Essentials of Data Analytics - Jcomp

Chandan Kumar

01/03/2021

Topic: Rainfall Prediction and its effect on Agriculture

Reg. No: 18BCE1020

Name: Chandan Kumar

Problem Statement and Objectives

Problem Statement

Rainfall prediction is a challenging task because of the dynamic nature of climate phenomena and random fluctuations involved in the physical process. Such a prediction is particularly challenging in Australia where in a long-term analysis the rate of change in the frequency and intensity of rainfall extremes can often be greater than the rate of change for average rainfall.

A bad rainfall prediction can affect the agriculture mostly framers as their whole crop is depend on the rainfall and agriculture is always an important part of every economy. So, making an accurate prediction of the rainfall somewhat good. There are number of techniques are used of machine learning but accuracy is always a matter of concern in prediction made in rainfall. There are number of causes made by rainfall affecting the world ex. Drought, Flood and intense summer heat etc. And it will also affect water resources around the world.

What data have you chosen?(Chosen Dataset, Source of dataset, Description of dataset, basic commands to describe dataset)

The Original Source of the datasets are given below:

http://www.bom.gov.au/climate/dwo/ , http://www.bom.gov.au/climate/data.

The dataset contains has 145460 rows and 23 columns. It has dataset contains both categorical as well as numerical values. It also contains a time period of 10 years of records which can be useful for Time Series model.

Along with this, I have used data from http://www.data.guv.au for the agriculture based data

Objectives

1. The goal of weather prediction is to provide information whether it will rain or not to people and organizations and they can use this to reduce weather-related losses and enhance societal benefits, including protection of life and property, public health and safety, and support of economic prosperity and quality of life.

2. I am also going to visualize various attributes of the dataset and draw insights from it.

3. The other objective is, using time series models, forecast the amount of rainfall.

4. The major objective of this project is to relate rainfall with agriculture and forecast the production based on the meteorological attributes present in the data.

Scope of the project:

This work can be used for the following problems:

i)Getting high yield of crops

ii)To decide favourable day to conduct outdoor sports events.

iii)To decide the operability of flights.

Algorithms to be used

This project has two type of problems: i) time-series based and ii) classification based.So, for both of them we have to use different models and compare them to build the most accurate and suitable model. Therfore for the time series model I am using the fllowing models:

i) ARIMA

ii)VAR

iii)VARMA

iv)Holt-Winter's Seasonal Method

For the classification I am using the following models:

i)Logistic regressiom

ii)Decision Tree

iii)KNN algorithm

Setup

Libraries

```
suppressMessages(library(dplyr))
suppressMessages(library(lubridate))
suppressMessages(library(ggplot2))
```

```
## Warning: package 'ggplot2' was built under R version 4.0.3
```

```
suppressMessages(library(forecast))
```

```
## Warning: package 'forecast' was built under R version 4.0.3
```

```
suppressMessages(library(tseries))
```

```
## Warning: package 'tseries' was built under R version 4.0.3
```

```
suppressMessages(library(vars)) ## VAR
```

```
## Warning: package 'vars' was built under R version 4.0.5

## Warning: package 'strucchange' was built under R version 4.0.4

## Warning: package 'sandwich' was built under R version 4.0.4

## Warning: package 'urca' was built under R version 4.0.3

## Warning: package 'lmtest' was built under R version 4.0.3
```

```
suppressMessages(library(MTS)) ## VARMA
```

```
## Warning: package 'MTS' was built under R version 4.0.5
```

```
suppressMessages(library(tidyverse))
suppressMessages(library(fpp2))
```

```
## Warning: package 'fpp2' was built under R version 4.0.5

## Warning: package 'fma' was built under R version 4.0.5

## Warning: package 'expsmooth' was built under R version 4.0.5
```

```
suppressMessages(library(astsa))
```

```
## Warning: package 'astsa' was built under R version 4.0.5
```

```
suppressMessages(library(GGally))
suppressMessages(library(caret))
suppressMessages(library(mice))
```

```
## Warning: package 'mice' was built under R version 4.0.5
```

```
suppressMessages(library(VIM))
```

```
## Warning: package 'VIM' was built under R version 4.0.5
```

```
suppressMessages(library(car))
suppressMessages(library(broom))
suppressMessages(library(rpart))
suppressMessages(library(caTools))
```

```
## Warning: package 'caTools' was built under R version 4.0.3
```

```
suppressMessages(library(rpart.plot))
```

```
## Warning: package 'rpart.plot' was built under R version 4.0.4
```

```
suppressMessages(library(e1071))
suppressMessages(library(randomForest))
```

```
## Warning: package 'randomForest' was built under R version 4.0.3
```

```
suppressMessages(library(party))
```

```
## Warning: package 'party' was built under R version 4.0.4

## Warning: package 'mvtnorm' was built under R version 4.0.3

## Warning: package 'modeltools' was built under R version 4.0.3
```

```
suppressMessages(library(class))
suppressMessages(library(gmodels))
```

```
## Warning: package 'gmodels' was built under R version 4.0.3
```

```
suppressMessages(library(grid))
suppressMessages(library(reshape2))

suppressMessages(library(tmap))
suppressMessages(library(sf))
suppressMessages(library(raster))
suppressMessages(library(spData))
```

```
## Warning: package 'spData' was built under R version 4.0.5
```

```
suppressMessages(library(maps))
```

```
## Warning: package 'maps' was built under R version 4.0.5
```

```
suppressMessages(library(leaflet))
suppressMessages(library(ggpubr))
```

Dataset Exploration

```
data=read.csv("weatherAUS.csv")
```

Reading Dataset

```
str(data)
```

Structure of the data

```
## 'data.frame':    145460 obs. of  23 variables:
##  $ Date         : chr  "2008-12-01" "2008-12-02" "2008-12-03" "2008-12-04" ...
##  $ Location     : chr  "Albury" "Albury" "Albury" "Albury" ...
##  $ MinTemp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
##  $ MaxTemp      : num  22.9 25.1 25.7 28 32.3 29.7 25 26.7 31.9 30.1 ...
##  $ Rainfall     : num  0.6 0 0 0 1 0.2 0 0 0 1.4 ...
##  $ Evaporation  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Sunshine     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ WindGustDir  : chr  "W" "WNW" "WSW" "NE" ...
##  $ WindGustSpeed: int  44 44 46 24 41 56 50 35 80 28 ...
##  $ WindDir9am   : chr  "W" "NNW" "W" "SE" ...
##  $ WindDir3pm   : chr  "WNW" "WSW" "WSW" "E" ...
##  $ WindSpeed9am : int  20 4 19 11 7 19 20 6 7 15 ...
##  $ WindSpeed3pm : int  24 22 26 9 20 24 24 17 28 11 ...
##  $ Humidity9am  : int  71 44 38 45 82 55 49 48 42 58 ...
##  $ Humidity3pm  : int  22 25 30 16 33 23 19 19 9 27 ...
##  $ Pressure9am  : num  1008 1011 1008 1018 1011 ...
##  $ Pressure3pm  : num  1007 1008 1009 1013 1006 ...
##  $ Cloud9am     : int  8 NA NA NA 7 NA 1 NA NA NA ...
##  $ Cloud3pm     : int  NA NA 2 NA 8 NA NA NA NA NA ...
##  $ Temp9am      : num  16.9 17.2 21 18.1 17.8 20.6 18.1 16.3 18.3 20.1 ...
##  $ Temp3pm      : num  21.8 24.3 23.2 26.5 29.7 28.9 24.6 25.5 30.2 28.2 ...
##  $ RainToday    : chr  "No" "No" "No" "No" ...
##  $ RainTomorrow : chr  "No" "No" "No" "No" ...
```

```
dim(data)
```

Rows and Columns in dataset

```
## [1] 145460     23
```

```
head(data)
```

First 6 elements of data

```
##         Date Location MinTemp MaxTemp Rainfall Evaporation Sunshine WindGustDir
## 1 2008-12-01   Albury    13.4    22.9      0.6          NA       NA           W
## 2 2008-12-02   Albury     7.4    25.1      0.0          NA       NA         WNW
## 3 2008-12-03   Albury    12.9    25.7      0.0          NA       NA         WSW
## 4 2008-12-04   Albury     9.2    28.0      0.0          NA       NA          NE
## 5 2008-12-05   Albury    17.5    32.3      1.0          NA       NA           W
## 6 2008-12-06   Albury    14.6    29.7      0.2          NA       NA         WNW
##   WindGustSpeed WindDir9am WindDir3pm WindSpeed9am WindSpeed3pm Humidity9am
## 1            44          W        WNW           20           24          71
## 2            44        NNW        WSW            4           22          44
## 3            46          W        WSW           19           26          38
## 4            24         SE          E           11            9          45
## 5            41        ENE         NW            7           20          82
## 6            56          W          W           19           24          55
##   Humidity3pm Pressure9am Pressure3pm Cloud9am Cloud3pm Temp9am Temp3pm
## 1          22      1007.7      1007.1        8       NA    16.9    21.8
## 2          25      1010.6      1007.8       NA       NA    17.2    24.3
## 3          30      1007.6      1008.7       NA        2    21.0    23.2
## 4          16      1017.6      1012.8       NA       NA    18.1    26.5
## 5          33      1010.8      1006.0        7        8    17.8    29.7
## 6          23      1009.2      1005.4       NA       NA    20.6    28.9
##   RainToday RainTomorrow
## 1        No           No
## 2        No           No
## 3        No           No
## 4        No           No
## 5        No           No
## 6        No           No
```

Data Cleaning

```
data$Date=as.POSIXct(data$Date, format="%Y-%m-%d")
str(data$Date)
```

Changing 'Date' from chr to Date type

```
##  POSIXct[1:145460], format: "2008-12-01" "2008-12-02" "2008-12-03" "2008-12-04" "2008-12-05" ...
```

```
Day=wday(data$Date, label=T)
data=data%>%
  mutate(Day)

head(data)
```

Adding day column in the dataset

```
##         Date Location MinTemp MaxTemp Rainfall Evaporation Sunshine WindGustDir
## 1 2008-12-01   Albury    13.4    22.9      0.6          NA       NA           W
## 2 2008-12-02   Albury     7.4    25.1      0.0          NA       NA         WNW
## 3 2008-12-03   Albury    12.9    25.7      0.0          NA       NA         WSW
## 4 2008-12-04   Albury     9.2    28.0      0.0          NA       NA          NE
## 5 2008-12-05   Albury    17.5    32.3      1.0          NA       NA           W
## 6 2008-12-06   Albury    14.6    29.7      0.2          NA       NA         WNW
##   WindGustSpeed WindDir9am WindDir3pm WindSpeed9am WindSpeed3pm Humidity9am
## 1            44          W        WNW           20           24          71
## 2            44        NNW        WSW            4           22          44
## 3            46          W        WSW           19           26          38
## 4            24         SE          E           11            9          45
## 5            41        ENE         NW            7           20          82
## 6            56          W          W           19           24          55
##   Humidity3pm Pressure9am Pressure3pm Cloud9am Cloud3pm Temp9am Temp3pm
## 1          22      1007.7      1007.1        8       NA    16.9    21.8
## 2          25      1010.6      1007.8       NA       NA    17.2    24.3
## 3          30      1007.6      1008.7       NA        2    21.0    23.2
## 4          16      1017.6      1012.8       NA       NA    18.1    26.5
## 5          33      1010.8      1006.0        7        8    17.8    29.7
## 6          23      1009.2      1005.4       NA       NA    20.6    28.9
##   RainToday RainTomorrow Day
## 1        No           No Mon
## 2        No           No Tue
## 3        No           No Wed
## 4        No           No Thu
## 5        No           No Fri
## 6        No           No Sat
```

```
sum(is.na(data$Date))
```

Checking for NA values in 'Date' Column

```
## [1] 0
```

```
unique(data$Location)
```

Checking for any missing value/ repetition of name etc in 'Location' column

```
##  [1] "Albury"           "BadgerysCreek"    "Cobar"            "CoffsHarbour"
##  [5] "Moree"            "Newcastle"        "NorahHead"        "NorfolkIsland"
##  [9] "Penrith"          "Richmond"         "Sydney"           "SydneyAirport"
## [13] "WaggaWagga"       "Williamtown"      "Wollongong"       "Canberra"
## [17] "Tuggeranong"      "MountGinini"      "Ballarat"         "Bendigo"
## [21] "Sale"             "MelbourneAirport" "Melbourne"        "Mildura"
## [25] "Nhil"             "Portland"         "Watsonia"         "Dartmoor"
## [29] "Brisbane"         "Cairns"           "GoldCoast"        "Townsville"
## [33] "Adelaide"         "MountGambier"     "Nuriootpa"        "Woomera"
## [37] "Albany"           "Witchcliffe"      "PearceRAAF"       "PerthAirport"
## [41] "Perth"            "SalmonGums"       "Walpole"          "Hobart"
## [45] "Launceston"       "AliceSprings"     "Darwin"           "Katherine"
## [49] "Uluru"
```

```
data%>%
  summarise(m1=sum(is.na(MinTemp)), m2=sum(is.na(MaxTemp)), m3=sum(is.na(Rainfall)), m4=sum(is.na(Evapo
```

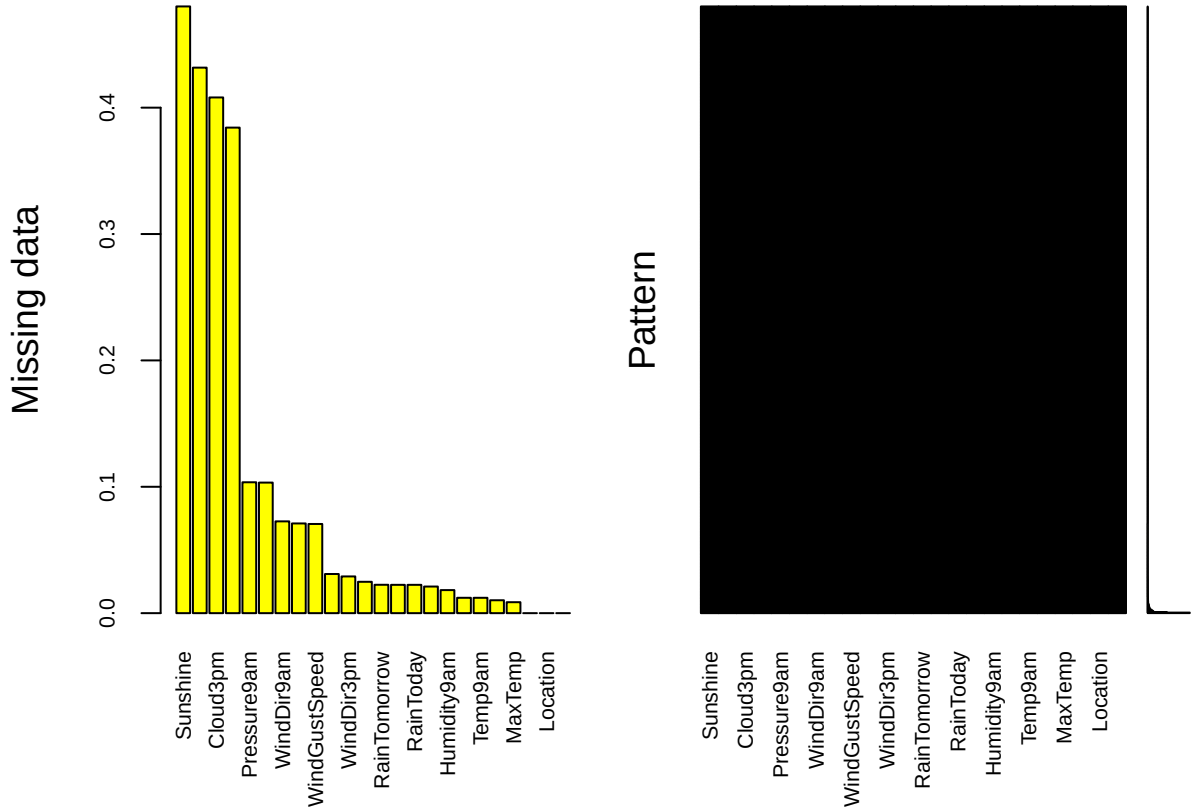
Checking NA values in all other numerical columns.

```
##     m1   m2   m3    m4    m5    m6   m7   m8   m9
## 1 1485 1261 3261 62790 69835 10566 4228 4507 4507
```

Visualising number of missing values

```
mice_plot <- aggr(data, col=c('navyblue','yellow'),
numbers=TRUE, sortVars=TRUE,
labels=names(data), cex.axis=.7,
gap=3, ylab=c("Missing data","Pattern"))
```

```
## Warning in plot.aggr(res, ...): not enough vertical space to display frequencies
## (too many combinations)
```




```
##
##  Variables sorted by number of missings:
##       Variable      Count
##       Sunshine 0.48009762
##    Evaporation 0.43166506
##       Cloud3pm 0.40807095
##       Cloud9am 0.38421559
##    Pressure9am 0.10356799
##    Pressure3pm 0.10331363
##     WindDir9am 0.07263853
##    WindGustDir 0.07098859
##  WindGustSpeed 0.07055548
##    Humidity3pm 0.03098446
##     WindDir3pm 0.02906641
##        Temp3pm 0.02481094
##   RainTomorrow 0.02245978
##       Rainfall 0.02241853
##      RainToday 0.02241853
##   WindSpeed3pm 0.02105046
##    Humidity9am 0.01824557
##   WindSpeed9am 0.01214767
##        Temp9am 0.01214767
##        MinTemp 0.01020899
##        MaxTemp 0.00866905
##           Date 0.00000000
##       Location 0.00000000
```

```
##            Day 0.00000000
```

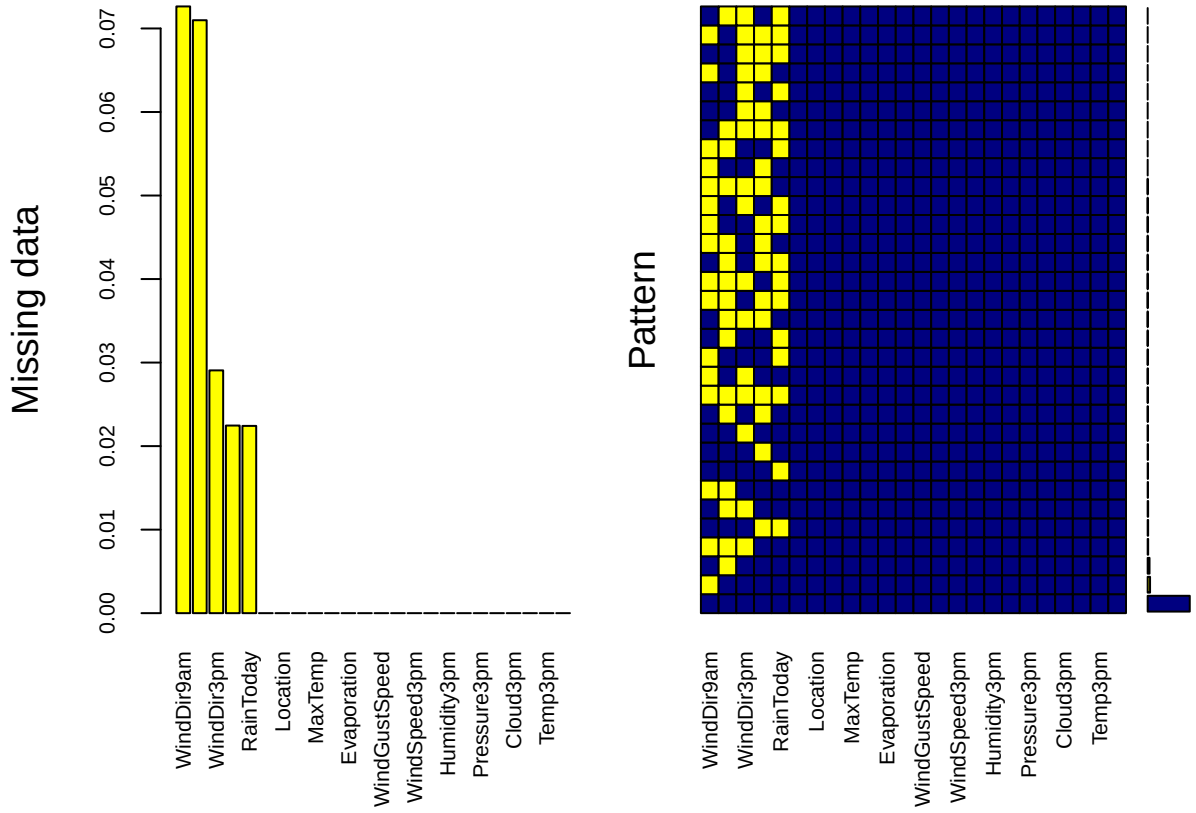
```
data=data%>%
  mutate(MinTemp=replace(MinTemp, is.na(MinTemp), mean(MinTemp, na.rm = T)),
         MaxTemp=replace(MaxTemp, is.na(MaxTemp), mean(MaxTemp, na.rm = T)),
         Rainfall=replace(Rainfall, is.na(Rainfall), mean(Rainfall, na.rm = T)),
         Humidity9am=replace(Humidity9am, is.na(Humidity9am), mean(Humidity9am, na.rm = T)),
         Humidity3pm=replace(Humidity3pm, is.na(Humidity3pm), mean(Humidity3pm, na.rm = T)),
         WindGustSpeed=replace(WindGustSpeed, is.na(WindGustSpeed), mean(WindGustSpeed, na.rm = T)),
         Sunshine=replace(Sunshine, is.na(Sunshine), mean(Sunshine, na.rm = T)),
         Cloud3pm=replace(Cloud3pm, is.na(Cloud3pm), mean(Cloud3pm, na.rm = T)),
         Evaporation=replace(Evaporation, is.na(Evaporation), mean(Evaporation, na.rm = T)),
         Cloud9am=replace(Cloud9am, is.na(Cloud9am), mean(Cloud9am, na.rm = T)),
         Pressure9am=replace(Pressure9am, is.na(Pressure9am), mean(Pressure9am, na.rm = T)),
         Pressure3pm=replace(Pressure3pm, is.na(Pressure3pm), mean(Pressure3pm, na.rm = T)),
         Temp3pm=replace(Temp3pm, is.na(Temp3pm), mean(Temp3pm, na.rm = T)),
         WindSpeed9am=replace(WindSpeed9am, is.na(WindSpeed9am), mean(WindSpeed9am, na.rm = T)),
         WindSpeed3pm=replace(WindSpeed3pm, is.na(WindSpeed3pm), mean(WindSpeed3pm, na.rm = T)),
         Temp9am=replace(Temp9am, is.na(Temp9am), mean(Temp9am, na.rm = T))
         )
```

Removing NA values with appriopriate value.

Visualising after cleaning

```
mice_plot <- aggr(data, col=c('navyblue','yellow'),
numbers=TRUE, sortVars=TRUE,
labels=names(data), cex.axis=.7,
gap=3, ylab=c("Missing data","Pattern"))
```

```
## Warning in plot.aggr(res, ...): not enough vertical space to display frequencies
## (too many combinations)
```




```
##
##  Variables sorted by number of missings:
##       Variable      Count
##     WindDir9am 0.07263853
##    WindGustDir 0.07098859
##     WindDir3pm 0.02906641
##   RainTomorrow 0.02245978
##      RainToday 0.02241853
##           Date 0.00000000
##       Location 0.00000000
##        MinTemp 0.00000000
##        MaxTemp 0.00000000
##       Rainfall 0.00000000
##    Evaporation 0.00000000
##       Sunshine 0.00000000
##  WindGustSpeed 0.00000000
##   WindSpeed9am 0.00000000
##   WindSpeed3pm 0.00000000
##    Humidity9am 0.00000000
##    Humidity3pm 0.00000000
##    Pressure9am 0.00000000
##    Pressure3pm 0.00000000
##       Cloud9am 0.00000000
##       Cloud3pm 0.00000000
##        Temp9am 0.00000000
##        Temp3pm 0.00000000
```

```
##            Day 0.00000000
```

```
data$RainToday=factor(data$RainToday, labels = c(0,1))
str(data$RainToday)
```

Making Rain or Not Rained as factor from chr

```
##  Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
```

```
data$RainTomorrow=factor(data$RainTomorrow, labels = c(0,1))
str(data$RainTomorrow)
```

```
##  Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
```

EDA

Statistical Analysis

```
mean(data$MinTemp)
```

Min Temprature

```
## [1] 12.19403
```

```
range(data$MinTemp)
```

The average minimum temprature of dataset is 12.19

```
## [1] -8.5 33.9
```

```
quantile(data$MinTemp)
```

The range of MinTemp is from -8.5 to 33.9

```
##   0%  25%  50%  75% 100%
## -8.5  7.7 12.1 16.8 33.9
```

```
sd(data$MinTemp)
```

From here we can observe that the median of the minTemp is 12.1

```
## [1] 6.36575
```

The standard deviation of minTemp is 6.36

```
mean(data$MaxTemp)
```

MaxTemp

```
## [1] 23.22135
```

```
quantile(data$MaxTemp)
```

The mean of MaxTemp is 23.22

```
##   0%  25%  50%  75% 100%
## -4.8 18.0 22.7 28.2 48.1
```

```
sd(data$MaxTemp)
```

Here we can see that MaxTemp ranges between -4.8 to 18.0 and the median is 22.7

```
## [1] 7.088124
```

The standard deviation of MaxTemp is 7.088

```
mean(data$Rainfall)
```

RainFall

```
## [1] 2.360918
```

```
quantile(data$Rainfall)
```

The average rainfall is 2.36 mm

```
##   0%  25%  50%  75% 100%
##    0    0    0    1  371
```

```
sd(data$Rainfall)
```

Here we can see that Rainfall ranges between 0 to 371 mm

```
## [1] 8.382488
```

The standard deviation of Rainfall is 8.38

```
table(data$WindGustDir)
```

WindGust Dir

```
##
##    E  ENE  ESE    N   NE  NNE  NNW   NW    S   SE  SSE  SSW   SW    W  WNW  WSW
## 9181 8104 7372 9313 7133 6548 6620 8122 9168 9418 9216 8736 8967 9915 8252 9069
```

Here we can observe that the maximum number of wind directon is towards West.

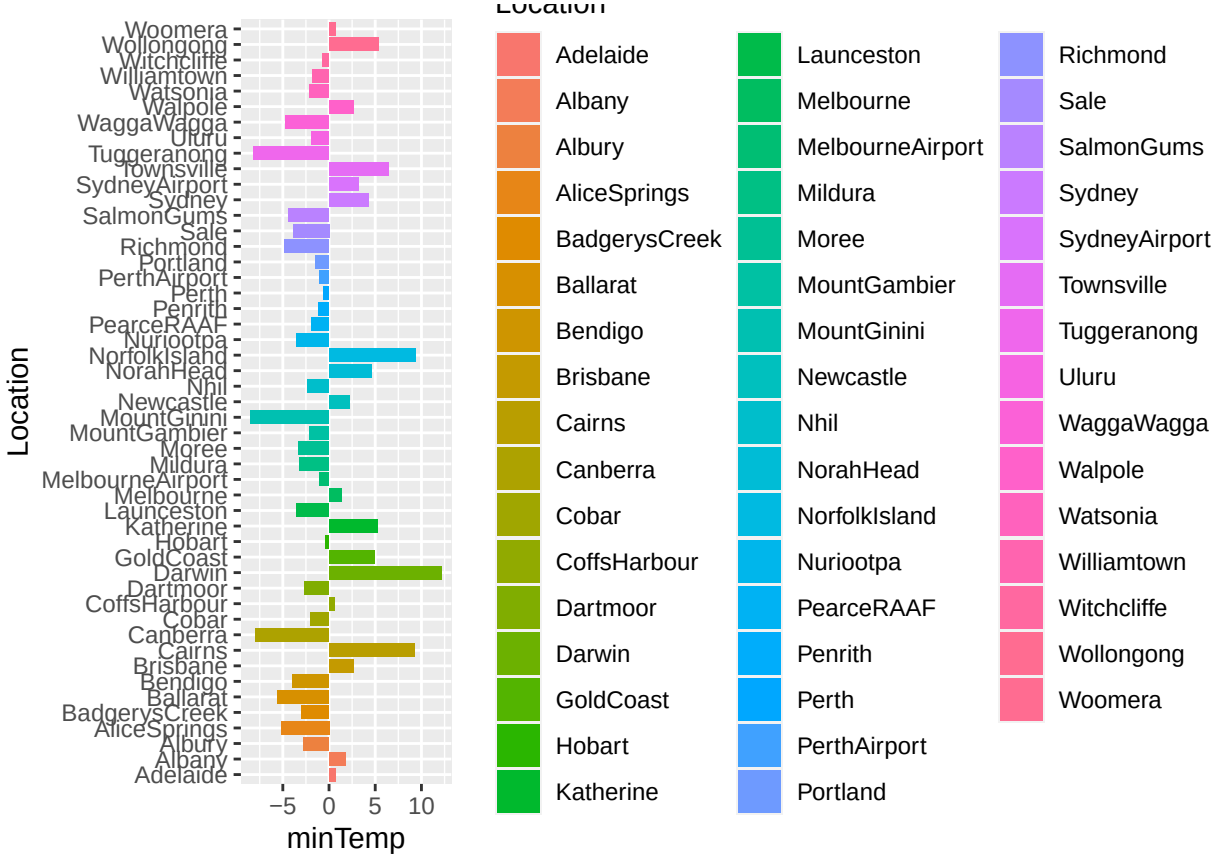
Visualisation

```
TempbyLoc=data%>%
  group_by(Location)%>%
  summarise(minTemp=min(MinTemp), maxTemp=max(MaxTemp), avgRain=mean(Rainfall))
```

Maximum and Minimum Temp Comaprison of different Location

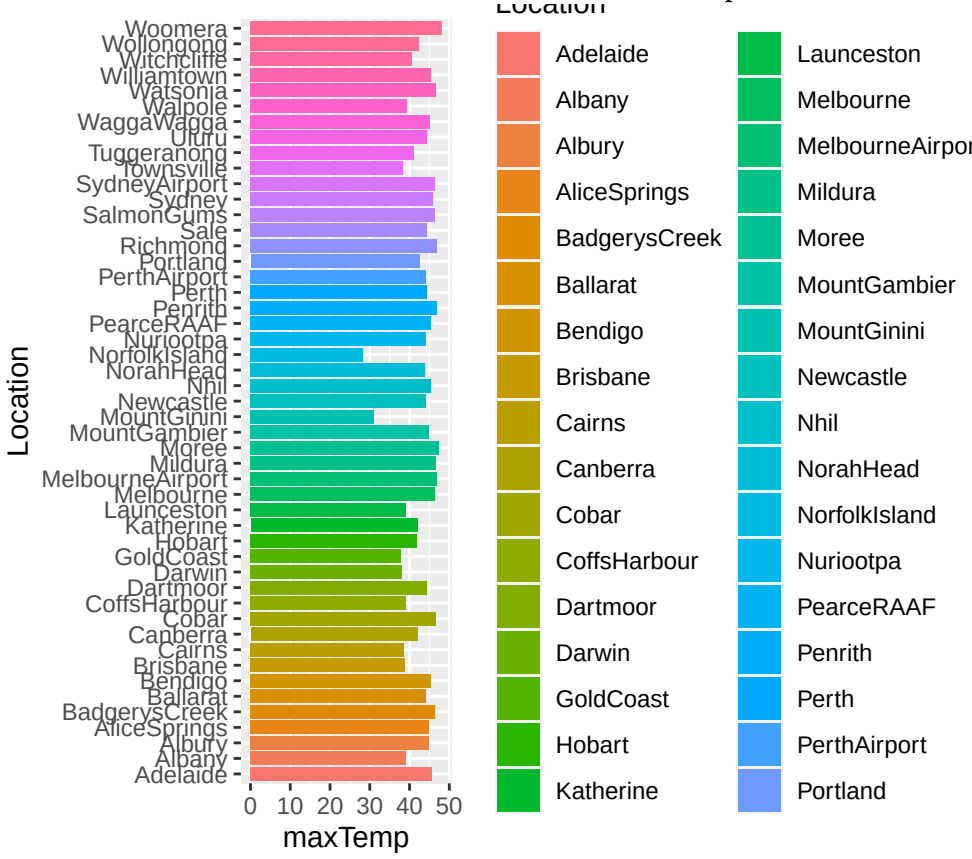
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
ggplot(data=TempbyLoc, aes(x=Location, y=minTemp, fill=Location)) + geom_bar(stat = 'identity') + coord_
```



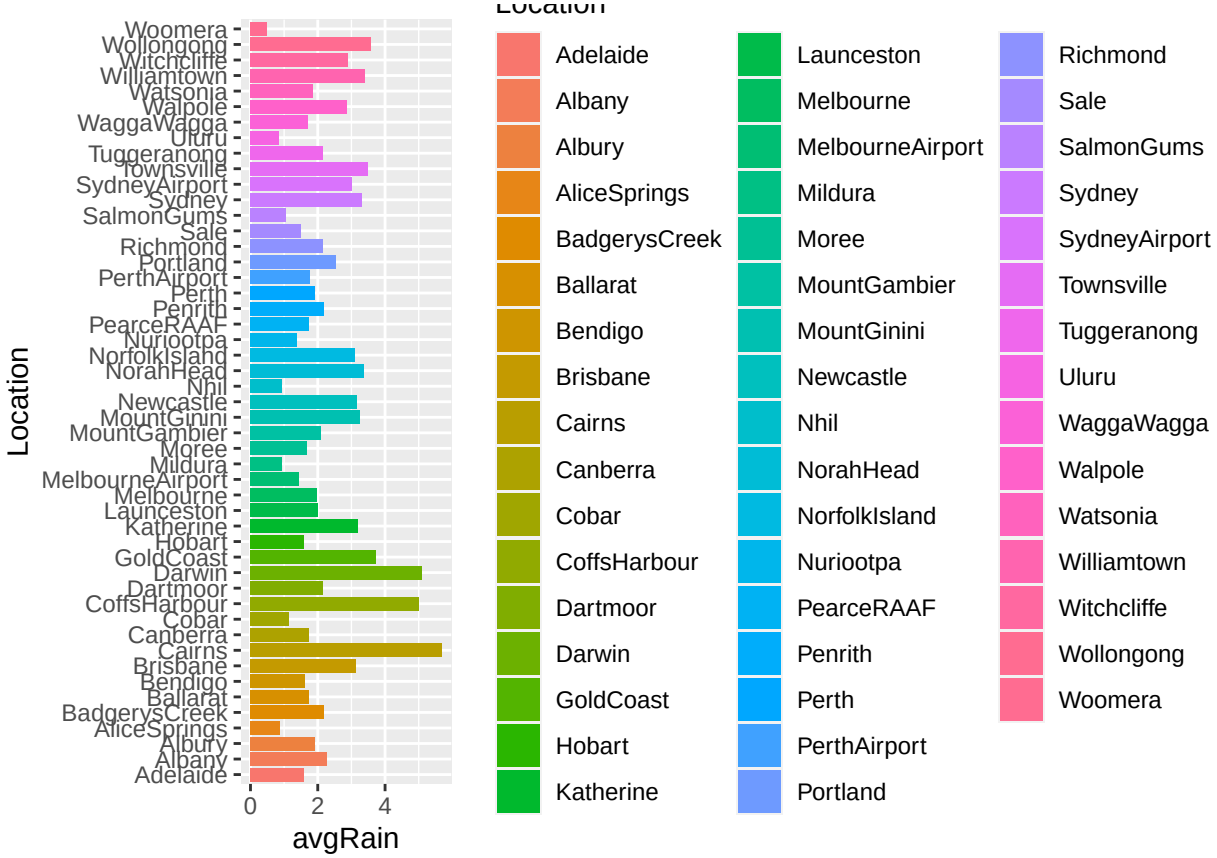

```
ggplot(data=TempbyLoc, aes(x=Location, y=maxTemp, fill=Location)) + geom_bar(stat = 'identity') + coord_
```

From the above bargraph we can observe that MountGinini has recorded the minimum tempra-



ture in the span of these 10 years.

```
ggplot(data=TempbyLoc, aes(x=Location, y=avgRain, fill=Location)) + geom_bar(stat = 'identity') + coord_
```



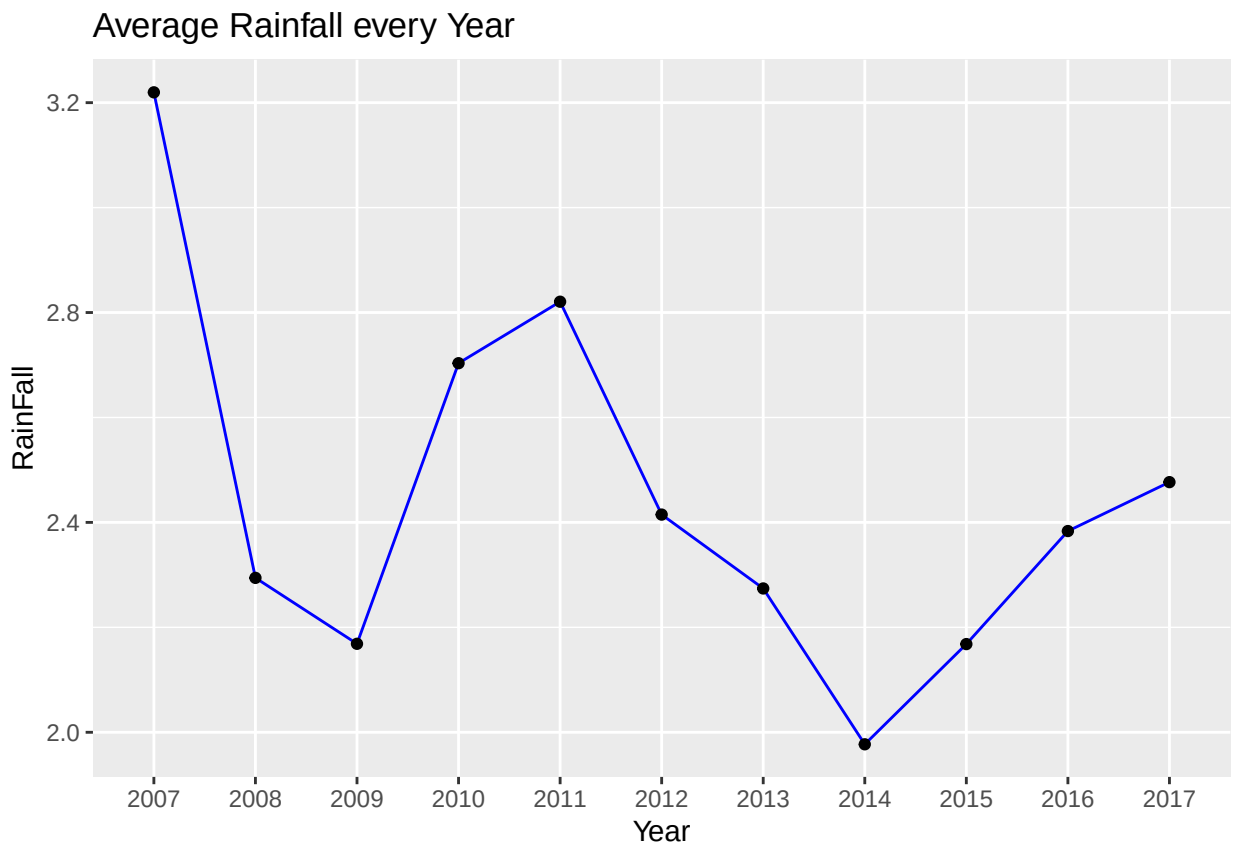
In the above we can observe that Cairns has recorded the maximum average rainfall in the span of these 10 years.

Average Rain Every Year / Month

```
data%>%
  mutate(year=format(Date,"%Y"))%>%
  group_by(year)%>%
  summarise(rain=mean(Rainfall))%>%
  ggplot(aes(x=year, y=rain, group=1))+geom_line(color="blue")+geom_point()+labs(title="Average Rainfal
```

Year-wise

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

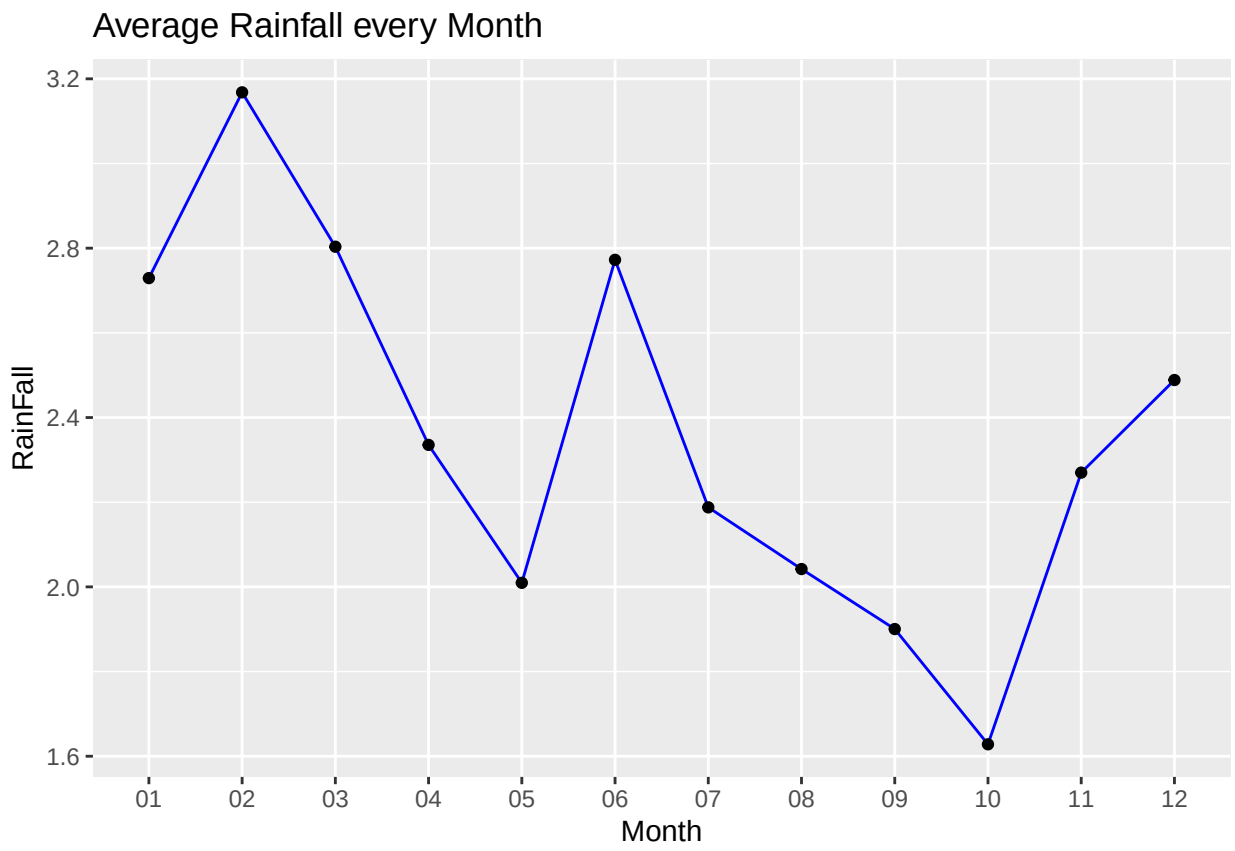


In the above line graph we can observe that Maximum avg rainfall is observed in 2007 and least in 2014.

```
data%>%
  mutate(month=format(Date,"%m"))%>%
  group_by(month)%>%
  summarise(rain=mean(Rainfall))%>%
  ggplot(aes(x=month, y=rain, group=1))+geom_line(color="blue")+geom_point()+labs(title="Average Rainfal
```

Month Wise

```
## `summarise()` ungrouping output (override with `.groups` argument)
```



In the above line graph we can observe that maximum Rainfall has occurred in the month of February and the least is in October.

Time Series Analysis

Model used:

1. ARIMA(Auto Regression Integrated Moving Average)

2. VAR(Vector Auto Regression)

3. VARMA(Vector Autoregression Moving Average)

4. Holt-Winter's Seasonal Method

Splitting Date into year, month and date

```
year <- as.numeric(format(data$Date,"%Y"))
mon <- as.numeric(format(data$Date,"%m"))
day <- as.numeric(format(data$Date,"%d"))
```

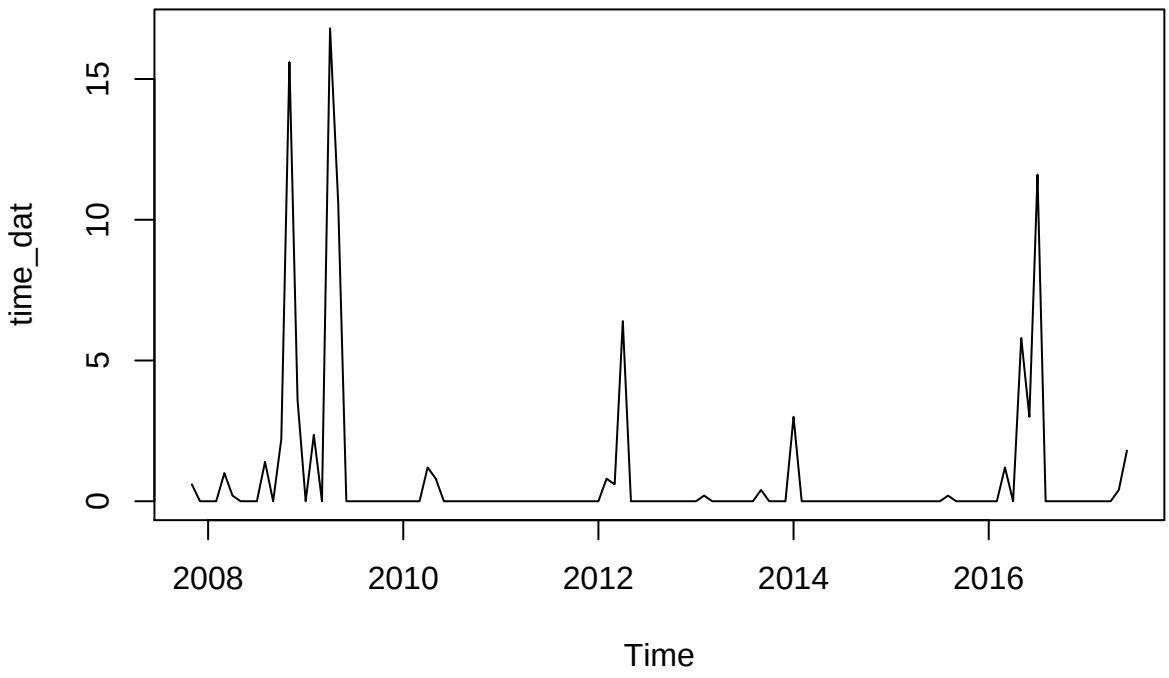
Combining old data with new formed columns

```
split_data=cbind(data,year,mon,day)
str(split_data)
```

```
## 'data.frame':    145460 obs. of  27 variables:
##  $ Date         : POSIXct, format: "2008-12-01" "2008-12-02" ...
##  $ Location     : chr  "Albury" "Albury" "Albury" "Albury" ...
##  $ MinTemp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
##  $ MaxTemp      : num  22.9 25.1 25.7 28 32.3 29.7 25 26.7 31.9 30.1 ...
##  $ Rainfall     : num  0.6 0 0 0 1 0.2 0 0 0 1.4 ...
##  $ Evaporation  : num  5.47 5.47 5.47 5.47 5.47 ...
##  $ Sunshine     : num  7.61 7.61 7.61 7.61 7.61 ...
##  $ WindGustDir  : chr  "W" "WNW" "WSW" "NE" ...
##  $ WindGustSpeed: num  44 44 46 24 41 56 50 35 80 28 ...
##  $ WindDir9am   : chr  "W" "NNW" "W" "SE" ...
##  $ WindDir3pm   : chr  "WNW" "WSW" "WSW" "E" ...
##  $ WindSpeed9am : num  20 4 19 11 7 19 20 6 7 15 ...
##  $ WindSpeed3pm : num  24 22 26 9 20 24 24 17 28 11 ...
##  $ Humidity9am  : num  71 44 38 45 82 55 49 48 42 58 ...
##  $ Humidity3pm  : num  22 25 30 16 33 23 19 19 9 27 ...
##  $ Pressure9am  : num  1008 1011 1008 1018 1011 ...
##  $ Pressure3pm  : num  1007 1008 1009 1013 1006 ...
##  $ Cloud9am     : num  8 4.45 4.45 4.45 7 ...
##  $ Cloud3pm     : num  4.51 4.51 2 4.51 8 ...
##  $ Temp9am      : num  16.9 17.2 21 18.1 17.8 20.6 18.1 16.3 18.3 20.1 ...
##  $ Temp3pm      : num  21.8 24.3 23.2 26.5 29.7 28.9 24.6 25.5 30.2 28.2 ...
##  $ RainToday    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ RainTomorrow : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ Day          : Ord.factor w/ 7 levels "Sun"<"Mon"<"Tue"<..: 2 3 4 5 6 7 1 2 3 4 ...
##  $ year         : num  2008 2008 2008 2008 2008 ...
##  $ mon          : num  12 12 12 12 12 12 12 12 12 12 ...
##  $ day          : num  1 2 3 4 5 6 7 8 9 10 ...
```

Assigning the start and end as well as the cycle of time series data

```
time_dat <- ts(data$Rainfall,start=c(2007,11,01),end=c(2017,06,25),frequency = 12)
plot(time_dat)
```



In the above plot we can observe that data does not have any trend, i.e the data is stationary.Therefore to use the data for time series model, we don't need to differentiate(d=0).

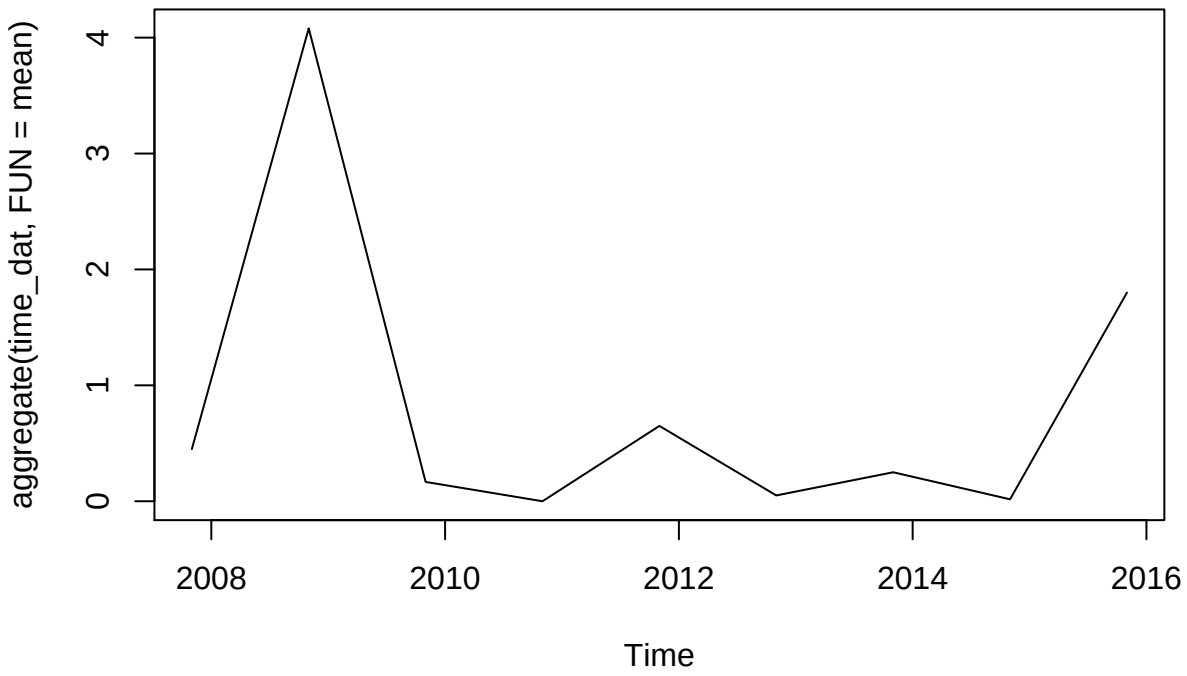
Data in each cycle

```
cycle(time_dat)
```

```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 2007                                             11  12
## 2008   1   2   3   4   5   6   7   8   9  10  11  12
## 2009   1   2   3   4   5   6   7   8   9  10  11  12
## 2010   1   2   3   4   5   6   7   8   9  10  11  12
## 2011   1   2   3   4   5   6   7   8   9  10  11  12
## 2012   1   2   3   4   5   6   7   8   9  10  11  12
## 2013   1   2   3   4   5   6   7   8   9  10  11  12
## 2014   1   2   3   4   5   6   7   8   9  10  11  12
## 2015   1   2   3   4   5   6   7   8   9  10  11  12
## 2016   1   2   3   4   5   6   7   8   9  10  11  12
## 2017   1   2   3   4   5   6
```

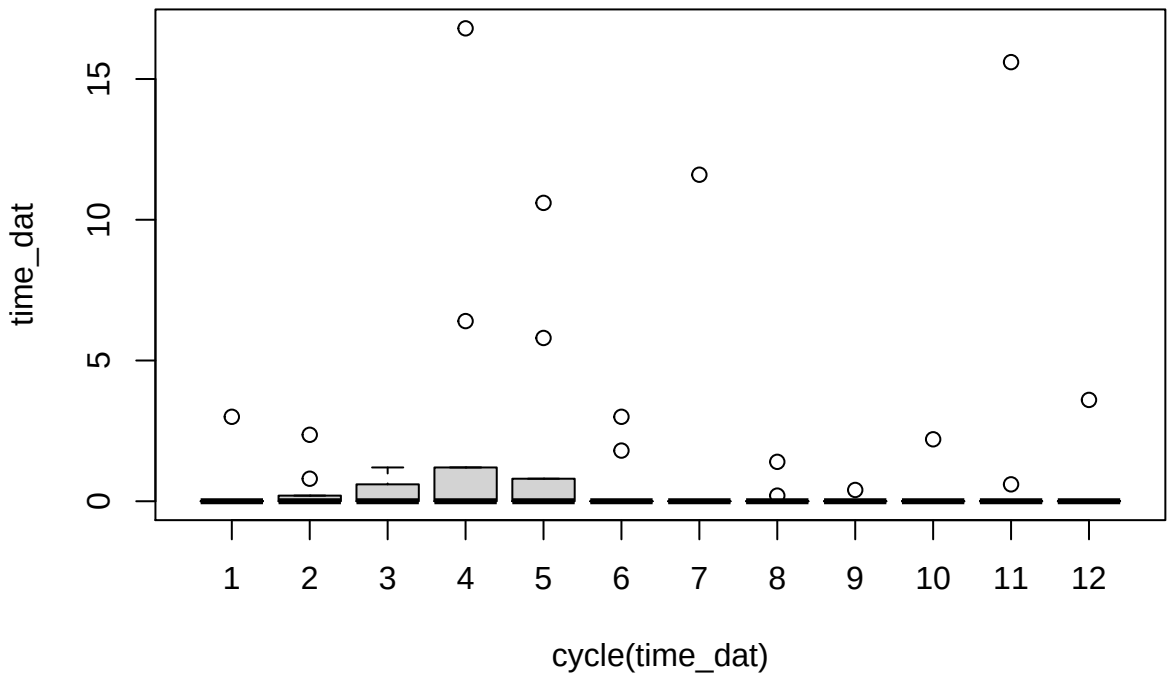
Plot for each cycle

```
plot(aggregate(time_dat,FUN=mean))
```



Box Plot of each cycle

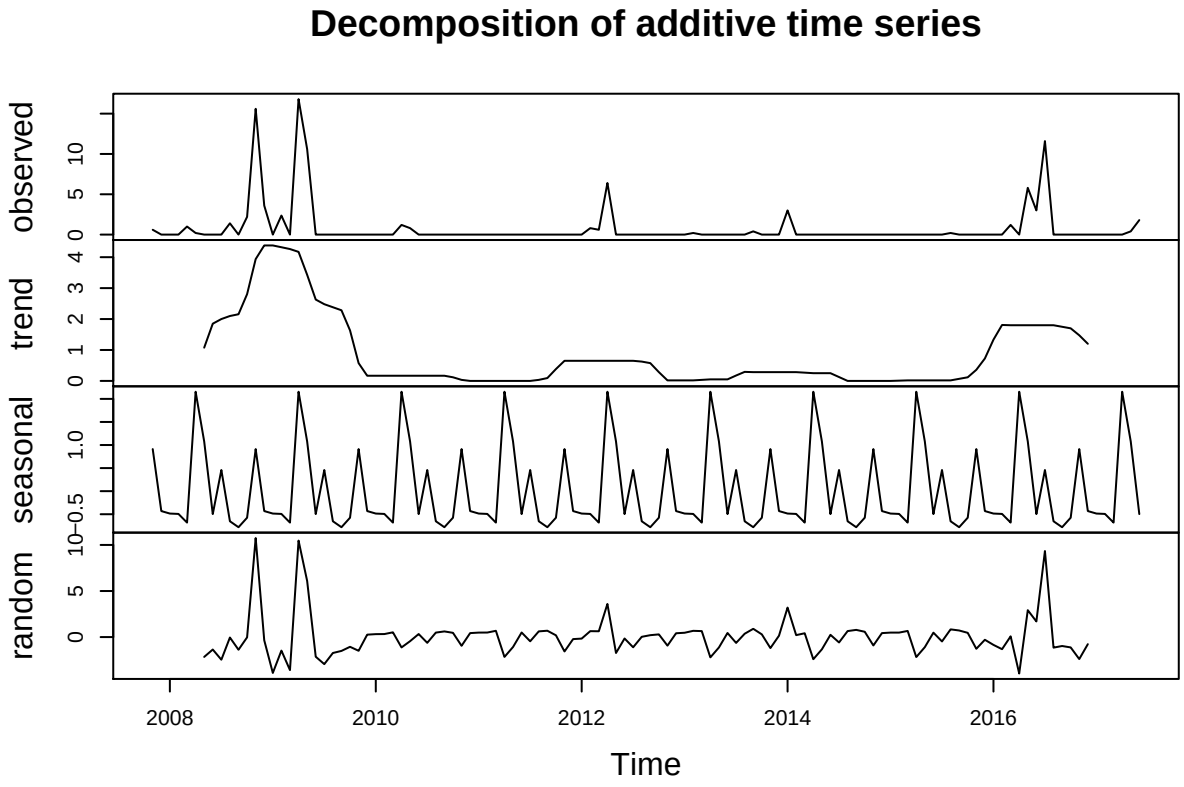
```
boxplot(time_dat~cycle(time_dat))
```



In the above boxplot we can observe 4th month has the maximum rainfall i.e in the month of April Australia recieves the maximum rainfall.

Visualising Decomposition

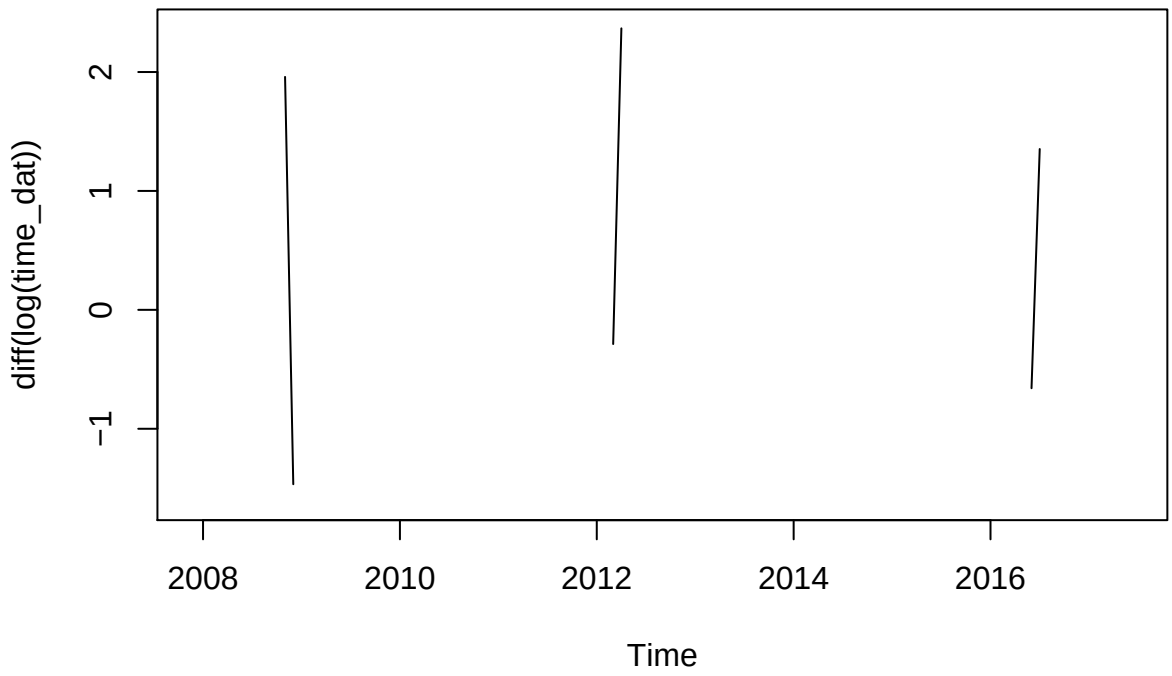
```
plot(decompose(time_dat))
```



The above plot shows that the data has only seasonal trend.

Making the data stationary

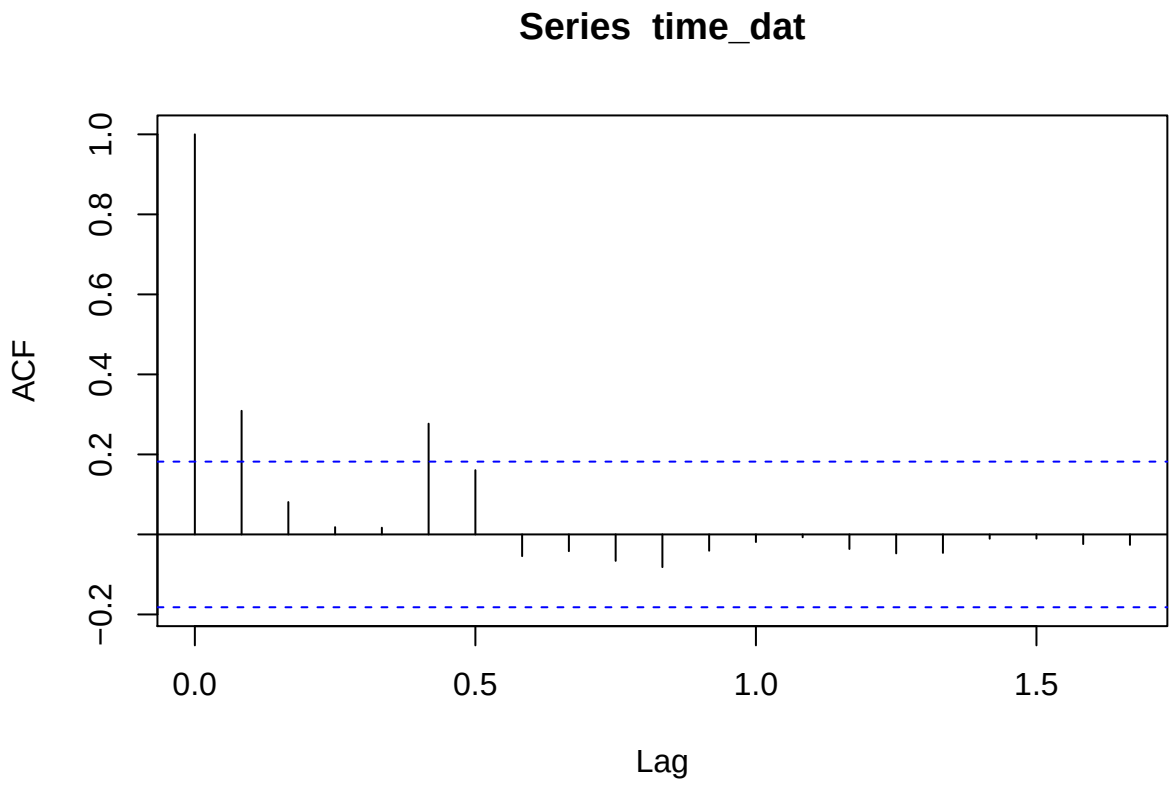
```
plot(diff(log(time_dat)))
```



Although, the data is stationary, still if we still try to differentiate it, the data won't be able make any sense, making it un-useful.

Auto correlation of data when it was not made stationary.

```
acf(time_dat)
```



1.ARIMA

Model Building using ARIMA

```
model1 <-auto.arima(time_dat)
model1
```

```
## Series: time_dat
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##          ar1    mean
##       0.3067  0.7941
## s.e.  0.0879  0.3389
##
## sigma^2 estimated as 6.564:  log likelihood=-272.77
## AIC=551.55   AICc=551.76   BIC=559.81
```

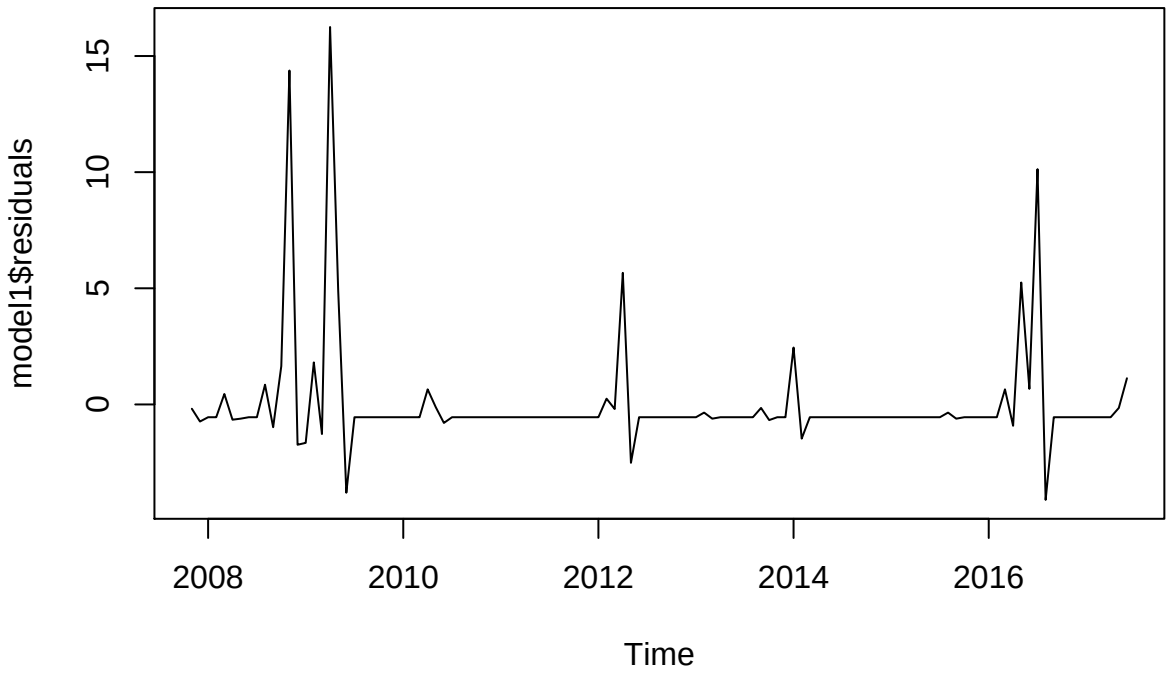
In the above result, we can observe that ARIMA model gives optimum result for (p,q,d)=(1,0,0). The AIC abd BIC values are 551.55 and 559.81 respectively.

AIC and BIC are both penalized-likelihood criteria. AIC is an estimate of a constant plus the relative distance between the unknown true likelihood function of the data and the fitted

likelihood function of the model, so that a lower AIC means a model is considered to be closer to the truth. BIC is an estimate of a function of the posterior probability of a model being true, under a certain Bayesian setup, so that a lower BIC means that a model is considered to be more likely to be the true model.

Residuls Plot

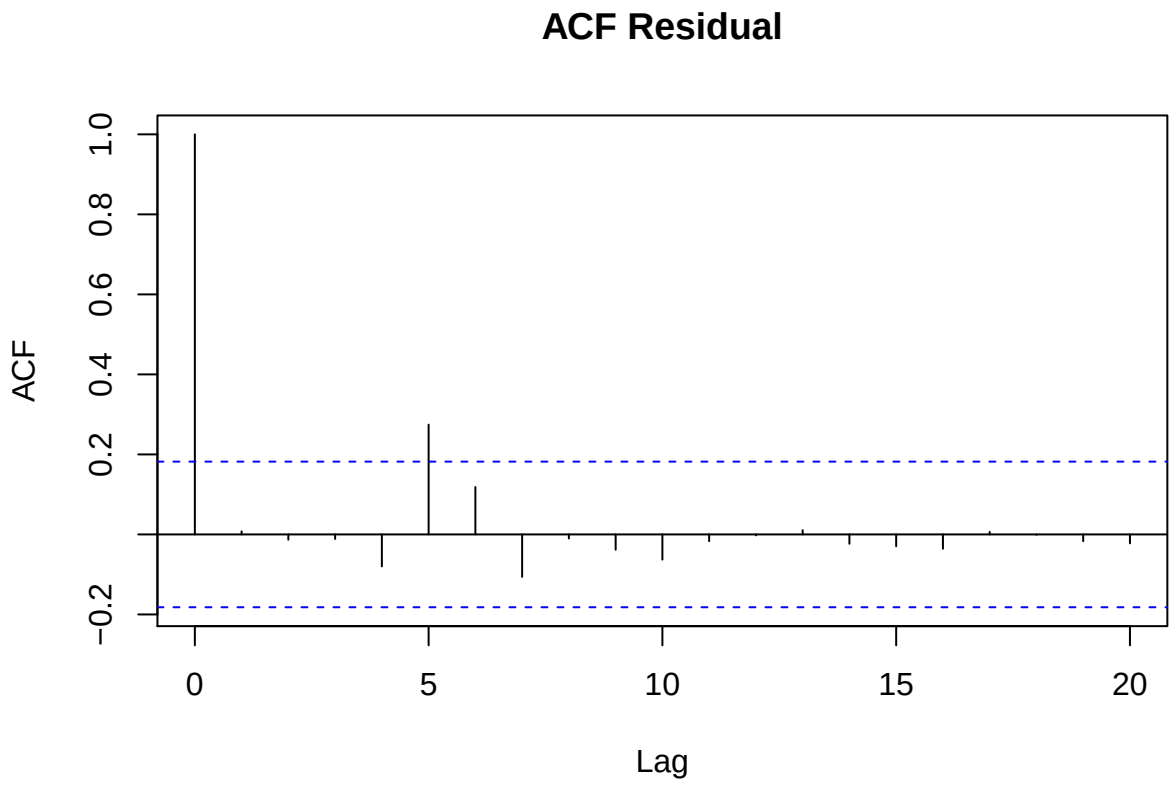
```
plot.ts(model1$residuals)
```



The "residuals" in a time series model are what is left over after fitting a model. For many (but not all) time series models, the residuals are equal to the difference between the observations and the corresponding fitted values.Residuals are useful in checking whether a model has adequately captured the information in the data.

q-value

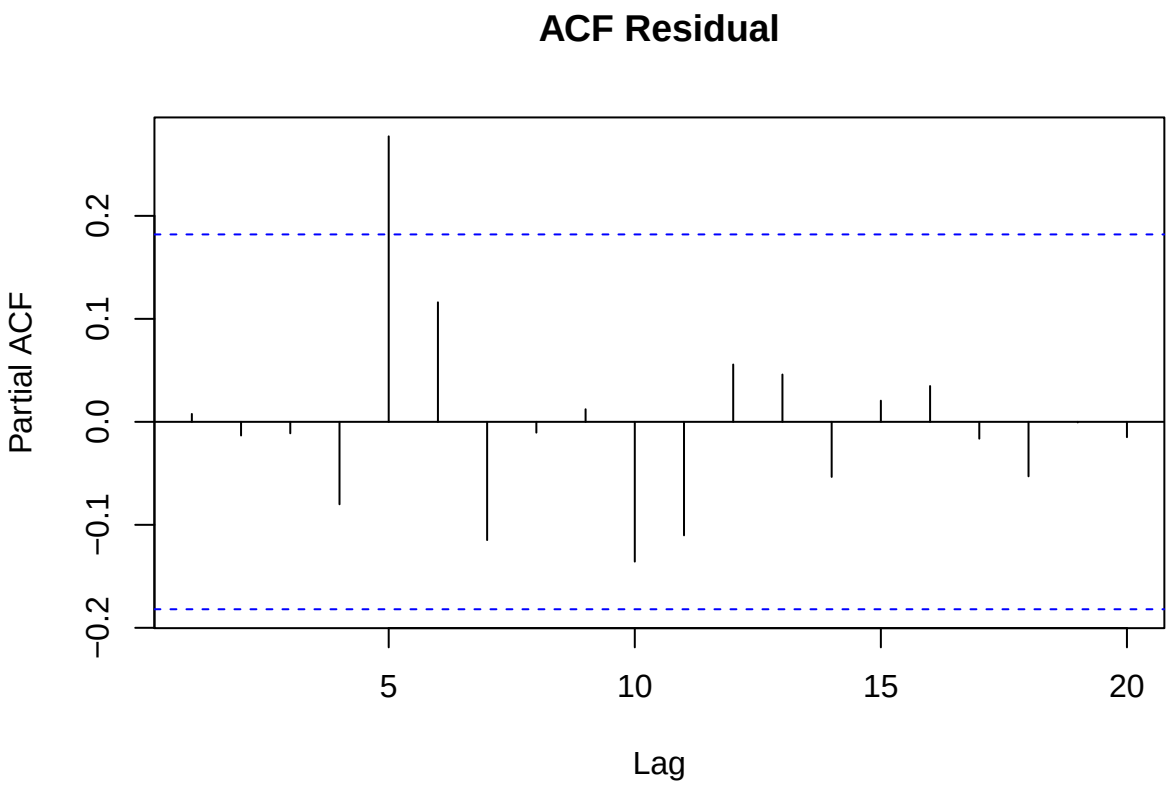
```
acf(ts(model1$residuals),main="ACF Residual")
```



Starting from first line (count it as 0), the line preceding the first inverted line is the value needed. Here, we can see that q=1.

p-val

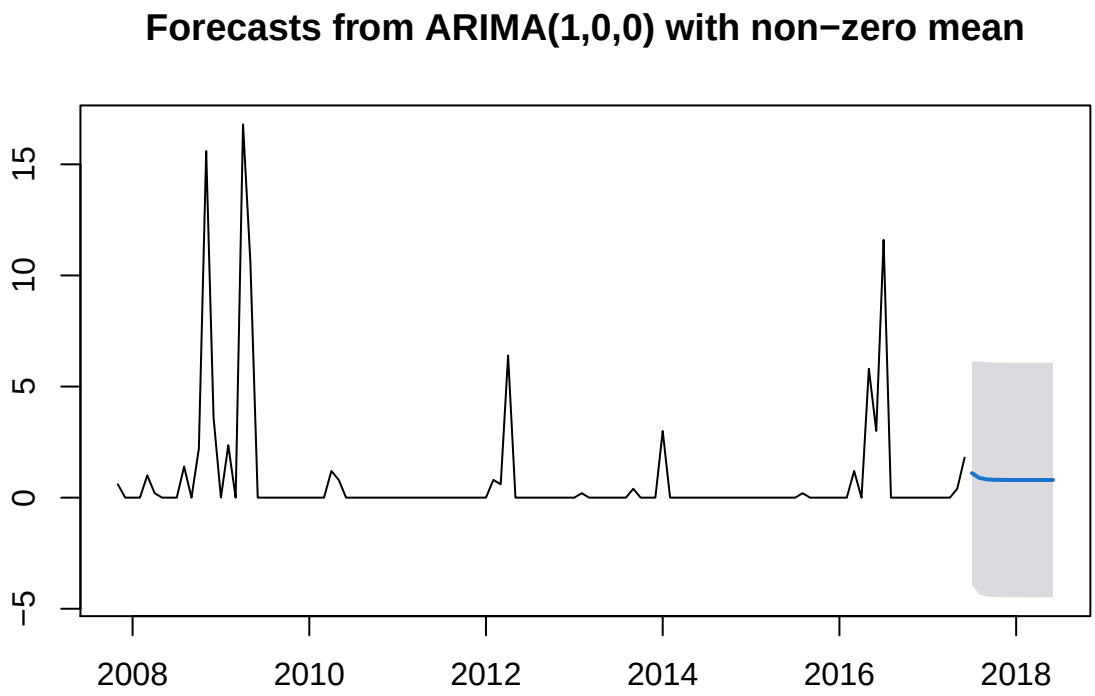
```
pacf(ts(model1$residuals),main="ACF Residual")
```



Similar to acf, here p=0.

Forecasting values

```
forect <- forecast(model1,level=c(95),h=1*12)
plot(forect)
```



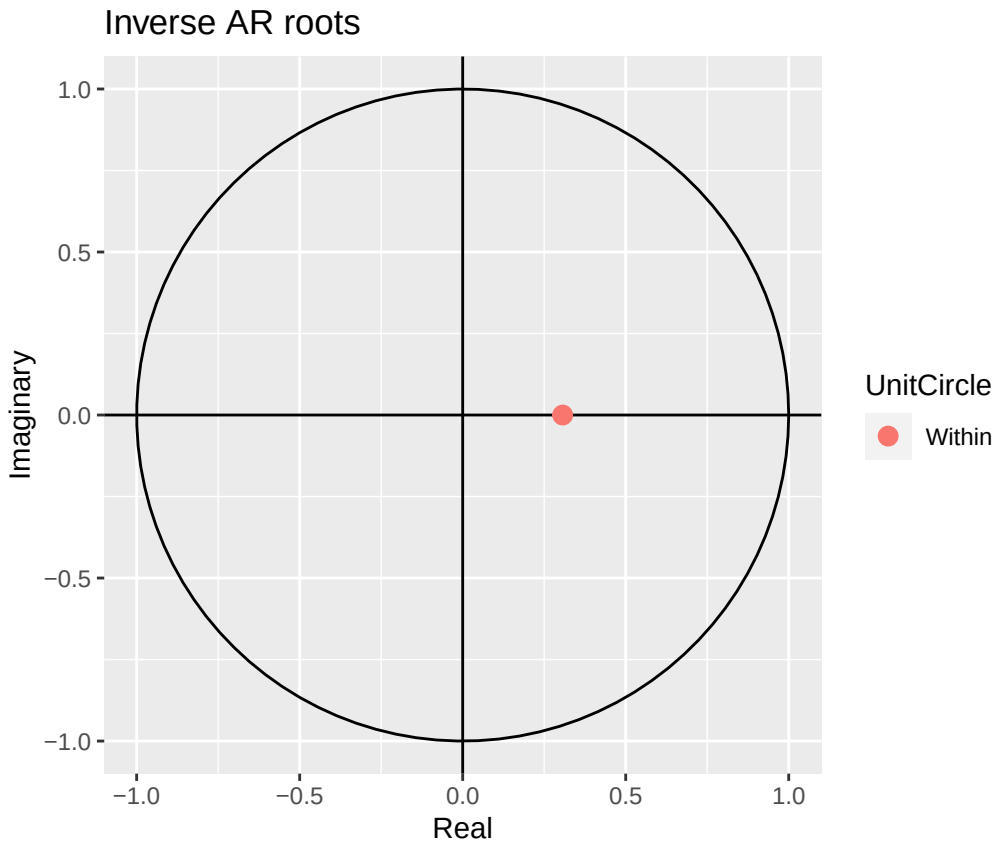
Summary of ARIMA model

```
summary(model1)
```

```
## Series: time_dat
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##          ar1    mean
##       0.3067  0.7941
## s.e.  0.0879  0.3389
##
## sigma^2 estimated as 6.564:  log likelihood=-272.77
## AIC=551.55   AICc=551.76   BIC=559.81
##
## Training set error measures:
##                       ME     RMSE      MAE  MPE MAPE      MASE       ACF1
## Training set 0.0006047933 2.539933 1.156613 -Inf  Inf 0.7814859 0.00766371
```

Plotting the characteristic roots

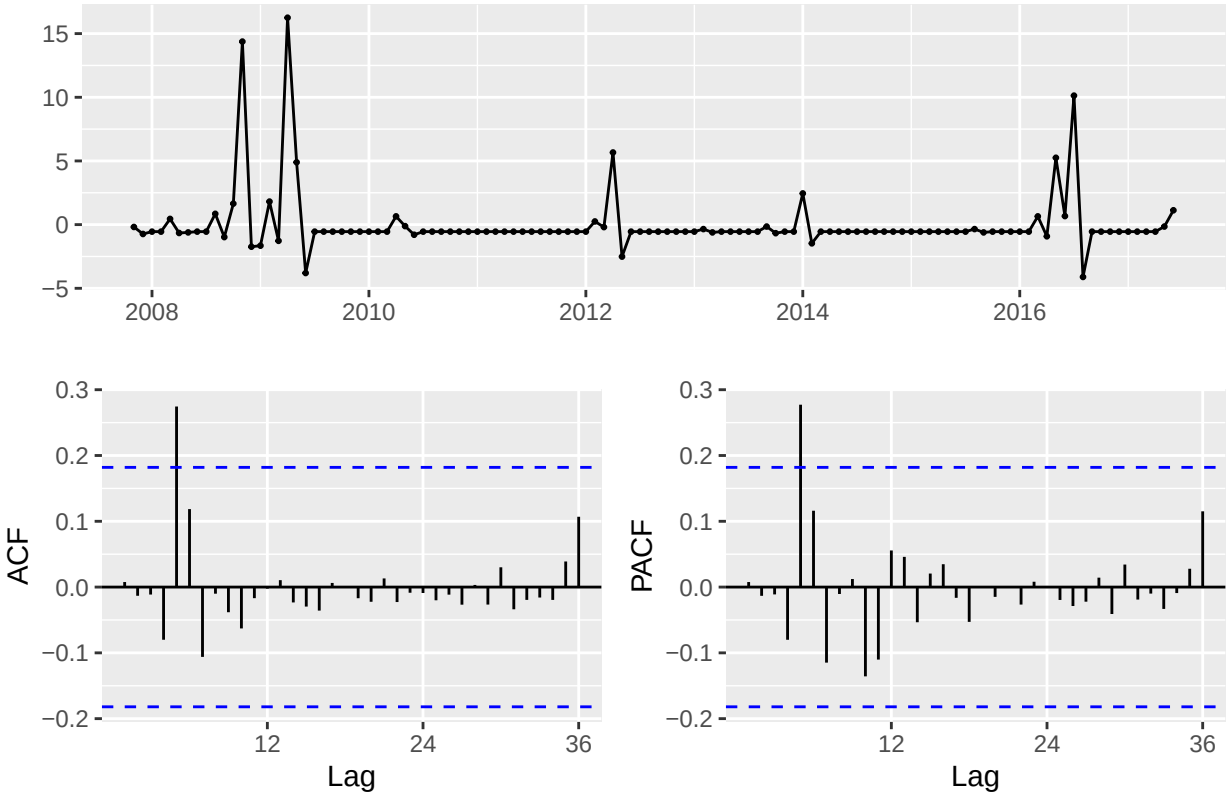
```
autoplot(model1)
```



The red dot in the plot correspond to the root of the polynomial of ARIMA model. It is inside the unit circle, as we would expect because R ensures the fitted model is both stationary and invertible. Any roots close to the unit circle may be numerically unstable, and the corresponding model will not be good for forecasting.

ACF, PACF plot

```
ggtsdisplay(model1$residuals)
```



2.Vector Autoregression Model(VAR)

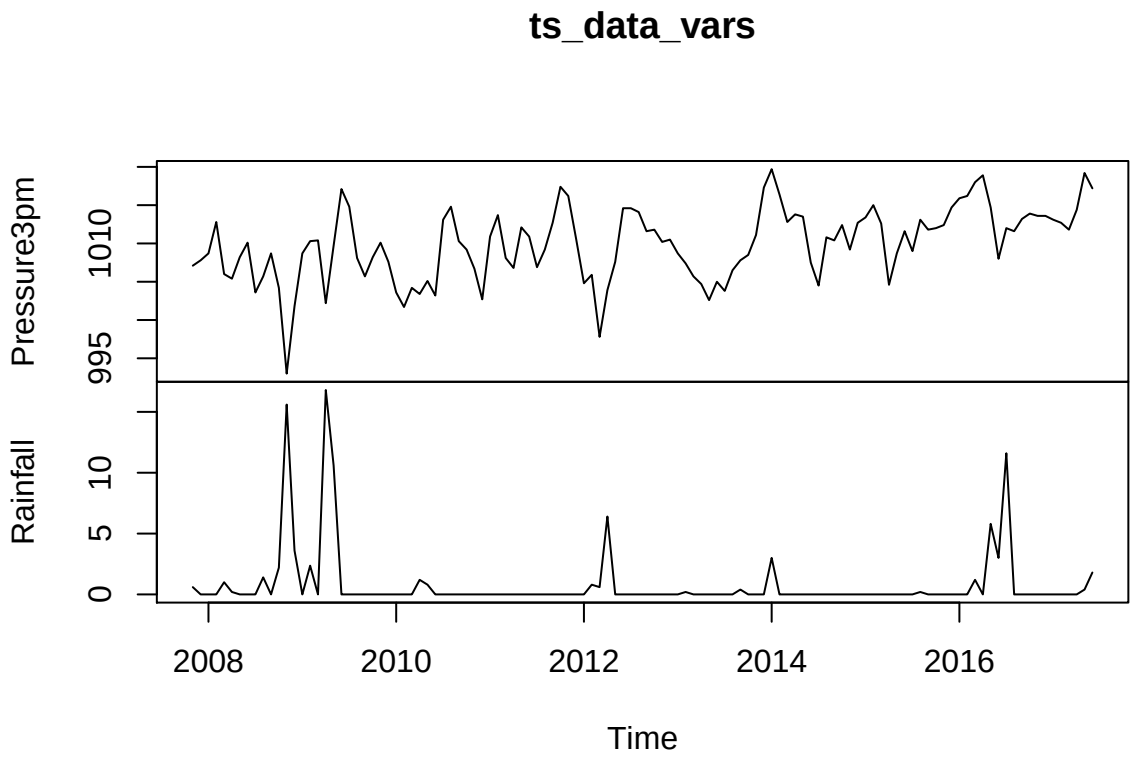
One limitation of the ARIMA model is that they impose a unidirectional relationship — the forecast variable is influenced by the predictor variables, but not vice versa. However, there are many cases where the reverse should also be allowed for — where all variables affect each other.

Selecting rows for forecasting.

```
temp=data[, c("Pressure3pm", "Rainfall")]
ts_data_vars=ts(temp, start=c(2007,11,01),end=c(2017,06,25),frequency = 12)
```

Plotting graph for Pressure and Rainfall

```
plot.ts(ts_data_vars)
```



Model building

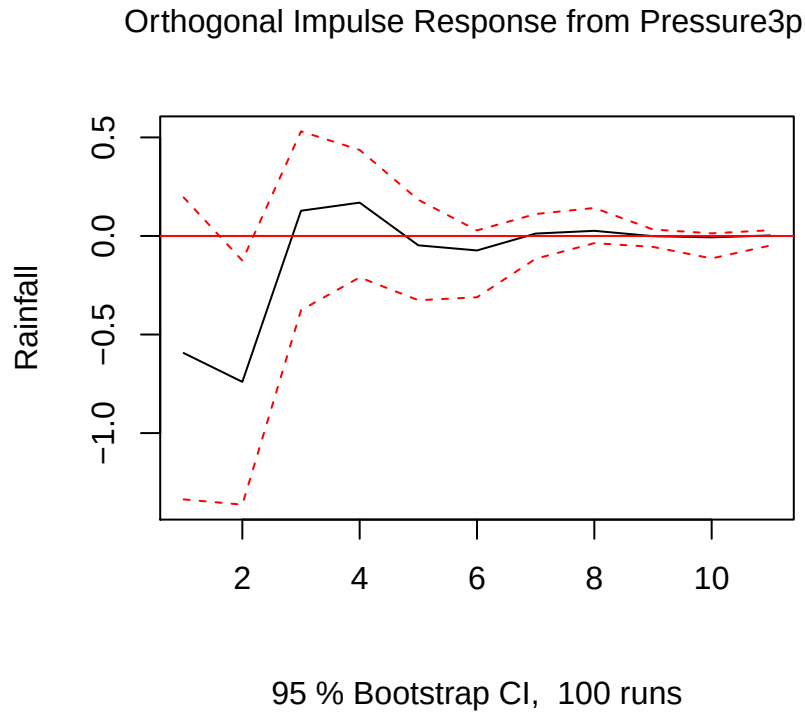
```
model2=vars::VAR(ts_data_vars, type = "none", lag.max = 5, ic = "AIC")
summary(model2)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: Pressure3pm, Rainfall
## Deterministic variables: none
## Sample size: 113
## Log Likelihood: -563.994
## Roots of the characteristic polynomial:
##     1 0.5886 0.5886 0.4332 0.3399 0.1201
## Call:
## vars::VAR(y = ts_data_vars, type = "none", lag.max = 5, ic = "AIC")
##
##
## Estimation results for equation Pressure3pm:
## ============================================
## Pressure3pm = Pressure3pm.l1 + Rainfall.l1 + Pressure3pm.l2 + Rainfall.l2 + Pressure3pm.l3 + Rainfal
##
##                Estimate Std. Error t value Pr(>|t|)
## Pressure3pm.l1  0.97181    0.09609  10.114  < 2e-16 ***
```

```
## Rainfall.l1     0.20782    0.14290   1.454  0.14881
## Pressure3pm.l2 -0.24269    0.13155  -1.845  0.06783 .
## Rainfall.l2     0.24756    0.14794   1.673  0.09718 .
## Pressure3pm.l3  0.27064    0.09814   2.758  0.00685 **
## Rainfall.l3    -0.02985    0.14538  -0.205  0.83769
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 3.668 on 107 degrees of freedom
## Multiple R-Squared:     1,   Adjusted R-squared:     1
## F-statistic: 1.428e+06 on 6 and 107 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation Rainfall:
## =========================================
## Rainfall = Pressure3pm.l1 + Rainfall.l1 + Pressure3pm.l2 + Rainfall.l2 + Pressure3pm.l3 + Rainfall.l3
##
##                Estimate Std. Error t value Pr(>|t|)
## Pressure3pm.l1 -0.15487    0.06679  -2.319  0.02232 *
## Rainfall.l1     0.28984    0.09934   2.918  0.00430 **
## Pressure3pm.l2  0.25068    0.09145   2.741  0.00717 **
## Rainfall.l2     0.07424    0.10284   0.722  0.47194
## Pressure3pm.l3 -0.09528    0.06822  -1.397  0.16540
## Rainfall.l3    -0.01213    0.10106  -0.120  0.90470
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 2.55 on 107 degrees of freedom
## Multiple R-Squared: 0.2268,  Adjusted R-squared: 0.1834
## F-statistic:  5.23 on 6 and 107 DF,  p-value: 9.324e-05
##
##
##
## Covariance matrix of residuals:
##             Pressure3pm Rainfall
## Pressure3pm      13.455   -2.178
## Rainfall         -2.178    6.501
##
## Correlation matrix of residuals:
##             Pressure3pm Rainfall
## Pressure3pm      1.0000  -0.2329
## Rainfall        -0.2329   1.0000
```

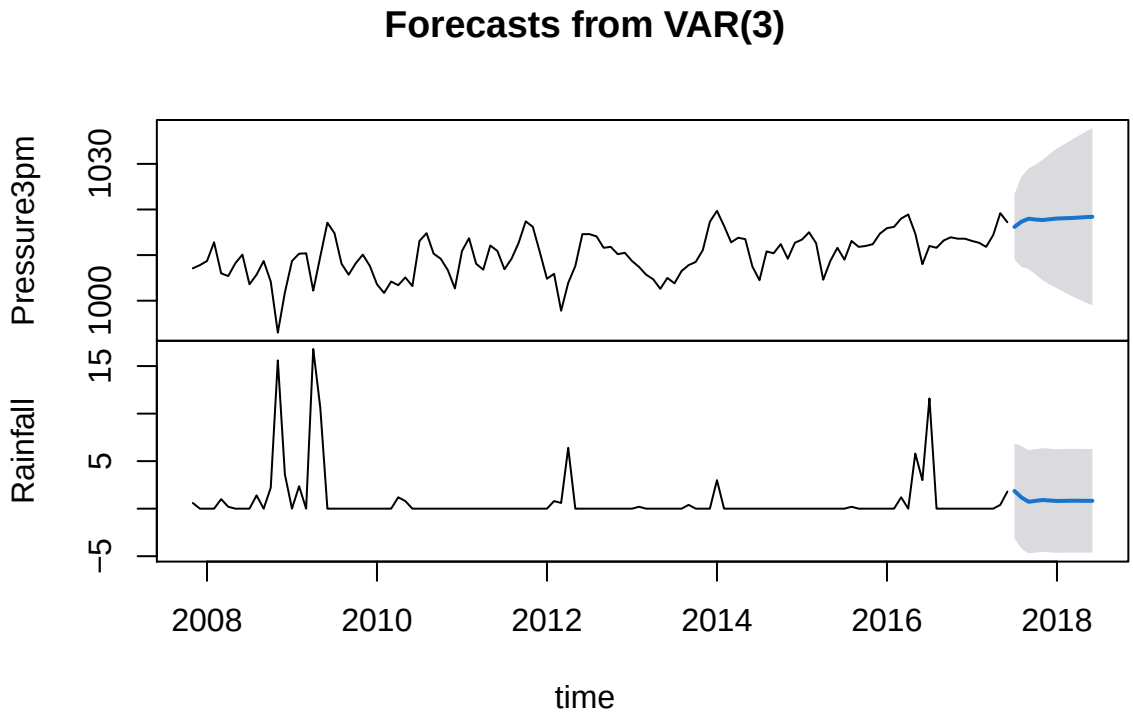
Impulse

```
ir.1 <- vars::irf(model2, impulse = "Pressure3pm", response = "Rainfall")
plot(ir.1)
```



Forecasting and plotting

```
forect2 <- forecast(model2,level=c(95),h=1*12)
plot(forect2)
```



3.Vector Autoregression Moving-Average(VARMA)

It is the combination of VAR and VMA and a generalized version of ARMA model for multivariate stationary time series. It is characterized by 'p' and 'q' parameters. Much like, ARMA is capable of acting like an AR model by setting 'q' parameter as 0 and as a MA model by setting 'p' parameter as 0, VARMA is also capable of acting like an VAR model by setting 'q' parameter as 0 and as a VMA model by setting 'p' parameter as 0.

Reasons to use VARMA model

a. More parsimonious representation

b. Closed with respect to linear transformations

Getting optimum value of p&q

```
Eccm(ts_data_vars)
```

```
## p-values table of Extended Cross-correlation Matrices:
## Column: MA order
## Row   : AR order
```

```
##        0      1      2      3      4      5      6
## 0 0.0000 0.0011 0.0777 0.1628 0.1591 0.5859 0.7096
## 1 0.0270 0.4774 0.0308 0.0001 0.0926 0.6089 0.7414
## 2 0.2878 0.2043 0.0627 0.0965 0.2134 0.9882 0.9827
## 3 0.6035 0.3002 0.5606 0.7125 0.0771 0.9957 0.9806
## 4 0.6757 0.3363 0.6363 0.9734 0.9756 0.9967 0.9960
## 5 0.9821 0.7236 0.9999 1.0000 0.9936 0.9994 0.9796
```

After computing the extended cross-correlation matrices and the associated two-way table ofmultivariate Ljung-Box statistics of a vector time series, we get p=5&q=3.

Model building

```
model3=VARMA(ts_data_vars, p = 5, q = 3, include.mean = T,
    fixed = NULL, beta=NULL, sebeta=NULL,
    prelim = F, details = F, thres = 2)
```

```
## Number of parameters:  34
## initial estimates:  586.2207 -182.7506 0.8409 0.4806 -0.3884 -0.1728 -0.4164 -0.844 0.46 0.4104 -0.0
## Par. lower-bounds:  -208.995 -774.9978 -0.4855 -0.7635 -1.9455 -1.2657 -1.9423 -1.996 -0.5056 -0.138
## Par. upper-bounds:  1381.436 409.4966 2.1673 1.7247 1.1687 0.9202 1.1094 0.3081 1.4256 0.9589 0.2189
## Final   Estimates:  586.2183 -182.7368 1.029617 0.7469378 -0.54771 0.4150632 -0.2827258 -0.8378011 0
```

```
## Warning in sqrt(diag(solve(Hessian))): NaNs produced
```

```
##
## Coefficient(s):
##               Estimate  Std. Error  t value Pr(>|t|)
## Pressure3pm  5.862e+02   2.801e+01   20.933  < 2e-16 ***
## Rainfall    -1.827e+02   9.159e+00  -19.951  < 2e-16 ***
## Pressure3pm  1.030e+00          NA       NA       NA
## Rainfall     7.469e-01          NA       NA       NA
## Pressure3pm -5.477e-01          NA       NA       NA
## Rainfall     4.151e-01          NA       NA       NA
## Pressure3pm -2.827e-01          NA       NA       NA
## Rainfall    -8.378e-01          NA       NA       NA
## Pressure3pm  3.222e-01   1.525e-01    2.112  0.03465 *
## Rainfall     3.551e-01   3.110e-01    1.142  0.25353
## Pressure3pm -1.023e-01   8.081e-02   -1.266  0.20543
## Rainfall     1.376e-03   1.125e-01    0.012  0.99025
## Pressure3pm -3.305e-02          NA       NA       NA
## Rainfall     2.682e-01   2.471e-01    1.085  0.27779
## Pressure3pm  7.995e-02          NA       NA       NA
## Rainfall    -2.898e-01          NA       NA       NA
## Pressure3pm  2.412e-01   1.013e-02   23.806  < 2e-16 ***
## Rainfall     2.621e-01          NA       NA       NA
## Pressure3pm -1.906e-01   3.179e-02   -5.996 2.03e-09 ***
## Rainfall    -1.082e-01   1.116e-01   -0.969  0.33233
## Pressure3pm  8.409e-02   4.195e-02    2.005  0.04500 *
## Rainfall     1.615e-01   5.935e-02    2.722  0.00649 **
##             -1.600e-01          NA       NA       NA
```

```
##             -5.800e-01          NA       NA       NA
##              3.387e-01          NA       NA       NA
##             -4.093e-01   4.091e-01   -1.001  0.31703
##              7.651e-01          NA       NA       NA
##              7.875e-01   3.684e-01    2.138  0.03255 *
##             -1.590e-01          NA       NA       NA
##             -5.500e-02   1.708e-01   -0.322  0.74744
##              1.688e-02          NA       NA       NA
##              3.702e-01          NA       NA       NA
##             -3.440e-01          NA       NA       NA
##             -2.163e-01   2.249e-02   -9.621  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## ---
## Estimates in matrix form:
## Constant term:
## Estimates:  586.2183 -182.7368
## AR coefficient matrix
## AR( 1 )-matrix
##        [,1]  [,2]
## [1,]  1.030 0.747
## [2,] -0.033 0.268
## AR( 2 )-matrix
##         [,1]   [,2]
## [1,] -0.5477  0.415
## [2,]  0.0799 -0.290
## AR( 3 )-matrix
##        [,1]   [,2]
## [1,] -0.283 -0.838
## [2,]  0.241  0.262
## AR( 4 )-matrix
##        [,1]   [,2]
## [1,]  0.322  0.355
## [2,] -0.191 -0.108
## AR( 5 )-matrix
##         [,1]    [,2]
## [1,] -0.1023 0.00138
## [2,]  0.0841 0.16153
## MA coefficient matrix
## MA( 1 )-matrix
##       [,1]  [,2]
## [1,] 0.160 0.580
## [2,] 0.159 0.055
## MA( 2 )-matrix
##         [,1]   [,2]
## [1,] -0.3387  0.409
## [2,] -0.0169 -0.370
## MA( 3 )-matrix
##        [,1]   [,2]
## [1,] -0.765 -0.787
## [2,]  0.344  0.216
##
## Residuals cov-matrix:
##           [,1]      [,2]
```

```
## [1,]  9.854490 -1.060693
## [2,] -1.060693  5.586300
## ----
## aic=  4.573802
## bic=  5.380889
```

After building the model, we get the values of aic and bic as 4.57 and 5.38 respectively.

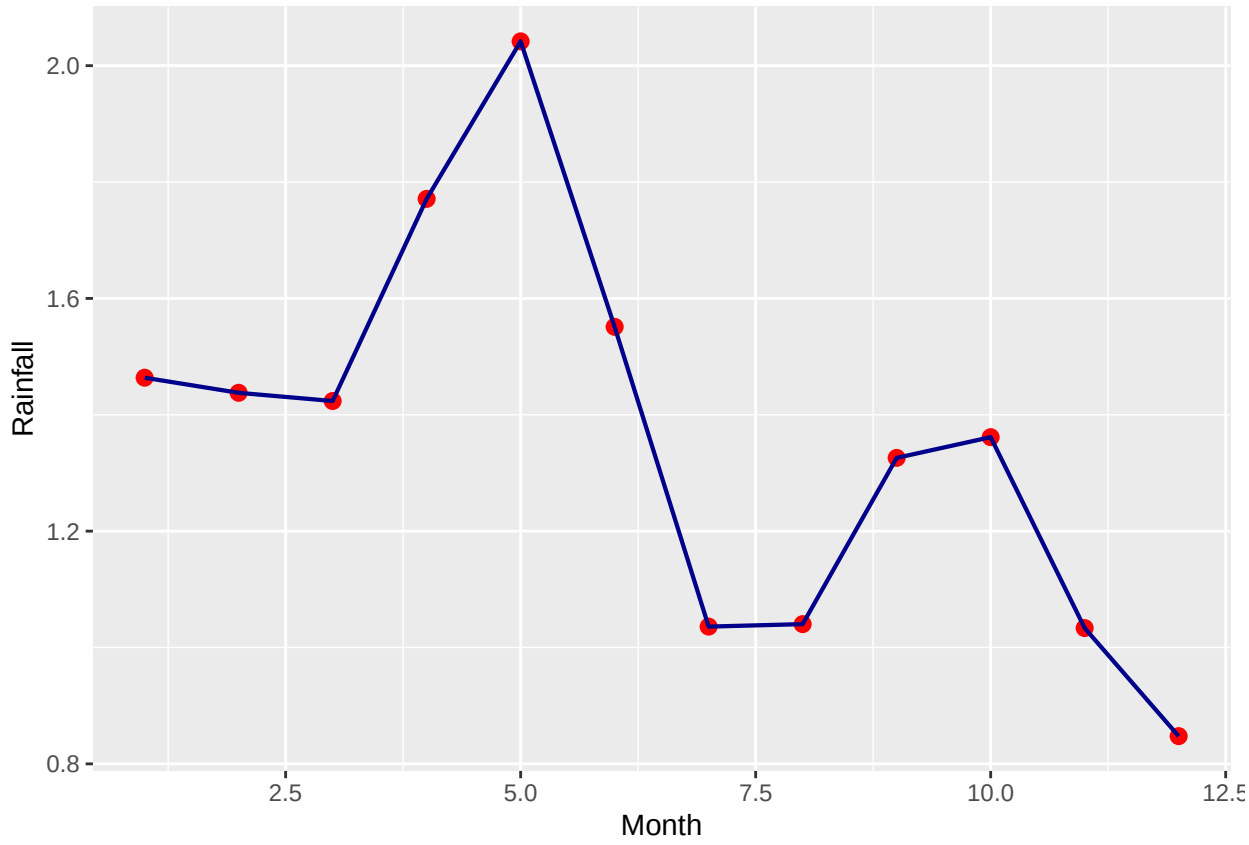
Predicting from VARMA model

```
forecast_varma=VARMApred(model3,h=1*12,orig=0)
```

```
## Predictions at origin  116
##       Pressure3pm Rainfall
##  [1,]        1015   1.4637
##  [2,]        1014   1.4377
##  [3,]        1013   1.4237
##  [4,]        1011   1.7711
##  [5,]        1010   2.0418
##  [6,]        1010   1.5511
##  [7,]        1011   1.0359
##  [8,]        1011   1.0402
##  [9,]        1010   1.3259
## [10,]        1010   1.3615
## [11,]        1011   1.0334
## [12,]        1011   0.8476
## Standard errors of predictions
##        [,1]  [,2]
##  [1,] 3.139 2.364
##  [2,] 4.142 2.508
##  [3,] 4.506 2.527
##  [4,] 4.816 2.528
##  [5,] 5.089 2.529
##  [6,] 5.115 2.593
##  [7,] 5.127 2.641
##  [8,] 5.162 2.655
##  [9,] 5.179 2.656
## [10,] 5.189 2.659
## [11,] 5.194 2.671
## [12,] 5.196 2.678
```

Vislualization of the prediction

```
tx=data.frame(forecast_varma$pred)
ggplot(tx, aes(y=Rainfall,x=seq(1,12,by=1)))+
  geom_point(color="red",size=2.5)+
  xlab("Month")+
  geom_line(color="darkblue", size=.75)
```



4.Holt-Winter's Seasonal Method

The Holt-Winter's Seasonal method is used for data with both seasonal patterns and trends. This method can be implemented either by using Additive structure or by using the Multiplicative structure depending on the data set. The Additive structure or model is used when the seasonal pattern of data has the same magnitude or is consistent throughout, while the Multiplicative structure or model is used if the magnitude of the seasonal pattern of the data increases over time. It uses three smoothing parameters,- alpha, beta, and gamma.

Building model

```
model4=ets(time_dat, model = "AAA")

summary(model4)
```

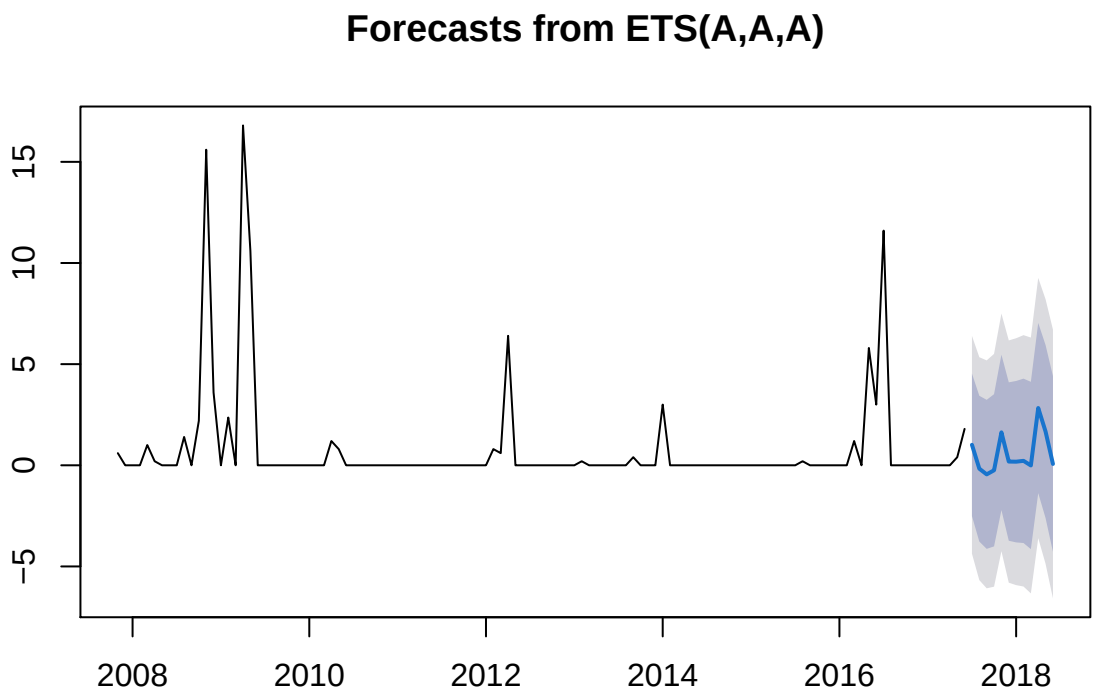
```
## ETS(A,A,A)
##
## Call:
##  ets(y = time_dat, model = "AAA")
##
##   Smoothing parameters:
##     alpha = 0.2173
##     beta  = 1e-04
```

```
##     gamma = 1e-04
##
##   Initial states:
##     l = 0.4665
##     b = 0.0074
##     s = -0.8059 -0.9998 -0.712 0.4744 -0.5488 1.0573
##           2.2309 -0.6021 -0.3668 -0.4011 -0.3882 1.0622
##
##   sigma:  2.7469
##
##      AIC     AICc      BIC
## 802.6314 808.8763 849.4425
##
## Training set error measures:
##                       ME     RMSE      MAE MPE MAPE    MASE      ACF1
## Training set -0.03117401 2.550454 1.520452 NaN  Inf 1.02732 0.1561261
```

After building the model, we get aic and bic values as 802.63 and 849.442 respectively.

Forecast plot

```
forect3 <- forecast(model4,h = 1*12)
plot(forect3)
```



Comparing all models on the basis of AIC and BIC

```
summary(model1)
```

Model-1

```
## Series: time_dat
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##          ar1    mean
##       0.3067  0.7941
## s.e.  0.0879  0.3389
##
## sigma^2 estimated as 6.564:  log likelihood=-272.77
## AIC=551.55   AICc=551.76   BIC=559.81
##
## Training set error measures:
##                        ME     RMSE      MAE  MPE MAPE      MASE       ACF1
## Training set 0.0006047933 2.539933 1.156613 -Inf  Inf 0.7814859 0.00766371
```

```
summary(model2)
```

Model-2

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: Pressure3pm, Rainfall
## Deterministic variables: none
## Sample size: 113
## Log Likelihood: -563.994
## Roots of the characteristic polynomial:
##     1 0.5886 0.5886 0.4332 0.3399 0.1201
## Call:
## vars::VAR(y = ts_data_vars, type = "none", lag.max = 5, ic = "AIC")
##
##
## Estimation results for equation Pressure3pm:
## ============================================
## Pressure3pm = Pressure3pm.l1 + Rainfall.l1 + Pressure3pm.l2 + Rainfall.l2 + Pressure3pm.l3 + Rainfall
##
##                Estimate Std. Error t value Pr(>|t|)
## Pressure3pm.l1  0.97181    0.09609  10.114  < 2e-16 ***
## Rainfall.l1     0.20782    0.14290   1.454  0.14881
## Pressure3pm.l2 -0.24269    0.13155  -1.845  0.06783 .
## Rainfall.l2     0.24756    0.14794   1.673  0.09718 .
## Pressure3pm.l3  0.27064    0.09814   2.758  0.00685 **
```

```
## Rainfall.l3    -0.02985    0.14538  -0.205  0.83769
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 3.668 on 107 degrees of freedom
## Multiple R-Squared:     1,   Adjusted R-squared:     1
## F-statistic: 1.428e+06 on 6 and 107 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation Rainfall:
## =========================================
## Rainfall = Pressure3pm.l1 + Rainfall.l1 + Pressure3pm.l2 + Rainfall.l2 + Pressure3pm.l3 + Rainfall.l3
##
##                Estimate Std. Error t value Pr(>|t|)
## Pressure3pm.l1 -0.15487    0.06679  -2.319  0.02232 *
## Rainfall.l1     0.28984    0.09934   2.918  0.00430 **
## Pressure3pm.l2  0.25068    0.09145   2.741  0.00717 **
## Rainfall.l2     0.07424    0.10284   0.722  0.47194
## Pressure3pm.l3 -0.09528    0.06822  -1.397  0.16540
## Rainfall.l3    -0.01213    0.10106  -0.120  0.90470
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 2.55 on 107 degrees of freedom
## Multiple R-Squared: 0.2268,  Adjusted R-squared: 0.1834
## F-statistic:  5.23 on 6 and 107 DF,  p-value: 9.324e-05
##
##
##
## Covariance matrix of residuals:
##             Pressure3pm Rainfall
## Pressure3pm      13.455   -2.178
## Rainfall         -2.178    6.501
##
## Correlation matrix of residuals:
##             Pressure3pm Rainfall
## Pressure3pm      1.0000  -0.2329
## Rainfall        -0.2329   1.0000
```

```
summary(model4)
```

Model-4

```
## ETS(A,A,A)
##
## Call:
##  ets(y = time_dat, model = "AAA")
##
##   Smoothing parameters:
```

```
##     alpha = 0.2173
##     beta  = 1e-04
##     gamma = 1e-04
##
##   Initial states:
##     l = 0.4665
##     b = 0.0074
##     s = -0.8059 -0.9998 -0.712 0.4744 -0.5488 1.0573
##           2.2309 -0.6021 -0.3668 -0.4011 -0.3882 1.0622
##
##   sigma:  2.7469
##
##      AIC     AICc      BIC
## 802.6314 808.8763 849.4425
##
## Training set error measures:
##                       ME     RMSE      MAE MPE MAPE    MASE      ACF1
## Training set -0.03117401 2.550454 1.520452 NaN  Inf 1.02732 0.1561261
```

Inferences:

1. The better model when two models are compared, the one with less AiC and BiC values are better models.

2. According to above summary, we can conclude that VARMA model is best performing model.

Classification

Preprocesssing

Structure of the data

```
class_data=data
class_data=class_data[,c(-1,-2,-8,-10,-11,-24)]
str(class_data)
```

```
## 'data.frame':    145460 obs. of  18 variables:
##  $ MinTemp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
##  $ MaxTemp      : num  22.9 25.1 25.7 28 32.3 29.7 25 26.7 31.9 30.1 ...
##  $ Rainfall     : num  0.6 0 0 0 1 0.2 0 0 0 1.4 ...
##  $ Evaporation  : num  5.47 5.47 5.47 5.47 5.47 ...
##  $ Sunshine     : num  7.61 7.61 7.61 7.61 7.61 ...
##  $ WindGustSpeed: num  44 44 46 24 41 56 50 35 80 28 ...
##  $ WindSpeed9am : num  20 4 19 11 7 19 20 6 7 15 ...
##  $ WindSpeed3pm : num  24 22 26 9 20 24 24 17 28 11 ...
##  $ Humidity9am  : num  71 44 38 45 82 55 49 48 42 58 ...
##  $ Humidity3pm  : num  22 25 30 16 33 23 19 19 9 27 ...
##  $ Pressure9am  : num  1008 1011 1008 1018 1011 ...
```

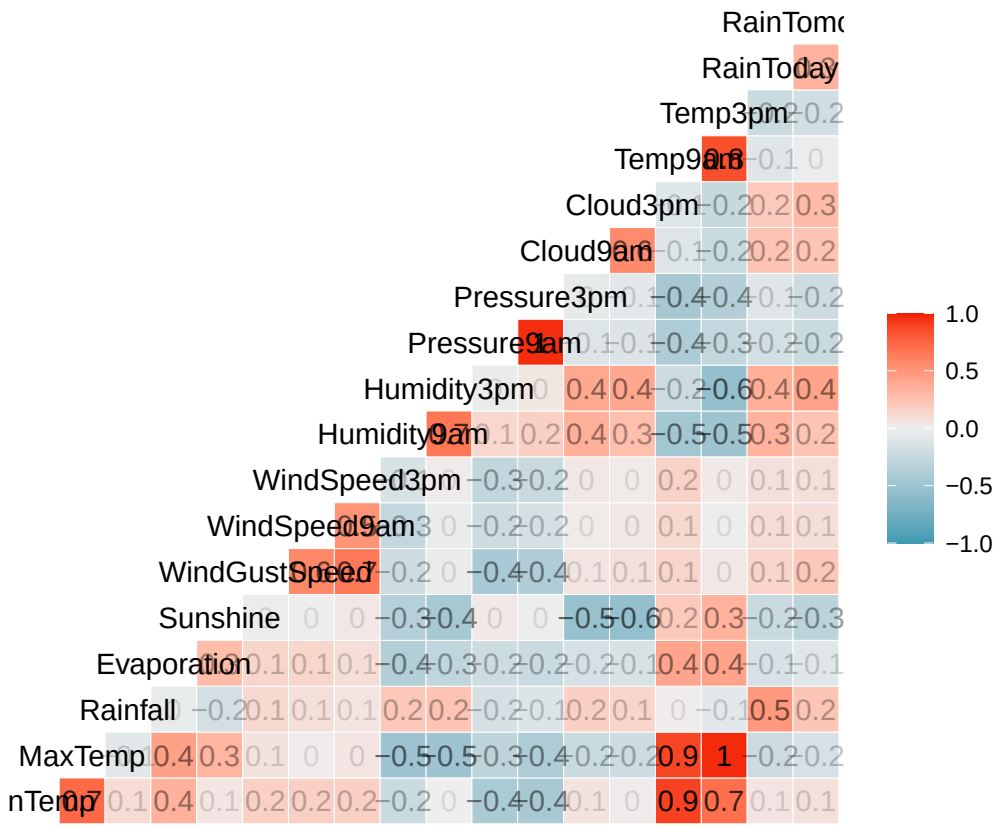
```
##  $ Pressure3pm  : num  1007 1008 1009 1013 1006 ...
##  $ Cloud9am     : num  8 4.45 4.45 4.45 7 ...
##  $ Cloud3pm     : num  4.51 4.51 2 4.51 8 ...
##  $ Temp9am      : num  16.9 17.2 21 18.1 17.8 20.6 18.1 16.3 18.3 20.1 ...
##  $ Temp3pm      : num  21.8 24.3 23.2 26.5 29.7 28.9 24.6 25.5 30.2 28.2 ...
##  $ RainToday    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ RainTomorrow : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
```

Encoding in 0,1

```
class_data$RainToday=as.numeric(class_data$RainToday)
class_data$RainTomorrow=as.numeric(class_data$RainTomorrow)

class_data=class_data%>%
  mutate(RainToday=replace(RainToday, is.na(RainToday), 2))

class_data=class_data%>%
  mutate(RainTomorrow=replace(RainTomorrow, is.na(RainTomorrow), 2))

class_data$RainToday=ifelse(class_data$RainToday>1,1,0)
class_data$RainTomorrow=ifelse(class_data$RainTomorrow>1,1,0)
```

Correlation Plot

```
ggcorr(class_data,label=TRUE,label_alpha = TRUE)
```



1. Logistic Regression

Splitting data into training and testing

```
set.seed(123)
split = sample.split(class_data$RainToday, SplitRatio = 0.8)
train_data = subset(class_data, split == TRUE)
test_data = subset(class_data, split == FALSE)
```

Here we are splitting the data into 8:2 ratio.

Model Building

Model1

```
model1 <- glm(RainToday~Humidity3pm+Temp3pm+Pressure3pm,data=train_data,family = "binomial")
summary(model1)$coeff
```

```
##                Estimate   Std. Error   z value     Pr(>|z|)
## (Intercept) 48.82465756 1.2145065650  40.20123  0.00000e+00
## Humidity3pm  0.04078243 0.0004611149  88.44310  0.00000e+00
```

```
## Temp3pm     -0.04425208 0.0014031076 -31.53862 2.56843e-218
## Pressure3pm -0.05057693 0.0011826901 -42.76431  0.00000e+00
```

Here we can see that Temp3pm is significant(since, p-val<0.05).

Model2

```
model2 <- glm(RainToday~Humidity9am+Temp9am+Pressure9am,data=train_data,family = "binomial")
summary(model2)$coeff
```

```
##                 Estimate   Std. Error   z value  Pr(>|z|)
## (Intercept) 79.689942686 1.2604098834  63.22542 0.0000000
## Humidity9am  0.060083299 0.0005605483 107.18665 0.0000000
## Temp9am      0.002929732 0.0013724842   2.13462 0.0327921
## Pressure9am -0.083853311 0.0012333707 -67.98711 0.0000000
```

Here we can see that Temp9am is significant(since, p-val<0.05).

Model3

```
model3 <- glm(RainToday~.-Rainfall-RainTomorrow,data=train_data,family = "binomial")
summary(model3)$coeff
```

```
##                   Estimate   Std. Error    z value      Pr(>|z|)
## (Intercept)   38.419693041 1.4688861187  26.155665 8.496321e-151
## MinTemp        0.125641346 0.0041786592  30.067383 1.294016e-198
## MaxTemp       -0.171433741 0.0053424283 -32.089105 6.256558e-226
## Evaporation   -0.081797293 0.0039697176 -20.605318  2.458999e-94
## Sunshine       0.010591375 0.0038831785   2.727502  6.381595e-03
## WindGustSpeed  0.026688095 0.0009153084  29.157491 6.713406e-187
## WindSpeed9am   0.017866448 0.0012387398  14.423084  3.704130e-47
## WindSpeed3pm  -0.020111073 0.0012869986 -15.626336  4.817169e-55
## Humidity9am    0.059614995 0.0008928436  66.769809  0.000000e+00
## Humidity3pm   -0.005759873 0.0008639659  -6.666782  2.614723e-11
## Pressure9am   -0.206066345 0.0048485608 -42.500518  0.000000e+00
## Pressure3pm    0.163669193 0.0048212945  33.947147 1.343917e-252
## Cloud9am       0.019989318 0.0048406194   4.129496  3.635596e-05
## Cloud3pm       0.028937830 0.0052451046   5.517112  3.446158e-08
## Temp9am        0.065363468 0.0056923831  11.482619  1.613181e-30
## Temp3pm        0.006307112 0.0056981723   1.106866  2.683519e-01
```

In model3, we can observe that, all the attributes are significant.

Predicting the output and getting the accuracy

Model1

```
proba <- model1 %>%
  predict(test_data,type="response")
head(proba)
```

```
##          4          5          8         15         19         23
## 0.05118248 0.11668525 0.06807374 0.12706318 0.13387468 0.08130494
```

```
pred_class <- ifelse(proba<0.5,0,1)
head(pred_class)
```

Class Prediction

```
##  4  5  8 15 19 23
##  0  0  0  0  0  0
```

Confusion Matrix for model1

```
table(test_data$RainToday,pred_class)
```

```
##    pred_class
##         0     1
##   0 20811  1253
##   1  5355  1673
```

Accuracy

```
mean(pred_class==test_data$RainToday)
```

```
## [1] 0.7728585
```

The accuracy of model-1 is 77.28%

Model2

```
proba <- model2 %>%
  predict(test_data,type="response")
head(proba)
```

```
##          4          5          8         15         19         23
## 0.05302997 0.47751405 0.08664939 0.18161931 0.11083268 0.18194896
```

```
pred_class <- ifelse(proba<0.5,0,1)
head(pred_class)
```

Class Prediction

```
##  4  5  8 15 19 23
##  0  0  0  0  0  0
```

Accuracy

```
mean(pred_class==test_data$RainToday)
```

```
## [1] 0.7876048
```

The accuracy of model-1 is 78.76%

Confusion Matrix for model1

```
table(test_data$RainToday,pred_class)
```

```
##    pred_class
##         0     1
##   0 20750  1314
##   1  4865  2163
```

Model3

```
proba <- model3 %>%
  predict(test_data,type="response")
head(proba)
```

```
##           4           5           8          15          19          23
## 0.008977724 0.155961016 0.015291257 0.050994567 0.119287176 0.140307665
```

```
pred_class <- ifelse(proba<0.5,0,1)
head(pred_class)
```

Class Prediction

```
##  4  5  8 15 19 23
##  0  0  0  0  0  0
```

Confusion Matrix for model3

```
table(test_data$RainToday,pred_class)
```

```
##    pred_class
##         0     1
##   0 20550  1514
##   1  3892  3136
```

Accuracy

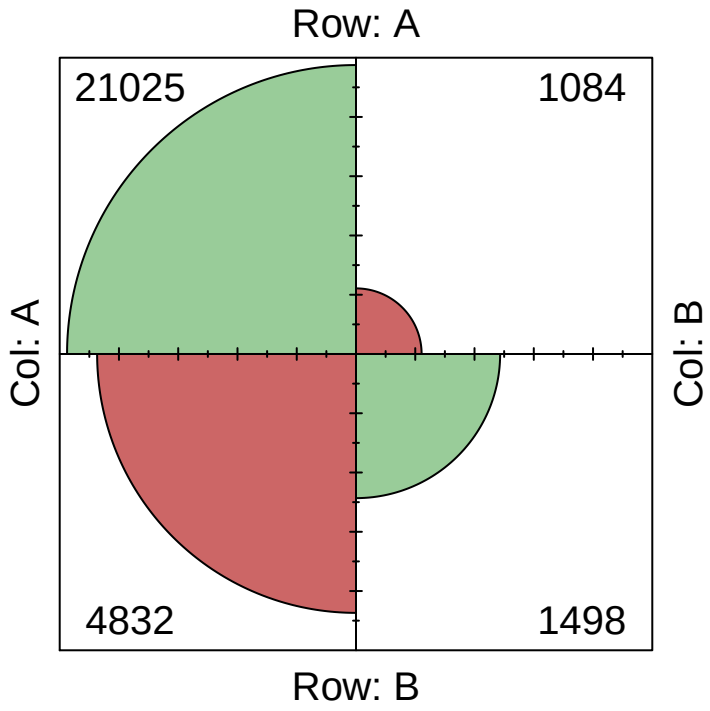
```
mean(pred_class==test_data$RainToday)
```

```
## [1] 0.8141757
```

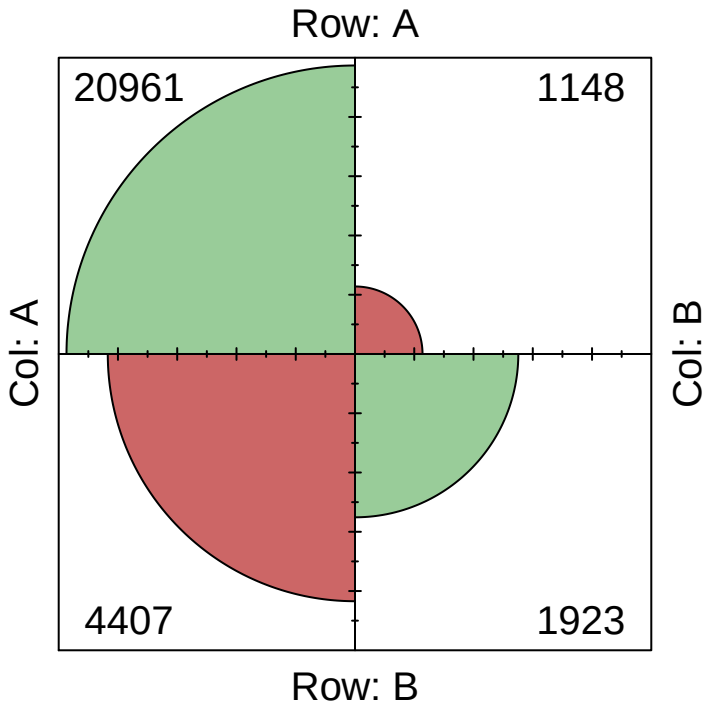
The accuracy of model-1 is 81.41%

Visualising the Confusion Matrix

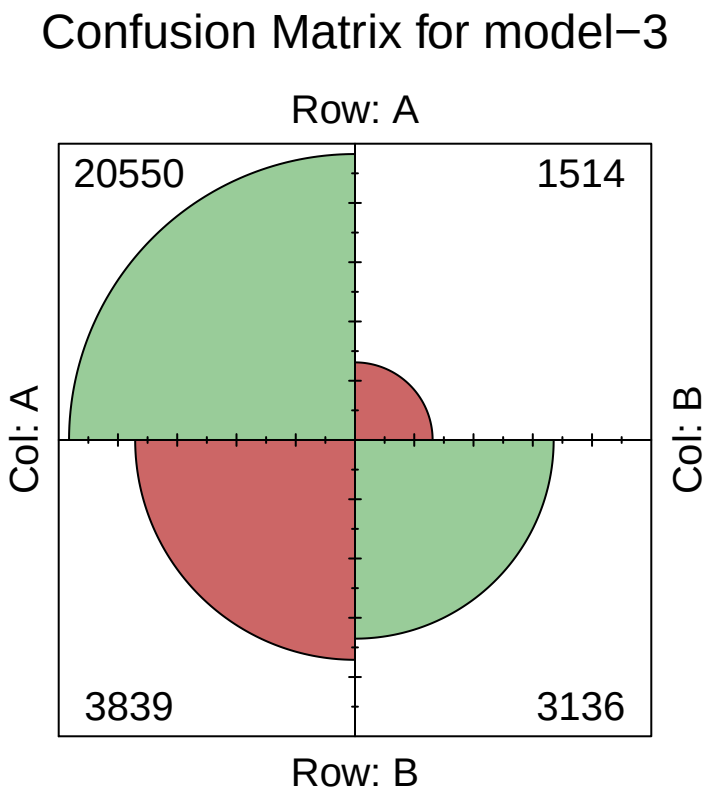
```
ctable <- as.table(matrix(c(21025, 1084, 4832, 1498), nrow = 2, byrow = TRUE))
plot1=fourfoldplot(ctable, color = c("#CC6666", "#99CC99"),
             conf.level = 0, margin = 1, main = "Confusion Matrix for model-1")
```




```
ctable <- as.table(matrix(c(20961, 1148, 4407, 1923), nrow = 2, byrow = TRUE))
plot1=fourfoldplot(ctable, color = c("#CC6666", "#99CC99"),
             conf.level = 0, margin = 1, main = "Confusion Matrix for model-2")
```




```
ctable <- as.table(matrix(c(20550, 1514, 3839, 3136), nrow = 2, byrow = TRUE))
plot1=fourfoldplot(ctable, color = c("#CC6666", "#99CC99"),
             conf.level = 0, margin = 1, main = "Confusion Matrix for model-3")
```



Checking Assumptions

```
vif(model1)
```

a.Multicollinearity

```
## Humidity3pm     Temp3pm Pressure3pm
##    1.212549    1.357598    1.138781
```

```
vif(model2)
```

Since, all the values are less than 5, we can conclude that there is no multicollinearity.

```
## Humidity9am     Temp9am Pressure9am
##    1.245679    1.397464    1.180271
```

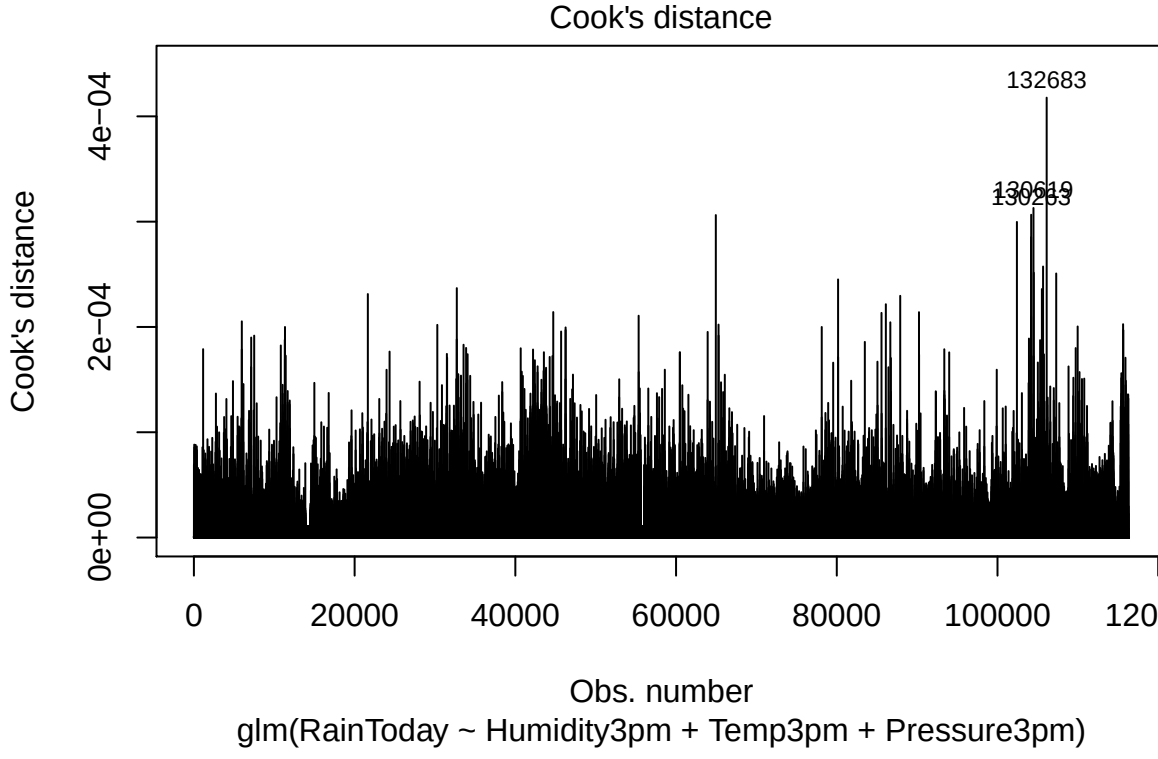
```
vif(model3)
```

Since, all the values are less than 5, we can conclude that there is no multicollinearity in model2 as well.

```
##       MinTemp       MaxTemp   Evaporation      Sunshine WindGustSpeed
##     10.684660     17.985812      1.317414      1.678540      2.308052
##  WindSpeed9am  WindSpeed3pm   Humidity9am   Humidity3pm   Pressure9am
##      1.897066      2.015996      2.572086      3.365428     16.359538
##   Pressure3pm      Cloud9am      Cloud3pm       Temp9am       Temp3pm
##     16.324931      1.523306      1.578307     19.637307     19.006800
```

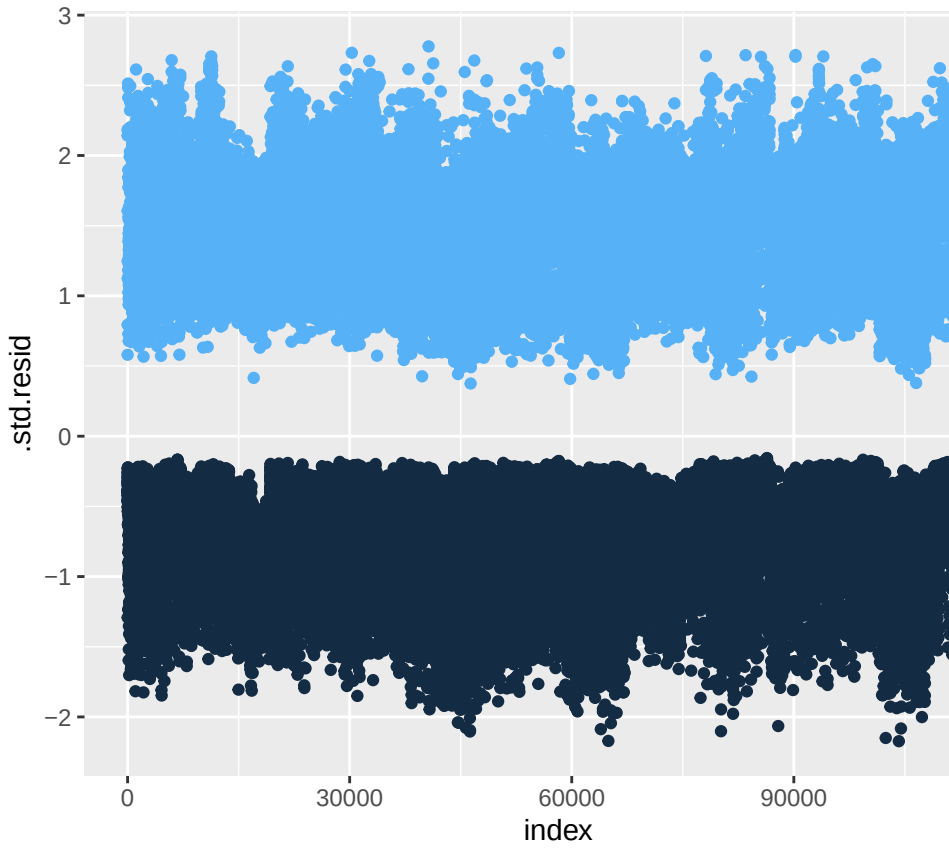
Since, some of the values are not than 5, we can conclude that there is multicollinearity in model3.

```
plot(model1,4)
```



b. Finding Outliers

```
model_data <- augment(model1) %>%
              mutate(index=1:n())

ggplot(model_data,aes(index,.std.resid))+geom_point(aes(col=RainToday))
```



Plotting the standardized residuals

2.Decision Tree

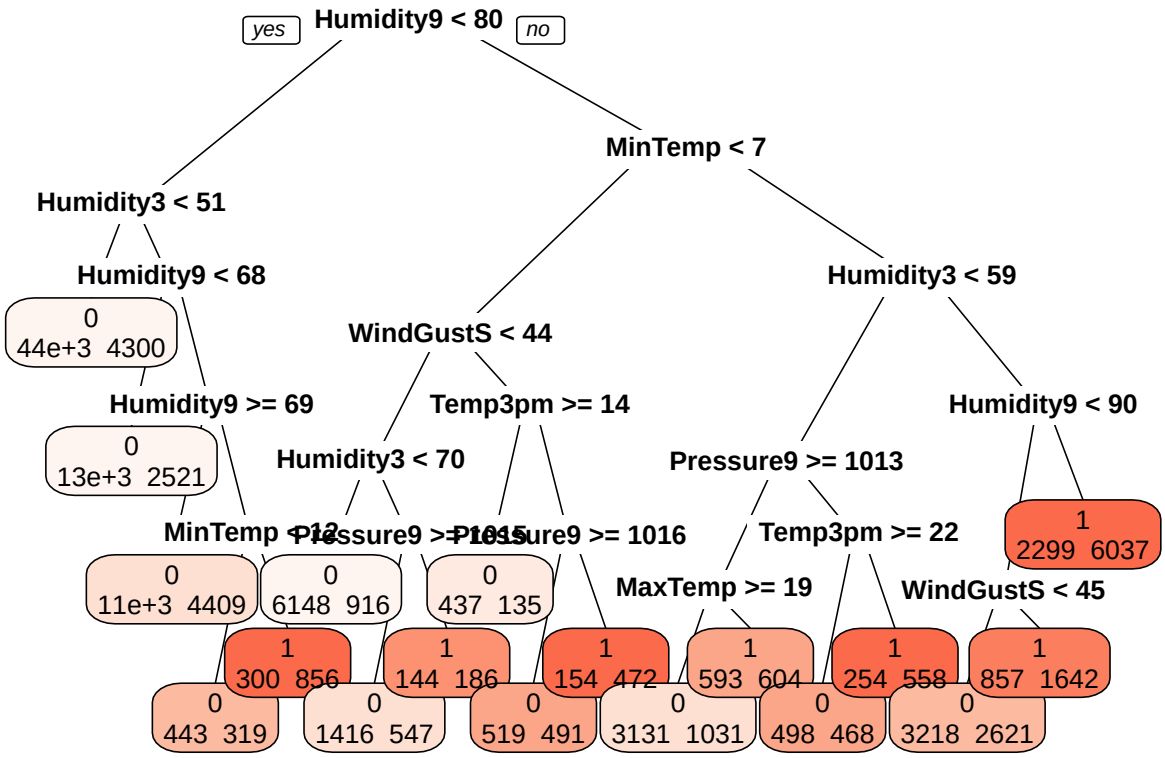
Model Building

```
model1<-rpart(RainToday~.-Rainfall-RainTomorrow,data=train_data, method="class",control =rpart.control(
```

Here we are taking all the rows, except Rainfall and RainTommorow.

Visualisation1 of the decision tree

```
prp(model1, faclen = 0,box.palette = "Reds", cex = 0.8, extra = 1)
```



Accuracy before pruining

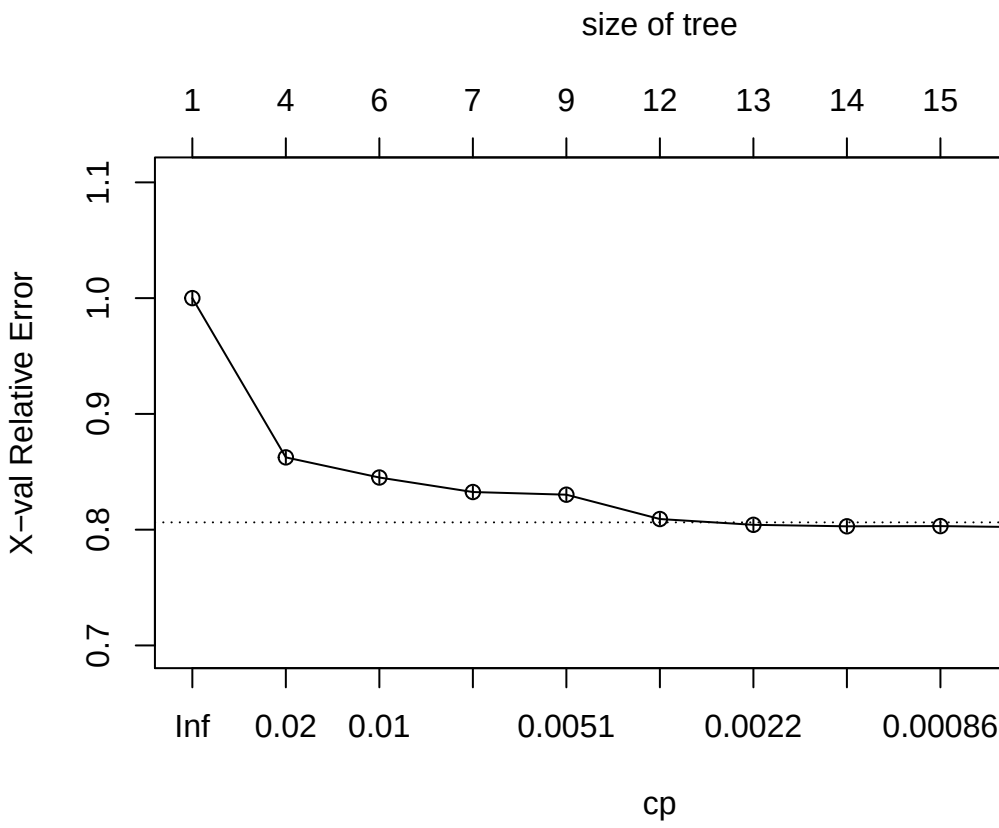
```
pred <- model1%>%predict(test_data,type="class")
mean(pred==test_data$RainToday)
```

```
## [1] 0.8034167
```

The accuracy of Decision Tree is 80.31% before pruining.

Pruning the tree

```
plotcp(model1)
```

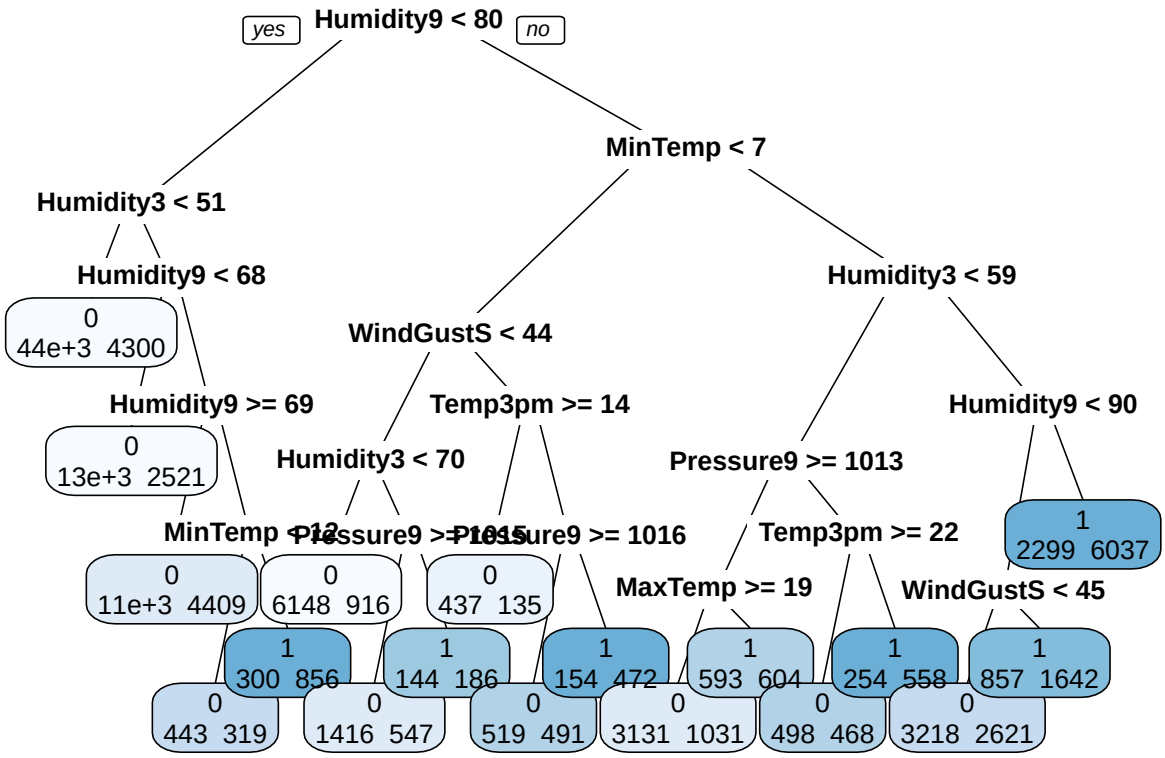


Finding the best cp using graph

From the above plot, we can observe, the cp-val is '0' for least error.

Building model after pruining:

```
bestcp <- model1$cptable[which.min(model1$cptable[,"xerror"]),"CP"]
pruned <- prune(model1, cp = bestcp)
prp(pruned, box.palette = "Blues",faclen = 0, cex = 0.8, extra = 1)
```



After pruining also, we got the same tree as of before.

Accuracy after pruining

```
pred <- pruned%>%predict(test_data,type="class")
mean(pred==test_data$RainToday)
```

```
## [1] 0.8034167
```

Since, both the trees are same the accuracy is also same i.e 80.34%

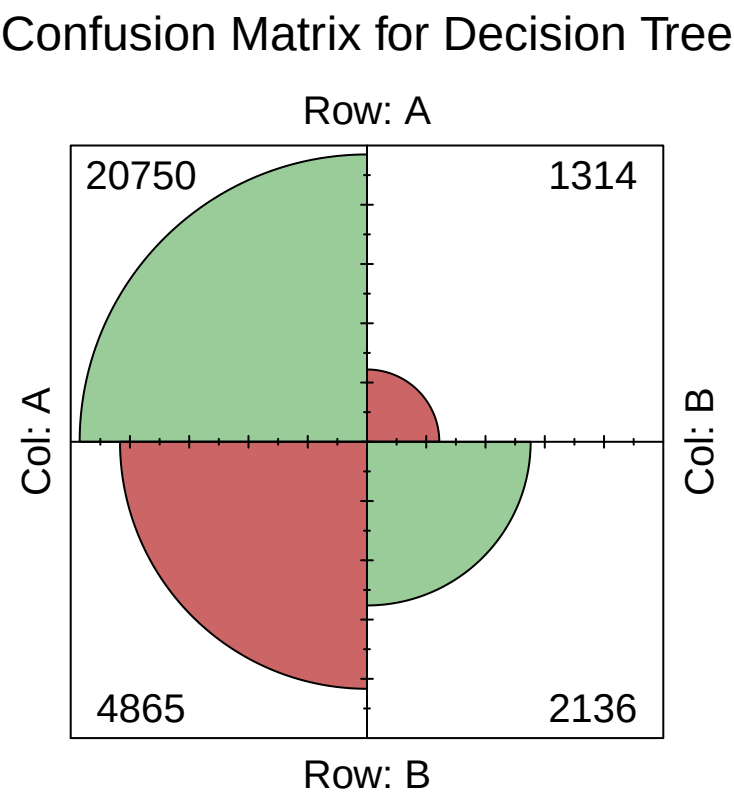
Confusion matrix

```
table(test_data$RainToday,pred_class)
```

```
##    pred_class
##         0     1
##   0 20550  1514
##   1  3892  3136
```

Visualising Confusion matrix

```
ctable <- as.table(matrix(c(20750, 1314, 4865, 2136), nrow = 2, byrow = TRUE))
plot1=fourfoldplot(ctable, color = c("#CC6666", "#99CC99"),
            conf.level = 0, margin = 1, main = "Confusion Matrix for Decision Tree")
```



3. K-Nearest Neighbour(KNN Algorithm)

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

Model Building

```
prc_test_pred <- knn(train = train_data, test = test_data,cl = train_data$RainToday , k=1)
```

Accuracy

```
mean(prc_test_pred==test_data$RainToday)
```

```
## [1] 0.8635707
```

The accuracy of KNN model for k=1 is 86.35%

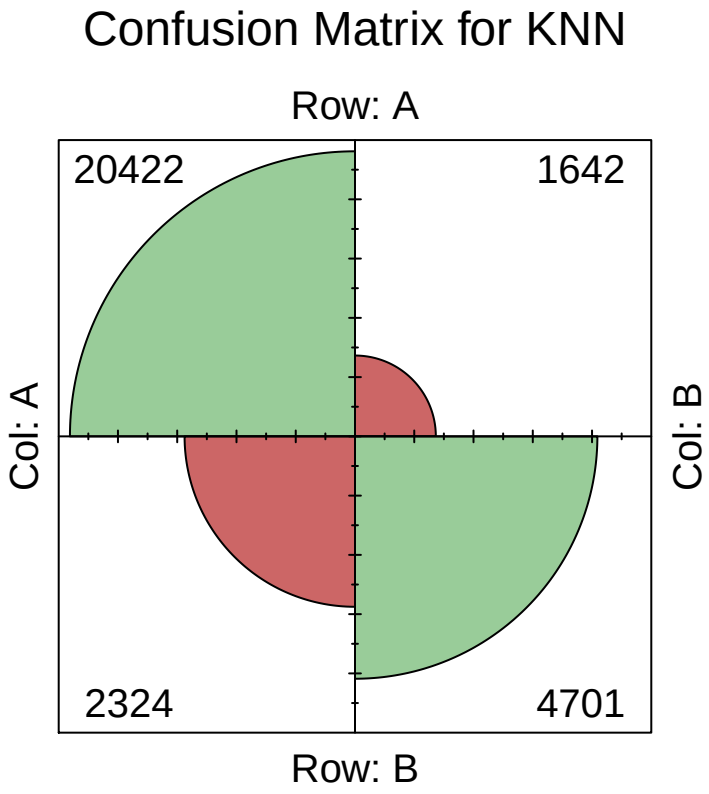
Confusion Matrix

```
table(test_data$RainToday,prc_test_pred)
```

```
##    prc_test_pred
##         0     1
##   0 20422  1642
##   1  2327  4701
```

Visualising confusion matrix

```
ctable <- as.table(matrix(c(20422, 1642, 2324, 4701), nrow = 2, byrow = TRUE))
plot1=fourfoldplot(ctable, color = c("#CC6666", "#99CC99"),
             conf.level = 0, margin = 1, main = "Confusion Matrix for KNN")
```



Tuning the value of k

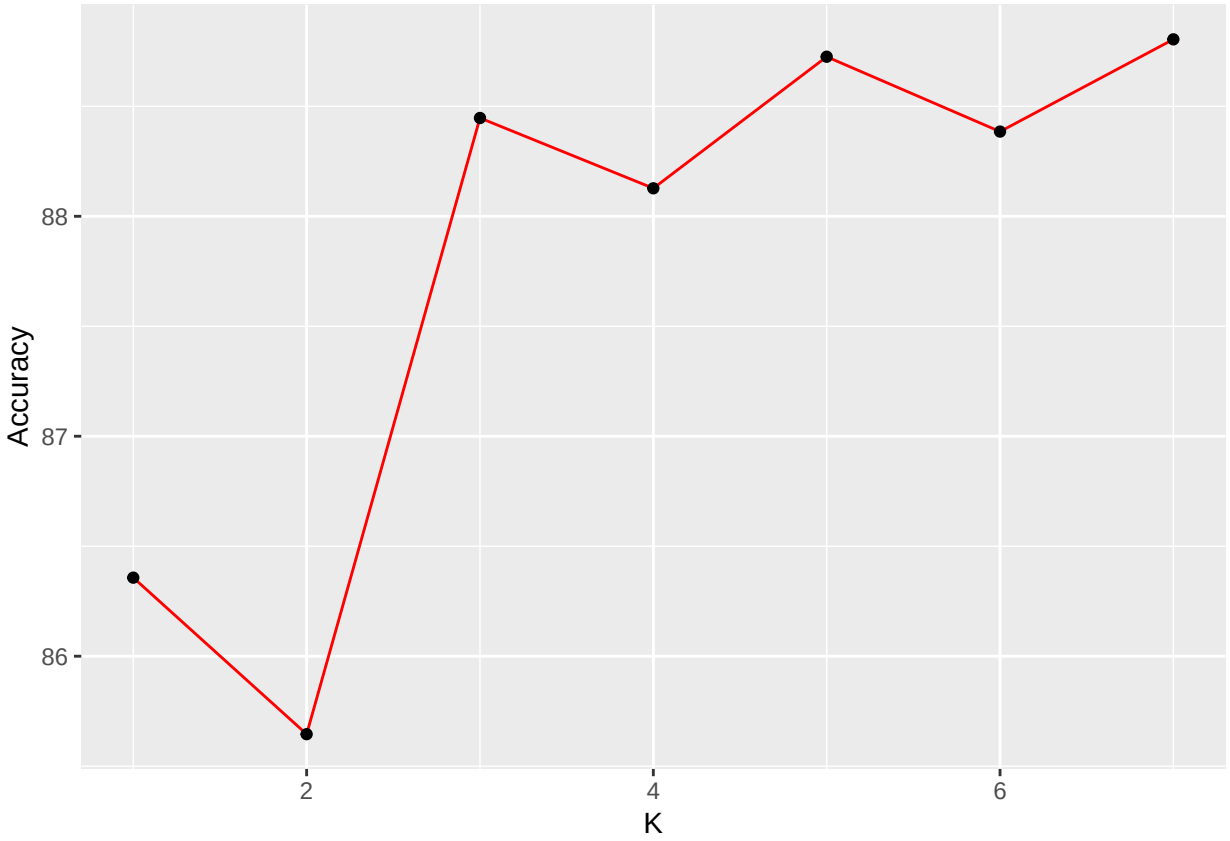
```
i=1                          # declaration to initiate for loop
k.optm=1                     # declaration to initiate for loop
for (i in 1:7){
    knn.mod <-  knn(train = train_data, test = test_data,cl = train_data$RainToday , k=i)
    k.optm[i] <- 100 * sum(test_data$RainToday == knn.mod)/NROW(test_data$RainToday)
    k=i
    cat(k,'=',k.optm[i],'\n')       # to print % accuracy
}
```

```
## 1 = 86.35707
## 2 = 85.64554
## 3 = 88.447
## 4 = 88.12732
## 5 = 88.72542
## 6 = 88.38512
## 7 = 88.80448
```

Above table shows how accuracy, changes with 'K' value

Visulising the best value of 'k'

```
k_values=seq(1,7, by=1)
temp_df=data.frame(k_values, k.optm)

ggplot(data=temp_df, aes(x=k_values, y=k.optm, group=1)) +
  geom_line(color="red")+
  geom_point()+
  ylab("Accuracy")+
  xlab("K")
```



From above plot, we can observe that K=7, we are getting highest accuracy.

Effect of Rainfall on Agriculture in Australia

Reading Dataset

```
summercrop=read.csv("./AgriCultureData/SummerCropProduction.csv")
wintercrop=read.csv("./AgriCultureData/WinterCropProduction.csv")
```

Structure of dataset

```
str(summercrop)
```

SummerCrop

```
## 'data.frame':    32 obs. of  7 variables:
##  $ Year                      : chr  "1989-90" "1990-91" "1991-92" "1992-93" ...
##  $ New.South.Wales.Area      : int  481 464 543 488 517 524 574 655 635 914 ...
##  $ New.South.Wales.Production: int  1716 1612 2079 1612 1806 1869 2077 2653 2595 3227 ...
```

```
##  $ Queensland.Area           : int  440 578 658 515 651 761 846 747 692 810 ...
##  $ Queensland.Production     : int  941 995 1530 641 1214 1279 1535 1553 1186 1813 ...
##  $ Australia.Area            : int  932 1053 1212 1014 1183 1306 1434 1416 1341 1734 ...
##  $ Australia.Production      : int  2676 2625 3627 2280 3052 3187 3637 4240 3816 5065 ...
```

```
head(summercrop)
```

Top rows

```
##      Year New.South.Wales.Area New.South.Wales.Production Queensland.Area
## 1 1989-90                  481                        1716             440
## 2 1990-91                  464                        1612             578
## 3 1991-92                  543                        2079             658
## 4 1992-93                  488                        1612             515
## 5 1993-94                  517                        1806             651
## 6 1994-95                  524                        1869             761
##   Queensland.Production Australia.Area Australia.Production
## 1                   941            932                 2676
## 2                   995           1053                 2625
## 3                  1530           1212                 3627
## 4                   641           1014                 2280
## 5                  1214           1183                 3052
## 6                  1279           1306                 3187
```

```
str(wintercrop)
```

WinterCrop

```
## 'data.frame':    32 obs. of  8 variables:
##  $ Year             : chr  "1989-90" "1990-91" "1991-92" "1992-93" ...
##  $ Unit             : chr  "kt" "kt" "kt" "kt" ...
##  $ New.South.Wales  : chr  "4841" "5827" "3910" "5951" ...
##  $ Victoria         : chr  "3,343" "2,712" "2,815" "4,121" ...
##  $ Queensland       : chr  "1,829" "2,487" "462" "1,054" ...
##  $ South.Australia  : chr  "4,860" "3,901" "4,534" "4,798" ...
##  $ Western.Australia: chr  "6,606" "7,347" "7,183" "8,500" ...
##  $ Australia        : chr  "21,519" "22,326" "18,962" "24,487" ...
```

```
head(wintercrop)
```

Top rows

```
##      Year Unit New.South.Wales Victoria Queensland South.Australia
## 1 1989-90   kt            4841    3,343      1,829           4,860
```

```
## 2 1990-91   kt            5827    2,712      2,487          3,901
## 3 1991-92   kt            3910    2,815        462          4,534
## 4 1992-93   kt            5951    4,121      1,054          4,798
## 5 1993-94   kt            7619    4,480        834          4,923
## 6 1994-95   kt            1528    1,864        312          3,044
##   Western.Australia Australia
## 1             6,606    21,519
## 2             7,347    22,326
## 3             7,183    18,962
## 4             8,500    24,487
## 5             9,873    27,797
## 6             7,924    14,719
```

Data Cleaning

```
summercrop$Year=substr(summercrop$Year,1,4)
summercrop$Year=as.numeric(summercrop$Year)
str(summercrop)
```

Converting year from chr to numeric of summercrop

```
## 'data.frame':    32 obs. of  7 variables:
##  $ Year                      : num  1989 1990 1991 1992 1993 ...
##  $ New.South.Wales.Area      : int  481 464 543 488 517 524 574 655 635 914 ...
##  $ New.South.Wales.Production: int  1716 1612 2079 1612 1806 1869 2077 2653 2595 3227 ...
##  $ Queensland.Area           : int  440 578 658 515 651 761 846 747 692 810 ...
##  $ Queensland.Production     : int  941 995 1530 641 1214 1279 1535 1553 1186 1813 ...
##  $ Australia.Area            : int  932 1053 1212 1014 1183 1306 1434 1416 1341 1734 ...
##  $ Australia.Production      : int  2676 2625 3627 2280 3052 3187 3637 4240 3816 5065 ...
```

```
wintercrop$Year=substr(wintercrop$Year,1,4)
wintercrop$Year=as.numeric(wintercrop$Year)
wintercrop$New.South.Wales=as.numeric(gsub(",","",wintercrop$New.South.Wales))
wintercrop$Victoria=as.numeric(gsub(",","",wintercrop$Victoria))
wintercrop$Queensland=as.numeric(gsub(",","",wintercrop$Queensland))
wintercrop$South.Australia=as.numeric(gsub(",","",wintercrop$South.Australia))
wintercrop$Western.Australia=as.numeric(gsub(",","",wintercrop$Western.Australia))
wintercrop$Australia=as.numeric(gsub(",","",wintercrop$Australia))

str(wintercrop)
```

Converting from chr to numeric of summercrop

```
## 'data.frame':    32 obs. of  8 variables:
##  $ Year             : num  1989 1990 1991 1992 1993 ...
##  $ Unit             : chr  "kt" "kt" "kt" "kt" ...
##  $ New.South.Wales  : num  4841 5827 3910 5951 7619 ...
```

```
##  $ Victoria         : num  3343 2712 2815 4121 4480 ...
##  $ Queensland       : num  1829 2487 462 1054 834 ...
##  $ South.Australia  : num  4860 3901 4534 4798 4923 ...
##  $ Western.Australia: num  6606 7347 7183 8500 9873 ...
##  $ Australia        : num  21519 22326 18962 24487 27797 ...
```

```
agriculture_data=split_data
agriculture_data=agriculture_data%>%
  mutate(State=case_when(
    Location=="Albury" | Location=="BadgerysCreek" | Location=="Cobar" | Location=="CoffsHarbour" | Loca
    Location=="Tuggeranong" | Location=="MountGinini" | Location=="Ballarat" | Location=="Bendigo" | Lo
    Location=="Brisbane" | Location=="Cairns" | Location=="GoldCoast" | Location=="Townsville" ~ "Queens
    Location=="Adelaide" | Location=="MountGambier" | Location=="Nuriootpa" | Location=="Woomera" ~ "Sou
    Location=="Albany" | Location=="Witchcliffe" | Location=="PearceRAAF" | Location=="PerthAirport" | 
    Location=="Hobart" | Location=="Hobart" | Location=="Launceston" | Location=="AliceSprings" | Locati
  ))

str(agriculture_data)
```

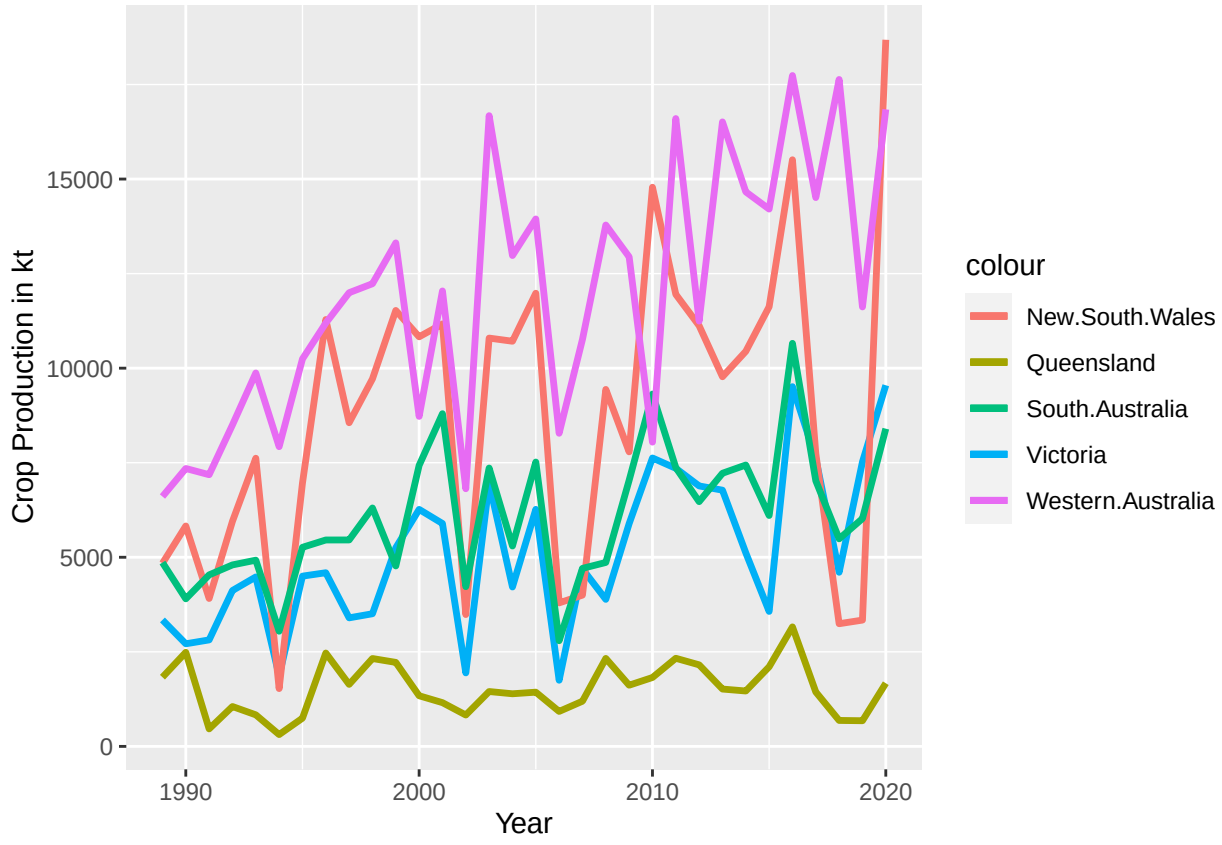
Grouping location on the basis of the states they correspond to.

```
## 'data.frame':    145460 obs. of  28 variables:
##  $ Date         : POSIXct, format: "2008-12-01" "2008-12-02" ...
##  $ Location     : chr  "Albury" "Albury" "Albury" "Albury" ...
##  $ MinTemp      : num  13.4 7.4 12.9 9.2 17.5 14.6 14.3 7.7 9.7 13.1 ...
##  $ MaxTemp      : num  22.9 25.1 25.7 28 32.3 29.7 25 26.7 31.9 30.1 ...
##  $ Rainfall     : num  0.6 0 0 0 1 0.2 0 0 0 1.4 ...
##  $ Evaporation  : num  5.47 5.47 5.47 5.47 5.47 ...
##  $ Sunshine     : num  7.61 7.61 7.61 7.61 7.61 ...
##  $ WindGustDir  : chr  "W" "WNW" "WSW" "NE" ...
##  $ WindGustSpeed: num  44 44 46 24 41 56 50 35 80 28 ...
##  $ WindDir9am   : chr  "W" "NNW" "W" "SE" ...
##  $ WindDir3pm   : chr  "WNW" "WSW" "WSW" "E" ...
##  $ WindSpeed9am : num  20 4 19 11 7 19 20 6 7 15 ...
##  $ WindSpeed3pm : num  24 22 26 9 20 24 24 17 28 11 ...
##  $ Humidity9am  : num  71 44 38 45 82 55 49 48 42 58 ...
##  $ Humidity3pm  : num  22 25 30 16 33 23 19 19 9 27 ...
##  $ Pressure9am  : num  1008 1011 1008 1018 1011 ...
##  $ Pressure3pm  : num  1007 1008 1009 1013 1006 ...
##  $ Cloud9am     : num  8 4.45 4.45 4.45 7 ...
##  $ Cloud3pm     : num  4.51 4.51 2 4.51 8 ...
##  $ Temp9am      : num  16.9 17.2 21 18.1 17.8 20.6 18.1 16.3 18.3 20.1 ...
##  $ Temp3pm      : num  21.8 24.3 23.2 26.5 29.7 28.9 24.6 25.5 30.2 28.2 ...
##  $ RainToday    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ RainTomorrow : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ Day          : Ord.factor w/ 7 levels "Sun"<"Mon"<"Tue"<..: 2 3 4 5 6 7 1 2 3 4 ...
##  $ year         : num  2008 2008 2008 2008 2008 ...
##  $ mon          : num  12 12 12 12 12 12 12 12 12 12 ...
##  $ day          : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ State        : chr  "New South Wales" "New South Wales" "New South Wales" "New South Wales" ...
```

EDA

Production of Winter Crop State-wise

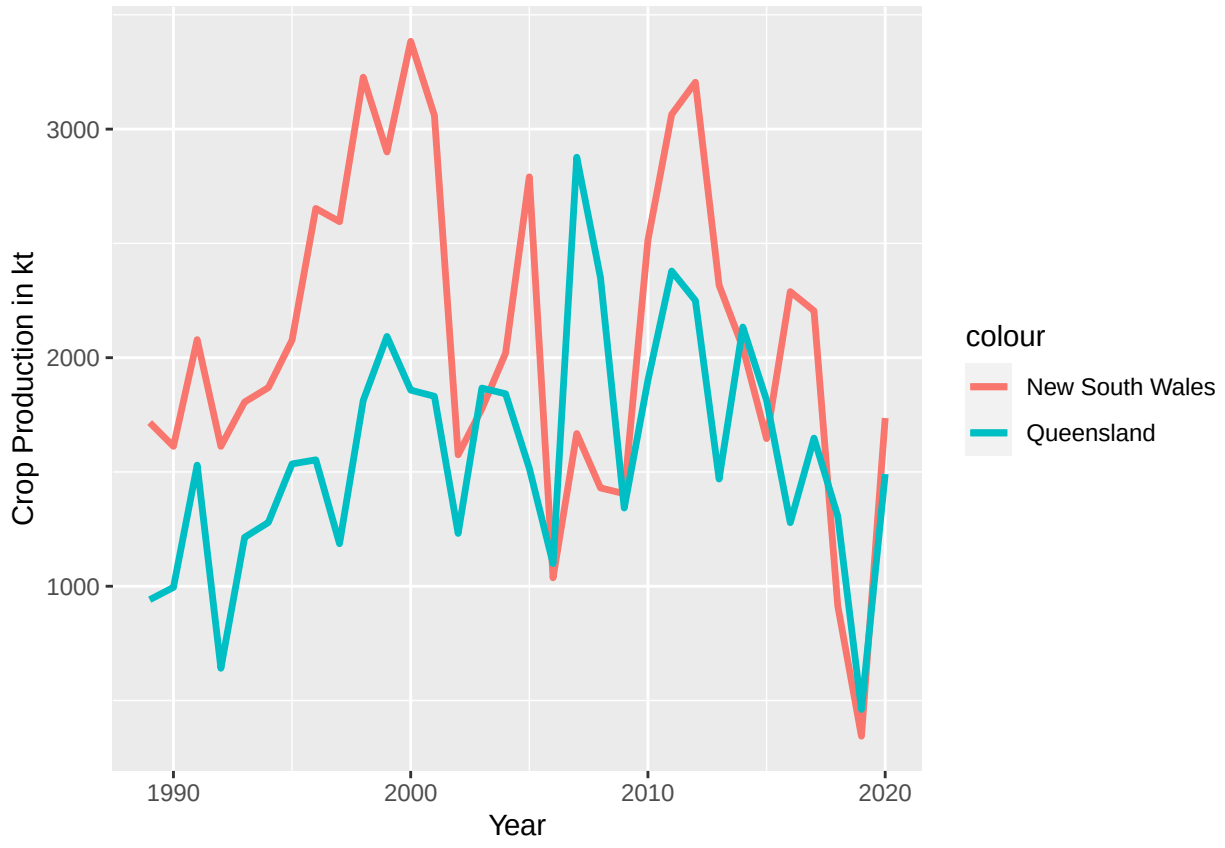
```
ggplot(wintercrop, aes(Year)) +
  geom_line(aes(y = Victoria, colour = "Victoria"),size=1.2) +
  geom_line(aes(y = Queensland, colour = "Queensland"), size=1.2)+
  geom_line(aes(y = New.South.Wales, colour = "New.South.Wales"), size=1.2)+
  geom_line(aes(y = South.Australia, colour = "South.Australia"), size=1.2)+
  geom_line(aes(y = Western.Australia, colour = "Western.Australia"), size=1.2)+
  ylab("Crop Production in kt")+
  xlab("Year")
```



The above line plot shows, how the production changes with each year. It can be observed that Western Autralia has the highest production every winter.

Production of Summer Crop State-wise

```
ggplot(summercrop, aes(Year)) +
  geom_line(aes(y = New.South.Wales.Production, colour = "New South Wales"),size=1.2) +
  geom_line(aes(y = Queensland.Production, colour = "Queensland"), size=1.2)+
  ylab("Crop Production in kt")+
  xlab("Year")
```



The above line plot we can observe that there are only two states which have production in Summer Season.

Visualisation of winter crop 2017 on Australian Map

```
aus <- world.cities[world.cities$country.etc == "Australia",]

aus=aus%>%
  mutate(State=case_when(
    name=="Albury" | name=="BadgerysCreek" | name=="Cobar" | name=="CoffsHarbour" | name=="Moree" | nam
    name=="Tuggeranong" | name=="MountGinini" | name=="Ballarat" | name=="Bendigo" | name=="Sale" | nam
    name=="Brisbane" | name=="Cairns" | name=="GoldCoast" | name=="Townsville" ~ "Queensland",
    name=="Adelaide" | name=="MountGambier" | name=="Nuriootpa" | name=="Woomera" ~ "South Australia",
    name=="Albany" | name=="Witchcliffe" | name=="PearceRAAF" | name=="PerthAirport" | name=="Perth" | 
    name=="Hobart" | name=="Hobart" | name=="Launceston" | name=="AliceSprings" | name=="Darwin" | name=
  ))
```

Data Cleaning

```
aus=aus%>%
  mutate(winter2017=case_when(
    State=="New South Wales"~7743,
    State=="Victoria"~7612,
    State=="Queensland"~1438,
    State=="South Australia"~7022,
    State=="Western Australia"~14510,
    State=="Northern Territory"~0
  ))

aus=aus%>%
  mutate(Rain2017=case_when(
    State=="New South Wales"~3.0582515,
    State=="Victoria"~1.4062729,
    State=="Queensland"~4.6504147,
    State=="South Australia"~1.1469991,
    State=="Western Australia"~1.7753573,
    State=="Northern Territory"~3.3233085
  ))
aus=na.omit(aus)
```

Assigning values

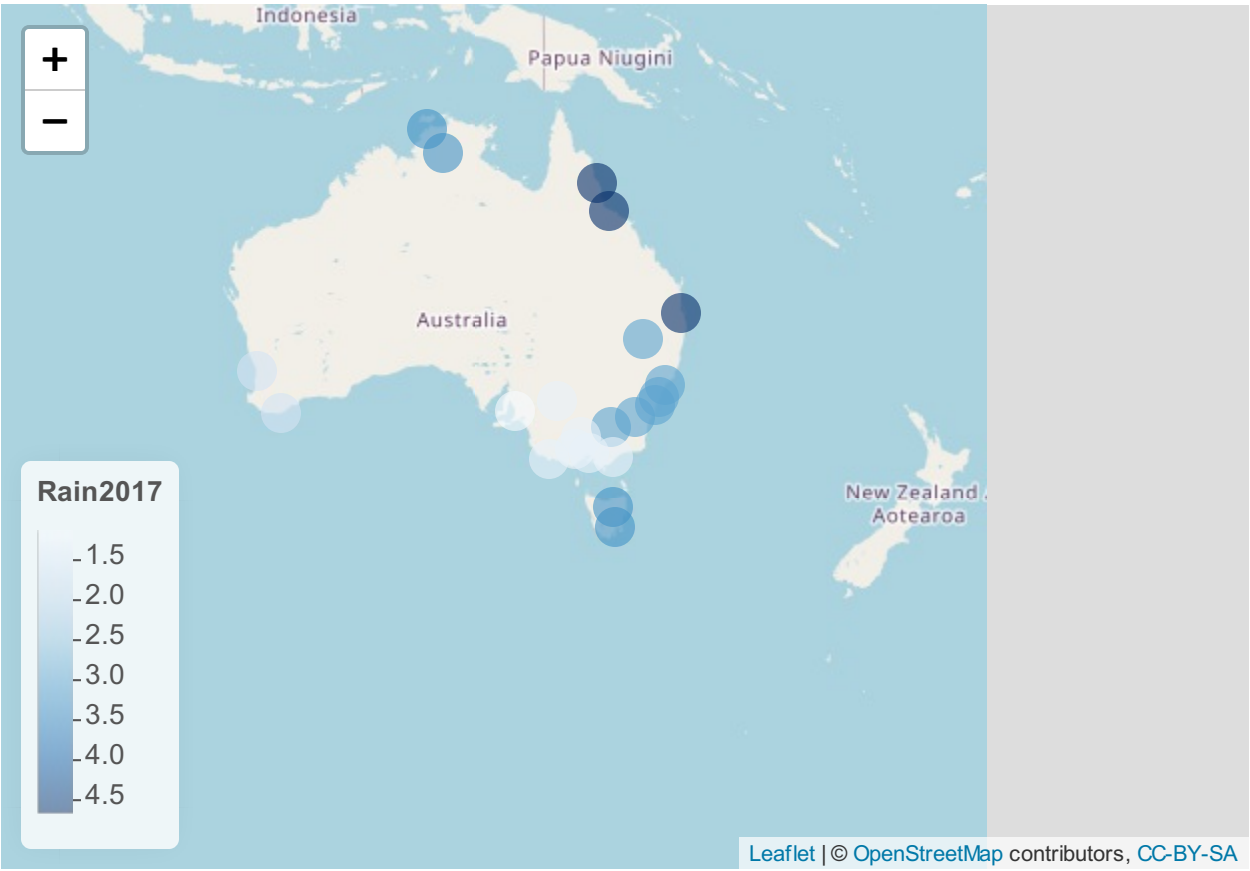
```
pal <- colorNumeric(palette = "Greens",
                    domain = aus$winter2017)

leaflet(data = aus) %>%
    addTiles() %>%
    addCircleMarkers(lat = ~lat, lng = ~long, popup = ~name,
                     color = ~pal(winter2017), stroke = FALSE, fillOpacity = 0.6) %>%
    addLegend(position = "bottomleft", pal = pal, values = ~winter2017)
```

Plotting agriculture data on map for winter season production

In the map we can observe, Eastern Aus has a large number of cities which are doing agriculture activities. But, the Western Aus has more production but less area for agriculture.

Visualising Rain on Map

```
pal <- colorNumeric(palette = "Blues",
                    domain = aus$Rain2017)
leaflet(data = aus) %>%
    addTiles() %>%
    addCircleMarkers(lat = ~lat, lng = ~long, popup = ~name,
                     color = ~pal(Rain2017), stroke = FALSE, fillOpacity = 0.6) %>%
    addLegend(position = "bottomleft", pal = pal, values = ~Rain2017)
```



In the above map we can observe that the western and the north-western region has more rainfall compared to other regions.

How Rainfall Effects winter crops

Selecting suitable rows

```
winterRain=agriculture_data%>%
  filter(mon==6 | mon==7 | mon==8 | mon==9 | mon==10)

DatabyState=agriculture_data%>%
  group_by(year,State)%>%
  summarise(avgRain=mean(Rainfall), avgLight=mean(Sunshine))
```

```
## `summarise()` regrouping output by 'year' (override with `.groups` argument)
```

```
DatabyState
```

```
## # A tibble: 61 x 4
## # Groups:   year [11]
##     year State              avgRain avgLight
##    <dbl> <chr>                <dbl>    <dbl>
```

```
##  1  2007 New South Wales       3.22     8.09
##  2  2008 New South Wales       2.22     7.61
##  3  2008 Northern Territory    2.00     8.00
##  4  2008 Queensland            4.01     8.26
##  5  2008 South Australia       1.84     7.50
##  6  2008 Victoria              1.91     6.92
##  7  2008 Western Australia     1.95     8.72
##  8  2009 New South Wales       2.22     7.76
##  9  2009 Northern Territory    2.35     8.15
## 10  2009 Queensland            4.48     8.13
## # ... with 51 more rows
```

```
## New South Wales

temp=DatabyState%>%
  filter(State=="New South Wales", year>2007)

temp2=wintercrop%>%
  filter(Year>2007 & Year<2018)

NSW_production=temp2$New.South.Wales
temp1=cbind(temp,production=NSW_production)

## Queensland

temp=DatabyState%>%
  filter(State=="Queensland")

temp2=wintercrop%>%
  filter(Year>2007 & Year<2018)

NSW_production=temp2$New.South.Wales
temp02=cbind(temp,production=NSW_production)

## South Australia

temp=DatabyState%>%
  filter(State=="South Australia")

temp2=wintercrop%>%
  filter(Year>2007 & Year<2018)

NSW_production=temp2$New.South.Wales
temp3=cbind(temp,production=NSW_production)

## Victoria

temp=DatabyState%>%
  filter(State=="Victoria")

temp2=wintercrop%>%
```

```
  filter(Year>2007 & Year<2018)

NSW_production=temp2$New.South.Wales
temp4=cbind(temp,production=NSW_production)

## Western Australia

temp=DatabyState%>%
  filter(State=="Western Australia")

temp2=wintercrop%>%
  filter(Year>2007 & Year<2018)

NSW_production=temp2$New.South.Wales
temp5=cbind(temp,production=NSW_production)
```

Making data frame for winter production

Converting data into time series

```
ts_winter1=ts(temp1[,c(-1,-2,-4)], start=c(2008),end=c(2017), frequency = 1)
ts_winter2=ts(temp02[,c(-1,-2,-4)], start=c(2008),end=c(2017), frequency = 1)
ts_winter3=ts(temp3[,c(-1,-2,-4)], start=c(2008),end=c(2017), frequency = 1)
ts_winter4=ts(temp4[,c(-1,-2,-4)], start=c(2008),end=c(2017), frequency = 1)
ts_winter5=ts(temp5[,c(-1,-2,-4)], start=c(2008),end=c(2017), frequency = 1)
```

Building Model(using VAR model)

```
model1=vars::VAR(ts_winter1, type = "none", lag.max = 5, ic = "AIC")
model2=vars::VAR(ts_winter2, type = "none", lag.max = 5, ic = "AIC")
model3=vars::VAR(ts_winter3, type = "none", lag.max = 5, ic = "AIC")
model4=vars::VAR(ts_winter4, type = "none", lag.max = 5, ic = "AIC")
model5=vars::VAR(ts_winter5, type = "none", lag.max = 5, ic = "AIC")
```

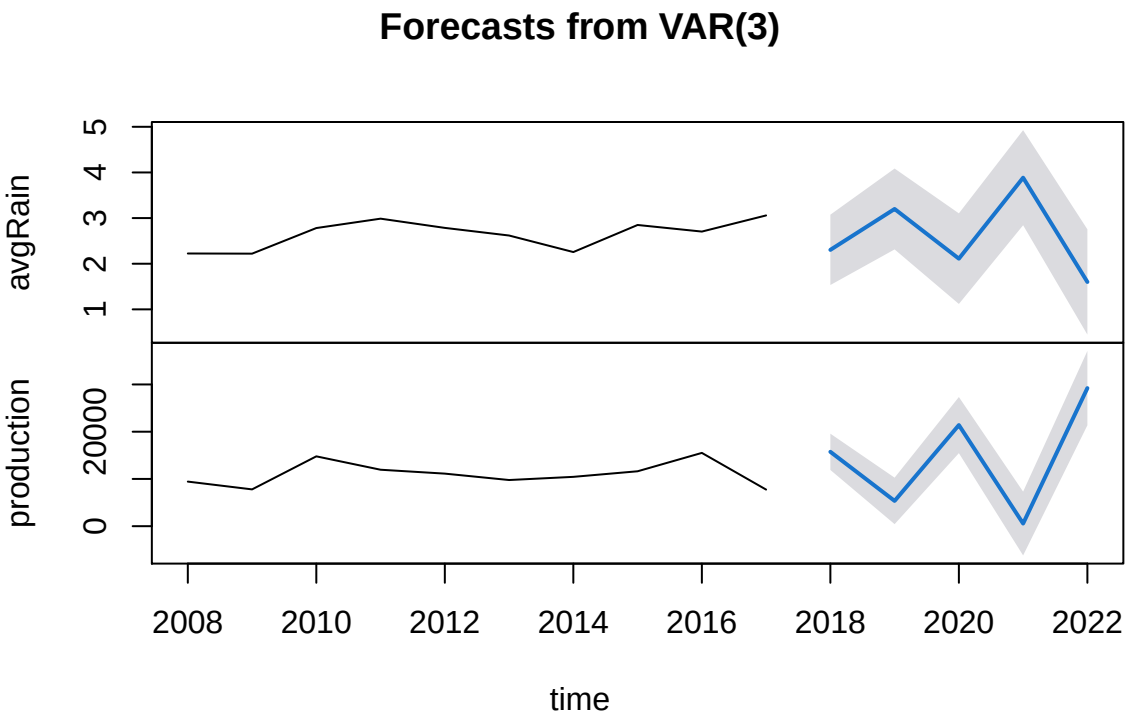
```
## Warning in log(sigma.det): NaNs produced

## Warning in log(sigma.det): NaNs produced

## Warning in log(sigma.det): NaNs produced
```

Forecasting production

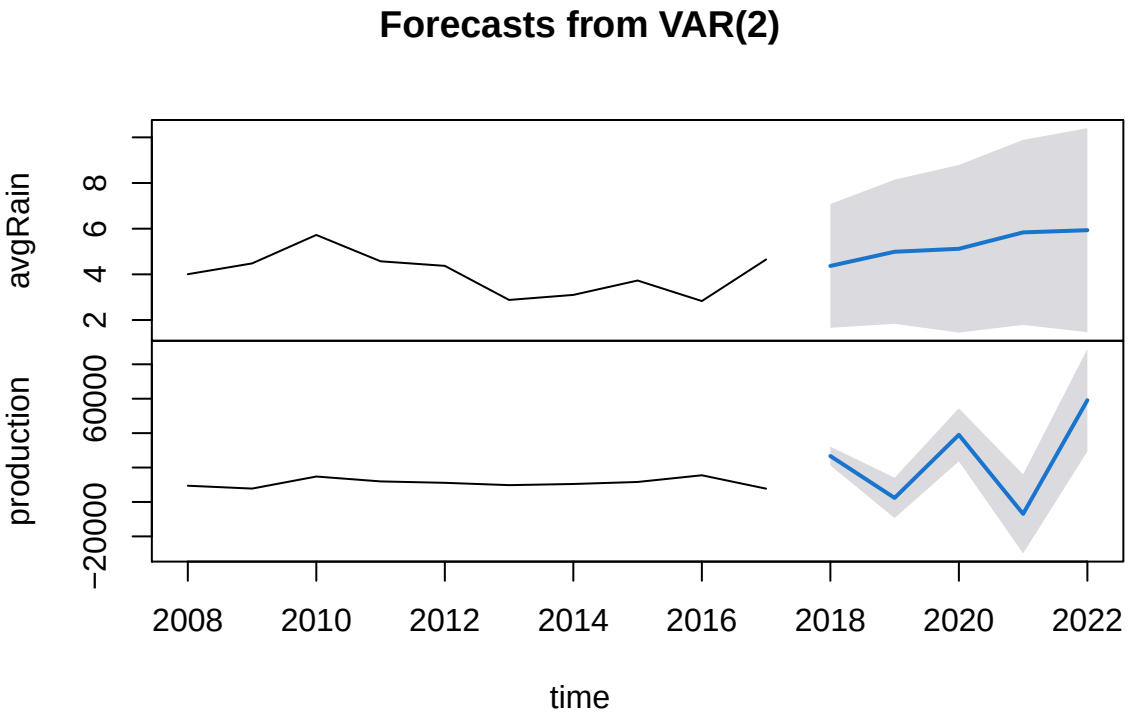
```
fc1 <- forecast(model1,level=c(95),h=1*5)
fc2 <- forecast(model2,level=c(95),h=1*5)
fc3 <- forecast(model3,level=c(95),h=1*5)
fc4 <- forecast(model4,level=c(95),h=1*5)
fc5 <- forecast(model5,level=c(95),h=1*5)
```

Visualising the forecasted values

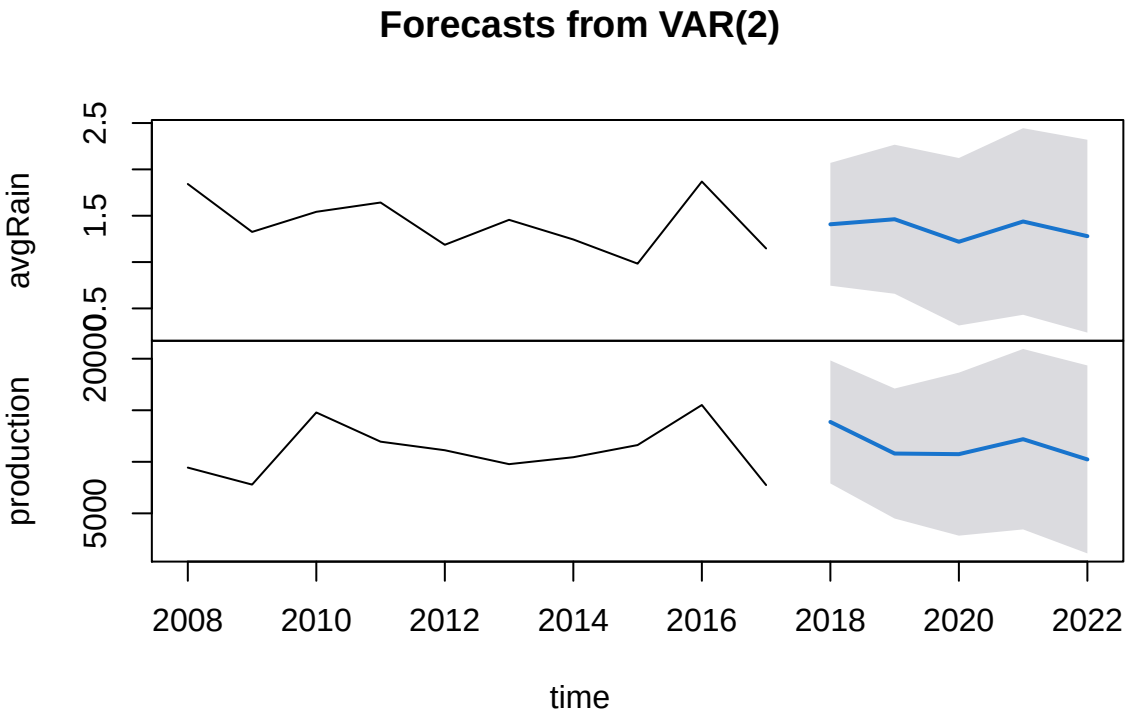
```
plot(fc1)
```



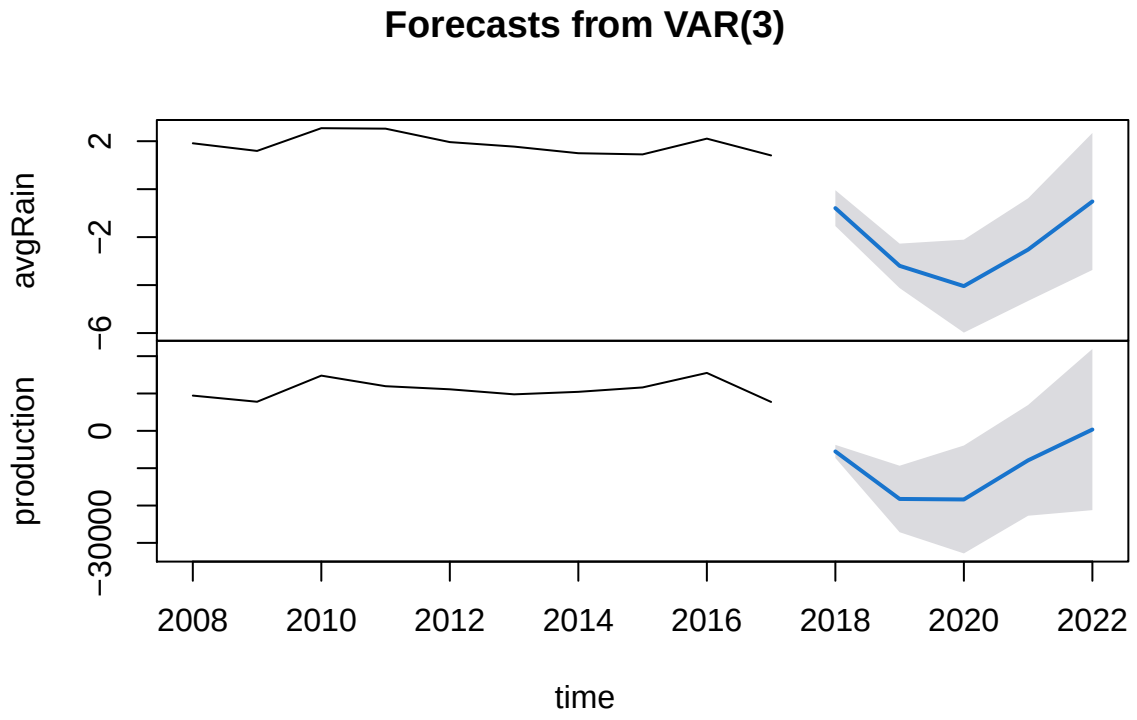

```
plot(fc2)
```


Forecasts from VAR(2)
avgRain
2
4
6
8
production
−20000
60000
2008
2010
2012
2014
2016
2018
2020
2022
time

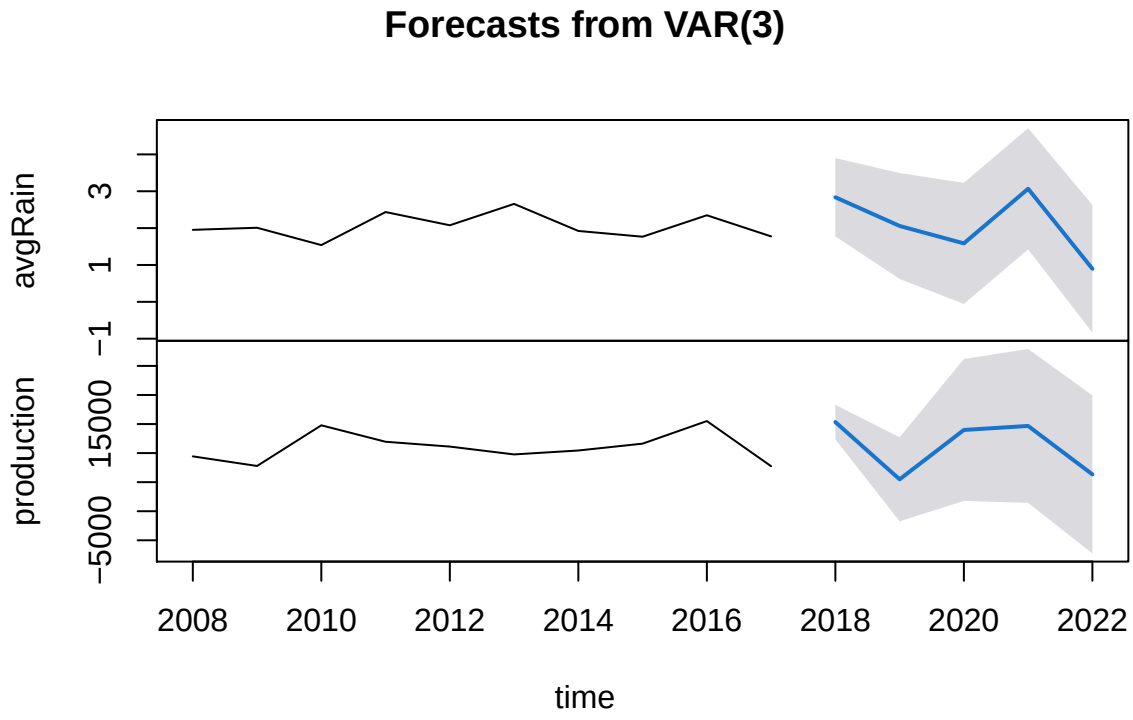
```
plot(fc3)
```


Forecasts from VAR(2)
avgRain
production
2.5
1.5
20000.5
5000
2008
2010
2012
2014
2016
2018
2020
2022
time

```
plot(fc4)
```


Forecasts from VAR(3)
avgRain
production
2
−2
−6
0
−30000
2008
2010
2012
2014
2016
2018
2020
2022
time

```
plot(fc5)
```



From the above 5 plots, we can conclude that the overall production will increase, but in some regions the production will decrease in the upcoming years.

How Rainfall Effects summer rows

```
winterRain=agriculture_data%>%
  filter(mon==12 | mon==1 | mon==2 | mon==3 | mon==4)

DatabyState=agriculture_data%>%
  group_by(year,State)%>%
  summarise(avgRain=mean(Rainfall), avgLight=mean(Sunshine))
```

```
## `summarise()` regrouping output by 'year' (override with `.groups` argument)
```

```
DatabyState
```

```
## # A tibble: 61 x 4
## # Groups:   year [11]
##     year State              avgRain avgLight
##    <dbl> <chr>                <dbl>    <dbl>
##  1  2007 New South Wales       3.22     8.09
##  2  2008 New South Wales       2.22     7.61
```

```
##  3  2008 Northern Territory    2.00     8.00
##  4  2008 Queensland            4.01     8.26
##  5  2008 South Australia       1.84     7.50
##  6  2008 Victoria              1.91     6.92
##  7  2008 Western Australia     1.95     8.72
##  8  2009 New South Wales       2.22     7.76
##  9  2009 Northern Territory    2.35     8.15
## 10  2009 Queensland            4.48     8.13
## # ... with 51 more rows
```

```
## New South Wales

temp=DatabyState%>%
  filter(State=="New South Wales", year>2007)

temp2=summercrop%>%
  filter(Year>2007 & Year<2018)

NSW_production=temp2$New.South.Wales.Production
temp1=cbind(temp,production=NSW_production)

## Queensland

temp=DatabyState%>%
  filter(State=="Queensland")

temp2=summercrop%>%
  filter(Year>2007 & Year<2018)

NSW_production=temp2$Queensland.Production
temp02=cbind(temp,production=NSW_production)
```

Making data frame for winter production

Converting data into time series

```
ts_summer1=ts(temp1[,c(-1,-2,-4)], start=c(2008),end=c(2017), frequency = 1)
ts_summer2=ts(temp02[,c(-1,-2,-4)], start=c(2008),end=c(2017), frequency = 1)
```

Building Model(using VAR model)

```
model1=vars::VAR(ts_summer1, type = "none", lag.max = 5, ic = "AIC")
model2=vars::VAR(ts_summer2, type = "none", lag.max = 5, ic = "AIC")
```

```
## Warning in log(sigma.det): NaNs produced
```

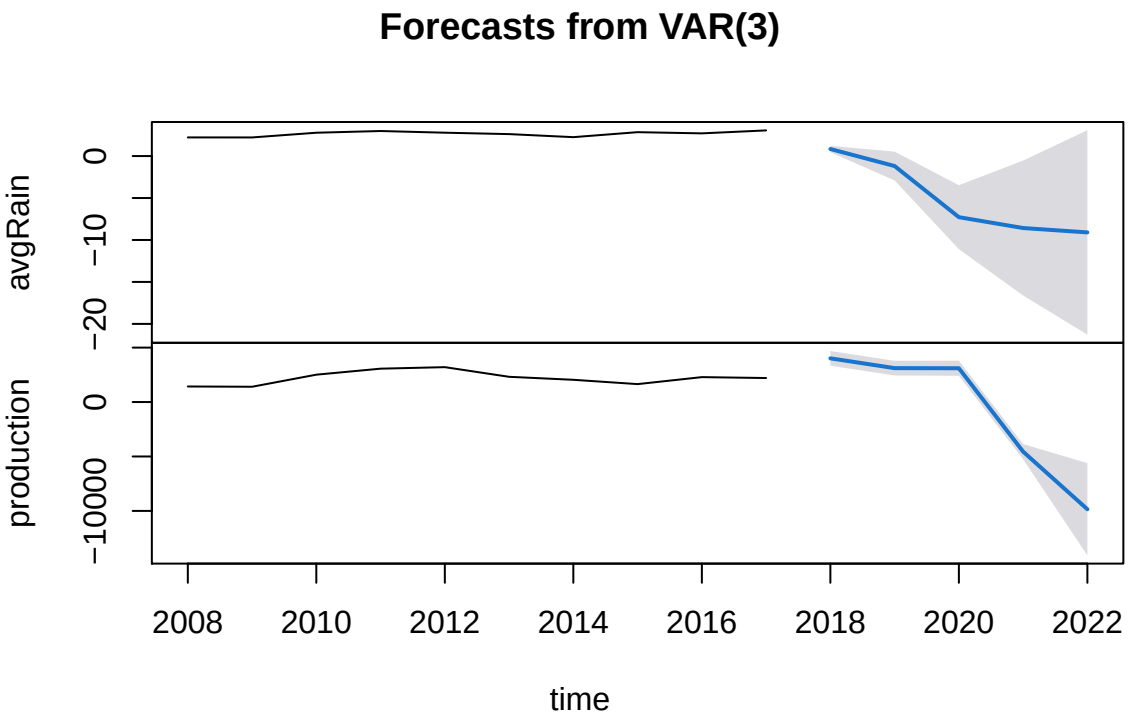
```
## Warning in log(sigma.det): NaNs produced

## Warning in log(sigma.det): NaNs produced
```

Forecasting production

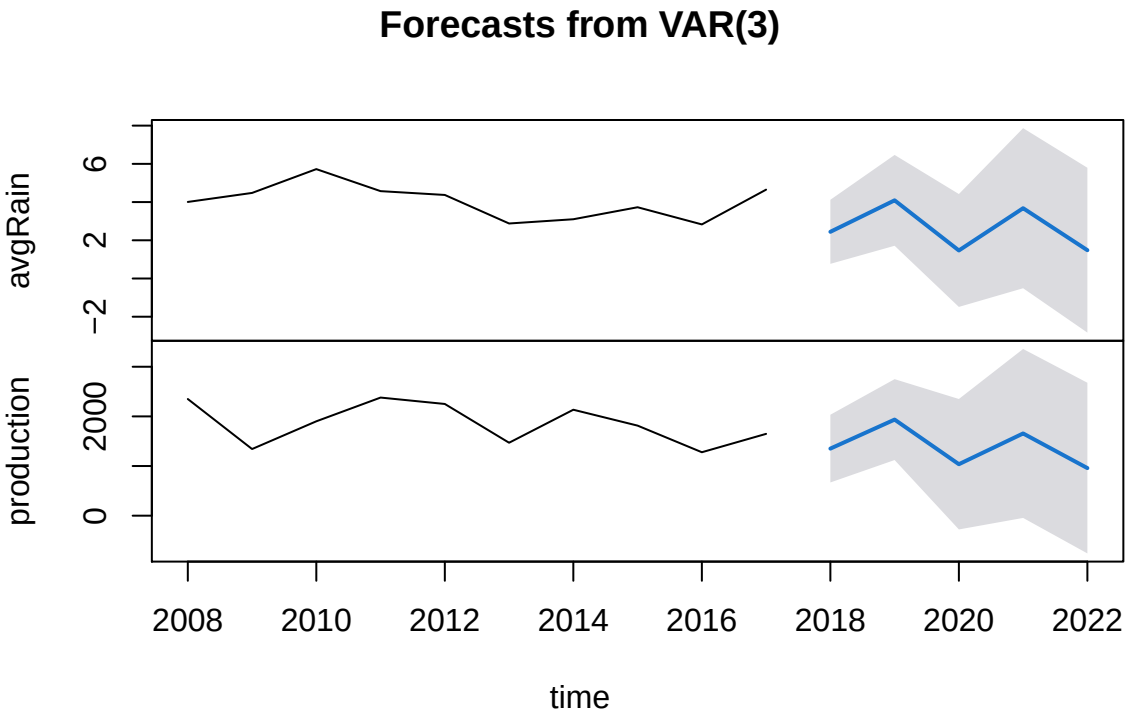
```
fc1 <- forecast(model1,level=c(95),h=1*5)
fc2 <- forecast(model2,level=c(95),h=1*5)
```

Visualising the forecasted values

```
plot(fc1)
```




```
plot(fc2)
```



From the above 2 plots, we can conclude that the overall production will decrease in the upcoming years for summer crops.

END