MSIN0025: DATA ANALYTICS II
GROUP ASSIGNMENT: ***TEAM B.6***

US ELECTION & BIRTHS CASE STUDY

WORD COUNT: 1998

Part I: Obama-Clinton Case Study

Section 1: The Problem

1.1 Problems

The Obama-Clinton case study describes the segmentation of their voters based on their demographic data. By understanding the vote patterns from attributes, we can generate insights and give predictions for the rest of the voting period.

1.2 Sub-problems

We are going to work for the Obama campaign to help analyse the important demographic factors that influence voting behaviour and results. The target attributes we focused on are wealth and age. They are the significant features that affect Obama's success. The major types supporting the visualisations in this report are Tableau and R code.

1.3 Attributes

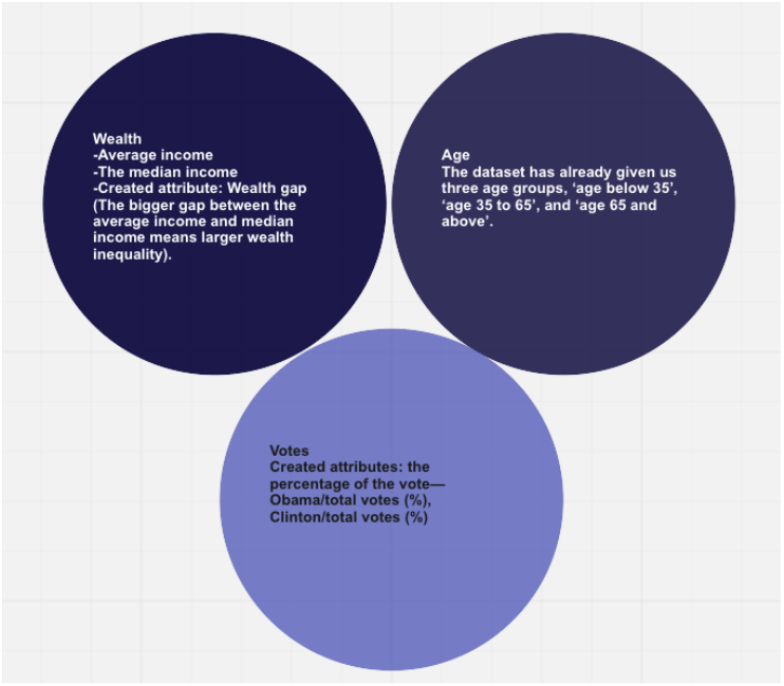


Figure1: Attributes

Section 2: Understand the Data

2.1 Nature, size and source of data

The nature of this set contains both numerical data and categorical data. There are many ratio data in the set for numerical data like the males to females ratio, white and black ratio, etc, that are all continuous. The discrete data represent the countable items, such as the number of retired workers, disabilities, and populations. In categorical data. It labels the data and its characteristics- different county and region names.

This dataset has 2868 rows and 41 columns. The demographic data is found in the County and City Data Book. They were extracted from these tables and placed in a student spreadsheet file. The vote data is collected by the major news outlets.

2.2 Attributes related to sub-problems

Background:

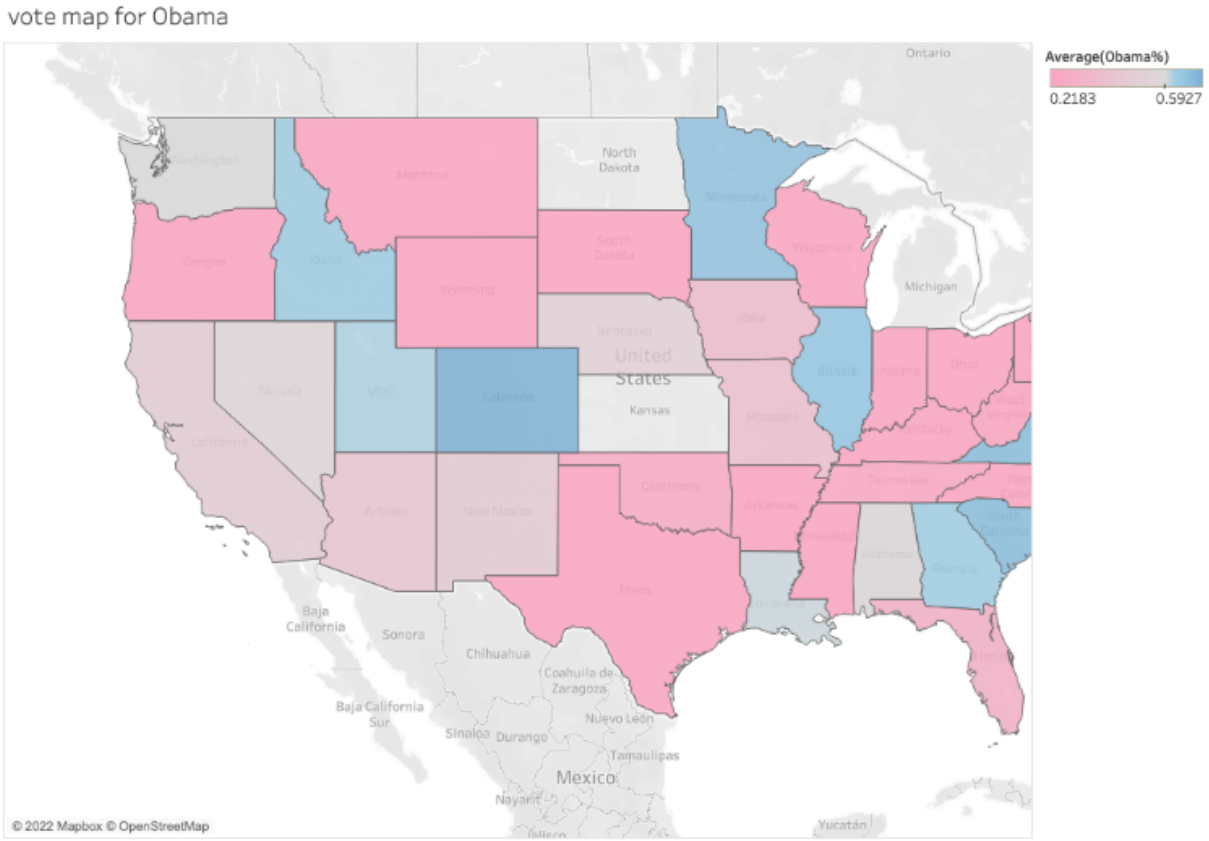


Figure 2: Vote map for Obama

The area of blue represents over 50% of people in this area vote for Obama.

Wealth:

From the bar plots of the average income for Obama and Clinton, we can find that CT, NJ, and MA are the top three states with the highest average income, the shades of the colours represent the vote level for this state. e.g. in the Obama plot, the blue colour shows that there are over 50% of people in this state that vote for Obama. Therefore, for both, there are 9 states that satisfy this requirement.

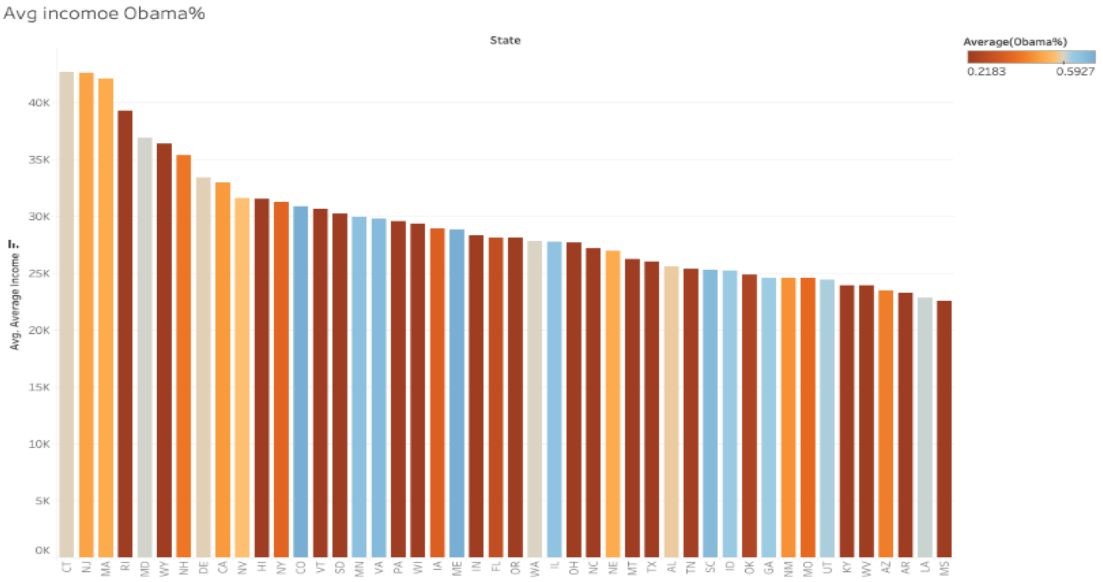


Figure 3: Average income vs. votes for Obama

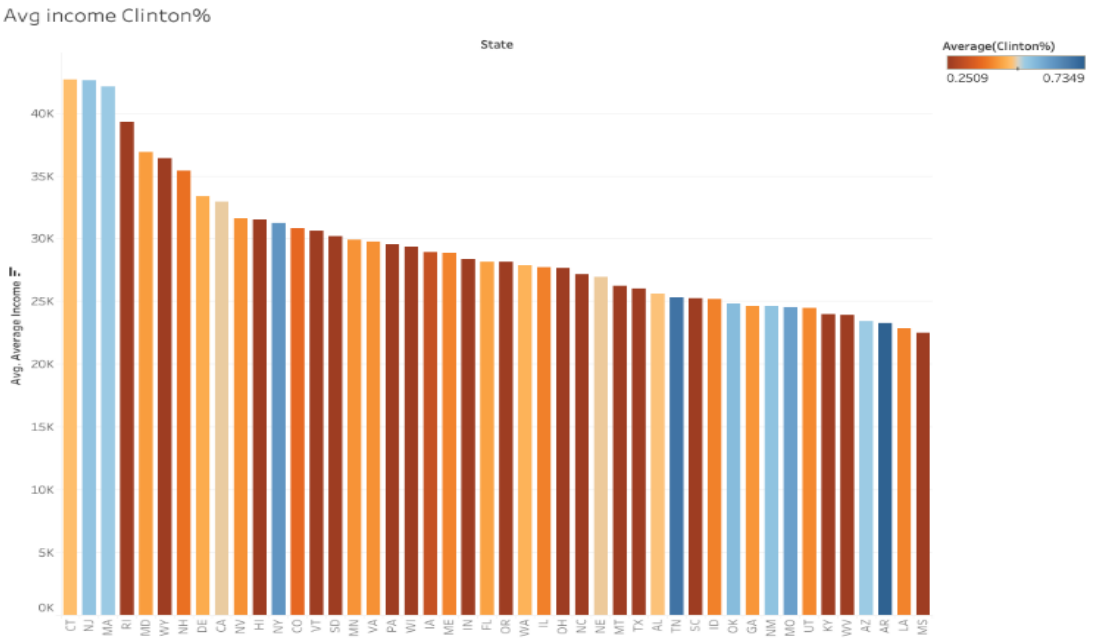


Figure 4: Average income vs votes for Clinton

The blue columns for the Obama plots concentrate on the middle distribution, but for the Clinton plot, these are focused on the states with lower average income. According to the average income bar plots (by different income groups), the higher the wage, the more people voted for Obama (more than 75K) Also, the states with a wider wealth gap contribute more votes to Obama. While for Clinton, she was more popular with lower waged people.

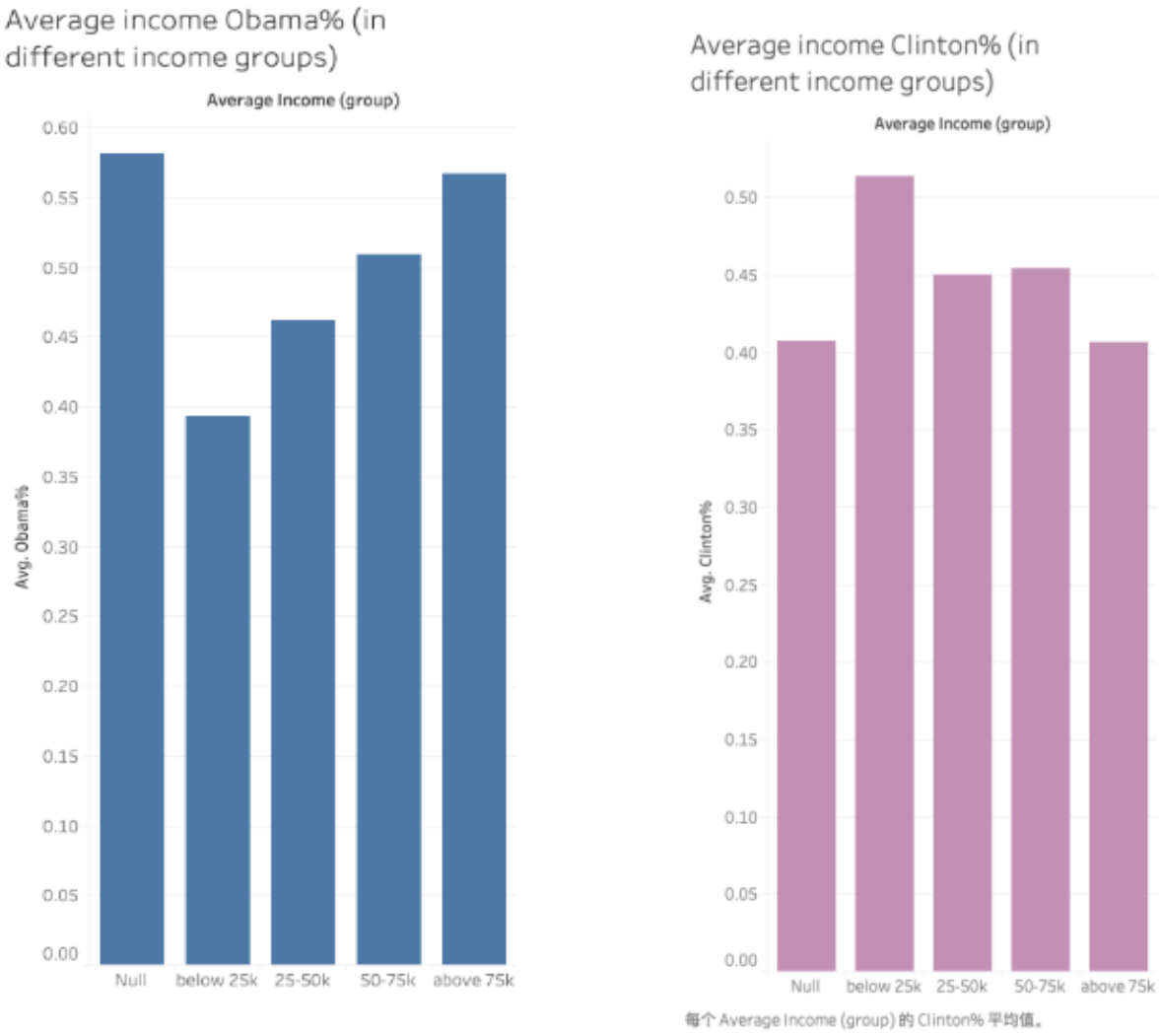


Figure 5: Average income vs. votes (by different income groups)

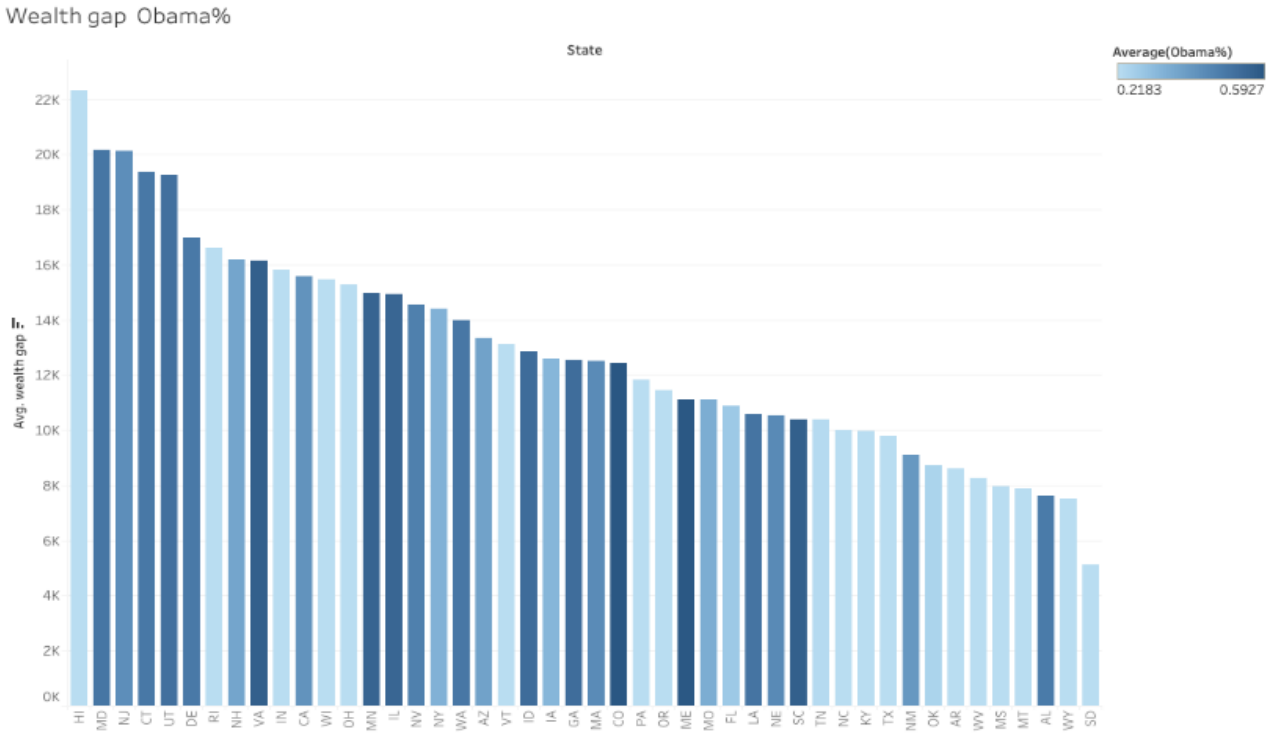


Figure 6: Wealth gap vs. votes for Obama

Age:

Age is separated into three categories which are 'AgeBelow35', 'Age35to65' and 'Age65andAbove'. The table shows population proportion in various age groups within the county. Blue circles in the scatter plots represent a county. Clinton's support among AgeBelow35 voters was negatively correlated. The support rate is positively connected among 'Age35to65' and 'Age65andAbove', and Buchanan County showed the most prominent support rate of all counties.

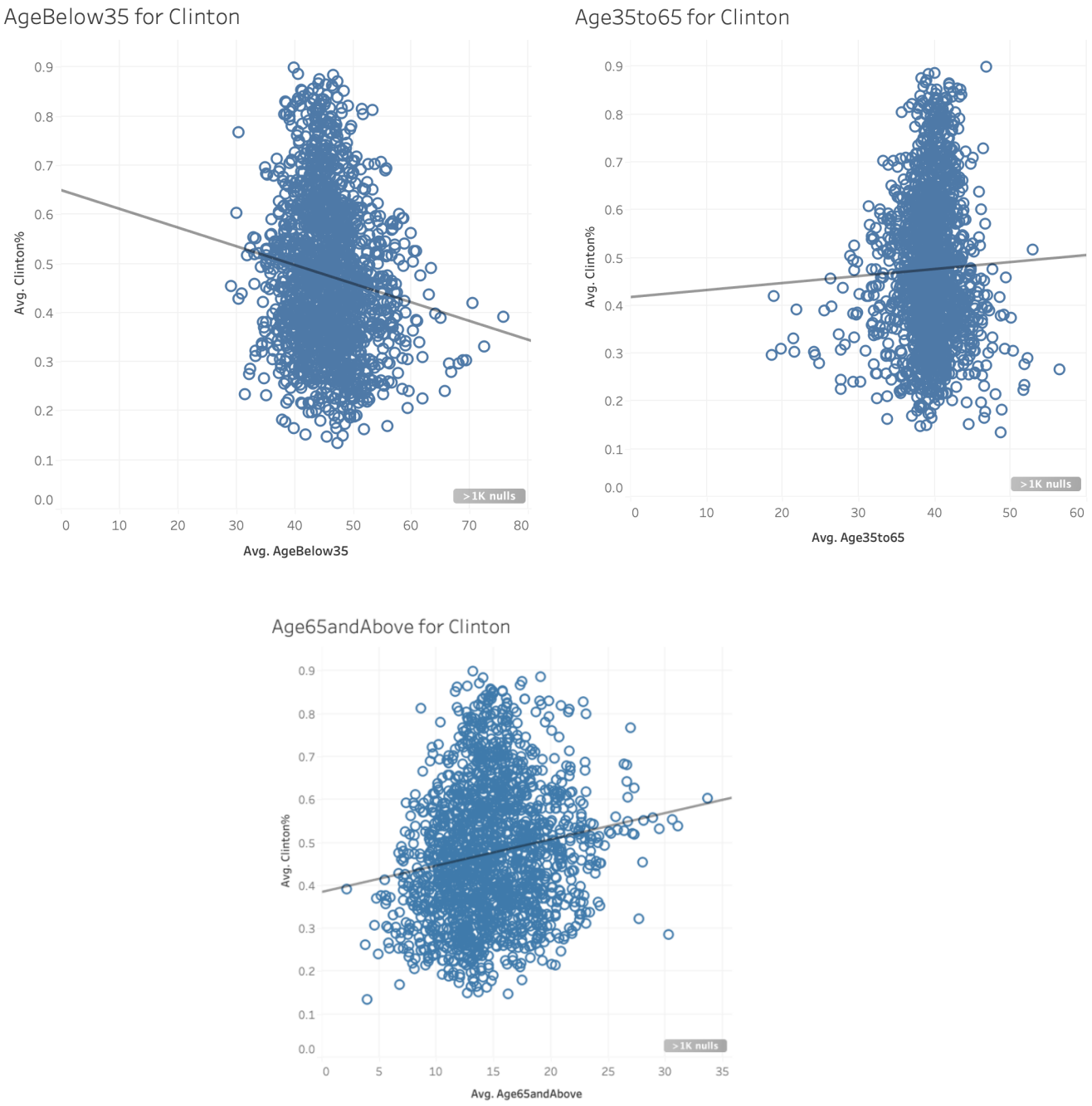


Figure 7: Three age groups vs. votes for Clinton

Obama's support among the 'AgeBelow35' category is favourably correlated. San Miguel had an excellent acceptance rating but 'Age35to65' and 'Age65andAbove', showed a negative association.

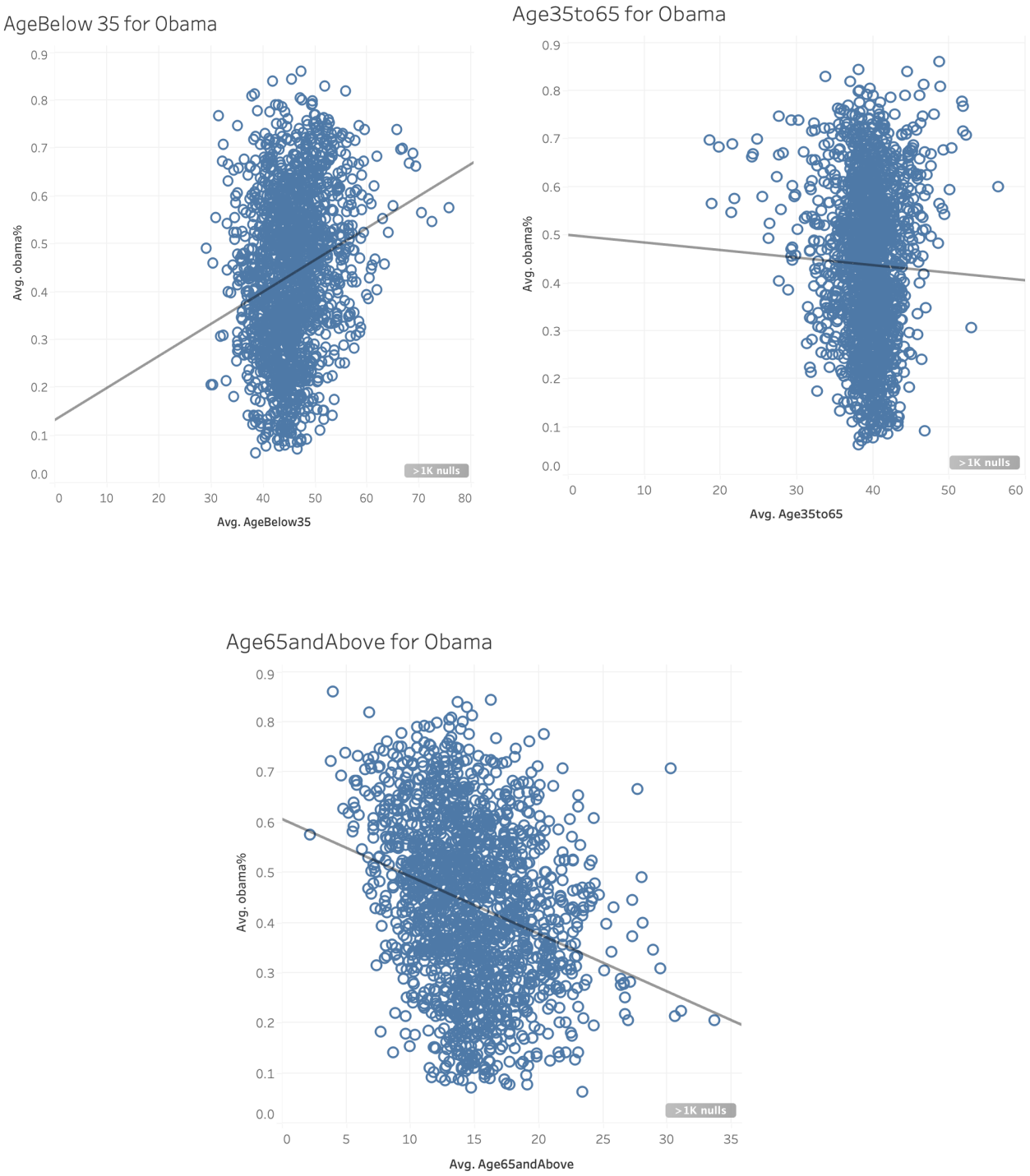


Figure 8: Three age groups vs. votes for Obama

2.3 Correlations between variables

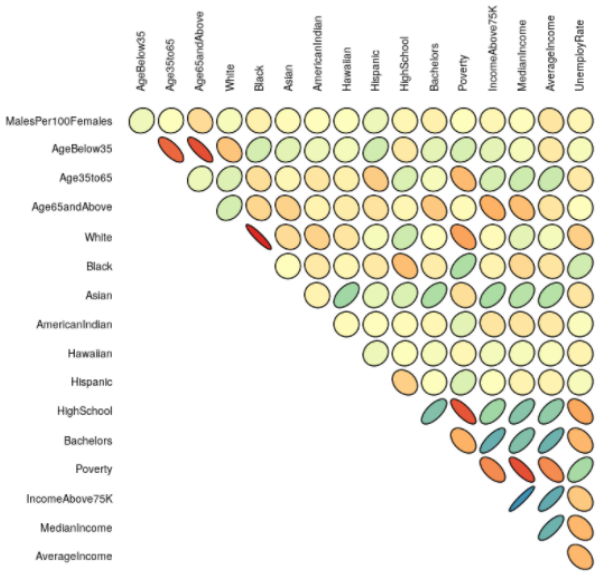


Figure 9: The correlograms of 16 attributes

The category 'Age65andAbove' is negatively related to wealth and 'IncomeAbove75k' and 'MedianIncome,' with age 35 to 65 is positively correlated to wealth but negatively related to poverty.

After making the aggregation table, we notice that from age 35 to 65, the older people are, the higher their income.

```
[10]: cor(elect.df[,c(12,25)],use="complete.obs")
```

A matrix: 2 × 2 of type dbl

	Age35to65	AverageIncome
Age35to65	1.0000000	0.3105978
AverageIncome	0.3105978	1.0000000

Figure 10: Aggregation table for 1 age group vs. Average income

Section 3: Prepare the Data

3.1 Derived the chosen target attributes in Tableau and in R.

To work with our selected attributes, we processed the available data from the Clinton-Obama dataset through the use of Tableau and R. In R, we derived 1 code, indicated as 'ObamaRate', which records the percentage vote cast in the election for Obama.

```
# Load and prepare the data.
elect.df  <- read.csv('Obama.csv')

# Create the derived target attribute.
elect.df$ObamaRate <- 100 * elect.df$Obama / elect.df$TotalVote
summary(elect.df$ObamaRate)

print(head(elect.df))
```

3.2 Steps taken to prepare data

Check for the missing value:

```
# Identify missing value in the data.
countNAs <- function (v) sum(is.na(v))

elect.countNAs <- sapply(elect.df, countNAs)

elect.countNAs[elect.countNAs != 0]
```

TotalVote: 1131 **Clinton:** 1131 **Obama:** 1131 **Black:** 80 **Asian:** 94 **AmericanIndian:** 99 **HighSchool:** 1 **Bachelors:** 1 **Poverty:** 1 **IncomeAbove75K:** 2 **MedianIncome:** 1 **AverageIncome:** 30 **UnemployRate:** 1 **ManfEmploy:** 293 **SpeakingNonEnglish:** 1 **Medicare:** 1 **MedicareRate:** 1 **SocialSecurity:** 1 **SocialSecurityRate:** 1 **RetiredWorkers:** 1 **Disabilities:** 8 **DisabilitiesRate:** 8 **Homeowner:** 2 **SameHouse1995and2000:** 1 **LandArea:** 1 **FarmArea:** 87 **ObamaRate:** 1131 **RateDifferent:** 1131 **MidIncome:** 2

The above code resulted in missing values, so we have to compute them in the following ways:

Impute "reasonable" values for missing data.

Missing 'AverageIncome' is replaced by MedianIncome values. Missing Minorities (Black, Asian, AmericanIndian values) are replaced by 0. Then, we delete the rest of the missing values and assign them to the final record, named: "Finalelect.df".

```
> elect.df$AverageIncome <- ifelse(is.na(elect.df$AverageIncome),
+                                  elect.df$MedianIncome,
+                                  elect.df$AverageIncome)
```

```
> for (attributes in c("Black","Asian","AmericanIndian")){elect.df[[attributes]]<-ifelse(is.na(elect.df[[attributes
]]),0,elect.df[[attributes]])}
```

```
> Finalelect.df <- na.omit(elect.df)
```

3.3 Prepare suitable separate "Training" and "Testing" subsets of the dataset.

Convert the ElectionDate into "Date data type"

```
> View(Finalelect.df)
> head(elect.df$ElectionDate)
[1] "1/3/2008" "1/3/2008" "1/3/2008" "1/3/2008" "1/3/2008" "1/3/2008"
> elect.df$ElectionDate <- as.Date(elect.df$ElectionDate, format="%m/%d/%Y")
> head(elect.df$ElectionDate)
[1] "2008-01-03" "2008-01-03" "2008-01-03" "2008-01-03" "2008-01-03" "2008-01-03"
```

Create dataset

We have two records from our dataframe, they are - elect.df.known and elect.df.unknown. elect.df.known is a known dataset, which is used to construct and evaluate the models, by splitting it into "train and test" dataset; elect.df.unknown consists of counties that did not vote, the best fit model will use this dataset.

```
> elect.df.known <- Finalelect.df[Finalelect.df$ElectionDate <
+                                 as.Date("2/19/2008", format = "%m/%d/%Y"), ]
> elect.df.unknown <- elect.df[elect.df$ElectionDate >=
+                                as.Date("2/19/2008", format = "%m/%d/%Y"), ]
```

Split the data

Here, we need to consider the rows in the known dataset and therefore set seed for a random sample, which we show below:

```
> nrow(elect.df.known)
[1] 1457
> nrow(elect.df.unknown)
[1] 1131
> nKnown <- nrow(elect.df.known)
> set.seed(250)
```

We sample 75% of the row indices randomly in the known dataset, further we name the dataset "elect.df.train". The 25% which is left, is seen as the test dataset.

```
> rowIndicesTrain <- sample(1:nKnown, size = round(nKnown*0.75),replace = FALSE)
> elect.df.training <- elect.df.known[rowIndicesTrain, ]
> elect.df.test <- elect.df.known[-rowIndicesTrain, ]
```

Section 4: Generate and Test Prediction Model

The three prediction models used are ***Linear Regression***, ***Lasso*** and ***Regression Tree***. Linear Regression is used as the most basic form of prediction model where x (independent variable of Obama rate) and y (dependent variables) are assumed a linear relationship but are not penalised for its choice of weights. Thus, using Lasso where the model is penalised for the sum of absolute values of the weights will avoid overfitting in small datasets. Lastly, Regression Tree is used to avoid the linear assumption of the other two models that could limit their capacity on modelling the underlying dataset.

All model types are configured using different predictors. The first configuration is using all census attributes in the dataset, which acts as a benchmark for other configurations in terms of error rates. The second configuration (Insight1) uses age, wealth and other demographic related attributes (minority, gender) as the predictors. Finally, the last configuration (Insight2) will only be using age and wealth-related attributes. Attributes of Insight1 and Insight2 were chosen after it was noticed that certain age and wealth attributes were highly correlated with the Obama rate.

Model	MAE	RMSE
Linear regression (all attributes)	9.328	11.69
Linear regression (age, income and demographics)	10.950	13.33
Linear regression (age and income)	13.470	16.23
Linear regression (step backward)	10.950	13.33
Linear regression (step forward)	10.950	13.33
Decision tree (all attributes)	8.623	10.60
Decision tree (age, income and demographics)	9.357	11.61
Decision tree (age and income)	11.290	13.49
LASSO regression (all attributes)	9.303	11.39
LASSO regression (age, income and demographics)	11.190	13.42
LASSO regression (age and income)	13.630	16.30

Figure 11: MAE and RMSE results

From the regression models, Insight2 model proved there is a slight correlation between chosen attributes and 'ObamaRate'. The LASSO regression showed a low significance level for age-related attributes but higher significance for 'IncomeOver75K' attribute. Lastly, the decision tree is inferred that the attribute 'Black' is the root node if demographic attributes are involved. This can imply that 'Black' is the most significant attribute in classifying votes (with most votes in this favouring Obama). However, in the age and income model, 'IncomeAbove75k' became the root node reflecting the correlation between higher income groups and Obama voters.

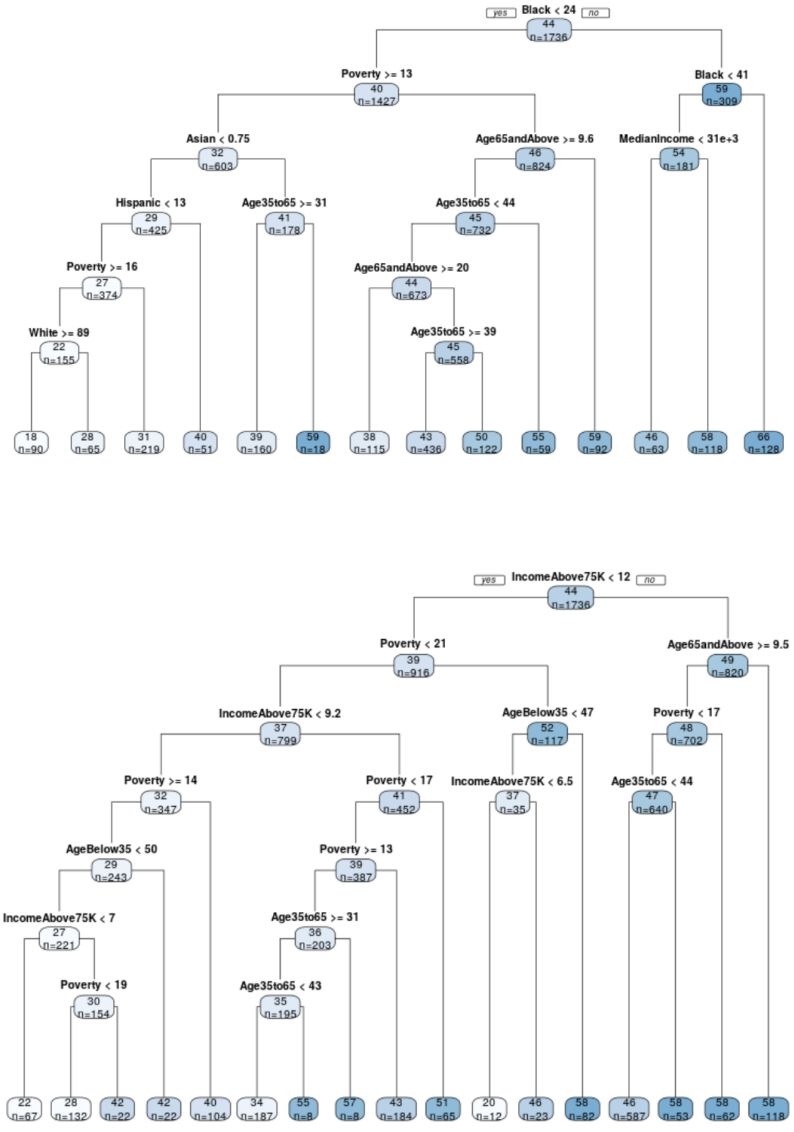


Figure 12: Regression trees for Insight1 and Insight2 model

From the three different models, the regression tree provides the lowest error rates with RMSE of 11.61 and was thus used to predict the unknown data. We checked if age and income variables had a moderating effect on each other and interactions were run

between attributes but the results were not significant. Further, a Principal Component Analysis (PCA) analysis was conducted on the multiple age and income variables and counties were grouped on their vote for Obama or Clinton. There was no clear pattern however, implying these weren't strong predictors (Appendix).

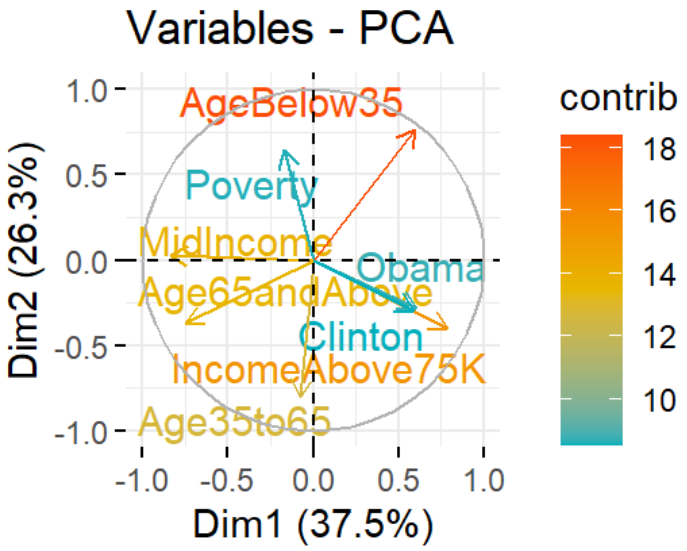


Figure 13: PCA plot 1

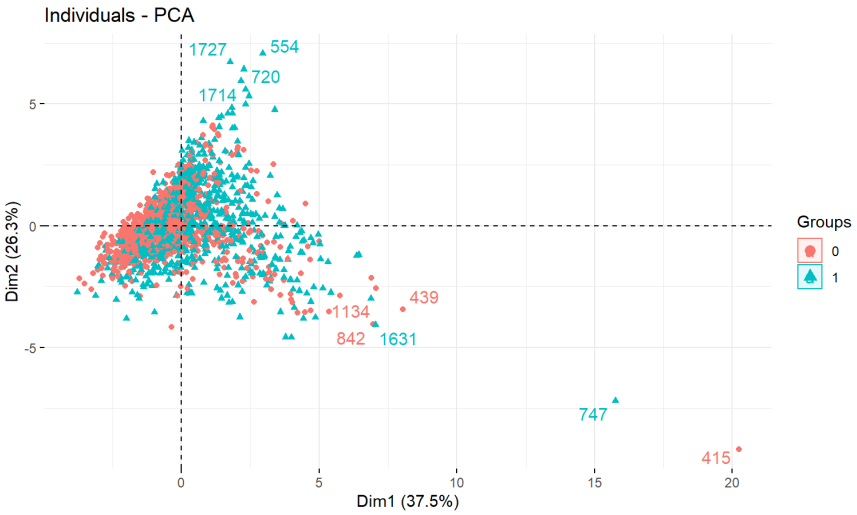


Figure 14: PCA plot 2

Section 5: Problem Conclusions and Recommendations

It can be concluded that the regression tree is the best model to use to produce the lowest error rates. It is also inferred that Obama must focus on states that include a greater black population and income greater than 75K. The model promotes Obama to focus on certain counties of Wyoming, Montana and Mississippi, but the election results show that he did not win a majority in these states. Conversely, the model does not encourage Obama to focus on Oregan and the east coast but he did win a majority in these states. Hence, while age and income-related attributes in the regression tree provide the best model, the results are not significant enough for Obama's campaign to rely on.

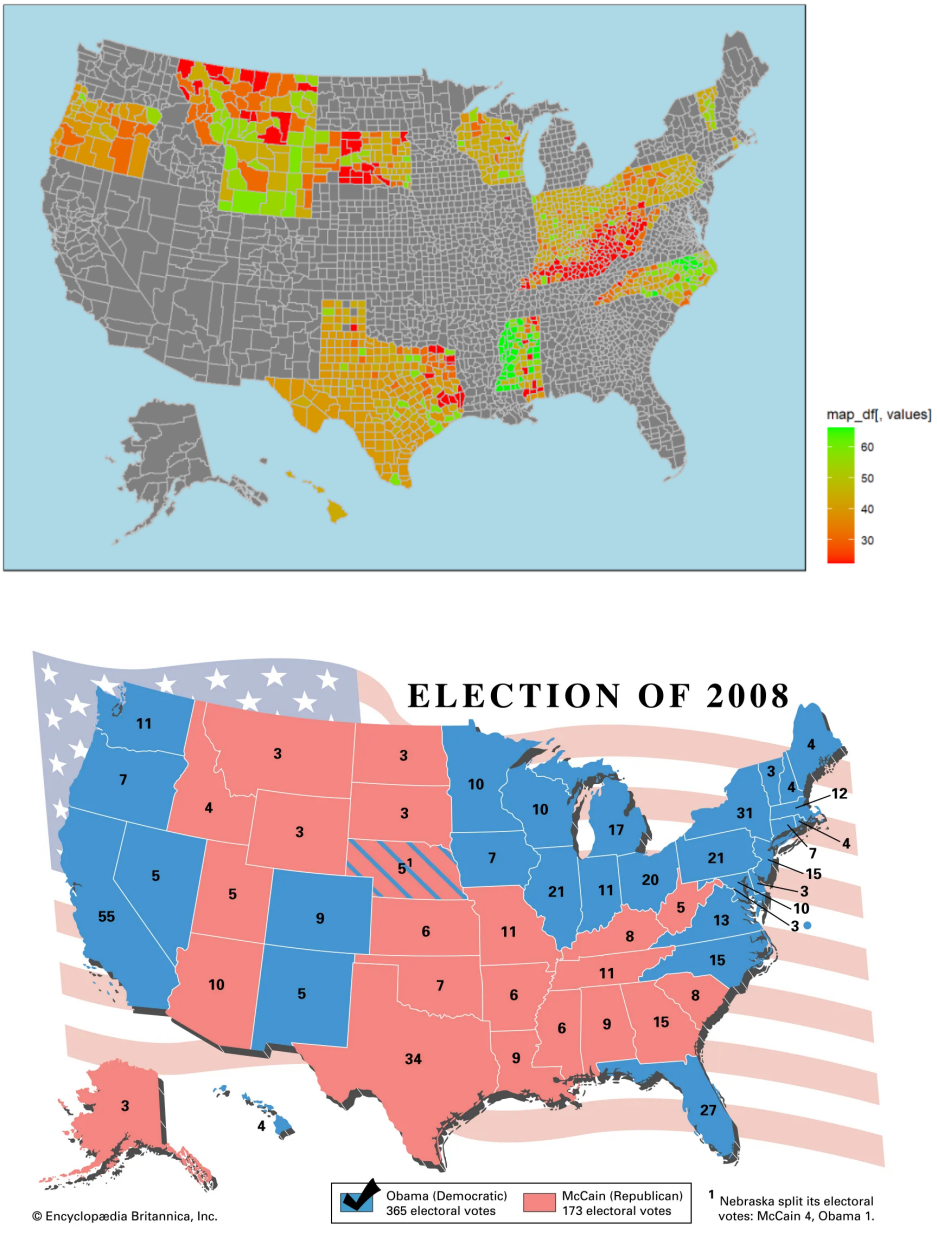


Figure 15: Prediction using Insight1 tree model vs actual result in 2008

Part II: NICU Case Study

US Births data contains the total number of US births from January 2007 to June 2012. Based on analysis of data we aim to give recommendations as to whether the COO of Neonatal Care should spend $1m to expand NICU.

A data.frame: 66 × 4

Yr_Mo	Live.Births	Year	Month
<int>	<int>	<chr>	<chr>
200701	354943	2007	01
200702	326891	2007	02
200703	360828	2007	03
200704	338224	2007	04
200705	362319	2007	05
200706	358606	2007	06
200707	379616	2007	07
200708	390378	2007	08

Figure 16: Pre-processed data-frame

Creating time-series plots, we distinguish insights regarding annual and seasonal trends. Below plots show the births time-series object and the centred moving average displaying the level of births.

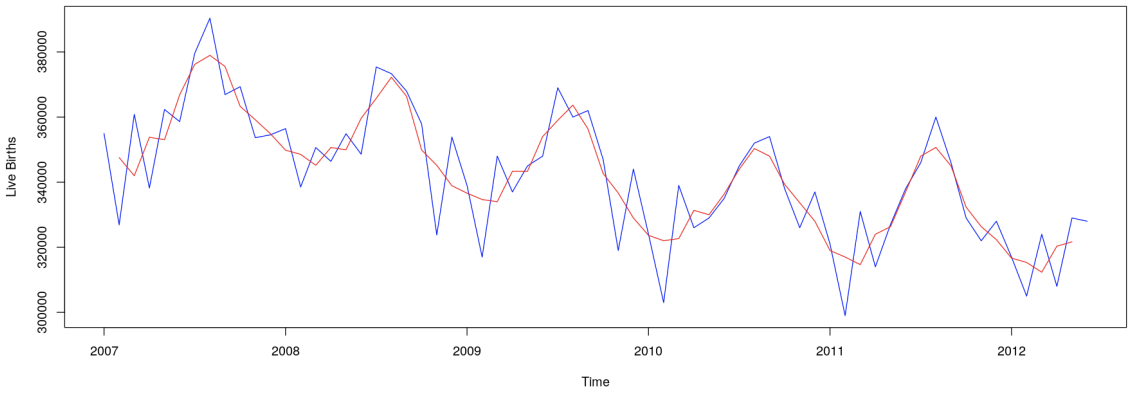


Figure 17: R time-series plot

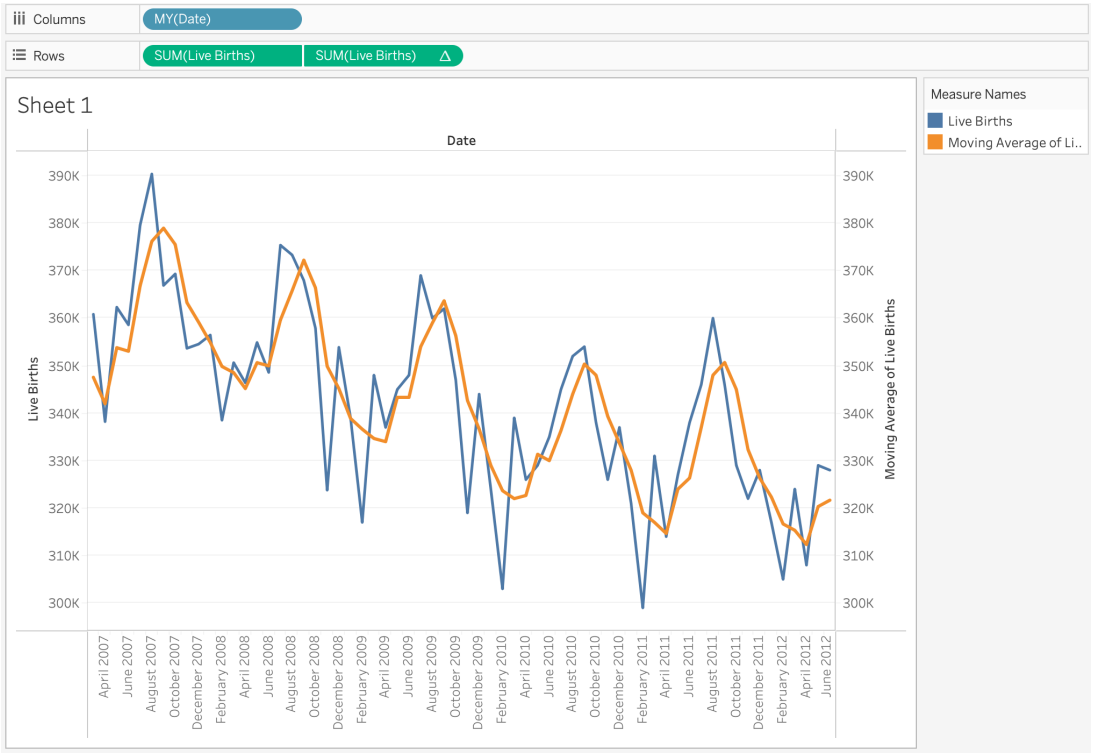


Figure 18: Tableau time-series plot

Although fluctuations between years suggest seasonal trends, an overall decline in total number of births implies that there may not be as much demand for increased bed occupancy in the future. Main reason contributing to this downturn is the economic stress caused by the Great Recession.[1]

[1] (Kearney, et al., 2022)

Additionally, by plotting across monthly data-points we generate seasonal plots allowing us to observe similarities of seasonality patterns for each year, however Tableau plot uses an aggregated average of years for simplicity.

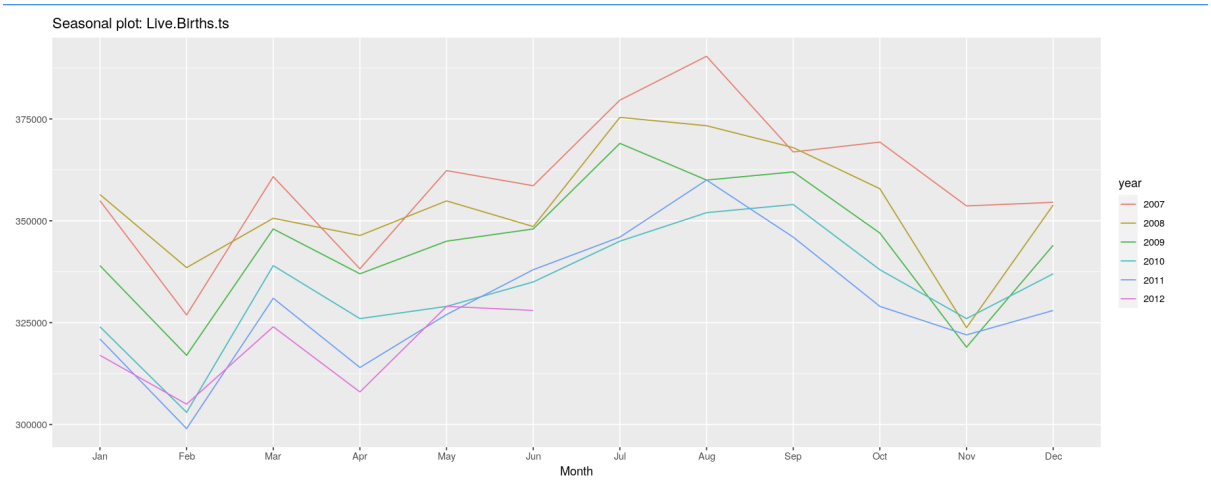


Figure 19: R seasonal plot

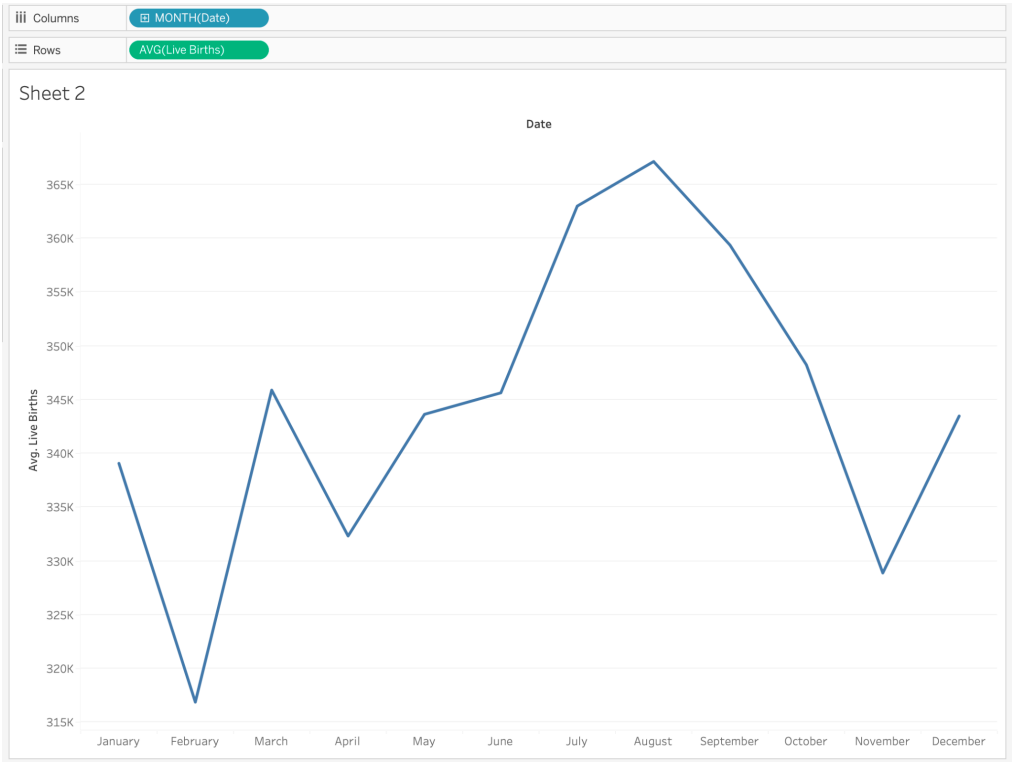


Figure 20: Tableau seasonal plot

In both plots, births' peak in August and trough in February. This seasonal trend is more-or-less representative of US births for the past 50 years. Reasons include increased photoperiod and coital frequency during summer holidays and seasonal preferences in pregnancy planning".[2]

Considering our time-series data vary seasonally, we use Holt's linear trend model for forecasting, observing smoothing parameters corresponding to level and trend. To see the goodness of fit of the model to our dataset, we check the RMSE.

Similarly, by adding a seasonality component, we create an Additive seasonality model to find corresponding RMSE and append it to Model 'AAN' to compare errors.

A data.frame: 2 × 2

RMSE	Model
<dbl>	<chr>
15551.192	AAN
5602.903	AAA

Figure 21: RMSE comparison

With model AAA significant improvement in RMSE compared to model AAN, it is evident that when we incorporate a seasonality component, model AAA will represent our data better.

Model AAA is further leveraged to generate forecast plots (80% confidence levels) for the number of US births up to February 2013 from Tableau and R.

```
February 2013: mean births = 291363.3
               upper 80% confid. births = 300087.2
```

[2] (Darrow, et al., 2009)

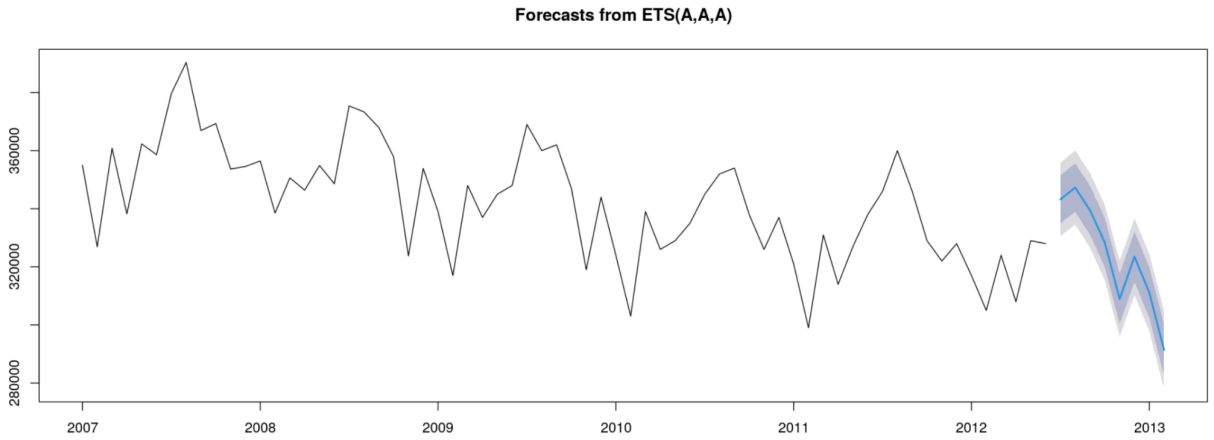


Figure 22: AAA Forecast of US births in R

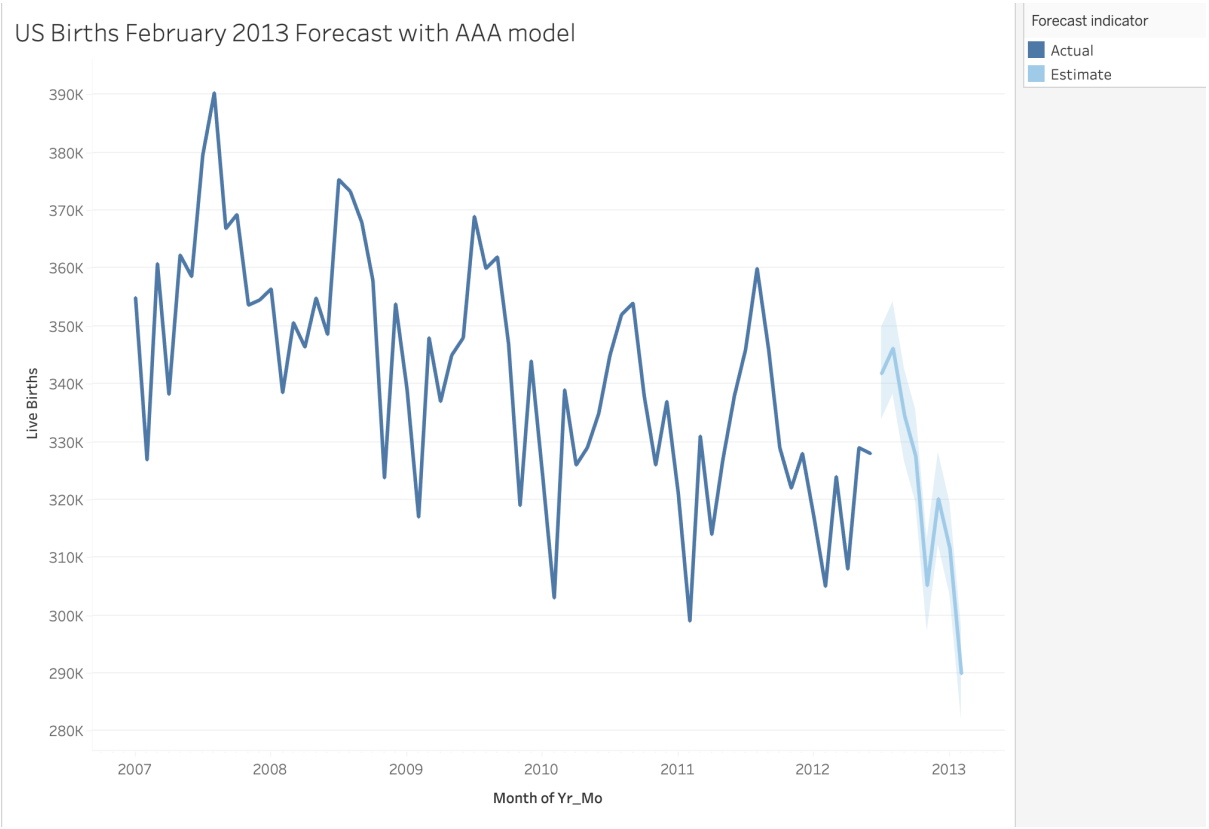


Figure 23: AAA Forecast of US births in Tableau

In line with the seasonality pattern, the number of births first went up in warm seasons and plummeted in its counterpart, showing an overall downward trend up until February 2013. The mean value of births data declines to 291363.3.

Comparison of the seasonality pattern in Births, ALOS, and admission

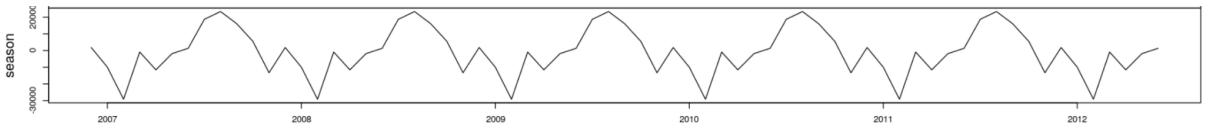


Figure 24: Seasonality trend in US births

With the aforementioned factors that might affect the seasonality trend, the birth rate reaches its peak in July (summer), while drops to its lowest point in February (winter).

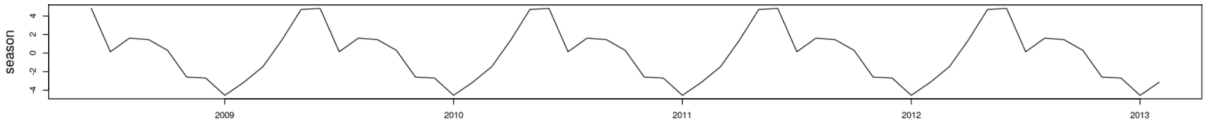


Figure 25: Seasonality trend in NICU Average Length of Stay (ALOS)

The NICU ALOS follows the same pattern as the US birth rates.

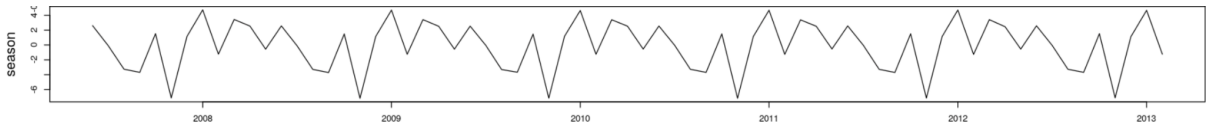


Figure 26: Seasonality trend in NICU admission

On the other hand, the NICU admission does not follow the trend of the other two. One major reason may be the demographic covers in NICU is not large enough to reflect the general US birth trend; whereas a subsidiary factor supporting the peak in the winter may be the increase in the number of infants who failed to defeat cold weather [3].

Recommendations

Based on our analysis, $1 million earmarked on the project should be redirected to the innovation of other units in Children's Hospital instead of increasing the number of beds. The downward-sloping pattern for US live births (Figure 22) depicts a continuous drop in NICU admission up to July 2014[4] (shown as Figure 27), further suggesting that the current number of beds is more than enough to accommodate the needs.

[3] https://www.uofmhealth.org/health-library/tw9031
[4] From Data Analytics II Lecture 3 analysis

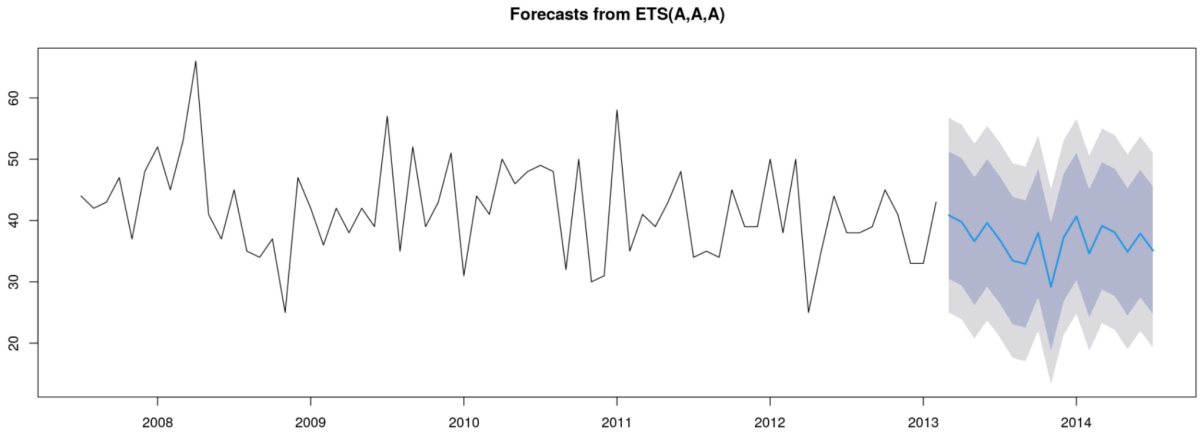


Figure 27: AAA forecast on NICU Admission

REFERENCES

Darrow, L.A., Strickland, M.J., Klein, M., Waller, L.A., Flanders, W.D., Correa, A., Marcus, M. and Tolbert, P.E., 2009. *Seasonality of birth and implications for temporal studies of preterm birth*. Epidemiology (Cambridge, Mass.), 20(5), p.699.

Kearney, M.S., Levine, P.B. and Pardue, L., 2022. *The Puzzle of Falling US Birth Rates since the Great Recession*. Journal of Economic Perspectives, 36(1), pp.151-76.

Michael Cotten, C., Oh, W., McDonald, S., Carlo, W., Fanaroff, A.A., Duara, S., Stoll, B., Laptook, A., Poole, K., Wright, L.L. and Goldberg, R.N. (2005). *Prolonged Hospital Stay for Extremely Premature Infants*: Risk Factors, Center Differences, and the Impact of Mortality on Selecting a Best-Performing Center. *Journal of Perinatology*, 25(10), pp.650–655.

Staff, H. (2020). *Babies and Older Adults Have an Increased Risk of Cold Injury | Michigan Medicine*. [online] www.uofmhealth.org. Available at: https://www.uofmhealth.org/health-library/tw9031 [Accessed 15 Feb. 2022].

APPENDIX

PCA Analysis

We found that counties high on the first dimension (PC1) have more people aged below 35 and income above 75K and low on PC1 include more people aged above 65 with mid-income. Counties high on the second dimension (PC2) have people below 35 years but in poverty and low on PC2 has people aged 35 to 65 with income above 75K.

Code Appendix

Starting from the page below

PART 1: Obama - Clinton

Section 1: The Problem

No coding required

Section 2: Understand the Data

```
#Place the data file in the same directory of your notebook

 elect.df  <- read.csv('Obama.csv')
```

```
#find the corelations between age and wealth
cor(elect.df[,c(11,25)],use="complete.obs")
cor(elect.df[,c(12,25)],use="complete.obs")
cor(elect.df[,c(13,25)],use="complete.obs")
```

[9]:
```
cor(elect.df[,c(11,25)],use="complete.obs")
```

A matrix: 2 × 2 of type dbl

	AgeBelow35	AverageIncome
AgeBelow35	1.0000000	-0.1087067
AverageIncome	-0.1087067	1.0000000

[10]:
```
cor(elect.df[,c(12,25)],use="complete.obs")
```

A matrix: 2 × 2 of type dbl

	Age35to65	AverageIncome
Age35to65	1.0000000	0.3105978
AverageIncome	0.3105978	1.0000000

[11]:
```
cor(elect.df[,c(13,25)],use="complete.obs")
```

A matrix: 2 × 2 of type dbl

	Age65andAbove	AverageIncome
Age65andAbove	1.0000000	-0.1075631
AverageIncome	-0.1075631	1.0000000

[12]:
```
elect.df$Winner <- ifelse(elect.df$Obama>elect.df$Clinton,
                          "Obama",
                          "Clinton")
```

[15]:
```
aggregate(cbind(IncomeAbove75K,Poverty) ~ Winner,
                  data=elect.df,
                  FUN=mean)
```

A data.frame: 2 × 3

```
#Create a simple Winner attribute with two possible values "Obama" or "Clinton"
elect.df$Winner <- ifelse(elect.df$Obama>elect.df$Clinton,
                          "Obama",
                          "Clinton")
```

```
#aggregation table
aggregate(cbind(IncomeAbove75K,Poverty) ~ Winner,
                  data=elect.df,
                  FUN=mean)
```

[15]:
```
aggregate(cbind(IncomeAbove75K,Poverty) ~ Winner,
                    data=elect.df,
                    FUN=mean)
```

A data.frame: 2 × 3

Winner	IncomeAbove75K	Poverty
<chr>	<dbl>	<dbl>
Clinton	12.82738	13.89058
Obama	16.50535	13.00498

```
#Broke the table further down by Region
roundmean <- function(x) round(mean(x),2)

(ag <- aggregate(cbind(IncomeAbove75K,Poverty) ~ Winner + Region,
         data=elect.df,
         FUN=roundmean))
```

[16]:
```
roundmean <- function(x) round(mean(x),2)

(ag <- aggregate(cbind(IncomeAbove75K,Poverty) ~ Winner + Region,
          data=elect.df,
          FUN=roundmean))
```

A data.frame: 8 × 4

Winner	Region	IncomeAbove75K	Poverty
<chr>	<chr>	<dbl>	<dbl>
Clinton	Midwest	10.28	12.39
Obama	Midwest	15.15	9.89
Clinton	Northeast	22.15	10.99
Obama	Northeast	24.24	9.41
Clinton	South	11.67	15.05
Obama	South	16.13	16.10
Clinton	West	14.76	15.09
Obama	West	17.71	11.42


```
#Use the xtabs function to re-shape the above table as a cross-tab table
xtabs(cbind(IncomeAbove75K,Poverty)~ ., data = ag)
```

[17]:
```
xtabs(cbind(IncomeAbove75K,Poverty)~ ., data = ag)
```

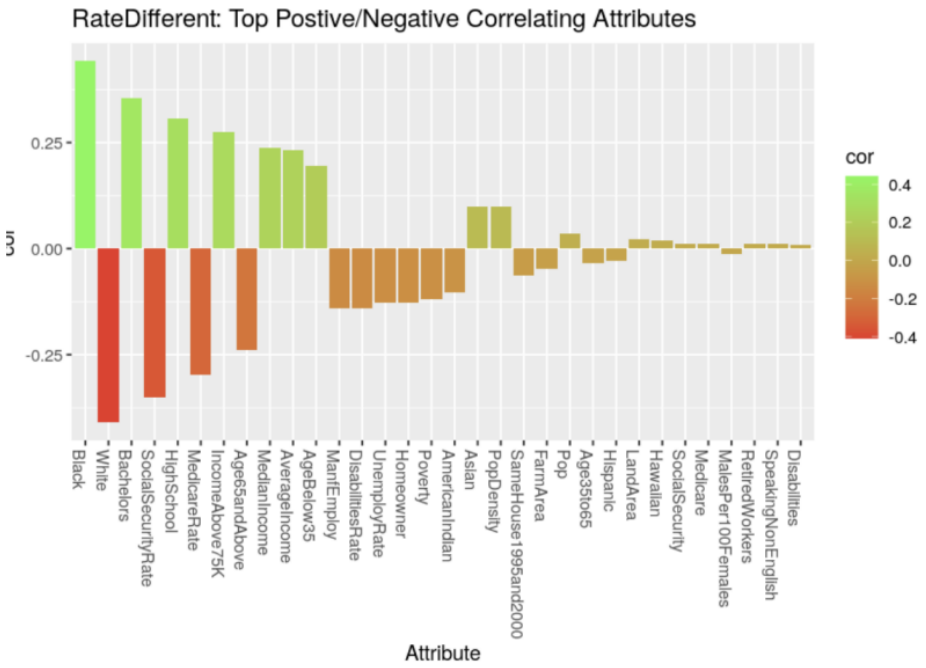
```
, ,  = IncomeAbove75K

         Region
Winner    Midwest Northeast South  West
  Clinton   10.28     22.15 11.67 14.76
  Obama     15.15     24.24 16.13 17.71

, ,  = Poverty

         Region
Winner    Midwest Northeast South  West
  Clinton   12.39     10.99 15.05 15.09
  Obama      9.89      9.41 16.10 11.42
```

[ ]:

Section 3: Prepare the Data

```
# Load and prepare the data.
elect.df  <- read.csv('Obama.csv')

# Create the derived target attribute.
elect.df$ObamaRate <- 100 * elect.df$Obama / elect.df$TotalVote
summary(elect.df$ObamaRate)

print(head(elect.df))
```

```
#Looking for correlation between target attributes and likely predictors

install.packages("ggplot2")

library("ggplot2")

options(repr.plot.height=5)

ggplot(cor.RateDifferent.df, aes(x=reorder(row.names(cor.RateDifferent.df),-abs.cor), y=cor, fill=cor)) +
    geom_col() + ggtitle("RateDifferent: Top Postive/Negative Correlating Attributes") + xlab("Attribute") +
    scale_fill_gradient(low="red", high="green") +
    theme(axis.text.x=element_text(angle=-90, hjust=0))
```




```
# Identify missing value in the data.
countNAs <- function (v) sum(is.na(v))

elect.countNAs <- sapply(elect.df, countNAs)

elect.countNAs[elect.countNAs != 0]
```

TotalVote: 1131 **Clinton:** 1131 **Obama:** 1131 **Black:** 80 **Asian:** 94 **AmericanIndian:** 99 **HighSchool:** 1 **Bachelors:** 1 **Poverty:** 1 **IncomeAbove75K:** 2 **MedianIncome:** 1 **AverageIncome:** 30 **UnemployRate:** 1 **ManfEmploy:** 293 **SpeakingNonEnglish:** 1 **Medicare:** 1 **MedicareRate:** 1 **SocialSecurity:** 1 **SocialSecurityRate:** 1 **RetiredWorkers:** 1 **Disabilities:** 8 **DisabilitiesRate:** 8 **Homeowner:** 2 **SameHouse1995and2000:** 1 **LandArea:** 1 **FarmArea:** 87 **ObamaRate:** 1131 **RateDifferent:** 1131 **MidIncome:** 2

```
# Imputing missing values:
# Missing values for AverageIncome are replaced by the MedianIncome for that same record
elect.df$AverageIncome <- ifelse(is.na(elect.df$AverageIncome), elect.df$MedianIncome, elect.df$AverageIncome)

# Missing values for the following list of attributes are replaced by 0.
for (attr in c("Black","Asian","AmericanIndian","ManfEmploy",
               "Disabilities","DisabilitiesRate","FarmArea"))
    {elect.df[[attr]] <- ifelse(is.na(elect.df[[attr]]), 0, elect.df[[attr]])}

# There still remain several attributes with 1 or 2 missing values.
# It turns out that all these final missing values are in 2 attributes.
# The following codes removes these records (rows) with missingness entirely.
elect.df <- elect.df[is.na(elect.df$HighSchool)==FALSE,]
elect.df <- elect.df[is.na(elect.df$Poverty)==FALSE,]

countNAs <- function (v) sum(is.na(v))
elect.countNAs <- sapply(elect.df, countNAs)
elect.countNAs[elect.countNAs != 0]
#All missing data has addressed except for the missing vote data
```

TotalVote: 1130 **Clinton:** 1130 **Obama:** 1130 **ObamaRate:** 1130

Section 4: Generate and Test Prediction Models

```
#Inspect the ElectionDate column
head(elect.df$ElectionDate)

#Convert to the Date type
elect.df$ElectionDate <- as.Date(elect.df$ElectionDate, format="%m/%d/%Y")

#Check the conversion results
head(elect.df$ElectionDate)

# Create "known" and "unknown" vote data sets.

elect.df.known <- elect.df[elect.df$ElectionDate <
                           as.Date("2/19/2008", format = "%m/%d/%Y"), ]

elect.df.unknown <- elect.df[elect.df$ElectionDate >=
                             as.Date("2/19/2008", format = "%m/%d/%Y"), ]

# Check the number of rows there are in our known and unknown datasets
nrow(elect.df.known)
nrow(elect.df.unknown)

# Find the number of rows in the known dataset
nKnown <- nrow(elect.df.known)

# Set the seed for a random sample
set.seed(201)

# Randomly sample 75% of the row indices in the known dataset
rowIndicesTrain <- sample(1:nKnown, size = round(nKnown*0.75),replace = FALSE)

# Split the known dataset into the training dataset and the test dataset using the sampled indices.

elect.df.training <- elect.df.known[rowIndicesTrain, ]  # Training dataset

elect.df.test <- elect.df.known[-rowIndicesTrain, ]  #Test dataset
```

'1/3/2008' · '1/3/2008' · '1/3/2008' · '1/3/2008' · '1/3/2008' · '1/3/2008'

2008-01-03 · 2008-01-03 · 2008-01-03 · 2008-01-03 · 2008-01-03 · 2008-01-03

1736

1130

Linear Regression

```
# Creat a linear regression model with all census attributes
lmAll <- lm(ObamaRate ~ MalesPer100Females+AgeBelow35+Age35to65+
Age65andAbove+White+Black+Asian+AmericanIndian+
Hawaiian+Hispanic+HighSchool+Bachelors+Poverty+
IncomeAbove75K+MedianIncome+AverageIncome+
UnemployRate+ManfEmploy+SpeakingNonEnglish+
Medicare+MedicareRate+SocialSecurity+
SocialSecurityRate+RetiredWorkers+Disabilities+
DisabilitiesRate+Homeowner+SameHouse1995and2000+
Pop+PopDensity+LandArea,
data = elect.df.training)

summary(lmAll)
```

```
Call:
lm(formula = ObamaRate ~ MalesPer100Females + AgeBelow35 + Age35to65 +
    Age65andAbove + White + Black + Asian + AmericanIndian +
    Hawaiian + Hispanic + HighSchool + Bachelors + Poverty +
    IncomeAbove75K + MedianIncome + AverageIncome + UnemployRate +
    ManfEmploy + SpeakingNonEnglish + Medicare + MedicareRate +
    SocialSecurity + SocialSecurityRate + RetiredWorkers + Disabilities +
    DisabilitiesRate + Homeowner + SameHouse1995and2000 + Pop +
    PopDensity + LandArea, data = elect.df.training)

Residuals:
    Min      1Q  Median      3Q     Max
-38.167  -6.862  -0.028   7.394  35.494

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)          -2.597e+02  3.371e+02  -0.770 0.441208
MalesPer100Females   -4.906e-02  3.627e-02  -1.353 0.176394
AgeBelow35            4.488e-01  3.362e+00   0.133 0.893821
Age35to65             2.072e-01  3.359e+00   0.062 0.950818
Age65andAbove        -3.104e-01  3.354e+00  -0.093 0.926287
White                 2.368e+00  4.801e-01   4.932 9.22e-07 ***
Black                 3.225e+00  4.778e-01   6.749 2.26e-11 ***
Asian                 2.256e+00  5.916e-01   3.814 0.000144 ***
AmericanIndian        2.626e+00  5.455e-01   4.813 1.66e-06 ***
Hawaiian              6.726e+00  3.211e+00   2.095 0.036386 *
Hispanic              2.286e-01  9.088e-02   2.516 0.012004 *
HighSchool            5.679e-01  8.033e-02   7.069 2.57e-12 ***
Bachelors             5.635e-01  9.269e-02   6.079 1.59e-09 ***
Poverty              -1.187e+00  2.069e-01  -5.735 1.22e-08 ***
IncomeAbove75K        1.880e-02  1.713e-01   0.110 0.912661
MedianIncome         -3.385e-04  1.367e-04  -2.477 0.013394 *
AverageIncome         2.772e-06  9.146e-05   0.030 0.975829
UnemployRate          5.965e-01  2.467e-01   2.418 0.015765 *
ManfEmploy           -1.059e-01  4.578e-02  -2.313 0.020858 *
SpeakingNonEnglish   -9.037e-02  1.183e-01  -0.764 0.445131
```

```
Medicare               5.643e-04  1.956e-04   2.885 0.003978 **
MedicareRate           8.751e-05  1.294e-04   0.676 0.498904
SocialSecurity        -6.204e-04  1.669e-04  -3.718 0.000210 ***
SocialSecurityRate     5.762e-06  2.475e-04   0.023 0.981433
RetiredWorkers        -1.205e-04  2.555e-04  -0.471 0.637401
Disabilities          -1.640e-04  1.279e-04  -1.282 0.200003
DisabilitiesRate      -1.105e-03  5.045e-04  -2.189 0.028749 *
Homeowner              8.288e-02  6.937e-02   1.195 0.232442
SameHouse1995and2000   1.368e-01  7.082e-02   1.932 0.053605 .
Pop                    2.155e-05  6.239e-06   3.455 0.000569 ***
PopDensity            -1.533e-04  1.785e-04  -0.859 0.390536
LandArea               1.512e-03  2.681e-04   5.638 2.11e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.76 on 1270 degrees of freedom
Multiple R-squared:  0.5785,    Adjusted R-squared:  0.5682
F-statistic: 56.22 on 31 and 1270 DF,  p-value: < 2.2e-16
```

```
# Create a linear regression model with demographics, age and wealth related attributes.
lm.Insight1 <- lm(ObamaRate ~ MalesPer100Females+White+Black+Asian+AmericanIndian+Hawaiian+Hispanic+AgeBelow35+Age35to65+Age65andAbove
                  +IncomeAbove75K+AverageIncome+MedianIncome,
data = elect.df.training)

summary(lm.Insight1)
```

```
Call:
lm(formula = ObamaRate ~ MalesPer100Females + White + Black +
    Asian + AmericanIndian + Hawaiian + Hispanic + AgeBelow35 +
    Age35to65 + Age65andAbove + Poverty + IncomeAbove75K + AverageIncome +
    MedianIncome, data = elect.df.training)

Residuals:
    Min      1Q  Median      3Q     Max
-37.985  -8.571   0.724   7.686  43.693

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)        -1.514e+02  3.847e+02  -0.393  0.69402
MalesPer100Females  1.258e-02  3.655e-02   0.344  0.73075
White               3.138e+00  4.795e-01   6.544 8.64e-11 ***
Black               3.882e+00  4.756e-01   8.161 7.84e-16 ***
Asian               2.993e+00  5.583e-01   5.361 9.79e-08 ***
AmericanIndian      3.691e+00  5.362e-01   6.883 9.15e-12 ***
Hawaiian            1.090e+01  3.608e+00   3.020  0.00258 **
Hispanic            1.034e-01  4.124e-02   2.508  0.01227 *
AgeBelow35         -5.514e-01  3.835e+00  -0.144  0.88568
Age35to65          -1.037e+00  3.832e+00  -0.271  0.78668
Age65andAbove      -1.279e+00  3.836e+00  -0.333  0.73888
Poverty            -2.276e+00  1.796e-01 -12.670  < 2e-16 ***
IncomeAbove75K      4.975e-01  1.684e-01   2.955  0.00318 **
AverageIncome       2.457e-04  8.221e-05   2.989  0.00285 **
MedianIncome       -6.281e-04  1.458e-04  -4.309 1.76e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 12.36 on 1287 degrees of freedom
Multiple R-squared:  0.436,     Adjusted R-squared:  0.4299
F-statistic: 71.07 on 14 and 1287 DF,  p-value: < 2.2e-16
```

```
# Create a linear regression model with age and wealth related attributes only.
lm.Insight2 <- lm(ObamaRate ~ AgeBelow35+Age35to65+Age65andAbove+Poverty+IncomeAbove75K+AverageIncome+MedianIncome,
data = elect.df.training)

summary(lm.Insight2)
```

```
Call:
lm(formula = ObamaRate ~ AgeBelow35 + Age35to65 + Age65andAbove +
    Poverty + IncomeAbove75K + AverageIncome + MedianIncome,
    data = elect.df.training)

Residuals:
    Min      1Q  Median      3Q     Max
-33.864 -11.047  -0.268  11.144  42.032

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     3.425e+02  4.704e+02   0.728  0.46676
AgeBelow35     -2.666e+00  4.708e+00  -0.566  0.57131
Age35to65      -2.935e+00  4.704e+00  -0.624  0.53283
Age65andAbove  -3.581e+00  4.709e+00  -0.760  0.44713
Poverty        -3.086e-01  1.871e-01  -1.649  0.09929 .
IncomeAbove75K  8.488e-01  1.999e-01   4.246 2.33e-05 ***
AverageIncome   2.537e-04  9.796e-05   2.589  0.00972 **
MedianIncome   -5.776e-04  1.779e-04  -3.247  0.00119 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 15.2 on 1294 degrees of freedom
Multiple R-squared:  0.1429,    Adjusted R-squared:  0.1382
F-statistic: 30.81 on 7 and 1294 DF,  p-value: < 2.2e-16
```

```
# The metrics package includes the mae and rmse functions.
library(Metrics)

genError <- function(prediction, actual)
    return(list(MAE =signif(mae(actual,prediction),4),RMSE = signif(rmse(actual,prediction),4)))

# Make predictions from Model All for test dataset
lmAll.pred <- predict(lmAll, elect.df.test)
# Generate the error rates and add the results to model.results.
error <- genError(lmAll.pred, elect.df.test$ObamaRate)
model.results <- data.frame(MAE = error$MAE, RMSE = error$RMSE, Model="lmAll")
# Make predictions from Model Insight1 for test dataset
lm.Insight1.pred <- predict(lm.Insight1, elect.df.test)
# Generate the error rates and add the results to model.results.
error <- genError(lm.Insight1.pred, elect.df.test$ObamaRate)
model.results <- rbind(model.results, data.frame(MAE = error$MAE, RMSE = error$RMSE, Model="lm.Insight1"))
# Make predictions from Model Insight2 for test dataset
lm.Insight2.pred <- predict(lm.Insight2, elect.df.test)
# Generate the error rates and add the results to model.results.
error <- genError(lm.Insight2.pred, elect.df.test$ObamaRate)
model.results <- rbind(model.results, data.frame(MAE = error$MAE, RMSE = error$RMSE, Model="lm.Insight2"))

model.results
```

MAE	RMSE	Model
<dbl>	<dbl>	<chr>
9.328	11.69	lmAll
10.950	13.33	lm.Insight1
13.470	16.23	lm.Insight2

```
# Create a Backward Stepwise Linear Regression Model for Model Insight1
lm.step.backward <- step(lm.Insight1, direction = "backward")

summary(lm.step.backward)

# Create a Forward Stepwise Linear Regression Model for Model Insight1
lm.min <- lm(ObamaRate ~ 1,
         data = elect.df.training)

lm.step.forward <- step(lm.min,
                        direction='forward',
                        scope=ObamaRate ~ MalesPer100Females+AgeBelow35+Age35to65+Age65andAbove+
         White+Black+Asian+AmericanIndian+Hawaiian+Hispanic+Poverty+IncomeAbove75K+
         MedianIncome+AverageIncome)

summary(lm.step.forward)

# Make predictions, generate the error rates, and add the results to model.results.

lm.step.backward.pred <- predict(lm.step.backward, elect.df.test)

error <- genError(lm.step.backward.pred, elect.df.test$ObamaRate)

model.results <- rbind(model.results, data.frame(MAE = error$MAE, RMSE = error$RMSE, Model="lm.step.backward"))

# Make predictions, generate the error rates, and add the results to model.results.

lm.step.forward.pred <- predict(lm.step.forward, elect.df.test)

error <- genError(lm.step.forward.pred, elect.df.test$ObamaRate)
```

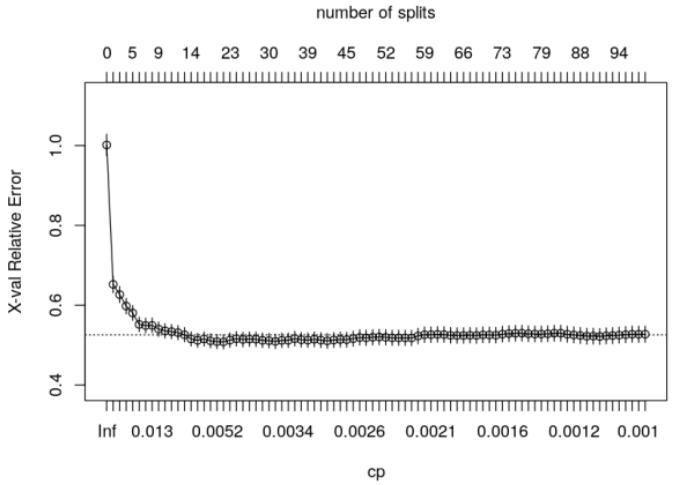
```
model.results <- rbind(model.results, data.frame(MAE = error$MAE, RMSE = error$RMSE, Model="lm.step.forward"))

model.results
```

MAE <dbl>	RMSE <dbl>	Model <chr>
9.328	11.69	lmAll
10.950	13.33	lm.Insight1
13.470	16.23	lm.Insight2
10.950	13.33	lm.step.backward
10.950	13.33	lm.step.forward

Regression Tree

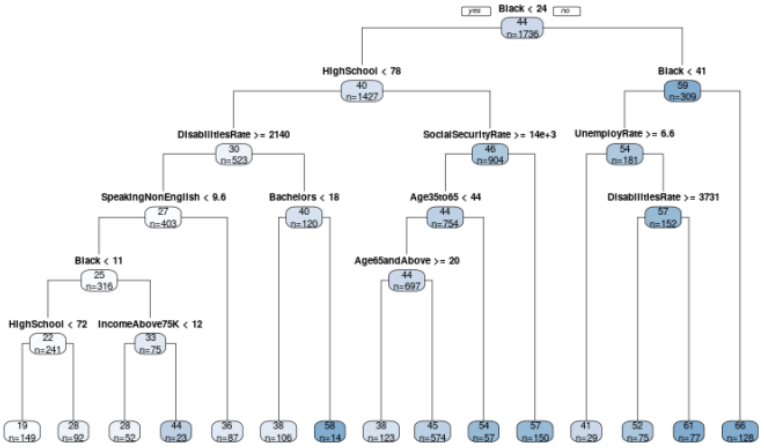
```
# Load packages
library(rpart)
library(rpart.plot)

#Fit the model to all census attributes

rt.all <- rpart(ObamaRate ~ MalesPer100Females+AgeBelow35+Age35to65+
                  Age65andAbove+White+Black+Asian+AmericanIndian+
                  Hawaiian+Hispanic+HighSchool+Bachelors+Poverty+
                  IncomeAbove75K+MedianIncome+AverageIncome+
                  UnemployRate+ManfEmploy+SpeakingNonEnglish+
                  Medicare+MedicareRate+SocialSecurity+
                  SocialSecurityRate+RetiredWorkers+Disabilities+
                  DisabilitiesRate+Homeowner+SameHouse1995and2000+
                  Pop+PopDensity+LandArea,
                data = elect.df.known,
                xval=10,
                cp = 0.001)

#Diagnose the fitted decision tree
plotcp(rt.all,upper = "splits")

#Optimise the Decision Tree
# This function determines the optimal cp corresponding to the tree with the smallest number of splits that has
# a xerror value less than the tree with the best (minimum) xerror value plus its standard error (xstd)
optimalCP <- function(rt.model){
  df<-as.data.frame(rt.model$cptable)
  minerr <- min(df[,"xerror"])
  minerr.xstd <- df[df$xerror==minerr,"xstd"]
  df[df$xerror<minerr+minerr.xstd,][1,"CP"]}

(best_cp <- optimalCP(rt.all))

# We can now "prune" the rpart model back to an optimal number of splits using the optimised cp value
rt.all.opt <- prune(rt.all, cp=best_cp)

#Plot the Optimised Decision Tree
rpart.plot(rt.all.opt, type = 1, extra = 1)

#Determine the Test Error Rate
rt.all.opt.pred <- predict(rt.all.opt, elect.df.test)

error <- genError(rt.all.opt.pred, elect.df.test$ObamaRate)

model.results <- rbind(model.results, data.frame(MAE = error$MAE, RMSE = error$RMSE, Model="rt.all.opt"))

model.results
```

0.00684138897651237



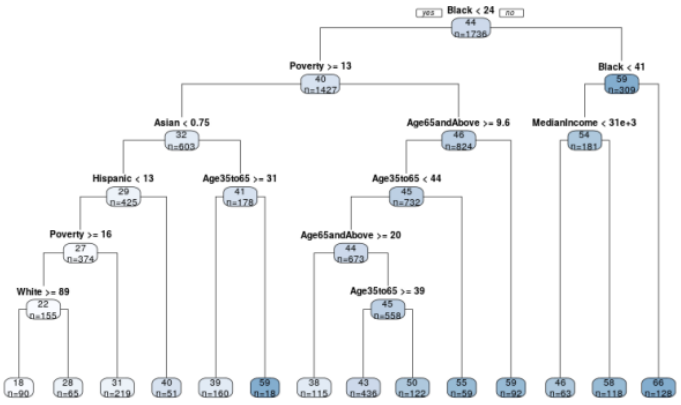
A data.frame: 6 × 3

MAE	RMSE	Model
<dbl>	<dbl>	<chr>
9.328	11.69	lmAll
10.950	13.33	lm.Insight1
13.470	16.23	lm.Insight2
10.950	13.33	lm.step.backward
10.950	13.33	lm.step.forward
8.623	10.60	rt.all.opt




```
#Fit the model to demographics, age and wealth related attributes.

rt.insight1 <- rpart(ObamaRate ~ MalesPer100Females+AgeBelow35+Age35to65+
                  Age65andAbove+White+Black+Asian+AmericanIndian+
                  Hawaiian+Hispanic+Poverty+
                  IncomeAbove75K+MedianIncome+AverageIncome,
                data = elect.df.known,
                xval=10,
                cp = 0.001)

#Diagnose the fitted decision tree
plotcp(rt.all,upper = "splits")

#Optimise the Decision Tree
# This function determines the optimal cp corresponding to the tree with the smallest number of splits that has
# a xerror value less than the tree with the best (minimum) xerror value plus its standard error (xstd)
optimalCP <- function(rt.model){
  df<-as.data.frame(rt.model$cptable)
  minerr <- min(df[,"xerror"])
  minerr.xstd <- df[df$xerror==minerr,"xstd"]
  df[df$xerror<minerr+minerr.xstd,][1,"CP"]}

(best_cp <- optimalCP(rt.insight1))

# We can now "prune" the rpart model back to an optimal number of splits using the optimised cp value
rt.insight1.opt <- prune(rt.insight1, cp=best_cp)

#Plot the Optimised Decision Tree
rpart.plot(rt.insight1.opt, type = 1, extra = 1)

#Determine the Test Error Rate
rt.insight1.opt.pred <- predict(rt.insight1.opt, elect.df.test)

error <- genError(rt.insight1.opt.pred, elect.df.test$ObamaRate)

model.results <- rbind(model.results, data.frame(MAE = error$MAE, RMSE = error$RMSE, Model="rt.insight1.opt"))

model.results
```

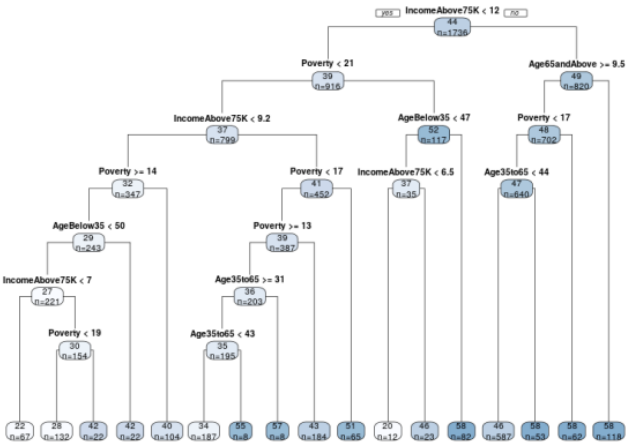
MAE	RMSE	Model
<dbl>	<dbl>	<chr>
9.328	11.69	lmAll
10.950	13.33	lm.Insight1
13.470	16.23	lm.Insight2
10.950	13.33	lm.step.backward
10.950	13.33	lm.step.forward
8.623	10.60	rt.all.opt
9.357	11.61	rt.insight1.opt




```
#Fit the model to age and wealth related attributes only.

rt.insight2 <- rpart(ObamaRate ~ AgeBelow35+Age35to65+
                   Age65andAbove+Poverty+
                   IncomeAbove75K+MedianIncome+AverageIncome,
                 data = elect.df.known,
                 xval=10,
                 cp = 0.001)

#Diagnose the fitted decision tree
plotcp(rt.all,upper = "splits")

#Optimise the Decision Tree
# This function determines the optimal cp corresponding to the tree with the smallest number of splits that has
# a xerror value less than the tree with the best (minimum) xerror value plus its standard error (xstd)
optimalCP <- function(rt.model){
  df<-as.data.frame(rt.model$cptable)
  minerr <- min(df[,"xerror"])
  minerr.xstd <- df[df$xerror==minerr,"xstd"]
  df[df$xerror<minerr+minerr.xstd,][1,"CP"]}

(best_cp <- optimalCP(rt.all))

# We can now "prune" the rpart model back to an optimal number of splits using the optimised cp value
rt.insight2.opt <- prune(rt.insight2, cp=best_cp)
```

```
#Plot the Optimised Decision Tree
rpart.plot(rt.insight2.opt, type = 1, extra = 1)

#Determine the Test Error Rate
rt.insight2.opt.pred <- predict(rt.insight2.opt, elect.df.test)

error <- genError(rt.insight2.opt.pred, elect.df.test$ObamaRate)

model.results <- rbind(model.results, data.frame(MAE = error$MAE, RMSE = error$RMSE, Model="rt.insight2.opt"))

model.results
```

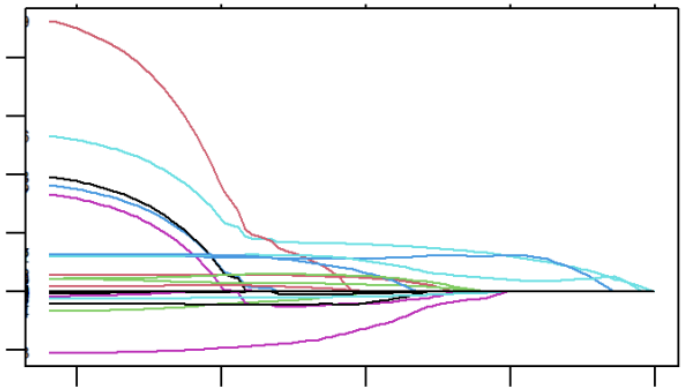
MAE	RMSE	Model
<dbl>	<dbl>	<chr>
9.328	11.69	lmAll
10.950	13.33	lm.Insight1
13.470	16.23	lm.Insight2
10.950	13.33	lm.step.backward
10.950	13.33	lm.step.forward
8.623	10.60	rt.all.opt
9.357	11.61	rt.insight1.opt
11.290	13.49	rt.insight2.opt

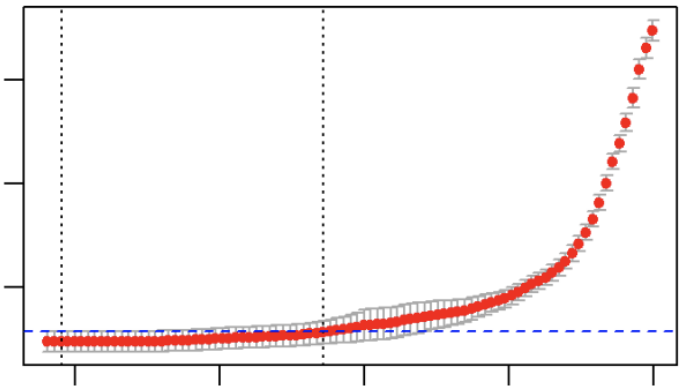


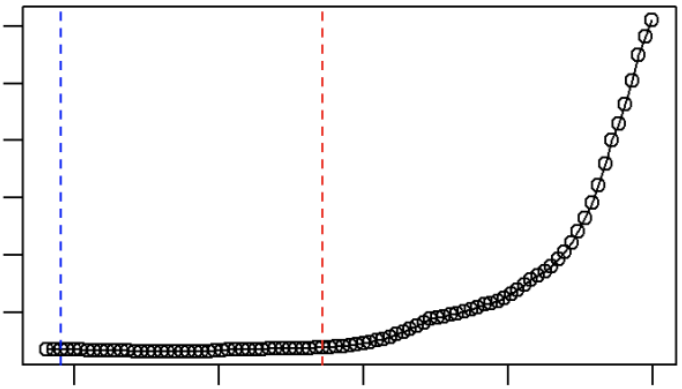
LASSO Regression

```
##### (1)CREATING LASSO MODEL WITH ALL ATTRIBUTES

install.packages("glmnet", repos = "http://cran.us.r-project.org")
library(glmnet)

#splitting dependent and independent variables

startCol <- which(names(elect.df)=="MalesPer100Females")
endCol <- which(names(elect.df)=="FarmArea")
```

```
xknown <- as.matrix(elect.df.known[, startCol:endCol])

yknown <- elect.df.known$ObamaRate


#Generate lasso model
lm.lasso <- glmnet(xknown, yknown, family = "gaussian")

#Plotting the model
par("mar")
par(mar=c(1,1,1,1))
plot(lm.lasso, xvar = "lambda", label = TRUE)

#Coefficients given lambda
coef(lm.lasso, s = 1)

#Cross-validation with 10 folds
set.seed(101)

lm.lasso.cv <- cv.glmnet(xknown, yknown, nfolds = 10, family = "gaussian")

#Finding minimum lambda and log lambda values
lm.lasso.cv$lambda.min
log(lm.lasso.cv$lambda.min)

#Coefficients of linear regression model with lambda min
coef(lm.lasso.cv, s = "lambda.min")

lm.lasso.cv$lambda.1se
log(lm.lasso.cv$lambda.1se)

idx_min=which(lm.lasso.cv$lambda==lm.lasso.cv$lambda.min)
plot(lm.lasso.cv)
abline(h=lm.lasso.cv$cvup[idx_min], lty=2, col='blue')


#Determining test error rate
xtest <- as.matrix(elect.df.test[, startCol:endCol])

# For lambda.min
lm.lasso.min.pred <- predict(lm.lasso.cv, newx = xtest, s = "lambda.min")
error_min <- genError(lm.lasso.min.pred, elect.df.test$ObamaRate)
# For lambda.1se
lm.lasso.1se.pred <- predict(lm.lasso.cv, newx = xtest, s = "lambda.1se")
error_1se <- genError(lm.lasso.1se.pred, elect.df.test$ObamaRate)
# For different lambda values
errorvals <- as.data.frame(t(sapply(lm.lasso.cv$lambda, function(lambda){genError(predict(lm.lasso.cv, newx = xtest, s = lambda),
elect.df.test$ObamaRate)})))

#plot the test errors (RMSEs) of different regression models against different log lambda values.
plot(log(lm.lasso.cv$lambda),errorvals$RMSE, xlab="log lambda", ylab="error (rmse)", type="o" )
abline(v = log(lm.lasso.cv$lambda.min), lty=2, col='blue')
abline(v = log(lm.lasso.cv$lambda.1se), lty=2, col='red')

#Store the error rates of regression models with lambda

model.results <- data.frame(MAE = error_min$MAE, RMSE = error_min$RMSE, Model="lm.lasso (lambda.min)")
model.results <- rbind(model.results, data.frame(MAE = error_1se$MAE,  RMSE = error_1se$RMSE, Model="lm.lasso (lambda.1se)"))
model.results
```

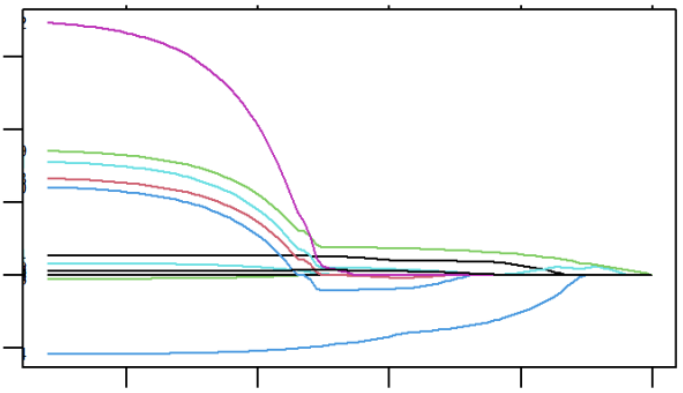


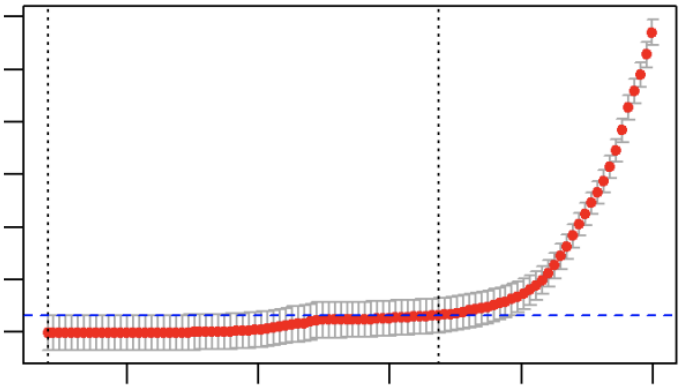


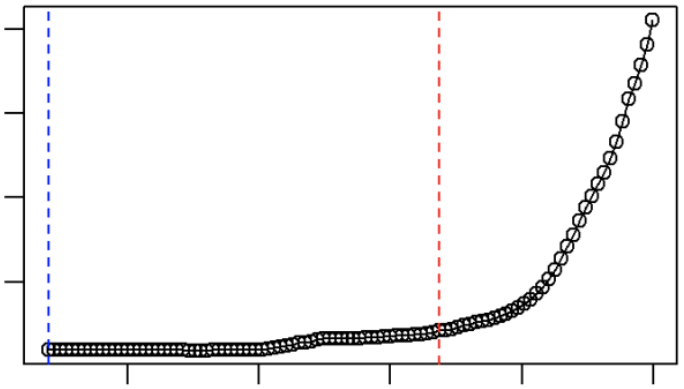


```
##### (2)CREATING LASSO MODEL WITH AGE,INCOME AND OTHER DEMOGRAPHICS
xknownFew <- as.matrix(subset(elect.df.known, select = c(AgeBelow35,Age35to65,Age65andAbove,Poverty,IncomeAbove75K,MedianIncome,Averag
yknown <- elect.df.known$ObamaRate
```

```
#Generate lasso model
lm.lasso2 <- glmnet(xknownFew, yknown, family = "gaussian")

#Plotting the model
par("mar")

par(mar=c(1,1,1,1))
plot(lm.lasso2, xvar = "lambda", label = TRUE)

#Coefficients given lambda
coef(lm.lasso2, s = 1)

#Cross-validation with 10 folds
set.seed(101)

lm.lasso2.cv <- cv.glmnet(xknownFew, yknown, nfolds = 10, family = "gaussian")

#Finding minimum lambda and log lambda values
lm.lasso2.cv$lambda.min
log(lm.lasso2.cv$lambda.min)

#Coefficients of linear regression model with lambda min
coef(lm.lasso2.cv, s = "lambda.min")

lm.lasso2.cv$lambda.1se
log(lm.lasso2.cv$lambda.1se)

idx_min2=which(lm.lasso2.cv$lambda==lm.lasso2.cv$lambda.min)
plot(lm.lasso2.cv)
abline(h=lm.lasso2.cv$cvup[idx_min2], lty=2, col='blue')

#Determining test error rate
xtest2 <- as.matrix(subset(elect.df.test, select = c(AgeBelow35,Age35to65,Age65andAbove,Poverty,IncomeAbove75K, MedianIncome,AverageIn

# For lambda.min
lm.lasso2.min.pred <- predict(lm.lasso2.cv, newx = xtest2, s = "lambda.min")
error2_min <- genError(lm.lasso2.min.pred, elect.df.test$ObamaRate)
# For lambda.1se
lm.lasso2.1se.pred <- predict(lm.lasso2.cv, newx = xtest2, s = "lambda.1se")
error2_1se <- genError(lm.lasso2.1se.pred, elect.df.test$ObamaRate)
# For different lambda values
errorvals2 <- as.data.frame(t(sapply(lm.lasso2.cv$lambda, function(lambda){genError(predict(lm.lasso2.cv, newx = xtest2, s = lambda),
                                                                                    elect.df.test$ObamaRate)})))
#plot the test errors (RMSEs) of different regression models against different log lambda values.
plot(log(lm.lasso2.cv$lambda),errorvals2$RMSE, xlab="log lambda", ylab="error (rmse)", type="o" )
abline(v = log(lm.lasso2.cv$lambda.min), lty=2, col='blue')
abline(v = log(lm.lasso2.cv$lambda.1se), lty=2, col='red')

#Store the error rates of regression models with lambda

model.results <- rbind(model.results, data.frame(MAE = error2_min$MAE,  RMSE = error2_min$RMSE, Model="lm.lasso2 (lambda.min)"))
model.results <- rbind(model.results, data.frame(MAE = error2_1se$MAE,  RMSE = error2_1se$RMSE, Model="lm.lasso2 (lambda.1se)"))
model.results
```

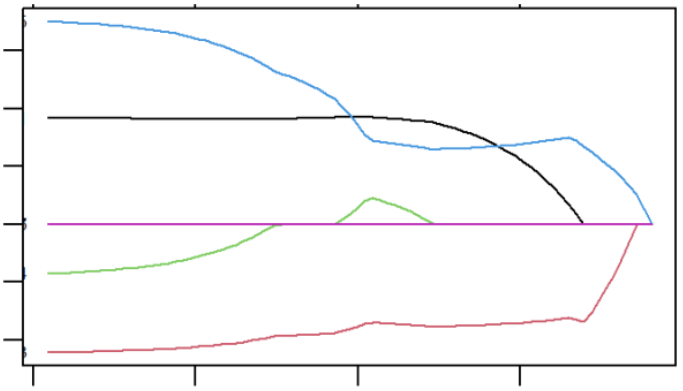


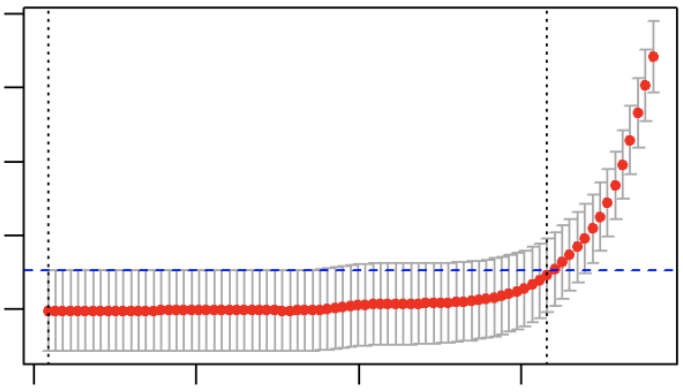


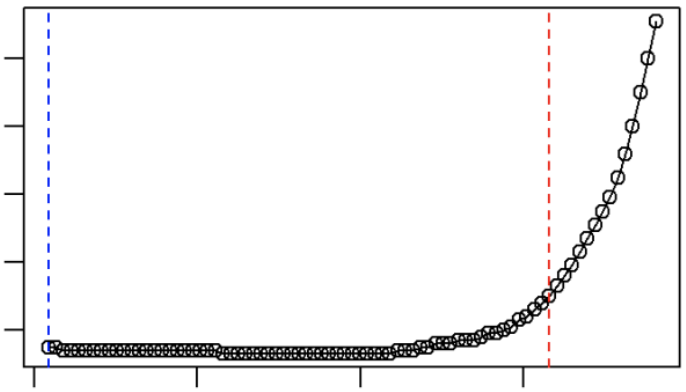


```
##### (3)CREATING LASSO MODEL WITH AGE AND INCOME ATTRIBUTES
xknownAI <- as.matrix(subset(elect.df.known, select = c(AgeBelow35,Age35to65,Age65andAbove,Poverty,IncomeAbove75K,MedianIncome,Average
yknown <- elect.df.known$ObamaRate


#Generate lasso model
lm.lasso3 <- glmnet(xknownAI, yknown, family = "gaussian")

#Plotting the model
par("mar")

par(mar=c(1,1,1,1))
plot(lm.lasso3, xvar = "lambda", label = TRUE)

#Coefficients given lambda
coef(lm.lasso3, s = 1)

#Cross-validation with 10 folds
set.seed(101)

lm.lasso3.cv <- cv.glmnet(xknownAI, yknown, nfolds = 10, family = "gaussian")

#Finding minimum lambda and log lambda values
lm.lasso3.cv$lambda.min
log(lm.lasso3.cv$lambda.min)

#Coefficients of linear regression model with lambda min
coef(lm.lasso3.cv, s = "lambda.min")

lm.lasso3.cv$lambda.1se
log(lm.lasso3.cv$lambda.1se)

idx_min3=which(lm.lasso3.cv$lambda==lm.lasso3.cv$lambda.min)
```

```
plot(lm.lasso3.cv)
abline(h=lm.lasso3.cv$cvup[idx_min3], lty=2, col='blue')

#Determining test error rate
xtest3 <- as.matrix(subset(elect.df.test, select = c(AgeBelow35,Age35to65,Age65andAbove,Poverty,IncomeAbove75K,MedianIncome,AverageInc

# For lambda.min
lm.lasso3.min.pred <- predict(lm.lasso3.cv, newx = xtest3, s = "lambda.min")
error3_min <- genError(lm.lasso3.min.pred, elect.df.test$ObamaRate)
# For lambda.1se
lm.lasso3.1se.pred <- predict(lm.lasso3.cv, newx = xtest3, s = "lambda.1se")
error3_1se <- genError(lm.lasso3.1se.pred, elect.df.test$ObamaRate)
# For different lambda values
errorvals3 <- as.data.frame(t(sapply(lm.lasso3.cv$lambda, function(lambda){genError(predict(lm.lasso3.cv, newx = xtest3, s = lambda),
                                                                                    elect.df.test$ObamaRate)})))
#plot the test errors (RMSEs) of different regression models against different log lambda values.
plot(log(lm.lasso3.cv$lambda),errorvals3$RMSE, xlab="log lambda", ylab="error (rmse)", type="o" )
abline(v = log(lm.lasso3.cv$lambda.min), lty=2, col='blue')
abline(v = log(lm.lasso3.cv$lambda.1se), lty=2, col='red')

#Store the error rates of regression models with lambda

model.results <- rbind(model.results, data.frame(MAE = error3_min$MAE,  RMSE = error3_min$RMSE, Model="lm.lasso3 (lambda.min)"))
model.results <- rbind(model.results, data.frame(MAE = error3_1se$MAE,  RMSE = error3_1se$RMSE, Model="lm.lasso3 (lambda.1se)"))
model.results
```



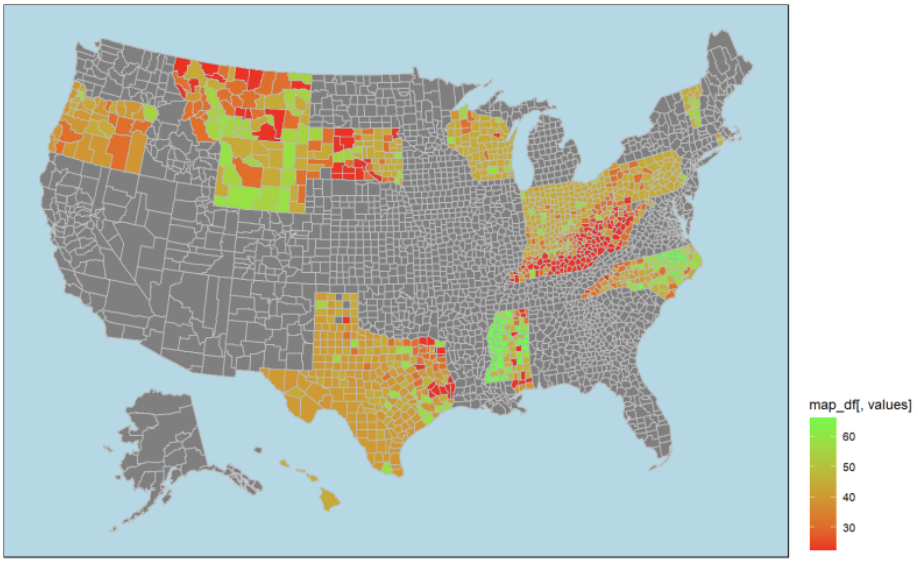




	MAE	RMSE	Model
1	9.144	11.34	lm.lasso (lambda.min)
2	9.303	11.39	lm.lasso (lambda.1se)
3	10.840	13.19	lm.lasso2 (lambda.min)
4	11.190	13.42	lm.lasso2 (lambda.1se)
5	13.420	16.15	lm.lasso3 (lambda.min)
6	13.630	16.30	lm.lasso3 (lambda.1se)

Testing on unknown data

```
unknown.pred <- predict(rt.insight1.opt, elect.df.unknown)
predictions.df <- elect.df.unknown
predictions.df$PredictedObamaRate <- unknown.pred

library(usmap)
library(ggplot2)

pred_obama=data.frame(fips=predictions.df$FIPS,value=predictions.df$PredictedObamaRate)

options(repr.plot.width=12, repr.plot.height=8)
plot_usmap(regions = "counties",data = pred_obama, values = "value", color = "grey") +
  scale_fill_gradient(low = "red", high = "green") +
  theme(panel.background = element_rect(color = "black", fill = "lightblue"),legend.position = "right")
```



Section 5: Problem Conclusions and Recommendations

PART 2: US Birth Data

1. Data processing

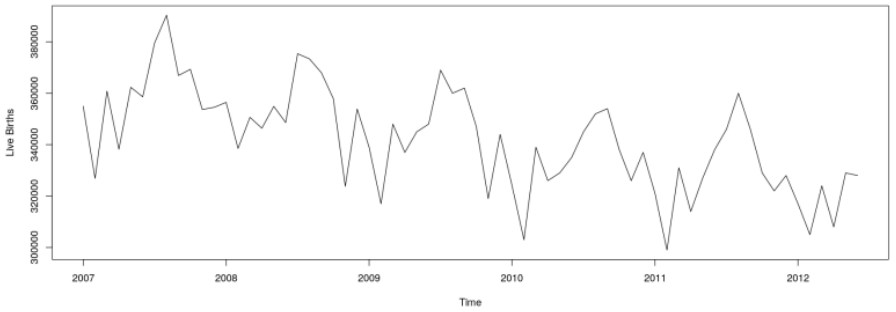
```
(install.packages("caret")
install.packages("zoo")
install.packages("forecast")
install.packages("tidyr")
install.packages("dplyr"))



library(tidyr)
library(dplyr)

# generate a separate Year column attribute and Month column attribute

baby.df <- read.csv('US_Births.csv')
baby.df$Date <- as.Date(paste(as.character(baby.df$Yr_Mo),"1", sep=""), format = "%Y%m%d")
baby.df_2 <- separate(baby.df, Date, c("Year", "Month", "Day"))

baby.df_2 <- baby.df_2 %>% select(Yr_Mo,Live.Births, Year, Month)
baby.df_2
```

A data.frame: 66 × 4

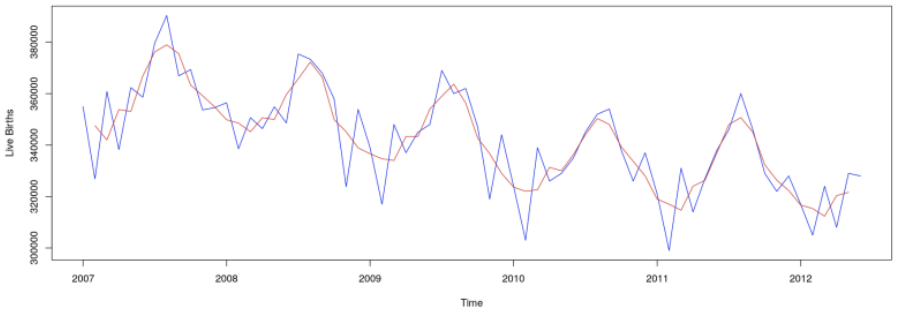
Yr_Mo	Live.Births	Year	Month
<int>	<int>	<chr>	<chr>
200701	354943	2007	01
200702	326891	2007	02
200703	360828	2007	03
200704	338224	2007	04
200705	362319	2007	05
200706	358606	2007	06
200707	379616	2007	07
200708	390378	2007	08

2. Using supporting Tableau visualisations, R plots or R models, discuss and justify any trend or seasonality patterns in this US Births data. **Investigate the likely reasons for these patterns, using your own research sources, supported by appropriate references.**

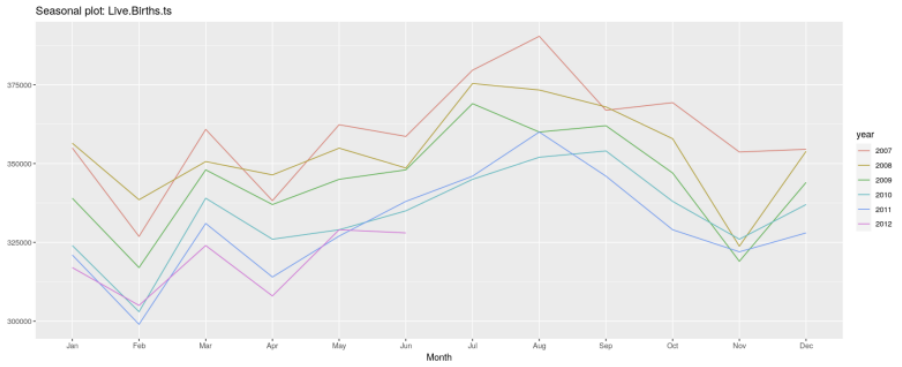
```
# Store data as a Time-Series Object

library(repr)

Live.Births.ts <- ts(baby.df_2$Live.Births,
                     start = c(2007, 1),
                     end = c(2012, 6),
                     freq = 12)

#plot the time series object

options(repr.plot.width = 15, repr.plot.height = 6)
plot(Live.Births.ts, ylab = "Live Births")
```




```
#plot the original time series object and the centred moving average
library(caret)
library(zoo)
library(forecast)
Live.Births.ma_c = rollmean(Live.Births.ts, 3, align = 'center')
plot(Live.Births.ts, ylab = "Live Births", col = 'blue')
lines(Live.Births.ma_c, col = 'red')
```




```
# make a seasonal plot for live births for each year
library(ggplot2)
ggseasonplot(Live.Births.ts)

# in the plot you must exclude the year 2012!!! since there is no full info
```



3. Fit both a trend (AAN) model and a seasonality (AAA) model to this data. Note their RMSE error rates. Which is the better fit?

```
# Generate a Holt's linear model
(Live.Births.ets.AAN <- ets(Live.Births.ts, model = "AAN"))
```

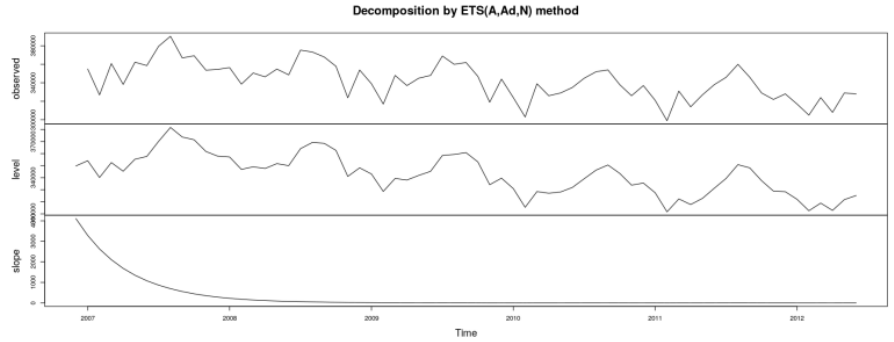
```
ETS(A,Ad,N)

Call:
 ets(y = Live.Births.ts, model = "AAN")

  Smoothing parameters:
    alpha = 0.555
    beta  = 1e-04
    phi   = 0.8002

  Initial states:
    l = 349753.5325
    b = 4107.9239

  sigma:  16175.99

     AIC     AICc      BIC
1562.567 1563.991 1575.705
```

```
# check RMSE error for the 'AAN' model
rmse.ets <- function(etsmodel) sqrt(etsmodel$mse)
(error <- rmse.ets(Live.Births.ets.AAN))

# save RMSE error for the 'AAN' model
Live.Births.rmse = data.frame(RMSE = error, Model ="AAN")
Live.Births.rmse
```

A data.frame: 1 × 2

RMSE	Model
<dbl>	<chr>
15551.19	AAN

```
# plot the decomposition of fitted model
plot(Live.Births.ets.AAN)
```




```
# Generate an additive seasonality model
(Live.Births.ets.AAA <- ets(Live.Births.ts, model = "AAA"))
```

```
ETS(A,A,A)

Call:
 ets(y = Live.Births.ts, model = "AAA")

  Smoothing parameters:
    alpha = 0.1296
    beta  = 1e-04
    gamma = 1e-04

  Initial states:
    l = 363803.3925
    b = -557.8676
    s = 1815.219 -13310.08 5510.557 16085.76 23364.93 18723.94
           1333.24 -1847.983 -11570.24 -899.8829 -29211.22 -9994.237

  sigma:  6437.245

     AIC     AICc      BIC
1449.814 1462.564 1487.039
```

```
# check RMSE error for the 'AAA' model
(error <- rmse.ets(Live.Births.ets.AAA))

# save RMSE error for the 'AAA' model
```

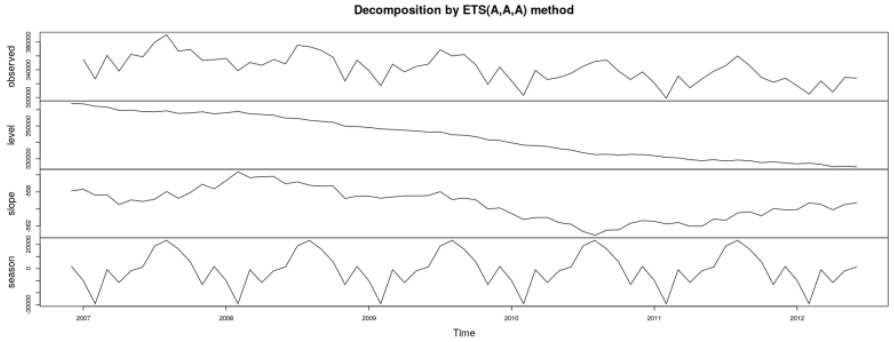
```
Live.Births.rmse = rbind(Live.Births.rmse, data.frame(RMSE = error, Model ="AAA"))
Live.Births.rmse
```

A data.frame: 2 × 2

RMSE	Model
<dbl>	<chr>
15551.192	AAN
5602.903	AAA

With model AAA we see a significant improvement in terms of RMSE compared to model AAN (major decrease) indicating that when we incorporate an additional seasonality component model AAA will represent our Live Births data better hence is the better fit.

```
# plot the decomposition of fitted model
plot(Live.Births.ets.AAA)
```



4. Use your best model to forecast US births to February 2013. Plot this forecast using both Tableau and R with 80% confidence levels.

```
#forecasting AAA model to February 2013
Live.Births.ets.AAA.pred <- forecast(Live.Births.ets.AAA, h = 8)

#plot the forecast of AAA model
par(mfrow = c(2,2)) #(1,1)??
plot(Live.Births.ets.AAA.pred)


#number of US births in February 2013 with 80% confidence levels.
forecast <- forecast(Live.Births.ets.AAA, h = 8, level = c(80,95))
cat('February 2013: mean births =', round(forecast$mean[8],1),"\n")
cat('             upper 80% confid. births =', round(forecast$upper[8,1],1),"\n")
```

5. Compare the seasonality patterns in US births, NICU admissions and NICU average length of stay.

```
#load NICU dataset
NICU.df <- read.csv("NICU.csv")

#store admits data as a time series object
admits.ts <- ts(NICU.df$Admits,
                start = c(2007, 7),
                end = c(2013, 2),
                freq = 12)

#plot admits of the AAA model
(admins.ets.AAA <- ets(admits.ts, model= 'AAA'))
plot(admins.ets.AAA)
```

```
#store ALOS data as a time series object
alos.ts <- ts(NICU.df$ALOS,
              start = c(2007,7),
              end = c(2013,2),
              freq = 12)

#plot ALOS of the AAA Model
(alos.ets.AAA <- ets(alos.ts, model = 'AAA'))
plot(alos.ets.AAA)
```