Time Series for NCEI

Ahmet Kilinc

2023-11-14

veriyi kaggle'den aldım, Londradaki heathrow havalimanına aittir,her gun icin yagis ve ortalama gunluk sicakliklari icerir. verinin kaynağı : https://www.kaggle.com/emafuma/ncei-heathrow-2010-2019

```
options(warn=-1)
library(readr)
data <- read_csv("C:/Users/Ahmet/Desktop/Zaman serileri analizi/NCEI Heathrow Meteo Data 2010-2019.csv")
```

```
## Rows: 3621 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (2): PRCP, TAVG
## date (1): DATE
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
View(data)
```

DATE: Veri setimizdeki ilgili tarih. TAVG: Veri setimizdeki ilgili ortalama sicaklik. PRCP: Veri setimizdeki yagis miktari.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(fpp2)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## -- Attaching packages ---------------------------------------------- fpp2 2.5 --
```

```
## v ggplot2   3.4.2      v fma       2.5
## v forecast  8.21.1     v expsmooth 2.3
```

```
##
```

veri setimizdeki date ve tavg degiskenini secerek, ilk 10 gozleme bakalim

```
data<-data%>%select(c("DATE","TAVG"))
head(data,10)
```

```
## # A tibble: 10 x 2
##    DATE        TAVG
##    <date>     <dbl>
##  1 2010-01-01   0.8
##  2 2010-01-02   1.8
##  3 2010-01-03   0.4
##  4 2010-01-04  -2.8
##  5 2010-01-05  -1.3
##  6 2010-01-06  -0.1
##  7 2010-01-07  -2.3
##  8 2010-01-08  -1.4
##  9 2010-01-09  -1.2
## 10 2010-01-10   1.4
```

verimizin ozetine bakalim

```
summary(data)
```

```
##       DATE                 TAVG
##  Min.   :2010-01-01   Min.   :-4.10
##  1st Qu.:2012-06-24   1st Qu.: 7.40
##  Median :2014-12-16   Median :11.60
##  Mean   :2014-12-26   Mean   :11.64
##  3rd Qu.:2017-06-29   3rd Qu.:16.00
##  Max.   :2019-12-31   Max.   :28.60
```

group_by komutu ile verimizin ilk sutunu olan "DATE" sutununu ay ve yil olarak ayirip verimize ayri sutunlar seklinde ekleyelim ve aylik veri olarak tanimlayalim.

```
data$Month<-lubridate::month(data$DATE)
data$Year<-lubridate::year(data$DATE)
datamonthly <- data%>%group_by(Year,Month)%>%summarise(TAVG = mean(TAVG))
```

```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

```
head(datamonthly,10)
```

```
## # A tibble: 10 x 3
## # Groups:   Year [1]
##     Year Month  TAVG
##    <dbl> <dbl> <dbl>
##  1  2010     1  2.18
##  2  2010     2  3.99
##  3  2010     3  7.06
##  4  2010     4 10.4
##  5  2010     5 12.3
##  6  2010     6 17.6
##  7  2010     7 19.4
##  8  2010     8 16.9
##  9  2010     9 14.8
## 10  2010    10 11.4
```

yeni olusturdugumuz aylik verimizin özet haline bakalim

```
summary(datamonthly)
```

```
##       Year          Month            TAVG
##  Min.   :2010   Min.   : 1.00   Min.   : 1.429
##  1st Qu.:2012   1st Qu.: 3.75   1st Qu.: 7.108
##  Median :2014   Median : 6.50   Median :11.519
##  Mean   :2014   Mean   : 6.50   Mean   :11.594
##  3rd Qu.:2017   3rd Qu.: 9.25   3rd Qu.:16.017
##  Max.   :2019   Max.   :12.00   Max.   :21.803
```

verimizi ts komutuyla zaman serisine cevirelim, aylik verinin frekansi 12 olarak alinir.

```
datamonthly<-ts(datamonthly[,3],start=c(2010,1),frequency =12)
datamonthly
```

```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2010  2.183871  3.989286  7.064516 10.410000 12.312903 17.586667 19.448387
## 2011  5.083871  7.339286  7.625806 13.576667 13.874194 15.533333 16.864516
## 2012  6.593548  4.734483  9.177419  8.610000 13.619355 15.023333 16.916129
## 2013  4.551613  3.764286  3.819355  8.593333 11.777419 15.436667 20.580645
## 2014  7.006452  7.453571  8.948387 11.600000 13.687097 17.166667 19.912903
## 2015  5.516129  4.896429  7.625806  9.200000 12.793548 16.416667 18.309677
## 2016  6.448387  6.110345  6.819355  8.826667 13.935484 16.150000 18.729032
## 2017  4.003226  7.103571 10.058065 10.696667 14.512903 18.606667 18.870968
## 2018  6.674194  3.557143  6.045161 11.320000 14.964516 18.013333 21.803226
## 2019  4.977419  7.571429  9.345161 10.246667 13.100000 16.396667 19.987097
##            Aug       Sep       Oct       Nov       Dec
## 2010 16.877419 14.770000 11.438710  6.466667  1.429032
## 2011 16.948387 16.186667 13.622581 10.280000  7.019355
## 2012 18.325806 14.663333 10.929032  7.670000  5.880645
## 2013 18.851613 15.143333 13.445161  7.490000  7.109677
## 2014 16.529032 16.630000 13.829032  9.653333  6.196774
## 2015 17.796774 13.893333 12.112903 10.830000 11.374194
## 2016 19.096774 17.476667 11.619355  6.973333  6.883871
## 2017 17.303226 14.630000 13.538710  7.963333  5.990323
## 2018 18.777419 15.280000 11.974194  9.700000  7.941935
## 2019 19.041935 15.973333 11.722581  7.383333  7.077419
```

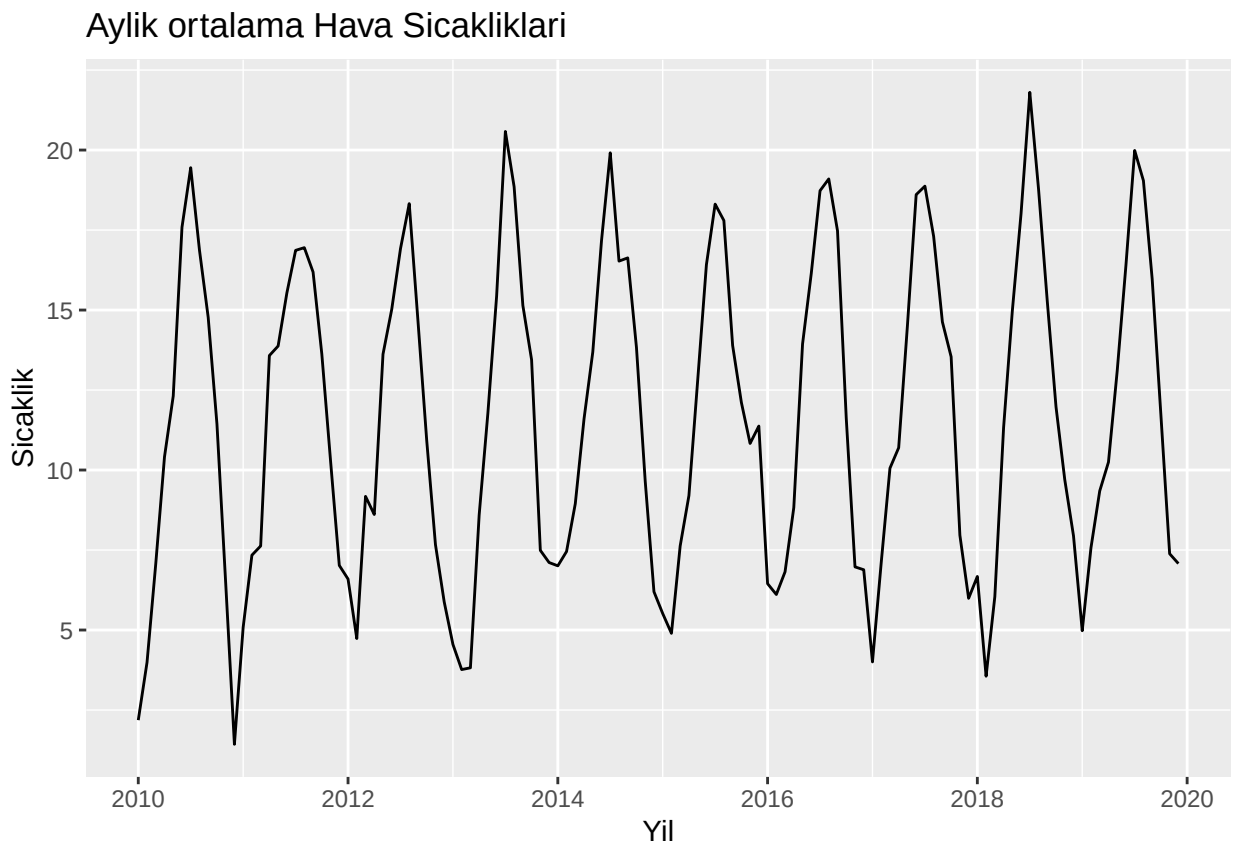
ts komutuyla verimiz zaman serisine cevrilmis mi class'ına bakarak test edelim

```
class(datamonthly)
```

```
## [1] "ts"
```

zaman serisi grafigi:

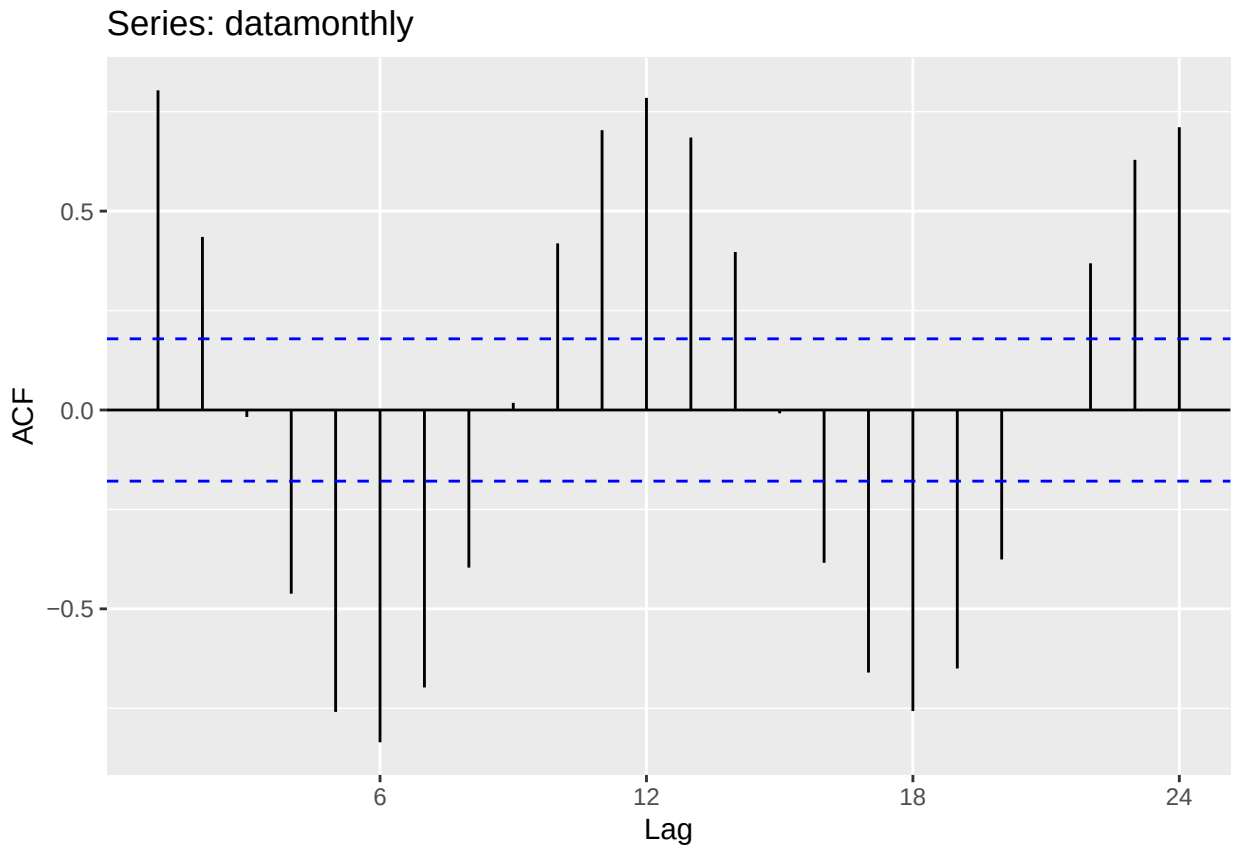
```
autoplot(datamonthly) +
  ggtitle("Aylik ortalama Hava Sicakliklari") +
  xlab("Yil") +
  ylab("Sicaklik")
```



-Grafige gore mevsimsellik vardir fakat trendle ilgili kesin bir sey soyleyemeyiz.

otokolerasyonu inceleme:

```
ggAcf(datamonthly)
```



tum lagler mavi cizgiyi gectigi icin otokolerasyon vardır deriz birinci lag bize orijinal seriyle gecikmeli arasindaki otokolerasyonu gosterir ve cok yuksek gelmistir altinci lag ise ocak-haziran ın denk geldigi yerdir ve negatif otokolerasyon vardir deriz cunku kisla yaz birbirine denk gelmektedir.

```
length(datamonthly)/5
```

```
## [1] 24
```

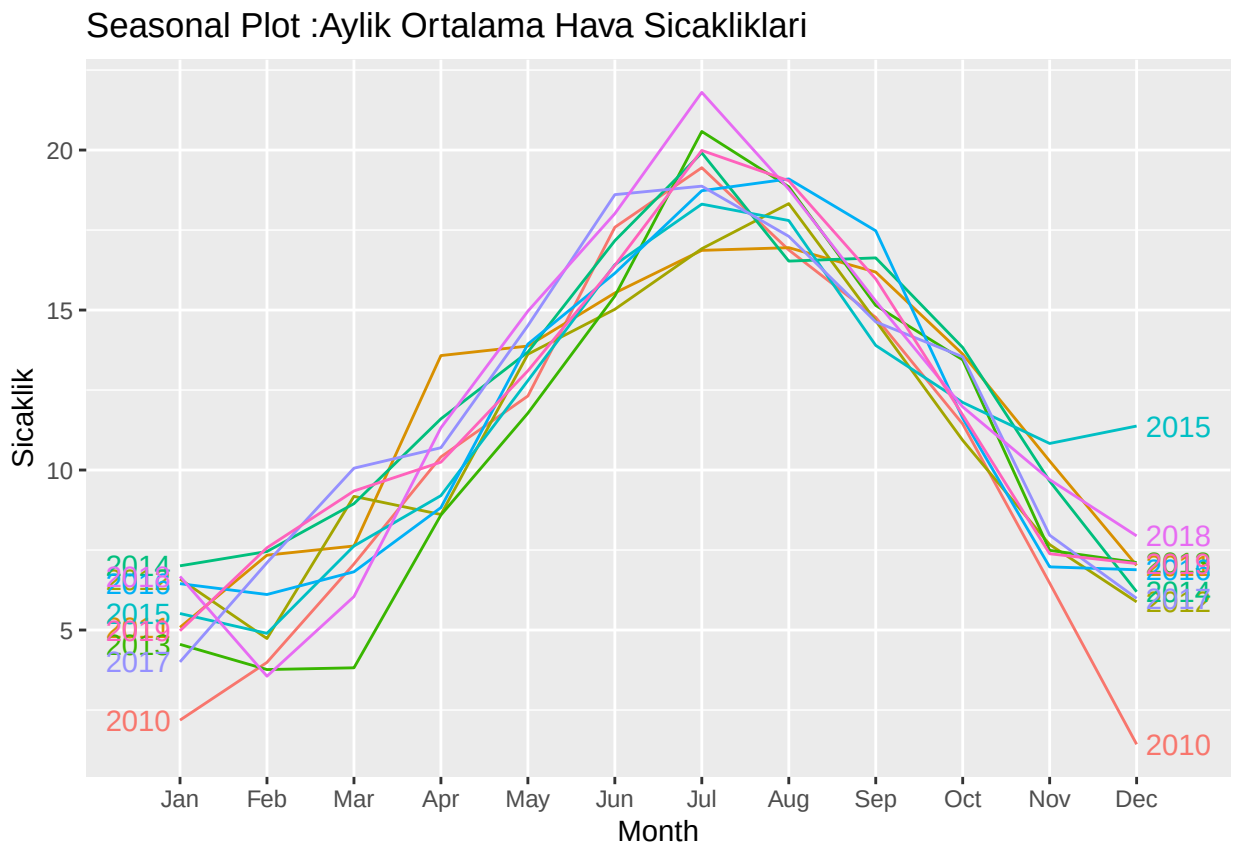
```
Box.test(datamonthly, lag=24, type = "Lj", fit=0)
```

```
##
##  Box-Ljung test
##
## data:  datamonthly
## X-squared = 1039.7, df = 24, p-value < 2.2e-16
```

H0; otokolerasyon yoktur , H1;otokolerasyon vardir p degerim 0.05' ten kucuk geldigi icin H0 red edilir, anlamli otokolerasyon vardir deriz.

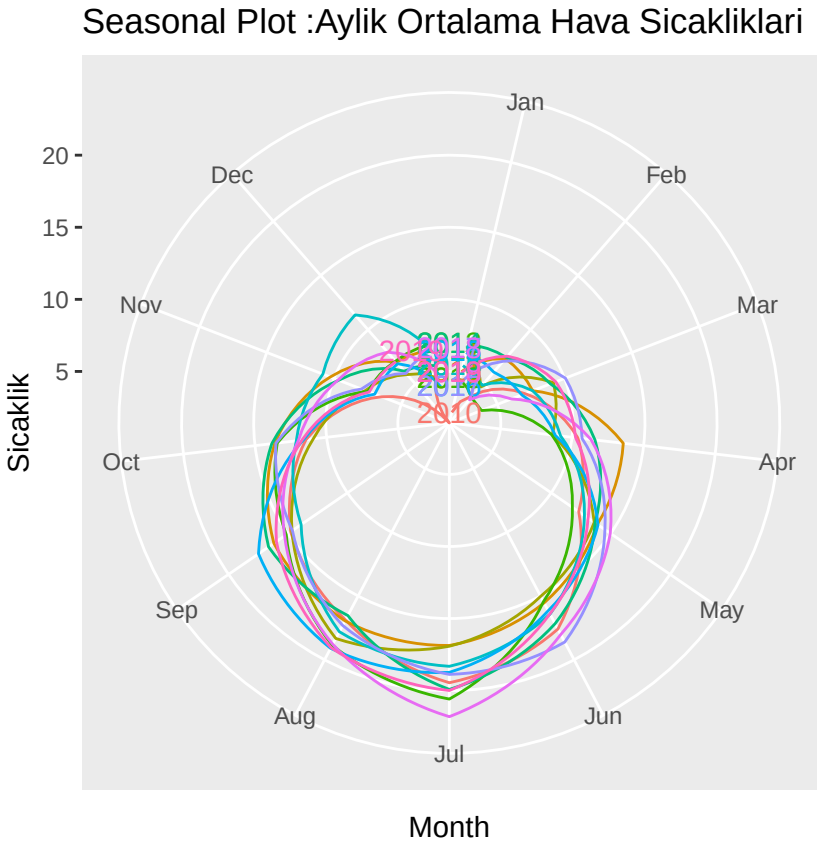
mevsimsellik var midir?:

```
ggseasonplot(datamonthly, year.labels=TRUE, year.labels.left=TRUE) +
  ylab("Sicaklik") +
  ggtitle("Seasonal Plot :Aylik Ortalama Hava Sicakliklari")
```



seasonplot grafigine bakarak her sene ayni hareketler oldugunu gormekteyiz bu da bize guclu mevsimsellik oldugunu gostermektedir.Her sene basi dusuk baslayip, sene ortasında yukselmis ve sene sonuna dogru sicaklik dususe gecmis.

```
ggseasonplot(datamonthly,polar=TRUE, year.labels=TRUE, year.labels.left=TRUE) +
  ylab("Sicaklik") +
  ggtitle("Seasonal Plot :Aylik Ortalama Hava Sicakliklari")
```



bu grafik de aynı sekilde her senede ayni hareket oldugu goruluyor bundan dolayi guclu mevsimsellik vardir deriz.

verimizi test ve train olarak ayiralim ve egitim seti uzerinden ortalama ve uygun gorulen naive modeli kuralim

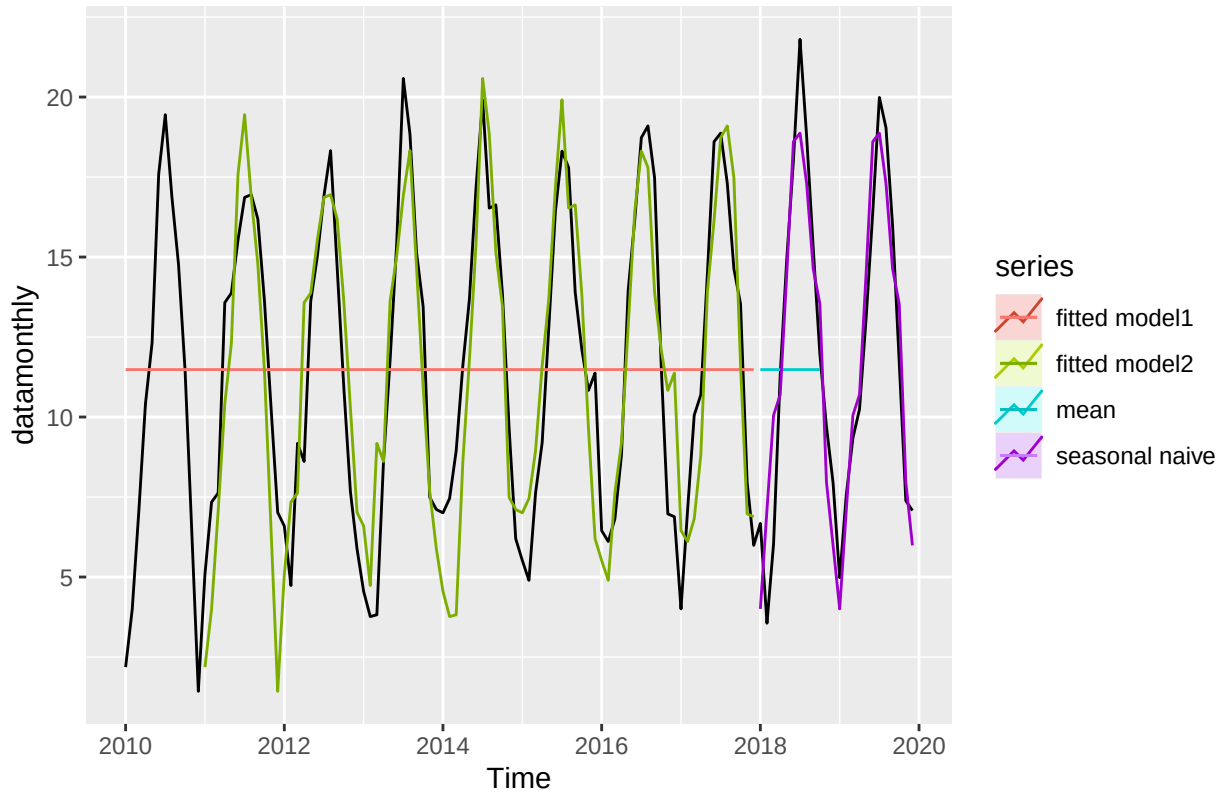
```
train<-window(datamonthly,end=c(2017,12))
test<-window(datamonthly, start=2018)
```

egitim seti uzerinden ortalama ve naive modeli kuralim.

```
model1<-meanf(train)
model2<-snaive(train)
```

Kurduğumuz modellerin test seti ve train seti tahmin performanslarini grafik uzerinde gosterelim

```
autoplot(datamonthly) +
  autolayer(model1, series = "mean", PI=FALSE) +
  autolayer(model2, series = "seasonal naive", PI=FALSE) +
  autolayer(fitted(model1), series = "fitted model1") +
  autolayer(fitted(model2), series = "fitted model2")
```



kurduğumuz modellerin test seti ve training set üzerindeki rmse değerlerini bulalim

```
accuracy(model1, test)
```

```
##                        ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -5.090645e-16 4.883399 4.278124 -31.92409 57.35382 2.284952
## Test set      1.357961e+00 5.870962 5.025027 -21.16053 56.09079 2.683874
##                    ACF1 Theil's U
## Training set 0.7956299        NA
## Test set     0.7674011  1.602166
```

mean ile olusturdugumuz model1 icin:

training set rmse degeri –> 4.883399

test set rmse degerimiz –> 5.870962

```
accuracy(model2, test)
```

```
##                       ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 0.22976428 2.302965 1.872304 -0.6553221 21.28249 1.0000000
## Test set     0.09661866 1.789722 1.504878 -2.5525627 17.62016 0.8037576
##                    ACF1 Theil's U
## Training set 0.4247751        NA
## Test set     0.1499386 0.8142866
```

seasonal naive ile olusturdugumuz model2 icin:

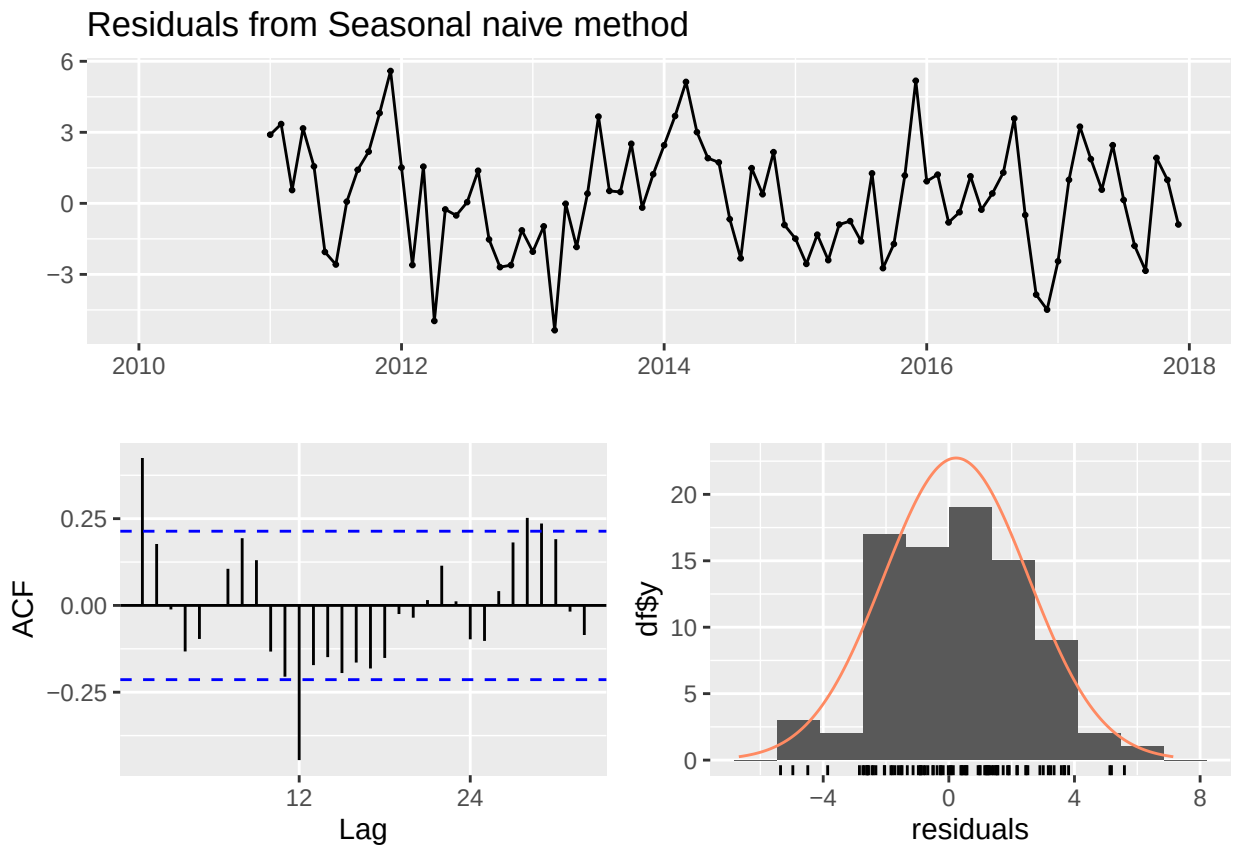
training set rmse degeri –> 2.302965

test set rmse degeri –> 1.789722

RMSE ye ve diger metriklere baktigimizda en dusuk degerli sonuclari aldigimiz modelimiz Seasonal Naive yontemi kullanilarak olusturdugumuz model2 dir.

kurduğumuz iki modelden test seti forcasting performansı daha iyi olan için model varsayımlarını kontrol edelim (normallik ve otokolerasyon)

```
checkresiduals(model2)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 71.267, df = 19, p-value = 5.656e-08
##
## Model df: 0.   Total lags used: 19
```

1. Grafige baktigimizda artiklar 0 etrafinda rastgele dagilmaktadir.Bu grafikte trend gozlenmemektedir.
2. Grafige baktigimizda laglerden bazilari mavi sinir cizgisini gectigi icin artiklarin otokorelasyon problemi vardir. Her bir gecikmeli icin ayri ayri hipotez testi yapilmalidir.Portmanteau Testleri olan Box-Pierce ve Ljung-Box ile otokorelasyon kontrolu yapariz.
3. Grafik Artiklarin Normal Dagilim Grafigidir.Grafige baktigimizda artiklar normal dagiliyor gibi gozukmemektedir, normallik testi grafigimizden daha guvenilir oldugu icin normalligi test etmemiz gerekmektedir. Gozlem sayisimiz 50 nin uzerinde oldugu icin Kolmogorov-Smirnov Testi ile normallige bakalim.

Checkresiduals kodumuzdaki Ljung-Box test sonucumuza gore kullanilan toplam gecikme sayisi(lag) 19 cikmistir.

Otokorelasyon icin portmanteau testlerini uygulayalim

H0:Beyaz gurultu serisidir.(Otokorelasyon problemi yoktur.)

H1:Beyaz gurultu serisi degildir.(Otokorelasyon problemi vardir.)

Box-Pierce testini uygulayalim:

```
Box.test(residuals(model2),lag=10, fitdf=0)
```

```
##
##  Box-Pierce test
##
## data:  residuals(model2)
## X-squared = 27.06, df = 10, p-value = 0.002548
```

Box-Pierce testimizin P-value degerimiz 0.05 den kucuk oldugu icin H0 hipotezi red edilir yani seri beyaz gurultu serisi degildir otokorelasyon problemi vardir.

Ljung-Box testini uygulayalim:

```
Box.test(residuals(model2),lag=10, fitdf=0, type="Lj")
```

```
##
##  Box-Ljung test
##
## data:  residuals(model2)
## X-squared = 28.892, df = 10, p-value = 0.001297
```

Ljung-Box testimizde P-value degerimiz 0.05 den kucuk oldugu icin H0 hipotezi red edilir yani seri beyaz gurultu serisi degildir otokorelasyon problemi vardir.

Şimdi normallik varsayimini kontrol edelim...

Normallik varsayimi kontrolu icin Kolmogorov-Smırnov testi uygulayalim :

H0:Artiklarin dagilimi normaldir. H1:Artiklarin dagilimi normal degildir.

```
x<-na.omit(residuals(model2))
ks.test(x,"pnorm", mean=mean(x), sd=sd(x))
```

```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  x
## D = 0.048289, p-value = 0.9843
## alternative hypothesis: two-sided
```

P-value degerimiz 0.9843 cıkmıstır. P-value degeri 0.05 den buyuk oldugu icin H0 hipotezi reddedilmez yani artiklarin dagilimi normaldir.

test seti forcasting performansı daha iyi olan modeli uygun lambda değerini belirleyip box-cox dönüşümü yaparak kuralım, bu kurduğumuz modelin test seti rmse değerine bakalim

```
lambda <- BoxCox.lambda(train)
model3 <- snaive(train, lambda = lambda, h = 24, biasadj = TRUE)
```

```
accuracy(model3, test)
```

```
##                    ME     RMSE      MAE      MPE     MAPE     MASE     ACF1
## Training set      NaN      NaN      NaN      NaN      NaN      NaN       NA
## Test set     1.171018 3.243112 2.289652 14.72541 30.49984 1.222906 0.158634
##              Theil's U
```

```
## Training set         NA
## Test set      1.171195
```

Box-cox donusumu yaparak kurdugumuz modelin test seti uzerindeki RMSE degeri 3.243112 dir.