Walmart Project

Yuha Yi

December 11, 2018

```
setwd("C:\\Users\\yiyuh\\Documents\\College\\Fall 2018\\Stat 517 - Machine Learning\\Final Project - Sta
#install.packages("reshape")
source('data_prep.R')
```

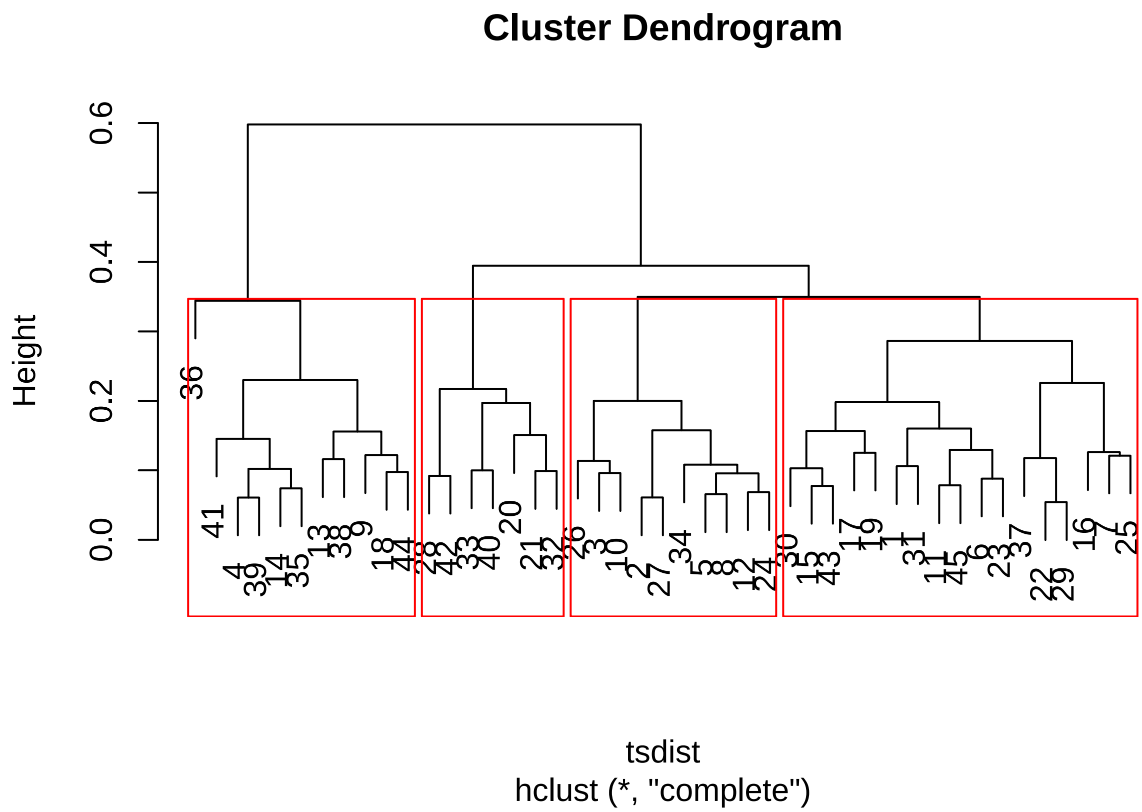
```
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
source('forecast.R')
```

```
##
## Attaching package: 'reshape'
## The following objects are masked from 'package:reshape2':
##
##     colsplit, melt, recast
## The following object is masked from 'package:dplyr':
##
##     rename
## Loading required package: wmtsa
## Loading required package: pdc
## Loading required package: cluster
```

```
source('clustering.R')
train <- read.csv("train.csv")
test <- read.csv("test.csv")
store.matrix <- reshape.by.stores(train)
```

```
#Perform and plot hierarchical clustering based on dissimilarity computation of weekly sales vs stores
tsdist<-calculate.ts.dist(store.matrix)
hc<-hclust(tsdist)
plot(hc)

#Upon visual inspection of the cluster plot, I decide to cluster the data into 4 clusters
rect.hclust(hc,k=4)
```




```r
clust.vec <- cutree(hc,k=4)
clust.vec[hc$order]
```

```
## 36 41  4 39 14 35 13 38  9 18 44 28 42 33 40 20 21 32 26  3 10  2 27 34  5
##  3  3  3  3  3  3  3  3  3  3  3  4  4  4  4  4  4  4  2  2  2  2  2  2  2
##  8 12 24 30 15 43 17 19  1 31 11 45  6 23 37 22 29 16  7 25
##  2  2  2  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
```

```r
#temp remove date column from store matrix
store.matrix.wodate <- store.matrix[,-1]
```

```r
##Creating clusters
cluster1 <- store.matrix.wodate[,clust.vec==1]
cluster2 <- store.matrix.wodate[,clust.vec==2]
cluster3 <- store.matrix.wodate[,clust.vec==3]
cluster4 <- store.matrix.wodate[,clust.vec==4]

##Force clusters in a ts() object
cluster1.ts <-ts(rowMeans(cluster1),frequency=52)
cluster2.ts <-ts(rowMeans(cluster2),frequency=52)
cluster3.ts <-ts(rowMeans(cluster3),frequency=52)
cluster4.ts <-ts(rowMeans(cluster4),frequency=52)
```

```r
### Time Series Forecasting
library(tseries)
#Test for stationarity by performing ADF test
adf.test(cluster1.ts, alternative='stationary') #Dickey-Fuller = -5.279, Lag order = 5, p-value = 0.01
```

```
## Warning in adf.test(cluster1.ts, alternative = "stationary"): p-value
## smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  cluster1.ts
## Dickey-Fuller = -5.279, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(cluster2.ts, alternative='stationary') #Dickey-Fuller = -5.2943, Lag order = 5, p-value = 0.01
```

```
## Warning in adf.test(cluster2.ts, alternative = "stationary"): p-value
## smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  cluster2.ts
## Dickey-Fuller = -5.2943, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

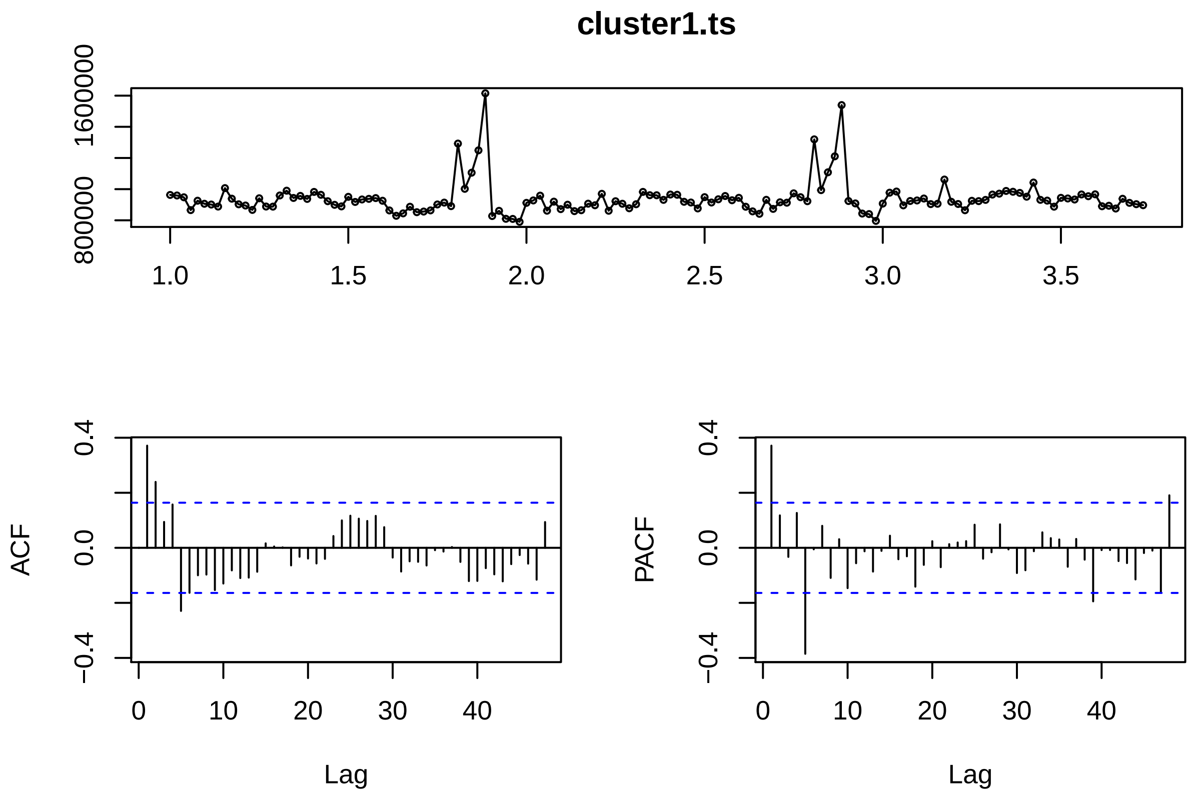
```
adf.test(cluster3.ts, alternative='stationary') #Dickey-Fuller = -5.3377, Lag order = 5, p-value = 0.01
```

```
## Warning in adf.test(cluster3.ts, alternative = "stationary"): p-value
## smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  cluster3.ts
## Dickey-Fuller = -5.3377, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

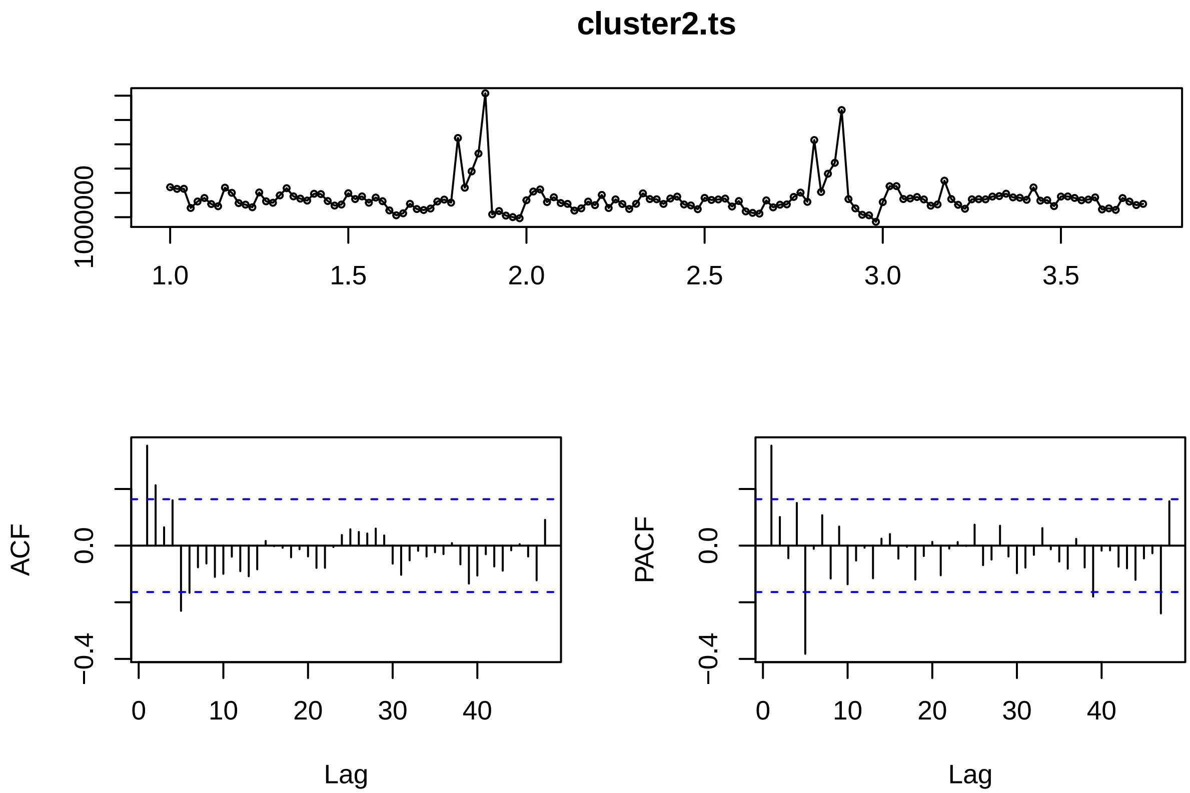
```
adf.test(cluster4.ts, alternative='stationary') #Dickey-Fuller = -5.1801, Lag order = 5, p-value = 0.01
```

```
## Warning in adf.test(cluster4.ts, alternative = "stationary"): p-value
## smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  cluster4.ts
## Dickey-Fuller = -5.1801, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

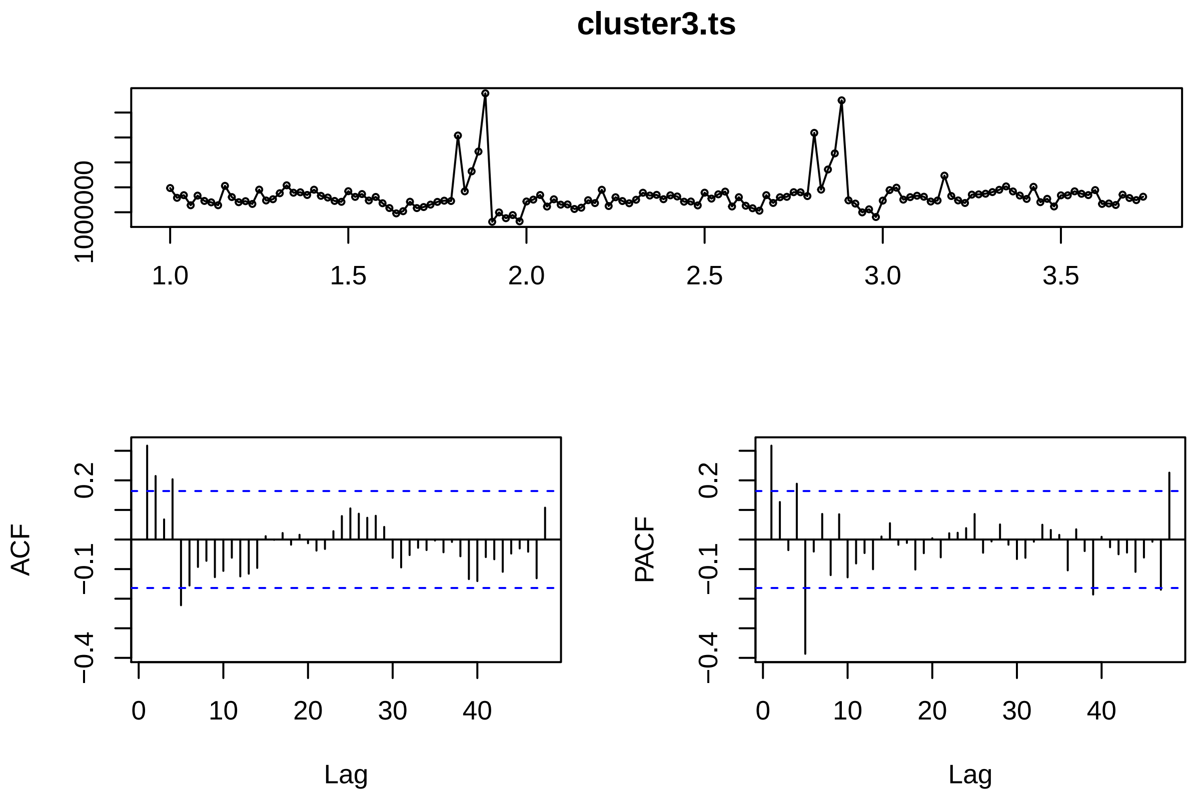
```
#To get an estimate coefficients for AR and MA, plot the ACF and PACF curve for each cluster
#The PACF and ACF lag orders which cross the confidence boundaries, are candidates for AR and MA coeffi
tsdisplay(cluster1.ts)
```



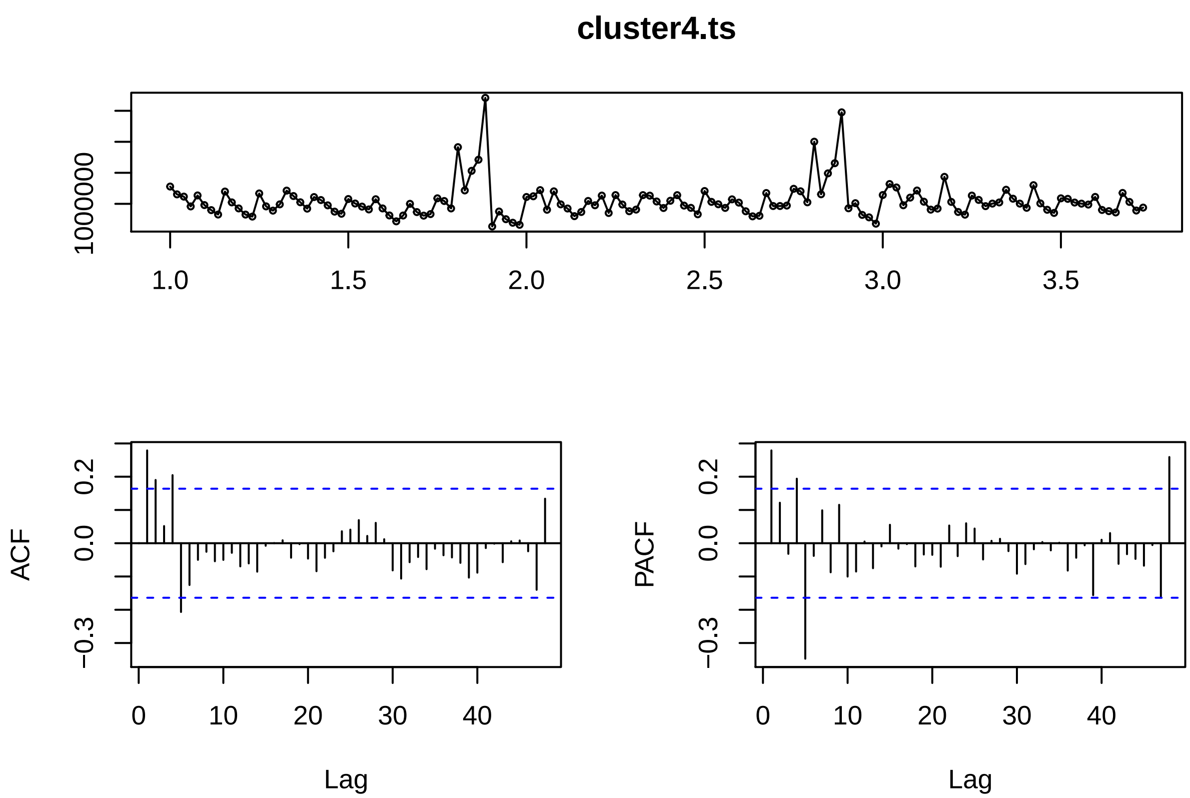

```r
tsdisplay(cluster2.ts)
```




```r
tsdisplay(cluster3.ts)
```

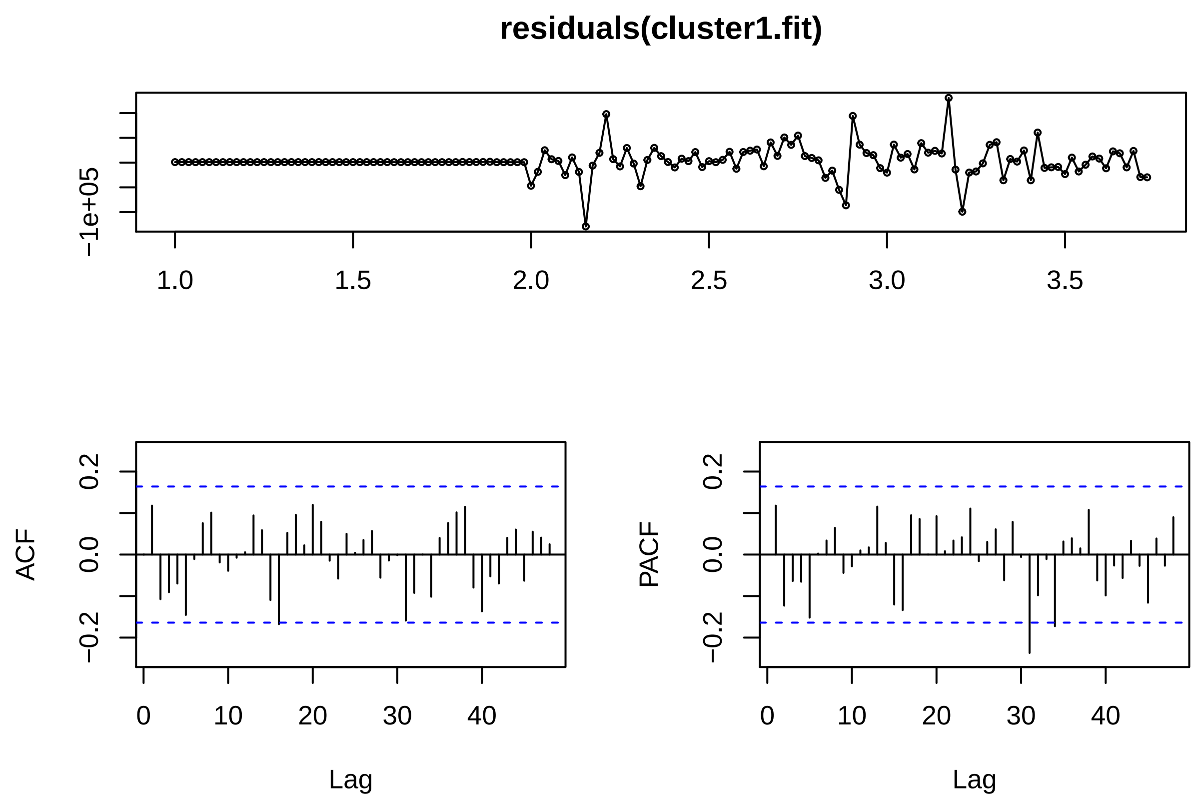



```r
tsdisplay(cluster4.ts)
```

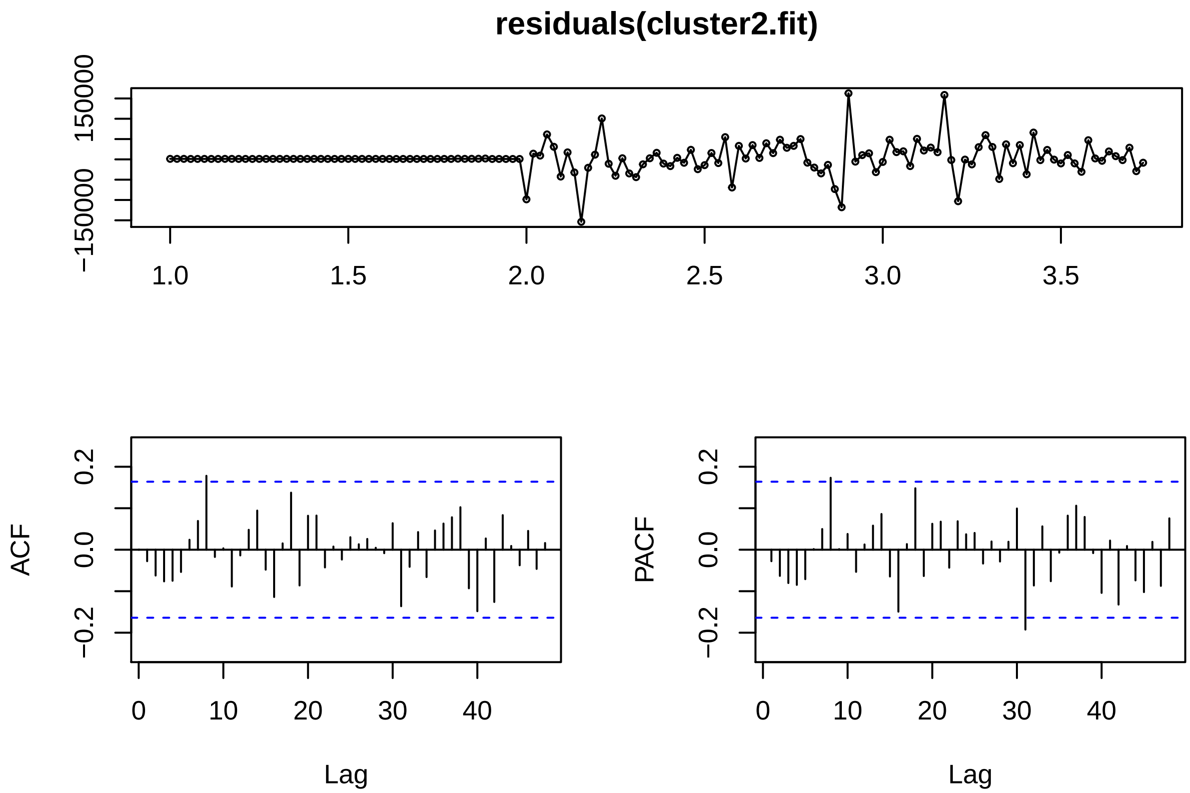



```
#It is observed that all 4 clusters have a clear seasonal pattern for period length of 52 weeks.
#Hence, the seasonal order for ARIMA modeling will be defaulted to 'seasonal= list(order = c(0,1,0), pe
#To find the optimal pdq coeffecients for the trend component, run the following function for each clus


#manually try out combinations of p,d,q
cluster1.fit<-Arima(cluster1.ts,order=c(1,0,1), seasonal = list(order = c(0,1,0), period = 52), include
cluster2.fit<-Arima(cluster2.ts,order=c(1,0,2), seasonal = list(order = c(0,1,0), period = 52), include
cluster3.fit<-Arima(cluster3.ts,order=c(1,0,1), seasonal = list(order = c(0,1,0), period = 52), include
cluster4.fit<-Arima(cluster4.ts,order=c(1,0,1), seasonal = list(order = c(0,1,0), period = 52), include
```

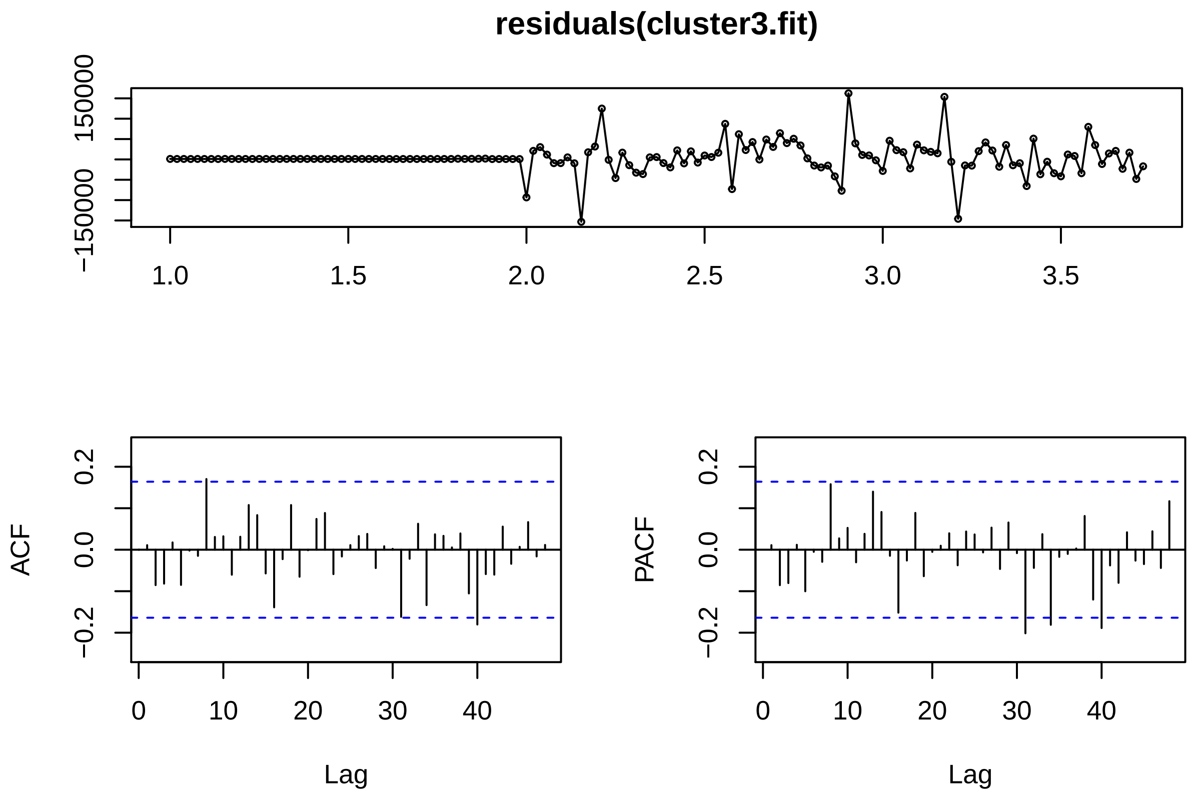
```
### Evaluating forecast accuracy
#
#
# Visually check the fit of the arima model by plotting the ACF, PACF graph of the residuals
# Residuals which fall within the confidence boundaries suggest a good fit
tsdisplay(residuals(cluster1.fit))
```



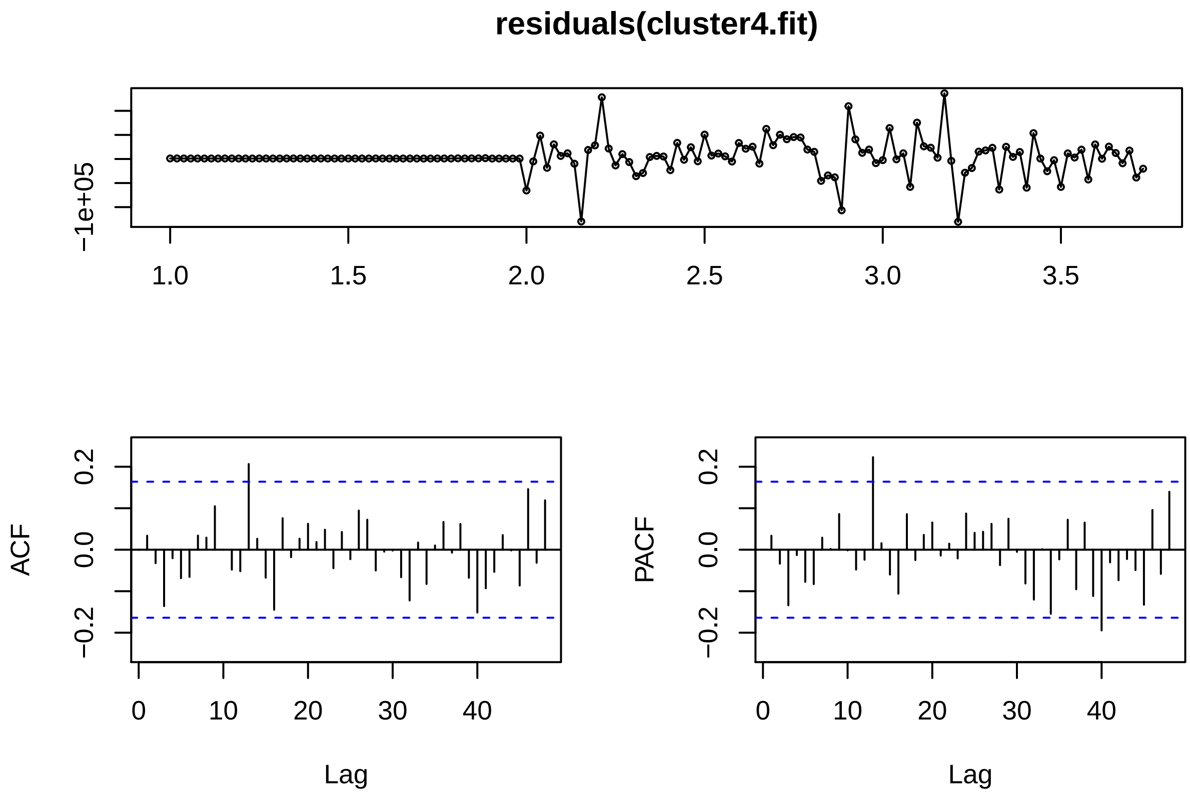

```
tsdisplay(residuals(cluster2.fit))
```




```
tsdisplay(residuals(cluster3.fit))
```




```r
tsdisplay(residuals(cluster4.fit))
```




```
#The mean absolute percentage error turns out to be
#5.837927 for cluster 1
#5.824512 for cluster 2
#5.570019 for cluster 3 and
#6.833386 for cluster 4
```