

**BOĞAZİÇİ UNIVERSITY**

**SPRING 2019 TERM**

**IE 360**

**COURSE PROJECT**

**GROUP MEMBERS:**

**SİNAN DEMİRHAN 2016402330**

**OĞULCAN ECE 2015402006**

**BERKAY ZÜHRE 2015402015**

**MUSTAFA KESER 2016402306**

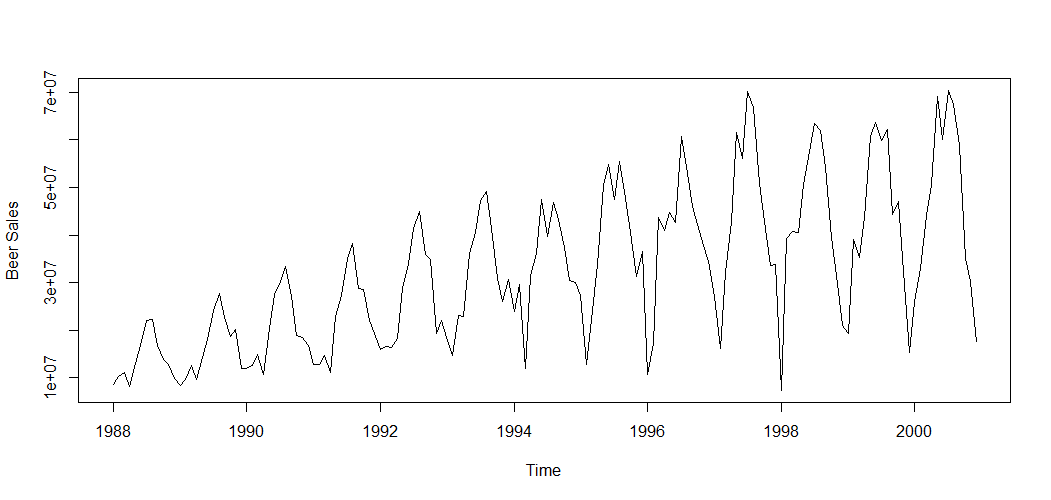
**INSTRUCTOR: REFİK GÜLLÜ**

**TEACHING ASSISTANT:ZEYNEP ŞUVAK**

1. **Introduction**

**Time Series Plot**

Time series plot is constructed by assigning time intervals to each data point. In this way, all data points can be combined with a line. In time series plots, data analysis is done properly. Trend and seasonality can be recognized if data have any.

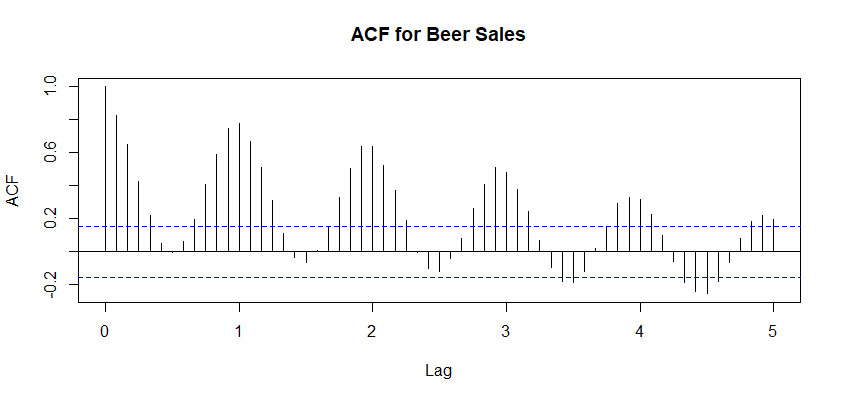


From the time series plot of beer sales, it can be easily seen that the data has an increasing trend and seasonality. Data is not stationary, because their variance are not constant. Also their means have increasing trend accordingly.

**Autocorrelation Plot**

Autocorrelation plot represents the correlation between data points. Let Yt shows the data point at time t. Then, “Autocorrelation at Lag k” term indicates the correlation among Yt and Yt+k for all t.

Trend and seasonality analysis can be made by interpreting the autocorrelation plot. For example, if data have a trend, high autocorrelation is seen at first lags. and then the correlation starts decreasing slowly. About seasonality, one can say that there will be some sudden spikes at lag values in the seasonality circle.



From the autocorrelation plots of the beer sales data,it is obvious that data have a trend because acf decreases and increases slowly. Also, seasonality can be observed by spikes at all lags.

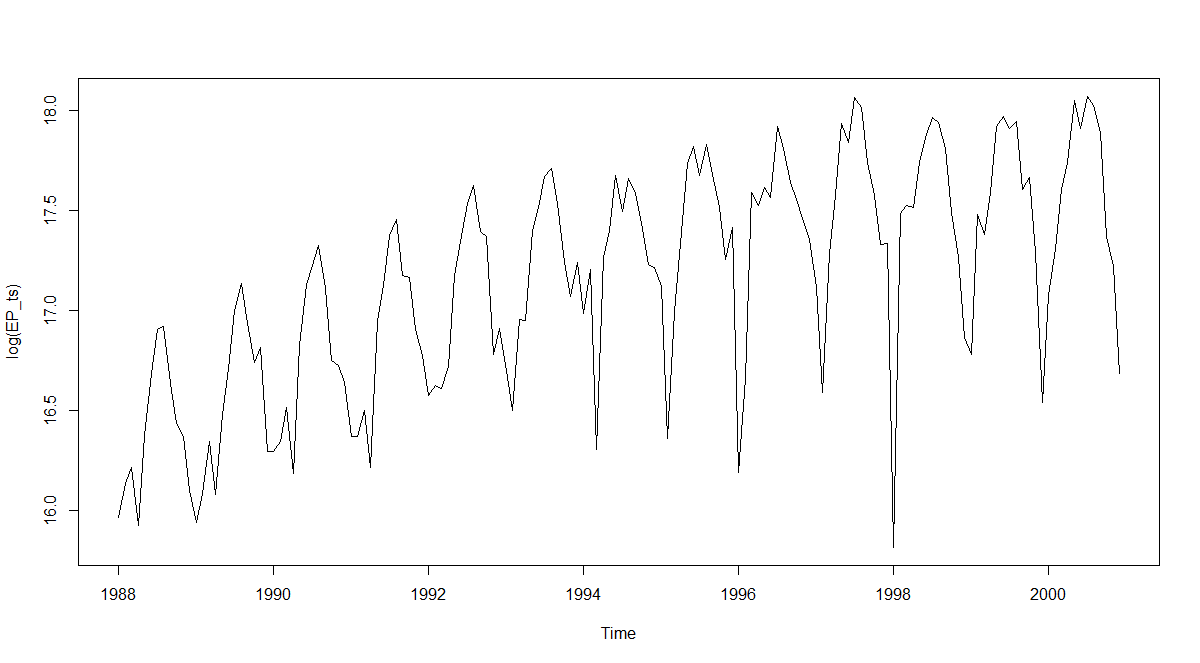
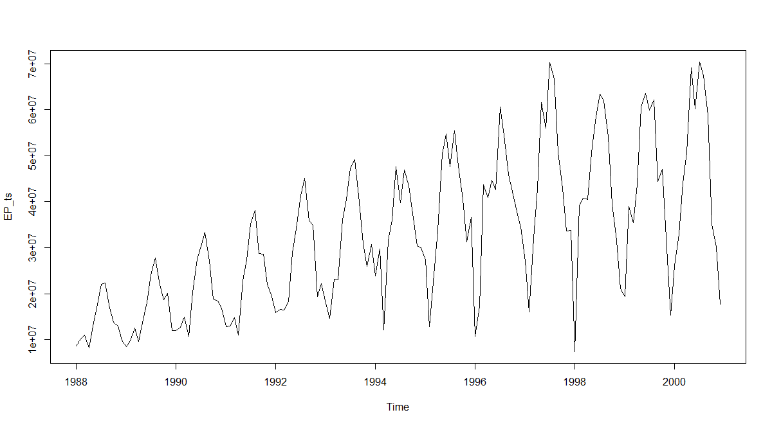
***METHOD A: FORECASTING WITH REGRESSION***

**1. Decide if you need a preliminary transformation to induce stationarity**

Our data isn’t stationary There is an upward trend, unstable variance and seasonality with a period of 12. To achieve constant mean and variance several methods can be applied. We choose differencing to have constant mean and logarithmic transformation to have constant variance. Logarithmic transformation will also help us have a more linear data.

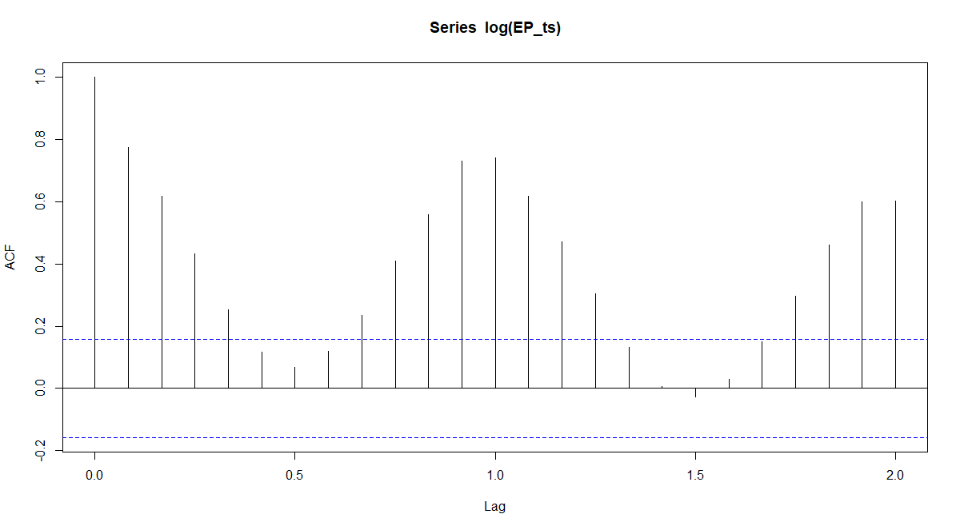
It should be noted that the apply order of differencing and log transformation is very important. Differencing can generate negative values but log transformation isn’t defined for negative numbers. So, when we decide to use these two methods together log transformation should always be applied first. It is also possible to use the absolute value of the differencing if the data allows it to do it.

plot(log(EP\_ts)) plot(EP\_ts)



Variance of the data is much more balanced compared to the previous plot. EP\_ts data looks linear but the variance in the data seems to increase as time passes. There might be outliers that cause some of this high variance image. log(EP\_ts) plot looks more balanced because we inhibited the effects of the high value changes by taking logarithm.

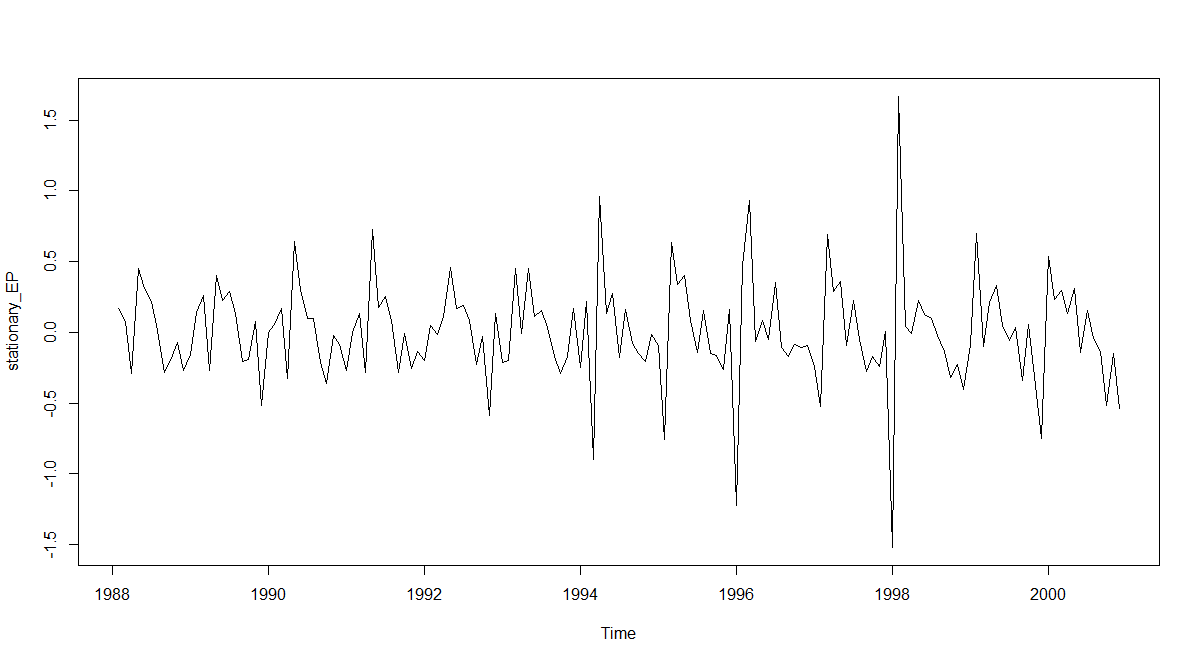
However, an upward trend can still be observed, so the mean can’t be constant yet.

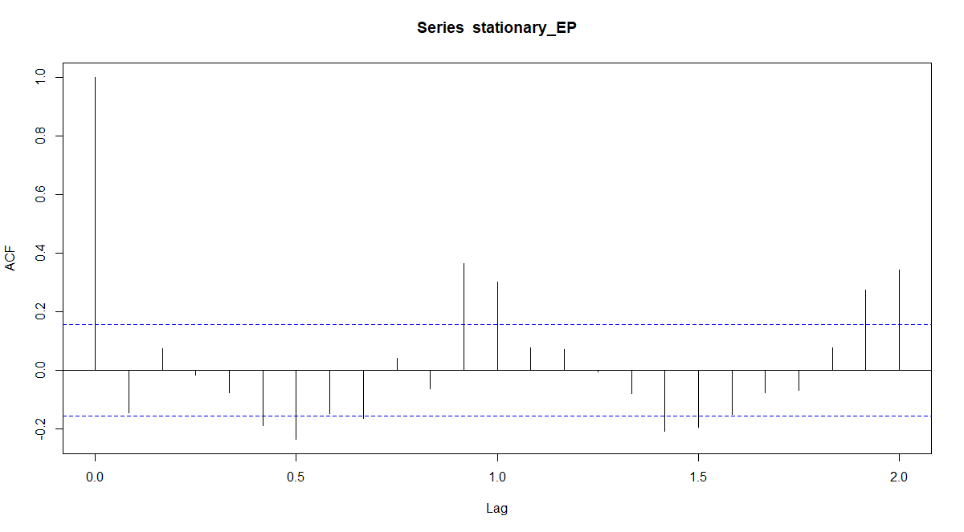
acf(log(EP\_ts),lag.max = 24)

We can still see the effects of trend and seasonality from the ACF plot.

stationary\_EP<-diff(log(EP\_ts))

plot(stationary\_EP)

acf(stationary\_EP,lag.max = 24)



After differencing we obtain a constant mean and our data becomes stationary.

**2. In addition to existing variables, you may need to define seasonality and trend related**

**variables. You may also want to include “lagged” variables. That is, you may want**

**to use Yt−1 (or any Xt−1) to explain Yt.**

We create a new data frame to continue. We start by taking the transformed SALES data.

new\_EP<-EP

new\_EP$SALES<-log(new\_EP$SALES)

**TREND**

We spotted an upwards trend. Consecutive numbers from 1 to 168 will represent our trend data.

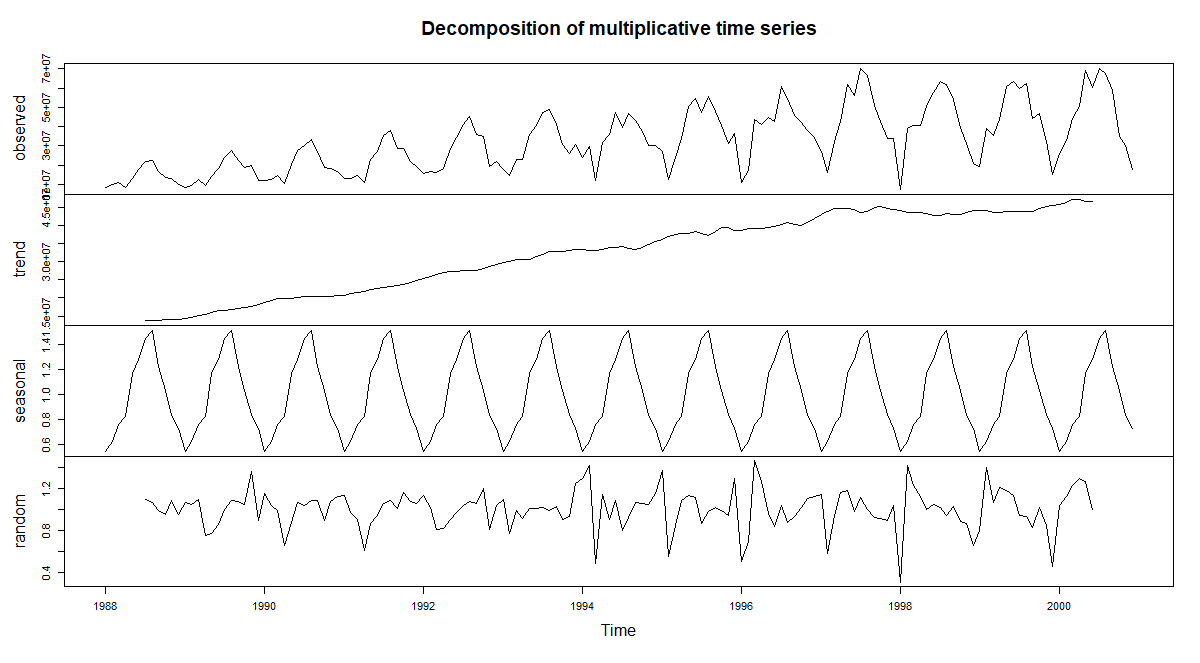
trend<-1:168

new\_EP$trend<-trend

**TREND CHANGE**

An interesting remark catches our eyes here in decomposition plot. Trend seems to change around 1997. So we add another variable to show whether it is before or after 1997.

decomp\_Ts<-decompose(EP\_ts,type="multiplicative")

plot(decomp\_Ts)

trend\_97<-c(rep(0,108),rep(1,60))

new\_EP$trend\_97<-trend\_97

**SEASONALITY**

We already knew period of the seasonality is 12 from the previous analyses. This indicates a monthly pattern. To represent this in our model we decided to use dummy variables for each month. Each of these variables can only have binary values of 0 or 1.

months<-rep(c("January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"),14)

new\_EP$month<-months

new\_EP<- dummy.data.frame(new\_EP, names = "month")

**LAGGED VARIABLES**

At this stage we need to separate training data and test data. After we build our regression model, we will check whether these lagged variables are included in the model. If they are included we need to forecast the test data one by one. Because our forecasts will become lagged variables for the other forecasts. So we just add lagged variables to training data. If they get included in the model, we will modify the test data as well.

train<-new\_EP[1:156,]

test<-new\_EP[157:nrow(new\_EP),]

**MONTHLY LAG**

We add the sales of the previous months to every next month as a predictive variable. We don’t think they will be important variables in our model. We already have trend so we would expect the intercept of the model to represent trend 0. The others will increase accordingly. Since every next month has +1 more trend value than the previous month, we hypothesize that the information we are adding with monthly lag is already in our existing variables. Of course, we want to prove that so we will check whether monthly lag is statistically meaningful or not.

First month doesn’t have a previous month. This information is not given in the data. First row could be deleted but we don’t think this is necessary. Missing value can be estimated by looking at the existing patterns in the data. We assume that difference between December 1987-January1988 and December 1988-January 1989 would be very similar. So sales of the December 1987 is estimated by subtracting the difference of the same two months of the next year from January 1988.

month\_lag<-train$SALES

fill\_month<-month\_lag[1]-(month\_lag[13]-month\_lag[12])

month\_lag<-c(fill\_month,month\_lag)

month\_lag<-month\_lag[-157]

train$month\_lag<-month\_lag

**YEARLY LAG**

Monthly consecutive sales have two general indicators for their change. Trend and seasonality. There is a small change in trend and a seasonal change which can be very high. To get rid of seasonality we can use yearly consecutive sales. However, there is a trade of, seasonality is absent but our trend value is much higher. We will check if this contributes to our model in a better way.

Yearly lag creates 12 months with no previous value. Lags are 12 instead of 1, so if we used the incremental method again to adjust our sales it had the chance to offset our data. Data points are not close to each other like the previous monthly case. So, the difference could be too high because of the trend. To overcome that we take a proportional approach.

SALES[i+12] = SALES[i]\*x

x = SALES[i+12] / SALES[i]

SALES[i] = SALES[i-12]\*x

SALES[i-12] = SALES[i] / x

SALES[i-12] = SALES[i]\*SALES[i]/SALES[i+12]

year\_lag<-train$SALES

fill\_year<-0

for(i in 1:12){

fill\_year<-c(fill\_year,year\_lag[i]\*(year\_lag[i]/year\_lag[i+12]))

}

fill\_year<-fill\_year[-1]

year\_lag<-c(fill\_year,year\_lag)

year\_lag<-year\_lag[1:156]

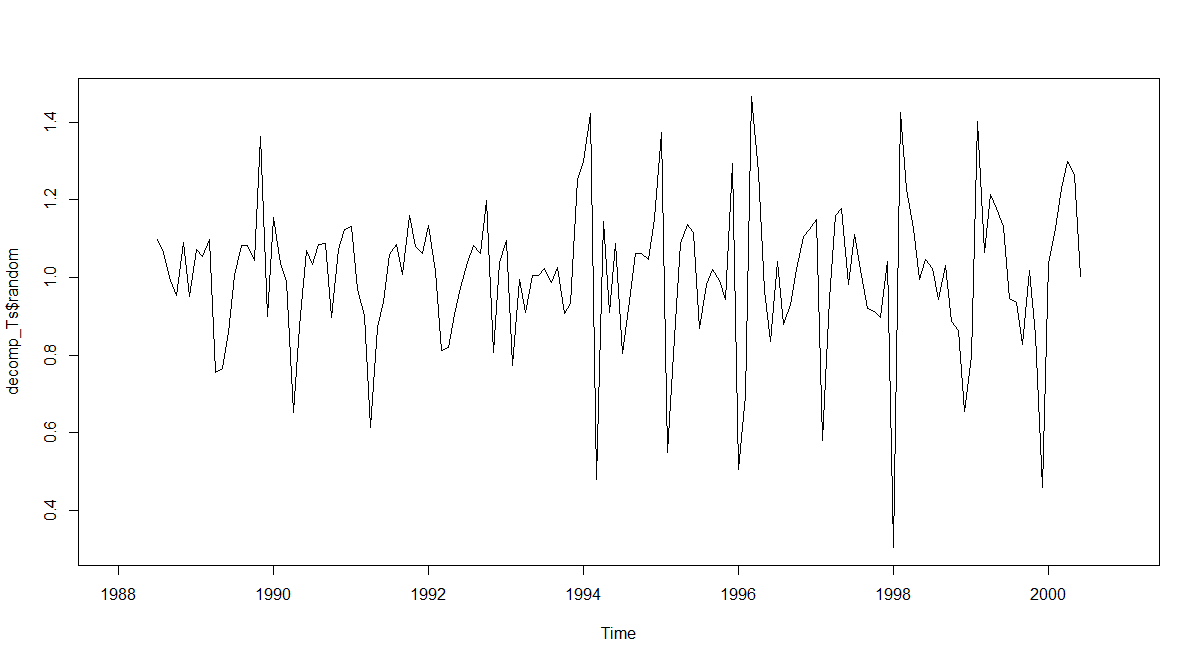
train$year\_lag<-year\_lag

**3. You may consider excluding some months (but you should only do so based on**

**statistical evidence) from your analysis.**

In this stage we need to find outliers inside the data. If we can find a statistical distribution to fit our data we can determine the outliers that doesn’t fit the distribution.

Our data has trend and seasonality. We need to get rid of these 2 before finding a statistical distribution to fit our data. If an increase or decrease happens because of seasonality, it doesn’t make the data point an outlier. An outlier point shouldn’t be explainable with the predictors we have.

 We already have the decomposed time series object from previous questions.

We defined a very narrow interval for outliers. Decomposition function can’t use predictors like RAMADAN or price parities. So most of the “outlier look-a-like” points can be explained with the predictors we have. We are looking for extreeme cases, that is why we choose 99% confidence interval.

error<-decomp\_Ts$random

error<-error[7:150]

alfa<-1-0.99

p<-1-alfa/2

dof<-length(error)-1

critical\_value<-qt(p, df=dof)

margin\_of\_Error<-sd(error)\*critical\_value

lower\_bound<-mean(error)-margin\_of\_Error

##[1] 0.512915

upper\_bound<-mean(error)+margin\_of\_Error

##[1] 1.495501

u<-0

l<-0

for(i in 1:131){

if(error[i]>upper\_bound)

{u<-cbind(u,i)

}

if(error[i]<lower\_bound)

{l<-cbind(l,i)}

}

u

##[1]

l

##[1,] 75 97 121

None of the points were found to be above the upper bound. 3 points were found below the lower bound. You can see the index values of these rows above.

total<-c(u[-1],l[-1])

total<-total+6

for(i in total){

train[i,]$SALES=mean(train[i-12,]$SALES,train[i+12,]$SALES)

}

We decided not to delete those outliers. Since there are only 3 of them we just adjusted these points by taking the mean of two points, 1 year after and 1 year prior.

**4. You should try various models (stepwise regression can be used) and come up with a model that explains beer sales in terms of independent and trend/seasonality variables**

1. We will use stepwise regression to find the best predictive variables.

Obviously we excluded DATE from our model because it is a character type variable and we already have the time information in our trend variable. monthDecember was excluded because we already have the December information in the rest of the 11 months. If all of them are 0 we can define this situation as existence of December. So we don’t need a predictor December.

stepAIC function tries to find the best AIC value for the model. Direction parameter is given as “both” so it will try both forward and backward regression then select the best model.

full.model <- lm(SALES ~.-DATE - monthDecember , data = train)

step.model <- stepAIC(full.model, direction = "both", trace = TRUE)

summary(step.model)

Call:

lm(formula = SALES ~ CORRECTED.PRICE + RAMADAN + TU.EP.PARITY +

RAKI.EP.PARITY + Cola.EP.Parity + trend + monthApril + monthAugust +

monthJuly + monthJune + monthMarch + monthMay + monthNovember +

monthOctober + monthSeptember + month\_lag + year\_lag, data = train)

Residuals:

Min 1Q Median 3Q Max

-0.48205 -0.07187 0.00293 0.06884 0.51601

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 9.9872359 1.3792167 7.241 2.83e-11 \*\*\*

CORRECTED.PRICE -0.0006787 0.0001539 -4.409 2.07e-05 \*\*\*

RAMADAN -0.0162950 0.0022344 -7.293 2.15e-11 \*\*\*

TU.EP.PARITY 1.7277013 0.8773292 1.969 0.050925 .

RAKI.EP.PARITY 0.0165291 0.0086183 1.918 0.057189 .

Cola.EP.Parity 0.1916284 0.1151307 1.664 0.098293 .

trend 0.0056130 0.0007250 7.742 1.89e-12 \*\*\*

monthApril 0.1881995 0.0488571 3.852 0.000179 \*\*\*

monthAugust 0.5604600 0.0708078 7.915 7.25e-13 \*\*\*

monthJuly 0.5472130 0.0685924 7.978 5.13e-13 \*\*\*

monthJune 0.4191266 0.0618981 6.771 3.37e-10 \*\*\*

monthMarch 0.2196334 0.0499980 4.393 2.21e-05 \*\*\*

monthMay 0.4081914 0.0573350 7.119 5.42e-11 \*\*\*

monthNovember 0.0721632 0.0521397 1.384 0.168582

monthOctober 0.2295112 0.0573530 4.002 0.000102 \*\*\*

monthSeptember 0.3645515 0.0647486 5.630 9.70e-08 \*\*\*

month\_lag 0.0831864 0.0557930 1.491 0.138248

year\_lag 0.2052418 0.0625849 3.279 0.001317 \*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1472 on 138 degrees of freedom

Multiple R-squared: 0.9343, Adjusted R-squared: 0.9262

F-statistic: 115.4 on 17 and 138 DF, p-value: < 2.2e-16

month\_lag is not significant but it is kept in the model. Reason might be that it is almost significant with a p-value close to 0.1. We explained earlier that information about month\_lag is probably is mostly covered by other predictors like trend or seasonality. So, we decide to exclude it from the model.

full.model <- lm(SALES ~.-DATE - monthDecember-month\_lag , data = train)

step.model <- stepAIC(full.model, direction = "both",

trace = FALSE)

**5. You should present all the statistical evidence supporting the validity of your analysis**

**(residual analysis, Durbin-Watson statistics, collinearity checks, significance of**

**coefficients, etc.).**

**Significance of Coefficients test:**

summary(step.model)

Call:

lm(formula = SALES ~ CORRECTED.PRICE + RAMADAN + TU.EP.PARITY +

RAKI.EP.PARITY + Cola.EP.Parity + trend + monthApril + monthAugust +

monthFebruary + monthJuly + monthJune + monthMarch + monthMay +

monthNovember + monthOctober + monthSeptember + year\_lag,

data = train)

Residuals:

Min 1Q Median 3Q Max

-0.45732 -0.06444 0.00621 0.07047 0.53006

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 10.4605889 1.3558506 7.715 2.18e-12 \*\*\*

CORRECTED.PRICE -0.0007062 0.0001538 -4.590 9.86e-06 \*\*\*

RAMADAN -0.0157961 0.0022128 -7.139 4.90e-11 \*\*\*

TU.EP.PARITY 1.9167048 0.8630042 2.221 0.027983 \*

RAKI.EP.PARITY 0.0173390 0.0085855 2.020 0.045365 \*

Cola.EP.Parity 0.1839828 0.1148306 1.602 0.111396

trend 0.0059437 0.0006754 8.801 5.00e-15 \*\*\*

monthApril 0.1661943 0.0518475 3.205 0.001676 \*\*

monthAugust 0.5640156 0.0705669 7.993 4.73e-13 \*\*\*

monthFebruary -0.0701058 0.0503971 -1.391 0.166443

monthJuly 0.5424376 0.0695379 7.801 1.36e-12 \*\*\*

monthJune 0.4093841 0.0635236 6.445 1.80e-09 \*\*\*

monthMarch 0.1821283 0.0520536 3.499 0.000629 \*\*\*

monthMay 0.3734542 0.0592126 6.307 3.60e-09 \*\*\*

monthNovember 0.0725900 0.0524514 1.384 0.168609

monthOctober 0.2364776 0.0562597 4.203 4.70e-05 \*\*\*

monthSeptember 0.3808667 0.0614991 6.193 6.35e-09 \*\*\*

year\_lag 0.2503417 0.0551784 4.537 1.23e-05 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1474 on 138 degrees of freedom

Multiple R-squared: 0.9342, Adjusted R-squared: 0.926

F-statistic: 115.2 on 17 and 138 DF, p-value: < 2.2e-16

There are two non-significant predictors. We decide to keep them. November is kept for the integrity of the month variables. Absence of 1 month might bring inconsistencies to our model. We keep the Cola parity for the parity integrity. Since parities are for competitive products against EP, we decided leaving 1 out might weaken our model.

**Durbin Watson test:**

dwtest(step.model)

Durbin-Watson test

data: step.model

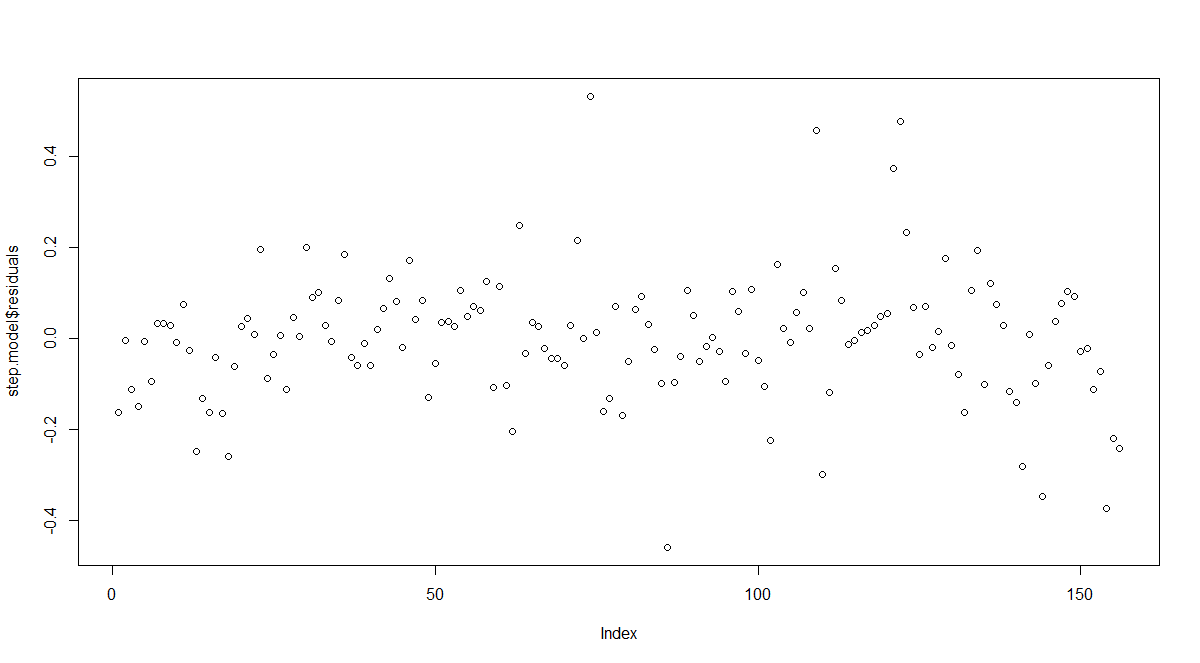
DW = 1.4606, p-value = 0.0001932

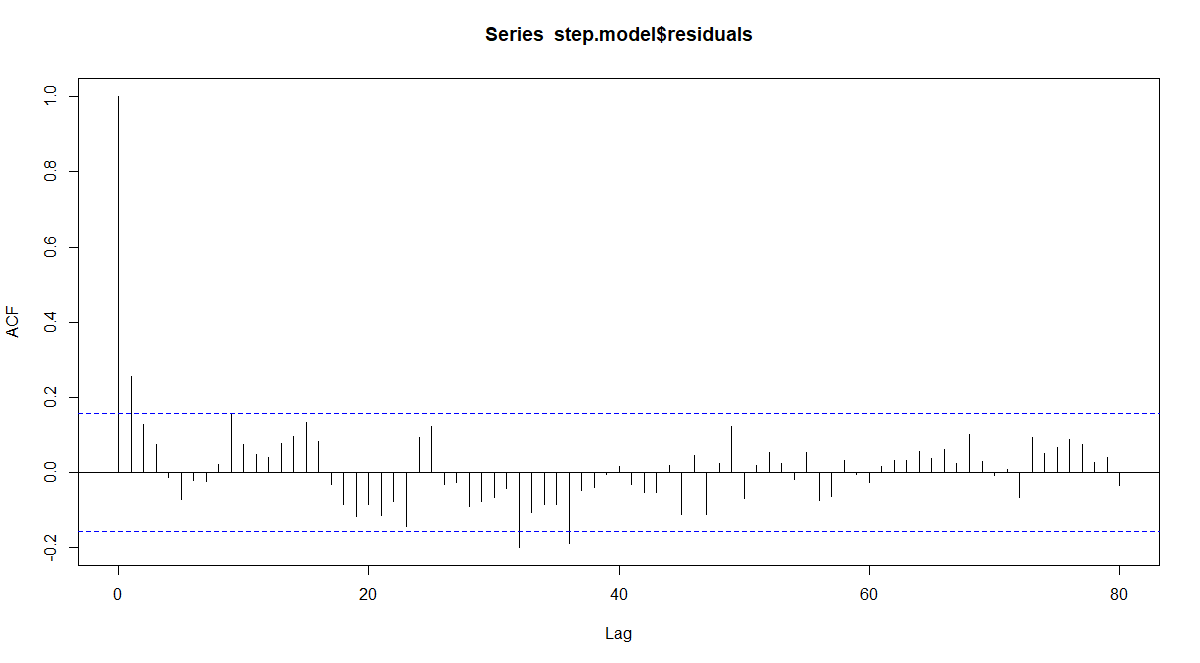
alternative hypothesis: true autocorrelation is greater than 0

DW value can take values between 0 and 4. Below 2 means negative correlation (which is common in time series data). Above 2 is positive correlation. 0 is no correlation. Also, it is relatively normal to have DW values around 1.5 or 2.5. In conclusion we decide that our model is not perfectly decorrelated (which is expected since we have lag variables in our model). But the problem is in acceptable levels, so we fail the reject the Null Hypothesis. Which is that the true correlation is equal to 0.

**Model Residuals:**

plot(step.model$residuals)

 acf(step.model$residuals,lag.max = 80)



Model residuals have 0 mean, they are distributed around 0. They also seem to have a constant variance, so there is no information left out to explain the residuals since they are random. And the ACF plot doesn’t show many significantly correlated lags.

**Collinearity:**

Collinearity is the case of including linearly dependent prediction variables in one model. However, since we used stepwise regression to choose our variables, linearly dependent variables were eliminated automatically. They would worsen the model if they were added together and stepwise methods detect that. Methods like checking the AIC value or looking at the F statistics enable us to get rid of collinearity.

**6. Using the fitted model forecast beer sales for 2001 (all months).**

t\_year\_lag<-train[145:156,]$year\_lag

test$year\_lag<-t\_year\_lag

We added year\_lag to our test data because it is included in the model.

forecasts<-forecast(step.model,test)

forecasts

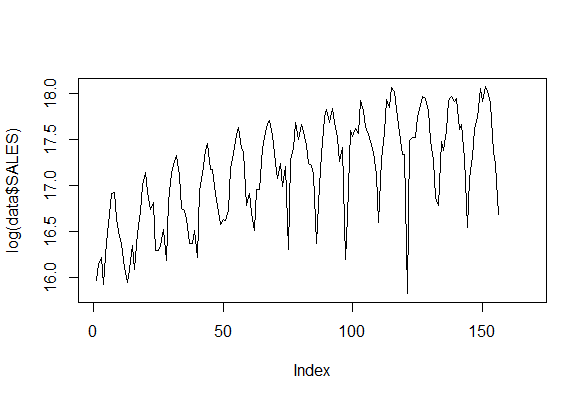
|  |
| --- |
| Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  1 17.33965 17.13839 17.54091 17.03062 17.64869  2 17.41194 17.21147 17.61241 17.10411 17.71977  3 17.68058 17.47831 17.88284 17.37000 17.99115  4 17.70280 17.50051 17.90509 17.39218 18.01342  5 17.91653 17.71309 18.11996 17.60415 18.22890  6 18.04398 17.84364 18.24431 17.73636 18.35160  7 18.05028 17.84580 18.25476 17.73630 18.36427  8 18.15243 17.95000 18.35485 17.84160 18.46325  9 17.97421 17.76960 18.17882 17.66003 18.28840  10 17.76166 17.56112 17.96221 17.45372 18.06960  11 17.23267 17.02642 17.43891 16.91598 17.54936  12 16.93070 16.72378 17.13763 16.61296 17.24844 |

|  |  |
| --- | --- |
| **Date** | **FORECASTED VALUES** |
| **1-Jan-01** | **33924567** |
| **1-Feb-01** | **36467791** |
| **1-Mar-01** | **47706585** |
| **1-Apr-01** | **48778490** |
| **1-May-01** | **60401832** |
| **1-Jun-01** | **68612137** |
| **1-Jul-01** | **69045758** |
| **1-Aug-01** | **76471601** |
| **1-Sep-01** | **63988248** |
| **1-Oct-01** | **51735771** |
| **1-Nov-01** | **30482704** |
| **1-Dec-01** | **22537700** |

***METHOD B: FORECASTING WITH TIME SERIES ANALYSIS***

**1. You should first check if you need a preliminary transformation to induce stationarity.**

plot(log(data$SALES),type = "l")



Logarithm reduces the variance fluctuation and differencing becomes the data stationary.To maket he data stationary we will take difference of the values in the second part.

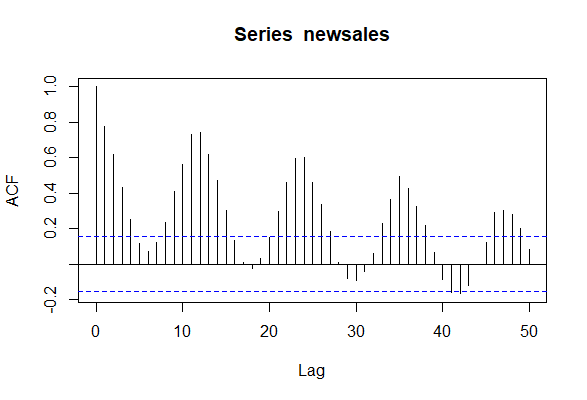
transformed\_data<-data.frame(log(data$SALES[1:156]))

**2. You should utilize time series plot, ACF and PACF plots in order to determine**

**regular and seasonal differencing to be applied.**

newsales<-transformed\_data$log.data.SALES.

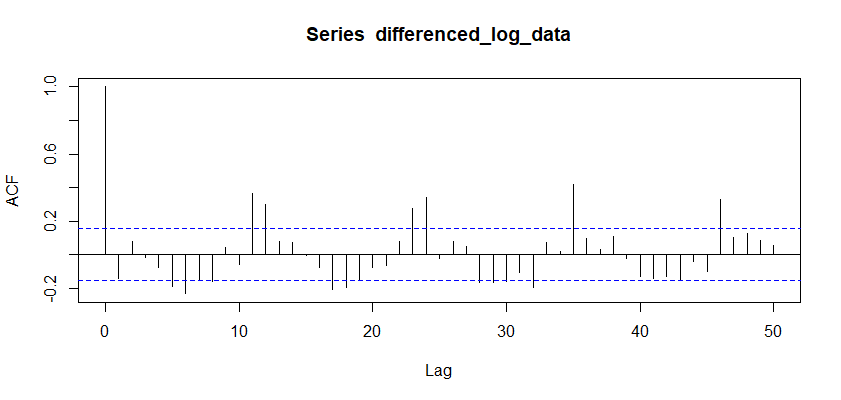
acf(newsales,lag.max = 50)



Differencing is necessary

differenced\_log\_data<-diff(newsales,differences = 1)

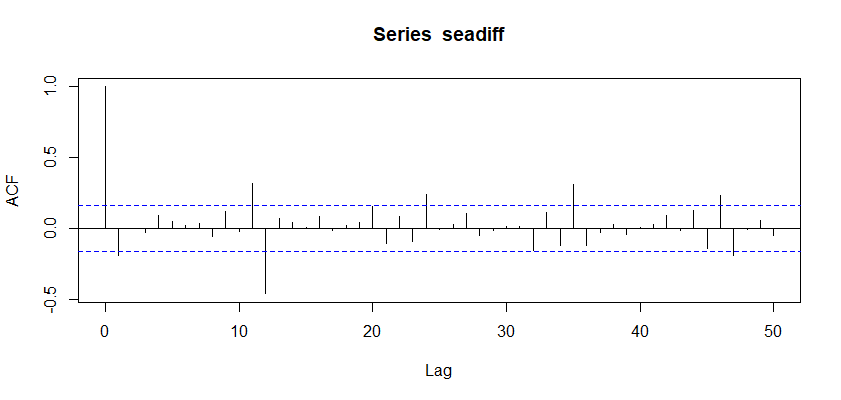
acf(differenced\_log\_data,lag.max = 50)



Seasonal differencing is necessary because even in a differenced data there can be seen some trends.

seadiff <- diff(newsales, lag = 12, differences = 1)

acf(seadiff,lag.max = 50)



Now,The trend disappeared.So,our model should be d=1 (Differencing)and D=1(Seasonal differencing).

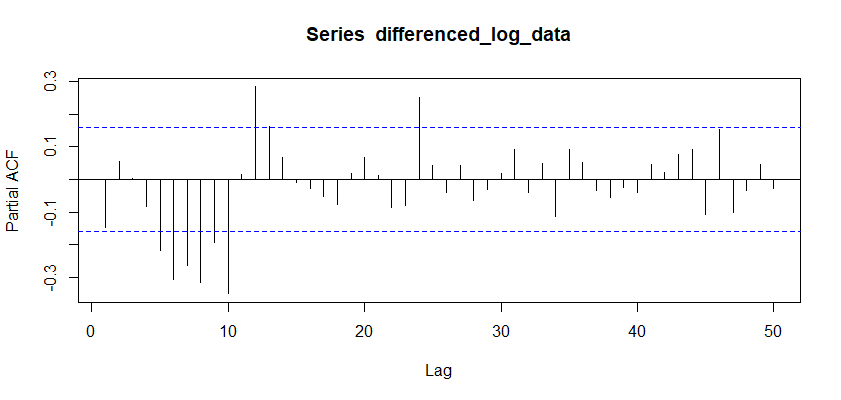
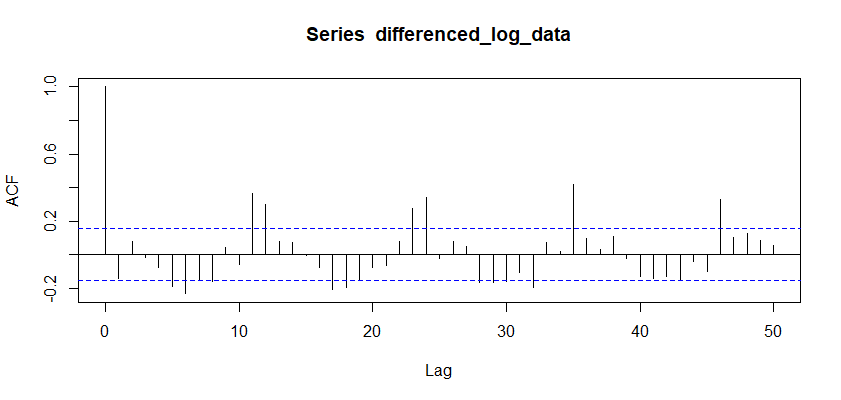
**4. You should come up with an initial ARIMA model based on inspection of ACF and PACF plots.**

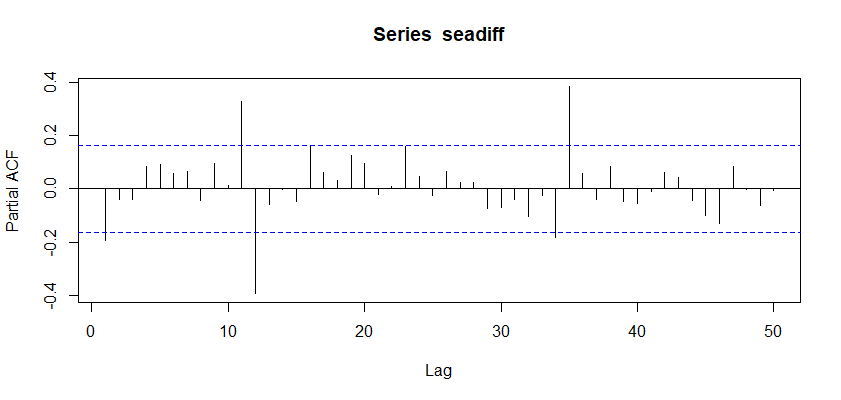
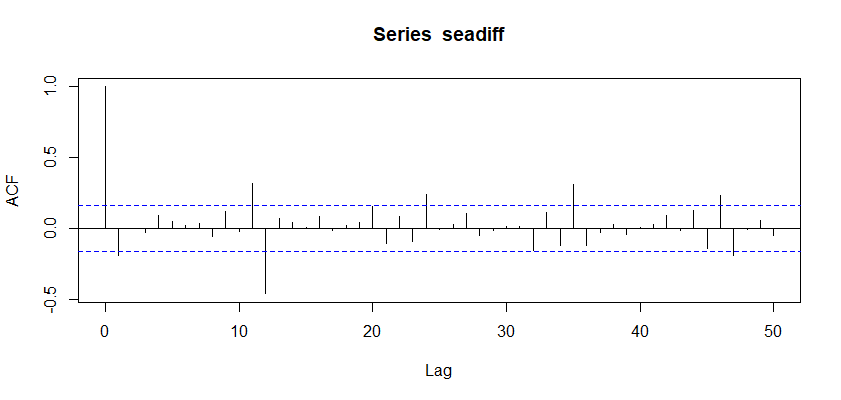
acf(differenced\_log\_data,lag.max = 50)

pacf(differenced\_log\_data,lag.max = 50)

acf(seadiff,lag.max = 50)

pacf(seadiff,lag.max = 50)





When we look at the ACF and the PACF plots ,as aninitial ARIMA model,we can use MA=2,AR=2,SMA=1 and SAR=1 which is order=c(2,1,2),seasonal=c(1,1,1).

**5. You should performs a neighborhood search of the initial model. That is, a search**

**around initial AR, MA, and seasonal AR, seasonal MA terms by increasing or**

**decreasing them. Limit yourself to six such candidate models.**

model\_time\_serie<-ts(transformed\_data$log.data.SALES.1.156..,start = 1988,frequency = 12)

Arima\_initial<-Arima(model\_time\_serie,order=c(2,1,2),seasonal=c(1,1,1))

Arima1<-Arima(model\_time\_serie,order=c(1,1,2),seasonal=c(1,1,1))

Arima2<-Arima(model\_time\_serie,order=c(1,1,1),seasonal=c(1,1,1))

Arima3<-Arima(model\_time\_serie,order=c(1,1,3),seasonal=c(1,1,1))

Arima4<-Arima(model\_time\_serie,order=c(1,1,2),seasonal=c(0,1,1))

Arima5<-Arima(model\_time\_serie,order=c(1,1,2),seasonal=c(1,1,0))

Arima6<-Arima(model\_time\_serie,order=c(1,1,2),seasonal=c(1,1,2))

Arima7<-Arima(model\_time\_serie,order=c(1,1,2),seasonal=c(2,1,1))

Arima8<-Arima(model\_time\_serie,order=c(1,1,2),seasonal=c(1,1,2))

Arima9<-Arima(model\_time\_serie,order=c(2,1,1),seasonal=c(1,1,1))

Arima10<-Arima(model\_time\_serie,order=c(2,1,1),seasonal=c(1,1,0))

Arima11<-Arima(model\_time\_serie,order=c(2,1,1),seasonal=c(0,1,1))

Arima12<-Arima(model\_time\_serie,order=c(3,1,1),seasonal=c(1,1,1))

Arima13<-Arima(model\_time\_serie,order=c(2,1,1),seasonal=c(1,1,2))

Arima14<-Arima(model\_time\_serie,order=c(2,1,1),seasonal=c(2,1,1))

Arima15<-Arima(model\_time\_serie,order=c(2,1,1),seasonal=c(2,1,2))

Arima15<-Arima(model\_time\_serie,order=c(3,1,2),seasonal=c(1,1,1))

Arima16<-Arima(model\_time\_serie,order=c(3,1,1),seasonal=c(1,1,1))

Arima17<-Arima(model\_time\_serie,order=c(3,1,3),seasonal=c(1,1,1))

Arima18<-Arima(model\_time\_serie,order=c(3,1,2),seasonal=c(0,1,1))

Arima19<-Arima(model\_time\_serie,order=c(3,1,2),seasonal=c(1,1,0))

Arima20<-Arima(model\_time\_serie,order=c(3,1,2),seasonal=c(0,1,0))

Arima21<-Arima(model\_time\_serie,order=c(3,1,2),seasonal=c(2,1,2))

Arima22<-Arima(model\_time\_serie,order=c(3,1,2),seasonal=c(2,1,1))

Arima23<-Arima(model\_time\_serie,order=c(3,1,2),seasonal=c(1,1,2))

Arima24<-Arima(model\_time\_serie,order=c(1,1,3),seasonal=c(1,1,1))

Arima25<-Arima(model\_time\_serie,order=c(2,1,3),seasonal=c(1,1,1))

Arima26<-Arima(model\_time\_serie,order=c(3,1,3),seasonal=c(1,1,1))

Arima27<-Arima(model\_time\_serie,order=c(2,1,3),seasonal=c(0,1,1))

Arima28<-Arima(model\_time\_serie,order=c(2,1,3),seasonal=c(1,1,0))

Arima29<-Arima(model\_time\_serie,order=c(2,1,3),seasonal=c(2,1,1))

Arima30<-Arima(model\_time\_serie,order=c(2,1,3),seasonal=c(1,1,2))

Arima31<-Arima(model\_time\_serie,order=c(2,1,2),seasonal=c(0,1,2))

Arima32<-Arima(model\_time\_serie,order=c(2,1,2),seasonal=c(2,1,0))

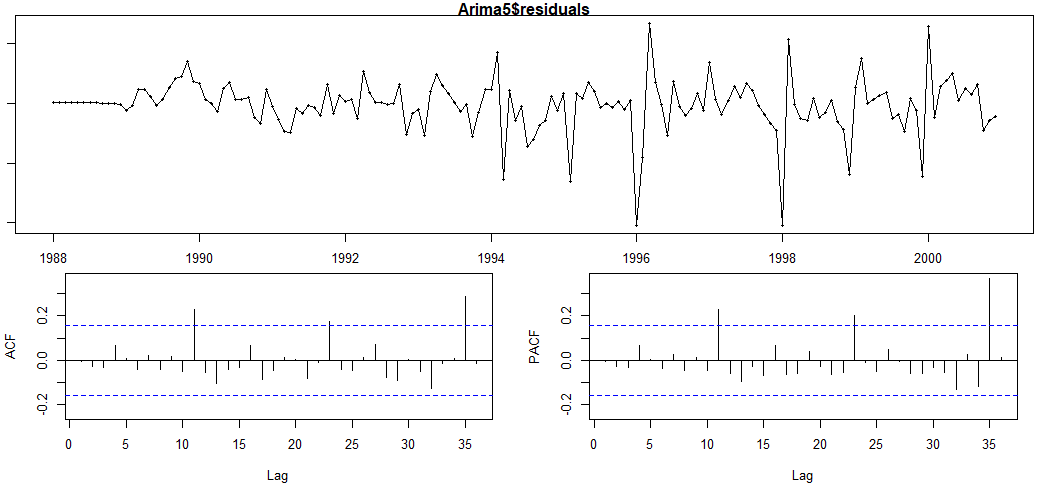
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MODELS** | **AİC VALUES** | **MODELS** | **AİC VALUES** | **MODELS** | **AİC VALUES** |
| **Arima\_initial** | **8.933635** | **Arima11** | **12.51981** | **Arima22** | **10.29914** |
| **Arima1** | **6.980101** | **Arima12** | **8.58213** | **Arima23** | **12.48963** |
| **Arima2** | **9.525377** | **Arima13** | **10.81208** | **Arima24** | **8.934012** |
| **Arima3** | **8.934012** | **Arima14** | **8.978454** | **Arima25** | **10.93324** |
| **Arima4** | **8.203166** | **Arima15** | **10.57378** | **Arima26** | **12.5747** |
| **Arima5** | **5.302366** | **Arima16** | **8.58213** | **Arima27** | **11.05842** |
| **Arima6** | **8.970557** | **Arima17** | **12.5747** | **Arima28** | **9.207721** |
| **Arima7** | **7.961236** | **Arima18** | **10.14722** | **Arima29** | **7.133034** |
| **Arima8** | **8.970557** | **Arima19** | **8.798499** | **Arima30** | **12.92876** |
| **Arima9** | **8.837557** | **Arima20** | **46.94284** | **Arima31** | **10.4555** |
| **Arima10** | **7.079925** | **Arima21** | **11.9428** | **Arima22** | **8.910646** |

|  |  |
| --- | --- |
| MODELS | AİC VALUES |
| Arima5 | 5.302366 |
| Arima1 | 6.980101 |
| Arima10 | 7.079925 |
| Arima29 | 7.133034 |
| Arima7 | 7.961236 |
| Arima4 | 8.203166 |

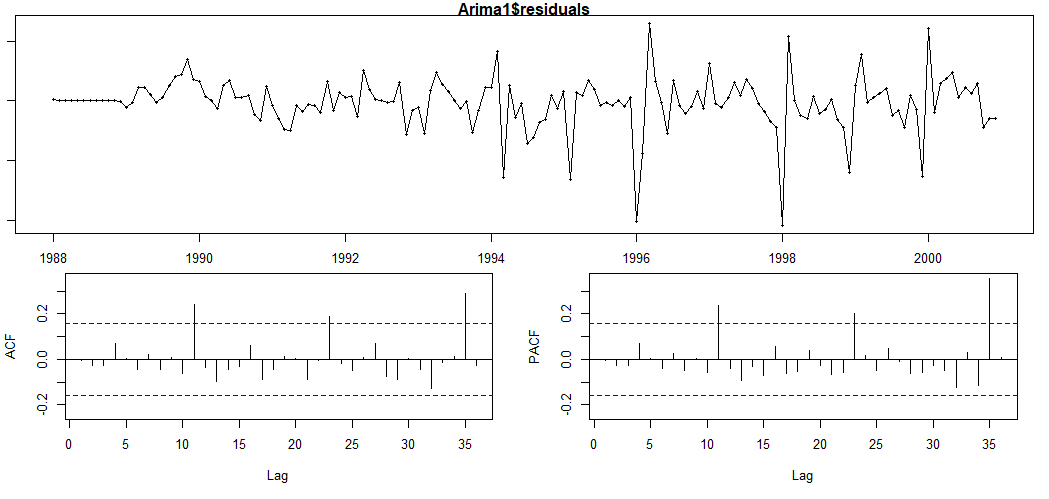
The models that gives the least aic values.

**6. These candidate models should always be accompanied with ACF and PACF plots.**

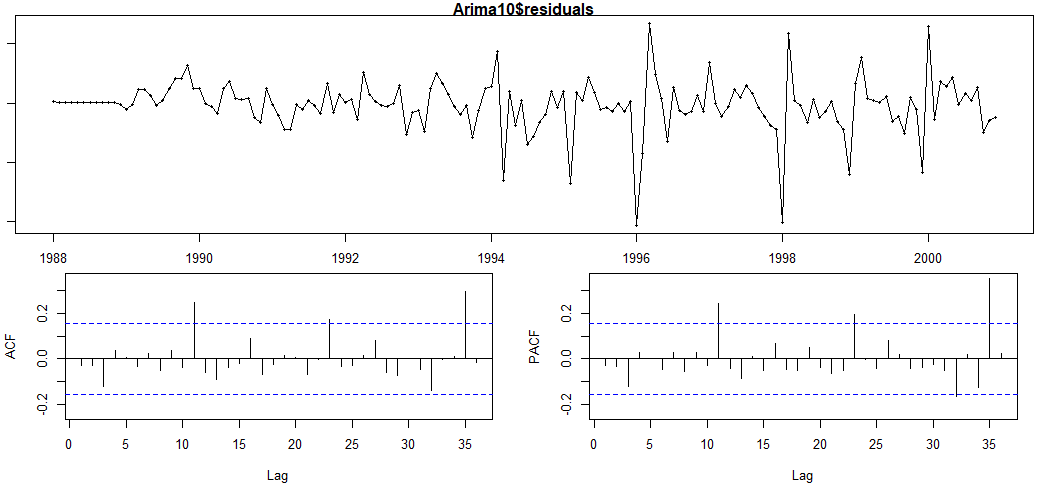
tsdisplay(Arima5$residuals)



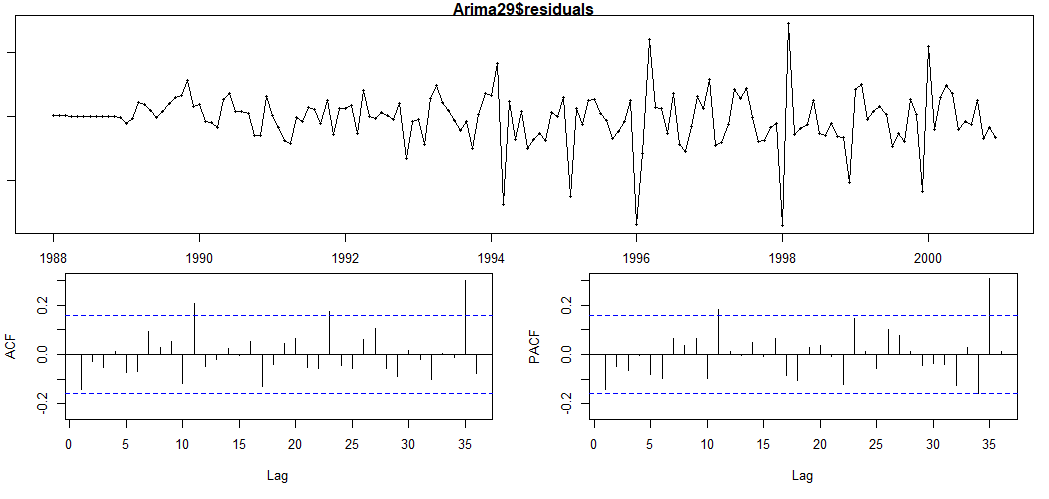
tsdisplay(Arima1$residuals)



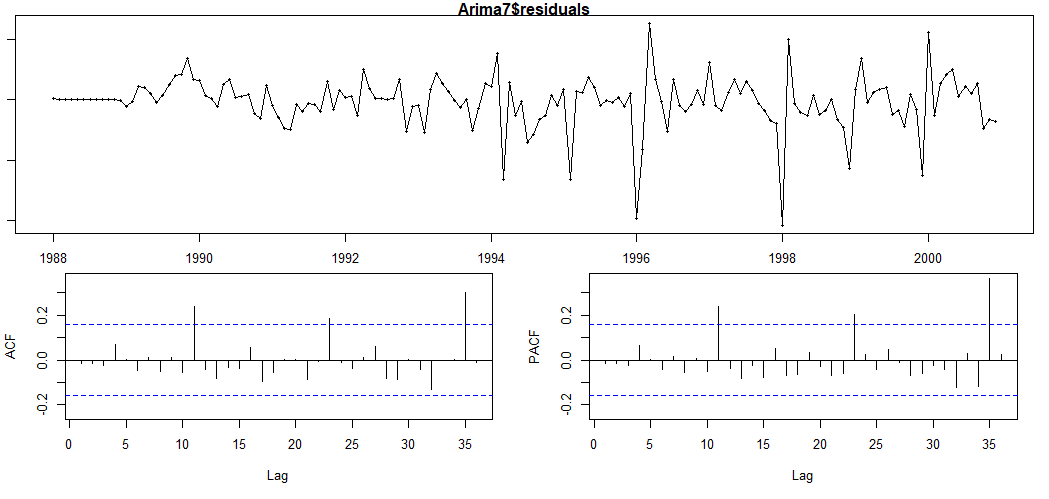
tsdisplay(Arima10$residuals)



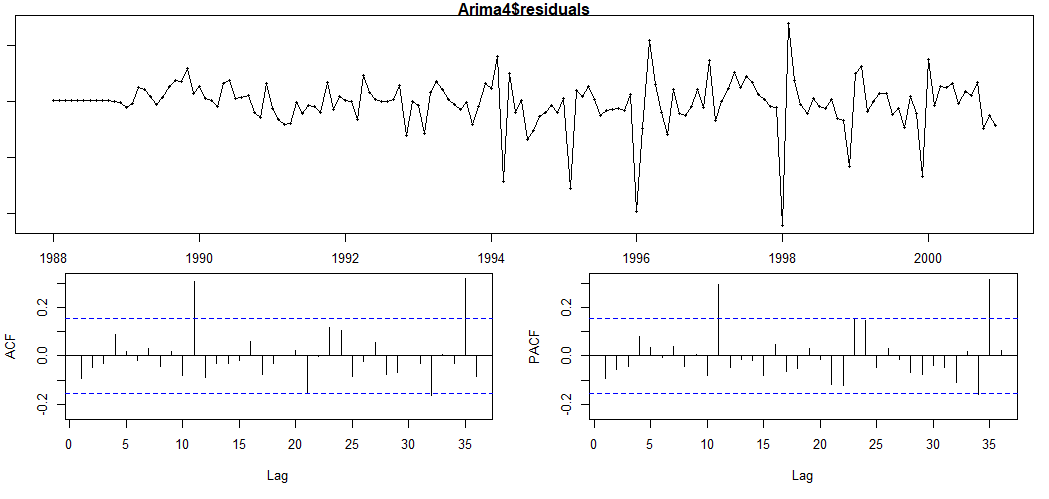
tsdisplay(Arima29$residuals)



tsdisplay(Arima7$residuals)



tsdisplay(Arima4$residuals)



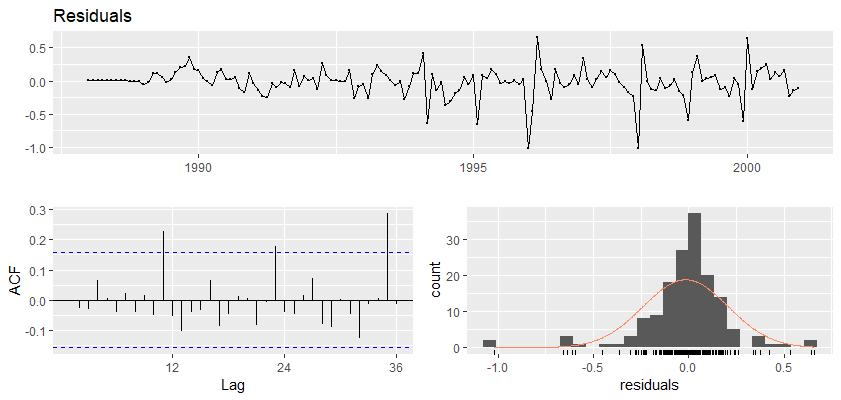
We can see from the ACF and PACF plots that all the six model are suitable for us.

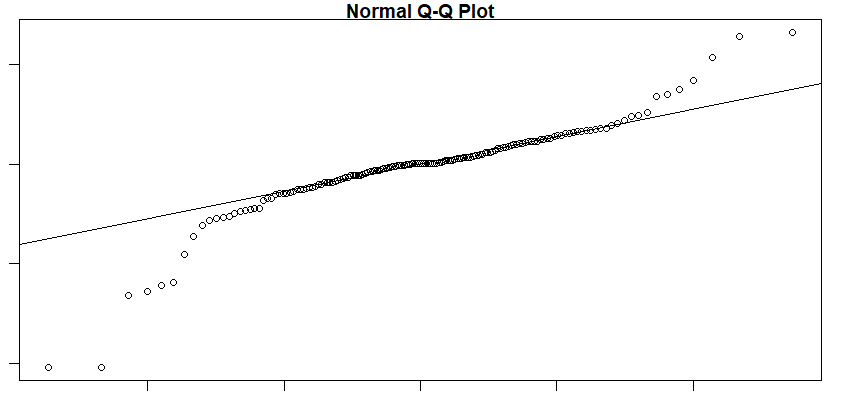
**7. You should decide on the “best” model to use in forecasting**.

|  |  |
| --- | --- |
| MODELS | AİC VALUES |
| Arima5 | 5.302366 |
| Arima1 | 6.980101 |
| Arima10 | 7.079925 |
| Arima29 | 7.133034 |
| Arima7 | 7.961236 |
| Arima4 | 8.203166 |

When we look at the Aic values for all models, The best fitted model that gives also best aic value is model ARIMA(1,1,2)(1,1,0)[12] (ARIMA5).

**8. You should present all the statistical evidence supporting the validity of your analysis.**





When we move away we may have some data out of the line, but most of our data fluctuates around the line in q-q plot.

ACF of the residuals shows that errors are uncorrelated, we provide statistical independence of #the errors.

From standardized residuals, we can observe the constant mean and variance of the errors. (Homoscedasticity).

**#H0:** The data are independently distributed (i.e. the correlations in the population from which #the sample is taken are 0, so that any observed correlations in the data result from randomness #of the sampling process).

**#Ha:** The data are not independently distributed; they exhibit serial correlation.

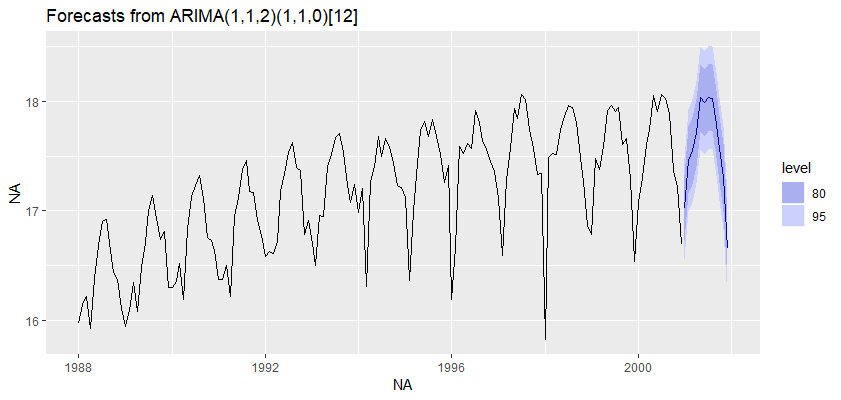
Our p-values are high at the first lags and decreases slowly towards the end, but since #they are above the critical level, we can fail to reject H0, which says there is an independence #between the errors.

There are spikes at lag 12, 24, 36 and so on due to seasonality of the ARIMA model.

**9. Forecast beer sales in 2001 (all months) with the selected method.**

forecast\_arima<-forecast(Arima5,h=12\*1)

autoplot((forecast\_arima))



print(exp(summary(forecast\_arima)))

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

Jan 2001 24609601 18186486 33301236 15495773 39083721

Feb 2001 38330705 28204894 52091772 23977331 61276333

Mar 2001 41762281 30719494 56774636 26110319 66796891

Apr 2001 49399173 36336657 67157479 30884507 79013026

May 2001 67892905 49931332 92315714 42435403 108622664

Jun 2001 64818508 47654996 88163662 40493923 103754803

Jul 2001 67974278 49955053 92493196 42439320 108873151

Aug 2001 67747075 49766065 92224816 42268868 108582661

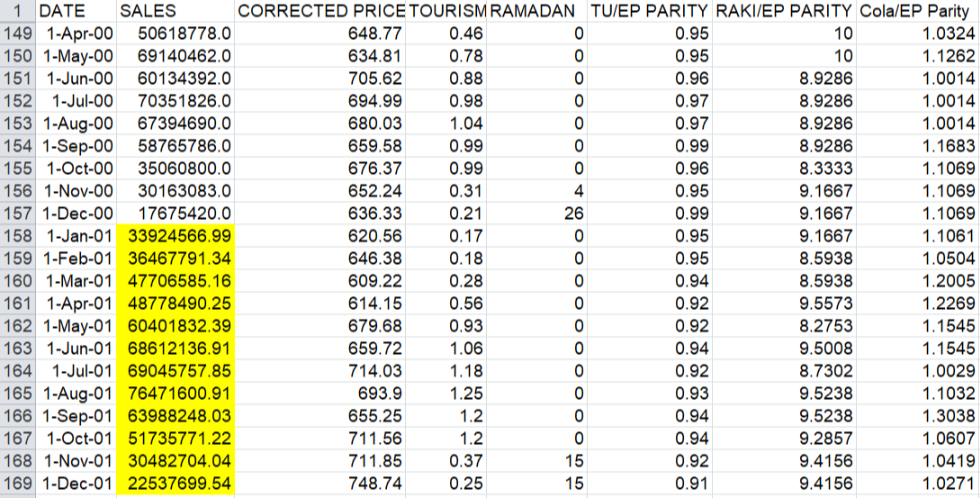
Sep 2001 53397971 39207328 72724756 33292664 85644794

Oct 2001 42562283 31236578 57994442 26517757 68314522

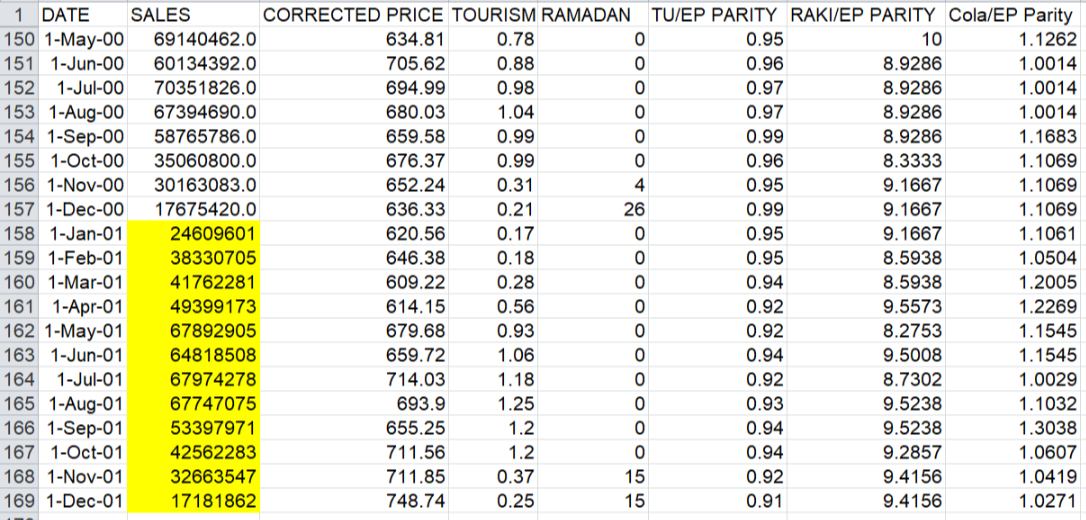
Nov 2001 32663547 23960528 44527703 20335785 52464523

Dec 2001 17181862 12597873 23433826 10689385 27617713

**RESULTS**



**Forecasts with Regression Model (Part A)**



**Forecasts with Arima Model (Part B)**

|  |  |  |  |
| --- | --- | --- | --- |
| **DATE** | **FORECAST VALUES** | **LOWER BOUNDS** | **UPPER BOUNDS** |
| **1-Jan-01** | **33924567** | **24906018** | **46209224** |
| **1-Feb-01** | **36467791** | **26805295** | **49613325** |
| **1-Mar-01** | **47706585** | **34969961** | **65081442** |
| **1-Apr-01** | **48778490** | **35754261** | **66547065** |
| **1-May-01** | **60401832** | **44196228** | **82548784** |
| **1-Jun-01** | **68612137** | **50443275** | **93325132** |
| **1-Jul-01** | **69045758** | **50440249** | **94515084** |
| **1-Aug-01** | **76471601** | **56041329** | **1.04E+08** |
| **1-Sep-01** | **63988248** | **46736219** | **87609500** |
| **1-Oct-01** | **51735771** | **38023692** | **70392691** |
| **1-Nov-01** | **30482704** | **22208374** | **41839859** |
| **1-Dec-01** | **22537700** | **16402757** | **30967227** |

**Forecasted Values in Regression part with 95% confidence interval**

|  |  |  |  |
| --- | --- | --- | --- |
| **DATE** | **FORECAST VALUES** | **LOWER BOUNDS** | **UPPER BOUNDS** |
| **1-Jan-01** | **24609601** | **15495773** | **39083721** |
| **1-Feb-01** | **38330705** | **23977331** | **61276333** |
| **1-Mar-01** | **41762281** | **26110319** | **66796891** |
| **1-Apr-01** | **49399173** | **30884507** | **79013026** |
| **1-May-01** | **67892905** | **42435403** | **1.09E+08** |
| **1-Jun-01** | **64818508** | **40493923** | **1.04E+08** |
| **1-Jul-01** | **67974278** | **42439320** | **1.09E+08** |
| **1-Aug-01** | **67747075** | **42268868** | **1.09E+08** |
| **1-Sep-01** | **53397971** | **33292664** | **85644794** |
| **1-Oct-01** | **42562283** | **26517757** | **68314522** |
| **1-Nov-01** | **32663547** | **20335785** | **52464523** |
| **1-Dec-01** | **17181862** | **10689385** | **27617713** |

**Forecasted Values in Arima part with 95% confidence interval**

As a consequence,when we controled the confidence intervals for both results,we concluded that forecasting with regression analysis seems to be the right one to use, due to its narrower confidence interval. Because we know that narrower confidence intervals give us better forecasting results,we end up this project with selecting regression analysis, rather than time series analysis.