

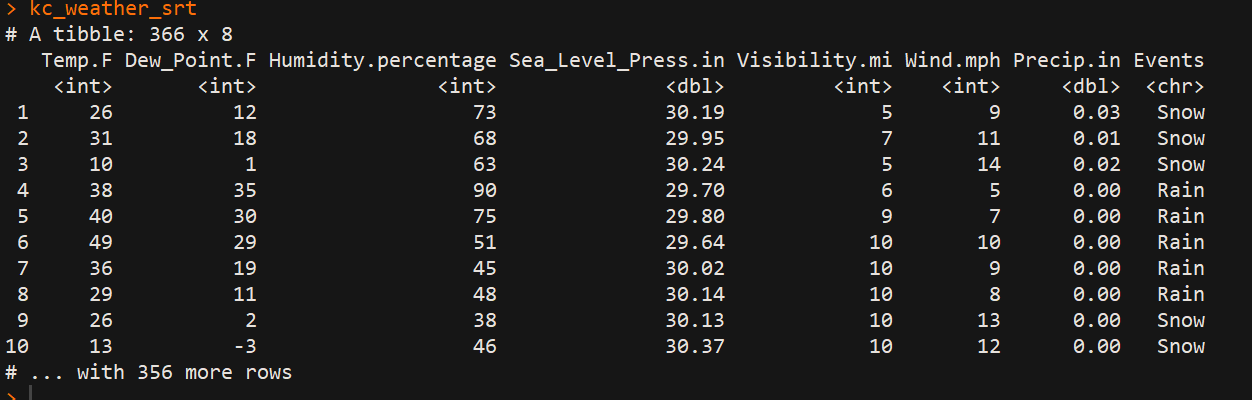
**PROGRAM**

*#import the dataset and make some changes*

library(readr)

kc\_weather\_srt <- read\_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/kc\_weather\_srt.csv")

kc\_weather\_srt=kc\_weather\_srt[,2:9]



*#first make the response column to 0-snow, 1-rain and 2-rain\_thunderstorm*

*#install.packages("plyr")*

library(plyr)

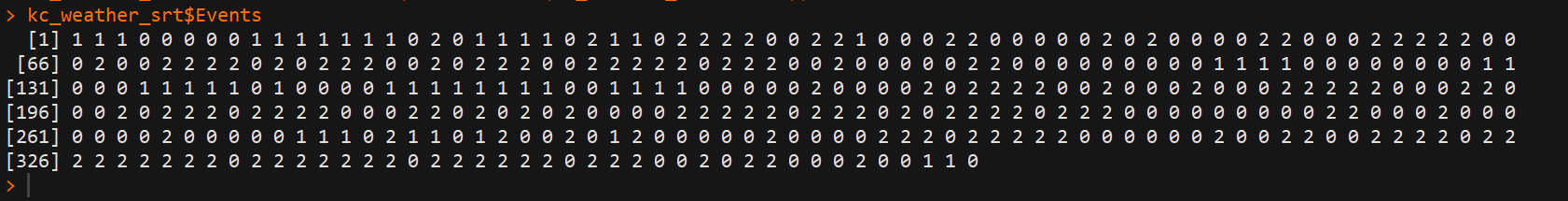
kc\_weather\_srt$Events <- revalue(kc\_weather\_srt$Events,c("Snow"=1))

kc\_weather\_srt$Events <- revalue(kc\_weather\_srt$Events,c("Rain"=0))

kc\_weather\_srt$Events <- revalue(kc\_weather\_srt$Events,c("Rain\_Thunderstorm"=2))

*#small changes to Events column , making it to numeric from character*

kc\_weather\_srt$Events<-as.numeric(as.character(kc\_weather\_srt$Events))



*#replications*

rep=100

*# newly added*

accuracy1=dim(rep)

precision\_snow1=dim(rep)

precision\_rain1=dim(rep)

precision\_rainThunderstorm1=dim(rep)

recall\_snow1=dim(rep)

recall\_rain1=dim(rep)

recall\_rainThunderstorm1=dim(rep)

*#splitting the dataset into training and test sets, also install caTools packages*

*#install.packages('caTools')*

library(caTools)

set.seed(123)

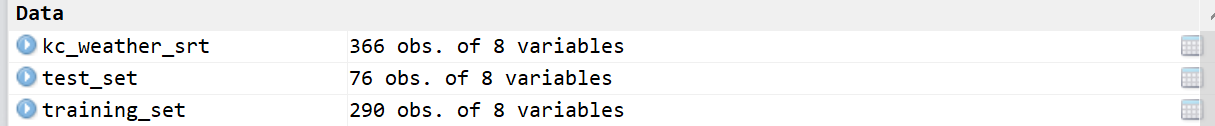
for(k in 1:rep)

{

split=sample.split(kc\_weather\_srt$Events,SplitRatio = 0.7923)

training\_set=subset(kc\_weather\_srt,split==TRUE)

test\_set=subset(kc\_weather\_srt,split==FALSE)



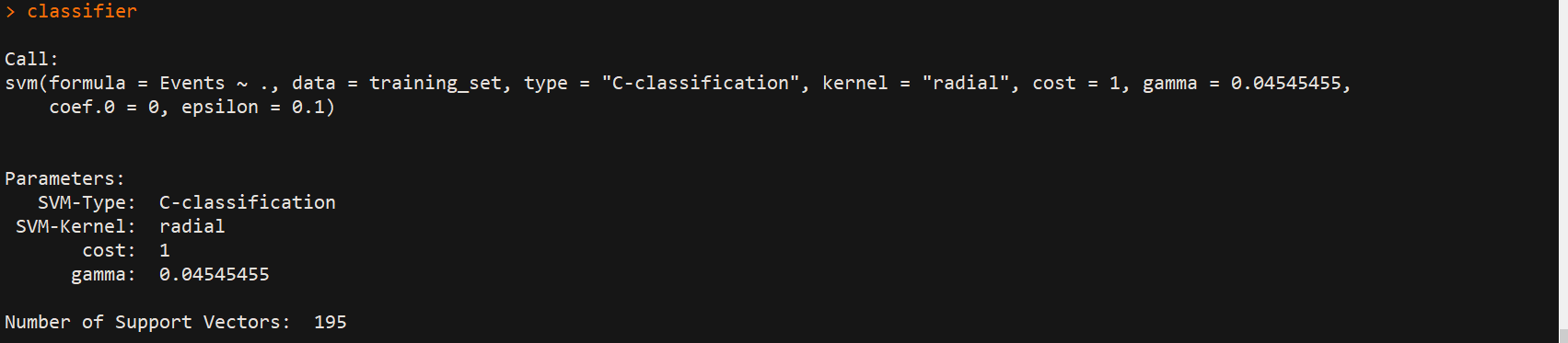
*#\*\*\*\*\*SVM\*\*\*\*\*#*

*#fitting SVM to the training set*

*#install.packages('e1071')*

library(e1071)

classifier=svm(Events~.,data=training\_set,type='C-classification',kernel="radial",cost=1,gamma=0.04545455,coef.0=0,epsilon=0.1)



y\_pred1=predict(classifier,newdata = test\_set[-8])

*#making the confusion matrix*

cm1=table(test\_set$Events,y\_pred1)



*#calculating the accuracy*

accuracy1[k]=mean(y\_pred1==test\_set$Events)

*#Precision of rain, rain\_thunderstorm and snow results*

precision1=precision1<-diag(cm1)/colSums(cm1)

precision\_rainThunderstorm1[k]=precision1[3]

precision\_snow1[k]=precision1[2]

precision\_rain1[k]=precision1[1]

*#Recall of rain, rain\_thunderstorm and snow results*

recall1=recall1<-diag(cm1/rowSums(cm1))

recall\_rainThunderstorm1[k]=recall1[3]

recall\_snow1[k]=recall1[2]

recall\_rain1[k]=recall1[1]

}

*#Calculating the end results using mean*

mean(accuracy1)

mean(precision\_rain1)

mean(precision\_rainThunderstorm1)

mean(precision\_snow1)

mean(recall\_rain1)

mean(recall\_rainThunderstorm1)

mean(recall\_snow1)

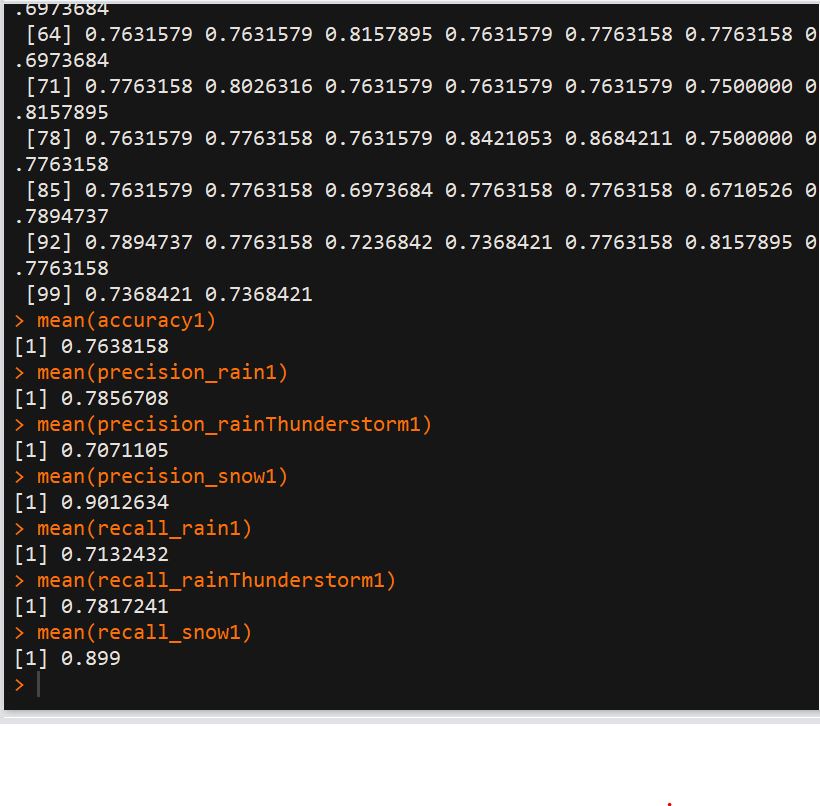
SVM Radial Kernel Results



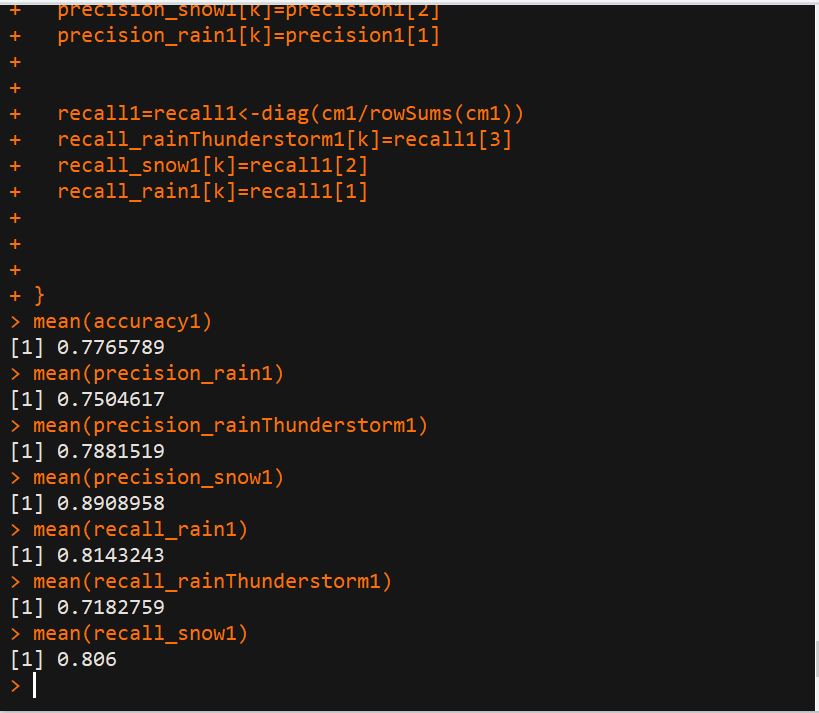
RESULTS:

**I also changed the tuning parameters under SVM tuning to see the best results. I have used Kernels like linear, radial and sigmoid with different cost and gamma parameters. We see some differences.**

**SVM Linear Results**



SVM RADIAL WITH GAMMA =0 AND COST =1 RESULTS



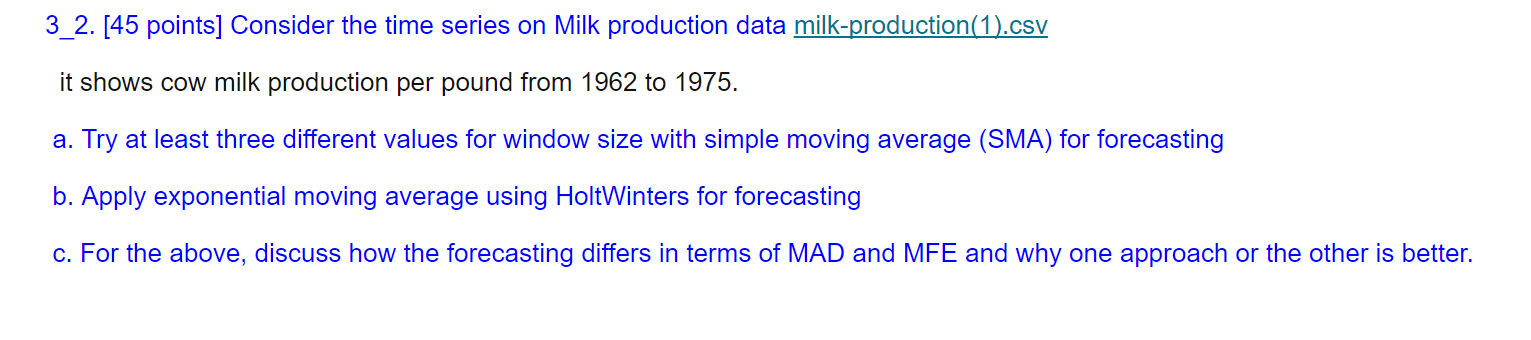
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Tuning Parameters | Accuracy | Precision Snow | Precision Rain | Precision Rain Thunderstorm | Recall Snow | Recall Rain | Recall ThunderStorm |
| SVM | kernel="radial”,cost=1, gamma=0.04545455,coef.0=0,epsilon=0.1 | 0.7736842 | 0.8981612 | 0.7938852 | 0.7231856 | 0.887 | 0.7278378 | 0.7931034 |
| SVM | kernel="linear" | 0.7638158 | 0.9012634 | 0.7856708 | 0.7071105 | 0.899 | 0.7132432 | 0.7817241 |
| SVM | kernel="radial",cost=1,gamma=0 | 0.7765789 | 0.8908958 | 0.7504617 | 0.7881519 | 0.806 | 0.8143243 | 0.7182759 |
| SVM | kernel="sigmoid”,cost=1, gamma=0.04545455,coef.0=0,epsilon=0.1 | 0.7515789 | 0.8825927 | 0.7829367 | 0.6924296 | 0.868 | 0.68 | 0.8027586 |

**Comparing to the other models using in Assignment 2**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Accuracy | Precision Snow | Precision Rain | Precision Rain Thunderstorm | Recall Snow | Recall Rain | Recall ThunderStorm |
| LDA | 0.9026316 | 0.6407459 | 0.9115871 | 0.9906168 | 0.911675 | 0.902705 | 0.9906152 |
| QDA | 0.7389474 | 0.945 | 0.697027 | 0.7213793 | 0.7950919 | 0.7514844 | 0.7115056 |
| KNN (K=5) | 0.745 | 0.895 | 0.7305405 | 0.7117241 | 0.9098042 | 0.7444542 | 0.701065 |

**Discussion Note:**

1. From Accuracy Results, we see that SVM model performs better than QDA and KNN, but LDA outperforms SVM too.
2. From Precision of Snow Results, SVM does better than LDA and KNN
3. From Precision of Rain Results, SVM does better than QDA and KNN, but less than LDA
4. From Precision of thunderstorm Results, SVM does better than QDA and KNN, but less than LDA.
5. From Recall of Snow Results, SVM does better than QDA and KNN
6. From Recall of Rain Results, SVM does better than QDA and KNN, but less than LDA
7. From Recall of thunderstorm Results, SVM does better than QDA and KNN, but less than LDA.
8. So, overall if we compare performance with respect to classifiers, LDA>SVM>KNN>QDA.

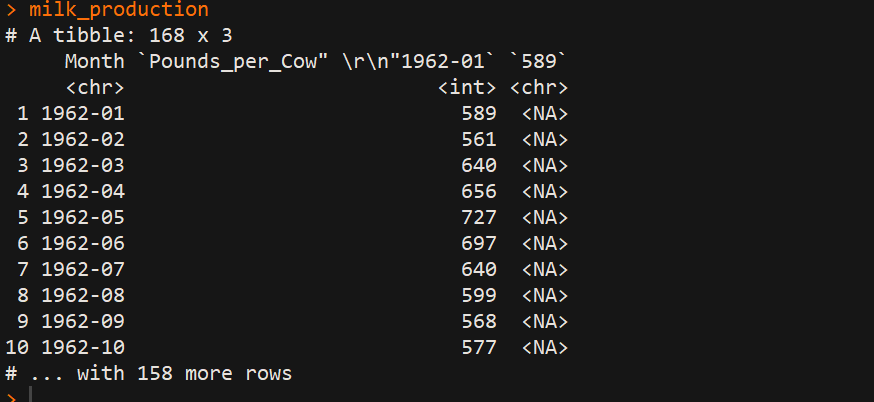


**PROGRAM**

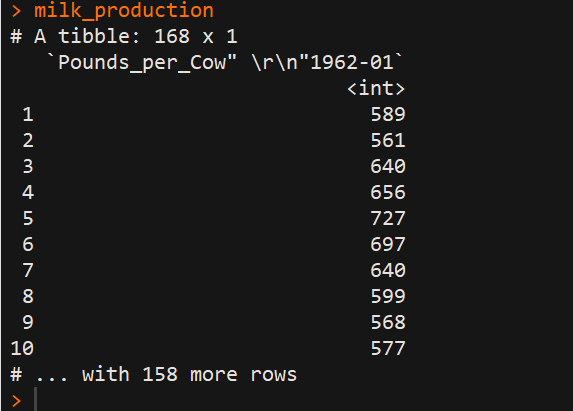
*#import the dataset and make some changes*

library(readr)

milk\_production <- read\_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/assignment3/milk-production(1).csv")



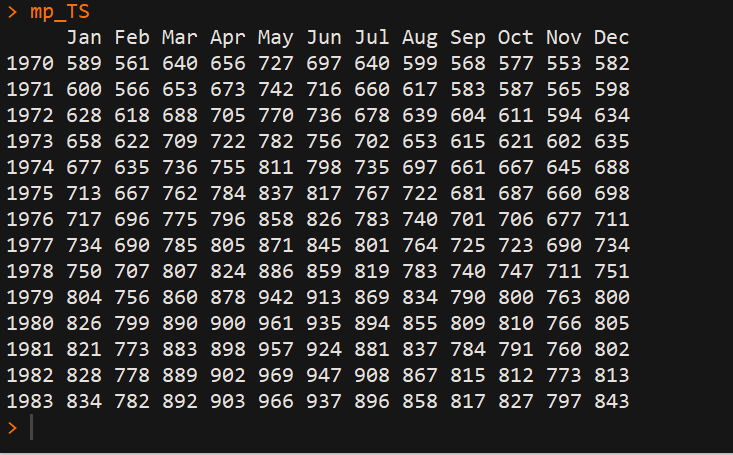
milk\_production<-milk\_production[,2]



milk\_production\_timeseries<-ts(milk\_production)

*#contains monthly milk productions for January 1970-Decemeber 1983*

mp\_TS<-ts(milk\_production,frequency = 12,start=c(1970,1))

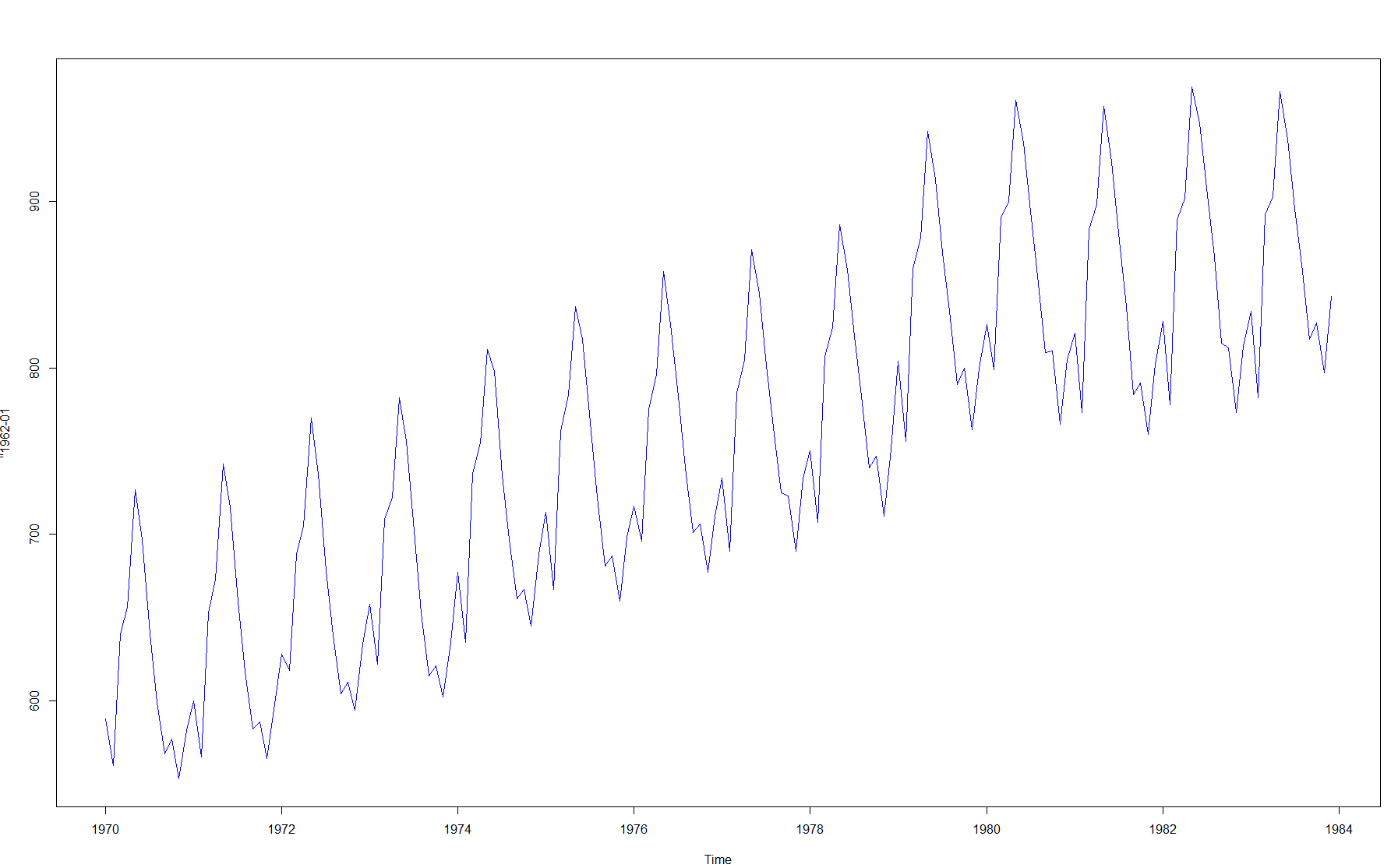


*#plotting time series*

plot.ts(milk\_production\_TS)

lines(mp\_TS,col="blue")

**Plot of Timeseries**



**\*\*Question #2 – Part A\*\***

*#\*\*\*Simple Moving Average(SMA)\*\*\*# ->it is used to smooth time series data*

*#install.packages("TTR")*

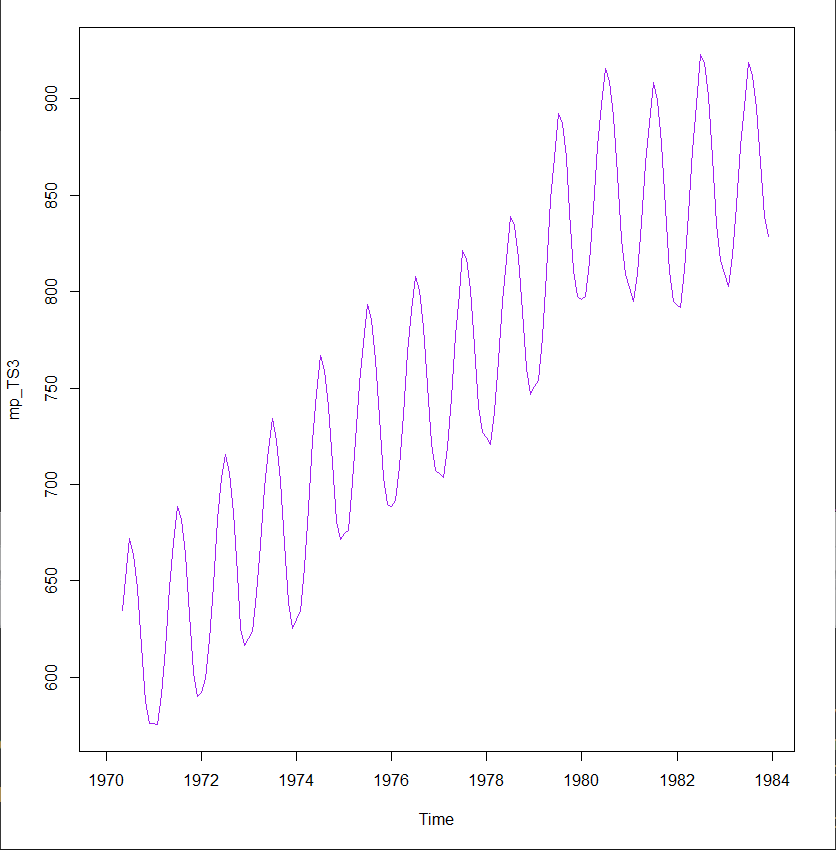
library("TTR")

mp\_TS5<-SMA(milk\_production\_TS,n=5)

plot.ts(mp\_TS5)

lines(mp\_TS5,col="purple")

Simple Moving Average: Window size 5

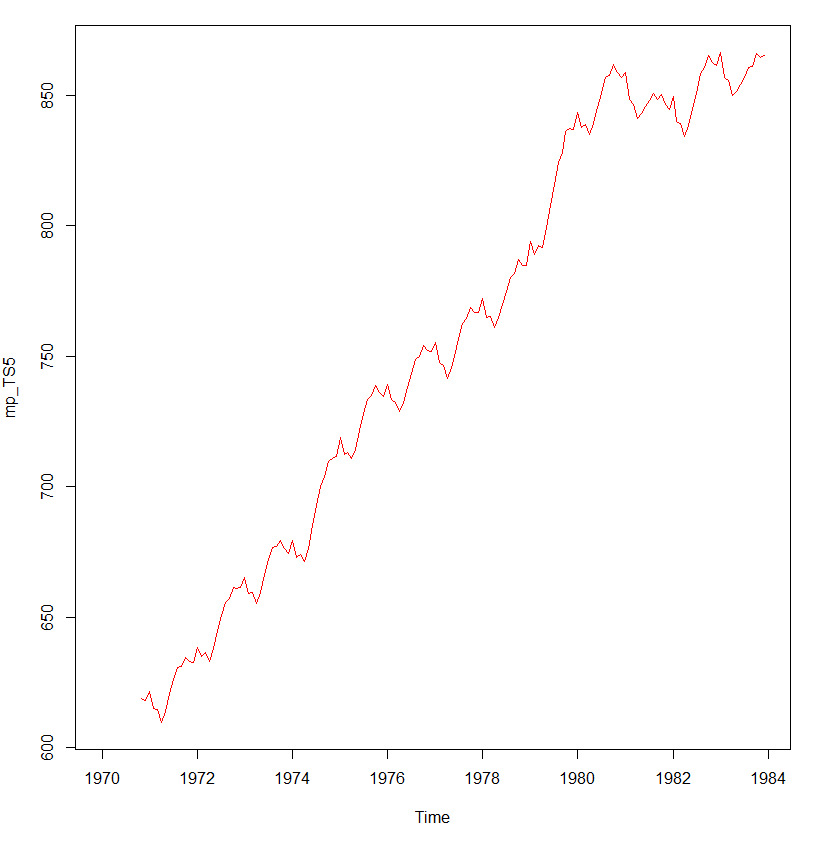


mp\_TS5<-SMA(milk\_production\_TS,n=11)

plot.ts(mp\_TS11)

lines(mp\_TS11,col="red")

Simple Moving Average: Window size 11

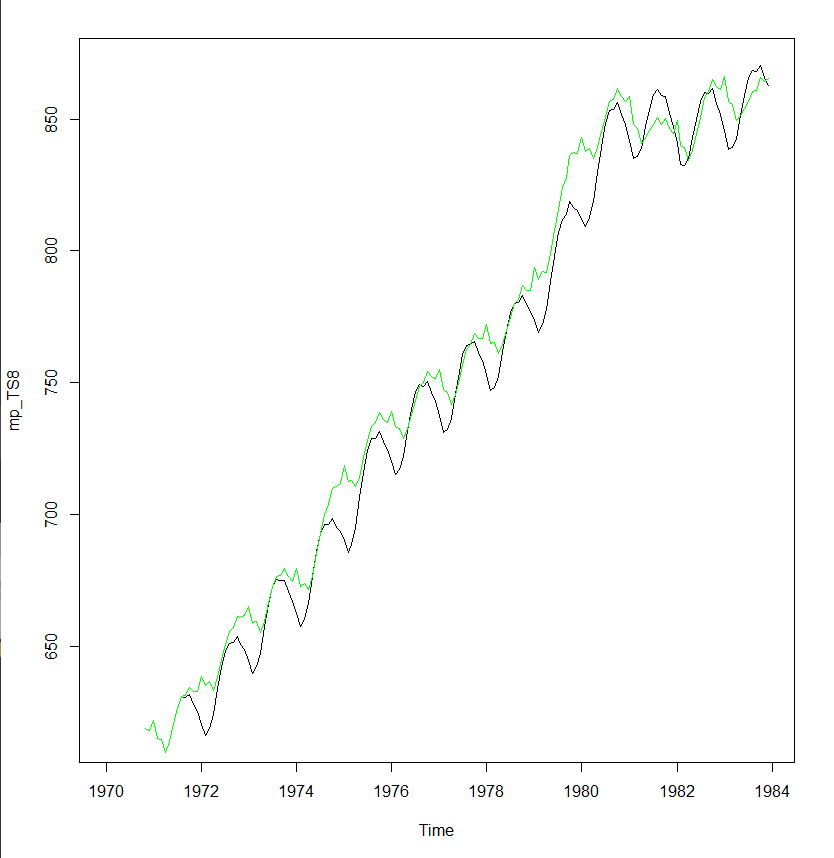


mp\_TS20<-SMA(milk\_production\_TS,n=20)

plot.ts(mp\_TS20)

lines(mp\_TS20,col="green")

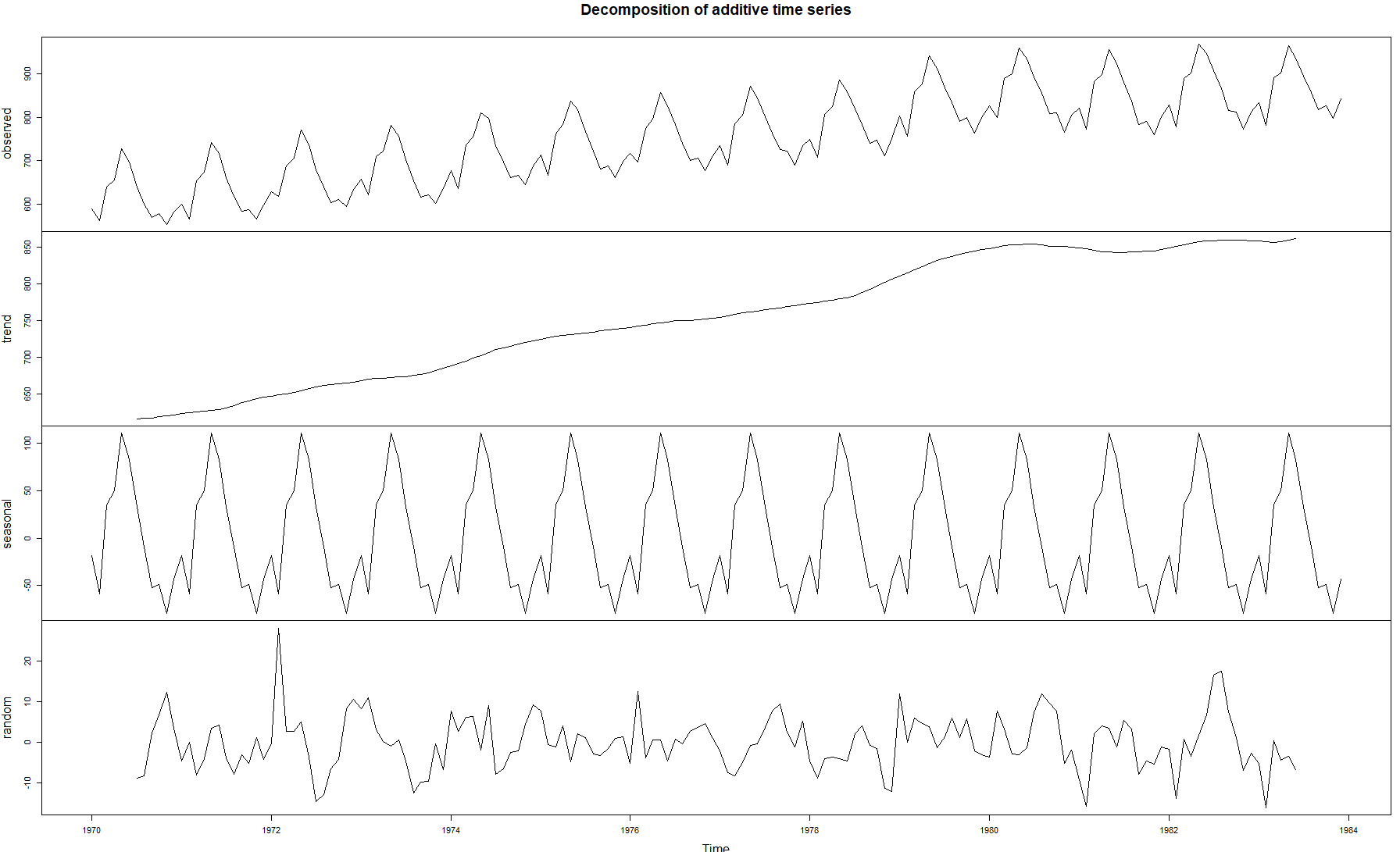
Simple Moving Average: Window size 20



*#To estimate the trend component and seasonal component of a seasonal time series that can be described using an additive model, we can use the “decompose ()” function in R. This function estimates the trend, seasonal, and irregular components of a time series that can be described using an additive model###*

mp\_decompose=decompose(milk\_production\_TS)

plot(mp\_decompose) *#The plot above shows the original time series (top), the estimated trend component (second from top), the estimated seasonal component (third from top), and the estimated irregular component (bottom)##*



**Discussion and observations:**

* The moving average model uses the last t periods to predict demand in period t+1.
* SMA is an arithmetic moving average calculated by adding the actual forecasts for many time periods and then dividing this total by the number of time periods.
* In the above program, simple moving average of milk-production dataset has been calculated, with three different windows, 5,11,20. I chose this window sizes because,
* When window size is 5, the graph is changing, but we can see that it is changing in in equally distributed patterns. Whereas when window size is 11, the graph is increasing but with some non-linearity. When the window size is 20, the graph is almost linearly increasing.

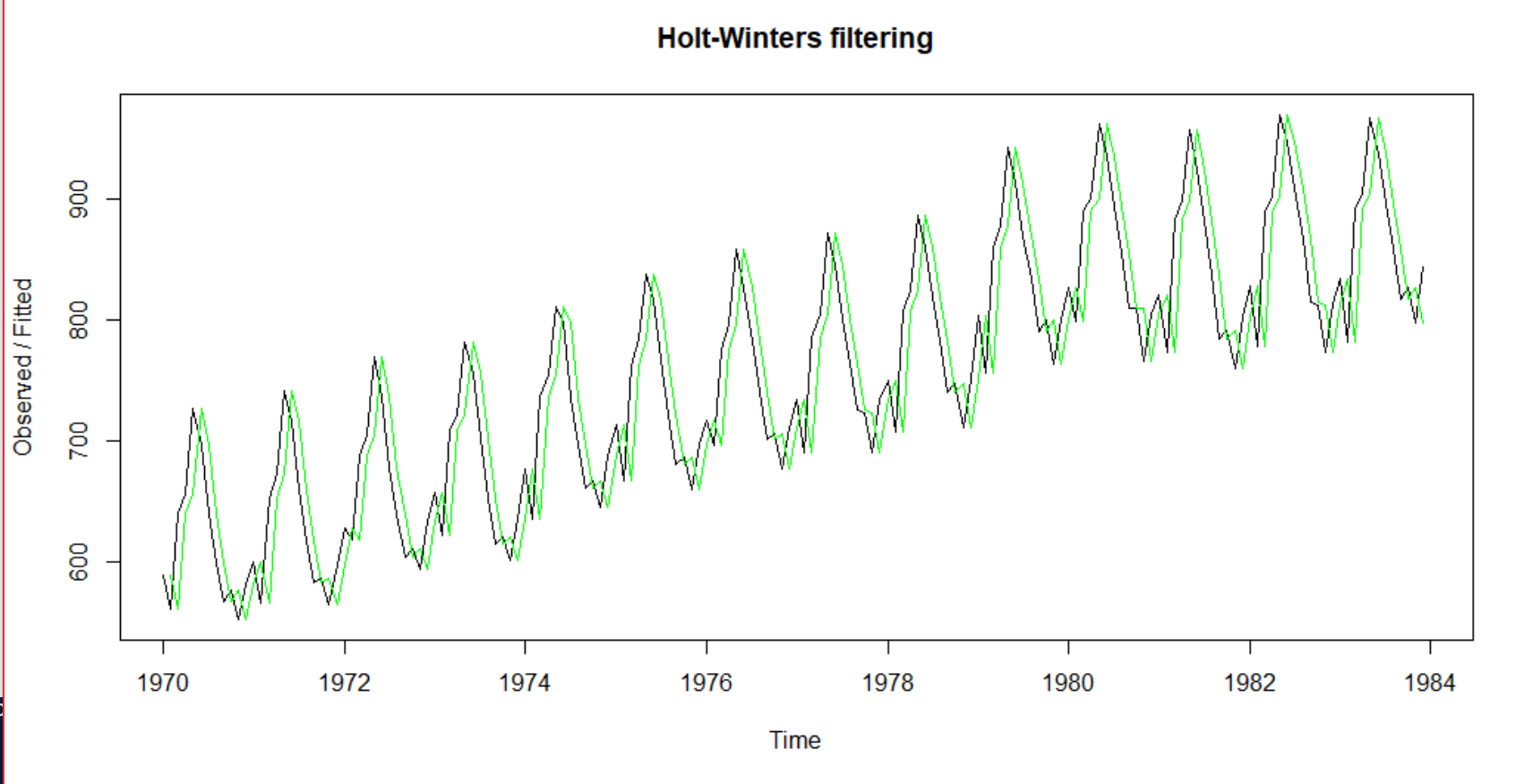
**\*\*\*Question #2 – Part B\*\*\***

*#\*\*\*\*Forecasts suing Exponential Smoothing\*\*\*###*

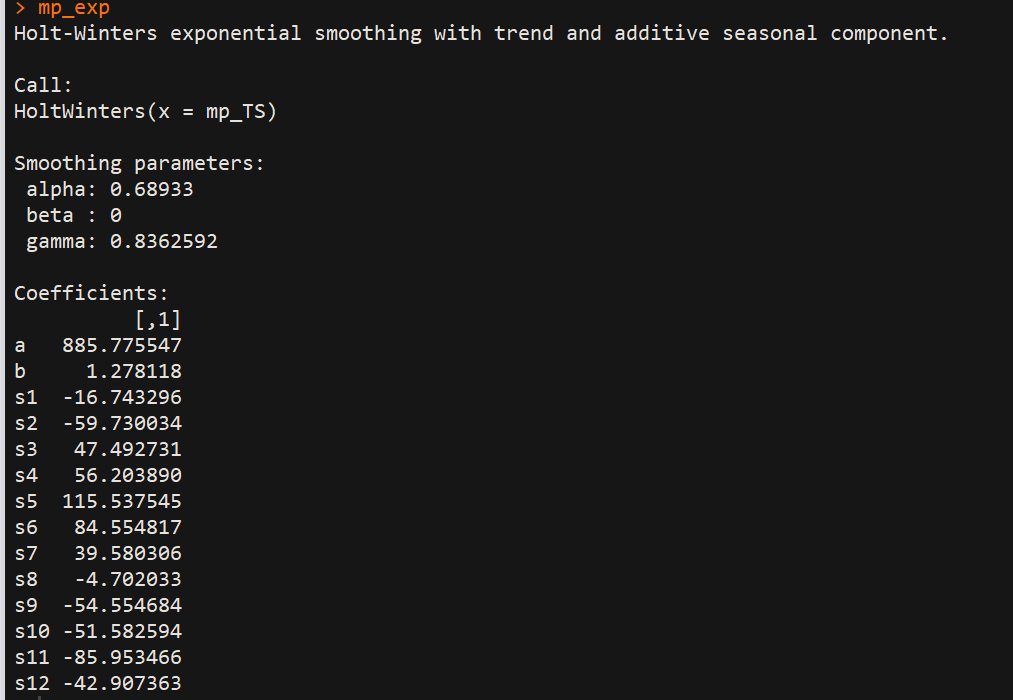
mp\_exp=HoltWinters(mp\_TS,beta=FALSE,gamma = FALSE)

plot(mp\_exp) *#The plot shows the original time series in black, and the forecasts as a red line. The time series of forecasts is much smoother than the time series of the original data here.*

lines(mp\_exp$fitted[,1],col="red")



mp\_exp

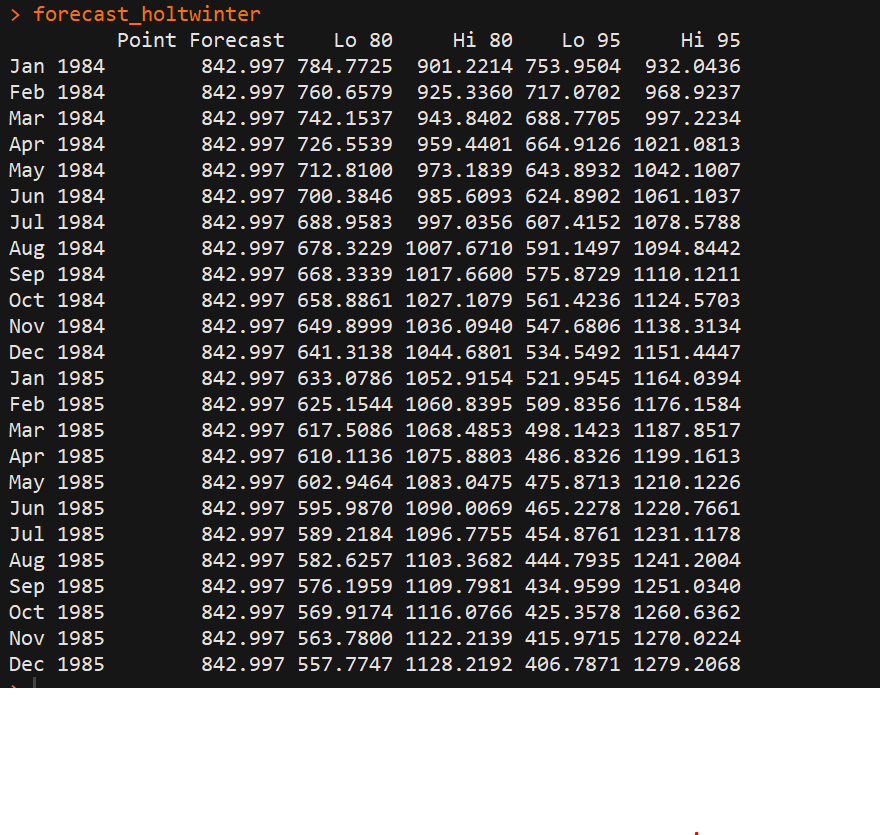


*#install.packages("TTR")*

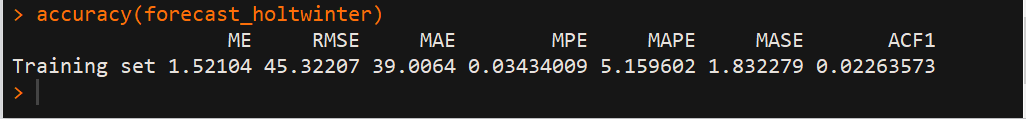
library(forecast)

forecast\_holtwinter=forecast(mp\_exp)

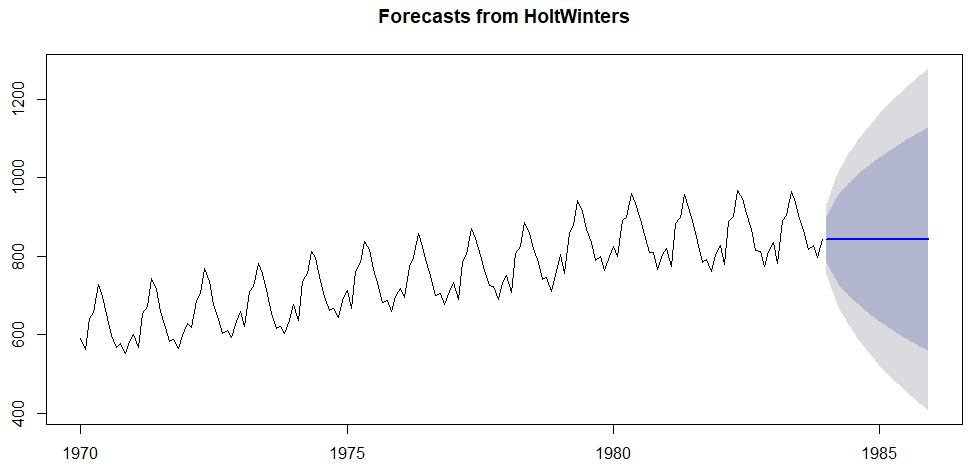
forecast\_holtwinter



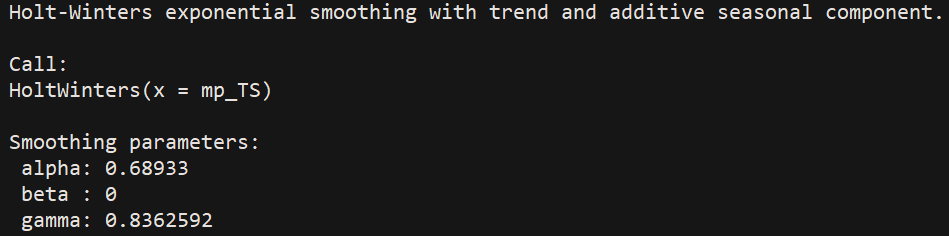
Accuracy(forecast\_holt\_winters)



plot(forecast\_holt\_winters)



**Discussion and observations:**

* The main idea is an exponential smoothing is that the prediction mostly depends on most recent observation and on the error of the latest forecast.
* If the time series can be described using an additive model with increasing or decreasing trend and seasonality, Holt-Winters exponential smoothing to make short-term forecasts.
* Smoothing is controlled by three parameters: alpha, beta, and gamma, for the estimates of the level, slope b of the trend component, and the seasonal component, respectively, at the current time point. The coefficients alpha, beta and gamma, usually ranges between 0 and 1.
* 
* In the above forecast, α is 0.68933 indicating that the estimate of the level at the current time is based upon observations in the more distant past as well as some recent observations.
* The value of beta is 0, which indicates that the estimate of the slope b of the trend component is not updated over the time series, and instead is set equal to its initial value. This makes a conclusion that as the level changes over the time series, slope b of the trend component remains almost constant.
* In contrast, the value of gamma (0.8362592) is high, indicating that the estimate of the seasonal component at the current time point is just based upon very recent observations.

**\*\*\*Question #2 – Part C\*\***

## Discuss how the forecasting differs in terms of MAD and MFE and##

##why one approach or the other is better##

count=168

x=mean(mp\_TS3[-(1:7)])

for (k in 8:count) {

mean\_abs\_dev3=mean(abs(mp\_TS3[k]-x))

mfe3=mean(mp\_TS3[k]-x)

mad3=mad(mp\_TS3[k], centre, constant = 1.4826, na.rm = FALSE, low = FALSE, high = FALSE)

}

y=mean(mp\_TS5[-(1:12)])

for (k in 13:count) {

mean\_abs\_dev4=mean(abs(mp\_TS5[k]-y))

mfe4=mean(mp\_TS5[k]-y)

mad4=mad(mp\_TS5[k], centre, constant = 1.4826, na.rm = FALSE, low = FALSE, high = FALSE)

}

z=mean(mp\_TS8[-(1:23)])

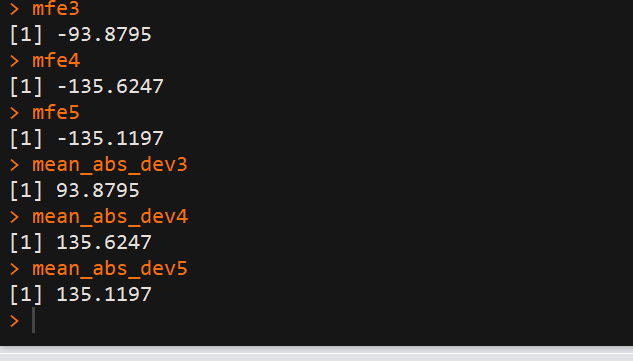
for (k in 24:count) {

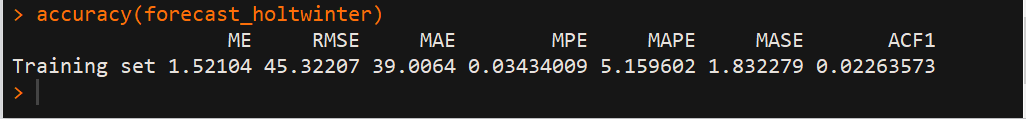
mean\_abs\_dev5=mean(abs(mp\_TS8[k]-z))

mfe5=mean(mp\_TS8[k]-z)

mad(mp\_TS8[k], centre, constant = 1.4826, na.rm = FALSE, low = FALSE, high = FALSE)

}





**Observations:**

* According to the above values of MAD and MFE of Simple moving average (three different windows) and MAE of Holt-winter forecasting, I find Holt-winters exponential smoothing and forecasting more precise than SMA.