterations')

show()
```

Figure B.1: Python code for convergence plot.
Finding the root of $xe^x - 1$ and showing the convergence plot of bisection, Newton and improved Newton's method

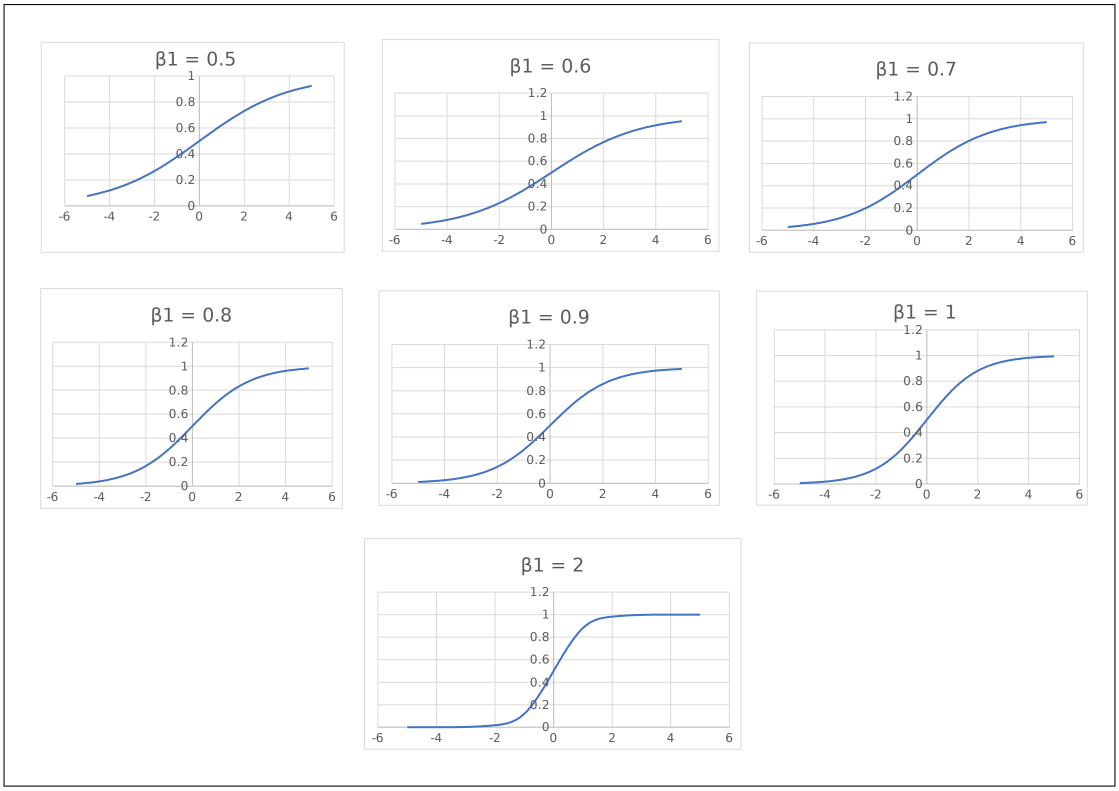


Figure B.2: Varying graphs of logistic regression model with zero intercept.
Graph of varying values of β_1 to compare steepness of logistic regression model with zero intercept

VITA

Agboola Oluwagbenga David

Born March 7, 1992 in Ado-Ekiti, Nigeria

Education

- Master of Arts. Marshall University, May 2018. Thesis Advisor: Dr. Laura Adkins.
- Bachelor of Science. Federal University of Agriculture Abeokuta, Nigeria, August 2012