Course: ALY6020
Term: A_Fall_2020

Name: Pragati Koladiya
NUID: 001029445

Assignment 2

Introduction:

Data science has become such a neat and a functional part of the market and our daily lives.We practically learn the very core of data science all while a common business problem. Here the sole purpose is to show the basic models and techniques of data science like data cleaning, encoding, feature engineering and model training.

The objectives of this assignment are as follows,

- To perform data visualization techniques to understand the insight of the data.
- This led us to perform the KNN and Time series forecasting for the chosen dataset.
- Apply various R tools to get a visual understanding of the data
- Clean it to make it ready to apply machine learning model KNN thus predicting the
- outcome.

Problem Statement:

Consider a real estate company that has all the data containing the prices of properties in a region. It wishes to use the data to optimise the sale prices of the properties based on important factors such as lot size, bedrooms, stories, etc.

Essentially, the company wants,

To identify the variables affecting house prices, e.g. lot size(area), number of rooms, bathrooms, etc.

- To create a linear model that quantitatively relates house prices with variables such as the number of rooms, area, number of bathrooms, etc.

- To answer question like what factors are highly correlated to a satisfied (or dissatisfied) passenger? Can we predict passenger satisfaction?

- To know the accuracy of the model, i.e. how well these variables can predict the customer satisfaction.

Data Acquisition :

The dataset is chosen from Kaggle with file name “AirlineSatisfaction”. This dataset contains an airline passenger satisfaction survey.

The dataset contains 103904 observations(airline customer’s samples) and 25 attributes related to housing.

Data Dictionary:

- The data dictionary is provided by the Kaggle with detailed explanation for each variable.
- link: https://www.kaggle.com/teejmahal20/airline-passenger-satisfaction

Gender: Gender of the passengers (Female, Male)
Customer Type: The customer type (Loyal customer, disloyal customer)
Age: The actual age of the passengers
Type of Travel: Purpose of the flight of the passengers (Personal Travel, Business Travel)

Class: Travel class in the plane of the passengers (Business, Eco, Eco Plus)
Flight distance: The flight distance of this journey
Inflight wifi service: Satisfaction level of the inflight wifi service (raiting from 1-5)
Departure/Arrival time convenient: Satisfaction level of Departure/Arrival time convenient
Ease of Online booking: Satisfaction level of online booking
Gate location: Satisfaction level of Gate location
Food and drink: Satisfaction level of Food and drink
Online boarding: Satisfaction level of online boarding
Seat comfort: Satisfaction level of Seat comfort
Inflight entertainment: Satisfaction level of inflight entertainment
On-board service: Satisfaction level of On-board service
Leg room service: Satisfaction level of Leg room service
Baggage handling: Satisfaction level of baggage handling
Check-in service: Satisfaction level of Check-in service
Inflight service: Satisfaction level of inflight service
Cleanliness: Satisfaction level of Cleanliness
Departure Delay in Minutes: Minutes delayed when departure
Arrival Delay in Minutes: Minutes delayed when Arrival
Satisfaction: Airline satisfaction level(Satisfaction, neutral or dissatisfaction)

Data Overview:

Required packages have been imported before loading the dataset into R.

- Head of the dataset with columns and first few rows. I have removed the space from the variables labels for simplicity.

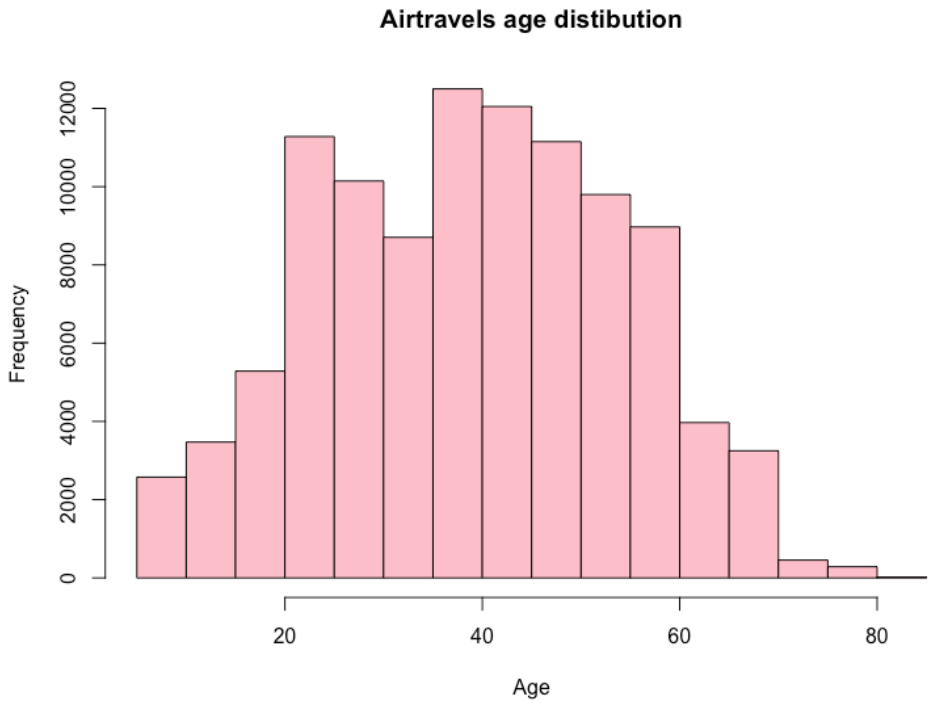
```
# A tibble: 6 x 25
  Sr_No     id Gender CustomerType   Age TypeofTravel Class FlightDistance Inflightwifiser… Departure_Arriv… EaseofOnlineboo… Gatelocation Foodanddrink Onlineboarding Seatcomfort Inflightenterta…
  <dbl>  <dbl> <chr>  <chr>        <dbl> <chr>        <chr>          <dbl>            <dbl>            <dbl>            <dbl>        <dbl>        <dbl>           <dbl>       <dbl>            <dbl>
1     0  70172 Male   Loyal Custo…    13 Personal Tr… Eco …            460                3                4                3            1            5               3           5                5
2     1   5047 Male   disloyal Cu…    25 Business tr… Busi…            235                3                2                3            3            1               3           1                1
3     2 110028 Female Loyal Custo…    26 Business tr… Busi…           1142                2                2                2            2            5               5           5                5
4     3  24026 Female Loyal Custo…    25 Business tr… Busi…            562                2                5                5            5            2               2           2                2
5     4 119299 Male   Loyal Custo…    61 Business tr… Busi…            214                3                3                3            3            4               5           5                3
6     5 111157 Female Loyal Custo…    26 Personal Tr… Eco              1180                3                4                2            1            1               2           1                1
# … with 9 more variables: `On-boardservice` <dbl>, Legroomservice <dbl>, Baggagehandling <dbl>, Checkinservice <dbl>, Inflightservice <dbl>, Cleanliness <dbl>, DepartureDelayinMinutes <dbl>,
#   ArrivalDelayinMinutes <dbl>, satisfaction <chr>
```

Exploratory Data Analysis(EDA):

The most important step is to understand the data and identify if there is some obvious multicollinearity present. Here's where we will also identify if predictors have a strong association with the outcome variable.

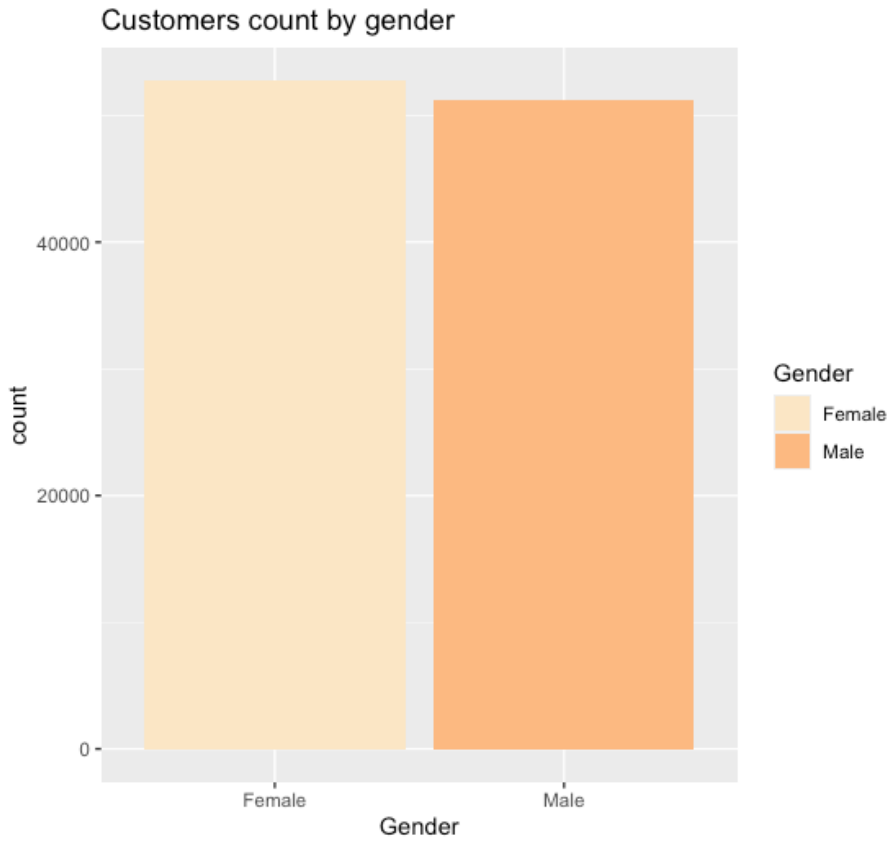
1) Age distribution using histogram

Name: Pragati Koladiya
NUID: 001029445



- Distribution of age is normal distribution. Most passengers are of age 20 to 80.

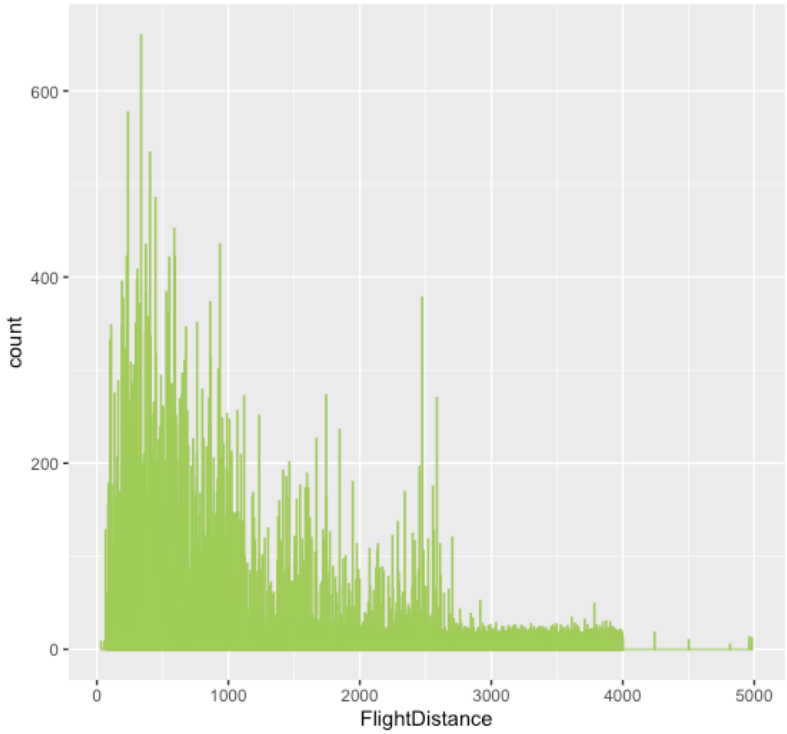
2) Gender wise distribution using bar plot



- Customer count by gender is almost identical.

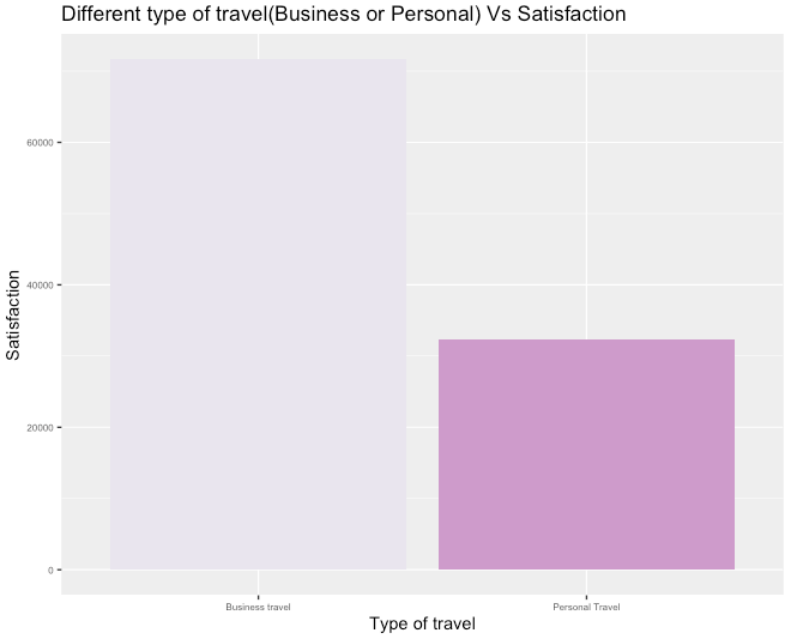
Name: Pragati Koladiya
NUID: 001029445

3) Distribution of `FlightDistance` using histogram



- The above graph shows the number of times the distance covered by flights. Most of the distance covered by airlines is between 100miles to 1000miles. Which shows that most flights are flying between cities, e.g. California to New York city has distance of 213.67 miles which shows the given data set is about domestic flights

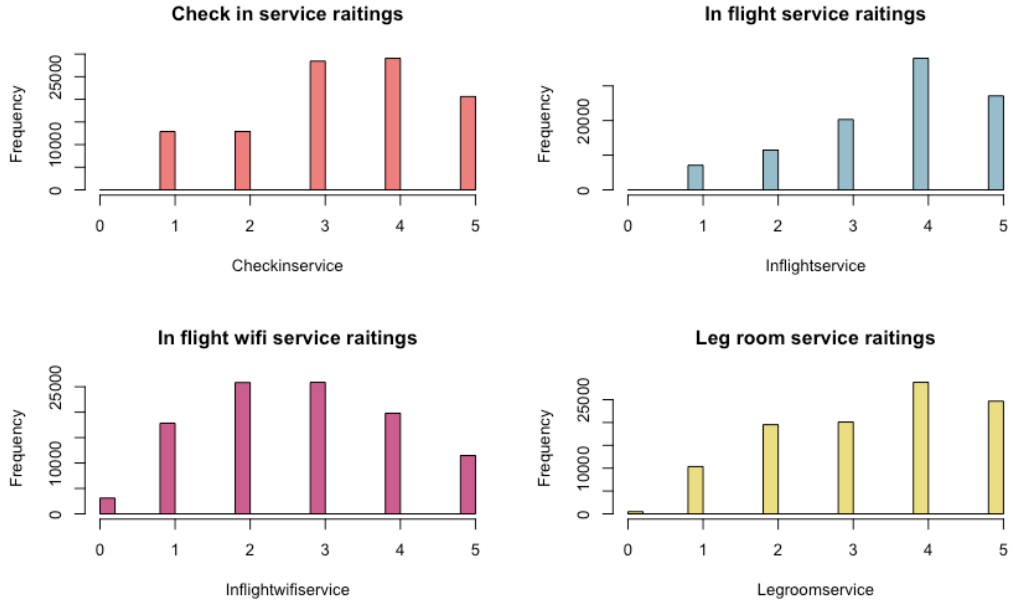
4) Different purpose of travel Vs Satisfaction



- Passengers who are traveling for business purposes have more satisfaction compared to those who are traveling for other purposes. Like people who are having a holiday have half the satisfaction of that of those who are travelling for business purposes.

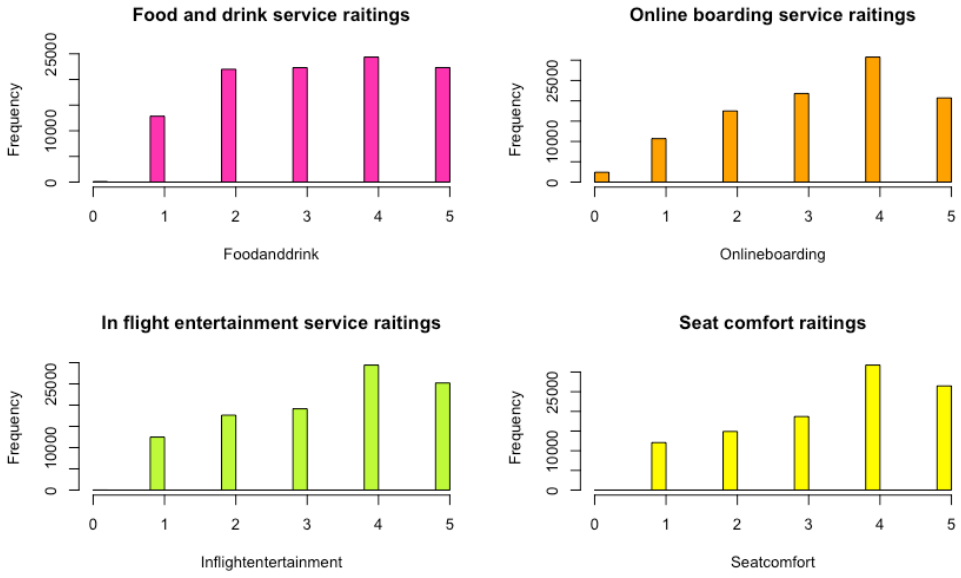
Name: Pragati Koladiya
NUID: 001029445

5) Distribution of different variables of flight service



- The flight service ratings on the scale of 0 to 5 has been plotted in the above graphs.
    - Customers of the airline have given rating 3 and 4 rating to check in services which means the service is moderate.
    - In-flight services like serving food or responding immediately when customers need help, has the highest rating 4 which is pretty good.
    - Wifi service of the airline seems moderate as most of the service rating is between 2 and 3.
    - Leg room service has the highest rating 4 which shows people are impressed by the leg room.

6) Price Vs Stories



- The remaining four flight service ratings on the scale of 0 to 5 has been plotted in the above graphs.
    - Airline's food and drink services are moderate to high as the highest rating range between 2 and 5.

- Online boarding service has the maximum rating of 4 which shows this service is not bad.
- Flight entertainment service is the heart as people may get bored without it. As customers seem happy with it.
- Majority of passengers are comfortable with their seats.

Data Preparation:

The first step is to check for null values in the dataset.

```
#checking data contains any missing values or not
sum(is.na(Air_Num_Data))
#Removing null values
Air_Num_Data <- na.omit(Air_Num_Data)
#again printing sum of null values
sum(is.na(Air_Num_Data))
```

```
> sum(is.na(Air_Num_Data))
[1] 310
> Air_Num_Data <- na.omit(Air_Num_Data)
> sum(is.na(Air_Num_Data))
[1] 0
```

- We see that the sum of null values is 310 which is using the function `na.omit()`. After that when we take the sum of null, it shows 0 which means there are no null values in our dataset.

The dataset has many columns with categorical values. In order to fit the KNN model, we will need numerical values and not string. Hence, we convert them to 1s and 0s, where 0 is a 'Male' and 1 is a 'Female'.

```
air$Gender <- factor(air$Gender, levels=c('Male','Female'),labels=c(0,1))
table(air$Gender)
typeof(air$Gender)
```

- To convert string to number factor() function is used. It stores the categorical values as a vector of integers in the range [1... k], (where k is the number of unique values in the nominal variable). Here 0 and 1 are nominal values and an internal vector of character strings (the original values) like male and female, mapped to these integers.

```
Air_Num_Data <- as.data.frame(apply(air, 2, as.numeric))
Air_Num_Data <- Air_Num_Data[,-1]
```

- First line is used to convert the string data type to numerical after categorical string values to numerical, stored in the `Air_Num_Data`
- Second line of code is to drop the first column name No. It shows the sequence of the numbers which is not important for further analysis.

Course: ALY6020 Name: Pragati Koladiya
Term: A_Fall_2020 NUID: 001029445

Before moving further let's consider the correlation plot which allows highlighting the variable that is most (positively or negatively) correlated.
As we are going to predict the satisfaction of the customers, knowing the correlation between each variable is very important. This will help the model to predict the price with higher accuracy.

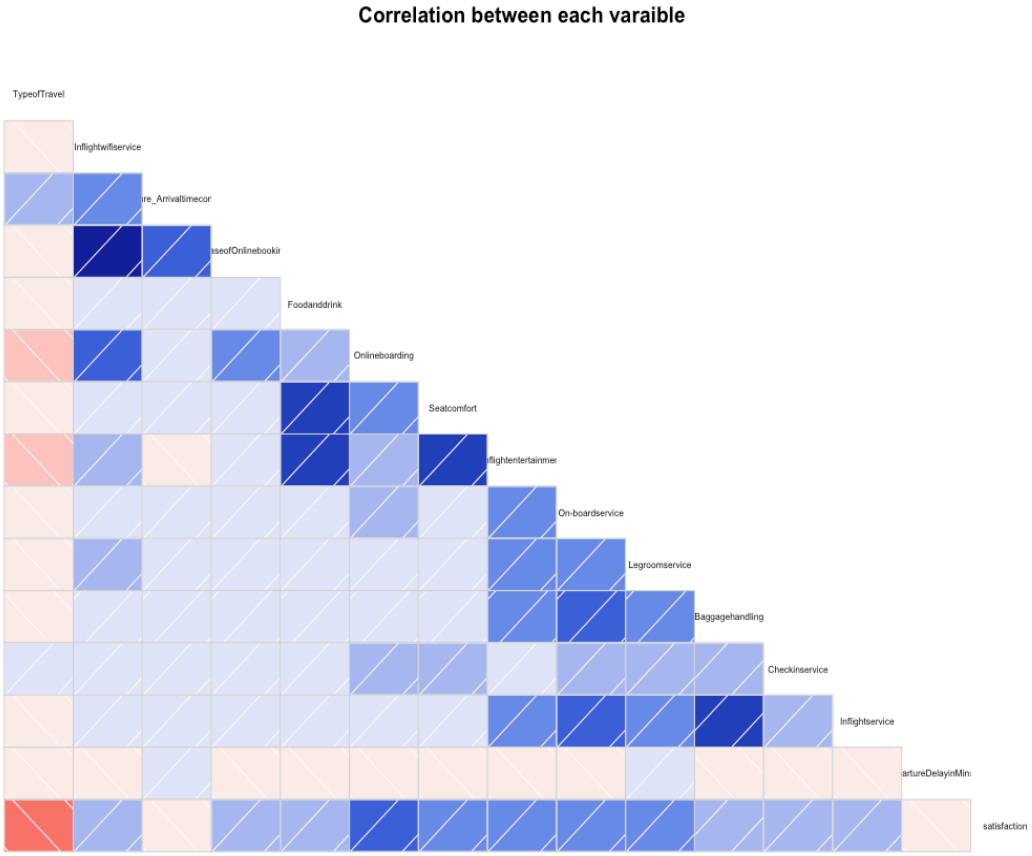
- Here I have plotted correlation twice before dropping and after dropping the column. The code for a list of columns has been dropped because of its negative and less correlation.

```
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'id']
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'Gender']
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'CustomerType']
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'Age']
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'Class']
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'FlightDistance']
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'Gatelocation']
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'ArrivalDelayinMinutes']
Air_Num_Data =Air_Num_Data[,!names(Air_Num_Data) %in% 'Cleanliness']
```

- Head of the dataset after dropping above columns

```
> head(Air_Num_Data)
  TypeofTravel Inflightwifiservice Departure_Arrivaltimeconvenient EaseofOnlinebooking Foodanddrink Onlineboarding Seatcomfort Inflightentertainment On-boardservice Legroomservice Baggagehandling
1            1                   3                              4                   3             5              3           5                     5              4              3                4
2            0                   3                              2                   3             1              3           1                     1              1              5                3
3            0                   2                              2                   2             5              5           5                     5              4              3                4
4            0                   2                              5                   5             2              2           2                     2              2              5                3
5            0                   3                              3                   3             4              5           5                     3              3              4                4
6            1                   3                              4                   2             1              2           1                     1              3              4                4
  Checkinservice Inflightservice DepartureDelayinMinutes satisfaction
1              4                5                      25            0
2              1                4                       1            0
3              4                4                       0            1
4              1                4                      11            0
5              3                3                       0            1
6              4                4                       0            0
```

- Correlation graph after dropping few columns



- Darker the grid higher the correlation.
- The positive correlation has been observed between –
  - satisfaction and online boarding
- The negative correlation is in a lighter shade of pink.
- Few of them are,
  - Type of travel and satisfaction,
  - Departure arrival time connivence.

KNN Model Implemented:

In this step, we will implement the KNN classification model that will be appropriate for the given dataset. As discussed earlier the aim is to predict the satisfaction for the customer. Considering that independent variables(X) would be all the features(after dropping a few columns) except the satisfaction. The dependent variable(Y) will be price.

As the dependent variable is categorical we will implement a KNN classification model that falls under the category of KNN model.

KNN Model:

The first step in the model implementation is to split the dataset to eliminate the bias to training data in machine learning algorithms.

- Split the dataset into train and test sets with ratio 70/30.

```
> set.seed(3033)
> intrain <- createDataPartition(y = Air_Num_Data$satisfaction, p= 0.7, list = FALSE)
> training <- Air_Num_Data[intrain,]
> testing <-  Air_Num_Data[-intrain,]
> dim(training)
[1] 72516    15
> dim(testing)
[1] 31078    15
```

- Train set has 72516x15 and the test set has 31078x15. The data is split into a 70:30 ratio.

The caret package provides a method createDataPartition() for partitioning our data into train and test sets. We are passing 3 parameters. The "y" parameter takes the value of the variable according to which data needs to be partitioned. In our case, the target variable is at satisfaction, so we are passing Air_Num_Data$satisfaction.

Creating separate data frame for 'satisfaction' feature which is our target:

```
> training_labels <- Air_Num_Data[intrain,1]
> testing_labels  <-Air_Num_Data[-intrain,1]
> library(class)
> knn.5 <- knn(train=training, test=testing, cl=training_labels, k=5)
> ACC.5 <- 100 * sum(testing_labels == knn.5)/NROW(testing_labels)
```

- The basic function for fitting a KNN model is *knn(formula, data)*.
- The line training <- df[intrain,] is for putting the data from data frame to training data
- Remaining data is saved in the testing data frame, testing <- df[-intrain,].
- After building the model, it is time to calculate the accuracy of the created model

Name: Pragati Koladiya
NUID: 001029445

name knn.5.

```
> ACC.5
[1] 92.03617
> table(knn.5 ,testing_labels)
     testing_labels
knn.5     0     1
    0 20745  1732
    1   743  7858
```

- As shown in the above snippet, the accuracy for k=5 is 92.04, which is good since have moderate dataset.

```
> View(data.frame(knn.5, testing_labels))
```

- Viewing the predicted values generated by KNN. Here, column name `knn.5` is the prediction of satisfaction which classifies the results into 0 and 1. The column for given data column name `testing_labels` which makes observers task easer.

Filter

	knn.5	testing_labels
1	0	0
2	1	1
3	0	0
4	0	0
5	0	1
6	0	0
7	0	0
8	0	1

- The above table shows first few column of the entire table.

- *Confusion Matrix:* A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. The *below table* shows the confusion matrix statistics.

Name: Pragati Koladiya
NUID: 001029445

```
> confusionMatrix(table(knn.5 ,testing_labels))
Confusion Matrix and Statistics

     testing_labels
knn.5     0     1
    0 20745  1732
    1   743  7858

               Accuracy : 0.9204
                 95% CI : (0.9173, 0.9233)
    No Information Rate : 0.6914
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8079

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9654
            Specificity : 0.8194
         Pos Pred Value : 0.9229
         Neg Pred Value : 0.9136
             Prevalence : 0.6914
         Detection Rate : 0.6675
   Detection Prevalence : 0.7232
      Balanced Accuracy : 0.8924

       'Positive' Class : 0
```

- As we see the accuracy of the above predicted results seems to be 92% with 95% ci: is in 1st standard deviation.
- For more clear understanding I have plotted a confusion matrix table which shows the detail row and column count with total observation.

```
> CrossTable(x = testing_labels, y = knn.5,
+            prop.chisq = FALSE)


   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  31078


               | knn.5
testing_labels |         0 |         1 | Row Total |
---------------|-----------|-----------|-----------|
             0 |     20745 |       743 |     21488 |
               |     0.965 |     0.035 |     0.691 |
               |     0.923 |     0.086 |           |
               |     0.668 |     0.024 |           |
---------------|-----------|-----------|-----------|
             1 |      1732 |      7858 |      9590 |
               |     0.181 |     0.819 |     0.309 |
               |     0.077 |     0.914 |           |
               |     0.056 |     0.253 |           |
---------------|-----------|-----------|-----------|
  Column Total |     22477 |      8601 |     31078 |
               |     0.723 |     0.277 |           |
---------------|-----------|-----------|-----------|
```

- True positives (TP): Correctly predicted values.
    - The first row and first column shows the true positive (TP) cases, means the customers that already unsatisfied and KNN predicts they are unsatisfied with the airline.
- True negatives (TN): Correctly rejected the prediction.

  - The second row and first column is satisfied customers in real world but KNN predict they are not satisfied(FP).
- False positives (FP): We predicted yes, but the correct answer is no.
- False negatives (FN): We predicted no, but the correct answer is yes.
  - last column and last row is False Negative (FN) that means customers who are satisfied and KNN predict as they are actually satisfied.

After viewing the confusion matric we can calculate the following ,

- The ***precision*** is intuitively the ability of the classifier to not label a sample as positive if it is negative.
  - The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives.
- The ***recall*** is intuitively the ability of the classifier to find all the positive samples.
  - The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives.
- F measure: F-Measure provides a single score that balances both the concerns of precision and recall in one number.
  - F1 =2 * Precision * Recall/(Precision + Recall)

```
> Recall = (7858)/(7858+743)
> Recall
[1] 0.9136147
> Precision = (7858)/(7858+1732)
> Precision
[1] 0.8193952
> F.measure = (2*Recall*Precision)/(Recall+Precision)
> F.measure
[1] 0.8639437
```

Lastly, before deciding that the knn had predicted well with the highest accuracy for k value 3. It is important to know the optimum k value for our model.

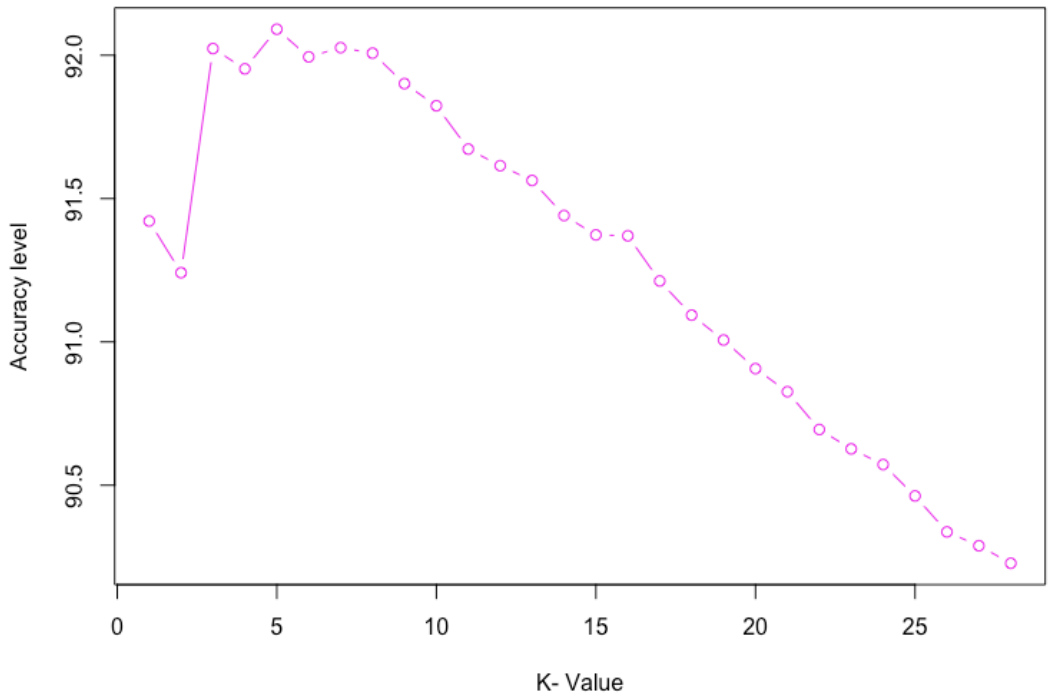
```
i=1                          #initialization of variable
k.optm=1                      #initialization of variable
for (i in 1:28){
  knn.mod <-  knn(train=training, test=testing, cl=training_labels, k=i)
  k.optm[i] <- 100 * sum(testing_labels == knn.mod)/NROW(testing_labels)
  k=i
  cat(k,'=',k.optm[i],'\n')       # to print % accuracy
}
#=> maximum accuracy  92.20671 was achieved for k=3
```

Here this for loop will give the accuracy score for 1 to 28 different k values and then we will plot to visualize the optimum values.

Name: Pragati Koladiya
NUID: 001029445

```
1 = 91.42158
2 = 91.24139
3 = 92.0233
4 = 91.95251
5 = 92.09087
6 = 91.99434
7 = 92.02651
8 = 92.00721
9 = 91.90102
10 = 91.8238
11 = 91.67257
12 = 91.61465
13 = 91.56316
14 = 91.44089
15 = 91.37332
16 = 91.3701
17 = 91.21243
18 = 91.09338
19 = 91.0065
20 = 90.90675
21 = 90.82631
22 = 90.69438
23 = 90.62681
24 = 90.57211
25 = 90.46271
26 = 90.33722
27 = 90.28895
28 = 90.22781
```

- Above list shows the list of optimum k values. The k values with highest accuracy will be considered as optimum.



After plotting all different k values from 1 to 28. We can say that the optimum k value for the given data set will be **k=5.**

Course: ALY6020 Name: Pragati Koladiya
Term: A_Fall_2020 NUID: 001029445

Time Series Forecasting:

Time series analysis is a statistical technique that deals with time series data, or trend analysis. Time series data means that data is in a series of particular time periods or intervals.

Problem Statement:

Consider a electricity production company that has a dataset containing the date of electric production and Industrial Production: Electric and gas utilities(IPG2211A2N). It wishes to use the data to analyse the pattern in the production through 1985 to 2018. the electricity demand need to be forecasted using historical data.

Essentially, the company wants,

- To forecast the power consumption that quantitatively relates demand.
- To answer question like what will be the future demand of power consumption? How much power will be need in the coming five years?
-

Data Acquisition:

The dataset is chosen from Kaggle with file name "Electric_Production". This dataset contains only two columns name date and IPG2211A2N(Industrial Production: Electric and gas utilities).

The dataset contains 397 observations and 2 attributes related to electric production.

Data Dictionary:

- The data dictionary is not provided by the data sourse as the column names are easily understood.

Date – dates from JAN-1985 to JAN-2018
IPG2211A2N - Industrial Production: Electric and gas utilities

Data Overview:

Required packages have been imported before loading the dataset into R.

- Head of the dataset with columns and first few rows.

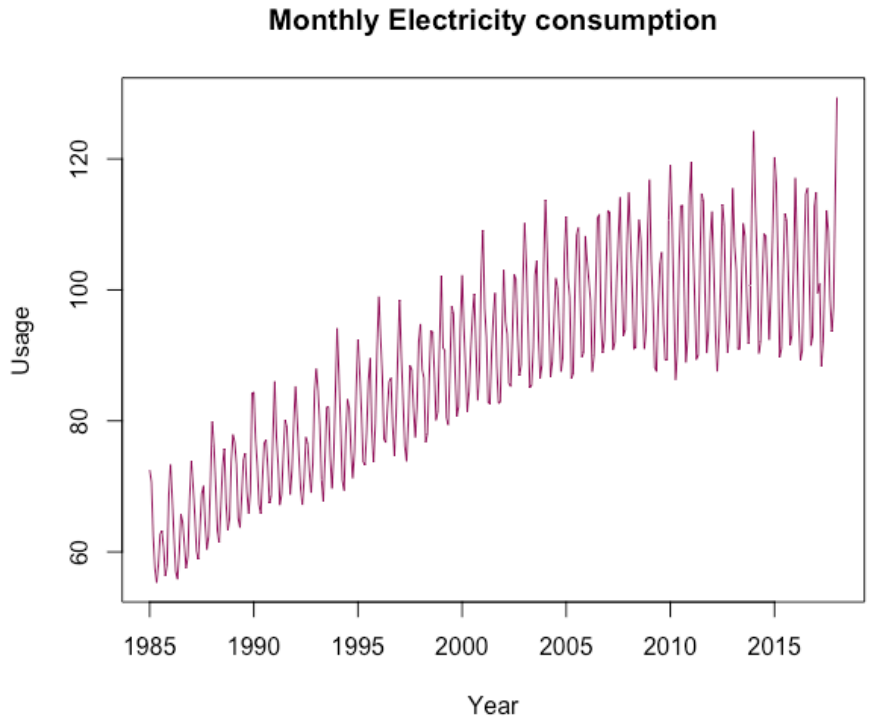
```
> head(ep)
# A tibble: 6 x 2
  DATE     IPG2211A2N
  <chr>         <dbl>
1 1/1/1985       72.5
2 2/1/1985       70.7
3 3/1/1985       62.5
4 4/1/1985       57.5
5 5/1/1985       55.3
6 6/1/1985       58.1
```

- After converting data frame into time series object it will looks like as below,

```
> ep.ts <- ts(ep,frequency=12,start = c(1985,1),end=c(2018,1))
> ep.ts.qtr <- aggregate(ep.ts,nfrequency=4)
> ep.ts.yr <- aggregate(ep.ts,nfrequency=1)
```

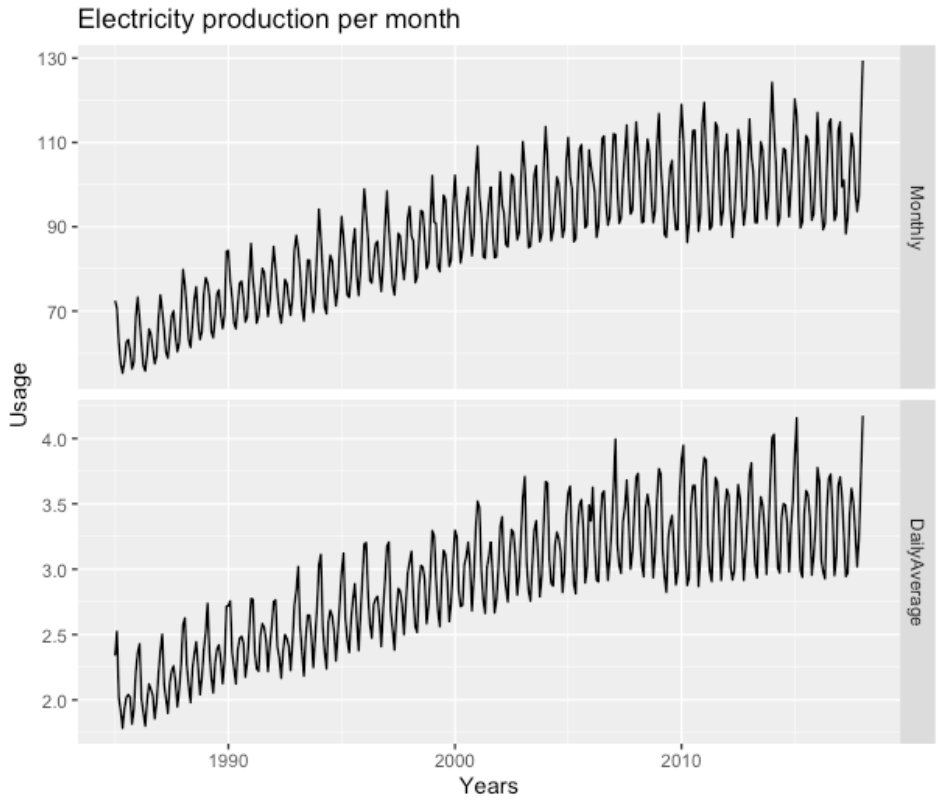
Name: Pragati Koladiya
NUID: 001029445

- Plotting the time series for electricity consumption yearly



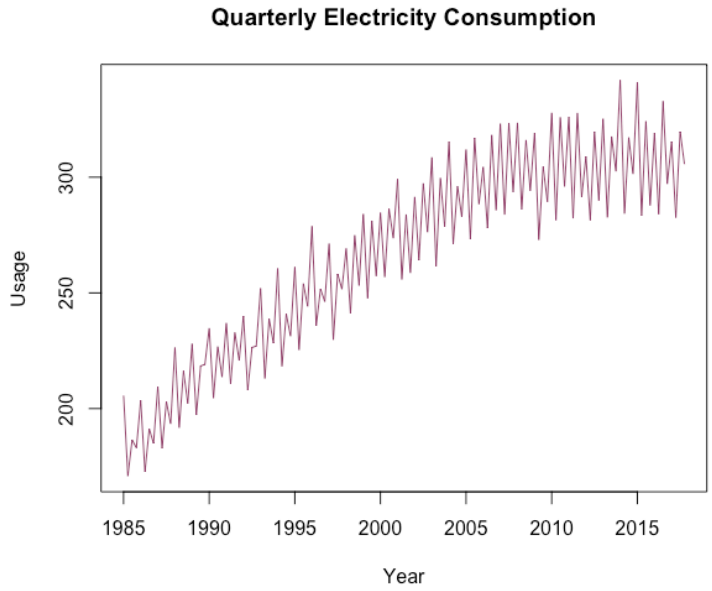
Some of the variation seen in seasonal data may be due to simple calendar effects. In such cases, it is usually much easier to remove the variation before fitting a forecasting model. The monthdays() function will compute the number of days in each month or quarter.

```
> df <- cbind(Monthly = ep.ts[,2],DailyAverage = ep.ts[,2]/monthdays(ep.ts[,2]))
> autoplot(df,facet=TRUE) + xlab("Years")+ylab("Usage")+ggtitle("Electricity production per month")
```

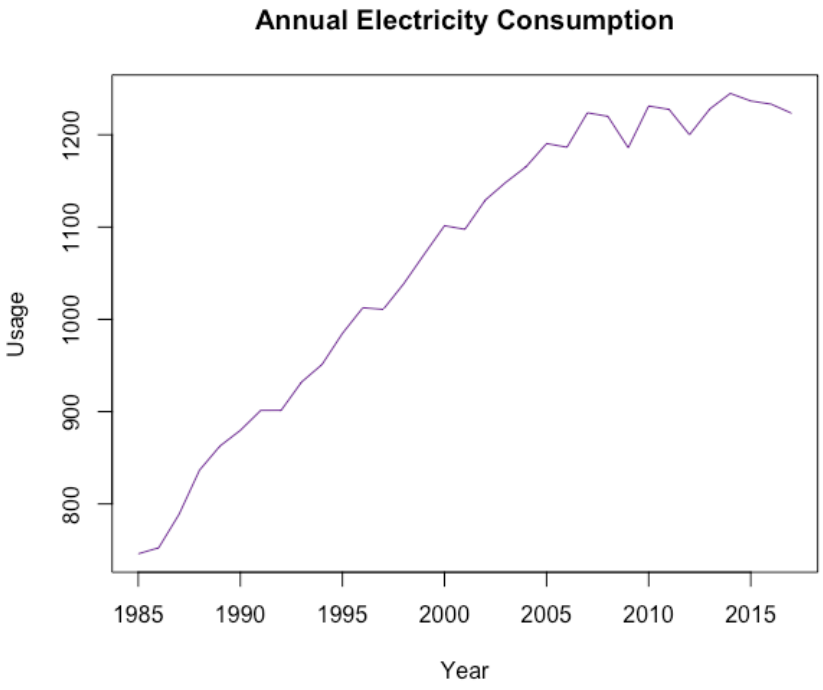


There is no significant effect of variation due to varying days in a month, since both the plots look similar. Forecasting horizon is daily, weekly, quarterly, monthly or annual data prediction is based on the requirement of the problem. For example, stock prices may need daily prediction, but amount of rainfall requires only annual prediction. An hourly data can have daily, weekly, monthly, quarterly and annual seasonality. if we have only 180 days of data, we can ignore annual seasonality.

Quarterly plot:



Annual Plot:

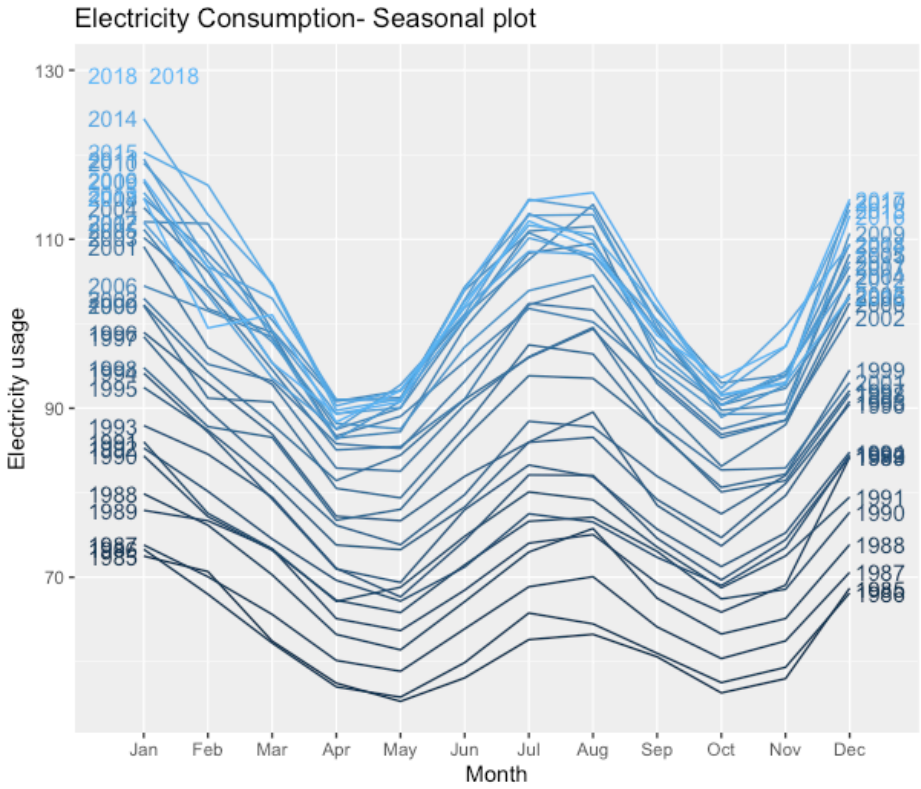


- As we can see there is a strong growth from 1985-2005, then the consumption was stable.

Course: ALY6020
Term: A_Fall_2020

Name: Pragati Koladiya
NUID: 001029445

Seasonal Plot:



- We can clearly see a strong seasonal pattern yearly in the time series.

Course: ALY6020
Term: A_Fall_2020

Name: Pragati Koladiya
NUID: 001029445

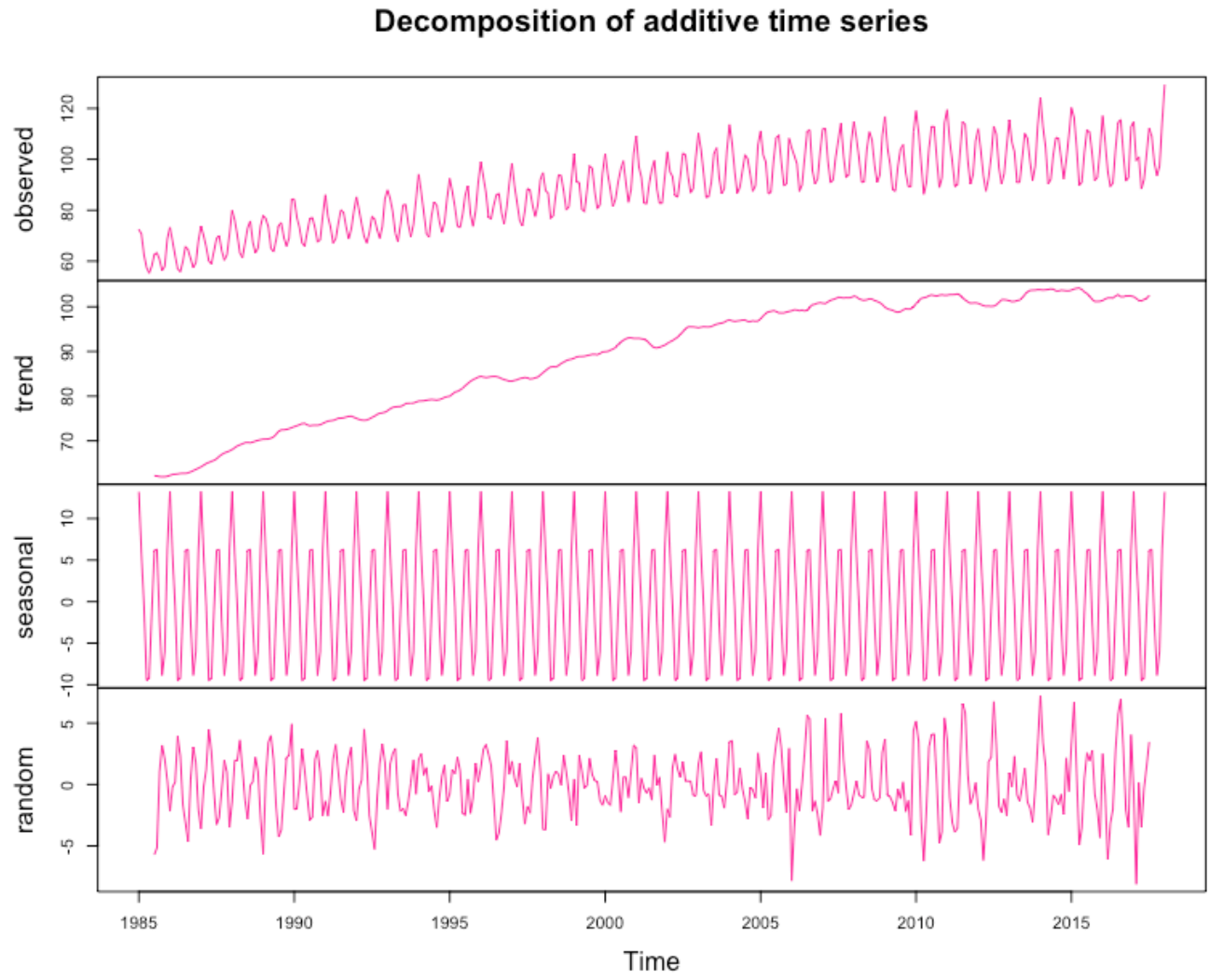
Additive decomposition:

Formula : $yt=Tt+St+Rt$

where T - Trend-cycle component,S - Seasonal component and R- Remainder component.

The additive decomposition is the most appropriate if the magnitude of the seasonal fluctuations, or the variation around the trend-cycle, does not vary with the level of the time series.

```
> x<-decompose(ep.ts[,2],type = "additive")
> plot(x,col=c("deeppink1"))
```

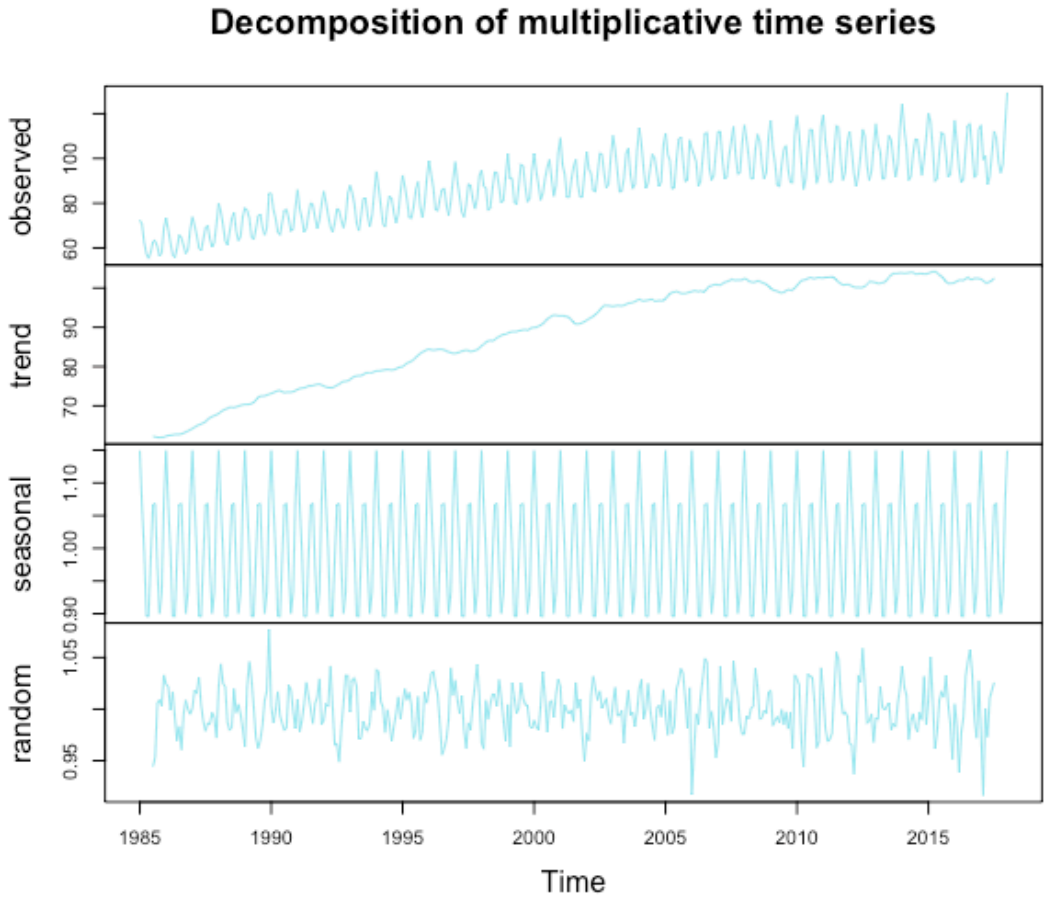


Name: Pragati Koladiya
NUID: 001029445

Multiplicative:

```
x<-decompose(ep.ts[,2],type = "multiplicative")
plot(x,col=c("cadetblue2"))
```

Multiplicative decomposition is more appropriate when the variation in the seasonal pattern, or the variation around the trend-cycle, appears to be proportional to the level of the time series.

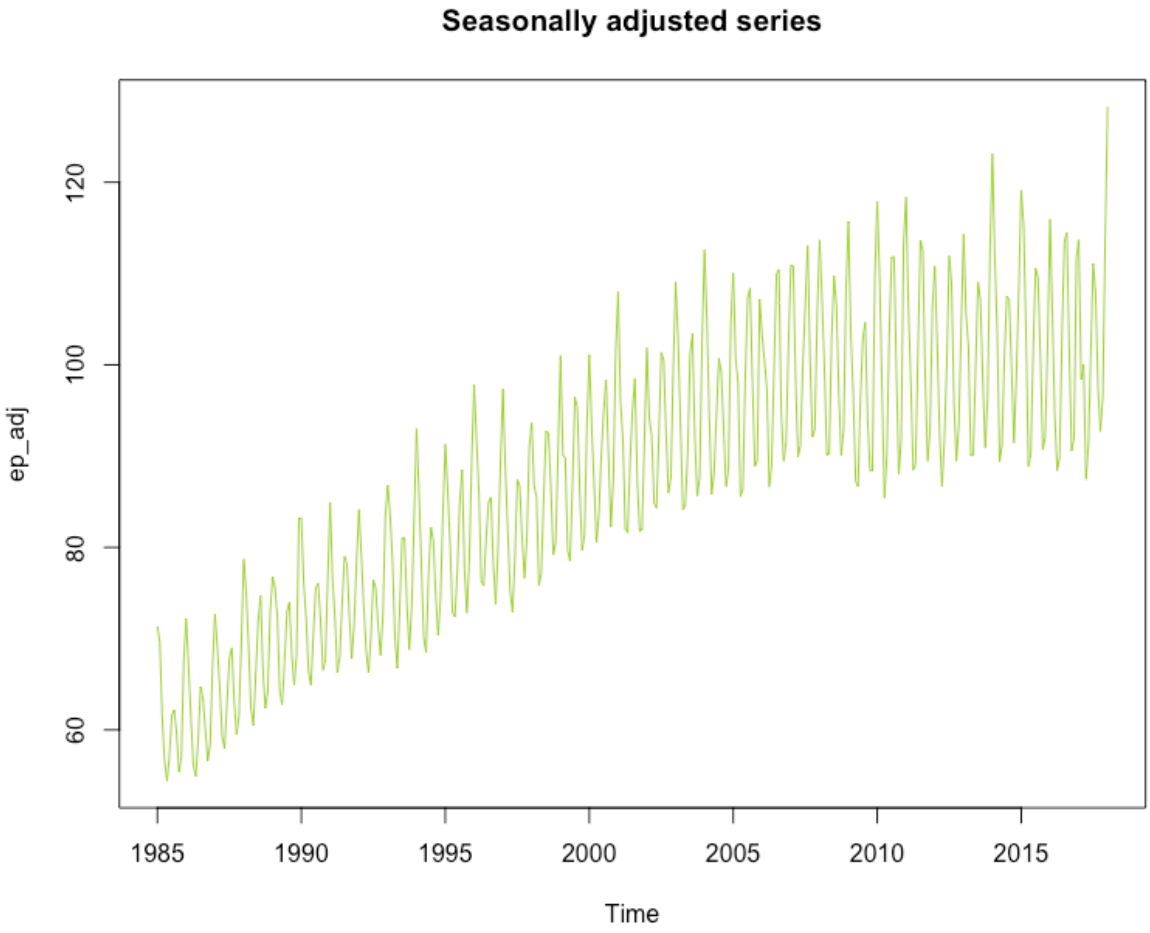


Course: ALY6020
Term: A_Fall_2020

Name: Pragati Koladiya
NUID: 001029445

Seasonality adjusted for further implementation

```
#seasonality adjusted
ep_adj <- ep.ts[,2] - x$seasonal
plot.ts(ep_adj,col=c(rep("olivedrab1"))) + title('Seasonally adjusted series')
```



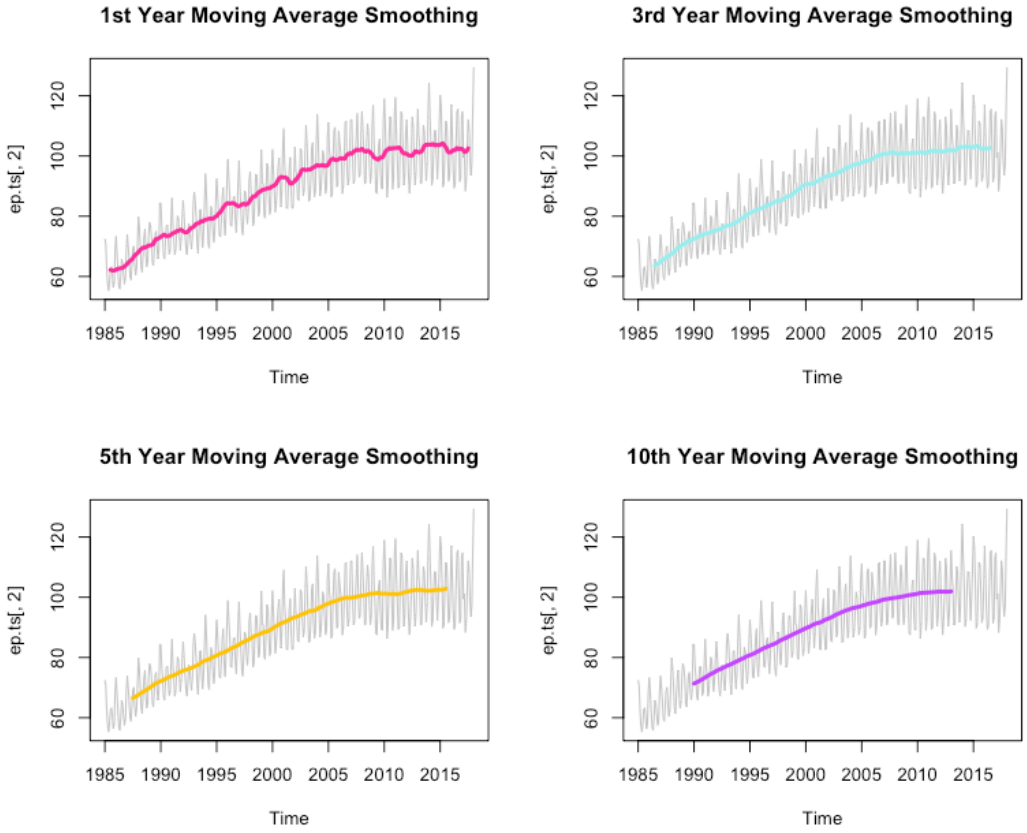
- For electricity consumption, we need seasonal component for analysis. So, we don't use seasonality adjustment. Similarly, we can define de-trended series for both additive and multiplicative models.

Course: ALY6020
Term: A_Fall_2020

Name: Pragati Koladiya
NUID: 001029445

Moving Average:

Moving average is used to estimate the trend-cycle.

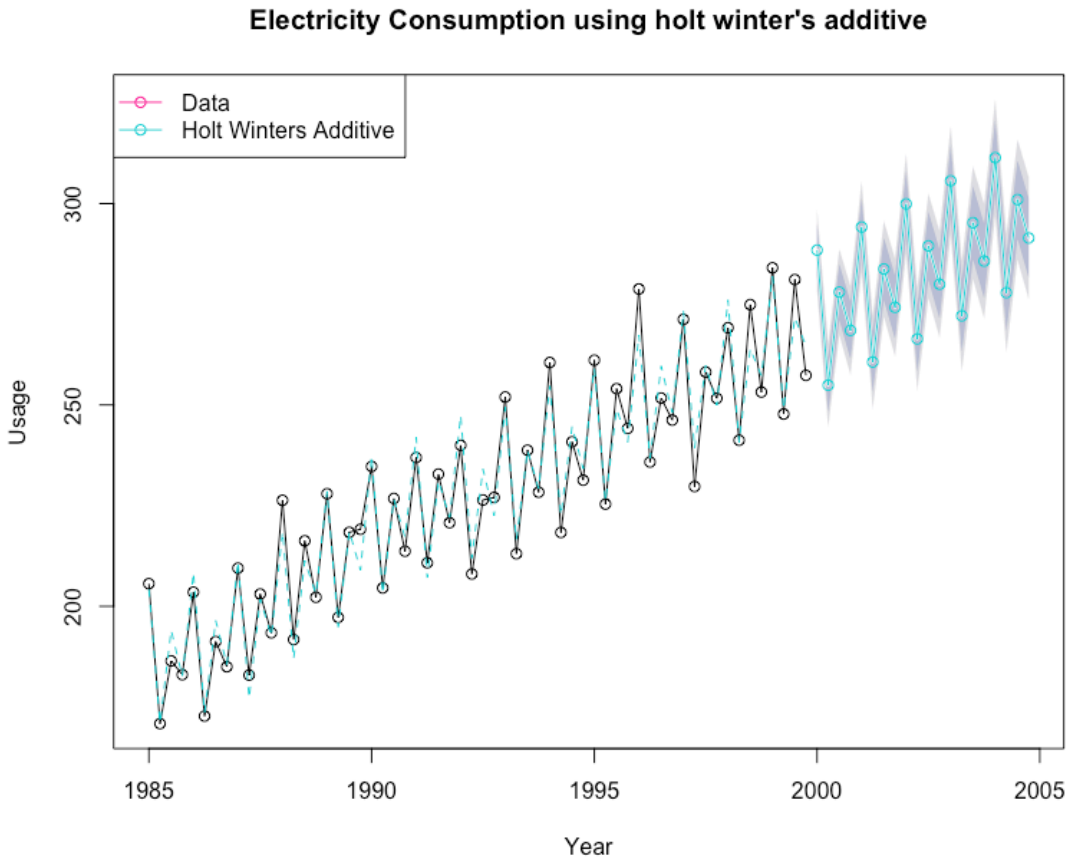


Trend Method:

- There are mainly three different methods for trend here I have implemented Holt winter's seasonal method. this model captures all the components of time series, level,trend and season.
- The additive model is the most appropriate if the magnitude of the seasonal fluctuations, or the variation around the trend-cycle, does not vary with the level of the time series.
- Multiplicative decomposition is more appropriate when the variation in the seasonal pattern, or the variation around the trend-cycle, appears to be proportional to the level of the time series.

```
#Winter holt's additive metod
ep.ts3 <- window(ep.ts, start = 1985, end = 2000)
ep.ts.qtr <- aggregate(ep.ts3, nfrequency=4)
ep.fit.hw <- hw(ep.ts.qtr[,2], h = 20, seasonal = "additive")
```

Name: Pragati Koladiya
NUID: 001029445



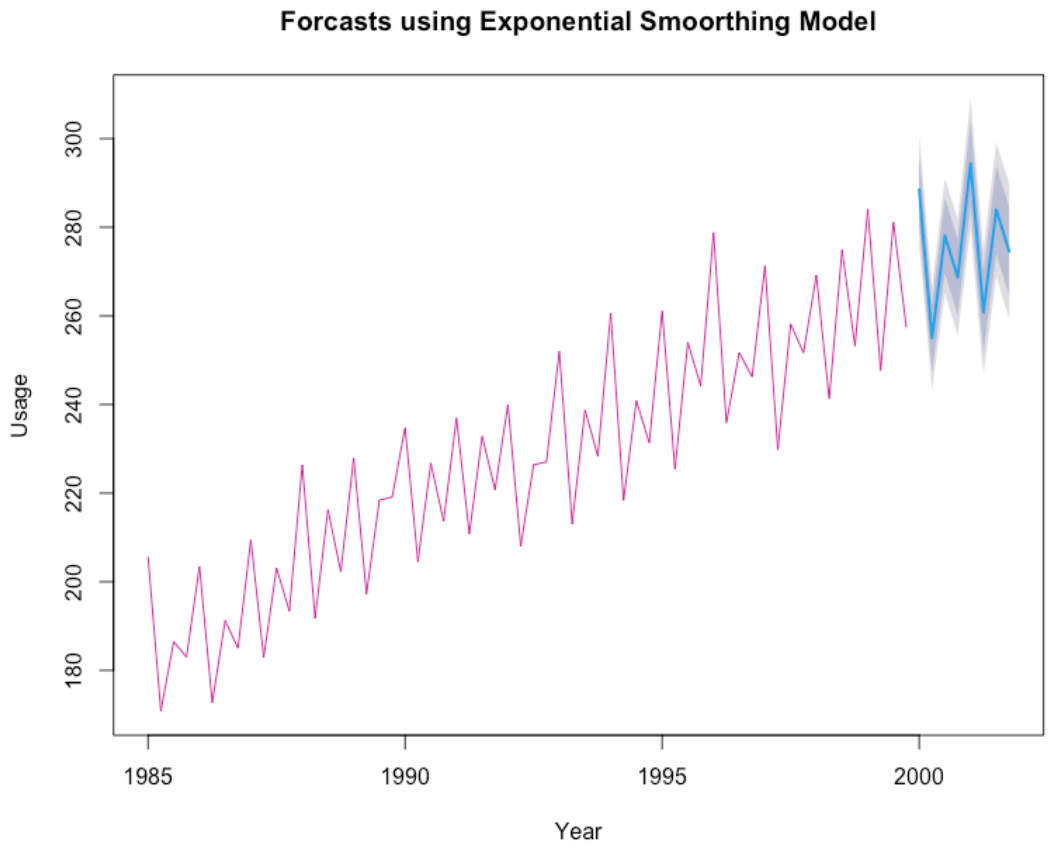
- You can see from the plat that there is roughly constant variance over time, thus we can describe this as an additive model, thus we can make forecasts using simple exponential smoothing.

Course: ALY6020
Term: A_Fall_2020

Name: Pragati Koladiya
NUID: 001029445

Forecasting Using Exponential Smoothing

This model finds the best possible combination of components from the exponential smoothing models. People often feel overwhelmed by which model to use since there are 30 different models to choose from. The ETS function in R helps us to estimate the best model which is the one with least AIC score.




```
> summary(ep.fit.ets)
ETS(M,A,A)

Call:
 ets(y = ep.ts.qtr[, 2])

  Smoothing parameters:
    alpha = 0.3136
    beta  = 1e-04
    gamma = 1e-04

  Initial states:
    l = 184.4619
    b = 1.4681
    s = -6.0621 4.786 -16.8399 18.116

  sigma:  0.0217

     AIC     AICc      BIC
446.0969 449.6969 464.9460

Training set error measures:
                     ME     RMSE      MAE        MPE     MAPE      MASE        ACF1
Training set -0.1923103 4.750744 3.707461 -0.1148442 1.596727 0.5270661 -0.03780257
```

A we can see that ETS estimate ETS(M,A,A) for us which is multiplicative errors with Additive Holt-Winters' method. We can also fit our own model and compare to the estimate model.

ARIMA MODEL

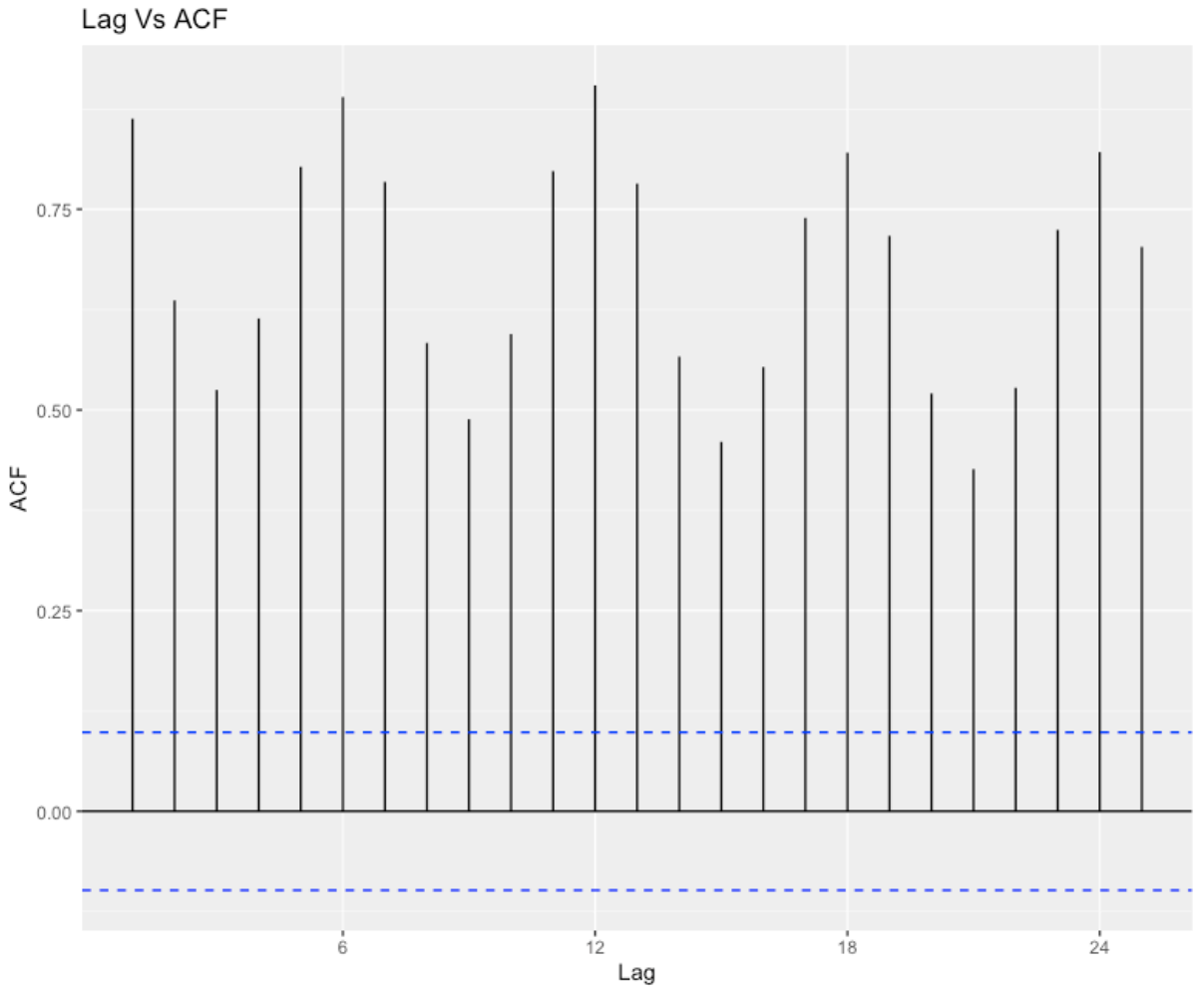
Exponential Smoothing and ARIMA are two most widely used approaches in time series forecasting.

ARIMA model aims to describe the autocorrelation in the data.

The autocorrelation coefficients are plotted to show the autocorrelation function or ACF. The plot is also known as a correlogram.

ACF Plot:

The autocorrelation coefficients are plotted to show the autocorrelation function or ACF. The plot is also known as a correlogram.



When data have a trend, the autocorrelations for small lags tend to be large and positive because observations nearby in time are also nearby in size. So the ACF of trended time series tend to have positive values that slowly decrease as the lags increase.

When data are seasonal, the autocorrelations will be larger for the seasonal lags (at multiples of the seasonal frequency) than for other lags.

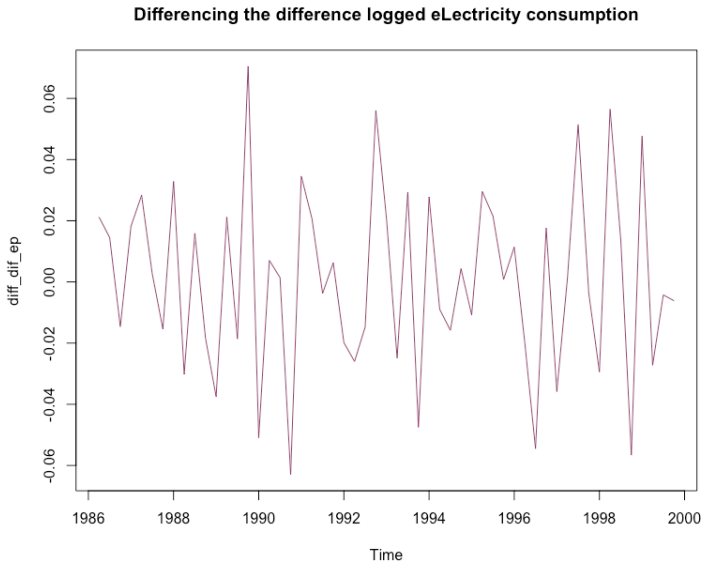
When data are both trended and seasonal, you see a combination of these effects

The dashed blue lines indicate whether the correlations are significantly different from zero.

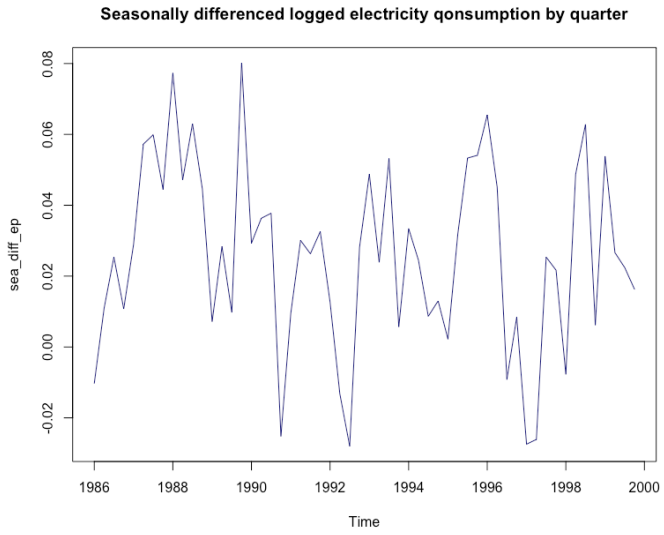
Differencing:

Transformations such as logarithms can help to stabilise the variance of a time series. Differencing can help stabilise the mean of a time series by removing changes in the level of a time series, and therefore eliminating (or reducing) trend and seasonality.

Differencing computes the differences between consecutive observations. By differencing the time series data, we can remove the trend and seasonality.



- The first differencing removed the trend but we can still see some seasonality affect. By doing another differencing with lag 12, we now removed the seasonality. Time series now should be now stationary. We can also see that from ACF charts.



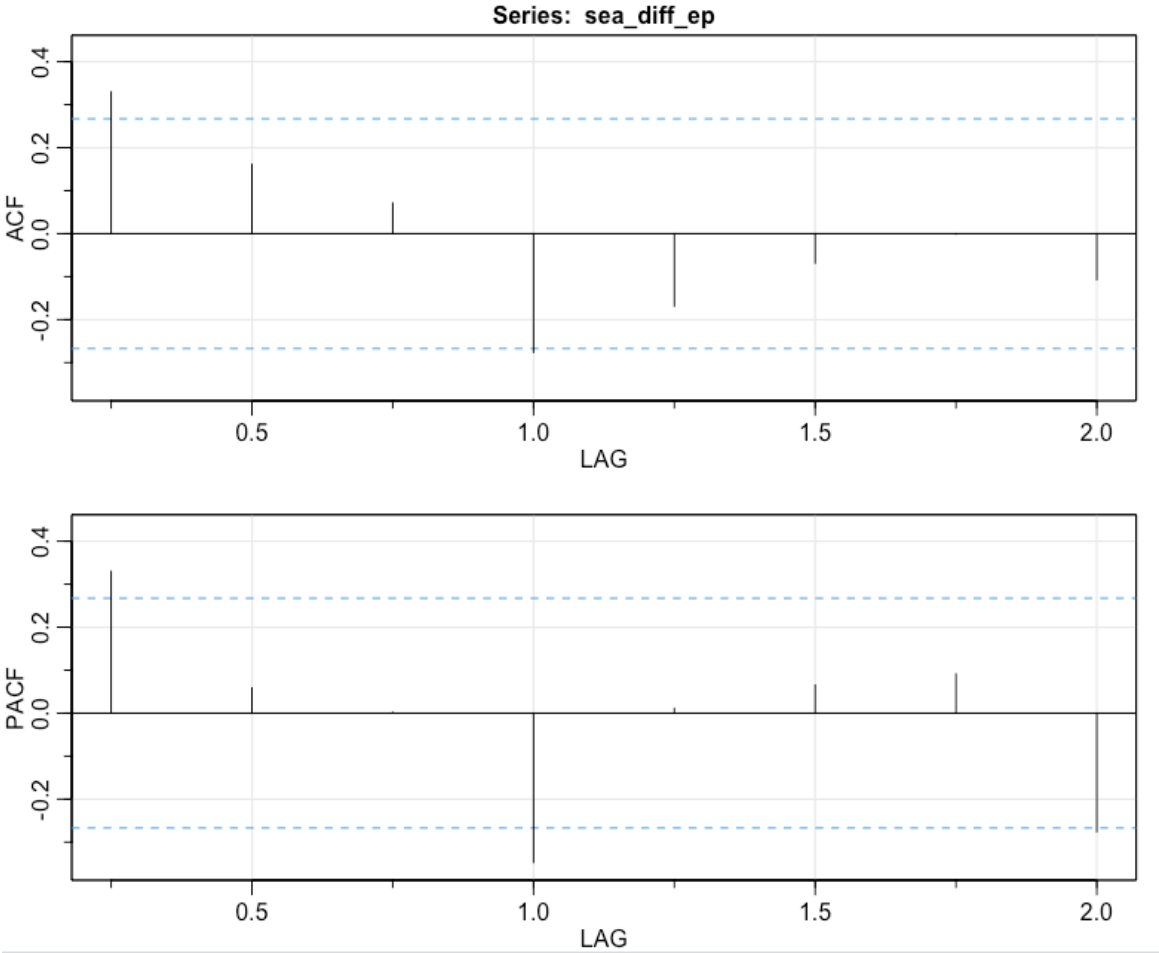
Course: ALY6020
Term: A_Fall_2020

Name: Pragati Koladiya
NUID: 001029445

Matrix of Acf

```
> acf2(diffep)
      [,1]  [,2]  [,3] [,4]  [,5]  [,6]  [,7] [,8]  [,9]
ACF  -0.82  0.67 -0.78 0.89 -0.75  0.62 -0.72 0.82 -0.71
PACF -0.82 -0.02 -0.74 0.30  0.22 -0.13 -0.15 0.08 -0.03
> acf2(diff_dif_ep)
      [,1]  [,2] [,3]  [,4]  [,5]  [,6] [,7]  [,8]
ACF  -0.37 -0.05 0.19 -0.35  0.04  0.02 0.11 -0.16
PACF -0.37 -0.21 0.11 -0.28 -0.22 -0.17 0.16 -0.19
```

Above matric is use to plot ACF is an (complete) auto-correlation function which gives us values of auto-correlation of any series with its lagged values .



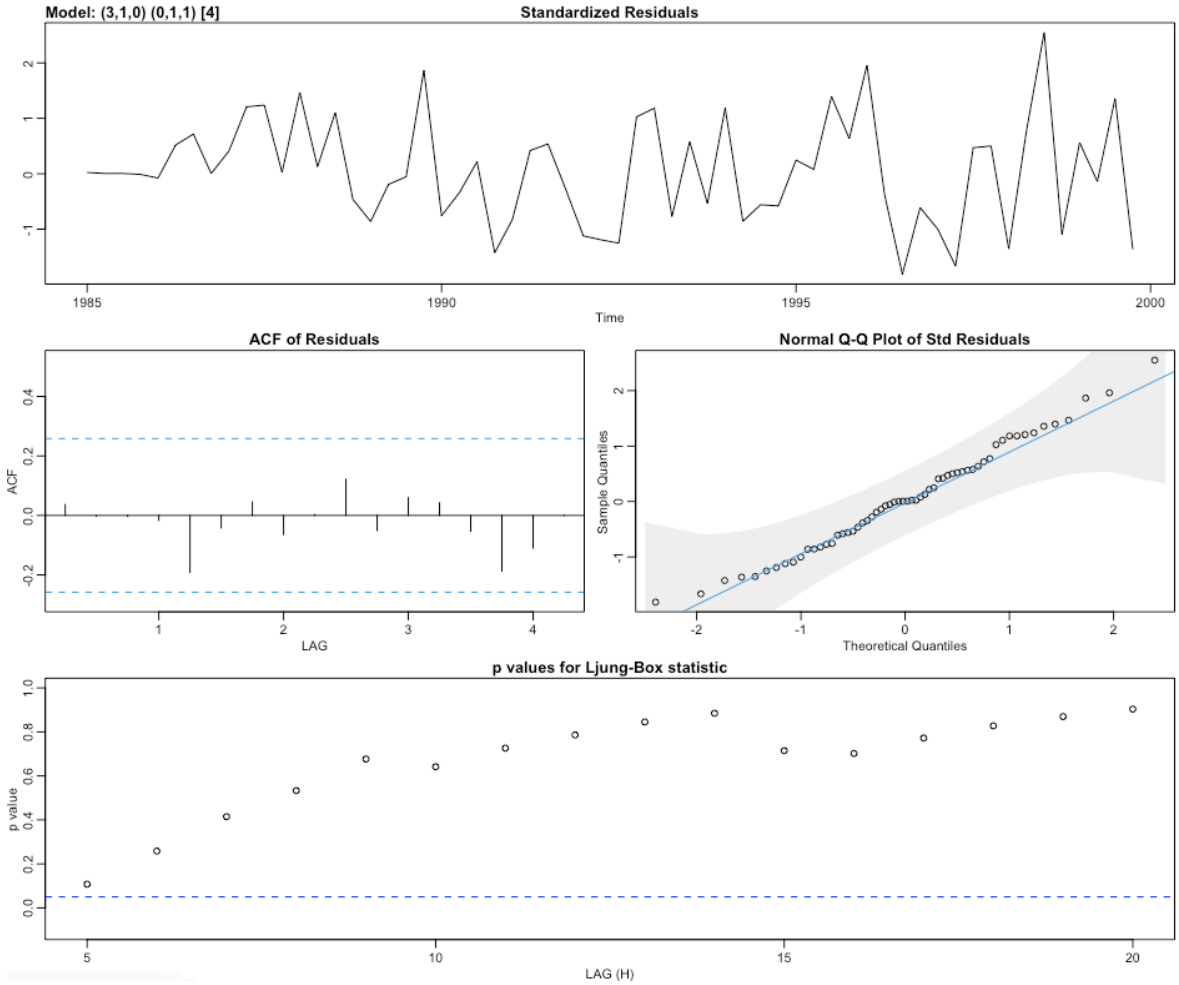
From the above plot, we can see that there is a exponential decay in ACF and after each lags, there is a statistical significant spike in PACF. *ARIMA*(3,1,0)(0,1,1).

By looking at the residual diagnostics, it looks like we have a workable model here since the residuals seem normally distributed,

ACF of the residuals are within 95% confidence interval. Therefore, we can now use this ARIMA model to forecast.

Course: ALY6020
Term: A_Fall_2020

Name: Pragati Koladiya
NUID: 001029445



Discussion:

What makes the problem interesting from the viewpoint of analytics?

- Analytics boils down to drawing conclusions from a given dataset. Learning about the customer satisfaction for a particular airline uses KNN classification. KNN classification does not require any parameters. It works by finding the Manhattan distance of the nearest neighbours. Then classifies on the basis of closest query. Such an algorithm gives a quality and a reliable output

How did the chosen technique help to illuminate, or solve the problem?

- We have visualized the data and implemented two different models with almost the same accuracy. This shows that even if we consider all the features during modelling only those features give a good impact on the model which has a higher correlation with the dependent variable. In order to predict the bidding price of the house we can use an advanced model which may improve our r score as it is low. This particular problem is a hands-on case for any data analyst. We get to implement what we learn in the theory and seeing it work and helpful in the market gives satisfaction to the person working on it.

What analysis do you think should be conducted next?

- Applying the analogy for timeseries and should we buy a stock at given price or not become a classification problem of KNN. Prediction of the stock market. A clear insight recent stock market trend, a structured and detailed study of stock market. It can be done by apply the analogy for timeseries and should we buy a stock at given price or not become a classification

problem of KNN. The combination study of both could help us understand past trend and we can forecast will prove to be beneficial for a lay-man.

Reference:

1] https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/
2] https://www.rdocumentation.org/packages/caret/versions/6.0-86/topics/confusionMatrix
3] https://rpubs.com/ryankelly/tsa6
4] https://www.rdocumentation.org/packages/gmodels/versions/2.18.1/topics/CrossTable
5] https://www.youtube.com/watch?v=uW3PQmzvUcw
6] https://pkg.robjhyndman.com/forecast/reference/seasonplot.html