Time Series Australia Beer Production:

Dataset: This data shows the monthly production of beer in Australia from January of 1956 to December 1978. The data shown is scaled in megaliters(1 Megaliter = 1million liters).

Problem Statement: Rename the following three columns in this fashion:

Old name	New name
X	index
Month	date
Monthly.beer.production	production

Create two new columns in the data frame accounting for the “year” and the “month” of the production, respectively.

Build a Time Series Model in R and answer the following questions.

Analysis

1. Show a line plot of the data using the index as ‘x’ and production as "y" variable.
2. Using all the rows parameterize a base time series simple regression model using "index" as the independent variable and production as dependent variable. State the slope of the regression line and the correlation coefficient between actual and predicted values for production.
3. Show a plot of the time series data with the simple regression line layered on the graph in a contrasting color.
4. Execute and interpret a Durbin-Watson test on the model results.
5. Original data appears to have a pronounced cyclical pattern. Assuming the complete cycles are 12 months long, construct a set of seasonal indices which describe the typical annual fluctuations in production. Use these indices to deseasonalize the production data.
6. Using the deseasonalized to data parameterize two different regression models. A simple regression model will be the base case and a second order polynomial (x^2) model which attempts to describe the non-linear secular fluctuations in the deseasonalized data.
7. Reseasonalize the fitted values for each of the two models. Drawing on analysis above, construct a plot showing the original data and the fitted values for each of the two regression models. From a visual review, which model appears to have the better fit to the original beer production data?

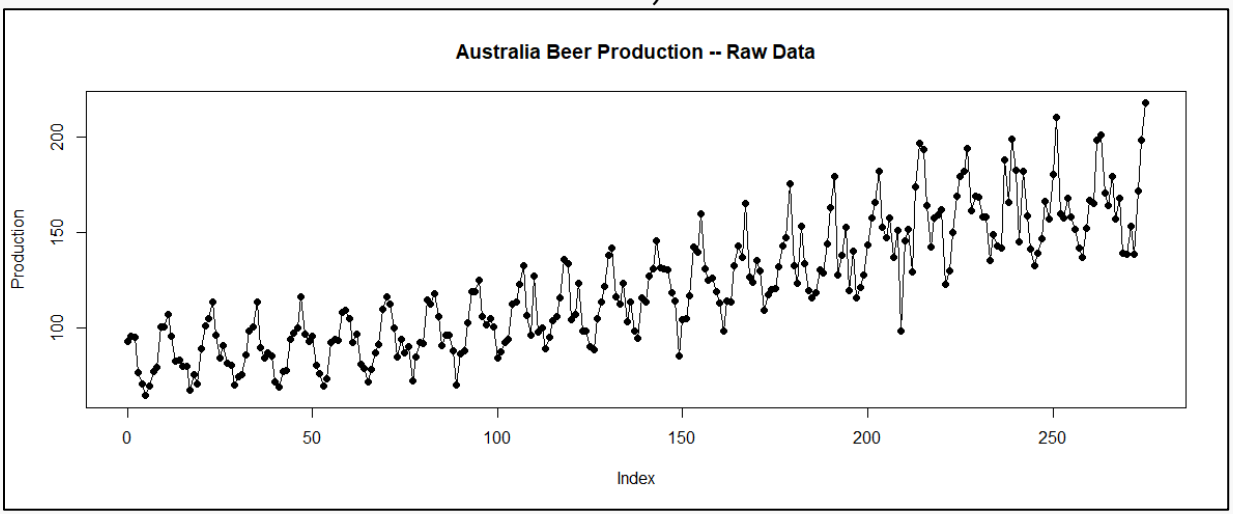
I. Preprocessing

```
#Author: Suryateja Chalapati

#Importing required libraries
rm(list=ls())
library(rio)
library(moments)
library(car)
library(dplyr)

#Setting the working directory, importing and creating new variables
setwd("C:/Users/surya/Downloads")

df = import("Time Series Data.xlsx", sheet = "Sheet1")
colnames(df)=tolower(make.names(colnames(df)))
df = df %>% rename(index = x, date = month, production = monthly.beer.production)
df$year=as.numeric(format(df$date,'%Y'))
df$month=as.numeric(format(df$date,'%m'))
attach(df)
```

II. Analysis

```
#Analysis_1
```

```
plot(index,production,type="o",pch=19,ylab = "Production",xlab = "Index",
     main="Australia Beer Production -- Raw Data")
```



- From the plot above, we can clearly see a cyclical pattern in the data. It follows a secular and seasonal trend. This type data is perfect for time series regression.

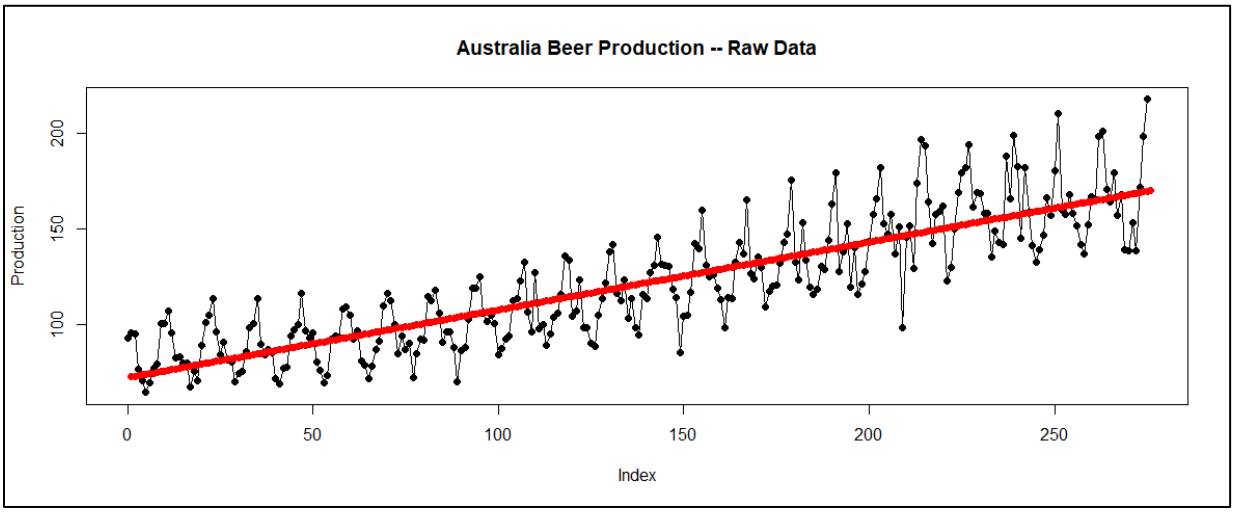
```
#Analysis_2
#Simple regression on Time Series data
regout=lm(production~index,data=df)
summary(regout)
##
## Call:
## lm(formula = production ~ index, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.175 -12.695  -1.649  11.210  48.452
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  72.8636     2.0810   35.01   <2e-16 ***
## index         0.3541     0.0131   27.04   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.33 on 274 degrees of freedom
## Multiple R-squared:  0.7274, Adjusted R-squared:  0.7264
## F-statistic: 731.3 on 1 and 274 DF,  p-value: < 2.2e-16
```

From the Simple Regression analysis:

- The estimated B – Coefficients for the independent variable is 0.35. The Intercept is 72.86.
- The p-value is < 0.05 for both Intercept and the slope. This states we can reject the null hypothesis and accept the alternate, so we can say the B – Coefficients are not zero and are significant.
- The Regression Equation is **y [Production] = 72.863 + 0.354*index.**
- The R-sq is 0.7274, that means the X variable explains about 72.74% of variation in Y. But the model is not seeing the cyclical pattern in the data. So, the regression line we see below is the "Best Fit" that R has come up with.
- We can say there is a direct proportionality between [index, Production].
- If index is zero then Production increases by **72.86 Megaliter**, which is something we cannot interpret.

- The correlation coefficient is **0.85** which aligns with the data we see in the plot. As the data points are not too spread apart.

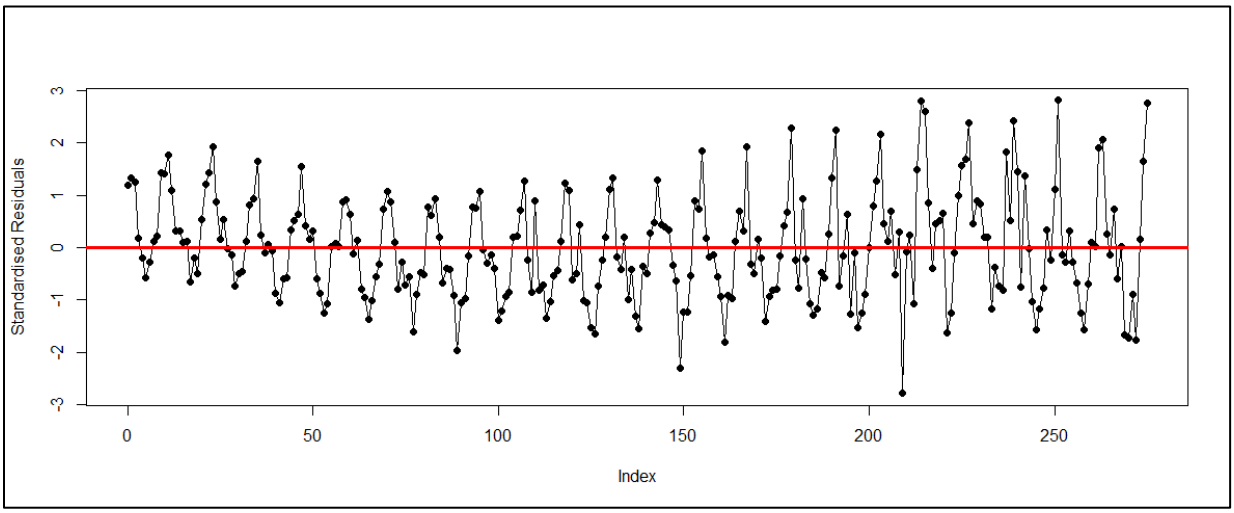
```
points(regout$fitted.values,type="o",pch=19,col="red")
```



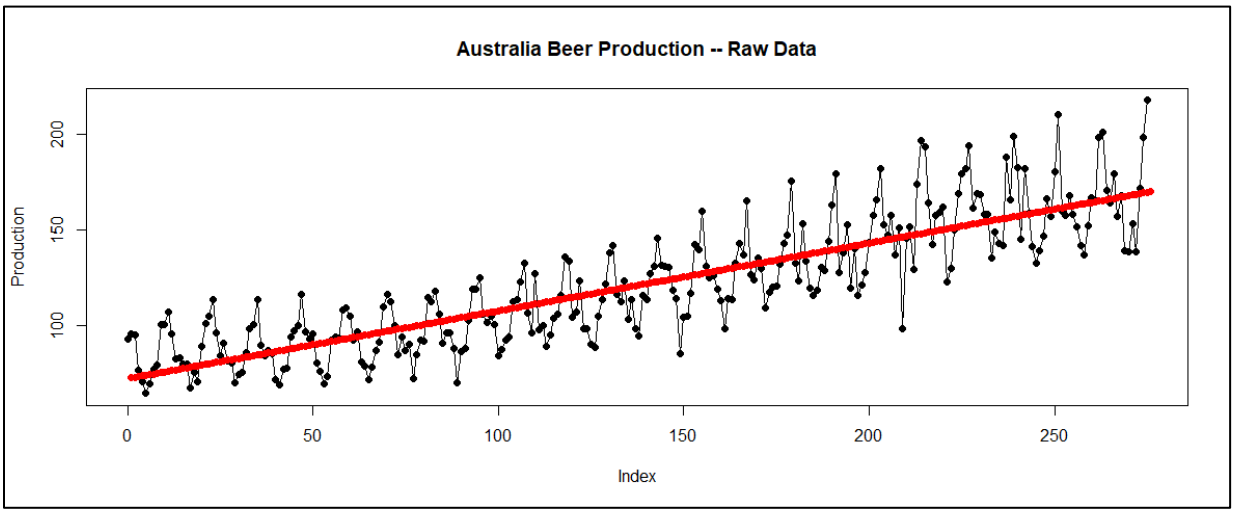

```
cor(df$production,regout$fitted.values)
```

```
## [1] 0.8528959
```

```
plot(df$index,rstandard(regout),ylab = "Standardised Residuals",xlab = "Index",pch=19,type="o"
)
abline(0,0,col="red",lwd=3)
```




```
#Analysis_3
plot(index,production,type="o",pch=19,ylab = "Production",xlab = "Index",
     main="Australia Beer Production -- Raw Data")
points(regout$fitted.values,type="o",pch=19,col="red")
```



- The plot shows the "Best Fit" of the regression line.

```
#Analysis_4
#Durbin Watson Test
dwt.out=durbinWatsonTest(regout)
dwt.out

##  lag Autocorrelation D-W Statistic p-value
##    1       0.5350599     0.8973878       0
##  Alternative hypothesis: rho != 0
```

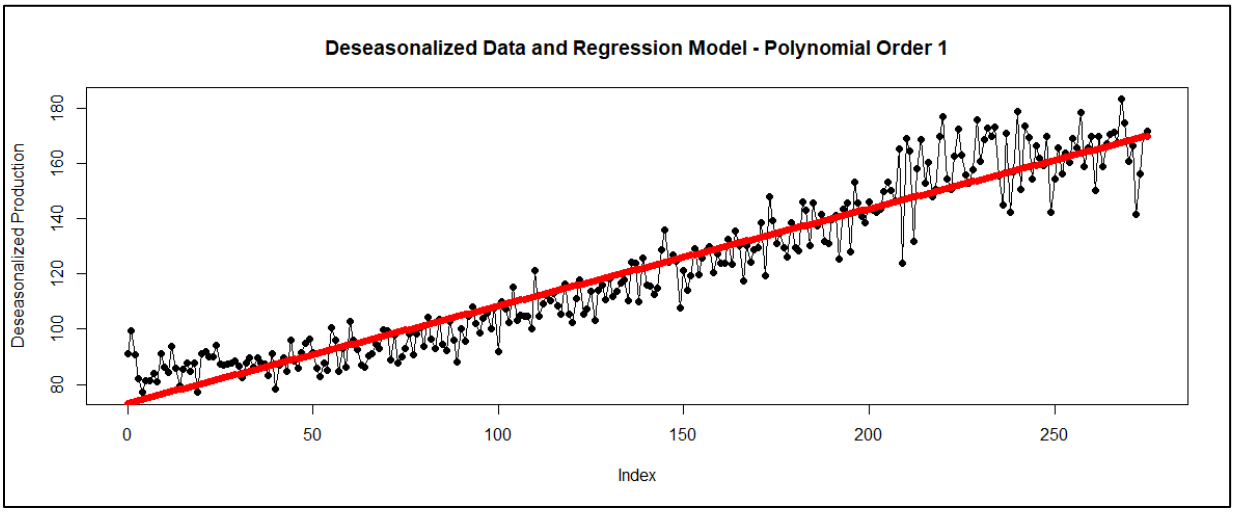
- From the Durbin-Watson test above, we can say that there is positive autocorrelation **[0.897, the preferred range is 1.5 to 2.5].** We need to do a little detective work going forward.
- P-value is 0, so rejecting the null and accepting the alternate which says there is definitely an autocorrelation.

```
#Analysis_5
#Making Seasonal Indices
indices=data.frame(month=1:12,average=0,index=0)
for(i in 1:12) {
  count=0
  for(j in 1:nrow(df)) {
    if(i==df$month[j]) {
      indices$average[i]=indices$average[i]+df$production[j]
      count=count+1
    }
  }
  indices$average[i]=indices$average[i]/count
  indices$index[i]=indices$average[i]/mean(df$production)}

#Deseasonalizing the original data
for(i in 1:12){
  for(j in 1:nrow(df)){
    if(i==df$month[j]){
      df$deseason.production[j]=df$production[j]/indices$index[i]
    }
  }
}
```

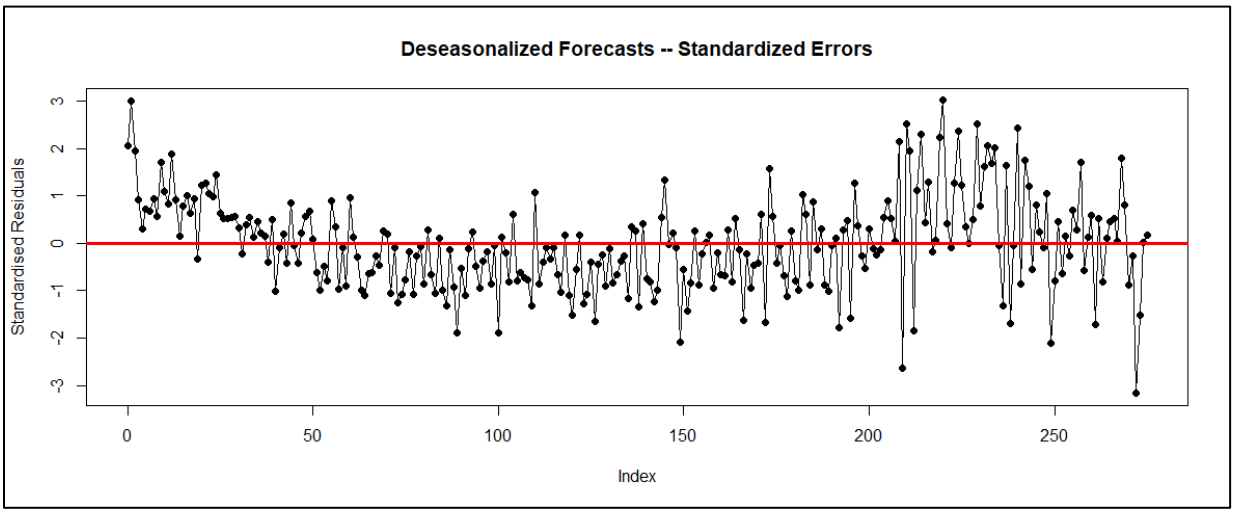
- Deseasonalizing the data and storing in the original object.

```
#Analysis_6
#Conducting the deseasonalized regression polynomial order 1
```

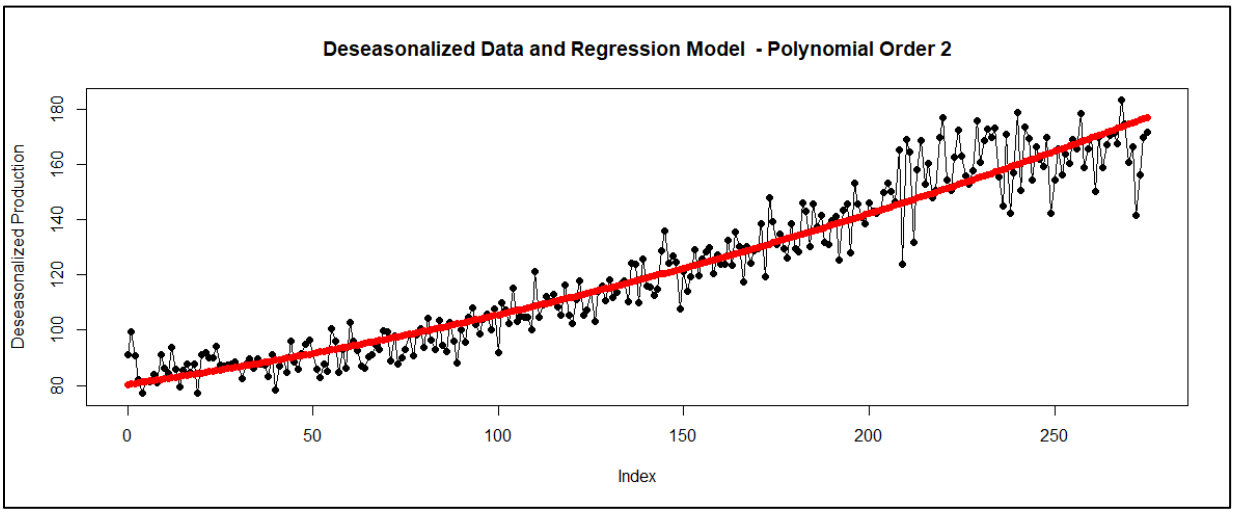
```r
dsreg.out.ord1=lm(deseason.production~index,data=df)
summary(dsreg.out.ord1)

##
## Call:
## lm(formula = deseason.production ~ index, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -27.434  -6.525  -0.431   4.735  26.192
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 73.214609   1.047160   69.92   <2e-16 ***
## index        0.351567   0.006589   53.35   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.722 on 274 degrees of freedom
## Multiple R-squared:  0.9122, Adjusted R-squared:  0.9119
## F-statistic:  2847 on 1 and 274 DF,  p-value: < 2.2e-16

plot(df$index,df$deseason.production,type="o",pch=19,ylab = "Deseasonalized Production",xlab =
"Index",
     main="Deseasonalized Data and Regression Model - Polynomial Order 1")
points(df$index,dsreg.out.ord1$fitted.values,type="o",
       pch=19,col="red")
```




```r
plot(df$index,rstandard(dsreg.out.ord1),pch=19,type="o",ylab = "Standardised Residuals",xlab =
"Index",
     main="Deseasonalized Forecasts -- Standardized Errors")
abline(0,0,col="red",lwd=3)
```

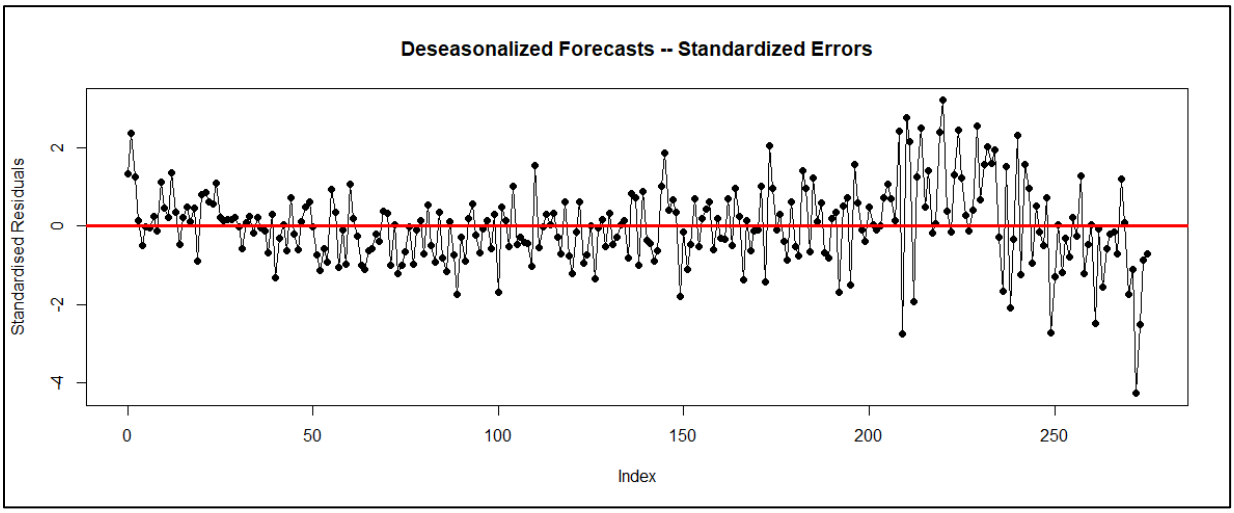



```
#Conducting the deseasonalized regression polynomial order 2
dsreg.out.ord2=lm(deseason.production~poly(index,2),data=df)
summary(dsreg.out.ord2)

##
## Call:
## lm(formula = deseason.production ~ poly(index, 2), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.104  -4.882  -0.064   4.102  25.932
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     121.5551     0.4884 248.883  < 2e-16 ***
## poly(index, 2)1 465.3484     8.1139  57.352  < 2e-16 ***
## poly(index, 2)2  53.5808     8.1139   6.604 2.09e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.114 on 273 degrees of freedom
## Multiple R-squared:  0.9243, Adjusted R-squared:  0.9237
## F-statistic:  1666 on 2 and 273 DF,  p-value: < 2.2e-16

plot(df$index,df$deseason.production,type="o",pch=19,ylab = "Deseasonalized Production",xlab =
"Index",
     main="Deseasonalized Data and Regression Model  - Polynomial Order 2")
points(df$index,dsreg.out.ord2$fitted.values,type="o",
       pch=19,col="red")
```

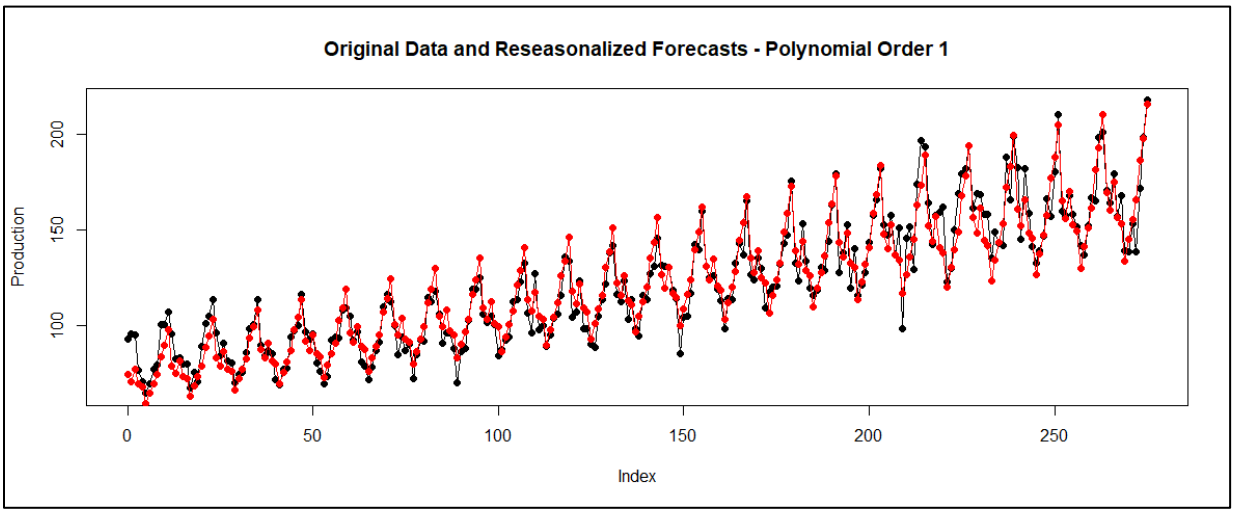



```
plot(df$index,rstandard(dsreg.out.ord2),pch=19,type="o",ylab = "Standardised Residuals",xlab =
"Index",
     main="Deseasonalized Forecasts -- Standardized Errors")
abline(0,0,col="red",lwd=3)
```

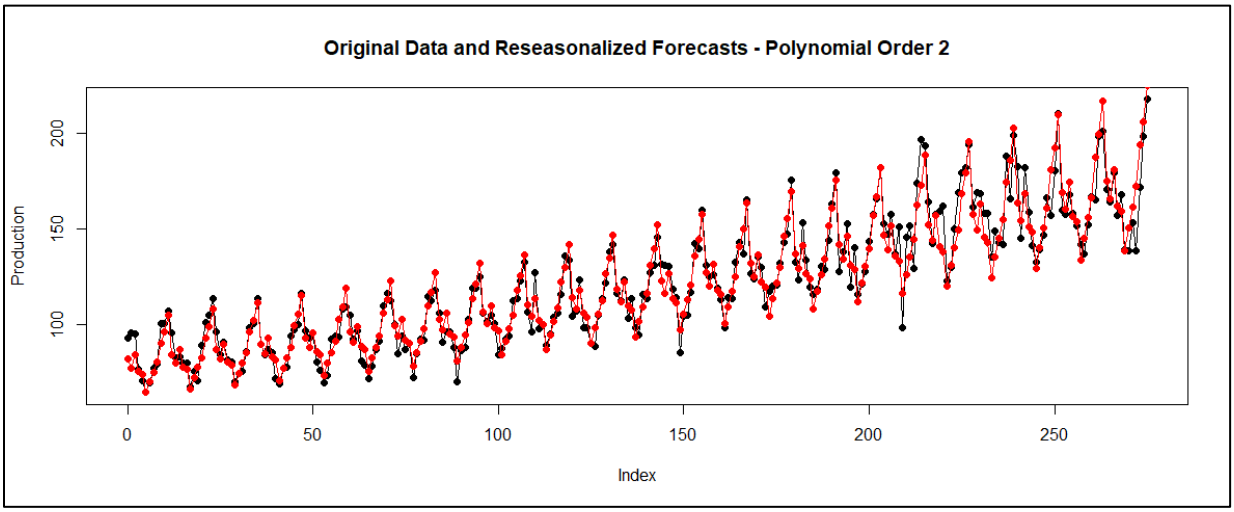
- Adding a second order polynomial term increases the R-sq from 91.2% to 92.4%.
- In both cases the p-values are below 0.05 so we can reject the null and accept the alternate and say the B-coefficient values are significant.




```
#Analysis_7
#Reseasonalizing Forecasts for polynomial order 1
df$deseason.forecast=dsreg.out.ord1$fitted.values
for(i in 1:12){
  for(j in 1:nrow(df)){
    if(i==df$month[j]){
      df$reseason.forecast[j]=df$deseason.forecast[j]*
        indices$index[i]
    }
  }
}

plot(df$index,df$production,type="o",pch=19,ylab = "Production",xlab = "Index",
     main="Original Data and Reseasonalized Forecasts - Polynomial Order 1")
points(df$index,df$reseason.forecast,
       type="o",pch=19,col="red")
```




```r
#Reseasonalizing Forecasts for polynomial order 2
df$deseason.forecast=dsreg.out.ord2$fitted.values
for(i in 1:12){
  for(j in 1:nrow(df)){
    if(i==df$month[j]){
      df$reseason.forecast[j]=df$deseason.forecast[j]*
        indices$index[i]
    }
  }
}

plot(df$index,df$production,type="o",pch=19,ylab = "Production",xlab = "Index",
     main="Original Data and Reseasonalized Forecasts - Polynomial Order 2")
points(df$index,df$reseason.forecast,
       type="o",pch=19,col="red")
```



- From the plots above, both seems to be very similar in terms of the fit to the actuals. But If I had to pick one I'll pick the second model as the R-sq is higher and if we look very closely, the second aligns with the actuals a bit more at the points near the end.