Name – Sarath Kumar Vatyam
NET_ID – ss2405

#BAN_673_CASE3

Creating Time series and Data partition –

```
> # Set working directory for locating files.
> setwd("/Users/sarathkumarvatyam/Downloads")
>
> walmart.data <- read.csv("walmart.csv")
>
> #TIME-SERIES.TS()
> revenue <- walmart.data$Revenue
> walmart.ts <- ts(walmart.data$Revenue,
+                  start = c(2005, 1), end = c(2023, 4), freq = 4)
>
> walmart.ts
      Qtr1  Qtr2  Qtr3  Qtr4
2005 71680 76697 75397 88327
2006 79676 85430 84467 98795
```

```
> # Data Partition
> nValid <- 16
> nTrain <- length(walmart.ts) - nValid
> train.ts <- window(walmart.ts, start = c(2005, 1), end = c(2019, 4))
> valid.ts <- window(walmart.ts, start = c(2020, 1), end = c(2023, 4))
> train.ts
       Qtr1   Qtr2   Qtr3   Qtr4
2005  71680  76697  75397  88327
2006  79676  85430  84467  98795
2007  86410  92999  91865 105749
2008  94940 102342  98345 108627
2009  94242 100876  99373 113594
2010  99811 103726 101952 116360
2011 104189 109366 110226 122728
2012 113010 114282 113800 127559
2013 114070 116830 115688 129706
2014 114960 120125 119001 131565
2015 114826 120229 117408 129667
2016 115904 120854 118179 130936
2017 117542 123355 123179 136267
2018 122690 128028 124894 138793
2019 123925 130377 127991 141671
> valid.ts
       Qtr1   Qtr2   Qtr3   Qtr4
2020 134622 137742 134708 152079
2021 138310 141048 140525 152871
2022 141569 152859 152813 164048
2023 152301 161632 160804 173388
```

Q1A)USING AR(1):

```
> walmart.ar1<- Arima(walmart.ts, order = c(1,0,0))
> summary(walmart.ar1)
Series: walmart.ts
ARIMA(1,0,0) with non-zero mean

Coefficients:
         ar1      mean
      0.9449  120267.7
s.e.  0.0444   16666.3

sigma^2 = 95387002:  log likelihood = -806.13
AIC=1618.27   AICc=1618.6   BIC=1625.26

Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE     MASE       ACF1
Training set 1022.498 9637.262 8133.299 0.2702847 6.990888 1.684678 -0.6532536
```

Observation –

The AR(1) model for Walmart revenue reveals a strong positive autocorrelation, indicating predictability in revenue patterns. With an estimated coefficient of approximately 0.9449, the model suggests that previous quarter revenues significantly influence current quarter revenues. However, further validation and forecasting assessments are needed to confirm the model's predictive power accurately.

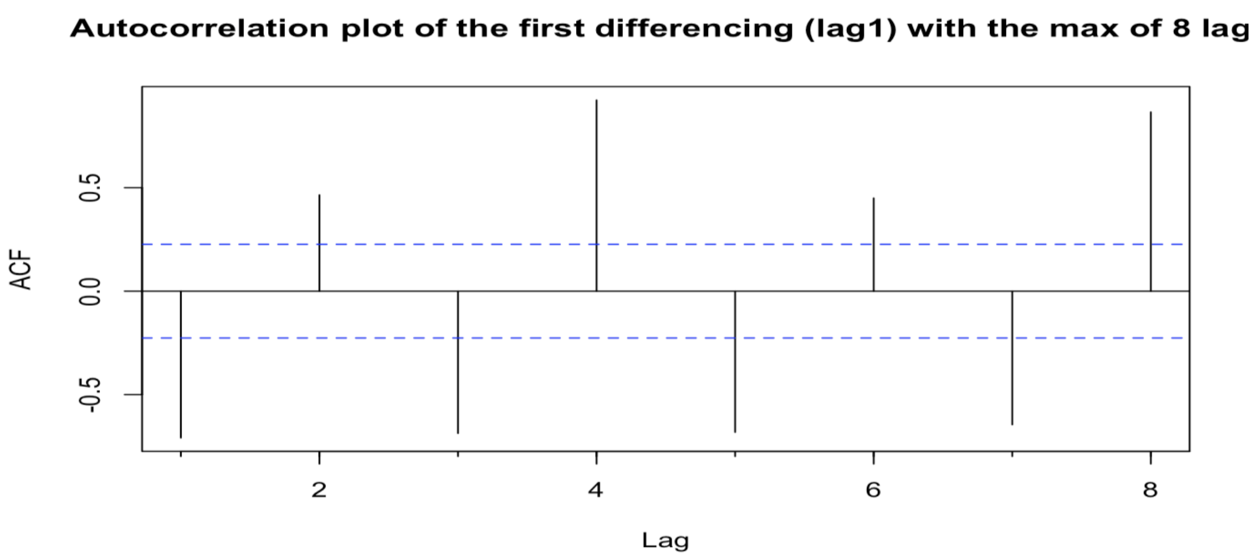
Model Equation –

$$Y_t = 120267.7 + 0.9449 \cdot Y_{t-1} + \epsilon_t$$

Q1b. **Using the first differencing (lag1) of the historical data and Acf() function:**

```
> # Create first difference of Shipment data using diff() function.
> diff.walmart <- diff(walmart.ts, lag = 1)
> diff.walmart
       Qtr1   Qtr2   Qtr3   Qtr4
2005          5017  -1300  12930
2006  -8651   5754   -963  14328
2007 -12385   6589  -1134  13884
2008 -10809   7402  -3997  10282
2009 -14385   6634  -1503  14221
2010 -13783   3915  -1774  14408
2011 -12171   5177    860  12502
2012  -9718   1272   -482  13759
2013 -13489   2760  -1142  14018
2014 -14746   5165  -1124  12564
2015 -16739   5403  -2821  12259
2016 -13763   4950  -2675  12757
2017 -13394   5813   -176  13088
2018 -13577   5338  -3134  13899
2019 -14868   6452  -2386  13680
2020  -7049   3120  -3034  17371
2021 -13769   2738   -523  12346
2022 -11302  11290    -46  11235
2023 -11747   9331   -828  12584
```

ACF PLOT-



Observation –

We identify that the data is significantly auto correlated based on the acf() plot shown above. Lag(2,4,6,8) shows positive correlation whereas Lag(1,3,5,7) shows negative correlation, however all the lags are above the threshold , thus they are significant.

Q2A – Applying regression-based models using tslm() function:
Regression model with linear trend and seasonality:

```
> # Use tslm() function to create linear trend and seasonal model.
> train.lin.season <- tslm(train.ts ~ trend + season)
> # See summary of linear trend equation and associated parameters.
> summary(train.lin.season)

Call:
tslm(formula = train.ts ~ trend + season)

Residuals:
    Min      1Q  Median      3Q     Max 
-9267.6 -3135.2   307.5  3637.7  8485.0 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 79914.73    1455.18  54.917  < 2e-16 ***
trend         848.63      32.25  26.312  < 2e-16 ***
season2      4327.44    1576.86   2.744  0.00817 ** 
season3      1895.41    1577.85   1.201  0.23480    
season4     14285.38    1579.50   9.044  1.8e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4318 on 55 degrees of freedom
Multiple R-squared:  0.9372,	Adjusted R-squared:  0.9326 
F-statistic: 205.1 on 4 and 55 DF,  p-value: < 2.2e-16
```

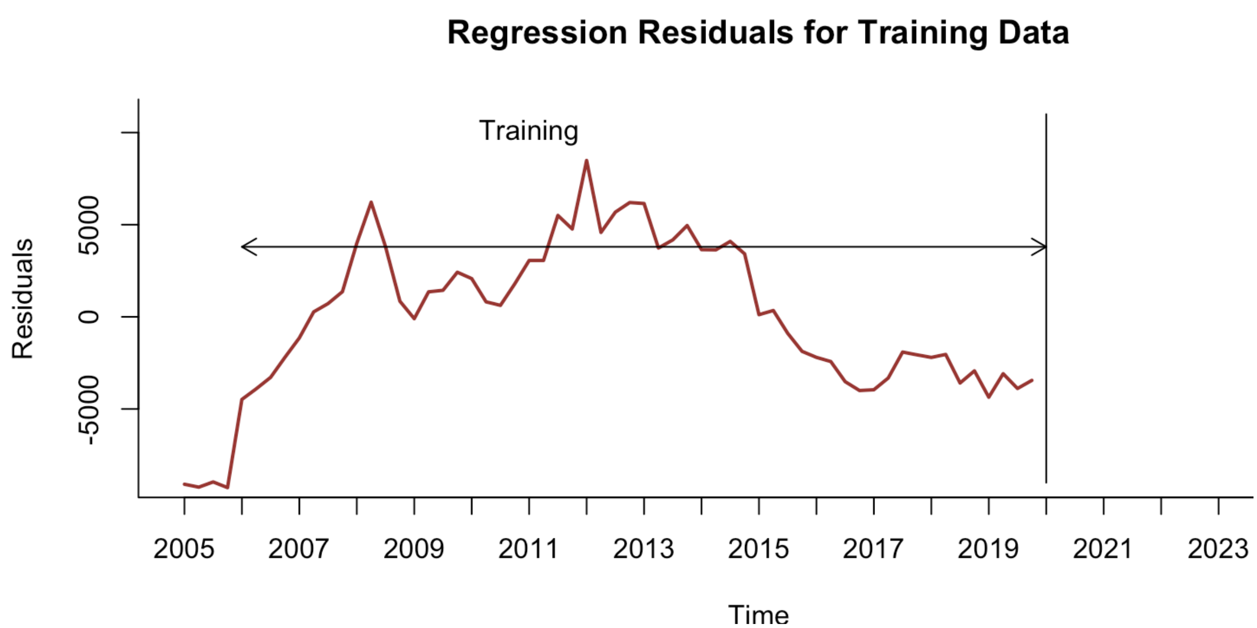
Model Equation - $Y_t = 79914.73 + 848.63 \cdot \text{trend} + 4327.44 \cdot \text{season}_2 + 1895.41 \cdot \text{season}_3 + 14285.38 \cdot \text{season}_4 + \epsilon$

Forecast for validation period using the regression model:

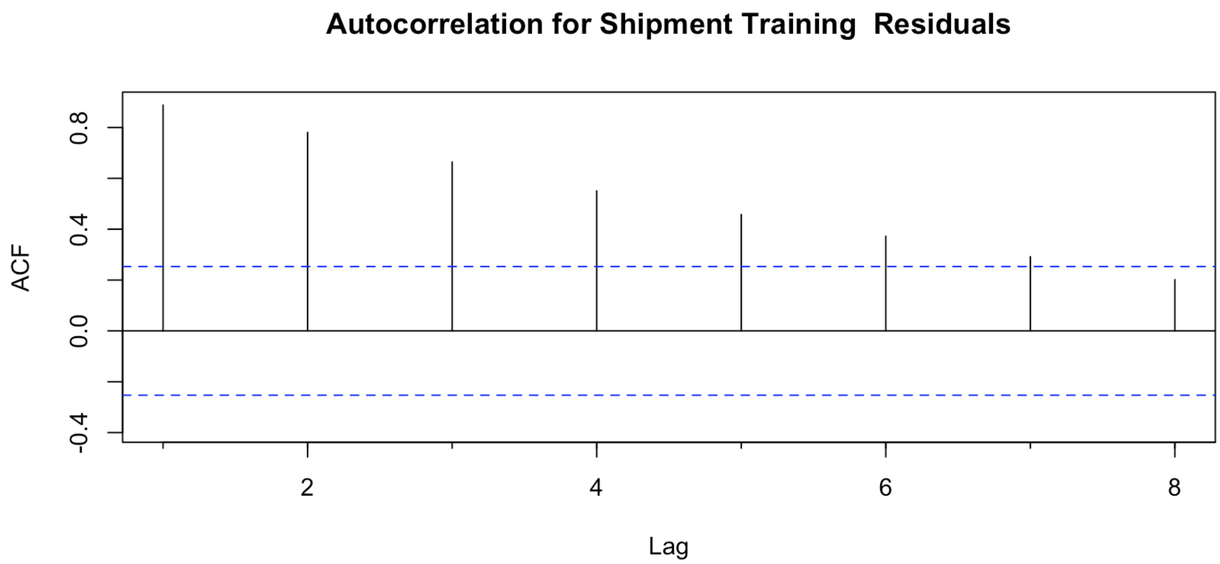
```
> # Apply forecast() function to make predictions for ts with
> # linear trend and seasonal model in validation set.
> train.lin.season.pred <- forecast(train.lin.season, h = nValid, level = 0)
> train.lin.season.pred
        Point Forecast     Lo 0     Hi 0
2020 Q1       131681.2 131681.2 131681.2
2020 Q2       136857.2 136857.2 136857.2
2020 Q3       135273.8 135273.8 135273.8
2020 Q4       148512.4 148512.4 148512.4
2021 Q1       135075.7 135075.7 135075.7
2021 Q2       140251.8 140251.8 140251.8
2021 Q3       138668.4 138668.4 138668.4
2021 Q4       151907.0 151907.0 151907.0
2022 Q1       138470.2 138470.2 138470.2
2022 Q2       143646.3 143646.3 143646.3
2022 Q3       142062.9 142062.9 142062.9
2022 Q4       155301.5 155301.5 155301.5
2023 Q1       141864.7 141864.7 141864.7
2023 Q2       147040.8 147040.8 147040.8
2023 Q3       145457.4 145457.4 145457.4
2023 Q4       158696.0 158696.0 158696.0
```

2b Extracting the residuals for the model's training partition:

```
> train_residuals <- residuals(train.lin.season)
> train_residuals
            Qtr1        Qtr2        Qtr3        Qtr4
2005 -9083.35625 -9242.42292 -8959.02292 -9267.62292
2006 -4481.87679 -3903.94345 -3283.54345 -2194.14345
2007 -1142.39732   270.53601   719.93601  1365.33601
2008  3993.08214  6219.01548  3805.41548   848.81548
2009   -99.43839  1358.49494  1438.89494  2421.29494
2010  2075.04107   813.97440   623.37440  1792.77440
2011  3058.52054  3059.45387  5502.85387  4766.25387
2012  8485.00000  4580.93333  5682.33333  6202.73333
2013  6150.47946  3734.41280  4175.81280  4955.21280
2014  3645.95893  3634.89226  4094.29226  3419.69226
2015   117.43839   344.37173  -893.22827 -1872.82827
2016 -2199.08214 -2425.14881 -3516.74881 -3998.34881
2017 -3955.60268 -3318.66935 -1911.26935 -2061.86935
2018 -2202.12321 -2040.18988 -3590.78988 -2930.38988
2019 -4361.64375 -3085.71042 -3888.31042 -3446.91042
```



Using the Acf() function with the maximum of 8 lags:



Observation –
From the ACF function we can see almost all the lags are statistically significant for all the lags except for the lag 8 and adding the residuals for the model makes it stronger and more precise for the prediction.

Q2C. AR(1) model for the regression residuals

AR (1) MODEL for Regression Residuals –

```
> residuals.ar1 <- Arima(train.lin.season$residuals, order = c(1,0,0))
> residuals.ar1
Series: train.lin.season$residuals
ARIMA(1,0,0) with non-zero mean

Coefficients:
         ar1       mean
      0.9502  -2438.784
s.e.  0.0380   3254.484

sigma^2 = 2304385:  log likelihood = -524.79
AIC=1055.59   AICc=1056.02   BIC=1061.87
```

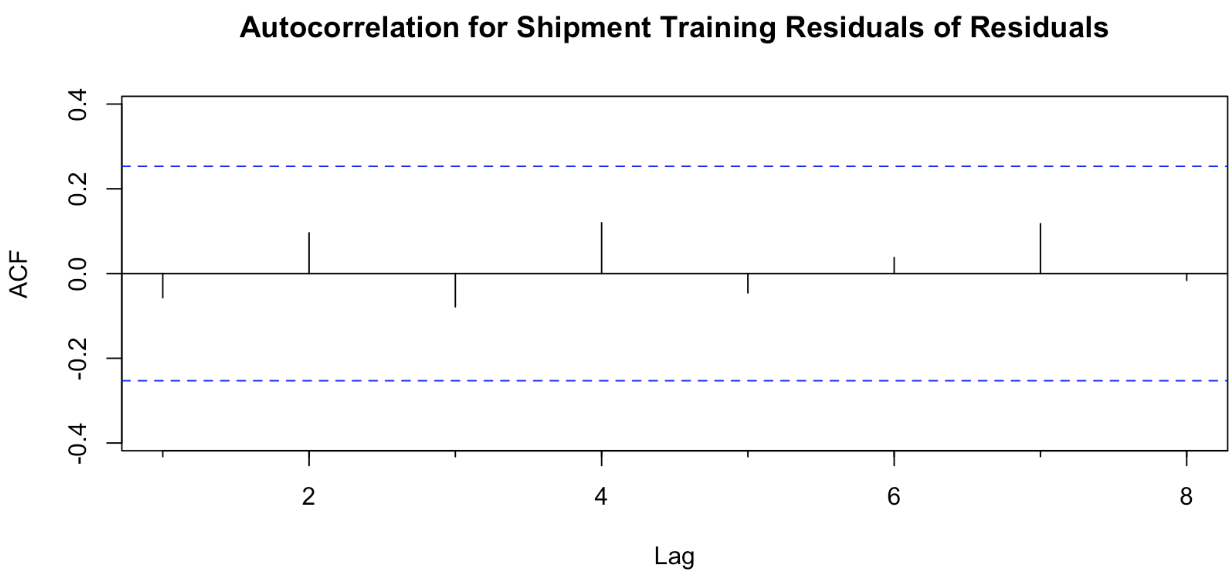
Observation –
The ARIMA(1,0,0) model fitted to the residuals of the linear trend and seasonal model suggests a positive autocorrelation at lag 1, with an estimated AR(1) coefficient of 0.9502, indicating a strong persistence in the residual series. The model equation is $et = -2438.784 + 0.9502 \cdot et - 1$ $e_t = -2438.784 + 0.9502 \cdot e_{t-1}$, implying that each residual is approximately equal to the previous residual multiplied by 0.9502, with an

added constant term of -2438.784. This suggests that the residuals exhibit a strong one-period memory effect.

Summary of AR(1)-

```
> residuals.ar1$fitted
             Qtr1         Qtr2         Qtr3         Qtr4
2005 -7012.422950 -8752.386428 -8903.529896 -8634.246207
2006 -8927.474670 -4380.109205 -3830.963051 -3241.465519
2007 -2206.329136 -1206.971063   135.583279   562.598416
2008  1175.850684  3672.707325  5787.765685  3494.388439
2009   685.058322  -215.962496  1169.350370  1245.745602
2010  2179.211713  1850.204941   651.952704   470.846596
2011  1581.998134  2784.696749  2785.583592  5107.276485
2012  4407.366939  7940.880265  4231.277322  5277.815977
2013  5772.294565  5722.643490  3426.922443  3846.336064
2014  4586.913718  3342.874510  3332.359080  3768.876112
2015  3127.878311    -9.888463   205.741195  -970.213263
2016 -1901.018845 -2211.021828 -2425.827989 -3463.054788
2017 -3920.666025 -3880.049103 -3274.841771 -1937.545145
2018 -2080.643676 -2213.911422 -2060.044077 -3533.407832
2019 -2905.902722 -4265.865043 -3053.486656 -3816.108706
```

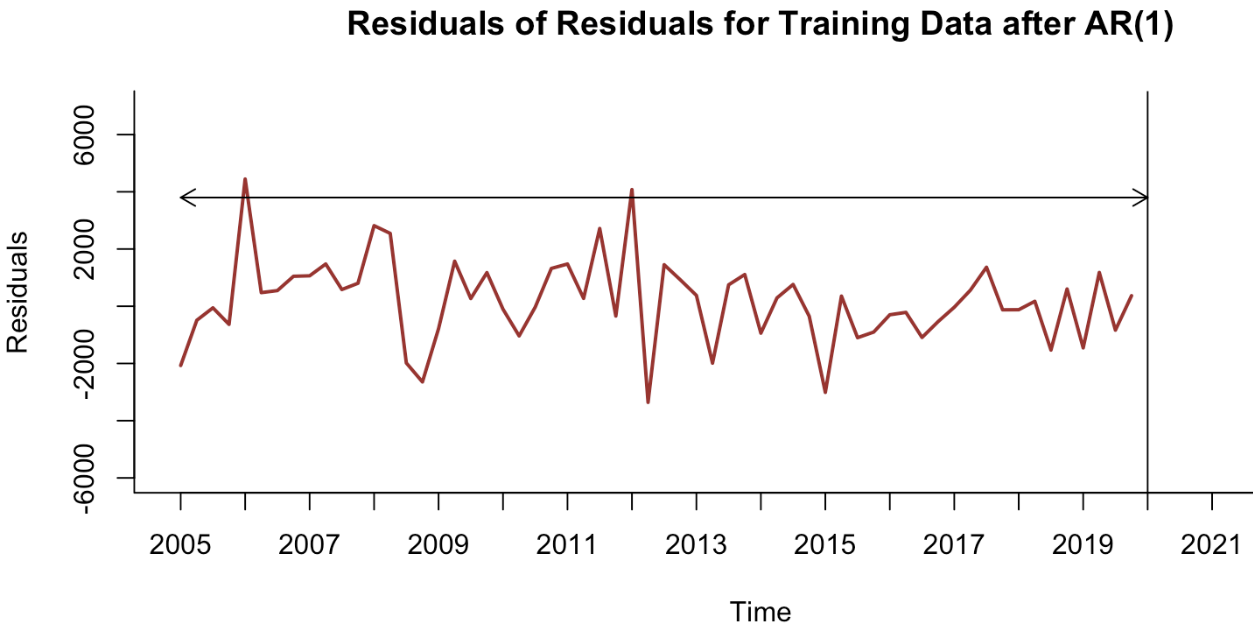
ACF () for Residuals of Residuals –



Observation –

From the ACF plot we can see that all the lags are randoms or zero for the model which makes this model very strong for the prediction and the residuals is taken into consideration

Plotting the residuals of residuals –



Q2D. **Developing the two level forecast:**

```
> two.level.forecast <- train.lin.season.pred$mean + residuals.ar1.pred$mean
> two.level.forecast
         Qtr1     Qtr2     Qtr3     Qtr4
2020 128284.5 133508.2 131970.2 145251.9
2021 131856.1 137071.0 135524.6 148798.3
2022 135394.9 140602.7 139049.4 152316.6
2023 138907.1 144109.0 142550.2 155812.1
```

```
> Table.df <- round(data.frame(valid.ts, train.lin.season.pred$mean,
+                              residuals.ar1.pred$mean, two.level.forecast),3)
> names(Table.df) <- c("walmart_revenue", "Regression.Forecast",
+                      "AR(1) Forecast", "Combined.Forecast")
> Table.df
   walmart_revenue Regression.Forecast AR(1) Forecast Combined.Forecast
1           134622            131681.2      -3396.695          128284.5
2           137742            136857.2      -3348.981          133508.2
3           134708            135273.8      -3303.644          131970.2
4           152079            148512.4      -3260.564          145251.9
5           138310            135075.7      -3219.631          131856.1
6           141048            140251.8      -3180.737          137071.0
7           140525            138668.4      -3143.780          135524.6
8           152871            151907.0      -3108.663          148798.3
9           141569            138470.2      -3075.296          135394.9
10          152859            143646.3      -3043.591          140602.7
11          152813            142062.9      -3013.466          139049.4
12          164048            155301.5      -2984.840          152316.6
13          152301            141864.7      -2957.641          138907.1
14          161632            147040.8      -2931.796          144109.0
15          160804            145457.4      -2907.239          142550.2
16          173388            158696.0      -2883.905          155812.1
> 
```

OBSERVATION –

Above is the a table with the validation data of the walmarts, regression forecast for the validation data, AR(1) forecast for the validation data, and combined forecast for the validation period.

Q2E. Two_level forecast with entire dataset –

```
> #two-level forecast for entire data set
> lin.trend.season.walmart<- tslm(walmart.ts ~ trend + season)
> summary(lin.trend.season.walmart)

Call:
tslm(formula = walmart.ts ~ trend + season)

Residuals:
    Min      1Q  Median      3Q     Max 
-7427.9 -4275.5   524.9  3108.0 10593.4 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 77548.36    1460.13  53.111  < 2e-16 ***
trend         940.62      25.46  36.945  < 2e-16 ***
season2      4539.38    1577.91   2.877   0.0053 ** 
season3      2115.49    1578.52   1.340   0.1845    
season4     14444.08    1579.55   9.144 1.27e-13 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4863 on 71 degrees of freedom
Multiple R-squared:  0.9547,	Adjusted R-squared:  0.9522 
F-statistic: 374.4 on 4 and 71 DF,  p-value: < 2.2e-16
```

OBSERVATION –

The linear trend and seasonal model fitted to the Walmart revenue data strong explanatory power (Adjusted R-squared: 0.9522), with an equation represented as y^=77548.36+940.62·trend+4539.38·season2+2115.49·season3+14444.08·season4y, highlighting significant coefficients for trend and each season, except season3.

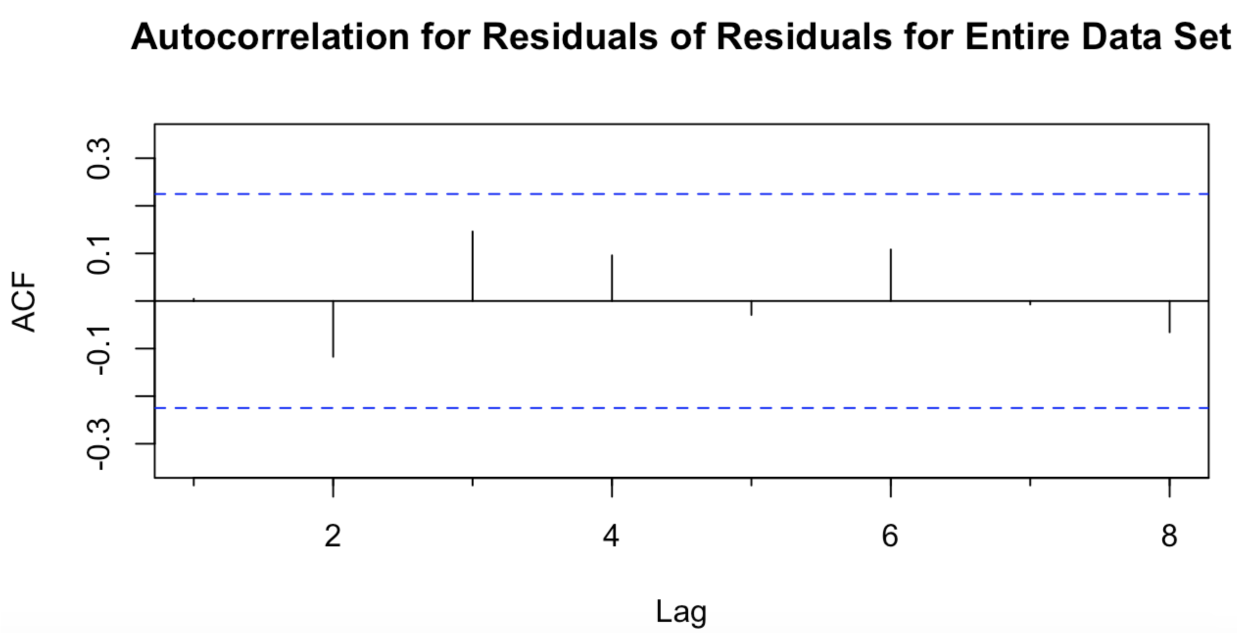
```
> res.ar1 <- Arima(lin.trend.season.walmart$residuals, order = c(1,0,0))
> summary(res.ar1)
Series: lin.trend.season.walmart$residuals
ARIMA(1,0,0) with non-zero mean

Coefficients:
         ar1       mean
      0.9384   463.9948
s.e.  0.0423  3000.0308

sigma^2 = 3705620:  log likelihood = -682.65
AIC=1371.3   AICc=1371.64   BIC=1378.3

Training set error measures:
                   ME     RMSE      MAE      MPE     MAPE      MASE
Training set 150.6701 1899.501 1390.921 -62.5867 116.2553 0.4623774
                    ACF1
Training set 0.004561032
```

ACF() plot for Residuals of Residuals –



OBSERVATION –

The absence of autocorrelation exceeding the significance threshold in the residuals of the residuals indicates that the ARIMA model adequately captures the temporal dependencies in the data, validating its effectiveness in removing autocorrelation structure.

Future forecast for linear trend and seasonality model, AR(1) model for residuals:

```
> Acf(res.ar1$residuals, lag.max = 08,
+     main = "Autocorrelation for Residuals of Residuals for Entire Data Set")
> linear.trend.seasonality.pred <- forecast(lin.trend.season.walmart, h = 08, level = 0)
> linear.trend.seasonality.pred
        Point Forecast     Lo 0     Hi 0
2024 Q1       149976.4 149976.4 149976.4
2024 Q2       155456.4 155456.4 155456.4
2024 Q3       153973.1 153973.1 153973.1
2024 Q4       167242.3 167242.3 167242.3
2025 Q1       153738.8 153738.8 153738.8
2025 Q2       159218.8 159218.8 159218.8
2025 Q3       157735.6 157735.6 157735.6
2025 Q4       171004.8 171004.8 171004.8
>
```

```
> res.ar1.pred <- forecast(res.ar1, h = 08, level = 0)
> res.ar1.pred
        Point Forecast     Lo 0     Hi 0
2024 Q1       9326.569 9326.569 9326.569
2024 Q2       8780.763 8780.763 8780.763
2024 Q3       8268.571 8268.571 8268.571
2024 Q4       7787.922 7787.922 7787.922
2025 Q1       7336.875 7336.875 7336.875
2025 Q2       6913.605 6913.605 6913.605
2025 Q3       6516.403 6516.403 6516.403
2025 Q4       6143.662 6143.662 6143.662
>
```

```
> linear.season.ar1.pred <- linear.trend.seasonality.pred$mean + res.ar1.pred$mean
> linear.season.ar1.pred
         Qtr1     Qtr2     Qtr3     Qtr4
2024 159302.9 164237.1 162241.7 175030.2
2025 161075.7 166132.4 164252.0 177148.5
>
```

```
> Data_Table <- round(data.frame(linear.trend.seasonality.pred$mean,
+                               res.ar1.pred$mean, linear.season.ar1.pred),3)
> names(Data_Table) <- c("Regression.Forecast", "AR(1)Forecast","Combined.Forecast")
> Data_Table
  Regression.Forecast AR(1)Forecast Combined.Forecast
1            149976.4      9326.569          159302.9
2            155456.4      8780.763          164237.1
3            153973.1      8268.571          162241.7
4            167242.3      7787.922          175030.2
5            153738.8      7336.875          161075.7
6            159218.8      6913.605          166132.4
7            157735.6      6516.403          164252.0
8            171004.8      6143.662          177148.5
>
```

3a. Using Arima function to fit ARIMA(1,1,1)(1,1,1) model for the training data set:

```
> #3a
> # Using Arima() function to fit ARIMA(1,1,1)(1,1,1) model for trend and seasonality.
> train.arima.seas <- Arima(train.ts, order = c(1,1,1),
+                          seasonal = c(1,1,1))
> summary(train.arima.seas)
Series: train.ts
ARIMA(1,1,1)(1,1,1)[4]

Coefficients:
          ar1     ma1    sar1     sma1
      -0.7543  0.6701  0.2202  -0.8295
s.e.   0.3046  0.3200  0.1994   0.1718

sigma^2 = 2687313:  log likelihood = -484.6
AIC=979.2   AICc=980.43   BIC=989.24

Training set error measures:
                    ME     RMSE      MAE        MPE      MAPE      MASE
Training set -301.8863 1511.362 1067.091 -0.2885631 0.9726524 0.2674402
                     ACF1
Training set -0.006241391
>
```

OBSERVATION –

The ARIMA(1,1,1)(1,1,1)[4] model captures both trend and seasonality effectively, with coefficients indicating a negative first-order autoregressive term (AR), a positive first-order moving average term (MA), a positive seasonal AR term, and a negative seasonal MA term. The model equation is $\Delta yt = -0.7543 \cdot \Delta yt - 1 + 0.6701 \cdot \epsilon t - 1 + 0.2202 \cdot \Delta yt - 4 - 0.8295 \cdot \epsilon t - 4 \Delta y_t = -0.7543 \cdot \Delta y_{t-1} + 0.6701 \cdot \epsilon_{t-1} + 0.2202 \cdot \Delta y_{t-4} - 0.8295 \cdot \epsilon_{t-4}$, where $\Delta yt \Delta y_t$ denotes differenced observations and $\epsilon t \epsilon_t$ represents residuals.

Applying Forecast() for Validation data-

```
> # Apply forecast() function to make predictions for ts with
> # ARIMA model in validation set.
> train.arima.seas.pred <- forecast(train.arima.seas, h = nValid, level = 0)
> train.arima.seas.pred
        Point Forecast     Lo 0     Hi 0
2020 Q1       127543.4 127543.4 127543.4
2020 Q2       133248.8 133248.8 133248.8
2020 Q3       131191.6 131191.6 131191.6
2020 Q4       144646.8 144646.8 144646.8
2021 Q1       130767.7 130767.7 130767.7
2021 Q2       136244.2 136244.2 136244.2
2021 Q3       134308.1 134308.1 134308.1
2021 Q4       147677.1 147677.1 147677.1
2022 Q1       133880.5 133880.5 133880.5
2022 Q2       139285.6 139285.6 139285.6
2022 Q3       137391.9 137391.9 137391.9
2022 Q4       150730.1 150730.1 150730.1
2023 Q1       136960.6 136960.6 136960.6
2023 Q2       142343.3 142343.3 142343.3
2023 Q3       140464.0 140464.0 140464.0
2023 Q4       153791.6 153791.6 153791.6
```

3b. Using the auto.arima() function to develop an ARIMA model:

```
> # FIT AUTO ARIMA MODEL.
> # Use auto.arima() function to fit ARIMA model.
> # Use summary() to show auto ARIMA model and its parameters.
> train.auto.arima <- auto.arima(train.ts)
> summary(train.auto.arima)
Series: train.ts
ARIMA(0,1,0)(0,1,1)[4]

Coefficients:
         sma1
      -0.6785
s.e.   0.1605

sigma^2 = 2698549:  log likelihood = -485.99
AIC=975.98   AICc=976.22   BIC=980

Training set error measures:
                    ME     RMSE      MAE        MPE     MAPE      MASE       ACF1
Training set -242.4378 1558.427 1146.186 -0.2410367 1.039209 0.2872633 -0.134278
```

Observation: The auto.arima() function suggests an ARIMA(0,1,0)(0,1,1)[4] model, indicating no autoregressive (AR) terms but a seasonal moving average (SMA) term. The model equation simplifies to $\Delta yt = -0.6785 \cdot \epsilon t - 4 \Delta y_t = -0.6785 \cdot \epsilon_{t-4}$, where $\Delta yt \Delta y_t$ represents differenced observations and $\epsilon t \epsilon_t$ denotes residuals. This model demonstrates relatively higher mean absolute error (MAE) and root mean squared error (RMSE) compared to the ARIMA(1,1,1)(1,1,1)[4] model, suggesting a less accurate fit to the training data.

Forecast() in Validation data-

```
> # Apply forecast() function to make predictions for ts with
> # auto ARIMA model in validation set.
> train.auto.arima.pred <- forecast(train.auto.arima, h = nValid, level = 0)
> train.auto.arima.pred
        Point Forecast     Lo 0     Hi 0
2020 Q1       127552.1 127552.1 127552.1
2020 Q2       133115.3 133115.3 133115.3
2020 Q3       131035.1 131035.1 131035.1
2020 Q4       144424.5 144424.5 144424.5
2021 Q1       130305.6 130305.6 130305.6
2021 Q2       135868.8 135868.8 135868.8
2021 Q3       133788.6 133788.6 133788.6
2021 Q4       147178.0 147178.0 147178.0
2022 Q1       133059.1 133059.1 133059.1
2022 Q2       138622.3 138622.3 138622.3
2022 Q3       136542.1 136542.1 136542.1
2022 Q4       149931.6 149931.6 149931.6
2023 Q1       135812.7 135812.7 135812.7
2023 Q2       141375.9 141375.9 141375.9
2023 Q3       139295.6 139295.6 139295.6
2023 Q4       152685.1 152685.1 152685.1
>
```

3c. Applying the accuracy function to the arima and auto arima models:

```
> #3c
> #comparing of arima models
> round(accuracy(train.arima.seas.pred$mean, valid.ts), 3)
               ME    RMSE      MAE   MPE  MAPE  ACF1 Theil's U
Test set 10677.74 12111.8 10677.74 6.948 6.948 0.813     1.209
> round(accuracy(train.auto.arima.pred$mean, valid.ts), 3)
               ME     RMSE      MAE  MPE MAPE  ACF1 Theil's U
Test set 11295.41 12807.88 11295.41 7.35 7.35 0.818     1.279
>
```

Observation –

After Observing the Accuracy measures , ARIMA is better model in this scenario comparative with AUTO ARIMA , where ARIMA has lower error rate i.e., high interms of accuracy of MAPE 6.948 and RMSE of 10677.74.

Q3D – USING ARIMA & AUTO ARIMA ON ENTIRE DATASET-

```
> # Using Arima() function to fit ARIMA(1,1,1)(1,1,1) model for
> # trend and seasonality for entire data set.
> arima.seas <- Arima(walmart.ts, order = c(1,1,1),
+                     seasonal = c(1,1,1))
> summary(arima.seas)
Series: walmart.ts
ARIMA(1,1,1)(1,1,1)[4]

Coefficients:
         ar1      ma1    sar1     sma1
      0.3067  -0.3875  0.1932  -1.0000
s.e.  0.6198   0.5978  0.1321   0.1177

sigma^2 = 3843367:  log likelihood = -642.07
AIC=1294.15   AICc=1295.07   BIC=1305.46

Training set error measures:
                    ME     RMSE      MAE        MPE     MAPE      MASE
Training set -123.1643 1840.715 1270.032 -0.1648121 1.047856 0.2630661
                    ACF1
Training set 0.005624717
```

Observation –

The ARIMA(1,1,1)(1,1,1)[4] model, representing trend and seasonality for the entire dataset, is described by the equation

$\Delta yt=0.3067\cdot\epsilon t-1-0.3875\cdot\epsilon t-1*+0.1932\cdot\Delta yt-4-1.0000\cdot\epsilon t-4*\Delta y_t=0.3067\cdot\epsilon_{t-1}-0.3875\cdot\epsilon_{t-1*}+0.1932\cdot\Delta y_{t-4}-1.0000\cdot\epsilon_{t-4*}$, where $\Delta yt\Delta y_t$ denotes differenced observations, $\epsilon t\epsilon_t$ signifies residuals, and $\epsilon t*\epsilon_{t*}$ represents seasonal residuals.

FORECAST ()WITH ARIMA –

```
> # Apply forecast() function to make predictions for ts with
> # seasonal ARIMA model for the future 8 periods.
> arima.seas.pred <- forecast(arima.seas, h = 8, level = 0)
> arima.seas.pred
        Point Forecast     Lo 0     Hi 0
2024 Q1       161000.0 161000.0 161000.0
2024 Q2       167250.1 167250.1 167250.1
2024 Q3       165899.6 165899.6 165899.6
2024 Q4       179026.0 179026.0 179026.0
2025 Q1       166537.2 166537.2 166537.2
2025 Q2       172199.2 172199.2 172199.2
2025 Q3       170749.9 170749.9 170749.9
2025 Q4       183981.8 183981.8 183981.8
```

AUTO-ARIMA ON ENTIRE DATASET-

```
> # Use auto.arima() function to fit ARIMA model for entire data set.
> auto.arima <- auto.arima(walmart.ts)
> summary(auto.arima)
Series: walmart.ts
ARIMA(0,1,0)(2,1,0)[4]

Coefficients:
         sar1    sar2
      -0.5060  -0.253
s.e.   0.1148   0.122

sigma^2 = 4903509:  log likelihood = -647.24
AIC=1300.48   AICc=1300.84   BIC=1307.27

Training set error measures:
                   ME     RMSE      MAE         MPE     MAPE     MASE       ACF1
Training set 16.86703 2109.946 1495.664 -0.03204543 1.233425 0.309802 -0.1292701
>
```

Observation –

The auto.arima function suggests an ARIMA(0,1,0)(2,1,0)[4] model, indicating differencing at lag 1, seasonal differencing at lag 4, and including two seasonal AR terms. The model equation can be represented as $\Delta yt = -0.5060 \cdot \Delta yt-4 - 0.253 \cdot \Delta yt-8 + \epsilon t \Delta y_t = -0.5060 \cdot \Delta y_{t-4} - 0.253 \cdot \Delta y_{t-8} + \epsilon_t$, where $\Delta yt \Delta y_t$ denotes differenced observations and $\epsilon t \epsilon_t$ represents residuals.

FORECAST() WITH AUTO-ARIMA-

```
> # Apply forecast() function to make predictions for ts with
> # auto ARIMA model for the future 8 periods.
> auto.arima.pred <- forecast(auto.arima, h = 8, level = 0)
> auto.arima.pred
        Point Forecast     Lo 0     Hi 0
2024 Q1       161241.9 161241.9 161241.9
2024 Q2       169400.0 169400.0 169400.0
2024 Q3       168846.9 168846.9 168846.9
2024 Q4       181029.5 181029.5 181029.5
2025 Q1       169197.9 169197.9 169197.9
2025 Q2       178445.2 178445.2 178445.2
2025 Q3       177950.9 177950.9 177950.9
2025 Q4       189995.2 189995.2 189995.2
```

COMPARING PERFORMANCE MEASURES OF ALL MODELS WITH ACCURACY() FUNCTION –

```
> round(accuracy(linear.trend.seasonality.pred$fitted, walmart.ts),3)
         ME    RMSE      MAE   MPE  MAPE  ACF1 Theil's U
Test set  0 4700.12 4017.852 -0.17 3.428 0.875     0.478
> round(accuracy(linear.trend.seasonality.pred$fitted + res.ar1.pred$fitted, walmart.t
s),3)
             ME     RMSE      MAE   MPE  MAPE  ACF1 Theil's U
Test set 150.67 1899.501 1390.921 0.104 1.187 0.005     0.186
> round(accuracy(arima.seas.pred$fitted, walmart.ts), 3)
               ME     RMSE      MAE    MPE  MAPE  ACF1 Theil's U
Test set -123.164 1840.715 1270.032 -0.165 1.048 0.006     0.177
> round(accuracy(auto.arima.pred$fitted, walmart.ts), 3)
             ME     RMSE      MAE    MPE  MAPE   ACF1 Theil's U
Test set 16.867 2109.946 1495.664 -0.032 1.233 -0.129     0.203
> round(accuracy((snaive(walmart.ts))$fitted, walmart.ts), 3)
           ME     RMSE      MAE   MPE  MAPE  ACF1 Theil's U
Test set 4667 5863.128 4827.806 3.938 4.081 0.741     0.596
>
```

Observation –

Based on the above model accuracies we can conclude that ARIMA model has the lowes error rate i.e., highest comparative accuracy among the models present with MAPE of 1.048 & RMSE OF 1840.715

Thank you,
Sarath Kumar
Net_ID – ss2405