TimeSeries Report

Gustavo Chinchayan

12/15/2021

A) Load the necessary libraries for this assignment

library(forecast)

library(ggplot2)

library("readxl") this tool allows me to view the excel file for viewing purposes

library(tseries)

B) Extract Dataset

```
elecdata=read_excel("Elec-train.xlsx")
head(elecdata)
```

```
## # A tibble: 6 x 3
##   Timestamp      `Power (kW)` `Temp (C°)`
##   <chr>                 <dbl>       <dbl>
## 1 1/1/2010 1:15          165.        10.6
## 2 1/1/2010 1:30          152.        10.6
## 3 1/1/2010 1:45          147.        10.6
## 4 1/1/2010 2:00          154.        10.6
## 5 1/1/2010 2:15          154.        10.6
## 6 1/1/2010 2:30          159         10.6
```

Summary tool provides a summary statistics on the column of the data frame

```
summary(elecdata)
```

```
##   Timestamp           Power (kW)      Temp (C°)
##  Length:4603        Min.   :134.1   Min.   : 3.889
##  Class :character   1st Qu.:163.3   1st Qu.: 8.889
##  Mode  :character   Median :253.7   Median :11.111
##                     Mean   :231.6   Mean   :10.891
##                     3rd Qu.:277.5   3rd Qu.:12.778
##                     Max.   :355.1   Max.   :19.444
##                     NA's   :96
```

Do a Time Series forecast for Consumption (kW) without taking into account outdoor temperature and then do another Time series for Consumption with outdoor temperature in consideration.

B) Initate Time-series - Consumption (kW) without outdoor temperature

observations are every 15 minutes, convert it to an hour. This is 60/15 = 4, this is the rate for hourly frequency.

As observed below we also notice that the dataset where both Power(kW) and Temp (C) end on row 4507.

```
elecdata[c(4507,4508), ]
```

```
## # A tibble: 2 x 3
##   Timestamp       `Power (kW)` `Temp (C°)`
##   <chr>                  <dbl>       <dbl>
## 1 2/16/2010 23:45         150.        11.7
## 2 2/17/2010 0:00           NA         11.7
```

Thus, create a Time-series consumption from 1/10/2010 to 2/16/2010 between Row 1 and 4507 Each quarter represents a 15 min interval, and use this new data-set as a way to view that every row is an hour. This Allows for better visualization for the graphs.

```
elec_con <- ts(elecdata[1:4507,2], frequency = 4, start=c(1,2))
head(elec_con)
```

```
##    Qtr1  Qtr2  Qtr3  Qtr4
## 1       165.1 151.6 146.9
## 2 153.7 153.8 159.0
```

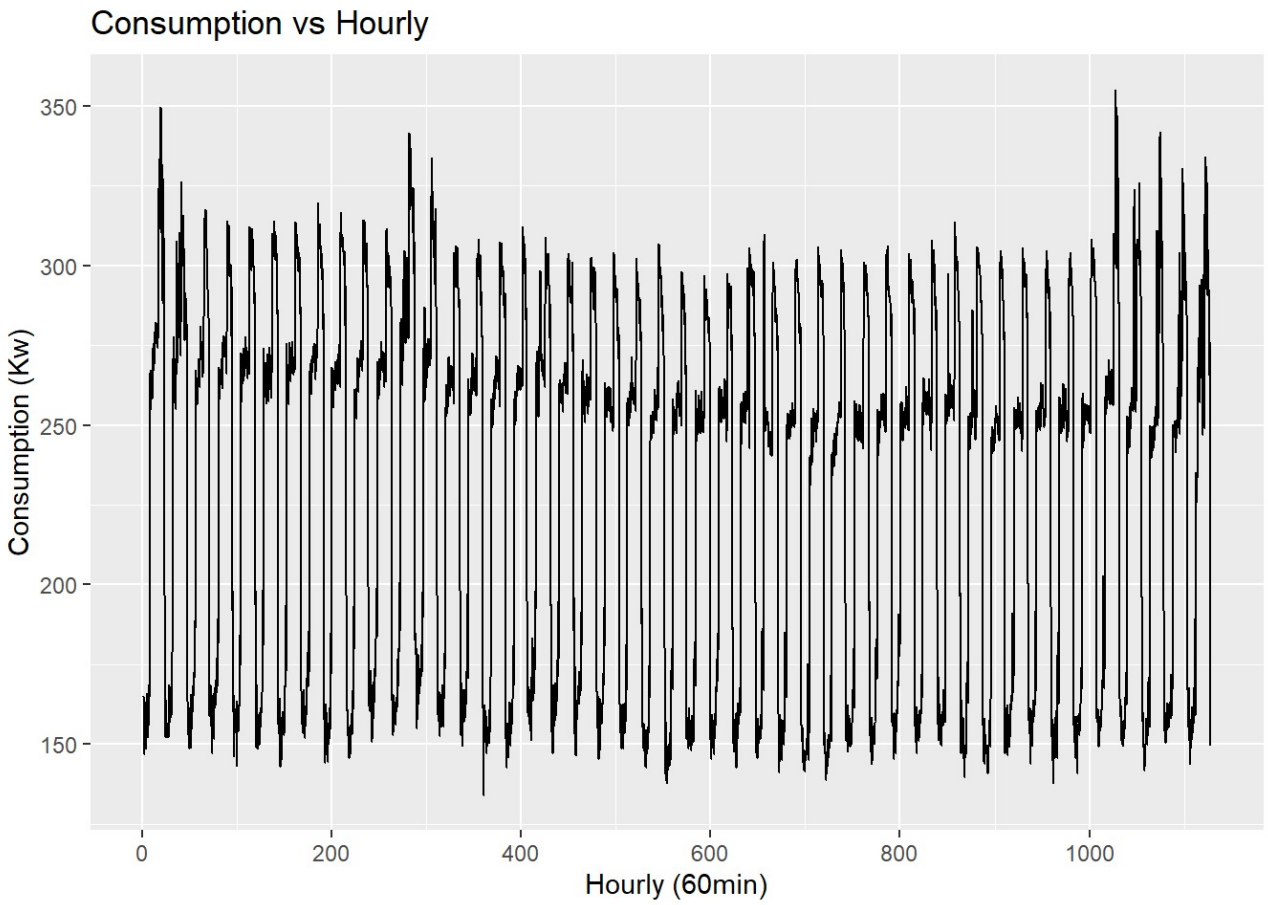
```
tail(elec_con)
```

```
##       Qtr1  Qtr2  Qtr3  Qtr4
## 1126             265.4 270.9
## 1127 276.2 192.7 187.1 149.5
```

C) Plotting the Data

Consumption vs. Hourly

```
autoplot(elec_con) +
  ggtitle('Consumption vs Hourly') +
  xlab('Hourly (60min)') +
  ylab('Consumption (Kw)')
```



D) Splitting Data

Make two sets of data, Train Data set (80%) and the Test Data set (20%), in order to run a proper time series forecast to evaluate the model appropriately, Since using the new elecdataset this will mean at 900 rows as Train and 227 rows remaining as the Test.

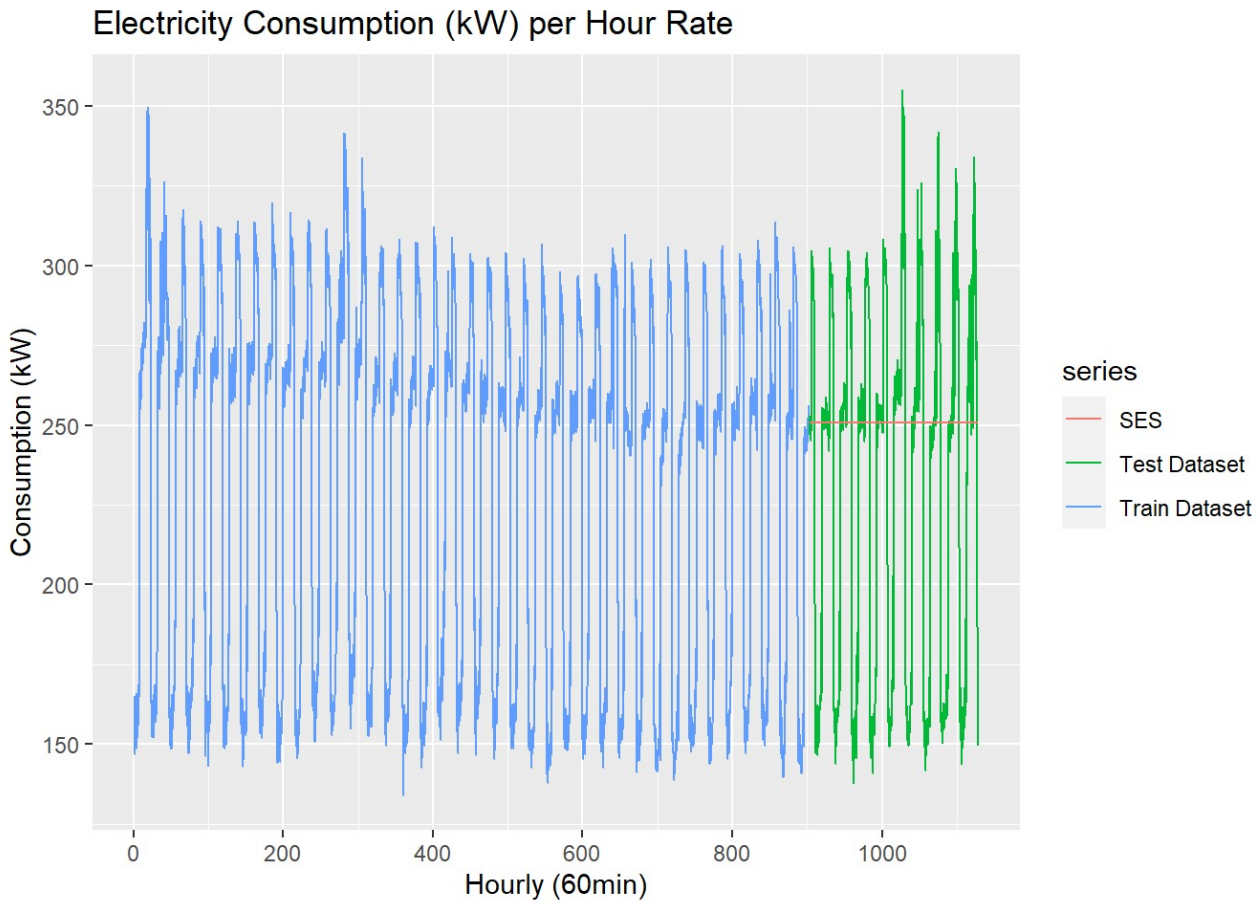
```
elec_con_train= window(elec_con, start=c(1,2), end=c(902,4))
elec_con_test= window(elec_con, start=c(903,1), end=c(1127,4))
autoplot(elec_con_train,series='Train Dataset') +
  autolayer(elec_con_test,series='Test Dataset')+
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```

E) Tesing on Different Models

Start forecasting with Simple Exponential Smoothing (SES) without a smoothing parameter in alpha.

```
elec_con_SES = ses(elec_con_train,h=900, alpha=NULL)
```

```
autoplot(elec_con_train,series="Train Dataset") +
  autolayer(elec_con_test,series='Test Dataset')+
  autolayer(elec_con_SES$mean,series='SES')+
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```



Compute the root mean square error (RMSE) of each model

```
print(sqrt(mean((elec_con_SES$mean-elec_con_test)^2)))
```

```
## [1] 60.76483
```

SES is a bad forecasting tool for this, so we move onto more sophisticated models with more parameters. Such as the usage of Holt-Winters Model

Additive Seasonal HW

```
elec_con_HW_add = hw(elec_con_train, seasonal='additive',h=900)
```

Damped additive seasonal HW

```
elec_con_DHW_add = hw(elec_con_train, seasonal='additive',h=900,damped=TRUE)
```

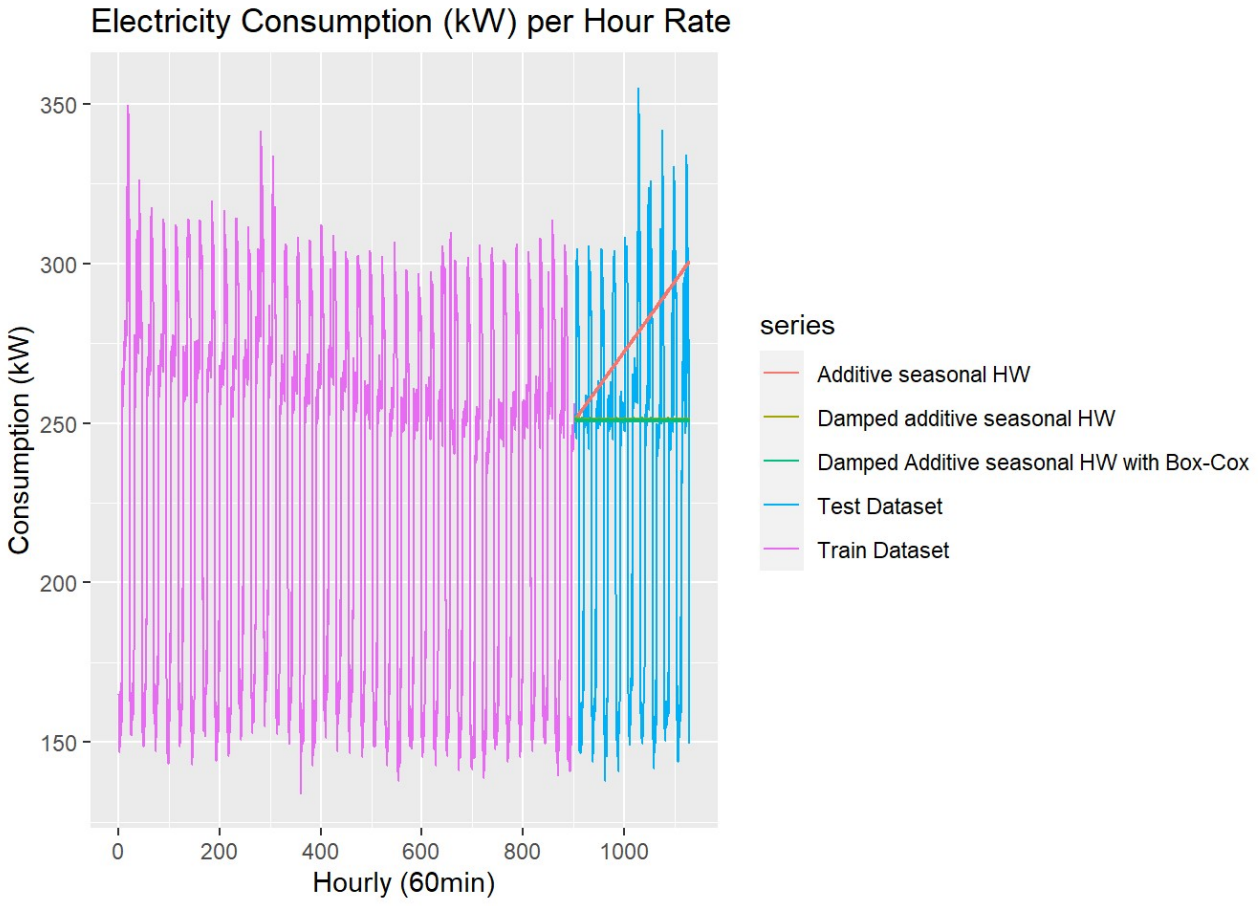
and

Add Box-Cox Transformation for stabilization purposes to the the Damped additive seasonal HW

```
elec_con_DHW_addBC = hw(elec_con_train, seasonal='additive',h=900,damped=TRUE,
lambda = 'auto')
```

Plot both Models to view their forecasting

```
autoplot(elec_con_train,series="Train Dataset") +
  autolayer(elec_con_test,series='Test Dataset')+
  autolayer(elec_con_HW_add$mean,series='Additive seasonal HW')+
  autolayer(elec_con_DHW_add$mean,series='Damped additive seasonal HW')+
  autolayer(elec_con_DHW_addBC$mean,series='Damped Additive seasonal HW with Bo
x-Cox')+
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```



Compute the root mean square error (RMSE) of each model

```
print(sqrt(mean((elec_con_HW_add$mean-elec_con_test)^2)))
```

```
## [1] 73.46383
```

```
print(sqrt(mean((elec_con_DHW_add$mean-elec_con_test)^2)))
```

```
## [1] 60.91925
```

```
print(sqrt(mean((elec_con_DHW_addBC$mean-elec_con_test)^2)))
```

```
## [1] 60.84712
```

There is not much difference comparing the RMSE errors for all different types of Holt-Winters models, As observed, the errors of both the Damped HW and Damped HW with Box-Cox remain the same as the Exponential Smoothing. Additionally, the Additive Seasonal HW error performs the worse, this may be due to the fact they are linear models. One more thing to note, as seen for Box-Cox, the Lambda is set to automatically, and not chosen which may be the case why it performs badly.

Forecast with ARIMA Model to see if there is any difference to previous models

```
elec_con_ARIMA = auto.arima(elec_con_train)
pred_elec_con_ARIMA = forecast(elec_con_ARIMA,h=900)
autoplot(elec_con_train,series="Train Dataset") +
  autolayer(elec_con_test,series='Test Dataset')+
  autolayer(pred_elec_con_ARIMA,series='ARIMA',PI=FALSE)+
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```

visually the Model predicts the Test set better than previous models observed.

```
print(sqrt(mean((pred_elec_con_ARIMA$mean-elec_con_test)^2)))
```

```
## [1] 56.392
```

Upon checking with the RMSE error. although the Error is lower than previous model, it still doesn't do good predictions as the model is able to forecast with very little significance, it then converges to zero as the ARIMA usually used in a stationary data set.

An Augmented Dicker Fuller Test, can prove that the this data-set is indeed stationary as the P-value is less than .05, Reject the Null hypothesis in other words it has some time-dependent structure and does not have a constant variance over time

```
adf.test(elec_con_train)
```

```
## Warning in adf.test(elec_con_train): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  elec_con_train
## Dickey-Fuller = -12.616, Lag order = 15, p-value = 0.01
## alternative hypothesis: stationary
```

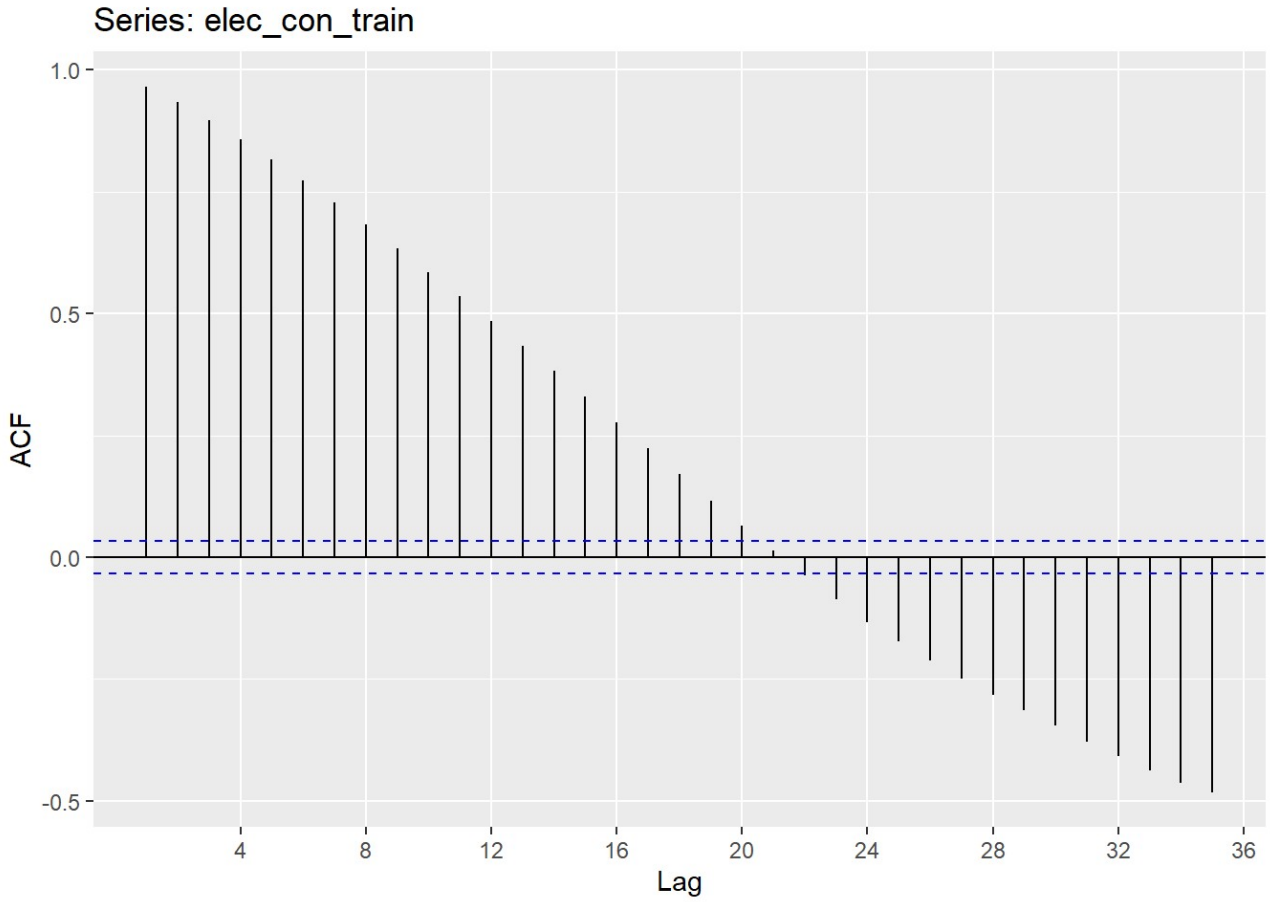
```
adf.test(elec_con_test)
```

```
## Warning in adf.test(elec_con_test): p-value smaller than printed p-value
```

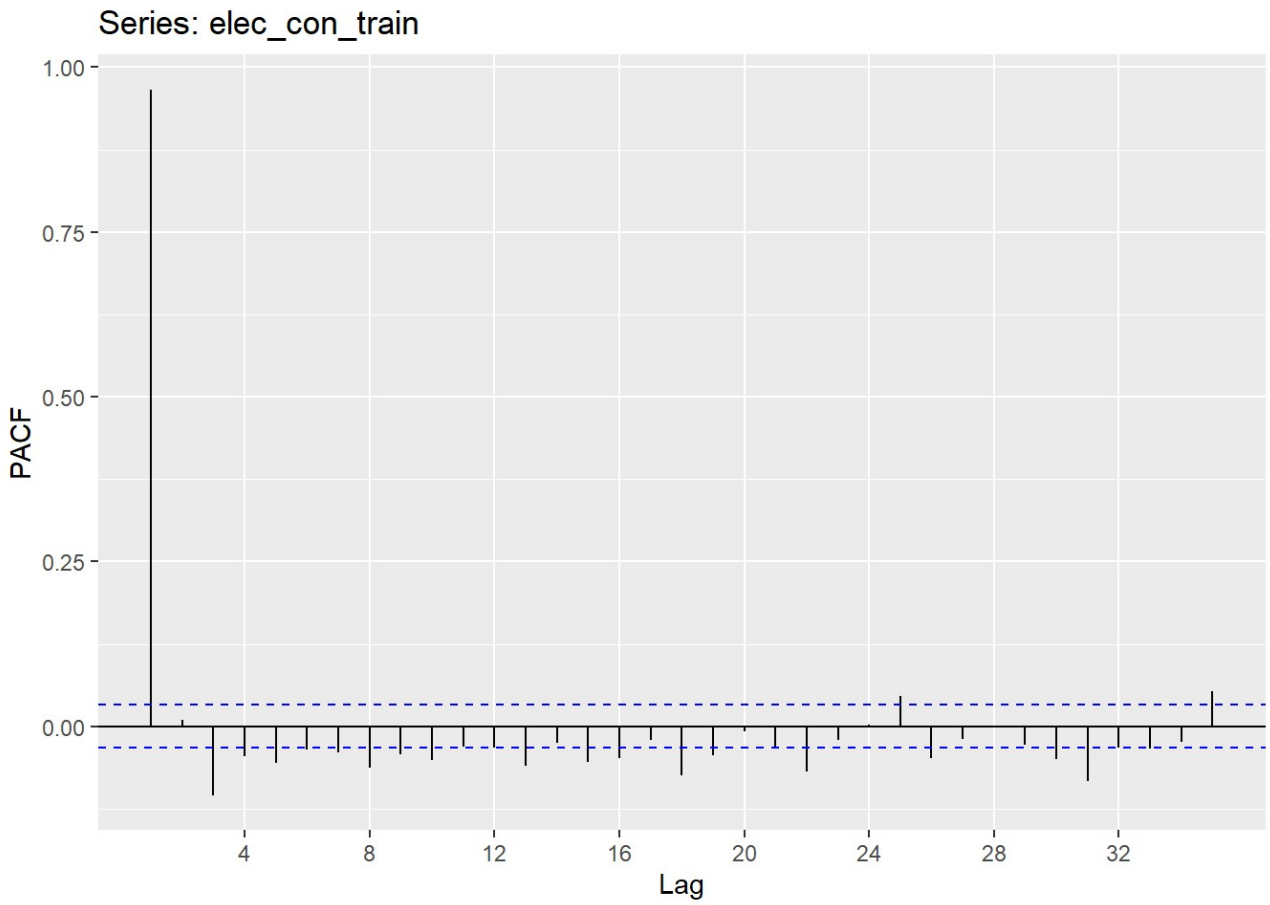
```
##
##  Augmented Dickey-Fuller Test
##
## data:  elec_con_test
## Dickey-Fuller = -5.5658, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

To proceed, a Correlation test can be used to identify patterns using the Auto-correlation plot and Partial Auto-correlation plot.

```
ggAcf(elec_con_train)
```




```
ggPacf(elec_con_train)
```



Theoretically, autocorrelation is the correlation between two values in a time series, the lags represent points of time in the data-set. As observed, the lags are sitting outside the 95% confidence interval which can mean that the lags may correlate between themselves. The autocorrelation function declines to near zero rapidly for a stationary time series. In this instance ACF and PACF prove this.

Lastly, it is possible to use a Neural Network to make better predictions of the Test Data set.

```
elec_con_train_NN = nnetar(elec_con_train)
pred_elec_con_train_NN = forecast(elec_con_train_NN, h = 900)
autoplot(elec_con_train,series="Train Dataset") +
  autolayer(elec_con_test,series='Test Dataset')+
  autolayer(pred_elec_con_train_NN$mean,series='Neural Network')+
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```

Visually, this model is likely the best model when comparing to the previous models for making predictions.

```
print(sqrt(mean((pred_elec_con_train_NN $mean-elec_con_test)^2)))
```

```
## [1] 80.05763
```

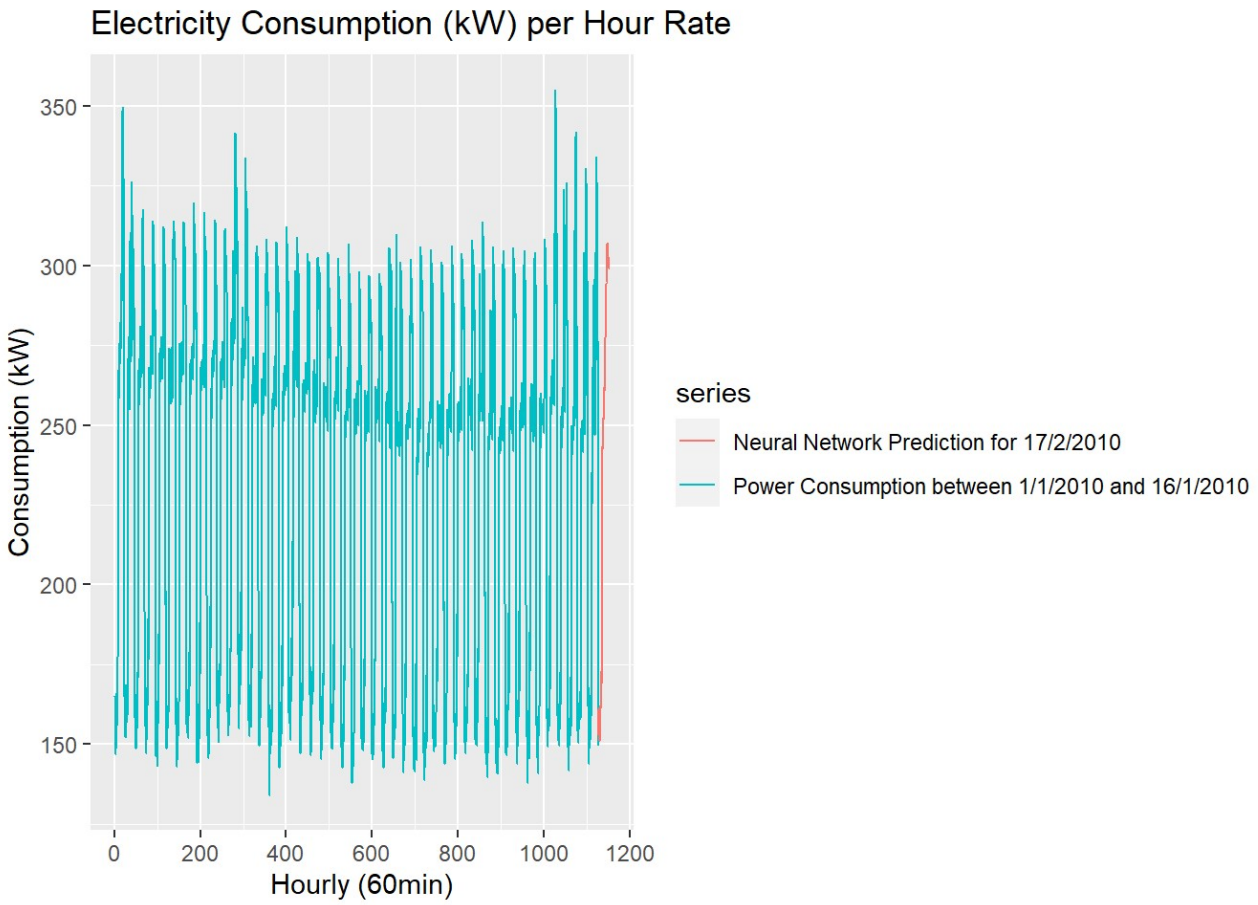
Computing the RMSE, gives an error rate that is the most least compared to previous models. Itis acceptable as it is doing a better prediction.

For better observations of predictions from the Neural Network, here the NN is predicted against the Test Dataset. It is not particularly the best at modelling, but it far more dominant model compared to previous ones. This particular model will be used to forecast predictions in consumption

```
autoplot(elec_con_test,series='Test Dataset') +
  autolayer(pred_elec_con_train_NN$mean,series='Neural Network')+
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```

96 observations to represent 24 hours as an entire day. Predictions are made to forecast consumption for the the next day for 17/2/2010.

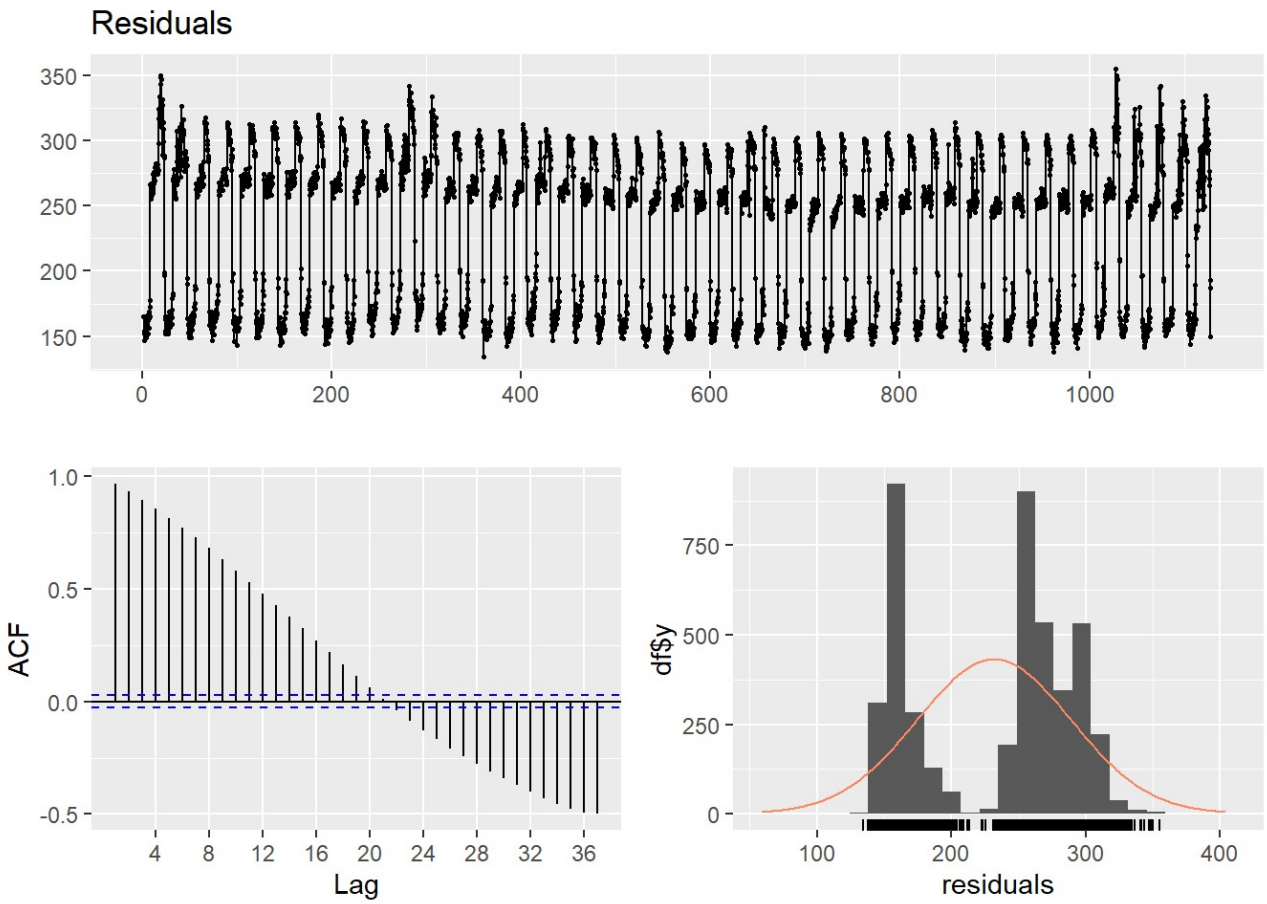
```
elec_con_2 = nnetar(elec_con, lambda = 'auto')
pred_con = forecast(elec_con_2, h = 96)
autoplot(elec_con,series="Power Consumption between 1/1/2010 and 16/1/2010") +
  autolayer(pred_con$mean,series='Neural Network Prediction for 17/2/2010')+
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```



Lastly, to assume that the residuals are independent and identically distributed.

```
checkresiduals(elec_con)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



File is saved

F) Initate Time-series - Consumption (kW) with outdoor temperature into consideration

Again, from 1/10/2010 to 2/16/2010 between Row 1 and 4507 Each quarter represents a 15 min interval, and we will use this new data-set as a way to view that every row is an hour.

```
temp <- ts(elecdata[1:4507,3], frequency = 4, start=c(1,2))
head(temp)
```

```
##       Qtr1     Qtr2     Qtr3     Qtr4
## 1          10.55556 10.55556 10.55556
## 2 10.55556 10.55556 10.55556
```

```
tail(temp)
```

```
##          Qtr1     Qtr2     Qtr3     Qtr4
## 1126                     11.66667 11.66667
## 1127 11.66667 11.66667 11.66667 11.66667
```

G) Splitting Data

Again here. Make two sets of data, Train Data set (80%) and the Test Data set (20%), in order to run a proper time series forecast to evaluate the model appropriately Since we are using the new elecdataset this will mean at 900 rows as Train and 227 rows remaining as the Test.

```
temp_train=window(temp, start=c(1,2), end=c(902,4))
temp_test=window(temp, start=c(903,1), end=c(1127,4))
```

H) Time-Series Linear Regression Model

In this part, a time-series regression model will first need to be made in order to prove the effect of outdoor temperature on electricity consumption (kW) exists.

```
temp_con_train=tslm(elec_con_train~temp_train)
summary(temp_con_train)
```

```
##
## Call:
## tslm(formula = elec_con_train ~ temp_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -120.990  -42.802    3.352   42.980  111.862
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 126.9146     3.5111   36.15   <2e-16 ***
## temp_train    9.8478     0.3201   30.76   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 51.23 on 3605 degrees of freedom
## Multiple R-squared:  0.2079, Adjusted R-squared:  0.2077
## F-statistic: 946.4 on 1 and 3605 DF,  p-value: < 2.2e-16
```

The P-value is less than .05, which means that we reject Null Hypothesis, and state that there exists an effect of outdoor temperature on electricity consumption (kW).

To further validate, Season is added to the existing Time-series regression model to see if there is any effect on this data-set.

```
temp_con_train_season=tslm(elec_con_train~temp_train + season)
summary(temp_con_train_season)
```

```
##
## Call:
## tslm(formula = elec_con_train ~ temp_train + season)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -121.341  -42.971    3.355   43.045  112.449
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 126.6115     3.8102  33.229   <2e-16 ***
## temp_train    9.8478     0.3202  30.752   <2e-16 ***
## season2       0.8423     2.4139   0.349    0.727
## season3       0.6544     2.4139   0.271    0.786
## season4      -0.2830     2.4139  -0.117    0.907
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 51.25 on 3602 degrees of freedom
## Multiple R-squared:  0.208,  Adjusted R-squared:  0.2071
## F-statistic: 236.5 on 4 and 3602 DF,  p-value: < 2.2e-16
```

P value in this instance is more than .05, thus accept Null hypothesis, season is not statistically significant. Season 2, Season 3, and Season 4, thus meaning that season plays no impact on this Time Series model

Sample Coefficient of Variation is used to see the scores.

```
CV(temp_con_train_season)
```

```
##           CV          AIC         AICc          BIC        AdjR2
## 2.629742e+03 2.840643e+04 2.840646e+04 2.844358e+04 2.071225e-01
```

Furthermore, Trend can also be added instead to the existing Time-series regression model to see if there is any effect on this data-set.

```
temp_con_train_trend=tslm(elec_con_train~temp_train + trend)
summary(temp_con_train_trend)
```

```
##
## Call:
## tslm(formula = elec_con_train ~ temp_train + trend)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -125.686  -42.643    2.713   43.220  121.808
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.367e+02  3.554e+00   38.46   <2e-16 ***
## temp_train   1.052e+01  3.199e-01   32.87   <2e-16 ***
## trend       -9.358e-03  8.186e-04  -11.43   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 50.33 on 3604 degrees of freedom
## Multiple R-squared:  0.2357, Adjusted R-squared:  0.2352
## F-statistic: 555.6 on 2 and 3604 DF,  p-value: < 2.2e-16
```

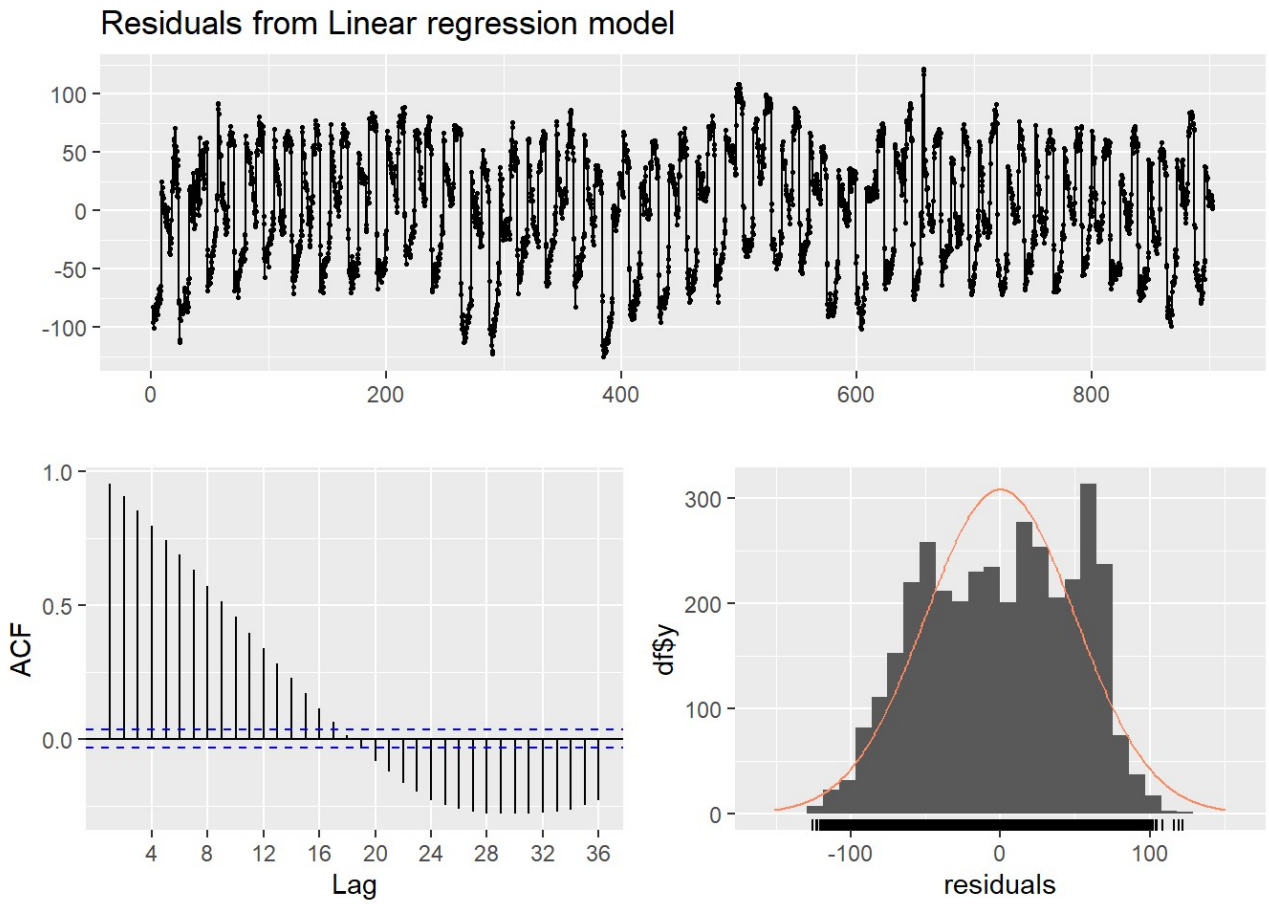
The P-value is less than .05, which means that we reject Null Hypothesis, and state that there exists an effect of Trend pattern on electricity consumption (kW), meaning it is Statistically significant.

Sample Coefficient of Variation is used to see the scores.

```
CV(temp_con_train_trend)
```

```
##           CV          AIC         AICc          BIC        AdjR2
## 2.535074e+03 2.827426e+04 2.827427e+04 2.829902e+04 2.352279e-01
```

```
checkresiduals(temp_con_train_trend,test="LB",plot=TRUE)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Linear regression model
## Q* = 17482, df = 5, p-value < 2.2e-16
##
## Model df: 3.   Total lags used: 8
```

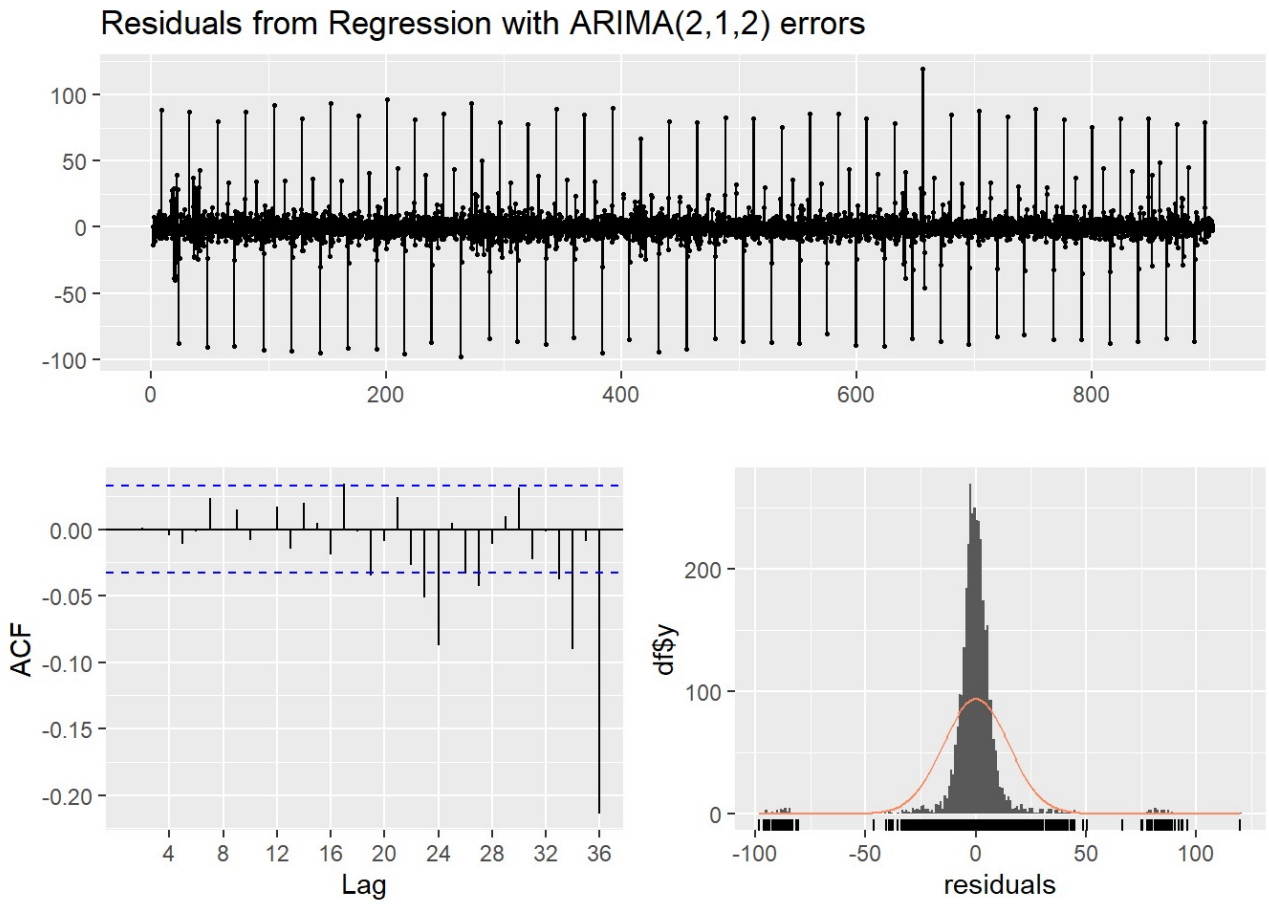
Here the residual are correlated, which means that this regression model (which assumes independent residuals) is not appropriated.

Instead, its recommended to then form Dynamic regression model modelizes the residuals with an ARIMA p,d,q model. The choice of the orders p, d, q can be done by examining the residuals or automatically with the auto.arima function.

```
temp_con_train_arima = auto.arima(elec_con_train, xreg=temp_train)
```

Checking resuiduals and A L-jung test is performed with the ARIMA model.

```
checkresiduals(temp_con_train_arima, test='LB', plot = TRUE)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,1,2) errors
## Q* = 2.5285, df = 3, p-value = 0.4702
##
## Model df: 5.   Total lags used: 8
```

With an ARIMA model of (2,1,2): Auto-correlations of the residuals have modeled

A L-jung P-value > 0.05, can make the assumption that these values are not dependent.

H) Dynamic Regression and Neural Network Model with Outdoor Temperature

Since the Nueral network model was a good model to make future predictions, this model will be used to forecast future predictions to also include Outdoor Temperature

Forecast predictions with ARIMA mode on Test Data-set

```
pred_test_temp = forecast(temp_con_train_arima, xreg = temp_test, h=900)

elec_con_train_NN_temp = nnetar(elec_con_train, lambda = 'auto', xreg = temp_tr
ain)
pred_elec_con_train_NN_temp = forecast(elec_con_train_NN_temp, h=900, xreg =tem
p_test)

autoplot(elec_con_train, series='Train Dataset') +
  autolayer(elec_con_test, series='Test Dataset') +
  autolayer(pred_elec_con_train_NN_temp$mean,series='Neural Network with Temper
ature') +
  autolayer(pred_test_temp$mean, series='Dynamic Rgression Model with Temperatu
re') +
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```

Following previos step, a Time series for temperature data set is created

```
temp_forecast <- ts(elecdata[4509:4603,3], frequency = 4, start=c(1,2))
head(temp_forecast)
```

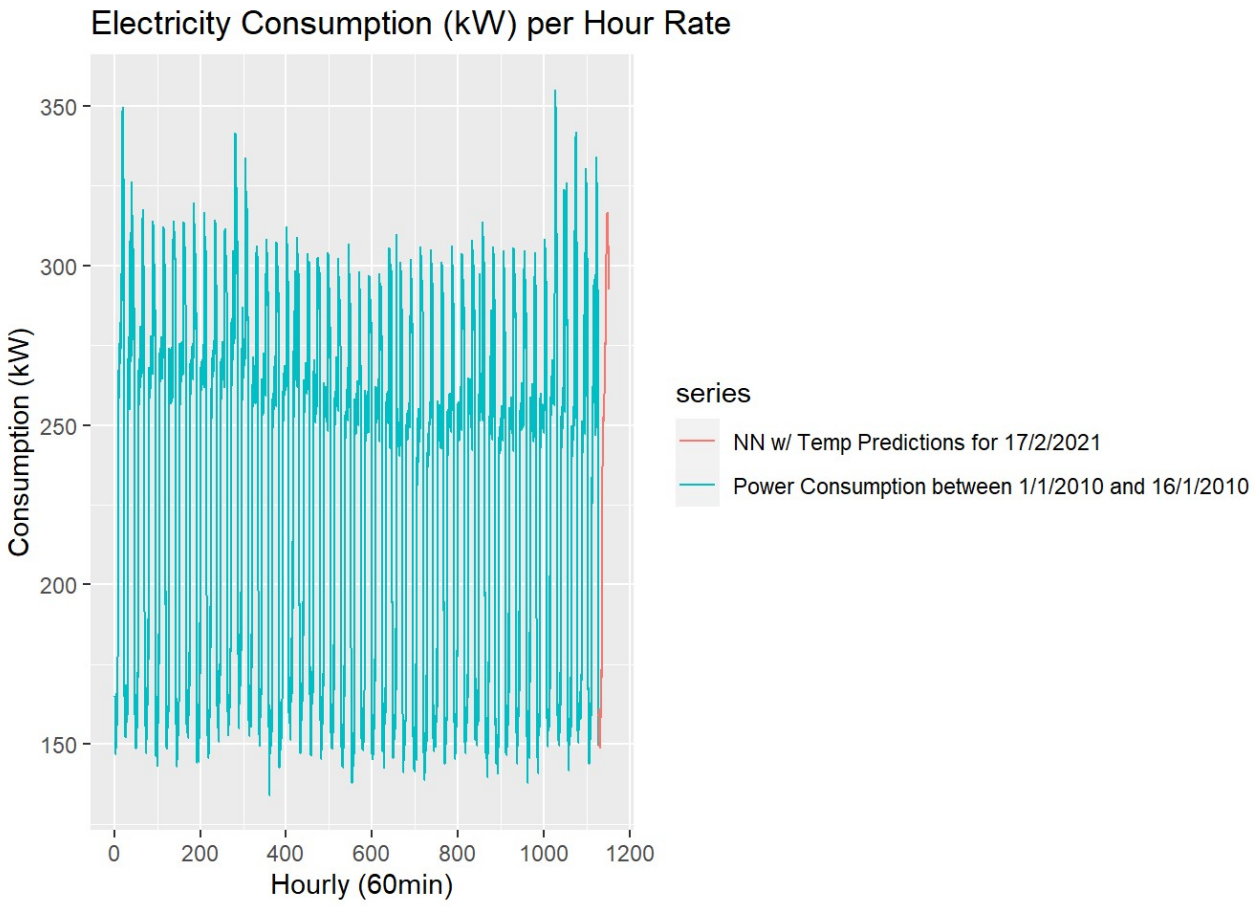
```
##       Qtr1     Qtr2     Qtr3     Qtr4
## 1          11.11111 11.11111 11.11111
## 2 11.11111 10.55556 10.55556
```

```
tail(temp_forecast)
```

```
##        Qtr1     Qtr2     Qtr3     Qtr4
## 23                    13.88889 13.88889
## 24 13.88889 12.77778 12.77778 12.77778
```

```
elec_con_train_NN_temp = nnetar(elec_con, lambda = 'auto', xreg = temp)
pred_elec_con_train_NN_temp = forecast(elec_con_train_NN_temp, h=900, xreg =tem
p_forecast)

autoplot(elec_con,series="Power Consumption between 1/1/2010 and 16/1/2010") +
  autolayer(pred_elec_con_train_NN_temp$mean,series='NN w/ Temp Predictions fo
r 17/2/2021') +
  ggtitle ('Electricity Consumption (kW) per Hour Rate') +
  xlab('Hourly (60min)') +
  ylab('Consumption (kW)')
```



Save the File

In Conclusion: The Neural Network model is the best model amongst other models to predict and forecast Consumption of kW with Temperature, statistical analysis like this can allow data scientists understand the rules of understanding time series with different factors.