Italy Unemployment predictions

Importing dataset:

```
rm(list=ls())
data_italy=read.csv("Italy_Unemployment.csv")
head(data_italy)
```

```
##      Time Unemployment_Rate Employment_Rate_15_89 Labour_Force_15_89
## 1 2004-01             0.084             0.4568646           24214141
## 2 2004-02             0.081             0.4571798           24180990
## 3 2004-03             0.083             0.4583856           24302432
## 4 2004-04             0.081             0.4561624           24168970
## 5 2004-05             0.081             0.4598407           24371828
## 6 2004-06             0.079             0.4604707           24361867
##   Unemployment Hourly_LabourWage_Rate      CPI Unemployed_Male
## 1      2018110               73.17862 82.04412          915768
## 2      1959908               73.71726 82.24374          908431
## 3      2013667               73.73799 82.50990          948543
## 4      1960452               74.14905 82.70952          902031
## 5      1975744               74.27891 82.90914          934025
## 6      1917759               74.45344 83.04222          937364
##   Unemployed_Female Activity_Rate_15_89
## 1           1102343           0.4984037
## 2           1051477           0.4975033
## 3           1065124           0.4997982
## 4           1058420           0.4964300
## 5           1041719           0.5004071
## 6            980395           0.4998160
```

```
tail(data_italy)
```

```
##        Time Unemployment_Rate Employment_Rate_15_89 Labour_Force_15_89
## 226 2022-10             0.079             0.4612092           25230811
## 227 2022-11             0.079             0.4608109           25190408
## 228 2022-12             0.079             0.4619117           25253133
## 229 2023-01             0.079             0.4627967           25328798
## 230 2023-02             0.080             0.4629750           25328878
## 231 2023-03             0.078             0.4633645           25328751
##     Unemployment Hourly_LabourWage_Rate   CPI Unemployed_Male Unemployed_Female
## 226      1997032               106.9299 118.1         1003511            993521
## 227      1974718               107.0319 118.7          974734            999983
## 228      1983906               107.1114 119.0          984871            999035
## 229      2014912               107.1974 119.1          979191           1035721
## 230      2001426               107.5405 119.3          990582           1010845
## 231      1979639               107.0000 118.8          976332           1003308
##     Activity_Rate_15_89
## 226           0.5008519
## 227           0.5000073
## 228           0.5012938
## 229           0.5027941
## 230           0.5026969
## 231           0.5026506
```

Feature Scaling - Creating a new variable:

```
data_italy$male_to_female_unemp=round((data_italy$Unemployed_Male/data_italy$Unemployed_Female),4)
print(data_italy$male_to_female_unemp[10])
```

```
## [1] 0.8805
```

This is to incorporate the factor - whether female are getting more unemployed or not compared to males over the years - as an external variable for overall unemployment rate.

Checking Multicollinearity using VIFs:

```
suppressWarnings(library(regclass))
```

```
## Loading required package: bestglm
```

```
## Loading required package: leaps
```

```
## Loading required package: VGAM
```

```
## Loading required package: stats4
```

```
## Loading required package: splines
```

```
## Loading required package: rpart
```

```
## Loading required package: randomForest
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
## Important regclass change from 1.3:
## All functions that had a . in the name now have an _
## all.correlations -> all_correlations, cor.demo -> cor_demo, etc.
```

```
VIF(lm(formula = Unemployment_Rate ~ Employment_Rate_15_89+Labour_Force_15_89+Unemployment+Hourly_
LabourWage_Rate+CPI+Activity_Rate_15_89+male_to_female_unemp, data = data_italy))
```

```
##  Employment_Rate_15_89     Labour_Force_15_89           Unemployment
##           33113.379976             220.571270           31546.672184
## Hourly_LabourWage_Rate                    CPI    Activity_Rate_15_89
##              21.513443              13.363526           12251.169023
##   male_to_female_unemp
##               9.911258
```

Removing Employment_Rate_15_89:

```
VIF(lm(formula = Unemployment_Rate ~ Labour_Force_15_89+Unemployment+Hourly_LabourWage_Rate+CPI+Ac
tivity_Rate_15_89+male_to_female_unemp, data = data_italy))
```

```
##     Labour_Force_15_89           Unemployment Hourly_LabourWage_Rate
##              17.454715               5.271459              20.637735
##                    CPI    Activity_Rate_15_89   male_to_female_unemp
##              13.357824              10.364289               9.693947
```

Removing Hourly_LabourWage_Rate:

```
VIF(lm(formula = Unemployment_Rate ~ Labour_Force_15_89+Unemployment+CPI+Activity_Rate_15_89+male_
to_female_unemp, data = data_italy))
```

```
##   Labour_Force_15_89         Unemployment                  CPI
##            15.944879             5.233670             2.927801
##  Activity_Rate_15_89 male_to_female_unemp
##            10.242361             9.088526
```

Removing Labour_Force_15_89:

```
VIF(lm(formula = Unemployment_Rate ~ Unemployment+CPI+Activity_Rate_15_89+male_to_female_unemp, da
ta = data_italy))
```

```
##         Unemployment                  CPI  Activity_Rate_15_89
##             5.124001             1.917796             2.482966
## male_to_female_unemp
##             6.416522
```

Removing male_to_female_unemp:

```
VIF(lm(formula = Unemployment_Rate ~ Unemployment+CPI+Activity_Rate_15_89, data = data_italy))
```

```
##        Unemployment                  CPI Activity_Rate_15_89
##            1.359424             1.400846            1.114968
```

This is the final set of variables free from multicollinearity.

Train-Test split of the dataset - last 6 months of the data would be taken into testing part:

```
df_italy_train1=data_italy[1:(nrow(data_italy)-6),]
df_italy_test1=data_italy[(nrow(data_italy)-5):nrow(data_italy),]
head(df_italy_train1)
```

```
##      Time Unemployment_Rate Employment_Rate_15_89 Labour_Force_15_89
## 1 2004-01             0.084             0.4568646           24214141
## 2 2004-02             0.081             0.4571798           24180990
## 3 2004-03             0.083             0.4583856           24302432
## 4 2004-04             0.081             0.4561624           24168970
## 5 2004-05             0.081             0.4598407           24371828
## 6 2004-06             0.079             0.4604707           24361867
##   Unemployment Hourly_LabourWage_Rate      CPI Unemployed_Male
## 1      2018110               73.17862 82.04412          915768
## 2      1959908               73.71726 82.24374          908431
## 3      2013667               73.73799 82.50990          948543
## 4      1960452               74.14905 82.70952          902031
## 5      1975744               74.27891 82.90914          934025
## 6      1917759               74.45344 83.04222          937364
##   Unemployed_Female Activity_Rate_15_89 male_to_female_unemp
## 1           1102343           0.4984037               0.8307
## 2           1051477           0.4975033               0.8640
## 3           1065124           0.4997982               0.8905
## 4           1058420           0.4964300               0.8522
## 5           1041719           0.5004071               0.8966
## 6            980395           0.4998160               0.9561
```

- The model would be trained on the train dataset.
- And the performance of the fitted model would be checked on the test dataset.
- If this performs fairly well, this model would be considered to get the future forecasts.

Time series plot:

```
suppressWarnings(library(fpp2))
```

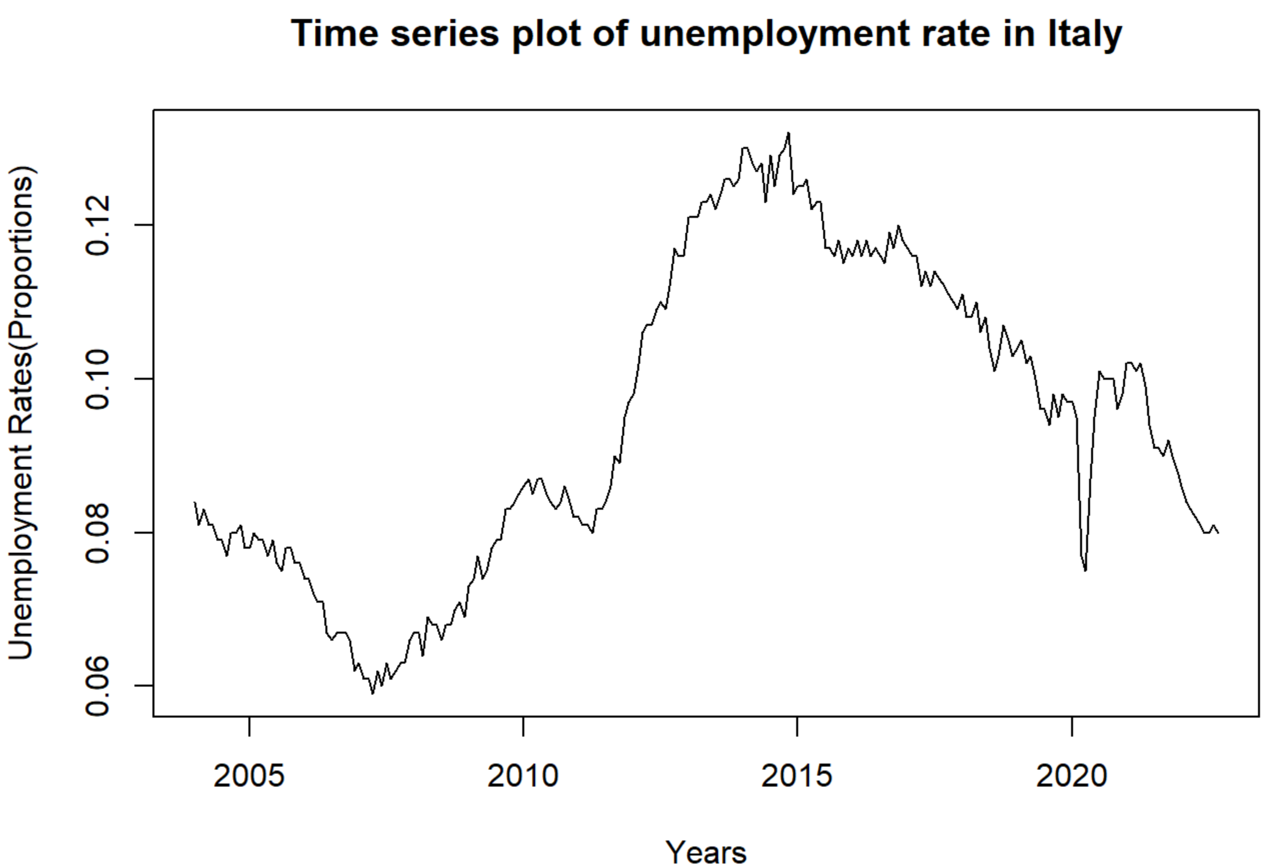
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## ── Attaching packages ──────────────────────────────────────────────── fpp2 2.5 ──
```

```
## ✔ ggplot2   3.3.6     ✔ fma       2.4
## ✔ forecast  8.18      ✔ expsmooth 2.3
```

```
## ── Conflicts ────────────────────────────────────────────────── fpp2_conflicts ──
## ✖ ggplot2::margin() masks randomForest::margin()
```

```
suppressWarnings(library(urca))
df.ts=ts(df_italy_train1$Unemployment_Rate, frequency = 12, start = c(2004,1))
plot(df.ts,xlab="Years",ylab="Unemployment Rates(Proportions)")
title(main="Time series plot of unemployment rate in Italy")
```



Testing stationarity:

```
df_italy_train1[,"Unemployment_Rate"] %>%
  ur.kpss() %>%
  summary()
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 4 lags.
##
## Value of test-statistic is: 2.1354
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

This series is non-stationary - 1st order differencing would be necessary.

Testing stationarity after 1st order differencing:

```
diff(df_italy_train1[,"Unemployment_Rate"]) %>%
  ur.kpss() %>%
  summary()
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 4 lags.
##
## Value of test-statistic is: 0.3682
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

We would undergo one more order of differencing.

Testing stationarity after 2nd order differencing:

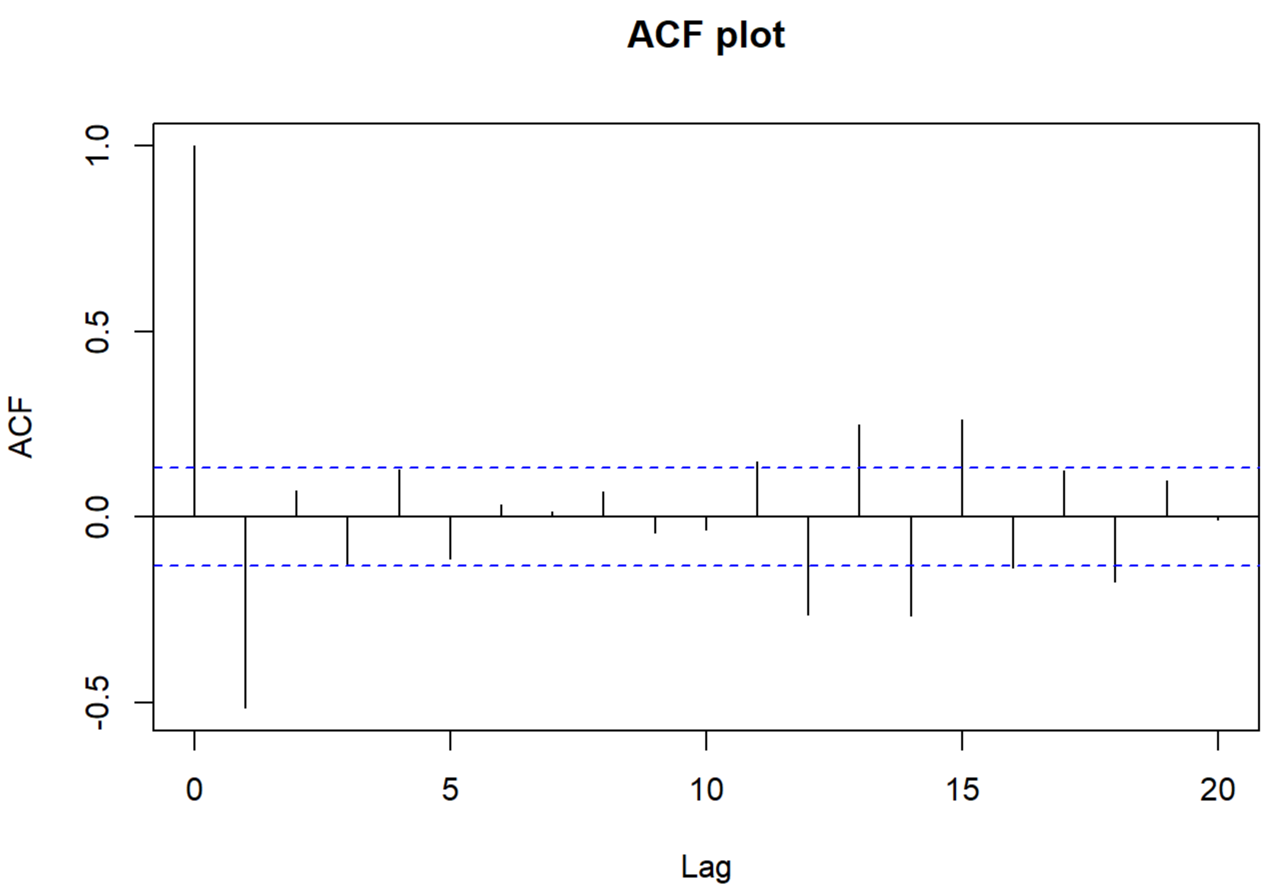
```
diff(diff(df_italy_train1[,"Unemployment_Rate"])) %>%
  ur.kpss() %>%
  summary()
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 4 lags.
##
## Value of test-statistic is: 0.0169
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

Thus the 2nd order differences are stationary.

ACF plot:

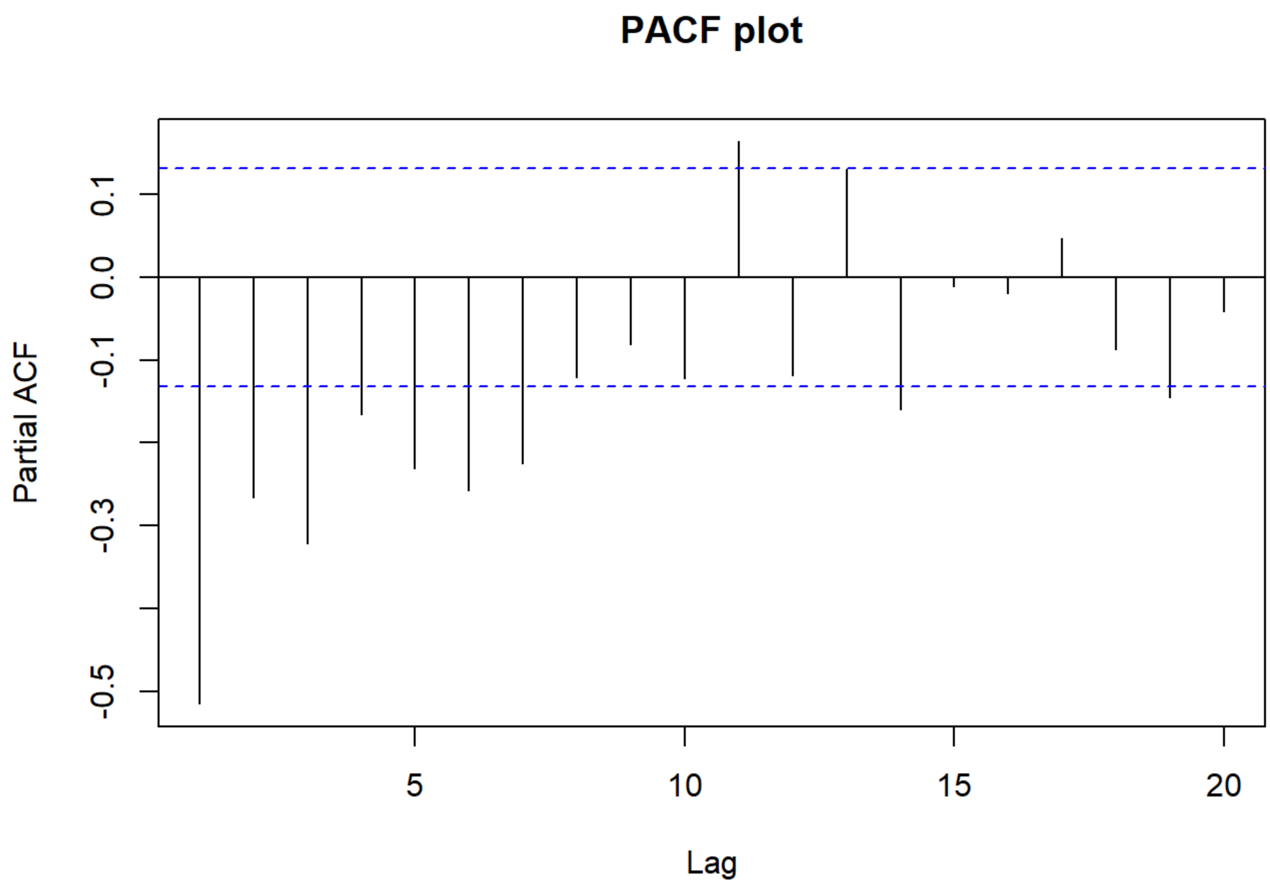
```
par(mfrow=c(1,1))
acf(diff(diff(df_italy_train1$Unemployment_Rate)), lag.max = 20, main = "ACF plot")
```



p can be taken as 0 or 1 based on the no. of significant lags.

PACF plot:

```
par(mfrow=c(1,1))
pacf(diff(diff(df_italy_train1$Unemployment_Rate)), lag.max = 20, main = "PACF plot")
```



q can be 1/2/3, based on the no. of significant lags.

Fitting ARIMAX model ignoring the variables that were eliminated due to high VIF:

Starting with the value of p & q as 1 & 3 respectively and with the rest of the regressors:

```
est_train=arima(df_italy_train1$Unemployment_Rate, order=c(1,2,3), xreg = as.matrix(df_italy_train
1[,c(5,7,10)]), method = "ML")
summary(est_train)
```

```
##
## Call:
## arima(x = df_italy_train1$Unemployment_Rate, order = c(1, 2, 3), xreg = as.matrix(df_italy_trai
n1[,
##     c(5, 7, 10)]), method = "ML")
##
## Coefficients:
```

```
## Warning in sqrt(diag(x$var.coef)): NaNs produced
```

```
##           ar1      ma1      ma2     ma3  Unemployment  CPI  Activity_Rate_15_89
##       -0.7935  -0.9203  -0.7793  0.7189             0    0              -0.1738
## s.e.   0.1472   0.1227   0.2046  0.0946           NaN  NaN                  NaN
##
## sigma^2 estimated as 1.05e-07:  log likelihood = 1472.07,  aic = -2928.13
##
## Training set error measures:
##                        ME         RMSE          MAE        MPE     MAPE
## Training set 4.98814e-05 0.0003226412 0.0002612393 0.05454112 0.289276
##                   MASE         ACF1
## Training set 0.1354574 -0.006125014
```

Test of significance of individual coefficients:

```
suppressWarnings(library(lmtest))
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
## Attaching package: 'lmtest'
```

```
## The following object is masked from 'package:VGAM':
##
##     lrtest
```

```
coeftest(est_train)
```

```
## Warning in sqrt(diag(se)): NaNs produced
```

```
##
## z test of coefficients:
##
##                        Estimate  Std. Error z value  Pr(>|z|)
## ar1                 -7.9354e-01  1.4720e-01 -5.3908 7.016e-08 ***
## ma1                 -9.2030e-01  1.2270e-01 -7.5003 6.367e-14 ***
## ma2                 -7.7933e-01  2.0462e-01 -3.8086 0.0001397 ***
## ma3                  7.1891e-01  9.4640e-02  7.5963 3.047e-14 ***
## Unemployment         4.0336e-08         NaN     NaN       NaN
## CPI                 -1.5843e-05         NaN     NaN       NaN
## Activity_Rate_15_89 -1.7383e-01         NaN     NaN       NaN
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We need to remove the variables one by one that are producing NaNs & the insignificant variables.

After doing that, the summary & test of significances of the final model would look like:

```
est_1=arima(df_italy_train1$Unemployment_Rate, order=c(0,2,1), xreg = as.matrix(df_italy_train1[,c
(7,10)]), method = "ML")
summary(est_1)
```

```
##
## Call:
## arima(x = df_italy_train1$Unemployment_Rate, order = c(0, 2, 1), xreg = as.matrix(df_italy_trai
n1[,
##     c(7, 10)]), method = "ML")
##
## Coefficients:
##           ma1      CPI  Activity_Rate_15_89
##       -0.9153  -0.0014               0.8162
## s.e.   0.0359   0.0005               0.0615
##
## sigma^2 estimated as 4.236e-06:  log likelihood = 1062.14,  aic = -2116.28
##
## Training set error measures:
##                        ME        RMSE         MAE        MPE     MAPE      MASE
## Training set 1.909429e-05 0.002049001 0.001586433 0.07856527 1.732102 0.8225948
##                     ACF1
## Training set -0.03906613
```

Test of significance of coefficients:

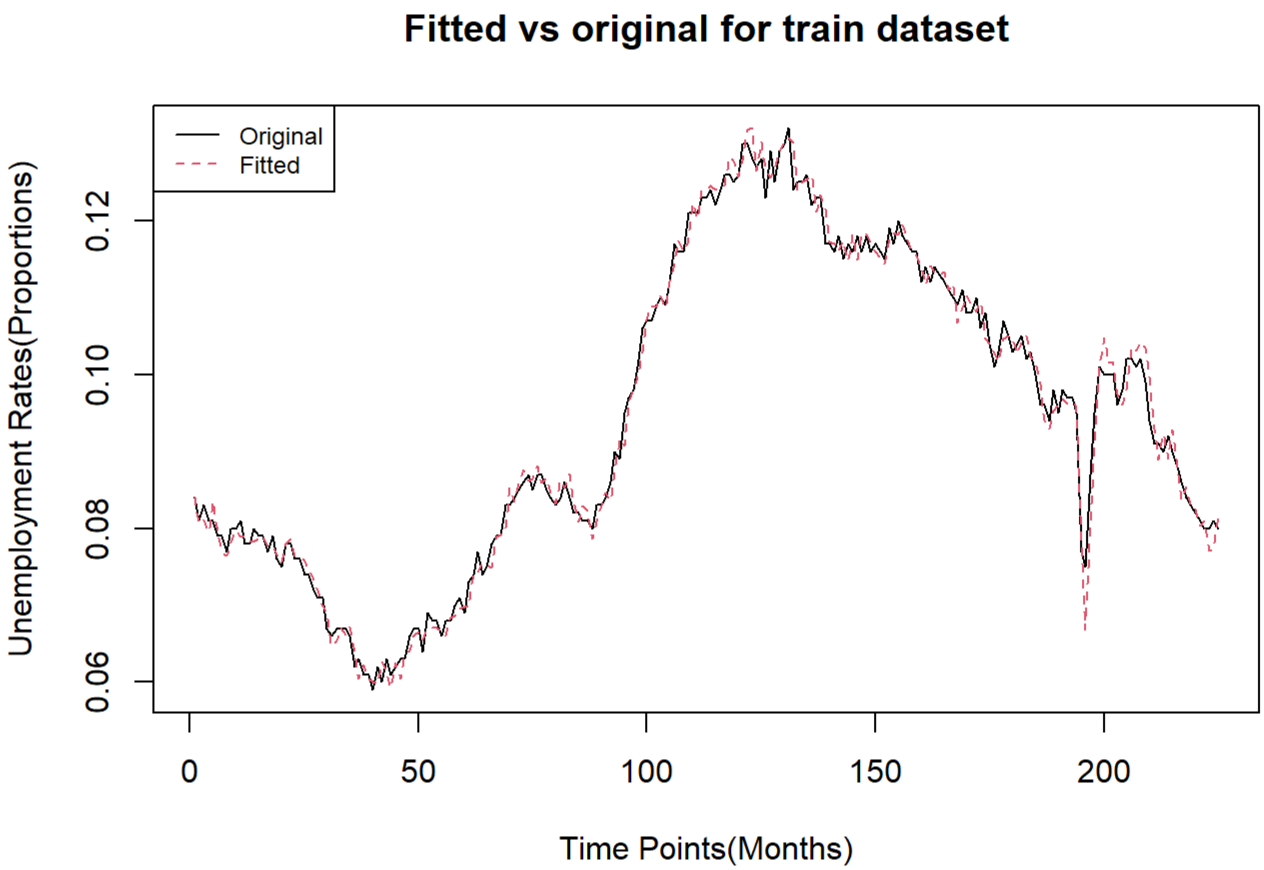
```
suppressWarnings(library(lmtest))
coeftest(est_1)
```

```
##
## z test of coefficients:
##
##                        Estimate  Std. Error  z value Pr(>|z|)
## ma1                 -0.91534725  0.03593566 -25.4718   <2e-16 ***
## CPI                 -0.00138677  0.00053692  -2.5828   0.0098 **
## Activity_Rate_15_89  0.81623175  0.06153933  13.2636   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Thus all the final parameters are kept which are significant in prediction of the target variable.

Plot of Fitted vs Original values for train dataset:

```
res=residuals(est_1)
data_fit=df_italy_train1$Unemployment_Rate-res
ts.plot(df_italy_train1$Unemployment_Rate, type="l", xlab="Time Points(Months)", ylab="Unemploymen
t Rates(Proportions)", main="Fitted vs original for train dataset")
points(data_fit, type="l", col=2, lty=2)
legend("topleft",c("Original","Fitted"), col=c(1,2), lty=c(1,2), cex=0.75)
```



Predictions of unemployment rates for the test dataset using above fitted model:

```
test_pred=predict(est_1, n.ahead=6, newxreg = as.matrix(df_italy_test1[, c(7,10)]), se.fit=FALSE,
method="ML")
```

Predicted values:

```
print(as.vector(test_pred))
```

```
## [1] 0.07548982 0.07377952 0.07422470 0.07512184 0.07457628 0.07504304
```

Original values:

```
print(df_italy_test1$Unemployment_Rate)
```

```
## [1] 0.079 0.079 0.079 0.079 0.080 0.078
```

Performane on test dataset:

MAPE (in %):

```
(1/length(df_italy_test1$Unemployment_Rate))*(sum(abs(df_italy_test1$Unemployment_Rate-as.vector(t
est_pred))/abs(df_italy_test1$Unemployment_Rate)))*100
```

```
## [1] 5.429308
```

RMSE:

```
sqrt(mean((df_italy_test1$Unemployment_Rate-as.vector(test_pred))^2))
```

```
## [1] 0.004388978
```

Thus, the fitted model is working well, more or less, for future datasets.

Now going with the same approach with the actual dataset for getting the future forecast of April,23:

Checking stationarity:

```
data_italy[,"Unemployment_Rate"] %>%
  ur.kpss() %>%
  summary()
```

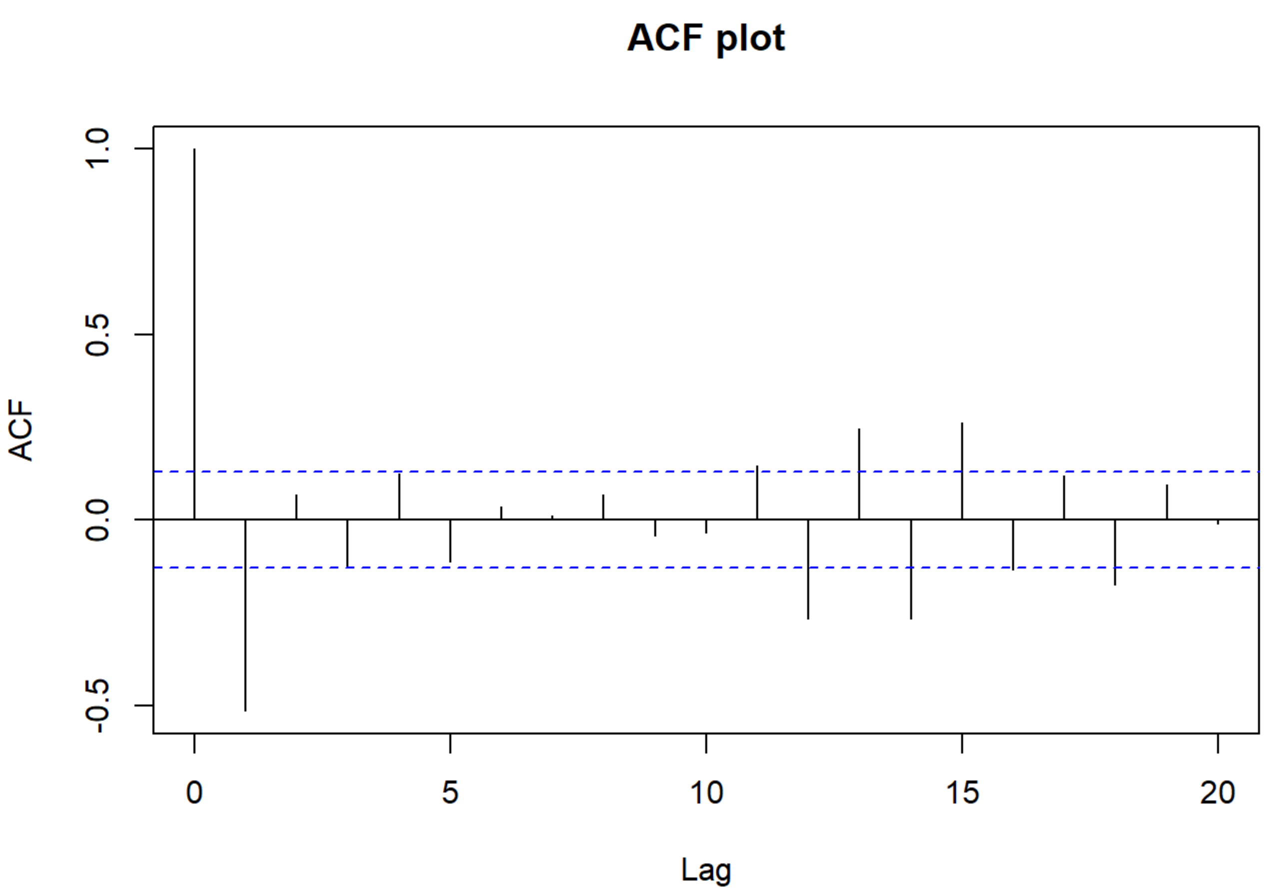
```
## 
## #######################
## # KPSS Unit Root Test #
## #######################
## 
## Test is of type: mu with 4 lags.
## 
## Value of test-statistic is: 1.9519
## 
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

```
diff(diff(data_italy[,"Unemployment_Rate"])) %>%
  ur.kpss() %>%
  summary()
```

```
##
## ######################
## # KPSS Unit Root Test #
## ######################
##
## Test is of type: mu with 4 lags.
##
## Value of test-statistic is: 0.0191
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

ACF plot:

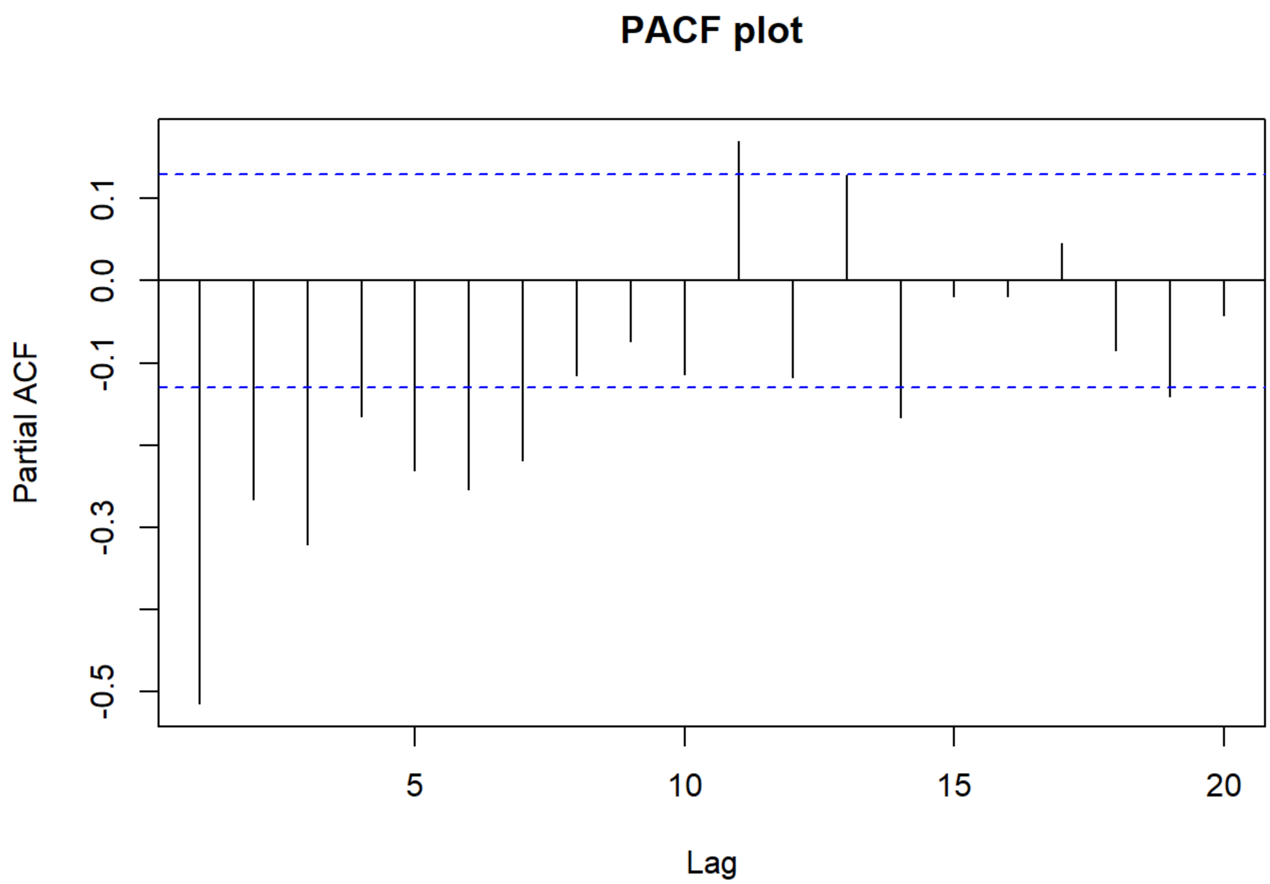
```
par(mfrow=c(1,1))
acf(diff(diff(data_italy$Unemployment_Rate)), lag.max = 20, main = "ACF plot")
```



p can be taken as 0/1, based on the no. of significant lags.

PACF plot:

```
par(mfrow=c(1,1))
pacf(diff(diff(data_italy$Unemployment_Rate)), lag.max = 20, main = "PACF plot")
```



q can be taken as 0/1/2/3/4, based on the no. of significant lags.

Fitting the model that we tested before - on the actual data:

```
est_actual=arima(data_italy$Unemployment_Rate, order=c(0,2,1), xreg = as.matrix(data_italy[,c(7,1
0)]), method = "ML")
summary(est_actual)
```

```
##
## Call:
## arima(x = data_italy$Unemployment_Rate, order = c(0, 2, 1), xreg = as.matrix(data_italy[,
##     c(7, 10)]), method = "ML")
##
## Coefficients:
##           ma1      CPI  Activity_Rate_15_89
##       -0.9142  -9e-04               0.8179
## s.e.   0.0376   4e-04               0.0612
##
## sigma^2 estimated as 4.204e-06:  log likelihood = 1091.62,  aic = -2175.24
##
## Training set error measures:
##                        ME        RMSE         MAE        MPE    MAPE      MASE
## Training set 1.394831e-05 0.002041496 0.001586548 0.06961346 1.73936 0.8369404
##                     ACF1
## Training set -0.04813185
```

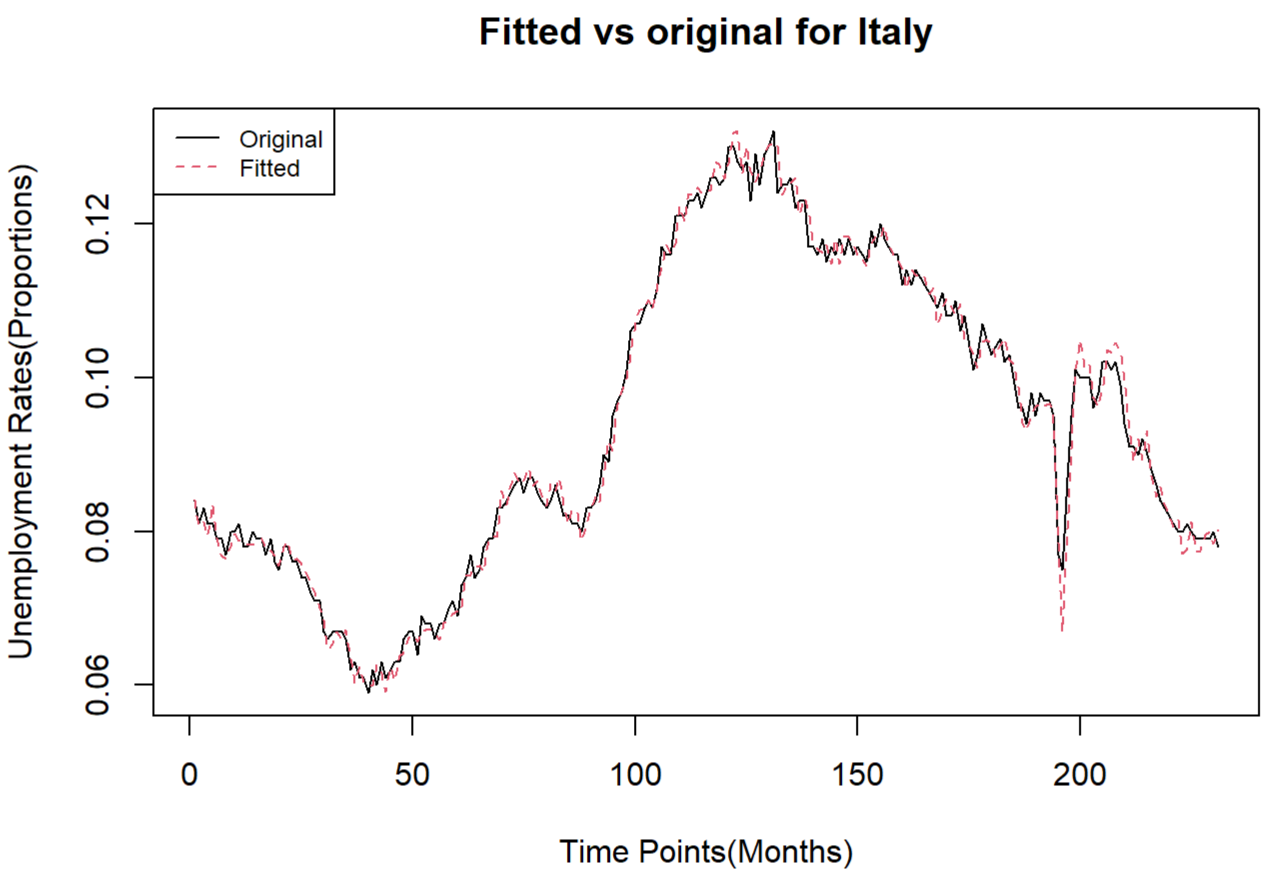
Test of significance of individual coefficients:

```
suppressWarnings(library(lmtest))
coeftest(est_actual)
```

```
##
## z test of coefficients:
##
##                        Estimate  Std. Error  z value Pr(>|z|)
## ma1                 -0.91424835  0.03756138 -24.3401  < 2e-16 ***
## CPI                 -0.00085019  0.00039013  -2.1793  0.02931 *
## Activity_Rate_15_89  0.81793564  0.06124522  13.3551  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Plot of Fitted vs Original on the actual data:

```
res=residuals(est_actual)
data_fit=data_italy$Unemployment_Rate-res
ts.plot(data_italy$Unemployment_Rate, type="l", xlab="Time Points(Months)", ylab="Unemployment Rat
es(Proportions)", main="Fitted vs original for Italy")
points(data_fit, type="l", col=2, lty=2)
legend("topleft",c("Original","Fitted"), col=c(1,2), lty=c(1,2), cex=0.75)
```



Need forecast of CPI and Activity Rate for the month of April 2023.

```
auto.arima(data_italy$CPI, trace = TRUE)
```

```
##
##  Fitting models using approximations to speed things up...
##
##  ARIMA(2,2,2)                    : 181.9039
##  ARIMA(0,2,0)                    : 304.8674
##  ARIMA(1,2,0)                    : 220.372
##  ARIMA(0,2,1)                    : 178.7774
##  ARIMA(1,2,1)                    : 181.9691
##  ARIMA(0,2,2)                    : 180.703
##  ARIMA(1,2,2)                    : Inf
##
##  Now re-fitting the best model(s) without approximations...
##
##  ARIMA(0,2,1)                    : 177.0502
##
##  Best model: ARIMA(0,2,1)
```

```
## Series: data_italy$CPI
## ARIMA(0,2,1)
##
## Coefficients:
##           ma1
##       -0.8625
## s.e.   0.0376
##
## sigma^2 = 0.1245:  log likelihood = -86.5
## AIC=177   AICc=177.05   BIC=183.86
```

```
est_CPI=arima(data_italy$CPI, order=c(0,2,1))
future_CPI=predict(est_CPI, n.ahead=1, se.fit=FALSE)
print(future_CPI)
```

```
## Time Series:
## Start = 232
## End = 232
## Frequency = 1
## [1] 119.3554
```

```
auto.arima(data_italy$Activity_Rate_15_89, trace = TRUE)
```

```
##
##  Fitting models using approximations to speed things up...
##
##  ARIMA(2,1,2) with drift         : -2151.803
##  ARIMA(0,1,0) with drift         : -2151.531
##  ARIMA(1,1,0) with drift         : -2150.588
##  ARIMA(0,1,1) with drift         : -2151.45
##  ARIMA(0,1,0)                    : -2153.55
##  ARIMA(1,1,1) with drift         : -2148.697
##
##  Now re-fitting the best model(s) without approximations...
##
##  ARIMA(0,1,0)                    : -2165.813
##
##  Best model: ARIMA(0,1,0)
```

```
## Series: data_italy$Activity_Rate_15_89
## ARIMA(0,1,0)
##
## sigma^2 = 4.723e-06:  log likelihood = 1083.92
## AIC=-2165.83   AICc=-2165.81   BIC=-2162.39
```

```
est_Activity_Rate=arima(data_italy$Activity_Rate_15_89, order=c(0,1,0))
future_Activity_Rate=predict(est_Activity_Rate, n.ahead=1, se.fit=FALSE)
print(future_Activity_Rate)
```

```
## Time Series:
## Start = 232
## End = 232
## Frequency = 1
## [1] 0.5026506
```

Obtaining prediction of Unemployment rate for April 2023:

```
april_input=data.frame(as.vector(future_CPI), as.vector(future_Activity_Rate))
future_unemp_pred=predict(est_actual, n.ahead=1, newxreg = as.matrix(april_input[, c(1,2)]), se.fi
t=FALSE, method="ML")
print(as.vector(future_unemp_pred))
```

```
## [1] 0.07714709
```

Upper & Lower limits (95% C.I.s):

```
upper=as.vector(future_unemp_pred)+(1.96*(sqrt(est_actual$sigma2)))
lower=as.vector(future_unemp_pred)-(1.96*(sqrt(est_actual$sigma2)))
```

Upper limit for April 2023 forecast:

```
print(as.vector(upper))
```

```
## [1] 0.0811656
```

Lower limit for April 2023 forecast:

```
print(as.vector(lower))
```

```
## [1] 0.07312859
```

April 23 forecast - 7.715 %

Upper & Lower limits - (7.313 %, 8.116 %)