FINAL PROJECT SUBMISSION

MSIS 5223: Programming for Data Science

Advisor: Dr. Bryan Hammer

PROJECT TITLE

Analysis and Prediction of Labour Condition Application (LCA) for H1B immigration visas

GROUP MEMBERS

Ravali Musty (A20101635)
Reno Rajan Christy (A20082578)
Shwetha Krishnan (A20103580)
Viswesh Muralitharan (A20095678)

Table of Contents

A. EXECUTIVE SUMMARY

An H1B visa is a temporary work visa for foreign worker with a specialty occupation to work in the United States. The H1B visa requires employers to file Labour Condition Application (LCA) that needs to be approved by the Department of Labour to sustain foreign workers. The current political environment suggests the heavy misuse of filing LCA's for variety of foreign workers with minimum potential that may not be reciprocated for the occupational purposes. We aim to discover the major factors contributing to successful LCA acceptances, so that companies may use this to sustain foreign workers in the USA under the H1B immigration policies. We also aim to identify if there exists any disparity based on demographic, geographic, occupational, remuneration or educational level of the foreign workers that need LCA acceptances. The motive of this study to identify the relevant industries and their existing job roles which require specialized skill sets that are only fulfilled by hiring foreign workers.

B. PROJECT SCHEDULE

There has been no change in project schedule since deliverable 1. The project has been on track as planned with time and efforts spent more on results documentation. As planned the project report was ready by 28th April 2018. Resources have worked between 1-3 hours on the deliverables allocated to them. Maximum time was spent on Data preparation, cleaning and consolidation with the whole team working on it for 9 hours straight for 4 days. This initial effort spent has bored fruits for a wonderful analysis and reporting that can be seen.

This project's biggest challenge has been the size of the dataset with approximately 2.5 million original data-set.

C. GANTT CHART

The below figures (C.1) and (C.2) show the project is on track as per Gantt chart

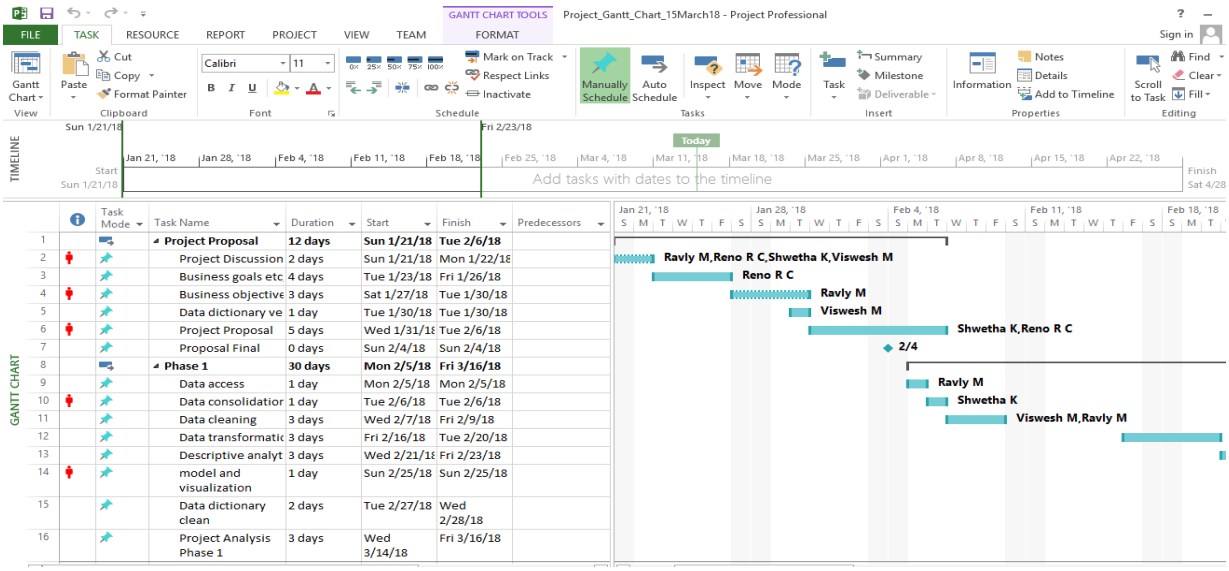


Figure (C.1)

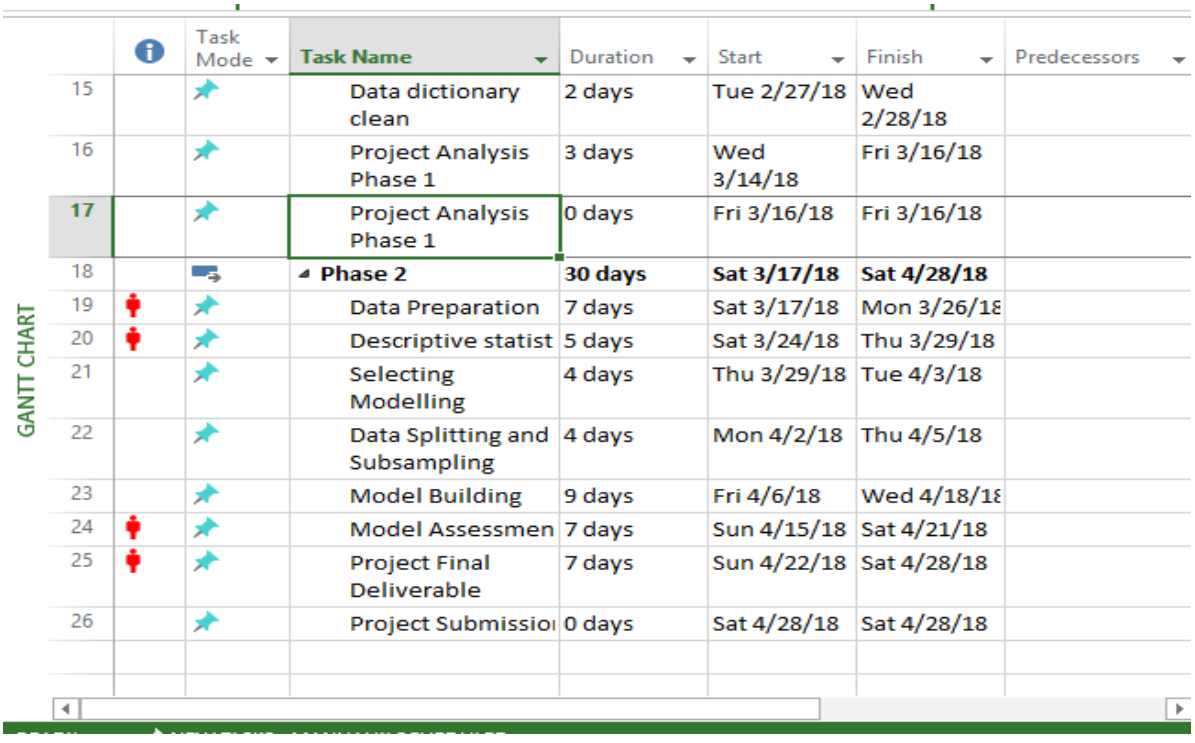


Figure (C.2)

D. WORK ALLOCATION MATRIX

The below Table (D.1) shows the resource assignment for the project.

Work Name	Work Description	Work Owner/s
Project problem	Identify a project analysis issue	Viswesh,Reno,Ravali,Shwetha
Data Collection	Obtaining Data from OFLC website	Shwetha,Ravali
Data Walkthrough	Studying Data variables	Viswesh,Reno,Ravali,Shwetha
Data Variable selection	Evaluating which variables would be needed for the analysis of the project.	Viswesh,Reno,Ravali,Shwetha
Data Cleaning	Filling in the missing values, renaming columns, Evaluating data quality	Viswesh,Ravali
Data Consolidation	Combining data files of all 5 years as one file	Shwetha,Reno
Data Transformation	Normalized the data and created new variables	Viswesh,Reno
Data Reduction and Sample selection	Selecting relevant data to analyze so data reduction from 5 years to 3 years	Viswesh,Reno,Ravali,Shwetha
Explanatory Analysis	Generating summary statistics of data	Shwetha,Ravali
Visual reports	Creating reports for visual understanding	Shwetha,Ravali
Project Deliverable 1 Report	Creating the report document	Viswesh,Reno
Identify Models	Understanding the different models to be built	Viswesh,Reno,Ravali,Shwetha
Validation of Model Results	Validating and understanding the model results	Viswesh,Reno,Ravali,Shwetha
Model Selection	Selecting the best model through comparisons	Viswesh,Reno,Ravali,Shwetha
Project Deliverable 2 Report	Creating the report document	Shwetha,Ravali

Table (D.1)

E. STATEMENT OF SCOPE

The purpose of the project is to unearth the major factors contributing to the successful LCA filing for foreigner workers under for H1B category.

F. PROJECT OBJECTIVES

i. Unveiling Top Jobs in market for H1B based on demand classification.
ii. Top Companies filing for LCA petitions
iii. Company wise analysis of filed LCA status
iv. Factors contributing to successful LCA petition filing.
v. Predict whether LCA petition will be Approved based on factored variables.
vi. Forecast the Average Wage Offered for the top Job category in H1B LCA Applicants for the period Oct 2017 to May 2018
vii. Forecast the number of H1B LCA applications for the period Oct 2017 to March 2018; Compare same with actual Data.

G. DATA ACCESS

The Office of Foreign Labour Certification (OFLC) generates program data, including data about H1-B visas. The disclosure data updated annually and is available online. Original site: Disclosure Data-> LCA Program data
https://www.foreignlaborcert.doleta.gov/performancedata.cfm
For our project work we downloaded the individual files from years 2014 till 2018 and placed it the google drive:
https://drive.google.com/drive/folders/1oBVSM_uuIAtCD0XIzOa1J5zAGNwSBzW6

Population

In completing its obligation regarding the preparing of work accreditation and work authentication applications program data, OFLC, also provides Disclosure data essential both for internal assessment of program effectiveness and for providing the Department's external stakeholders with useful information about the immigration programs. We have collected the

following as our population and samples from the population. The Table (G.1) below shows the actual dataset was nearly 2.5 million records.

Year	Number of total LCA records filed
2014	519505
2015	618805
2016	647853
2017	624651
2018	94623
Total population size	2505437

Table (G.1)

Working Sample

Working through the population data we realized, there is more data to analyse in the years 2015,2016 and 2017 than in the years 2014 and 2018. This was achieved after noting there is no biases if the omitted years were not to be considered as part of the samples to analyse. Table (G.2) below shows the sample size was now at 1.8 million

Year	Number of total LCA records filed
2015	618805
2016	647853
2017	624651
Total population size	1891309

Table (G.2)

Predictor Variables

The predictor variables to be used in logistic regression analysis are WAGE_RATE_OF_PAY_TO, WAGE_RATE_OF_PAY_FROM, H.1B_DEPENDENT, PREVAILING_WAGE and Days

Target Variable

The target variable for our analysis is CASE_STATUS

H. DATA DICTIONARY

The below dictionary table (H.1) is directly sourced from the OFLC website.

FIELD NAMES	DESCRIPTION	DATATYPE
CASE_NUMBER	Unique identifier assigned to each application submitted for processing	NUMERIC
CASE_STATUS	Status of LCA application which includes: "CERTIFIED", "DENIED", "CERTIFIED WITHDRAWN" and "WITHDRAWN"	FACTOR
CASE_SUBMITTED	Date and time the application was submitted.	DATE

FIELD NAMES	DESCRIPTION	DATATYPE
DECISION_DATE	Date on which the last significant event or decision was recorded by the Chicago National Processing Center.	DATE
VISA_CLASS	Indicates the type of temporary application submitted for processing. Values include H-1B, E-3 Australian, H-1B1 Chile, and H-1B1 Singapore.	FACTOR
EMPLOYMENT_START_DATE	Beginning date of employment.	DATE
EMPLOYMENT_END_DATE	Ending date of employment.	DATE
EMPLOYER_NAME	Name of employer submitting labor condition application.	STRING
EMPLOYER_BUSINESS_DBA	Contact information of the Employer requesting	STRING
EMPLOYER_ADDRESS		STRING
EMPLOYER_CITY		STRING
EMPLOYER_STATE		STRING
EMPLOYER_POSTAL_CODE		STRING

FIELD NAMES	DESCRIPTION	DATATYPE
EMPLOYER_COUNTRY	temporary labor certification.	STRING
EMPLOYER_PROVINCE		STRING
EMPLOYER_PHONE		NUMERIC
EMPLOYER_PHONE_EXT		NUMERIC
AGENT_REPRESENTING_EMPLOYER	Y = Employer is represented by an Agent or Attorney; N = Employer is not represented by an Agent or Attorney.	FACTOR
AGENT_ATTORNEY_NAME	Name of Agent or Attorney filing an H-1B application on behalf of the employer.	STRING
AGENT_ATTORNEY_CITY	City information for the Agent or Attorney filing an H-1B application on behalf of the employer.	STRING
AGENT_ATTORNEY_STATE	State information for the Agent or Attorney filing an H-1B application on behalf of the employer.	STRING
JOB_TITLE	Title of the job.	STRING

FIELD NAMES	DESCRIPTION	DATATYPE
SOC_CODE	Occupational code associated with the job being requested for temporary labor condition, as classified by the Standard Occupational Classification (SOC) System.	STRING
SOC_NAME	Occupational name associated with the SOC_CODE.	STRING
NAICS_CODE	Industry code associated with the employer requesting permanent labor condition, as classified by the North American Industrial Classification System (NAICS).	NUMERIC
TOTAL_WORKERS	Total number of foreign workers requested by the Employer(s).	NUMERIC

FIELD NAMES	DESCRIPTION	DATATYPE
NEW_EMPLOYMENT	Indicates requested worker(s) will begin employment for new employer as defined by USCIS I-29.	FACTOR
CONTINUED_EMPLOYMENT	Indicates requested worker(s) will be continuing employment with same employer, as defined by USCIS I-29.	FACTOR
CHANGE_PREVIOUS_EMPLOYMENT	Indicates requested worker(s) will be continuing employment with same employer without material change to job duties, as defined by USCIS I-29.	FACTOR
NEW_CONCURRENT_EMPLOYMENT	Indicates requested worker(s) will begin employment with additional employer, as defined by USCIS I-29.	FACTOR

FIELD NAMES	DESCRIPTION	DATATYPE
CHANGE_EMPLOYER	Indicates requested worker(s) will begin employment for new employer,	FACTOR
AMENDED_PETITION	Indicates requested worker(s) will be continuing employment with same employer with material change to job duties, as defined by USCIS I-29.	FACTOR
FULL_TIME_POSITION	Y = Full Time Position; N = Part Time Position.	FACTOR
PREVAILING_WAGE	Prevailing Wage for the job being requested for temporary labor condition.	NUMERIC
PW_UNIT_OF_PAY	Unit of Pay. Valid values include “Daily (DAI),” “Hourly (HR),” “Bi-weekly (BI),” “Weekly (WK),” “Monthly (MTH),” and “Yearly (YR)”	FACTOR

FIELD NAMES	DESCRIPTION	DATATYPE
	Variables include "I", "II", "III", "IV" or "N/A."	
PW_SOURCE	Variables include "OES", "CBA", "DBA", "SCA" or "Other".	FACTOR
PW_SOURCE_YEAR	Year the Prevailing Wage Source was Issued.	DATETIME
PW_SOURCE_OTHER	If "Other Wage Source", provide the source of wage.	STRING
WAGE_RATE_OF_PAY_FROM	Employer's proposed wage rate.	NUMERIC
WAGE_RATE_OF_PAY_TO	Maximum proposed wage rate	NUMERIC
WAGE_UNIT_OF_PAY	Unit of pay. Valid values include "Hour", "Week", "Bi-Weekly", "Month", or "Year"	STRING
H-1B_DEPENDENT	Y = Employer is H-1B Dependent; N = Employer is not H-1B Dependent.	FACTOR

FIELD NAMES	DESCRIPTION	DATATYPE
WILLFUL_VIOLATOR	Y = Employer has been previously found to be a Willful Violator; N = Employer has not been considered a Willful Violator.	FACTOR
SUPPORT_H1B	Y = Employer will use the temporary labor condition application only to support H-1B petitions or extensions of status of exempt H-1B worker(s); N = Employer will not use the temporary labor condition application to support H-1B petitions or extensions of status for exempt H-1B worker(s);	FACTOR
LABOR_CON_AGREE	Y = Employer agrees to the	FACTOR

FIELD NAMES	DESCRIPTION	DATATYPE
	responses to the Labor Condition Statements as in the subsection; N = Employer does not agree to the responses to the Labor Conditions Statements in the subsection.	
PUBLIC_DISCLOSURE_LOCATION	Variables include "Place of Business" or "Place of Employment."	FACTOR
WORKSITE_CITY	City information of the foreign worker's intended area of employment.	STRING
WORKSITE_COUNTY	County information of the foreign worker's intended area of employment.	STRING
WORKSITE_STATE	State information of the foreign worker's intended	FACTOR

FIELD NAMES	DESCRIPTION	DATATYPE
	area of employment.	
WORKSITE_POSTAL_CODE	Zip Code information of the foreign worker's intended area of employment.	NUMBER
ORIGINAL_CERT_DATE	Original Certification Date for a Certified Withdrawn application	DATE

Table (H.1)

I. DATA PREPARATION

The original dataset had more 2.5 million data records. After analysis it deemed fit to work with select years of data (2015 till 2017). Various data cleansing and transformations were done to the sample dataset of nearly 1.8 million data records to obtain our conclusions.

Data Consolidation

i. Post the data reduction we merged the 3 year files into a single consolidated .csv file and have placed it in the location: https://drive.google.com/drive/folders/13A-Ne9KBncMpRKkOMALPmCCb9_bgKtFi

ii. On the consolidated .csv file we now have 39 columns and nearly 1.7 million data records.

iii. Out of the 39 columns we analysed and obtained the below 15 columns to be used for our analysis. The reasons to select the 15 columns has been mentioned in the table (I.1) below after the screenshot. Please note our dataset has all the 39 columns for our

analysis we used a subset of the dataset to work ahead as seen in figures (I.1) and (I.2)

```
> ###Total Number of Rows and Columns After Subsetting
> nrow(h1b_filtered)
[1] 1733815
> ncol(h1b_filtered)
[1] 15
```

Figure (I.1)

```
> summary(h1b_filtered)
 CASE_NUMBER                    CASE_STATUS     CASE_SUBMITTED       EMPLOYMENT_START_DATE EMPLOYMENT_END_DATE  EMPLOYER_NAME
 Length:128037      CERTIFIED          :110807   Min.   :0001-01-20   Min.   :2011-01-31    Min.   :2013-10-16   Length:128037
 Class :character   CERTIFIED-WITHDRAWN: 10312   1st Qu.:0003-03-20   1st Qu.:2015-08-14    1st Qu.:2018-08-03   Class :character
 Mode  :character   DENIED             :  2115   Median :0003-12-20   Median :2016-05-09    Median :2019-03-15   Mode  :character
                    WITHDRAWN          :  4803   Mean   :0005-08-26   Mean   :2016-04-18    Mean   :2019-03-16
                    I-200-16081-868857 :     0   3rd Qu.:0008-02-20   3rd Qu.:2017-01-04    3rd Qu.:2019-11-21
                                                 Max.   :0012-12-20   Max.   :2018-03-14    Max.   :2021-03-14

 EMPLOYER_STATE                        JOB_TITLE         SOC_CODE                                        SOC_NAME     PREVAILING_WAGE
 Length:128037      SOFTWARE ENGINEER       :  6249   15-1132:26549   SOFTWARE DEVELOPERS; APPLICATIONS    :17473   Min.   : 15000
 Class :character   DEVELOPER               :  2850   15-1121:15664   COMPUTER SYSTEMS ANALYSTS            :15635   1st Qu.: 62130
 Mode  :character   PROGRAMMER ANALYST      :  2657   15-1131:13885   COMPUTER PROGRAMMERS                 :13864   Median : 74818
                    SENIOR SOFTWARE ENGINEER:  2648   15-1199:10154   SOFTWARE DEVELOPERS, APPLICATIONS    : 9067   Mean   : 79829
                    SOFTWARE DEVELOPER      :  2318   15-1133: 9197   COMPUTER OCCUPATIONS; ALL OTHER      : 6705   3rd Qu.: 94432
                    CONSULTANT              :  2185   17-2072: 3670   SOFTWARE DEVELOPERS; SYSTEMS SOFTWARE: 6071   Max.   :239566
                    (Other)                 :109130   (Other):48918   (Other)                              :59222
 WAGE_RATE_OF_PAY_FROM WAGE_RATE_OF_PAY_TO H.1B_DEPENDENT     Days
 Min.   : 15000        Min.   : 13500       :    0        Length:128037
 1st Qu.: 65645        1st Qu.: 84800      N:94151        Class :difftime
 Median : 80000        Median :104187      Y:33886        Mode  :numeric
 Mean   : 85334        Mean   :110023
 3rd Qu.:100000        3rd Qu.:130000
 Max.   :246715        Max.   :249652
```

Figure (I.2)

Columns Accepted for Analysis	**Reason to accept**
Case Number	Provides the exact case number that could be used for checking aggregation statistics.
Case Status	Provides value of the LCA case being accepted or rejected
Case Submitted	Provides the date values of application being submitted which is useful for time-gauging

Columns Accepted for Analysis	Reason to accept
Employment Start Date	Provides the date values which is useful for time-gauging
Employment End Date	Provides the date values which is useful for time-gauging
Employer name	Provides name of the employers that can be used to evaluate the top employers that frequently file for

Table (I.1)

Data Cleaning

The raw data from the OFLC website was inconsistent and messy. We did the following to obtain clean files:

i. The original files were in .xlsx format with incorrect and inconsistent column headers. For eg: the column name Case status in every file was different like CASE_STATUS or Case-status. Similarly, some files had EmployerAdress and also EmployerAdress1.These files were compared and the column names were made consistent.

ii. The employer names in the original .xlsx format had additional spaces or too many commas eg: ABC LTD, INC. Such inconsistencies were also corrected in each of the files by managing the spaces and changing the commas to semi-colons.

iii. The original .xlsx format of the file was then changed to .csv format as it is easier to work with .csv files as seen in figure (I.3) below.

```
> ########################################################
> #=============Setup the Shwetha Working Directory=============#
> #Set the working directory to the project folder by #
> #running the appropriate script below. Note, you can #
> #run the data off of your OneDrive or DropBox.#
> ########################################################
>
> ###Check for the presence of files prior to attempting to open it
> fl_check=file.exists("C:\\Users\\Shwetha Ak\\Desktop\\Coursework\\Sem2-Spring 2018\\R and Python\\Project\\H-1B_Disclosure_Data_FY15.csv")
>
>
> #### Open the files and data of each financial year to a data frame
> h1bfy15=read.csv("C:\\Users\\Shwetha Ak\\Desktop\\Coursework\\Sem2-Spring 2018\\R and Python\\Project\\H-1B_Disclosure_Data_FY15.csv", header=T, sep=",")
> h1bfy16=read.csv("C:\\Users\\Shwetha Ak\\Desktop\\Coursework\\Sem2-Spring 2018\\R and Python\\Project\\H-1B_Disclosure_Data_FY16.csv", header=T, sep=",")
> h1bfy17=read.csv("C:\\Users\\Shwetha Ak\\Desktop\\Coursework\\Sem2-Spring 2018\\R and Python\\Project\\H-1B_Disclosure_Data_FY17.csv", header=T, sep=",")
>
> ###Consolidate all the data files and append it into single data frame
> final_consolidated_H1B = rbind(h1bfy15,h1bfy16,h1bfy17)
Warning messages:
1: In `[<-.factor`(`*tmp*`, ri, value = c(0, 0, 451100, 6e+05, 0, 0,  :
  invalid factor level, NA generated
2: In `[<-.factor`(`*tmp*`, ri, value = c(67320, 57200, 0, 0, 0, 0,  :
  invalid factor level, NA generated
>
> ###Total Number of Original Consolidated columns and rows
> nrow(final_consolidated_H1B)
[1] 1891306
> ncol(final_consolidated_H1B)
[1] 39
>
```

Figure(I.3)

iv. The csv files of the 3 years were loaded into R. We checked the data types of all the 39 columns and found too many factor variables as seen in figure (I.4) below

```
'data.frame':   1733815 obs. of  39 variables:
 $ CASE_NUMBER          : Factor w/ 1849093 levels "I-200-09121-701936",..: 1 2 3 4 5 6 7 8 9 11 ...
 $ CASE_STATUS          : Factor w/ 4 levels "CERTIFIED","CERTIFIED-WITHDRAWN",..: 4 1 4 1 1 4 1 1 1 1 ...
 $ CASE_SUBMITTED       : Factor w/ 2086 levels "1/1/2015","1/10/2012",..: 490 309 76 511 327 49 123 327 173 147 ...
 $ DECISION_DATE        : Factor w/ 1077 levels "1/1/2015","1/10/2015",..: 143 100 97 155 106 15 43 106 85 51 ...
 $ VISA_CLASS           : Factor w/ 5 levels "E-3 Australian",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ EMPLOYMENT_START_DATE: Factor w/ 2331 levels "","09/31/2015",..: 711 132 140 1580 1204 88 389 1121 328 919 ...
 $ EMPLOYMENT_END_DATE  : Factor w/ 2403 levels "","1/1/2015",..: 704 132 140 1652 1261 88 355 1175 336 967 ...
 $ EMPLOYER_NAME        : Factor w/ 145488 levels "","'K' LINE LOGISTICS USA INC.",..: 40187 66658 44576 45930 22508 4675 68548 22509 68548 68548
 $ EMPLOYER_ADDRESS     : Factor w/ 140097 levels "","# 200 MIDDLESEX ESSEX TURNPIKE",..: 55918 62095 51898 63353 59777 31597 36627 59777 36627 36
 $ EMPLOYER_CITY        : Factor w/ 7446 levels "","10011","10019",..: 2916 3226 2180 2100 2803 2558 3486 2803 3486 3486 ...
 $ EMPLOYER_STATE       : Factor w/ 59 levels "","AK","AL","AR",..: 26 40 48 48 50 26 6 50 6 6 ...
 $ EMPLOYER_POSTAL_CODE : Factor w/ 15676 levels "","00717-9997",..: 5983 7590 7806 8285 2113 5892 10523 2113 10523 10523 ...
 $ EMPLOYER_COUNTRY     : Factor w/ 11 levels "","AFGHANISTAN",..: 7 7 7 7 7 7 7 7 7 7 ...
 $ EMPLOYER_PROVINCE    : Factor w/ 666 levels "","-","(201) 336-9431",..: 1 1 1 1 1 1 273 1 273 1 1 ...
 $ EMPLOYER_PHONE       : chr  "7635052710" "4053251826" "9728945000" "7138491700" ...
 $ EMPLOYER_PHONE_EXT   : chr  NA NA NA NA ...
 $ AGENT_ATTORNEY_NAME  : Factor w/ 21681 levels "","(DAVID) LIHWEI LIN",..: 1728 1 2694 2009 3040 1728 3013 3040 3013 3013 ...
 $ AGENT_ATTORNEY_CITY  : Factor w/ 1667 levels "","2915 WEST DEVON AVE",..: 738 1 275 1064 1233 738 68 1233 68 68 ...
 $ AGENT_ATTORNEY_STATE : Factor w/ 56 levels "","AK","AL","AR",..: 26 1 48 17 9 26 48 9 48 48 ...
 $ JOB_TITLE            : Factor w/ 203184 levels "","(FASHION) SHOE DESIGNER",..: 39954 4484 53696 15730 18911 9901 74090 16408 50365 40498 ...
 $ SOC_CODE             : Factor w/ 1111 levels "","<FONT><FONT>47-2031</FONT></FONT>",..: 286 464 205 266 163 381 165 214 158 165 ...
 $ SOC_NAME             : Factor w/ 1733 levels "","<FONT><FONT>CARPINTEROS</FONT></FONT>",..: 628 368 199 352 208 597 910 719 235 910 ...
 $ NAIC_CODE            : chr  "334510" "611310" "517212" "335314" ...
 $ TOTAL.WORKERS        : int  1 1 1 2 1 1 1 1 1 1 ...
 $ FULL_TIME_POSITION   : Factor w/ 3 levels "","N","Y": 3 3 3 3 3 3 3 3 3 3 ...
 $ PREVAILING_WAGE      : num  19000 42860 73965 65998 96907 ...
 $ PW_UNIT_OF_PAY       : Factor w/ 6 levels "","Bi-Weekly",..: 6 6 6 6 6 6 6 6 6 6 ...
 $ PW_WAGE_SOURCE       : Factor w/ 8 levels "","CBA","DBA",..: 4 7 4 4 4 4 2 7 4 7 7 ...
 $ PW_WAGE_SOURCE_YEAR  : int  2014 2014 2012 2014 2014 2015 2014 2014 2014 2014 ...
 $ PW_WAGE_SOURCE_OTHER : Factor w/ 8651 levels "","'FLC ONLINE DATA CENTER",..: 2530 2221 2221 2221 2221 963 2784 2221 2784 2784 ...
 $ WAGE_RATE_OF_PAY_FROM: num  20000 85000 94000 66000 97000 ...
 $ WAGE_RATE_OF_PAY_TO  : Factor w/ 13849 levels "","  N/A","0",..: 1 1 1 9630 4291 1 1 3177 1 1 ...
 $ WAGE_UNIT_OF_PAY     : Factor w/ 6 levels "","Bi-Weekly",..: 6 6 6 6 6 6 6 6 6 6 ...
 $ H.1B_DEPENDENT       : Factor w/ 3 levels "","N","Y": 2 2 2 2 2 2 2 2 2 2 ...
 $ WILLFUL.VIOLATOR     : Factor w/ 3 levels "","N","Y": 2 2 2 2 2 2 2 2 2 2 ...
 $ WORKSITE_CITY        : Factor w/ 13371 levels "","# 19100 DALLAS TX MSA",..: 2210 5237 6553 3416 4506 6880 5636 4506 5636 5636 ...
 $ WORKSITE_COUNTY      : Factor w/ 6376 levels "","-.",";DUPAGE",..: 3542 673 3186 1456 1124 1982 3270 1124 3270 3270 ...
 $ WORKSITE_STATE       : Factor w/ 59 levels "","AK","AL","AR",..: 6 42 6 51 53 22 6 53 6 6 ...
 $ WORKSITE_POSTAL_CODE : Factor w/ 22978 levels "","'02171","$48;797",..: 8391 10609 13992 11518 2855 8269 14572 2855 14572 14572 ...
```

Figure (I.5)

Normality and Outliers Assessment

Our assessment of outliers before cleansing had significant deviations and qualitative assessments and filtering were done as detailed above. We observed that the predictor variable, say prevailing wage did show some outliers even after qualitative assessment filtering, however these outliers were comparatively few in number and seemed valid. The distribution indicated normal distribution.

Outlier and Normality Analysis of Prevailing Wage can be seen in figure (I.6) and figure (I.7)

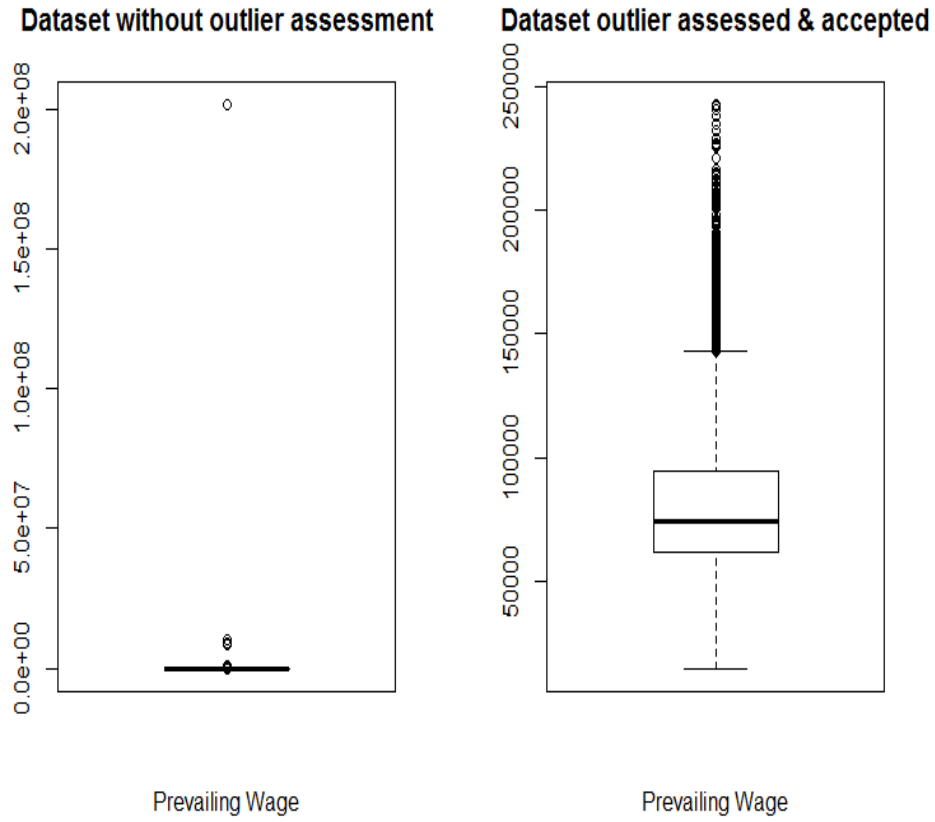


Figure (I.6)

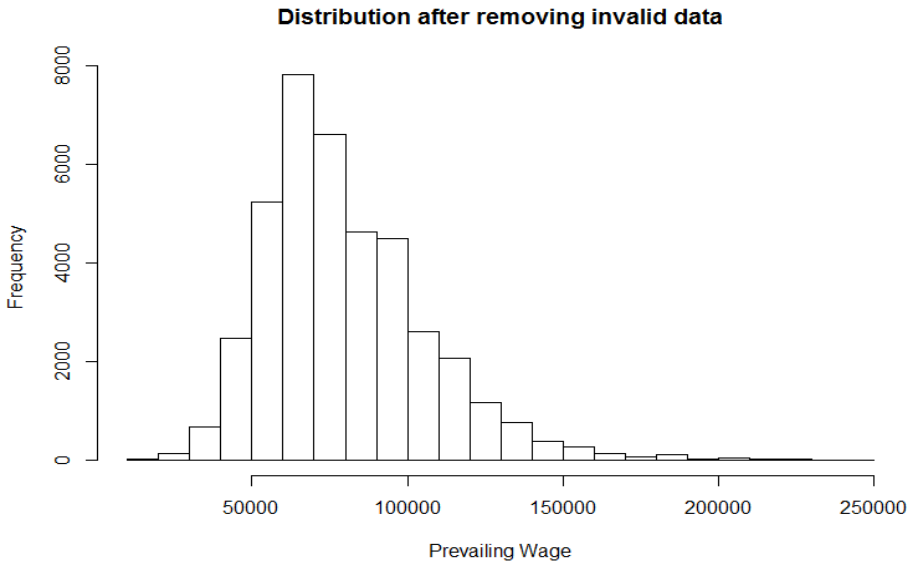


Figure (I.7)

Outlier and Normality of variable Wage_rate_of_pay_from can be seen in figure (I.8) and figure (I.9)

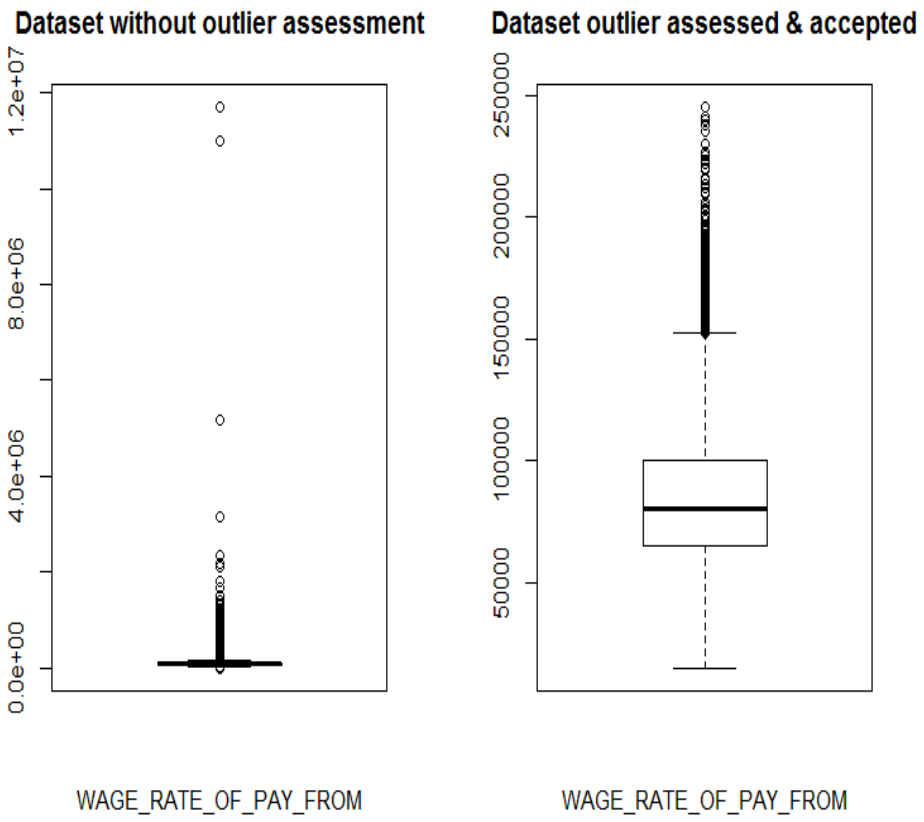


Figure (I.8)

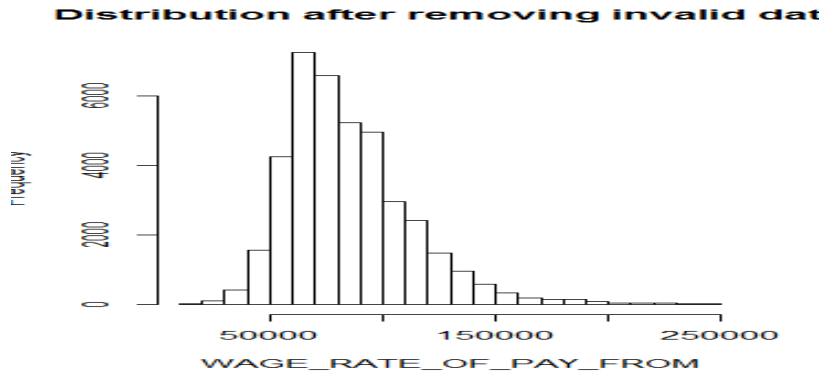


Figure (I.9)

Outlier Assessment of variable Wage_rate_pay_to can be seen in figure (I.10)

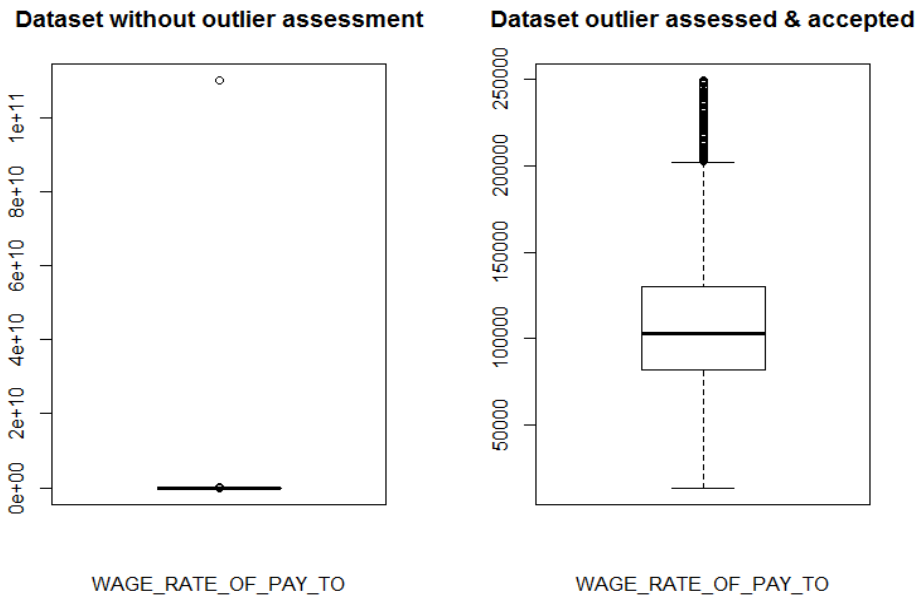


Figure (I.10)

Data Transformation

i. We then transformed the datatypes for the variables. We transformed the Employment_end_date column to convert to date type as seen in figure (I.11) below

```
> ##Conversion of Factor columns to needed datatypes
> hlb_analyzed$CASE_NUMBER = as.character(hlb_analyzed$CASE_NUMBER)
> hlb_analyzed$CASE_SUBMITTED = as.Date(hlb_analyzed$ CASE_SUBMITTED)
> hlb_analyzed$DECISION_DATE = as.Date(hlb_analyzed$DECISION_DATE)
> hlb_analyzed$EMPLOYMENT_START_DATE = as.Date(hlb_analyzed$EMPLOYMENT_START_DATE)
> hlb_analyzed$EMPLOYMENT_END_DATE  = as.Date(hlb_analyzed$EMPLOYMENT_END_DATE)
```

```
> hlb_analyzed$EMPLOYER_NAME  = as.character(hlb_analyzed$EMPLOYER_NAME)
> hlb_analyzed$EMPLOYER_ADDRESS  = as.character(hlb_analyzed$EMPLOYER_ADDRESS)
> hlb_analyzed$EMPLOYER_CITY  = as.character(hlb_analyzed$EMPLOYER_CITY)
> hlb_analyzed$EMPLOYER_STATE  = as.character(hlb_analyzed$EMPLOYER_STATE)
> hlb_analyzed$EMPLOYER_COUNTRY  = as.character(hlb_analyzed$EMPLOYER_COUNTRY)
> hlb_analyzed$EMPLOYER_PROVINCE  = as.character(hlb_analyzed$EMPLOYER_PROVINCE)
> hlb_analyzed$AGENT_ATTORNEY_NAME   = as.character(hlb_analyzed$AGENT_ATTORNEY_NAME)
> hlb_analyzed$AGENT_ATTORNEY_CITY  = as.character(hlb_analyzed$AGENT_ATTORNEY_CITY)
> hlb_analyzed$AGENT_ATTORNEY_STATE   = as.character(hlb_analyzed$AGENT_ATTORNEY_STATE)
> hlb_analyzed$PW_WAGE_SOURCE_OTHER    = as.character(hlb_analyzed$ PW_WAGE_SOURCE_OTHER )
> hlb_analyzed$WAGE_RATE_OF_PAY_TO   = as.numeric(hlb_analyzed$WAGE_RATE_OF_PAY_TO)
> hlb_analyzed$WORKSITE_CITY    = as.character(hlb_analyzed$WORKSITE_CITY)
> hlb_analyzed$WORKSITE_COUNTY   = as.character(hlb_analyzed$WORKSITE_COUNTY)
> hlb_analyzed$WORKSITE_STATE    = as.character(hlb_analyzed$WORKSITE_STATE)
> hlb_analyzed$WORKSITE_POSTAL_CODE    = as.character(hlb_analyzed$WORKSITE_POSTAL_CODE)
> hlb_analyzed$EMPLOYMENT_END_DATE  = as.Date(hlb_analyzed$EMPLOYMENT_END_DATE)
Error in charToDate(x) :

> hlb_analyzed$EMPLOYMENT_END_DATE =as.Date(strptime(hlb_analyzed$EMPLOYMENT_END_DATE , "%m/%d/%Y"))
>
```

Figure (I.11)

ii. Post transformations we get as below for the 39 columns as seen in figure (I.12)

```
'data.frame':   1733815 obs. of  39 variables:
 $ CASE_NUMBER          : chr  "I-200-09121-701936" "I-200-09146-796321" "I-200-09180-329758" "I-200-09183-259985" ...
 $ CASE_STATUS          : Factor w/ 4 levels "CERTIFIED","CERTIFIED-WITHDRAWN",..: 4 1 4 1 1 4 1 1 1 1 ...
 $ CASE_SUBMITTED       : Date, format: "0002-05-20" "0012-12-20" "0001-03-20" "0003-10-20" ...
 $ DECISION_DATE        : Date, format: "0002-05-20" NA NA NA ...
 $ VISA_CLASS           : Factor w/ 5 levels "E-3 Australian",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ EMPLOYMENT_START_DATE: Date, format: "0002-09-20" "0001-05-20" "0001-07-20" "0009-07-20" ...
 $ EMPLOYMENT_END_DATE  : Date, format: "2015-02-28" "2018-01-04" "2016-01-06" "2018-09-07" ...
 $ EMPLOYER_NAME        : chr  "MEDTRONIC; INC." "UNIVERSITY OF OKLAHOMA" "NOKIA INC." "OMRON OILFIELD AND MARINE; INC." ...
 $ EMPLOYER_ADDRESS     : chr  "710 MEDTRONIC PARKWAY NE" "905 ASP AVE" "6021 CONNECTION DRIVE" "9510 N. HOUSTON ROSSLYN ROAD" ...
 $ EMPLOYER_CITY        : chr  "MINNEAPOLIS" "NORMAN" "IRVING" "HOUSTON" ...
 $ EMPLOYER_STATE       : chr  "MN" "OK" "TX" "TX" ...
 $ EMPLOYER_POSTAL_CODE : Factor w/ 15676 levels "","00717-9997",..: 5983 7590 7806 8285 2113 5892 10523 2113 10523 10523 ...
 $ EMPLOYER_COUNTRY     : chr  "UNITED STATES OF AMERICA" "UNITED STATES OF AMERICA" "UNITED STATES OF AMERICA" "UNITED STATES OF AMERICA" ...
 $ EMPLOYER_PROVINCE    : chr  "" "" "" "" ...
 $ EMPLOYER_PHONE       : chr  "7635052710" "4053251826" "9728945000" "7138491700" ...
 $ EMPLOYER_PHONE_EXT   : chr  NA NA NA NA ...
 $ AGENT_ATTORNEY_NAME  : chr  "DEBRA SCHNEIDER" "" "HASEENA ENU" "ELDON KAKUDA" ...
 $ AGENT_ATTORNEY_CITY  : chr  "MINNEAPOLIS" "" "DALLAS" "SCHAUMBURG" ...
 $ AGENT_ATTORNEY_STATE : chr  "MN" "" "TX" "IL" ...
 $ JOB_TITLE            : Factor w/ 203184 levels "","(FASHION) SHOE DESIGNER",..: 39954 4484 53696 15730 18911 9901 74090 16408 50365 40498 ...
 $ SOC_CODE             : Factor w/ 1111 levels "","<FONT><FONT>47-2031</FONT></FONT>",..: 286 464 205 266 163 381 165 214 158 165 ...
 $ SOC_NAME             : Factor w/ 1733 levels "","<FONT><FONT>CARPINTEROS</FONT></FONT>",..: 628 368 199 352 208 597 910 719 235 910 ...
 $ NAIC_CODE            : chr  "334510" "611310" "517212" "335314" ...
 $ TOTAL.WORKERS        : int  1 1 1 2 1 1 1 1 1 1 ...
 $ FULL_TIME_POSITION   : Factor w/ 3 levels "","N","Y": 3 3 3 3 3 3 3 3 3 3 ...
 $ PREVAILING_WAGE      : num  19000 42860 73965 65998 96907 ...
 $ PW_UNIT_OF_PAY       : Factor w/ 6 levels "","Bi-Weekly",..: 6 6 6 6 6 6 6 6 6 6 ...
 $ PW_WAGE_SOURCE       : Factor w/ 8 levels "","CBA","DBA",..: 4 7 4 4 4 2 7 4 7 7 ...
 $ PW_WAGE_SOURCE_YEAR  : int  2014 2014 2012 2014 2014 2015 2014 2014 2014 2014 ...
 $ PW_WAGE_SOURCE_OTHER : chr  "ONLINE DATA SURVEY" "OFLC ONLINE DATA CENTER" "OFLC ONLINE DATA CENTER" "OFLC ONLINE DATA CENTER" ...
 $ WAGE_RATE_OF_PAY_FROM: num  20000 85000 94000 66000 97000 ...
 $ WAGE_RATE_OF_PAY_TO  : num  1 1 1 9630 4291 ...
 $ WAGE_UNIT_OF_PAY     : Factor w/ 6 levels "","Bi-Weekly",..: 6 6 6 6 6 6 6 6 6 6 ...
 $ H.1B_DEPENDENT       : Factor w/ 3 levels "","N","Y": 2 2 2 2 2 2 2 2 2 2 ...
 $ WILLFUL.VIOLATOR     : Factor w/ 3 levels "","N","Y": 2 2 2 2 2 2 2 2 2 2 ...
 $ WORKSITE_CITY        : chr  "EDEN PRAIRIE" "NORMAN" "SAN DIEGO" "HOUSTON" ...
 $ WORKSITE_COUNTY      : chr  "STERNS" "CLEVELAND" "SAN DIEGO" "HARRIS" ...
 $ WORKSITE_STATE       : chr  "CA" "OK" "CA" "TX" ...
 $ WORKSITE_POSTAL_CODE : chr  "55412" "73019" "92127" "77088" ...
```

Figure (I.12)

Data Reduction

i. The LCA Program of OFLC has LCA petitions filed for visas other than H1-B visa type also. We then applied a filter to target data that has visa type as H1-B, the wage pay type is yearly and the employer filing the petition is not a wilful violator. The need for this filter was to check fruitful and positive conditions of a petition that gets successfully accepted.

ii. Missing values in columns like Prevailing wages were comparatively negligible in number. Missing values or spaces or zero values for prevailing wages for a category is not acceptable and is considered invalid. Since the column being of required category and very limited in number we decided to remove it. The figure (I.13) below shows summary of data frame containing Prevailing wages.

```
> summary(h1b_filtered$PREVAILING_WAGE)
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
        0     57907     68827     74485     85738 201622735        17
> h1b_filtered$PREVAILING_WAGE[h1b_filtered$PREVAILING_WAGE == '' | h1b_filtere$
> h1b_filtered$WAGE_RATE_OF_PAY_FROM[h1b_filtered$WAGE_RATE_OF_PAY_FROM== '' | $
>
```

Figure (I.13)

iii. Prevailing wage is expected to be Not less than 10000 per year. When we analysed that situation here we found that there were around 1200 of those records and those being faulty entries we removed it. Figure (I.14) shows the reduced count of data.

```
> #Additional Filtering of the Dataset to eliminate wrong or invalid data entri$
> nrow(h1b_filtered)
[1] 1733815
> h1b_filtered=h1b_filtered[h1b_filtered$PREVAILING_WAGE>10000,]
> nrow(h1b_filtered)
[1] 1732568
```

Figure(I.14)

iv. Prevailing wages above 250000 are not expected to be present in the skilled category of H1B although there could be few exceptions. This filtering eliminated around 500+ records.

v. Wage Rate From" is also expected to be offered below 250000 for the skilled professionals for the H1B visa category and hence we have decided to go along with the filter condition below 250000. This eliminated just 7000 or so records. Figure(I.15) shows the massive reduction of records.

```
> h1b_filtered=h1b_filtered[h1b_filtered$WAGE_RATE_OF_PAY_FROM<250000,]
> nrow(h1b_filtered)
[1] 1723351
>
```

Figure (I.15)

vi. We checked for any prevailing imbalance in the data set. Since our dependent variable is a categorical variable, we verified for the scenario where the number of observations belonging to one class is significantly lower than those belonging to the other classes. Thus, we implemented re-sampling techniques to process the data with the objective to balance the dataset without any bias. This unbiased sampled data is having 39696 records. The figure (I.16) below shows that now we had nearly 40000 records to analyse our results.

```
> set.seed(1)
> index_sample_20=sample(1:nrow(cert),0.20*nrow(cert))
> sample_20=cert[index_sample_20,]
> sample_consol=rbind(sample_20,den,with,certwith)
```

Figure (I.16)

J. DESCRIPTIVE STATISTICS

We have used descriptive statistics mainly to obtain the following:

i. Unveiling Top Jobs in market for H1B based on demand classification.

ii. Top Companies filing for LCA petitions

iii. Company wise analysis of filed LCA status

The results of our analysis on the same is listed below:

Since we had already investigated the invalid data aspect and because the invalid outliers were removed, we have received a summary statistic that is as per our expectations. Figure (J.1) can show the results of the same.

```
> summary(sample_consol)
 CASE_NUMBER                    CASE_STATUS    CASE_SUBMITTED       EMPLOYMENT_START_DATE EMPLOYMENT_END_DATE
 Length:39391       CERTIFIED         :22161   Min.   :0001-01-20   Min.   :2011-01-31    Min.   :2013-10-16
 Class :character   CERTIFIED-WITHDRAWN:10312   1st Qu.:0003-03-20   1st Qu.:2015-07-15    1st Qu.:2018-06-12
 Mode  :character   DENIED            : 2115   Median :0003-11-20   Median :2016-01-19    Median :2018-11-16
                    WITHDRAWN         : 4803   Mean   :0005-08-05   Mean   :2016-02-10    Mean   :2019-01-06
                    I-200-16081-868857 :    0   3rd Qu.:0007-12-20   3rd Qu.:2016-10-06    3rd Qu.:2019-09-10
                                                Max.   :0012-12-20   Max.   :2018-03-09    Max.   :2021-03-09

 EMPLOYER_NAME      EMPLOYER_STATE                      JOB_TITLE        SOC_CODE
 Length:39391       Length:39391       SOFTWARE ENGINEER       : 2085   15-1132: 8398
 Class :character   Class :character   PROGRAMMER ANALYST      : 1005   15-1121: 4582
 Mode  :character   Mode  :character   SENIOR SOFTWARE ENGINEER:  834   15-1131: 3973
                                       SOFTWARE DEVELOPER      :  647   15-1199: 2934
                                       DEVELOPER               :  576   15-1133: 2755
                                       CONSULTANT              :  556   13-2051: 1136
                                       (Other)                 :33688   (Other):15613
                                  SOC_NAME     PREVAILING_WAGE  WAGE_RATE_OF_PAY_FROM WAGE_RATE_OF_PAY_TO
 SOFTWARE DEVELOPERS; APPLICATIONS    : 5472   Min.   : 15000   Min.   : 15000       Min.   : 13500
 COMPUTER SYSTEMS ANALYSTS            : 4555   1st Qu.: 61797   1st Qu.: 65000       1st Qu.: 81837
 COMPUTER PROGRAMMERS                 : 3959   Median : 74318   Median : 80000       Median :103000
 SOFTWARE DEVELOPERS, APPLICATIONS    : 2931   Mean   : 79373   Mean   : 84547       Mean   :108841
 COMPUTER OCCUPATIONS; ALL OTHER      : 1844   3rd Qu.: 94089   3rd Qu.: 99949       3rd Qu.:130000
 SOFTWARE DEVELOPERS; SYSTEMS SOFTWARE: 1788   Max.   :222498   Max.   :245000       Max.   :249365
 (Other)                              :18842
 H.1B_DEPENDENT     Days
  :    0       Length:39391
 N:29873       Class :difftime
 Y: 9518       Mode  :numeric
```

Figure (J.1)

i. The below graph seen in figure (J.2) below provides the details about the top 10 employers filing for the LCA petition. We can see that **INFOSYS LIMITED** stands the highest with 80000 LCA petitions.

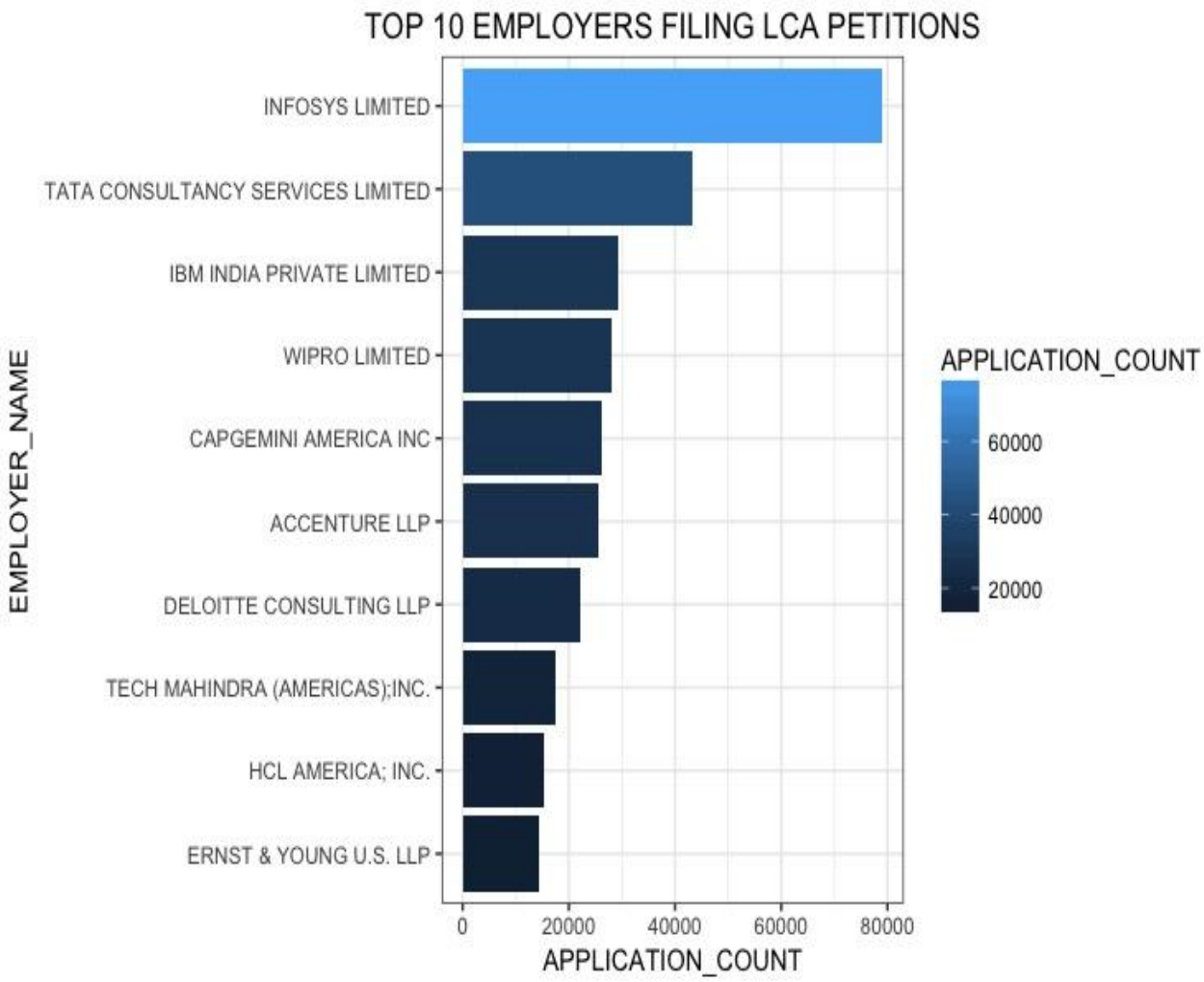


Figure (J.2)

ii. The below graph in figure (J.3) gives the details of top 10 employers and the case status of their LCA petitions. An interesting observation here is that INFOSYS LIMITED filed 80000 petitions out of which all the petitions are certified and none of them have been denied or withdrawn. On further analysis on the petitions filed by INFOSYS LIMITED may reveal some interesting points about the factors influencing the LCA petition approval.

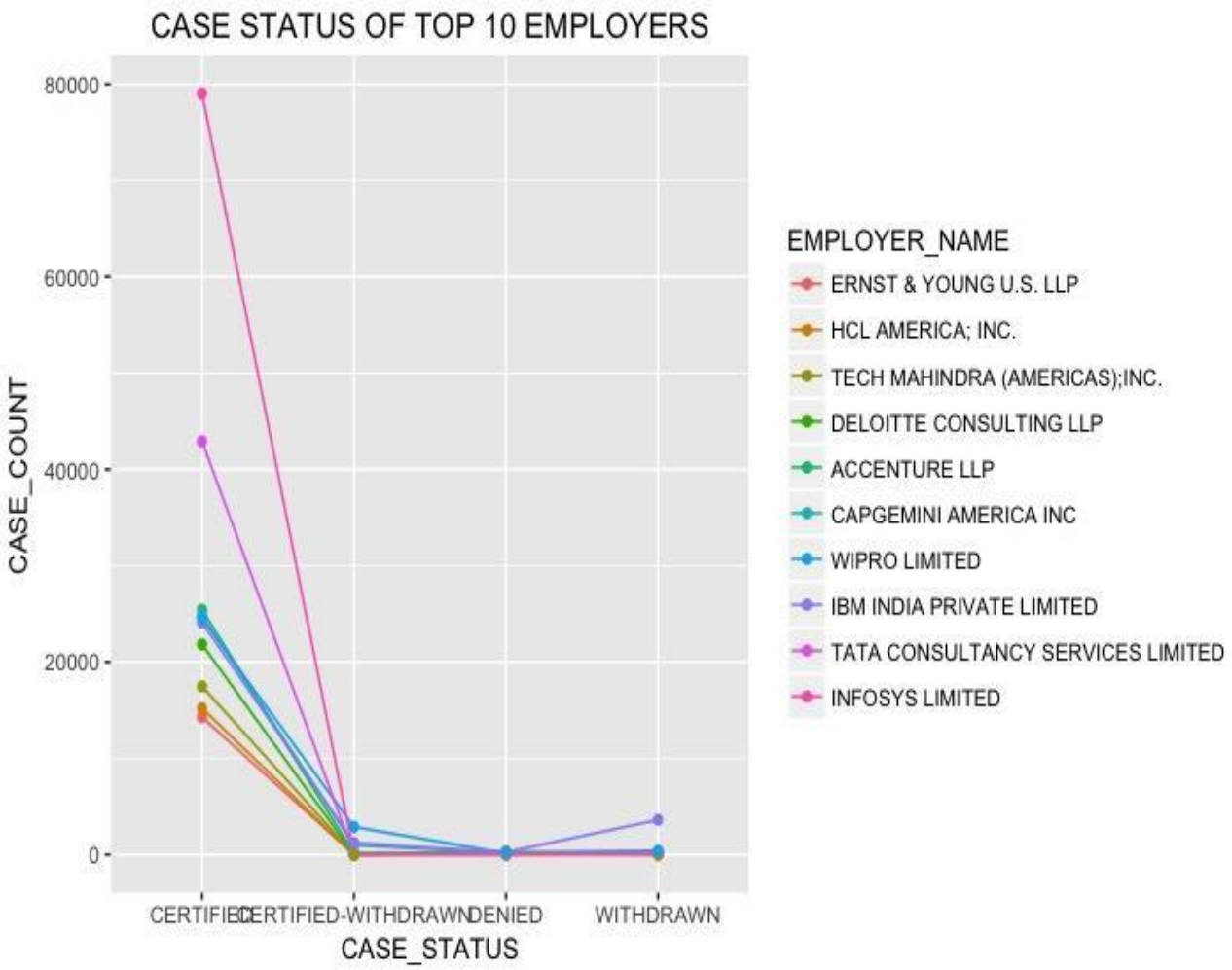


Figure (J.3)

iii. The below graph figure (J.4) reveals the top 10 hot jobs in the market for which most number of LCA petitions are filed. **Programmer Analyst** stands the top job in the market with highest number of petitions of around 1.5 million.

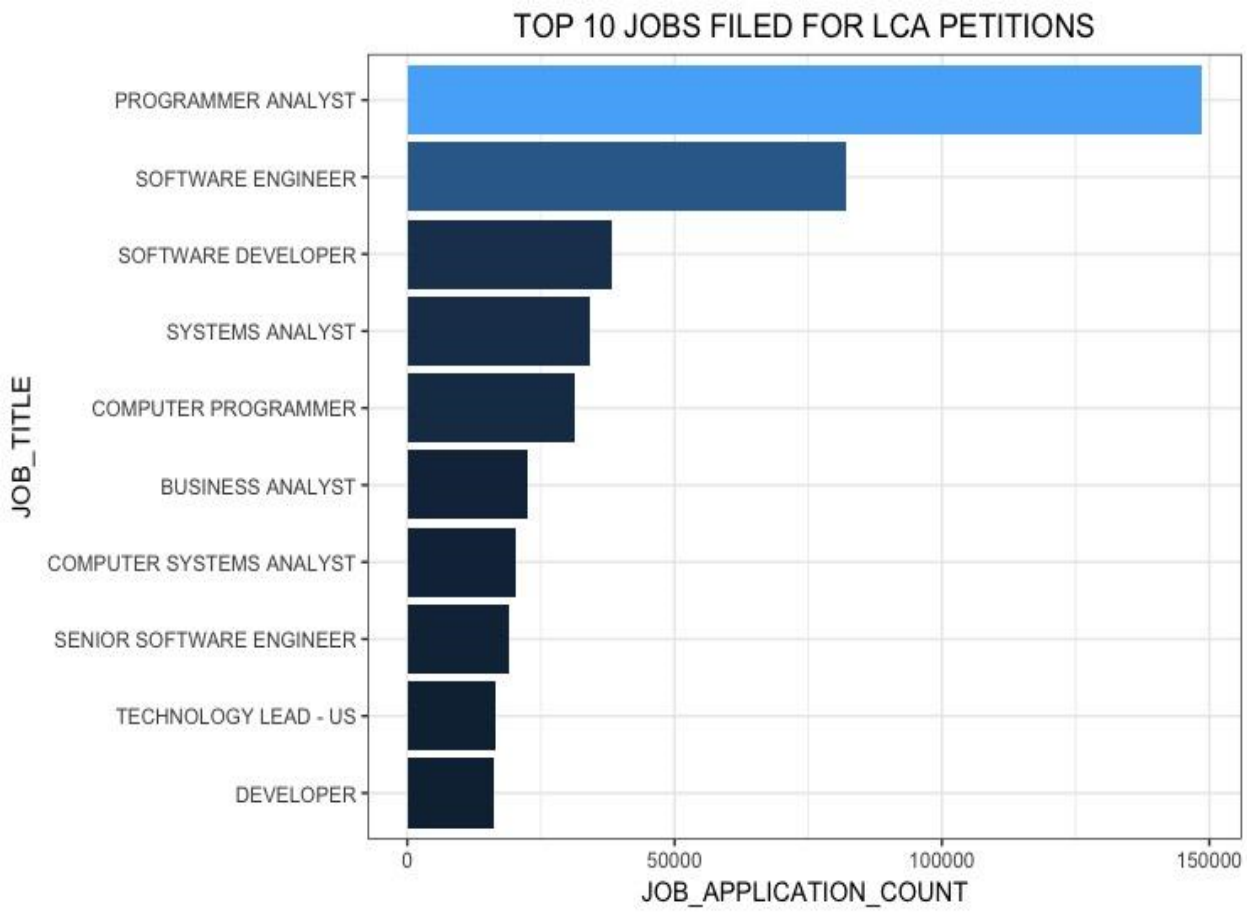


Figure (J.4)

K. MODELLING TECHNIQUES

In this project analysis report, the modelling techniques like Logistic Regression and Classification have been employed to unearth answers to the following:

i. Factors contributing to successful LCA petition filing.

ii. Predict whether LCA petition will be Approved based on factored variables.

Selection of Model

Our intention in using this dataset is to identify the factors that influence the LCA application approval. Since our variable of interest "CASE_STATUS" is an ordinal variable, with subgroups "CERTIFIED" and "DENIED", we will use logistic regression and classification

methods such that the outcome of our prediction model should predict if an LCA application is Approved or Denied.

MODEL 1 – LOGISTIC REGRESSION

MODEL 2 – CLASSIFICATION TREE

Assumptions of Logistic Regression

1. First, binary logistic regression requires the dependent variable to be binary and ordinal logistic regression requires the dependent variable to be ordinal. This assumption is satisfied in our model as the dependent variable "CASE_STATUS" is an ordinal variable in our final sample dataset with two subcategories.

2. Second, logistic regression requires there to be little or no multicollinearity among the independent variables. This means that the independent variables should not be too highly correlated with each other.

```
> cor(corr_subset)
                    PREVAILING_WAGE WAGE_RATE_OF_PAY_FROM WAGE_RATE_OF_PAY_TO       Days
PREVAILING_WAGE          1.00000000            0.91051510          0.7696078 0.09823053
WAGE_RATE_OF_PAY_FROM    0.91051510            1.00000000          0.8359021 0.09452761
WAGE_RATE_OF_PAY_TO      0.76960775            0.83590207          1.0000000 0.13542338
Days                     0.09823053            0.09452761          0.1354234 1.00000000
```

Figure (K.1)

We can see from the above figure K.1 correlation matrix that there is no multi-collinearity among the variables except between Wage_pay_from and Prevailing_wage.

Data Splitting and Subsampling

At this point, our sample dataset has around 40000 rows, out which the dependent variable "CASE_STATUS" has only 5920 rows which are LCA "Denied". When we take a wholistic view of the LCA applications filed, there are more number of LCA "Certified" applications when compared to the LCA "Denied" applications in dataset which makes our sample more biased.

Since, we are only interested to know if an LCA application got certified or denied, we further subsampled the model using stratified modelling technique, to get an unbiased sample with equal number of LCA "Certified" and "Denied" Applications.

Since our final subsample dataset has around 10000 rows, which is not a very large number, we needed to train our dataset more for it to understand the underlying inputs and their implications on the prediction variable. Since our dependent variable is categorical, there is a need to have the training dataset in a bigger ratio than the test dataset to achieve a better accuracy in our prediction model. Therefore, we divided our final subsample data into train data and test data in the ratio of 80:20 which consisted of 8000 rows of train data and 2000 rows of test data. Below are the screenshots that gives the gist of how the we achieved an unbiased data by following the stratified sampling technique for both train and test data.

Train data using stratified sampling for unbiased data – 80%

The code can be seen below in the Figure K.2

```
#############################################################################
#####OBTAIN THE TRAIN DATA AND THEN PERFORM THE LOGISTIC REGRESSION
#####USE STRATIFIED SAMPLING FOR UNBIASED DATA
#############################################################################

###Stratified Data for Train Sampling
set.seed(1)
train_data <- stratified(data_consolidated, c("CASE_STATUS"),replace = FALSE, 4000)
train_data$CASE_STATUS <- factor(train_data$CASE_STATUS)
unique(train_data$CASE_STATUS)
head(train_data)
nrow(train_data)


#########################################################
####TRAIN_DATA STRATIFIED SAMPLING RESULT
#########################################################@
##TOTAL ROWS
nrow(train_data)
##CERTIFIED
nrow(train_data[train_data$CASE_STATUS == "CERTIFIED",])
##DENIED
nrow(train_data[train_data$CASE_STATUS == "DENIED",])
```

Figure (K.2)

The output for the same can be seen in figure K.3 below

```
> ####################################################
> ####TRAIN_DATA STRATIFIED SAMPLING RESULT
> ####################################################@
> ##TOTAL ROWS
> nrow(train_data)
[1] 8000
> ##CERTIFIED
> nrow(train_data[train_data$CASE_STATUS == "CERTIFIED",])
[1] 4000
> ##DENIED
> nrow(train_data[train_data$CASE_STATUS == "DENIED",])
[1] 4000
```

Figure (K.3)

Test data using stratified sampling for unbiased data – 20%

The code can be seen below in the figure K.4

```
#############################################################################
#####OBTAIN THE TEST DATA AND THEN PERFORM THE LOGISTIC REGRESSION
#####USE STRATIFIED SAMPLING FOR UNBIASED DATA
#############################################################################

###Stratified Data for Test Sampling
set.seed(1)
test_data <- stratified(data_consolidated, c("CASE_STATUS"),replace = FALSE, 1000)
test_data$CASE_STATUS <- factor(test_data$CASE_STATUS)
head(test_data)
nrow(test_data)
```

Figure (K.4)

The output for the same can be seen in figure K.5 below

```
> ##########################################################
> ####TEST_DATA STRATIFIED SAMPLING RESULT
> ##########################################################@
> ##TOTAL ROWS
> nrow(test_data)
[1] 2000
> ##CERTIFIED
> nrow(test_data[test_data$CASE_STATUS == "CERTIFIED",])
[1] 1000
> ##DENIED
> nrow(test_data[test_data$CASE_STATUS == "DENIED",])
[1] 1000
```

figure (k.5)

L. DATA MODELLING

Before we start modelling our data, there is a need to identify any nominal variables and convert them into dummy variables to help the regression mode identify different subgroups.

In our dataset, we have “H1.B_DEPENDENT” as nominal variable with categories as “Yes” and “No”. Therefore we created dummy variables to represent the subgroups “Yes” and “No” as “H1.B_DEPENDENT_Y” and “H1.B_DEPENDENT_N” respectively.

Creation of Dummy variable for H1B_dependent

We can see the respective code in figure (L.1) and the corresponding output in figure (L.2) below.

```
###Create Dummy Variables for H1B Dependent
unique(data_consolidated$H.1B_DEPENDENT)
h1_b_dep = dummy(data_consolidated$H.1B_DEPENDENT, sep = '_')
colnames(h1_b_dep)
h1_b_dep = as.data.frame(h1_b_dep)
data_consolidated = data.frame(data_consolidated, h1_b_dep)
names(data_consolidated)
```

Figure (L.1)

```
> names(data_consolidated)
 [1] "CASE_NUMBER"          "CASE_STATUS"          "CASE_SUBMITTED"       "EMPLOYMENT_START_DATE"
 [5] "EMPLOYMENT_END_DATE"  "EMPLOYER_NAME"        "EMPLOYER_STATE"       "JOB_TITLE"
 [9] "SOC_CODE"             "SOC_NAME"             "PREVAILING_WAGE"      "WAGE_RATE_OF_PAY_FROM"
[13] "WAGE_RATE_OF_PAY_TO"  "H.1B_DEPENDENT"       "Days"                 "H.1B_DEPENDENT_N"
[17] "H.1B_DEPENDENT_Y"
```

Figure (L.2)

Model 1: Logistic Regression

USING TRAINING DATA

Logistic regression is performed with following variables

Dependent variable: CASE_STATUS

Independent variables: H.1B_DEPENDENT, PREVAILING_WAGE and Days.

The output is seen in the screen shot below in figure (L.3)

```
> ###############################################################################
> #            MODEL - 1a       LOGISTIC REGRESSION - CASE-STATUS PREDICTION - TRAIN DATA
> ###############################################################################
> case_status_prediction_train=glm(CASE_STATUS~PREVAILING_WAGE + Days + H.1B_DEPENDENT_Y ,family=binomial(link='logit'), data = train_data)
> summary(case_status_prediction_train)

Call:
glm(formula = CASE_STATUS ~ PREVAILING_WAGE + Days + H.1B_DEPENDENT_Y,
    family = binomial(link = "logit"), data = train_data)

Deviance Residuals:
     Min       1Q   Median       3Q      Max
-1.77125  -1.21001  -0.08277   1.12569   1.58745

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      1.490e+00  2.042e-01   7.297 2.93e-13 ***
PREVAILING_WAGE -3.104e-06  9.838e-07  -3.155   0.0016 **
Days            -1.022e-03  1.807e-04  -5.657 1.54e-08 ***
H.1B_DEPENDENT_Y -7.151e-01  6.338e-02 -11.283  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8317.8  on 5999  degrees of freedom
Residual deviance: 8155.1  on 5996  degrees of freedom
AIC: 8163.1

Number of Fisher Scoring iterations: 4
```

Figure (L.3)

From the regression model we can see that all the independent variables taken into consideration are significant (with p values < alpha at 0.5) in predicting the dependent variable

Equation for the Model

CASE_STATUS= 1.391e+00 – 3.18e^-06*PREVAILING_WAGE- 9.09e-04*Days – 7.700e-01*H1.B_DEPENDENT_Y

- For every one-dollar increase in PREVAILING_WAGE, the log odds of being CERTIFIED decreases by -3.198e-06 holding all other variables constant.
- For every one-day increase in Days, the log odds of being CERTIFIED decreases by -9.090e-04 holding all other variables constant.
- If a company is H1.B_DEPENDENT(I,e H.1B_DEPENDENT_Y is true), the log odds of being CERTIFIED decreases by -7.700 e-1 holding all other variables constant.
  From the above interpretation we can see that:

Model 1 Interpretation

1. When prevailing wage is increased beyond necessity for a particular skillset with less demand, there is higher chance of LCA application being denied.
2. The H1B Visa class is meant to bring in speciality skilled workers who would be serving the project functionality for a shorter duration as a stop gap measure to replace the shortage in skilled workers in USA. Taking this into consideration, regression results revealed that the more number of days the employee intend to spend within united states under the LCA/H1B, the more likely it is to be denied.
3. When a company is dependent on the foreign workers and want to retain its employees to perform its functions, it is highly likely that there is a greater chance of the LCA application being denied(rejected).

USING TESTING DATA

The needed code is seen in the screen shot below in figure (L.4) as below

```
#############################################################################
#           MODEL - 1b      LOGISTIC REGRESSION - CASE-STATUS PREDICTION - TEST DATA
#############################################################################

###TEST - DATA
###LOGISTIC REGRESSION PREDCITING THE CASE STATUS
case_status_prediction_test=glm(CASE_STATUS~PREVAILING_WAGE + Days + H.1B_DEPENDENT_Y ,family=binomial(link='logit'), data = test_data)
summary(case_status_prediction_test)

###After Building The model with train Data Check it with Test Data for Accuracy
predict_logit=predict(case_status_prediction_train,newdata = test_data,type='response')
summary(predict_logit)

#Cut-off value = 0.5
pred_cut_off <- ifelse(predict_logit > 0.5, 'CERTIFIED','DENIED') #Setting cut-off to be at 0.5
table(test_data$CASE_STATUS,pred_cut_off)
```

Figure (L.4)

The confusion matrix seen in the screen shot below in figure (L.5) as below

```
> #Cut-off value = 0.5
> pred_cut_off <- ifelse(predict_logit > 0.5, 'CERTIFIED','DENIED') #Setting cut-off to be at 0.5
> table(test_data$CASE_STATUS,pred_cut_off)
           pred_cut_off
            CERTIFIED DENIED
  CERTIFIED       664    336
  DENIED          800    200
```

Figure (L.5)

Below are the results of the confusion matrix

i. 664 records out of 1000 are correctly predicted as Certified

ii. 336 records out of 1000 are wrongly predicted as Denied when they are actually certified.

iii. 200 records out of 1000 are correctly predicted as Denied

iv. 800 records out of 1000 are Wrongly predicted as Certified when they are actually Denied.

Model Conclusion

From the above results we can observe that our prediction model has an accuracy of 43.2%.

Model 2: Classification Tree

Below is the model prediction using the classification:

Dependent variable: CASE_STATUS

Independent variables: H.1B_DEPENDENT, PREVAILING_WAGE and Days.

The figure (L.6) shows the code for the model

```
#############################################################################
#           MODEL - 1b      LOGISTIC REGRESSION - CASE-STATUS PREDICTION - TEST DATA
#############################################################################

###TEST - DATA
###LOGISTIC REGRESSION PREDCITING THE CASE STATUS
case_status_prediction_test=glm(CASE_STATUS~PREVAILING_WAGE + Days + H.1B_DEPENDENT_Y ,family=binomial(link='logit'), data = test_data)
summary(case_status_prediction_test)

###After Building The model with train Data Check it with Test Data for Accuracy
predict_logit=predict(case_status_prediction_train,newdata = test_data,type='response')
summary(predict_logit)

#Cut-off value = 0.5
pred_cut_off <- ifelse(predict_logit > 0.5, 'CERTIFIED','DENIED') #Setting cut-off to be at 0.5
table(test_data$CASE_STATUS,pred_cut_off)
```

Figure (L.6)

The output derived is observed in the figure (L.7) below

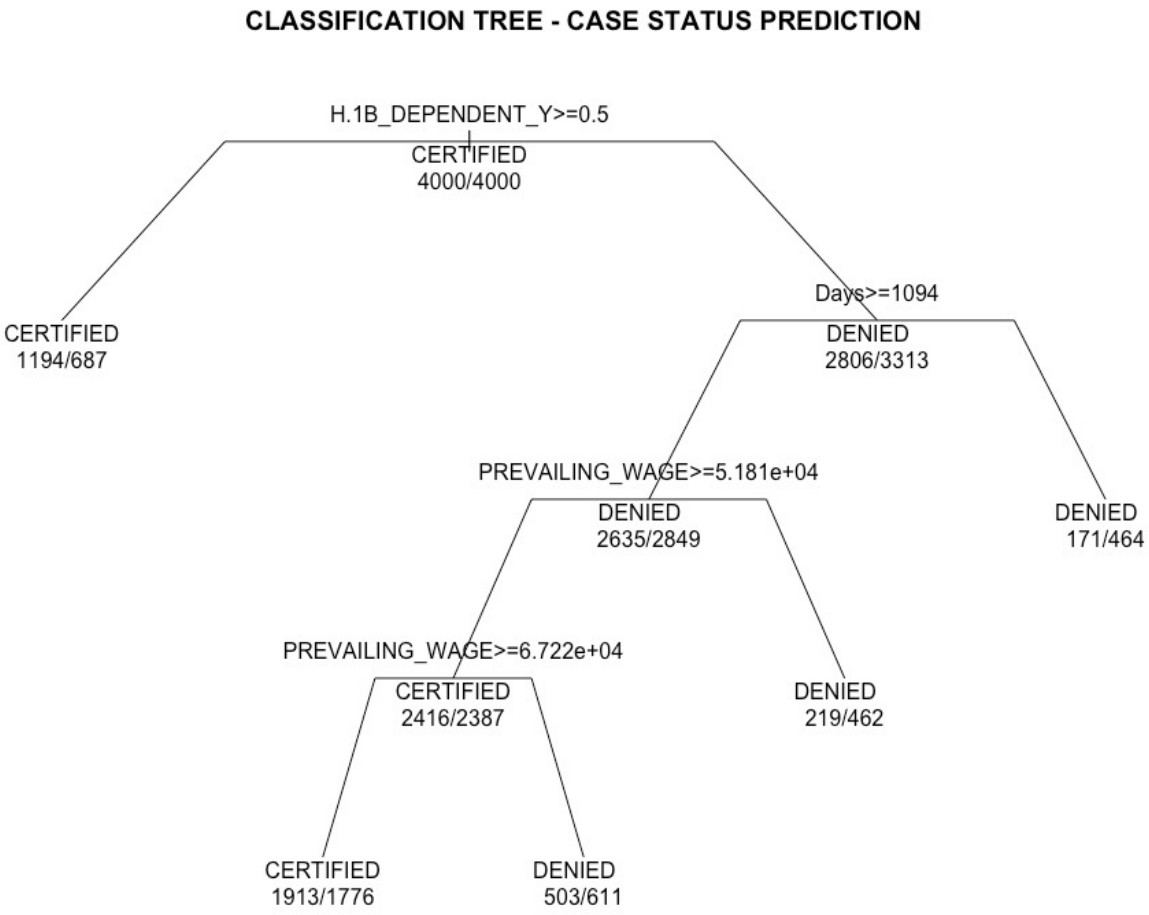


Figure (L.7)

Model 2 Interpretation

i. From the above classification tree, we can see that the root node is H1.B_DEPENDENT_Y which tells us that H1.B_DEPENDENT_Y is the most influential or significant predictor of the CASE_STATUS.
ii. The next significant variable is the Days and then PREVAILING_WAGE.
iii. It is obvious that H1.B_DEPENDENT is significant predictor of the CASE_STATUS because, when a company is dependent on the foreign workers

and want to retain its employees to perform its functions, it is highly likely that there is a greater chance of the LCA application being approved.

The confusion matrix seen in the screen shot below in figure (L.8) as below

```
#########################################################################################################################
###                          MODEL - 2b  - CLASSICFICATION TREE - CASE STATUS PRODUCTION USING CONFUSION MATRIX - TEST DATA
#########################################################################################################################

###CONFUSION MATRIX FOR THE TEST DATA AND SEE THE ACCURACY
case_status_check <- predict(classifier, test_data, type="class")
table(test_data$CASE_STATUS, case_status_check)
          case_status_check
           CERTIFIED DENIED
CERTIFIED       778    222
DENIED          589    411
```

Figure (L.8)

Below are the results of the confusion matrix

i. 778 records out of 1000 are correctly predicted as Certified
ii. 222 records out of 1000 are wrongly predicted as Denied when they are actually certified.
iii. 411 records out of 1000 are correctly predicted as Denied
iv. 589 records out of 1000 are Wrongly predicted as Certified when they are actually Denied.

Model Conclusion:

From the above results we can observe that our prediction model is 59.45% correct.

Model Assessment

i. Out of the above modelling techniques we think the best one is Classification technique
ii. The Classification technique had a prediction value of nearly 60%

iii. The Classification model had enough evidence to prove that the variables H.1B_DEPENDENT, PREVAILING_WAGE and Day were significant predictors for the Case_Status of a filed LCA application

iv. The model also showed that if a company is H1B_dependent this strongly favours for a successful LCA filing for its H1B employees.

M. FORECASTING MODELS USING TIME SERIES

In our endeavor, to research on the project, our objectives were to predict the following using time series approach:

- Forecasting the number of LCA applications to be filed between Oct 2017 to May 2018
- Forecasting the average wage offered to Programmer analyst between Oct 2017 to May 2018

Forecasting the number of LCA applications to be filed between Oct 2017 to May 2018

Data Extraction for time series

We have extracted the number of LCA applications from Jan 2015 through September 2017 and converted the same to time series data as seen in the figure (M.1) below

```
> #### COUNT OF APPLICATIONS
> pgm_df1=sqldf('select CASE_SUBMITTED,count(*)as "APP_COUNT" from da$
>
>
> # We are analyzing the H1B LCA application submissions from the yea$
>
> cnt_app=pgm_df1[53:85,]
> appcnt_ts=ts(cnt_app$APP_COUNT,frequency=12,start=c(2015,1))
> head(appcnt_ts,75)
 [1]  35234  84123 167082  39659  34991  39286  37701  32896  30675
[10]  28043  23565  29637  34432  88250 188297  37648  35178  38609
[19]  30575  34983  27484  25501  26085  27270  37077  75400 189483
[28]  30390  30654  30683  25882  31674  21475
> #appcnt_ts contains app for the H1B LCA from year 2015 till 2017
> appcnt_ts
        Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep
2015  35234  84123 167082  39659  34991  39286  37701  32896  30675
2016  34432  88250 188297  37648  35178  38609  30575  34983  27484
2017  37077  75400 189483  30390  30654  30683  25882  31674  21475
        Oct    Nov    Dec
2015  28043  23565  29637
2016  25501  26085  27270
2017
>
```

Figure (M.1)

Seasonality Assessment and Decomposition

Here we can see high level of seasonality with applications surging in March of every year. This is possibly because of the year end LCA application rush which can be verified as seen in figure (M.2) below.

```
> # Assessment of Seasonality
> # using boxplot
> boxplot(appcnt_ts ~ cycle(appcnt_ts))
```

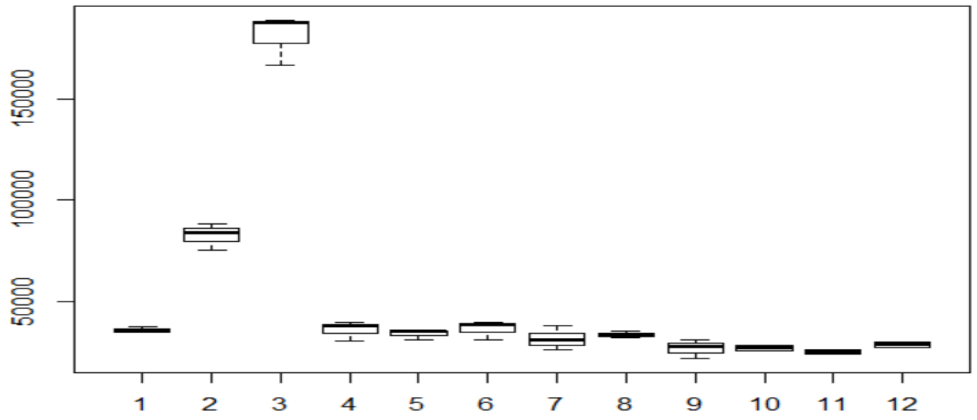


figure (M.2)

On decomposition, we see that the data has high seasonal component from the decomposed data series as well as seen in figure (M.3) seen below.

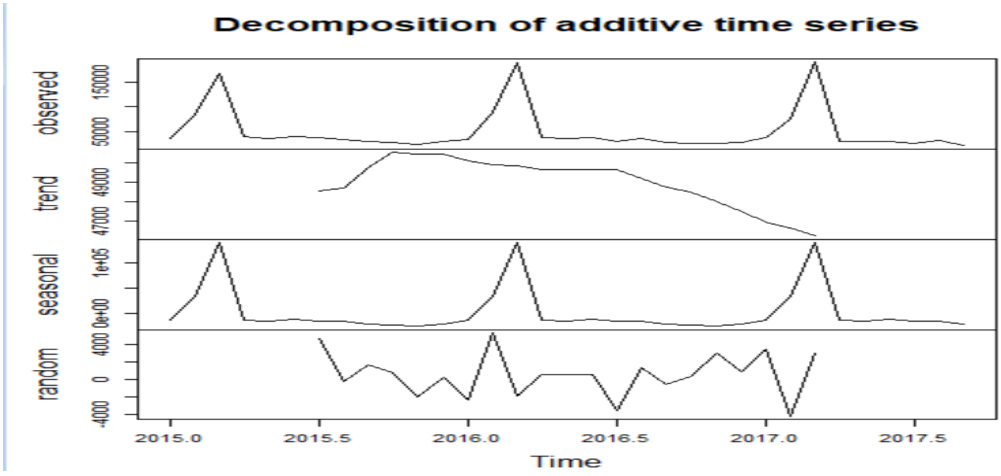


Figure (M.3)

Further seen in the figure (M.4) the standard deviation analysis with and without the seasonal component.

```
> # Using Std Dev ...
> sd(appcnt_ts)
[1] 45143.3
>
> sd(appcnt_ts-appcnt_ts_dec$seasonal)
[1] 5046.059
```

Figure (M.4)

Time Series Model Building

Simple MA model

As an initiation we are illustrating a Simple moving average model. The figure (M.5) shows the same.

```
> # Part 1: Using a Simple Moving Average to model the number of LCA $
>
> app_ma1=ma(appcnt_ts,order=2,centre=FALSE)
> app_ma1
        Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep
2015  59679 125603 103371  37325  37139  38494  35299  31786  29359
2016  61341 138274 112973  36413  36894  34592  32779  31234  26493
2017  56239 132442 109937  30522  30669  28283  28778  26575     NA
        Oct    Nov    Dec
2015  25804  26601  32035
2016  25793  26678  32174
2017
>
> app_ma2=ma(appcnt_ts,order=5,centre=FALSE)
> app_ma2
       Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct
2015    NA    NA 72218 73028 63744 36907 35110 33720 30576 28963
2016 72836 75653 76761 77596 66061 35399 33366 31430 28926 28265
2017 71063 71924 72601 71322 61418 29857 28074    NA    NA
       Nov   Dec
2015 29270 40785
2016 28683 38267
2017
>
> plot(appcnt_ts,main="No of LCA applications")
> lines(app_ma1,col='red')
> lines(app_ma2,col='green')
```

Figure (M.5)

Moving average illustrates more precision when the moving average order is less. This is because the number of LCA applications can be better predicted when the averaging component is referenced with number of applications in the preceding 2 months (Seen in Redline) as seen in figure (M.6)

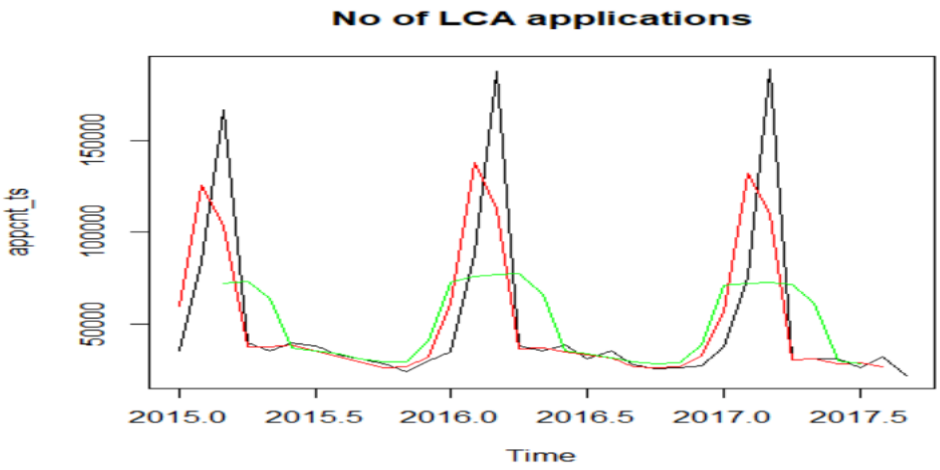


Figure (M.6)

We now forecast using simple MA models. We see the actual time series data as seen in figure (M.7) versus the forecasted values of MA order 2 in figure (M.8) and of MA order 5 in figure (M.9)

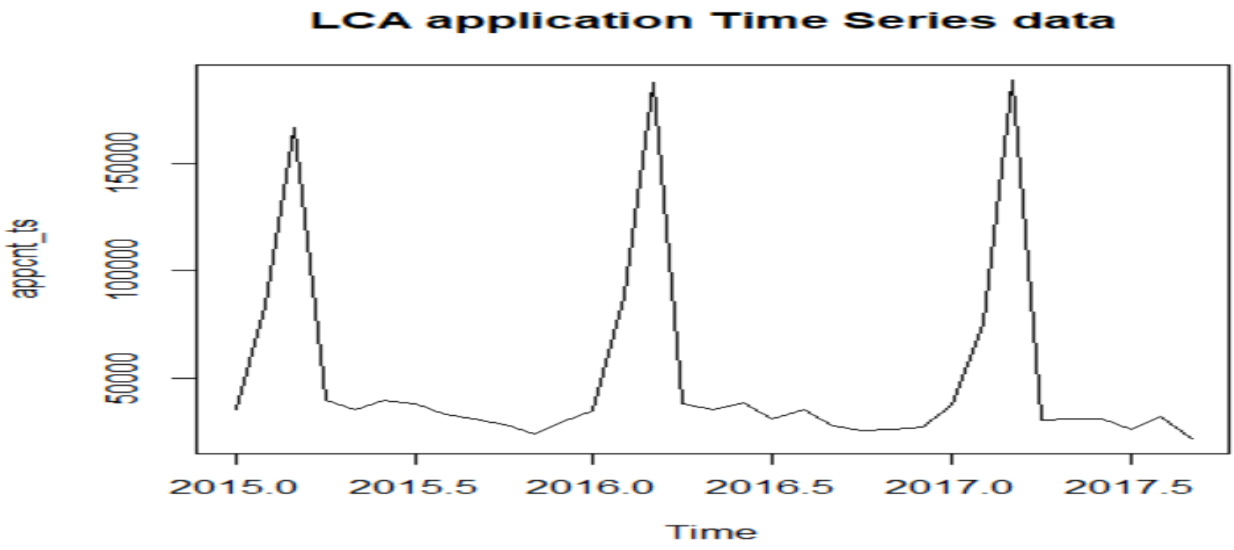


Figure (M.7)

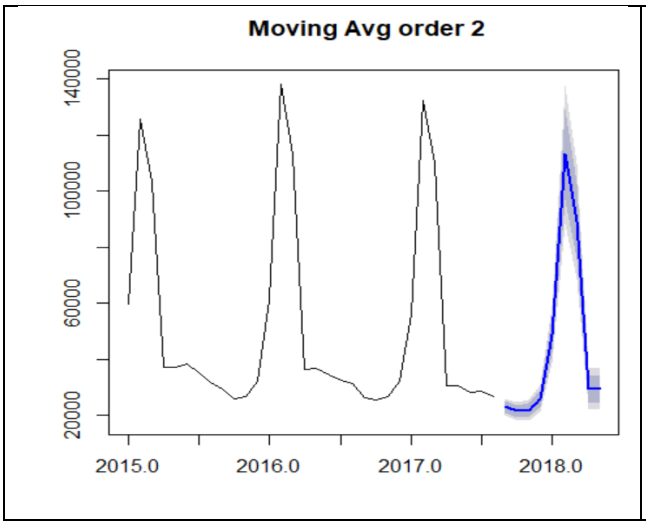


Figure (M.8)

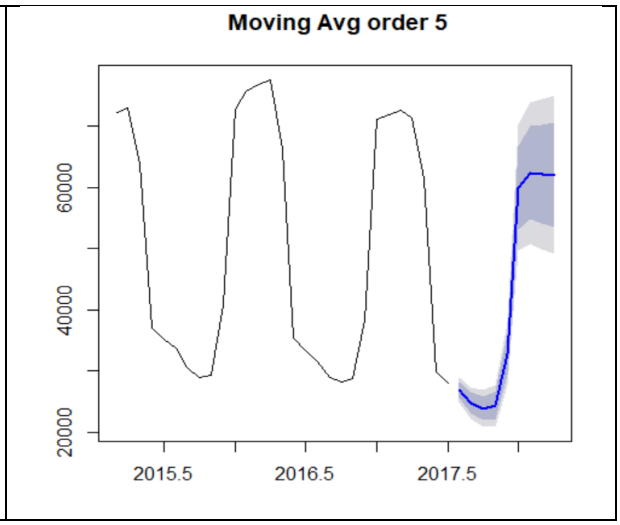


Figure (M.9)

We do see that the moving average model with MA order 2 resembles the historic trait and thus can be considered better model. The forecasted values for both order 2 and order 5 can be seen in the figure (M.10) below

```
> app_ma1_fc
          Point Forecast Lo 80  Hi 80 Lo 95  Hi 95
Sep 2017             23095 21123  25067 20079  26111
Oct 2017             21655 19502  23809 18362  24949
Nov 2017             21899 19451  24346 18155  25642
Dec 2017             26040 22840  29240 21146  30934
Jan 2018             48838 42339  55338 38898  58778
Feb 2018            113108 96982 129234 88446 137770
Mar 2018             89280 75756 102803 68597 109963
Apr 2018             29565 24837  34292 22335  36795
May 2018             29600 24629  34570 21998  37201
> app_ma2_fc
          Point Forecast Lo 80 Hi 80 Lo 95 Hi 95
Aug 2017             26900 25536 28264 24814 28986
Sep 2017             24772 23096 26449 22208 27336
Oct 2017             23865 21927 25802 20901 26828
Nov 2017             24246 21996 26495 20805 27686
Dec 2017             32692 29322 36062 27538 37845
Jan 2018             59933 53194 66672 49626 70240
Feb 2018             62381 54826 69935 50827 73934
Mar 2018             62118 54092 70143 49844 74391
Apr 2018             62011 53527 70495 49036 74986
```

Figure (M.10)

From the decomposed time series data, we see high seasonality and slight trend. There is presence of random noise as expected as seen in the figure (M.11). Because of the seasonality component we would opt for a Holt Winters model.

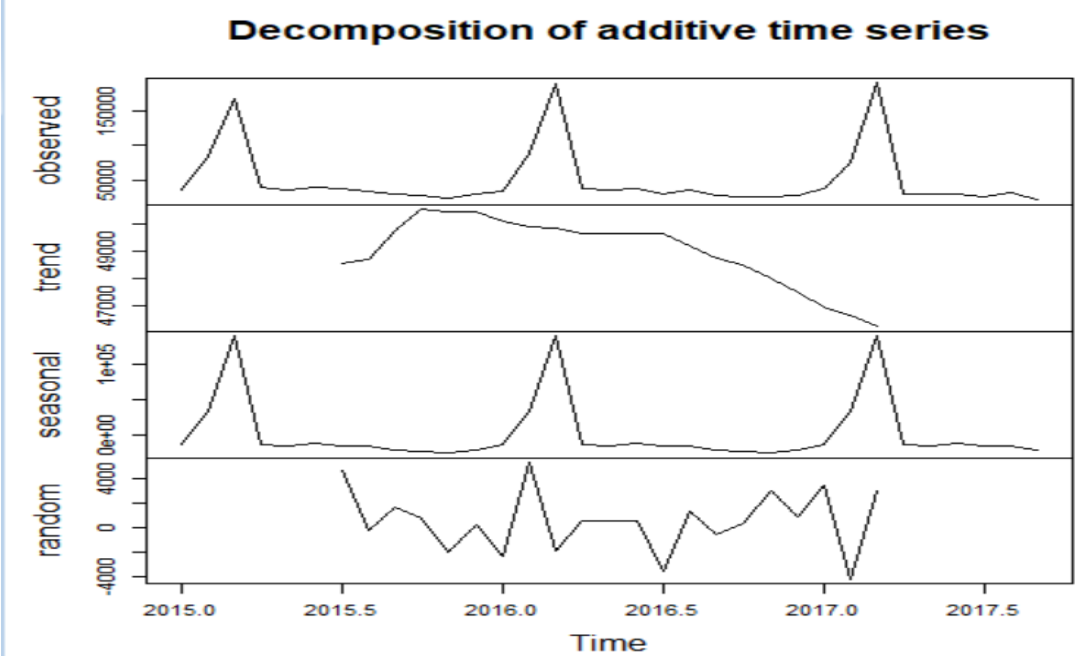


Figure (M.11)

Holt Winters model

The Holt winters models indicate that there is a strong trend component in the data from the beta value of 1. Further the large gamma value is indicating of strong seasonal component which indicate that the recent effects have strong influence in the model as seen in figure (M.12)

```
> #Using HoltWinters to predict and access the model
> par(mfrow=c(1,1))
> hwm=HoltWinters(appcnt_ts)
> plot(hwm)
> hwm
Holt-Winters exponential smoothing with trend and additive seasonal c$

Call:
HoltWinters(x = appcnt_ts)

Smoothing parameters:
 alpha: 0.05897
 beta : 1
 gamma: 0.9669
```

Figure (M.12)

We now forecast using Holt Winters. Constant variance is not observed in the model. Spikes in the residuals can most likely be accounted towards the seasonal spikes in the actual data as seen in figure (M.13) and (M.14)

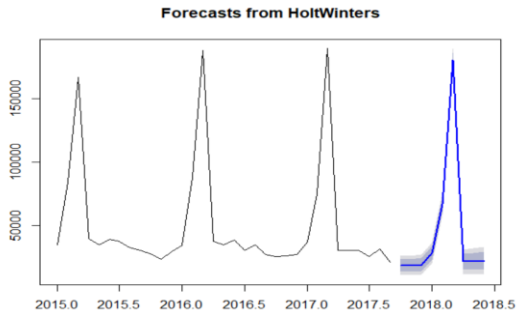


Figure (M.13)

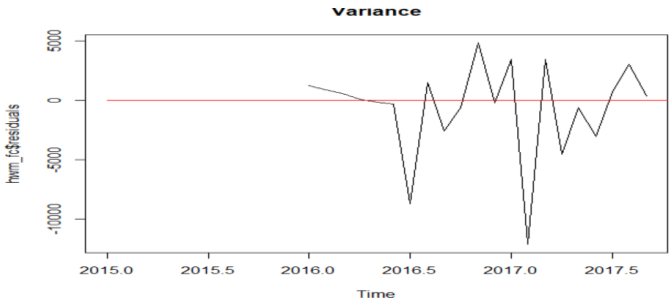


Figure (M.14)

Although normal distribution aspect is not complied, concentration of most errors closer to 0 is implicitly good enough amongst the models considered as seen in figure (M.15)

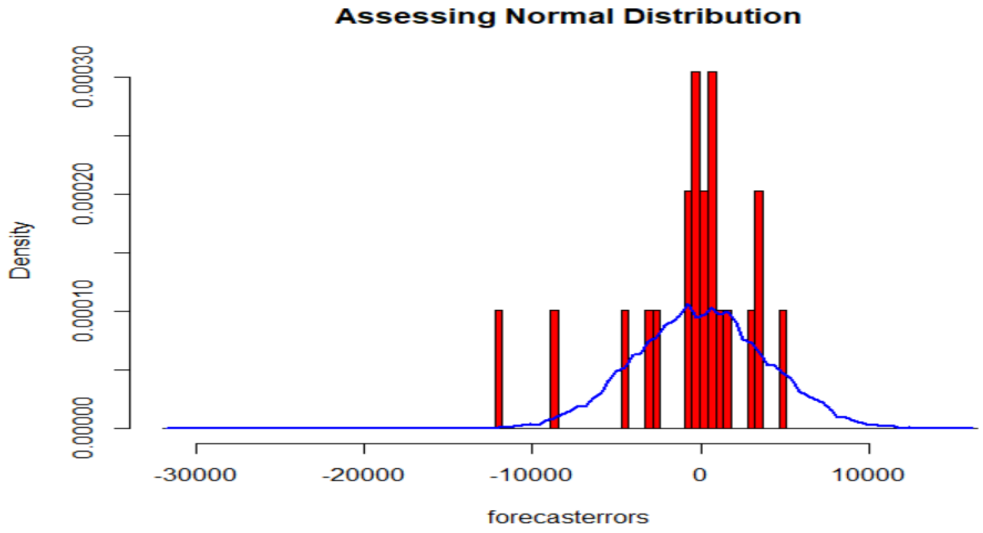


Figure (M.15)

Comparisons between the Manual holt winters trend exp smoothing and the Holt Winters models in figures (M.16) suggest that the Holt Winters suggested model is better in terms of Errors especially, Mean square errors in figure (M.17)

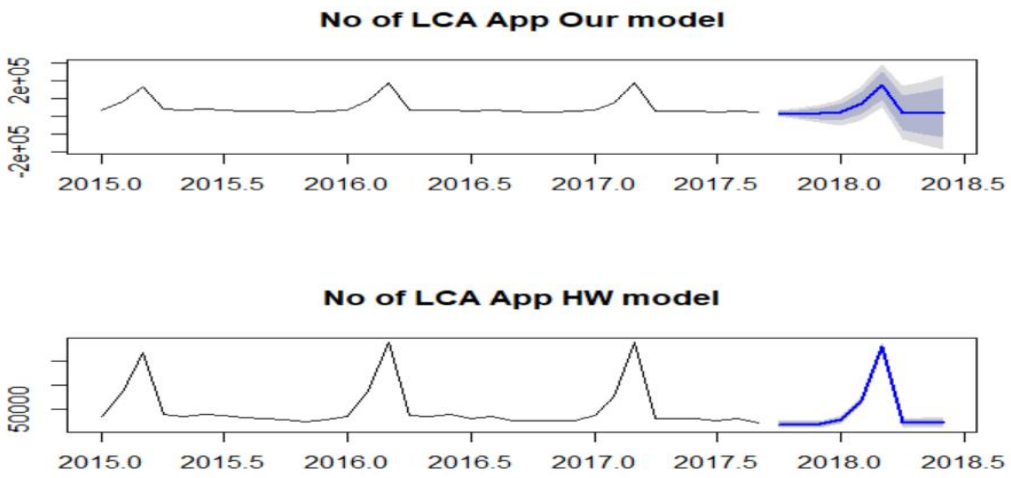


Figure (M.16)

```
> accuracy(hw_l_fc)
                    ME RMSE  MAE    MPE  MAPE  MASE    ACF1
Training set -82.25 8673 6322 -2.637 16.67 1.338 -0.7603
> accuracy(hwm_fc)
                    ME RMSE  MAE    MPE  MAPE   MASE    ACF1
Training set -592.7 3917 2527 -1.486 6.588 0.5346 -0.3773
```

Figure (M.17)

ARIMA model

We have already seen before that the LCA application data exhibit high seasonality. The LCA application count explained by time is not significant which is indicative of lack of trend component as seen in figure (M.18)

```
> appcnt_trend=appcnt_ts-appcnt_ts_dec$seasonal
> appcnt_trend_ts=data.frame(trend=c(appcnt_trend),time=c(time(appc$
> class(appcnt_trend_ts)
[1] "data.frame"
> appcnt_trend_reg=lm(appcnt_trend_ts$trend ~ appcnt_trend_ts$time)
> summary(appcnt_trend_reg)

Call:
lm(formula = appcnt_trend_ts$trend ~ appcnt_trend_ts$time)

Residuals:
   Min     1Q Median     3Q    Max
-22807   -826   1248   2470   6961

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          2947050    2207236    1.34     0.19
appcnt_trend_ts$time    -1438       1095   -1.31     0.20

Residual standard error: 4990 on 31 degrees of freedom
Multiple R-squared:  0.0527,    Adjusted R-squared:  0.0222
F-statistic: 1.73 on 1 and 31 DF,  p-value: 0.199
```

Figure (M.18)

Variation in the mean and variance values when comparing the series with and without the trend component is as seen below figure (M.19). This indicate that the trend component is very much influencing on the stationarity aspect.

```
> #---------------------------------------
> # Compare the means of the time series
> # with and without the trend
> #=======================================
> #Remove season; assess mean for trend data
> appcnt_trend1=appcnt_ts-appcnt_ts_dec$seasonal
> mean(appcnt_trend1)
[1] 47899
> var(appcnt_trend1)
[1] 25462712
>
> #Remove trend and season; assess mean for data without trend
> appcnt_ts_simple= appcnt_ts-appcnt_ts_dec$seasonal - appcnt_ts_dec$
> mean(na.omit(appcnt_ts_simple))
[1] 541.3
> var(na.omit(appcnt_ts_simple))
[1] 6218911
```

Figure (M.19)

The adf test indicate p value of 0.6 which fails to reject the null hypothesis of non-stationarity. kpss test indicate p value of 0.1 which fails to reject null hypothesis of stationarity. This conflicting result is indicative of a slight non-stationarity component in the LCA app data series.

Results from regression and kpss test indicate that the data appear to be largely stationary. The adf test may indicate only of residual non-stationary components as seen in figure (M.20) below

```
> adf.test(appcnt_trend1, k = 5, alternative = "stationary")

        Augmented Dickey-Fuller Test

data:  appcnt_trend1
Dickey-Fuller = -2, Lag order = 5, p-value = 0.6
alternative hypothesis: stationary

> #### Result: Not significant, indicating non-stationarity
>
> #================================================
> # Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test
> #================================================
> kpss.test(appcnt_trend1,lshort=FALSE)

        KPSS Test for Level Stationarity

data:  appcnt_trend1
KPSS Level = 0.28, Truncation lag parameter = 4, p-value =
0.1

Warning message:
In kpss.test(appcnt_trend1, lshort = FALSE) :
  p-value greater than printed p-value
>
> #kpss.test(appcnt_trend1,lshort=TRUE)
> #### Result: Significant, indicating non-stationarity
>
```

Figure (M.20)

The ACF functions clearly indicate a seasonal component which is especially relevant at every 12^{th} lag as seen in figure (M.21)

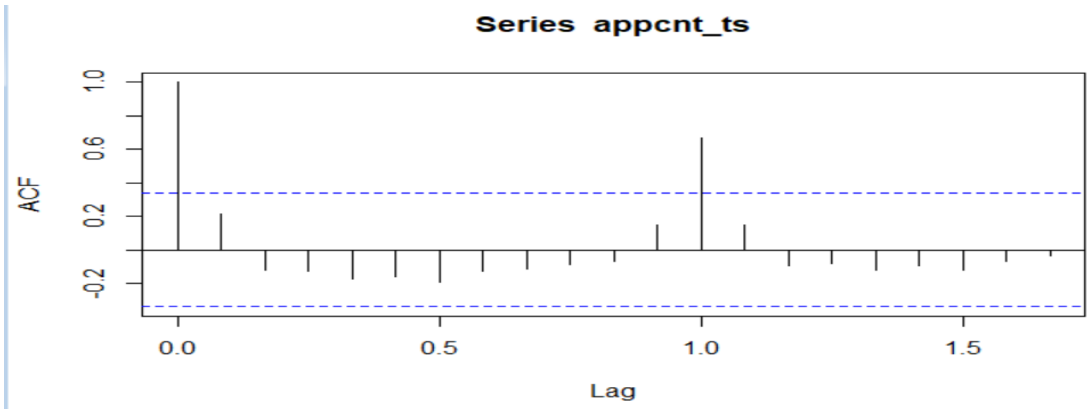


Figure (M.21)

Next, we would be removing seasonal component and also do differencing to account for the non-stationarity. The Stationarity assessment test after 1 differencing indicate that the data does follow stationarity as seen in figure (M.22) below which captures results of ADF and Kpss test.

```
> # Augmented Dickey-Fuller (ADF) t-test
> #=======================================
> adf.test(appcnt_diff1, k = 3, alternative = "stationary")

        Augmented Dickey-Fuller Test

data:  appcnt_diff1
Dickey-Fuller = -4.7, Lag order = 3, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(appcnt_diff1, k = 3, alternative = "stationary") :
  p-value smaller than printed p-value
> #### Result: Not significant, indicating non-stationarity
>
> #================================================
> # Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test
> #================================================
> kpss.test(appcnt_diff1,lshort=FALSE)

        KPSS Test for Level Stationarity

data:  appcnt_diff1
KPSS Level = 0.12, Truncation lag parameter = 4, p-value =
0.1

Warning message:
In kpss.test(appcnt_diff1, lshort = FALSE) :
  p-value greater than printed p-value
>
> #kpss.test(appcnt_trend,lshort=TRUE)
> #### Result: Significant, indicating non-stationarity
>
```

Figure (M.22)

We see from the ACF that the data does not possess seasonal or trend components. The ACF indicates strong Moving Average component and presence of autoregressive component as well. The gradual decline of PACF is again indicative of strong MA component as seen in the figure (M.23) below.

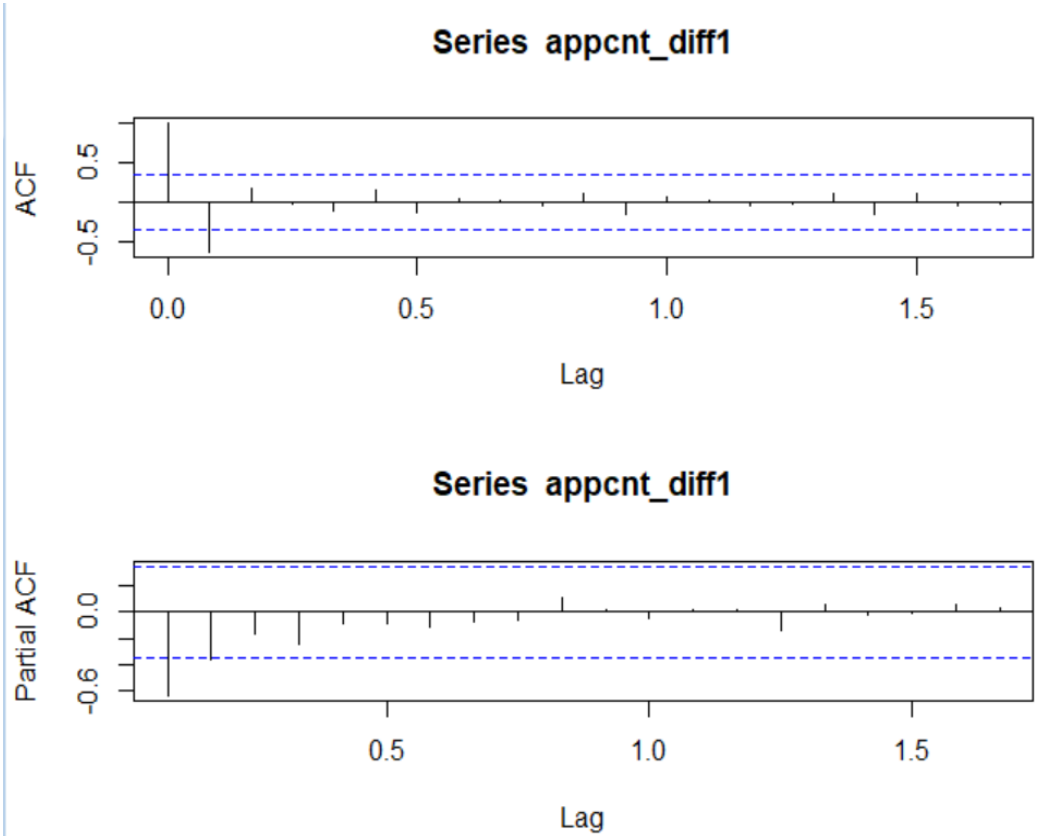


Figure (M.23)

We now propose the ARIMA models as below.

For ARIMA(011) as seen in figure (M.24) below

```
> #ARIMA Models
> #011
> appcnt_m1= Arima(appcnt_trend, order = c(0, 1, 1), method = "ML")
> appcnt_m1
Series: appcnt_trend
ARIMA(0,1,1)

Coefficients:
          ma1
      -0.830
s.e.   0.115

sigma^2 estimated as 27076392:  log likelihood=-319.3
AIC=642.6   AICc=643   BIC=645.5
> coeftest(appcnt_m1)

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ma1   -0.830      0.115   -7.23  4.7e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure (M.24)

For ARIMA(110) as seen in figure (M.25) below

```
> appcnt_m2= Arima(appcnt_trend, order = c(1, 1, 0), method = "ML")
> appcnt_m2
Series: appcnt_trend
ARIMA(1,1,0)

Coefficients:
          ar1
      -0.623
s.e.   0.133

sigma^2 estimated as 32797302:  log likelihood=-322
AIC=648.1   AICc=648.5   BIC=651
> coeftest(appcnt_m2)

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1   -0.623      0.133   -4.68  2.9e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure (M.25)

For ARIMA(111) as seen in figure (M.26) below

```
> appcnt_m3= Arima(appcnt_trend, order = c(1, 1, 1), method = "ML")
> appcnt_m3
Series: appcnt_trend
ARIMA(1,1,1)

Coefficients:
          ar1      ma1
      -0.303  -0.701
s.e.   0.199   0.155

sigma^2 estimated as 26277356:  log likelihood=-318.3
AIC=642.6   AICc=643.5   BIC=647
> coeftest(appcnt_m3)

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1   -0.303      0.199   -1.52     0.13
ma1   -0.701      0.155   -4.53  5.9e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure (M.26)

We have selected ARIMA(011) being the components with most significance and lowest AIC value. Here we observe that the same model has the least BIC value which is indicative of best fit model.

Among the 3 models ARIMA(011) has low errors as seen in figure (M.27)

```
> fc1=forecast(appcnt_m1,h=9)
> accuracy(fc1)
                    ME RMSE  MAE    MPE  MAPE   MASE    ACF1
Training set -333.4 5043 3297 -2.174 7.911 0.6976 -0.1898
>
> fc2=forecast(appcnt_m2,h=9)
> accuracy(fc2)
                    ME RMSE  MAE   MPE  MAPE   MASE    ACF1
Training set -233.7 5551 3292 -1.94 7.943 0.6966 -0.2276
>
> fc3=forecast(appcnt_m3,h=9)
> accuracy(fc3)
                ME RMSE  MAE    MPE MAPE  MASE    ACF1
Training set -369 4888 3067 -2.176 7.42 0.649 0.01093
>
```

Figure (M.27)

We run final forecasts from the ARIMA models as seen in the below figure (M.27)

```
> # Forecasts
> #ARIMA(011)
> fc1=forecast(appcnt_m1,h=9)
> accuracy(fc1)
                    ME RMSE  MAE    MPE  MAPE   MASE    ACF1
Training set -333.4 5043 3297 -2.174 7.911 0.6976 -0.1898
> fc1$mean
       Jan   Feb   Mar   Apr   May   Jun Jul Aug Sep   Oct   Nov
2017                                                 45169 45169
2018 45169 45169 45169 45169 45169 45169
       Dec
2017 45169
2018
> |
```

```
> #ARIMA(110)
> fc2=forecast(appcnt_m2,h=9)
> accuracy(fc2)
                   ME RMSE  MAE   MPE  MAPE   MASE    ACF1
Training set -233.7 5551 3292 -1.94 7.943 0.6966 -0.2276
> fc2$mean
       Jan   Feb   Mar   Apr   May   Jun Jul Aug Sep   Oct   Nov
2017                                                 45317 43354
2018 43815 44290 43994 44179 44063 44135
       Dec
2017 44578
2018
```

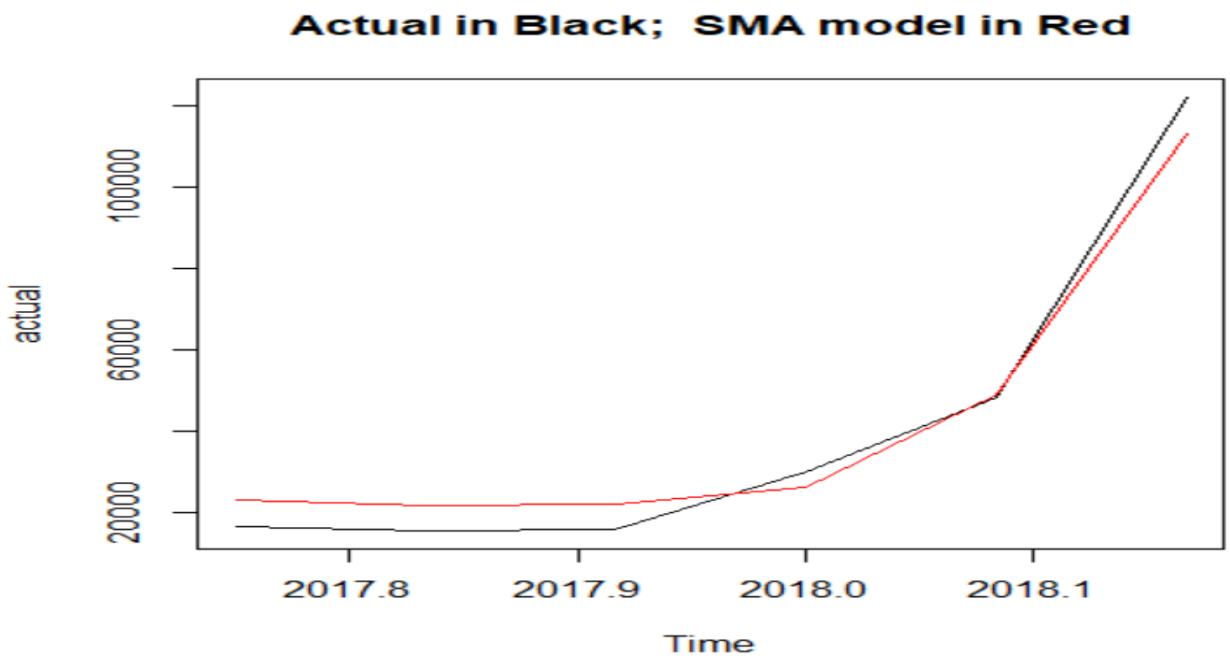
```
> #ARIMA(111)
> fc3=forecast(appcnt_m3,h=9)
> accuracy(fc3)
                ME RMSE  MAE    MPE MAPE  MASE    ACF1
Training set -369 4888 3067 -2.176 7.42 0.649 0.01093
> fc3$mean
       Jan   Feb   Mar   Apr   May   Jun Jul Aug Sep   Oct   Nov
2017                                                 44920 44084
2018 44261 44284 44277 44279 44279 44279
       Dec
2017 44338
2018
```

Figure (M.27)

Assessing Time series Forecast with Actual data

The forecasted values from above mentioned models were compared tantamount with that of original published data in the period Oct 2017 to March 2018. The accuracy results for each model can be seen as below.

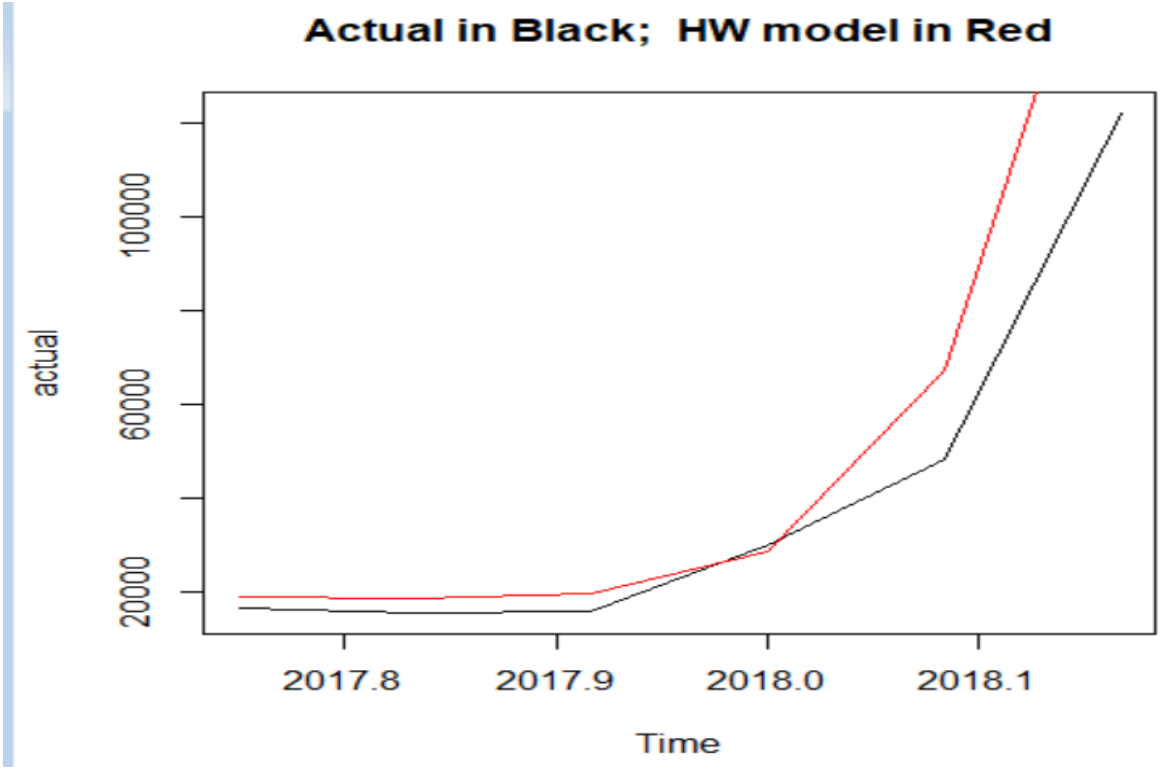
We see the accuracy of the Simple Moving average model in figure (M.28)




```
> # For Accuracy Assessment with actual data
> actual=ts(actual_sep17_2_March18$APP_COUNT,frequency=12,start=c(2017,10))
> actual
        Jan    Feb    Mar Apr May Jun Jul Aug Sep    Oct    Nov    Dec
2017                                                16595  15393  15860
2018  29986  48113 122306
> #SMA Sep 17 to March 18
> temp_SMA_MA2=window(newappmal,Start=c(2017,10),End=c(2018,3),Frequency=12)
> temp_SMA_MA2
        Jan    Feb    Mar Apr May Jun Jul Aug Sep    Oct    Nov    Dec
2017                                                23095  21655  21899
2018  26040  48838 113108
> accuracy(actual,temp_SMA_MA2)
             ME RMSE  MAE   MPE  MAPE   ACF1 Theil's U
Test set 1064 6036 5445 10.47 18.23 0.1622    0.2944
```

Figure (M.28)

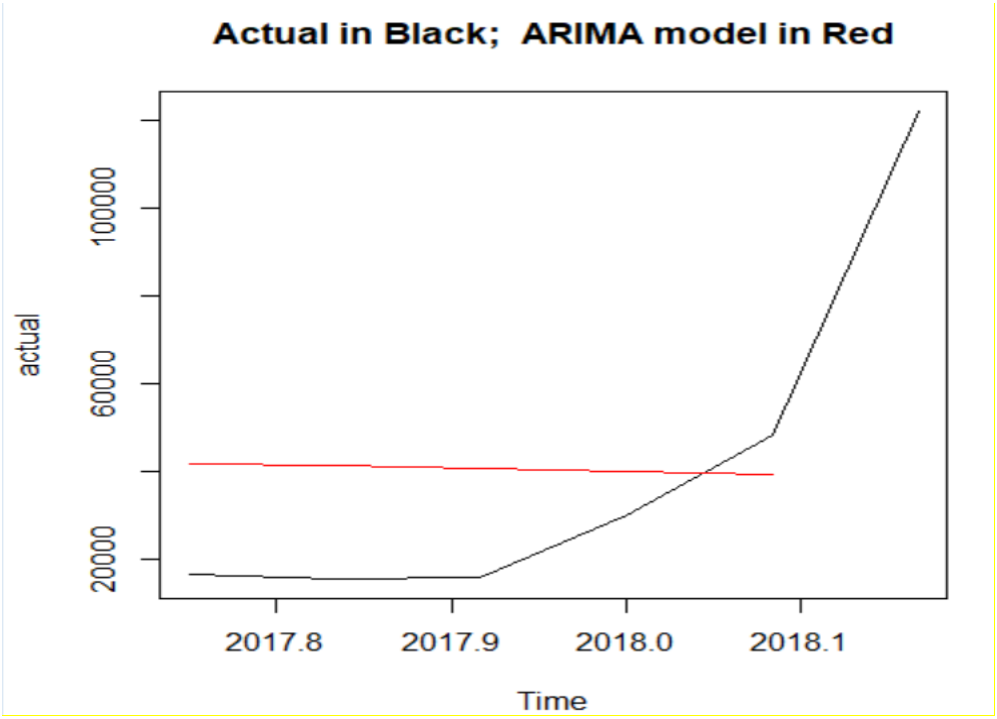
We see the accuracy of the Holt Winters model in figure (M.29)




```
> # Holt Winters Sep 17 to March 18
> hwm=HoltWinters(appcnt_ts)
> hwm_fc = forecast(hwm,h=6)
> plot(hwm_fc)
> hwm_fc$mean
        Jan    Feb    Mar Apr May Jun Jul Aug Sep    Oct    Nov    Dec
2017                                                18819  18684  19491
2018  28442  67225 180590
> temp_hw=window(hwm_fc$mean,Start=c(2017,10),End=c(2018,3),Frequency=12)
> accuracy(actual,temp_hw)
             ME  RMSE   MAE   MPE  MAPE   ACF1 Theil's U
Test set 14167 25145 14681 17.22 19.03 0.2126    0.5098
```

Figure (M.29)

We see the accuracy of the ARIMA model in figure (M.30)




```
                ME  RMSE   MAE   MPE  MAPE   ACF1 Theil's U
Test set 15410 20451 18904 37.35 46.22 0.3091     30.83
> plot(actual, main="Actual in Black;  ARIMA model in Red")
> lines(temp_ar,col='red')
```

Figure (M.30)

Number of LCA applications Conclusion

Among the 3 methods, the simple moving average yielded best results in comparison to others. This illustrates the fact that number of LCA application time series data has less of auto-regressive component and has less of correlation with previous time trends Hence, MA is more relevant.

Forecasting the average wage offered to Programmer analyst between Oct 2017 to May 2018

Data Extraction for time series

We have transformed and extracted the data for the wage offered as detailed earlier and converted the same to time series data as seen in figure (M.31)

```
> #### like PROGRAMMER ANALYST
> pgm_df=sqldf('select CASE_SUBMITTED,round(avg(WAGE_RATE_OF_PAY_FROM),0)
>
> head(pgm_df)
  CASE_SUBMITTED AVG_WAGE
1       Jul 2011    60000
2       Sep 2011    61932
3       Oct 2011    71267
4       Nov 2011    59636
5       Dec 2011    62679
6       Jan 2012    74852
>
```

figure (M.31)

We have extracted the values from year 2015 for continuous time series data as seen in figure (M.32)

```
> df1=pgm_df[6:74,]
> pgmts=ts(df1$AVG_WAGE,frequency=12,start=c(2012,1))
> head(pgmts)
       Jan   Feb   Mar   Apr   May   Jun
2012 74852 70367 62258 63002 63976 74610
>
> pgmts
       Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
2012 74852 70367 62258 63002 63976 74610 63299 67983 66510 74287 69697 66794
2013 63170 65277 61263 70782 74266 68782 69391 66967 68777 68915 68158 67340
2014 68273 65877 62475 67293 67530 67180 67013 70386 68145 69213 69811 68959
2015 68878 66705 63695 69489 70791 70928 71221 71268 71008 71147 71437 70033
2016 70621 67139 63956 70924 71457 71796 74212 73306 72103 73117 72456 71339
2017 71222 69894 66654 76166 78673 77000 79601 84206 81668
>
```

figure (M.32)

Seasonality Assessment and Decomposition

Here we can see some seasonality with wage of these programmer analyst application declining in in March of every year. This is possibly because of the year end LCA application rush. We see that the data has some seasonal component from the decomposed data series as well. as seen in figures (M.33)

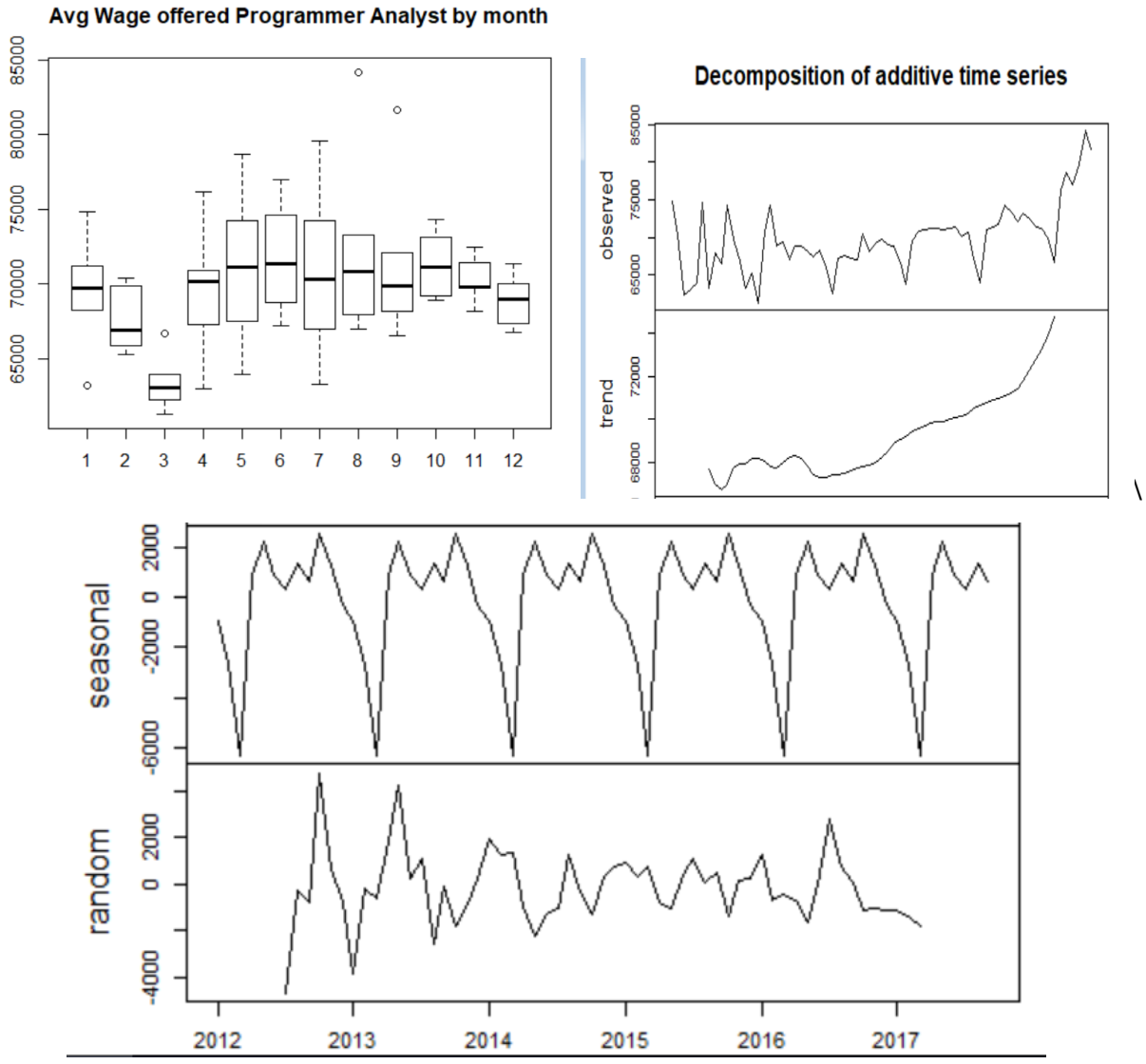


Figure (M.33)

Further seen in the standard deviation analysis with and without the seasonal component in figure (M.34)

```
> sd(pgmts)
[1] 4449
> sd(pgmts-pgmts_dec$seasonal)
[1] 3847
>
```

Figure (M.34)

Time Series Model Building

Simple MA model

As an initiation we are illustrating a Simple moving average model. The figure (M.35) shows the same.

```
> # Part 1: Using a Simple Moving Average
>
> pgm_ma1=ma(pgmts,order=2,centre=FALSE)
> pgm_ma1
      Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
2012 72610 66313 62630 63489 69293 68955 65641 67247 70399 71992 68246 64982
2013 64224 63270 66023 72524 71524 69087 68179 67872 68846 68537 67749 67807
2014 67075 64176 64884 67412 67355 67097 68700 69266 68679 69512 69385 68919
2015 67792 65200 66592 70140 70860 71075 71245 71138 71078 71292 70735 70327
2016 68880 65548 67440 71191 71627 73004 73759 72705 72610 72787 71898 71281
2017 70558 68274 71410 77420 77837 78301 81904 82937    NA
>
> pgm_ma2=ma(pgmts,order=5,centre=FALSE)
> pgm_ma2
      Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
2012    NA    NA 66891 66843 65429 66574 67276 69338 68355 69054 68092 67845
2013 65240 65457 66952 68074 68897 70038 69637 68566 68442 68031 68293 67713
2014 66425 66252 66290 66071 66298 67880 68051 68387 68914 69303 69001 68713
2015 67610 67545 67912 68322 69225 70739 71043 71114 71216 70979 70849 70075
2016 68637 68535 68819 69054 70469 72339 72575 72907 73039 72464 72047 71606
2017 70313 71055 72522 73677 75619 79129 80230    NA    NA
>
> plot(pgmts,main="Avg Wage offered Programmer Analyst")
> lines(pgm_ma1,col='red')
> lines(pgm_ma2,col='green')
>
```

Figure (M.35)

Moving average illustrates more precision when the moving average order is less. This is because the wage can be better predicted when the averaging component is referenced with recent wages in the preceding 2 months (Seen in Redline) figure (M.36)

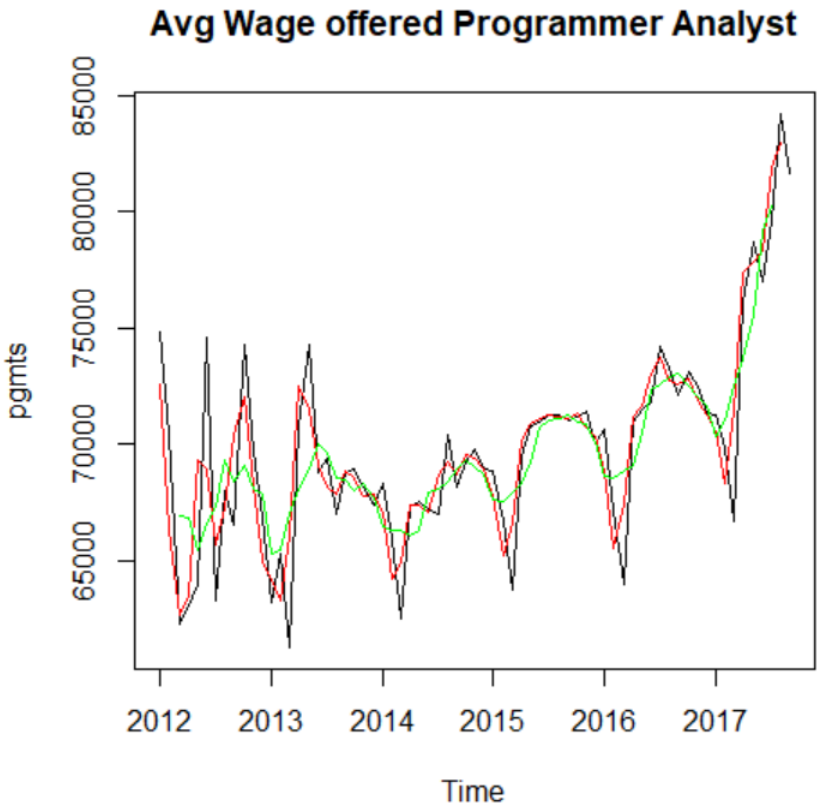


Figure (M.36)

We now forecast using simple MA models. We see the actual time series data as seen in figure (M.37) versus the forecasted values of MA order 2 in figure (M.38) and of MA order 5 in figure (M.39)

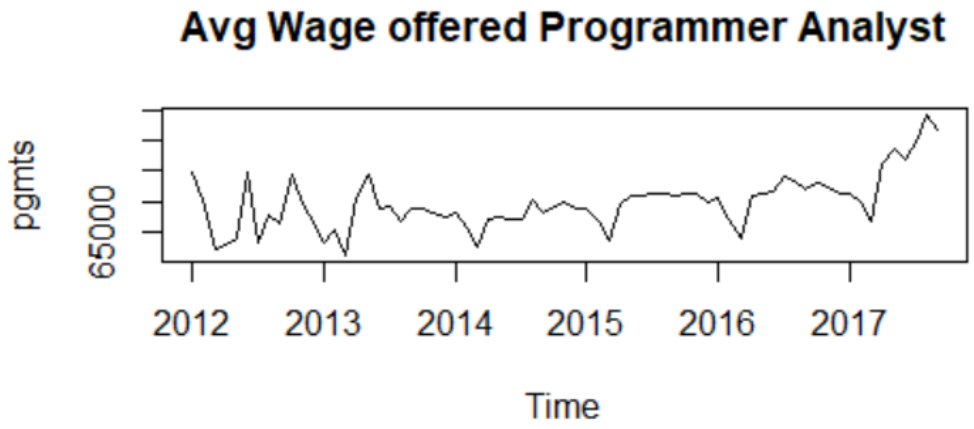


Figure (M.37)

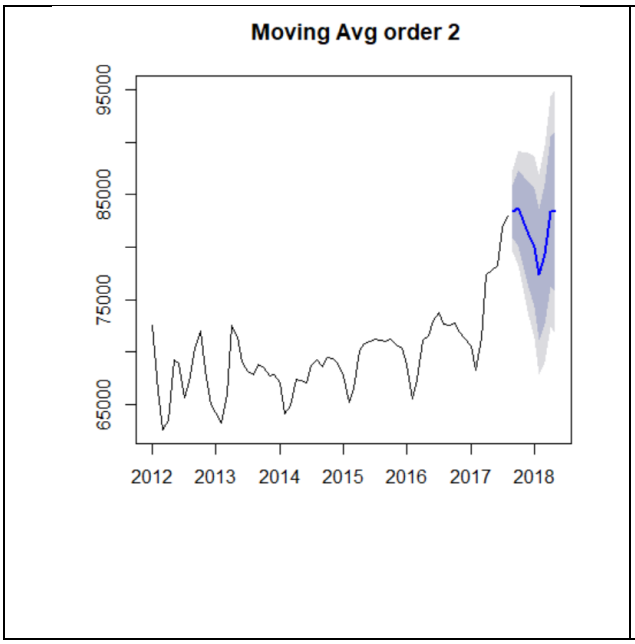


Figure (M.38)

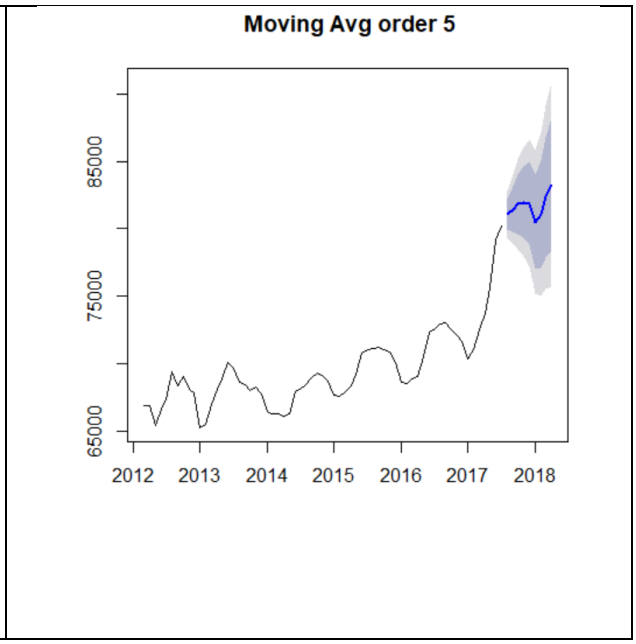


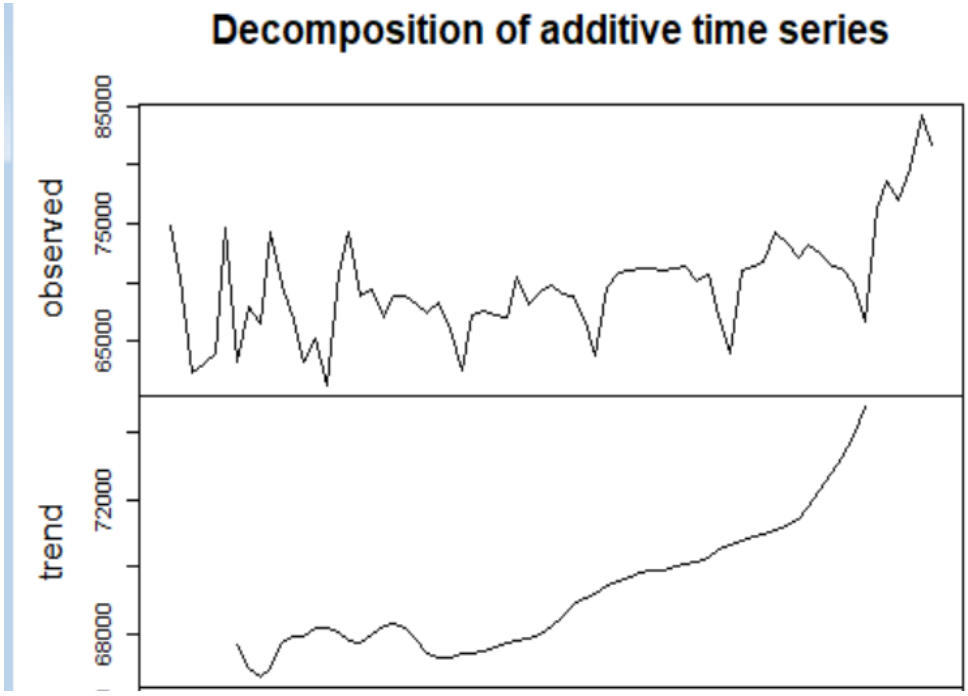
Figure (M.39)

We do see that the moving average model with MA order 2 resembles the historic trait and thus can be considered better model. The forecasted values for both order 2 and order 5 can be seen in the figure (M.40) below

```
> pgm_ma1_fc
          Point Forecast Lo 80 Hi 80 Lo 95 Hi 95
Sep 2017          83427 80893 85960 79552 87301
Oct 2017          83740 80157 87323 78261 89219
Nov 2017          82331 77943 86718 75620 89041
Dec 2017          81241 76175 86308 73493 88990
Jan 2018          79974 74309 85638 71310 88637
Feb 2018          77392 71187 83597 67902 86882
Mar 2018          79361 72658 86063 69110 89611
Apr 2018          83412 76246 90577 72453 94370
May 2018          83391 75791 90991 71768 95014
> pgm_ma2_fc
          Point Forecast Lo 80 Hi 80 Lo 95 Hi 95
Aug 2017          81064 79953 82175 79364 82763
Sep 2017          81452 79779 83126 78893 84012
Oct 2017          81809 79636 83983 78485 85133
Nov 2017          81964 79320 84609 77920 86009
Dec 2017          81884 78788 84979 77149 86618
Jan 2018          80502 77018 83986 75174 85830
Feb 2018          81073 77125 85022 75034 87112
Mar 2018          82341 77885 86796 75526 89155
Apr 2018          83269 78313 88224 75690 90847
```

Figure (M.40)

From the decomposed time series data, we see high seasonality and slight trend. There is presence of random noise as expected as seen in the figure (M.41). Because of the seasonality component we would opt for a Holt Winters model.



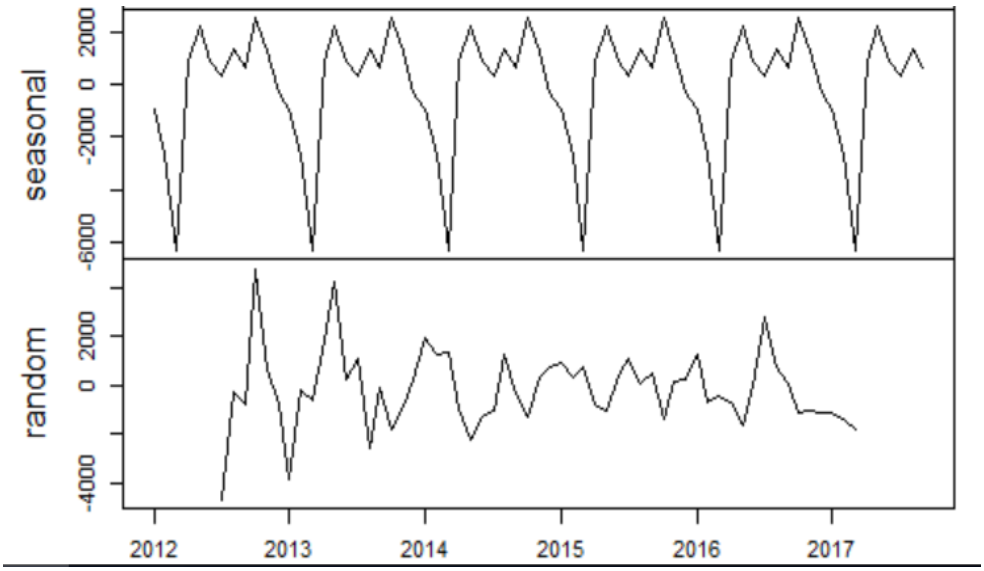


Figure (M.41)

Holt Winters model

The Holt winters models indicate that there is a negligible trend component in the data from the beta value of 0.08. Further the large gamma value is indicating of strong seasonal component which indicate that the recent effects have strong influence in the model as seen in figure (M.42).

```
> #Using HoltWinters to predict and access the model
> par(mfrow=c(1,1))
> hwm=HoltWinters(pgmts)
> plot(hwm)
> hwm
Holt-Winters exponential smoothing with trend and additive seasonal componen

Call:
HoltWinters(x = pgmts)

Smoothing parameters:
 alpha: 0.2777
 beta : 0.08933
 gamma: 0.975
```

Figure (M.42)

We now forecast using Holt Winters. Constant variance is not observed in the model. Spikes in the residuals can most likely be accounted towards the seasonal spikes in the actual data as seen in figure (M.43) and figure (M.44)

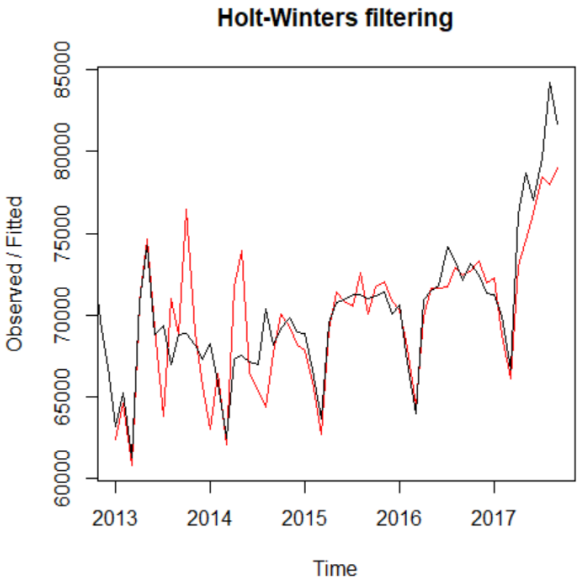


Figure (M.43)

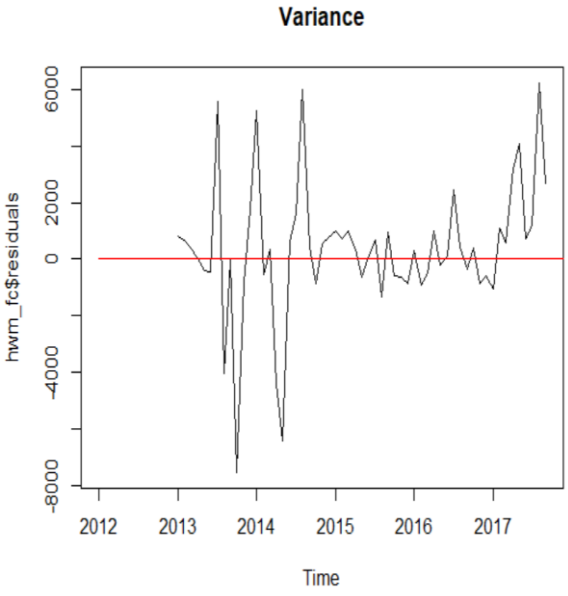


Figure (M.44)

Although normal distribution aspect is not complied, concentration of most errors closer to 0 is implicitly good enough amongst the models considered as seen in figure (M.45)

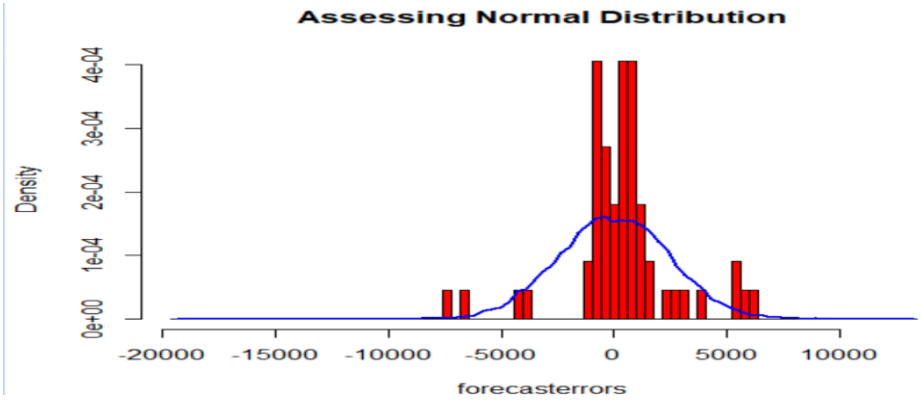


Figure (M.45)

Comparisons between the Manual holt winters trend exp smoothing and the Holt Winters models in figures (M.46) suggest that the Holt Winters suggested model is better in terms of Errors especially, Mean square errors in figure (M.47)

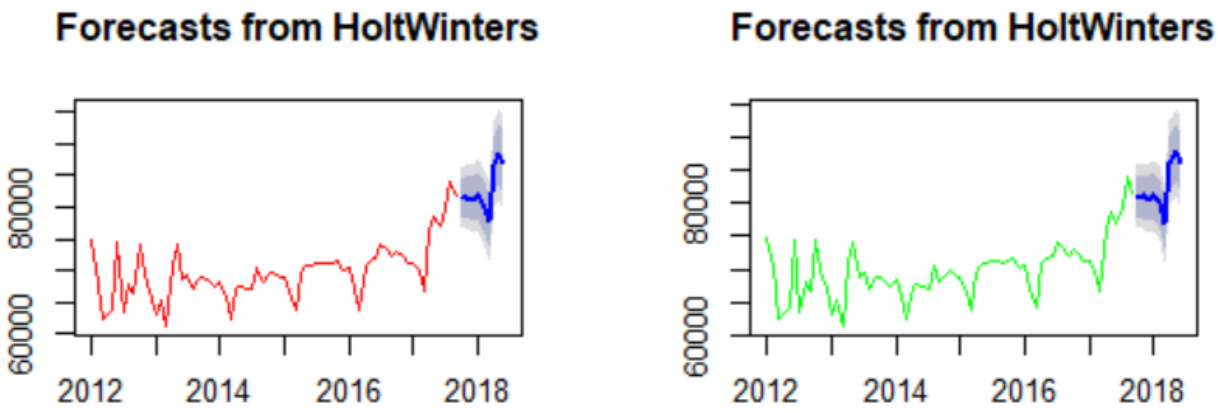


Figure (M.46)

```
> accuracy(hw_1_fc)
                    ME RMSE  MAE    MPE  MAPE   MASE   ACF1
Training set 334.3 2431 1580 0.4186 2.233 0.5085 0.1778
> accuracy(hwm_fc)
                    ME RMSE  MAE    MPE  MAPE   MASE   ACF1
Training set 354.1 2421 1548 0.4423 2.184 0.4983 0.1854
>
```

Figure (M.47)

ARIMA model

We have already seen before that the average wage data exhibit high seasonality. The average wage explained by time is significant indicative of trend component as seen in figure (M.48)

```
> #=======================================
> # Use regression to assess stationarity
> # by regressing trend onto time
> #=======================================
> pgm=pgmts-pgmts_dec$seasonal
> pgm_trend_ts=data.frame(trend=c(pgm),time=c(time(pgm)))
> class(pgmts)
[1] "ts"
> pgm_trend_reg=lm(pgm_trend_ts$trend ~ pgm_trend_ts$time)
> summary(pgm_trend_reg)

Call:
lm(formula = pgm_trend_ts$trend ~ pgm_trend_ts$time)

Residuals:
   Min     1Q Median     3Q    Max
 -4587  -1846   -680    506   9927

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       -2793115     445557   -6.27  3.0e-08 ***
pgm_trend_ts$time     1421        221    6.43  1.6e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3050 on 67 degrees of freedom
Multiple R-squared:  0.381,     Adjusted R-squared:  0.372
F-statistic: 41.3 on 1 and 67 DF,  p-value: 1.59e-08
```

Figure (M.48)

Variation in the mean and variance values when comparing the series with and without the trend component is as seen below figure (M.49). This indicate that the trend component is very much influencing on the stationarity aspect.

```
#=======================================
# Compare the means of the time series
# with and without the trend
#=======================================
#Remove season; assess mean for trend data
pgm=pgmts-pgmts_dec$seasonal
mean(pgm)
[1] 69862
var(pgm)
[1] 14801774
plot(pgm)

#Remove trend and season; assess mean for data without trend
pgm_simple= pgmts-pgmts_dec$seasonal - pgmts_dec$trend
mean(na.omit(pgm_simple))
[1] -166.4
var(na.omit(pgm_simple))
[1] 2583257
```

Figure (M.49)

Both ADF and KPSS test indicates data is not stationary as seen in figure (M.50) below

```
> #=======================================
> # Augmented Dickey-Fuller (ADF) t-test
> #=======================================
> adf.test(pgm, k = 3, alternative = "stationary")

        Augmented Dickey-Fuller Test

data:  pgm
Dickey-Fuller = -0.84, Lag order = 3, p-value = 1
alternative hypothesis: stationary

> #### Result: Not significant, indicating non-stationarity
>
> #================================================
> # Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test
> #================================================
> kpss.test(pgm,lshort=FALSE)

        KPSS Test for Level Stationarity

data:  pgm
KPSS Level = 0.83, Truncation lag parameter = 5, p-value = 0.01

Warning message:
In kpss.test(pgm, lshort = FALSE) : p-value smaller than printed p-value
>
```

Figure (M.50)

The ACF functions clearly indicate oscillatory patter indicative of a seasonal component as seen in figure (M.51)

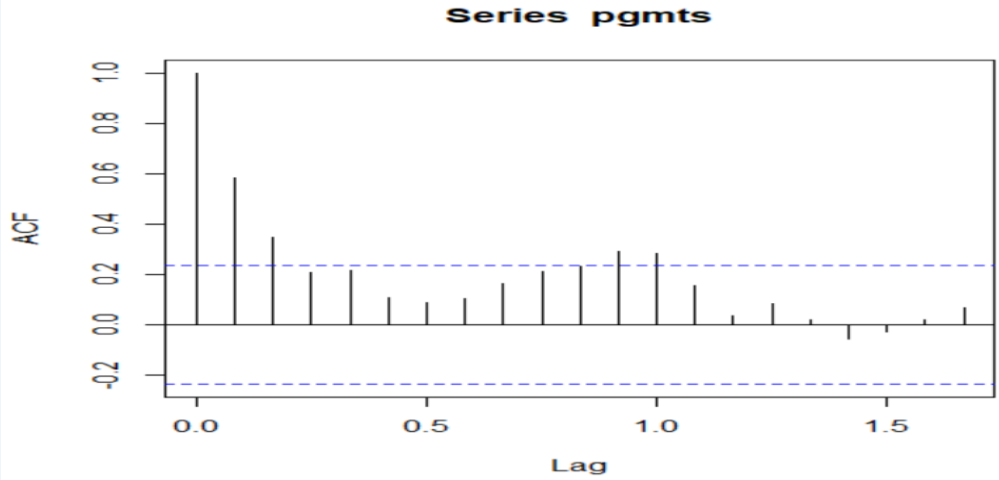


Figure (M.51)

Next, we would be removing seasonal component and also do differencing to account for the non-stationarity. The Stationarity assessment test after 1 differencing indicate that the data does follow stationarity as seen in figure (M.52) below which captures results of ADF and Kpss test.

```
> # Augmented Dickey-Fuller (ADF) t-test
> #=========================================
> adf.test(pgm_diff1, k = 3, alternative = "stationary")

        Augmented Dickey-Fuller Test

data:  pgm_diff1
Dickey-Fuller = -8.1, Lag order = 3, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(pgm_diff1, k = 3, alternative = "stationary") :
  p-value smaller than printed p-value
> #### Result: Not significant, indicating non-stationarity
>
> #=================================================
> # Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test
> #=================================================
> kpss.test(pgm_diff1,lshort=FALSE)

        KPSS Test for Level Stationarity

data:  pgm_diff1
KPSS Level = 0.37, Truncation lag parameter = 5, p-value = 0.09
```

Figure (M.52)

We see from the ACF that the data does not possess seasonal or trend components. The ACF indicates strong Moving Average component and presence of autoregressive component as well. The gradual decline of PACF is again indicative of strong MA component as seen in the figure (M.53) below.

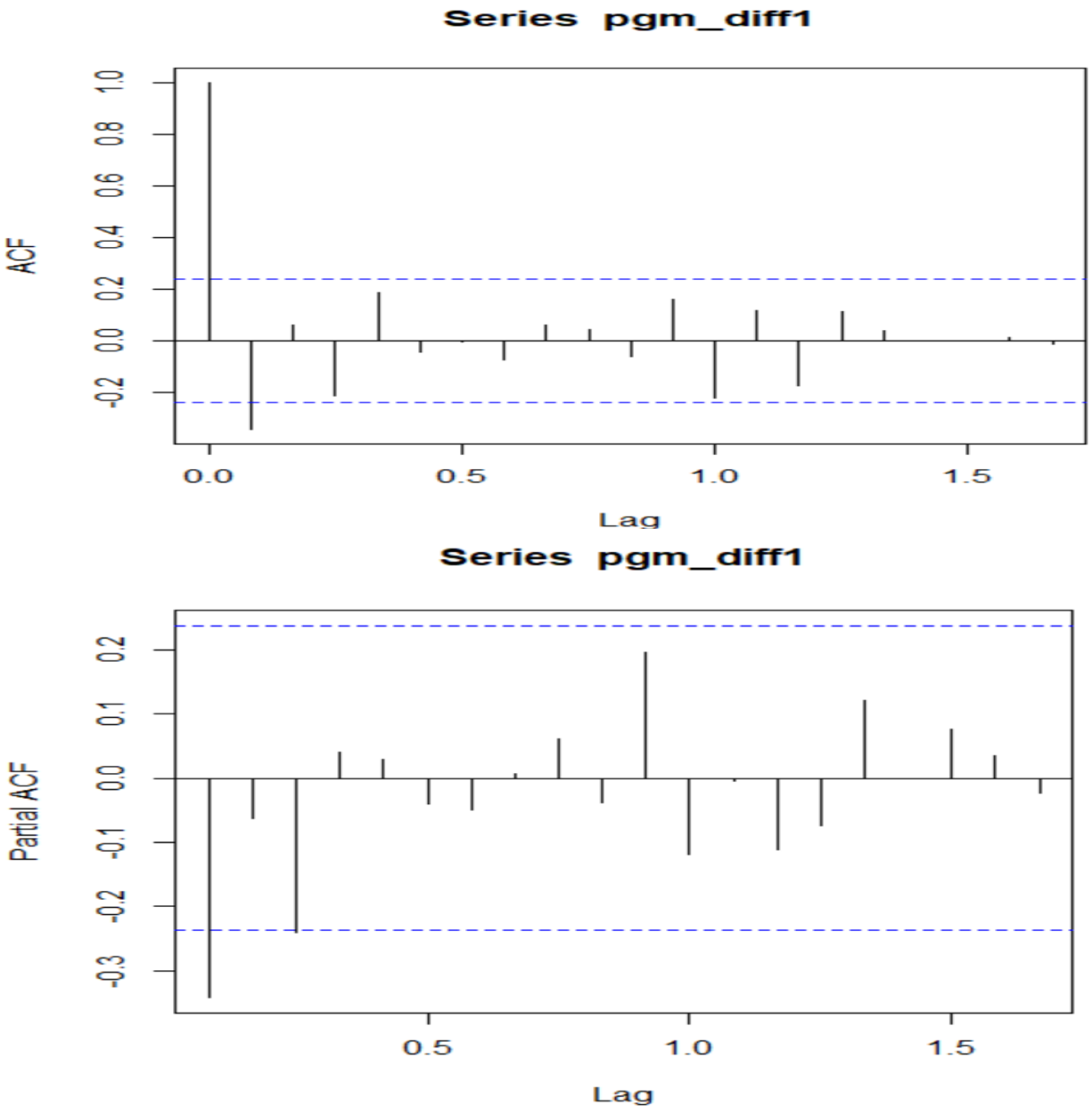


Figure (M.53)

We now propose the ARIMA models as below.

For ARIMA(011) as seen in figure (M.54) below

```
> #ARIMA Models
> #011
> appcnt_m1= Arima(appcnt_trend, order = c(0, 1, 1), method = "ML")
> appcnt_m1
Series: appcnt_trend
ARIMA(0,1,1)

Coefficients:
          ma1
      -0.830
s.e.   0.115

sigma^2 estimated as 27076392:  log likelihood=-319.3
AIC=642.6   AICc=643   BIC=645.5
> coeftest(appcnt_m1)

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ma1   -0.830      0.115   -7.23  4.7e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure (M.54)

For ARIMA(110) as seen in figure (M.55) below

```
> #110
> pgm_m2= Arima(pgmnew, order = c(1, 1, 0), method = "ML")
> pgm_m2
Series: pgmnew
ARIMA(1,1,0)

Coefficients:
         ar1
      -0.342
s.e.   0.114

sigma^2 estimated as 7344548:  log likelihood=-633.6
AIC=1271   AICc=1271   BIC=1276
> coeftest(pgm_m2)

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1   -0.342      0.114      -3   0.0027 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure (M.55)

For ARIMA (213) as seen in figure (M.56) below

```
> #213
> pgm_m5= Arima(pgmnew, order = c(2, 1, 3), method = "ML")
> pgm_m5
Series: pgmnew
ARIMA(2,1,3)

Coefficients:
          ar1      ar2     ma1     ma2      ma3
      -0.849  -0.555   0.451   0.247  -0.671
s.e.   0.171   0.139   0.169   0.150   0.141

sigma^2 estimated as 6188960:  log likelihood=-628.1
AIC=1268   AICc=1270   BIC=1281
> coeftest(pgm_m5)

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1   -0.849      0.171   -4.96  7.2e-07 ***
ar2   -0.555      0.139   -4.00  6.2e-05 ***
ma1    0.451      0.169    2.67   0.0075 **
ma2    0.247      0.150    1.65   0.0999 .
ma3   -0.671      0.141   -4.74  2.1e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure (M.56)

Following parsimony principle we had selected ARIMA(011) being the components with most significance and lowest AIC value. Here we observe that the same model has the least BIC value which is indicative of best fit model.

Among the 3 models ARIMA(011) has low errors as seen in figure (M.57)

```
> #ARIMA(011)
> accuracy(fc1)
                    ME RMSE  MAE     MPE  MAPE   MASE    ACF1
Training set 151.4 2630 1744 0.08793 2.515 0.5613 0.05578
> #ARIMA(110)
> accuracy(fc2)
                   ME RMSE  MAE     MPE  MAPE   MASE    ACF1
Training set 115 2671 1764 0.05184 2.542 0.5678 -0.0237
> #ARIMA(213)
> accuracy(fc5)
                    ME RMSE  MAE    MPE  MAPE   MASE    ACF1
Training set 175.2 2377 1588 0.1335 2.291 0.5113 0.07152
```

Figure (M.57)

We run final forecasts from the ARIMA models as seen in the below figure (M.58) and we select ARIMA(213) having had the best forecast when compared to the original data explained with RMSE forecast error as well.

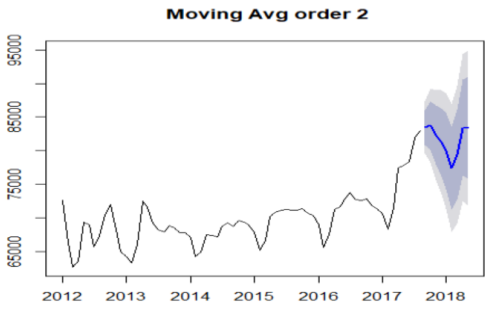
```
> #ARIMA(011)
> fc1=forecast(pgm_m1,h=9)
> fc1$mean
         Jan    Feb    Mar    Apr    May    Jun Jul Aug Sep    Oct    Nov    Dec
2017                                                          80750  80750  80750
2018 80750  80750  80750  80750  80750  80750
> #ARIMA(110)
> fc2=forecast(pgm_m2,h=9)
> fc2$mean
         Jan    Feb    Mar    Apr    May    Jun Jul Aug Sep    Oct    Nov    Dec
2017                                                          81653  81439  81512
2018 81487  81496  81493  81494  81493  81493
> #ARIMA(213)
> fc5=forecast(pgm_m5,h=9)
> fc5$mean
         Jan    Feb    Mar    Apr    May    Jun Jul Aug Sep    Oct    Nov    Dec
2017                                                          79103  78783  79604
2018 79084  79070  79370  79123  79167  79267
>
```

Figure (M.58)

Assessing Time series Forecast Accuracy

The accuracy of the forecasted values for each model is seen below.

We see the accuracy of the Simple Moving average model in figure (M.59)




```
> accuracy(pgm_ma1_fc)
                    ME RMSE  MAE    MPE  MAPE   MASE   ACF1
Training set 187 1762 1201 0.2064 1.735 0.4658 0.1284
```

Figure (M.59)

We see the accuracy of the Holt Winters model in figure (M.60)

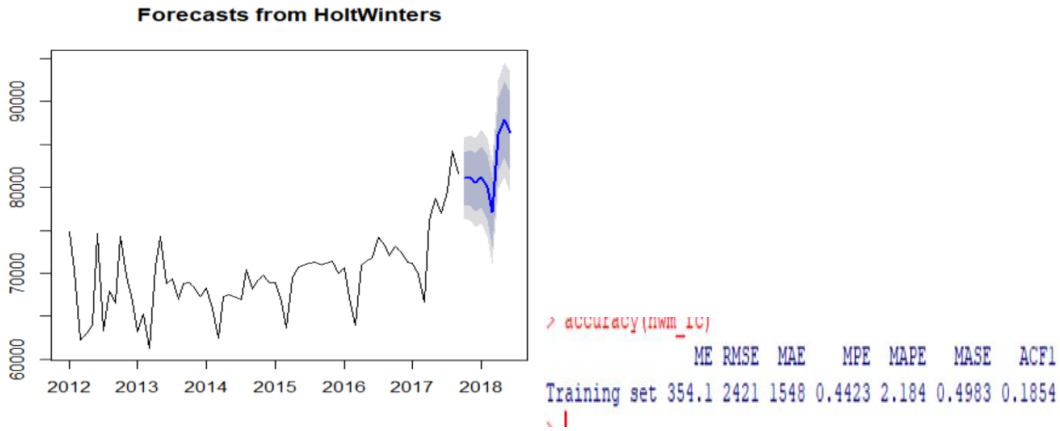


Figure (M.60)

We see the accuracy of the ARIMA model in figure (M.61)

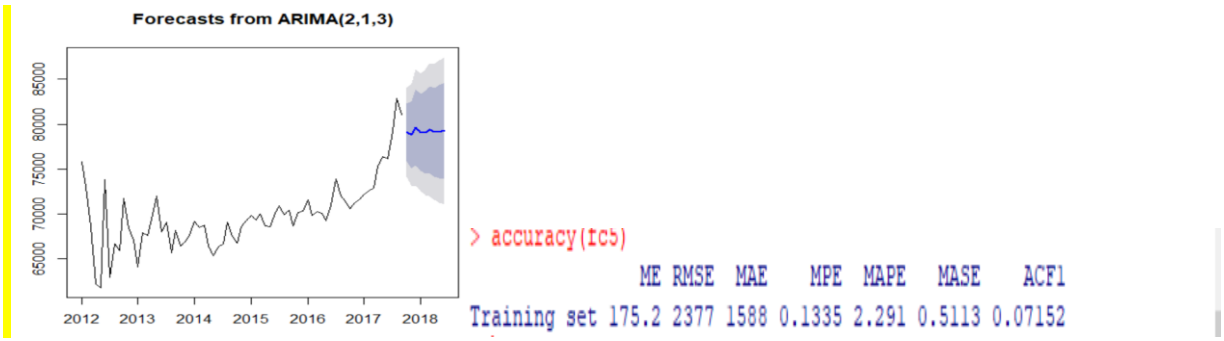


Figure (M.61)

Wage forecast Conclusion:

Among the 3 methods, the simple moving average yielded best results in comparison to others. All the three models seemed equally competitive with closer Root mean square errors. The best among them still seems to be moving average order 2 model. This again indicate strong moving average component of the data series. To conclude with the average wage offered annually for a programmer analyst for the LCA application in the timeframe between Sept 2017 to May 2018 is expected to be between 77392 USD to 83412 USD.

N. PROJECT CONCLUSION

The project objectives have been achieved as follows:

Using Descriptive Statistics we have achieved

i. Unveiling Top Jobs in market for H1B based on demand classification.
ii. Top Companies filing for LCA petitions
iii. Company wise analysis of filed LCA status

Using Logistic Regression and Classification we have achieved

i. Factors contributing to successful LCA petition filing.
ii. Predict whether LCA petition will be Approved based on factored variables.

Using Time series analysis, the below have been achieved

i. Forecast the Average Wage Offered for the top Job category by sponsors for the H1B LCA Applicants for the period Oct 2017 to May 2018
ii. Forecast the number of H1B LCA applications for the period Oct 2017 to March 2018; Compare same with actual Data.

O. REFERENCES

i. https://www.foreignlaborcert.doleta.gov/performancedata.cfm - OFLC website for data files on H1B LCA applicants