ISSN 2347 - 3983
Volume 8. No. 9, September 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter203892020.pdf
https://doi.org/10.30534/ijeter/2020/203892020

# Data Prediction of Reject Unit from Manufacturing Company using Auto Regressive Integrated Moving Average (ARIMA) Algorithm

**Ahamad Zaki Mohamed Noor[1], Muhammad Hafidz Fazli Md Fauadi[2], Fairul Azni Jafar[2], Kauthar A Rhaffor[3]**

[1]System Engineering and Energy Laboratory, Universiti Kuala Lumpur Malaysian Spanish Institute, Kulim Hi – Tech Park, 09000, Kulim, Kedah, Malaysia, ahamadzaki@unikl.edu.my

[2]Centre of Smart System and Innovative Design, Faculty of Manufacturing Engineering, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100, Durian Tunggal, Melaka, Malaysia, hafidz@utem.edu.my, fairul@utem.edu.my

[3]Manufacturing Section, Universiti Kuala Lumpur Malaysian Spanish Institute, Kulim Hi – Tech Park, 09000, Kulim, Kedah, Malaysia, kauthar@unikl.edu.my

## ABSTRACT

Big data is one of the nine pillars available in the nine pillars of industrial revolution 4.0. Malaysia came up with this incentive under the Ministry of International Trade and Industry whereby called Industry 4WRD. Problem faced by the industry people is not utilize the data obtain well. Rejects unit keeps on piling however, the data was not utilize to determine the trend of reject unit decrease. The objective of this research is to perform data prediction using ARIMA algorithm. The data were acquired, wrangled, explored, model and visualized in order to perform data prediction on reject units. The dataset on reject unit for 10 years respective to each month were obtained. ARIMA algorithm were utilized shows that the p – value decrease from 0.33 to 0.31. From these values, the next stage shows the prediction of reject units significantly decrease in the year 2021.

**Key words:** ARIMA, Big Data, Reject Units, Time – Series Analysis,

## 1. INTRODUCTION

World currently are moving towards industry revolution 4.0 (IR4.0). This defines as a production plant that operates with minimum labor size. Some country may use eleven pillars or nine pillars to fulfil the needs of IR4.0. All the nine pillars must converge. One of the highest importance from the nine pillar are big data. This big data serves as an algorithm to plot, optimize and predict the future of product output. A review was made to determine selection using artificial intelligence [1]. A research was done by hybrid fuzzy logic to Analytic Hierarchy Process (AHP) [2]. This shows that current trend are moving towards the needs of implementing IR 4.0. Problem faced nowadays among the manufacturer is to determine the number of reject. Each month, number of reject may increase and decrease. However, there is a need for the manufacturer to determine the time duration for the loss incur. This research is to utilize an algorithm for data prediction of reject produced by manufacturer. A set of rejects unit data extracted from a manufacturing company were used in this research. The software used for data analytics in this research is Python software. Previous experiment conducted by researchers by utilizing ARIMA. A research was conducted to predict the sea pattern in terms of speed and direction [3]. Other application is to observe the abnormal household water flow rate using ARIMA technique [4]. Next applications are subway passenger flow prediction [5] and passenger flow prediction in bus transportation system [6] both using ARIMA. There are five main stages need to be performed in data analytics by data scientist as shown in Figure 1.

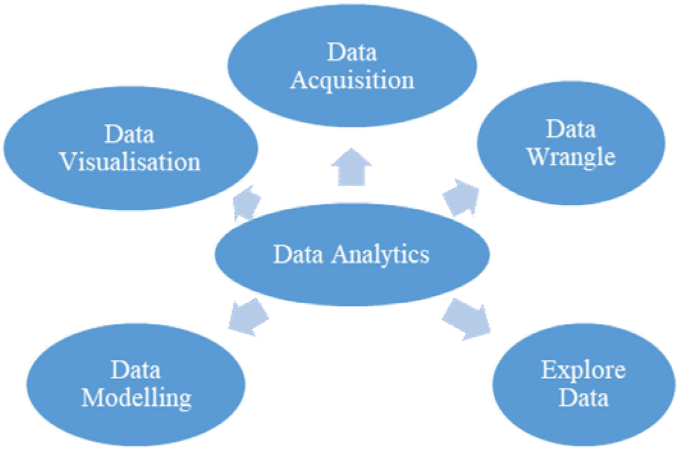


**Figure 1:** Data Analyst Tasks

The stages are data acquisition, data wrangle, explore, data modelling and data visualization. Example of a research conducted to assess the integrity of traffic data through short-term state prediction using ARIMA [7]. There are also other researchers conducted K – Nearest Neighbours technique in the implementation of subspace outlier[8]. Same concept of big data was utilized in cloud system[9] whereby needed numerous data need to be categorized and optimized for better visualization. For data acquisition, python were used. In order to wrangle the data, pandas package were utilized. To explore the data, matplotlib was used to plot. For data modelling, the package used were numPy and lastly to visualize the data, sciPy package used. In this research, an algorithm adapting to all five stages known as Auto Regressive Integrated Moving Average (ARIMA). This ARIMA model or algorithm used to model time series data using Python. ARIMA consist of three parameters that are auto regressive lags (P), differentiation order (d) and moving average (Q). Auto Regressive (AR) is the correlation between past and present time. Hence, Moving Average (MA) integrated with AR to reduce the noise.

## 2. METHODOLOGY

There are numerous steps before a prediction executed. Python library need to updated suit to the requirement of data analytics. The packages were install via anaconda command prompt. The packages are numpy, pandas, matplotlib and rcParams.

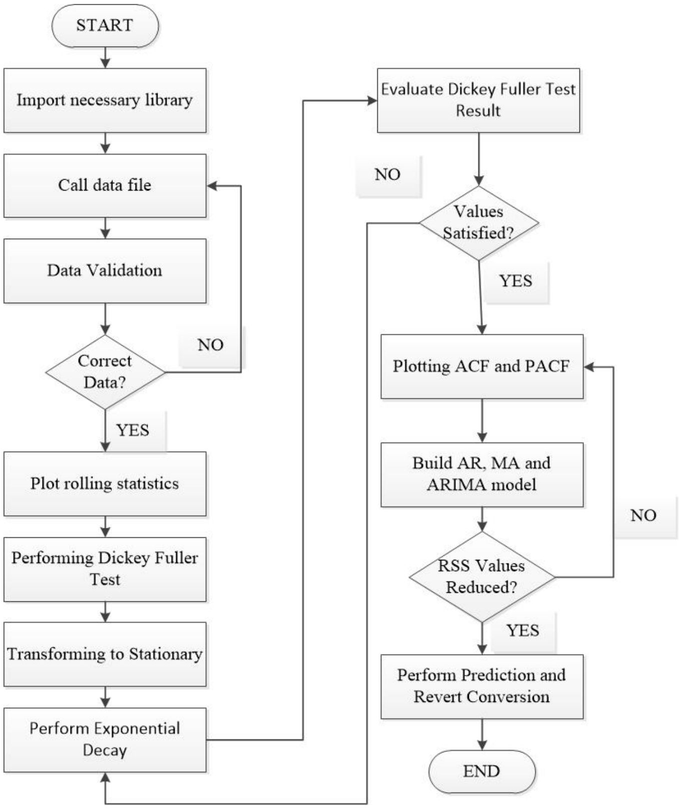

START
Import necessary library
Call data file
Data Validation
Correct Data?
NO
YES
Plot rolling statistics
Performing Dickey Fuller Test
Transforming to Stationary
Perform Exponential Decay
Evaluate Dickey Fuller Test Result
NO
Values Satisfied?
YES
Plotting ACF and PACF
Build AR, MA and ARIMA model
NO
RSS Values Reduced?
YES
Perform Prediction and Revert Conversion
END

**Figure 2:** ARIMA Algorithm Flowchart

From the flow chart in Figure 2, necessary packages were imported in the python library. Next step is to call the data file. The file consist of number of reject from early January 2000 until December 2011. There are a total of 144 months. The file imported were validated using python command. The python command used to check the imported data were head or tail. Head is to verify the data from the top. The tail command is to verify the data from the bottom. This tail or head verification is the third phase of this experiment.

Once the data file validated, the fourth phase of data analytics is to determine rolling statistics. The important component used for consideration are mean and standard deviation. The fifth phase of this experiment is to perform Dickey – Fuller test. The purpose of this phase is to test null hypothesis that a unit root present in autoregressive model. Alternative hypothesis is different depend on the version of test. However, the test outcome may show stationary or trend stationary. The next phase of experiment after Dickey – Fuller test is transform to stationary. This phase conducted by estimating trend by plotting in log scale.

Further proceed with this experiment, the seventh phase is to evaluate the result of Dickey – Fuller test. In order to obtain high accuracy, the critical value must be close to test statistics. If the value is not satisfied, then log scale the data or exponential decay weighted average was performed to obtained stationary time series. Python visualize the data with plot function for ease of comparing the log scale with exponential decay. If the result shown not better, then, this research is continued with log transformation. The p – value from previous Dickey Fuller must be less than the new Dickey Fuller test.

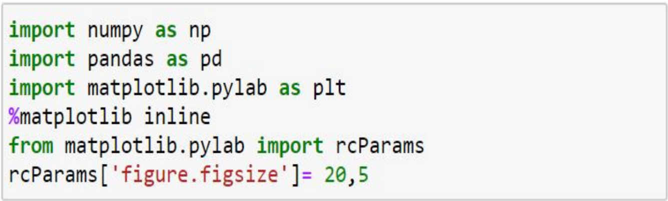
The ninth phase of this experiment is to plot Auto Correlation Function (ACF) and Partial Auto Correlation Function (PACF). X value is evaluated at both plot when y = 0. Once evaluation of Auto Correlation and Partial Auto Correlation Function was evaluated, this research moves to phase ten whereby Auto Regressive (AR), Moving Average (MA) and Auto Regressive Integrated Moving Average (ARIMA) were plotted. The Residual Sum of Squares (RSS) must obtain low value. Low value gives better AR, MA and ARIMA models. The final step of this experiment is to proceed with the prediction and revert conversion of rejects part. The prediction was manipulated by changing number of steps. For this data, a step defines a month.

## 3. RESULT AND DISCUSSION

This section breaks the output according to phases shown in research flow chart from Figure 2. Discussion were made on every output with respect to each phase outlined in the flow chart.

### 3.1 Phase 1: Import Necessary Library

Figure 3 shows the library imported for the ease of performing time series analysis. Numpy is a function that provide the program to compute multidimensional array. This function allows the user to manipulate these arrays.



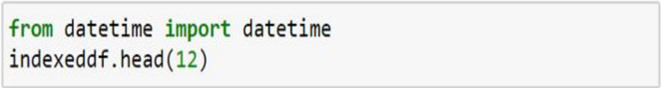
```
import numpy as np
import pandas as pd
import matplotlib.pylab as plt
%matplotlib inline
from matplotlib.pylab import rcParams
rcParams['figure.figsize']= 20,5
```

**Figure 3:** Import library to python

Pandas were import also for the ease of statistic computation. Matplotlib is a function to plot data for the user to get better data visualization and rcParams is to declare the size of the flowchart. The size is manipulated according to the suitability of the user.

### 3.2 Phase 2: Call Data File

This phase shows the step to call data using python web programming. The data that was called is the date time and the rejects unit. Figure 4 shows the programming for recall purpose. The file must be in the same directory as the saved python web programming.



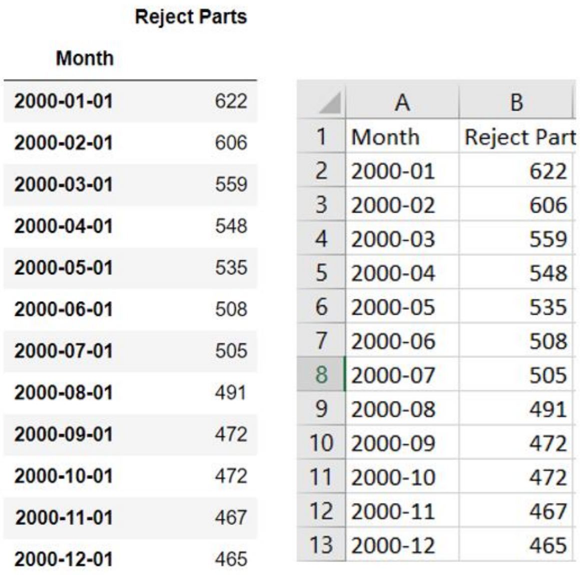
```
from datetime import datetime
indexeddf.head(12)
```

**Figure 4:** Call Data File

From the program, head is refer to the data at the top of the excel sheet, and tail refer to the bottom of the excel sheet. The number twelve indicates the number of data to be viewed in python software.

### 3.3 Phase 3: Data Validation

Phase 3 of data validation is to verify the called data and the content of excel sheet. Observe from Figure 5, there are 12 months of data belongs to year 2000 along with the reject parts. These data uses head command in python program.

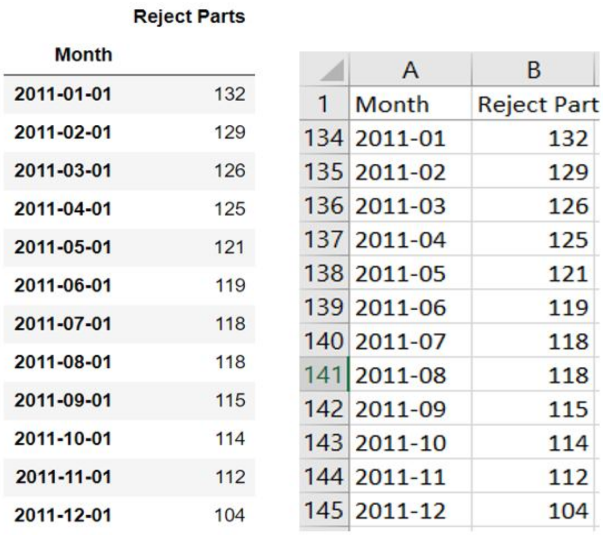


| Month | Reject Parts |
|---|---|
| 2000-01-01 | 622 |
| 2000-02-01 | 606 |
| 2000-03-01 | 559 |
| 2000-04-01 | 548 |
| 2000-05-01 | 535 |
| 2000-06-01 | 508 |
| 2000-07-01 | 505 |
| 2000-08-01 | 491 |
| 2000-09-01 | 472 |
| 2000-10-01 | 472 |
| 2000-11-01 | 467 |
| 2000-12-01 | 465 |

| | A | B |
|---|---|---|
| 1 | Month | Reject Part |
| 2 | 2000-01 | 622 |
| 3 | 2000-02 | 606 |
| 4 | 2000-03 | 559 |
| 5 | 2000-04 | 548 |
| 6 | 2000-05 | 535 |
| 7 | 2000-06 | 508 |
| 8 | 2000-07 | 505 |
| 9 | 2000-08 | 491 |
| 10 | 2000-09 | 472 |
| 11 | 2000-10 | 472 |
| 12 | 2000-11 | 467 |
| 13 | 2000-12 | 465 |

**Figure 5:** Data Validation (Head)

Further validate this file, another validation was carried out to observe the tail section of the file. Figure 6 shows the comparison of 12 months of data belongs to year 2011 along with reject parts. Each row were compared. Since the data called were correct, there is no necessity to recall the proper file again.



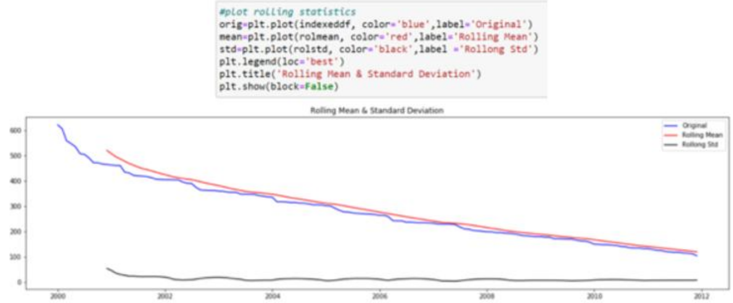
| Month | Reject Parts |
|---|---|
| 2011-01-01 | 132 |
| 2011-02-01 | 129 |
| 2011-03-01 | 126 |
| 2011-04-01 | 125 |
| 2011-05-01 | 121 |
| 2011-06-01 | 119 |
| 2011-07-01 | 118 |
| 2011-08-01 | 118 |
| 2011-09-01 | 115 |
| 2011-10-01 | 114 |
| 2011-11-01 | 112 |
| 2011-12-01 | 104 |

| | A | B |
|---|---|---|
| 1 | Month | Reject Part |
| 134 | 2011-01 | 132 |
| 135 | 2011-02 | 129 |
| 136 | 2011-03 | 126 |
| 137 | 2011-04 | 125 |
| 138 | 2011-05 | 121 |
| 139 | 2011-06 | 119 |
| 140 | 2011-07 | 118 |
| 141 | 2011-08 | 118 |
| 142 | 2011-09 | 115 |
| 143 | 2011-10 | 114 |
| 144 | 2011-11 | 112 |
| 145 | 2011-12 | 104 |

**Figure 6:** Data Validation (Tail)

### 3.4 Phase 4: Plot Rolling Statistics

This phase require to determine the mean and standard deviation from the obtained data. The purpose of this phase is for the ease before conducting Dickey – Fuller test. Figure 7 shows plot of reject units against years.



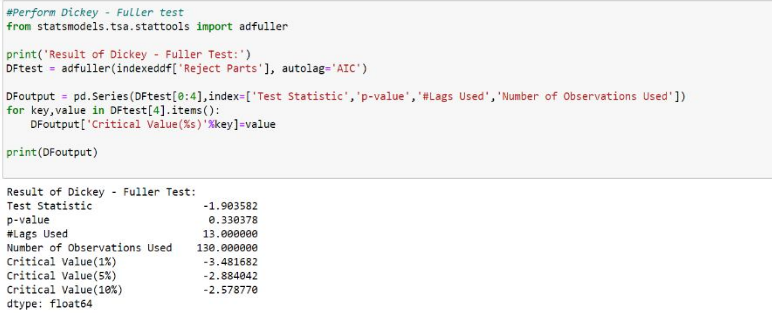
```
#plot rolling statistics
orig=plt.plot(indexeddf, color='blue',label='Original')
mean=plt.plot(rolmean, color='red',label='Rolling Mean')
std=plt.plot(rolstd, color='black',label ='Rollong Std')
plt.legend(loc='best')
plt.title('Rolling Mean & Standard Deviation')
plt.show(block=False)
```

**Figure 7:** Plotting Rolling Mean and Standard Deviation

From Figure 7, the mean was plot using red line and the standard deviation was plot with black line. The rolling mean and standard deviation were plotted starting 2001 due to insufficient data for computation for the year 2000.

### 3.5 Phase 5: Performing Dickey – Fuller Test

The purpose of this phase is to test null hypothesis that a unit root present in autoregressive model. Alternative hypothesis is different depend on the version of test. However, the test outcome may show stationary or trend stationary.



```
#Perform Dickey - Fuller test
from statsmodels.tsa.stattools import adfuller

print('Result of Dickey - Fuller Test:')
DFtest = adfuller(indexeddf['Reject Parts'], autolag='AIC')

DFoutput = pd.Series(DFtest[0:4],index=['Test Statistic','p-value','#Lags Used','Number of Observations Used'])
for key,value in DFtest[4].items():
    DFoutput['Critical Value(%s)'%key]=value

print(DFoutput)
```

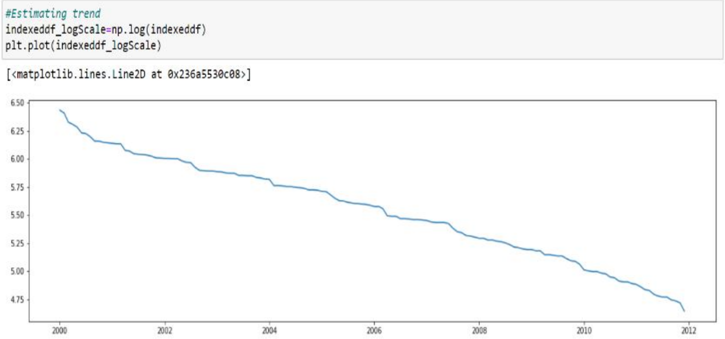
```
Result of Dickey - Fuller Test:
Test Statistic                  -1.903582
p-value                          0.330378
#Lags Used                      13.000000
Number of Observations Used    130.000000
Critical Value(1%)              -3.481682
Critical Value(5%)              -2.884042
Critical Value(10%)             -2.578770
dtype: float64
```

**Figure 8:** Dickey – Fuller Test Result

The current p – value is 0.330378 showing the plot or time series is not stationary. From the result shown in Figure 8, p – value is large than any of the critical values. None critical value are close to test statistic value. Hence, the time – series analysis is not stationary.

### 3.6 Phase 6: Transforming to Stationary

Since the Dickey – Fuller test shows not stationary, the data plot was computed again using Log scale. This phase was conducted in order to achieve smooth line compared to previous Figure 7.



```
#Estimating trend
indexeddf_logScale=np.log(indexeddf)
plt.plot(indexeddf_logScale)
```

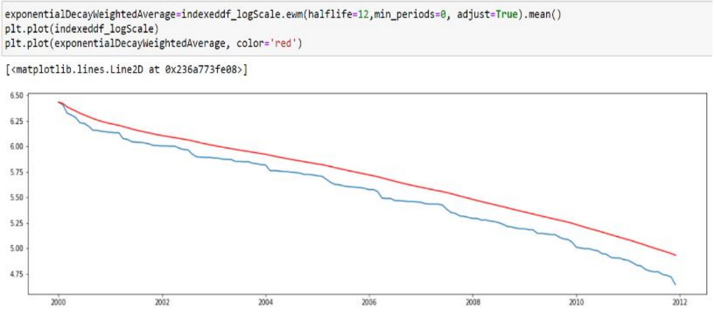
```
[<matplotlib.lines.Line2D at 0x236a5530c08>]
```

**Figure 9:** Transforming to Stationary (Log Scale)

From Figure 9, opting to log scale the data in order to transform the data to stationary. Since the plot is not yet stationary, this research was proceed with exponential decay weighted average.

### 3.7 Phase 7: Perform Exponential Decay Weighted Average

From Figure 9, this research is proceed by input program to compute exponential decay weighted average. From the previous plot, the blue line is not smooth hence not stationary. The result of exponential decay weighted average shown in Figure 10.



```
exponentialDecayWeightedAverage=indexeddf_logScale.ewm(halflife=12,min_periods=0, adjust=True).mean()
plt.plot(indexeddf_logScale)
plt.plot(exponentialDecayWeightedAverage, color='red')
```

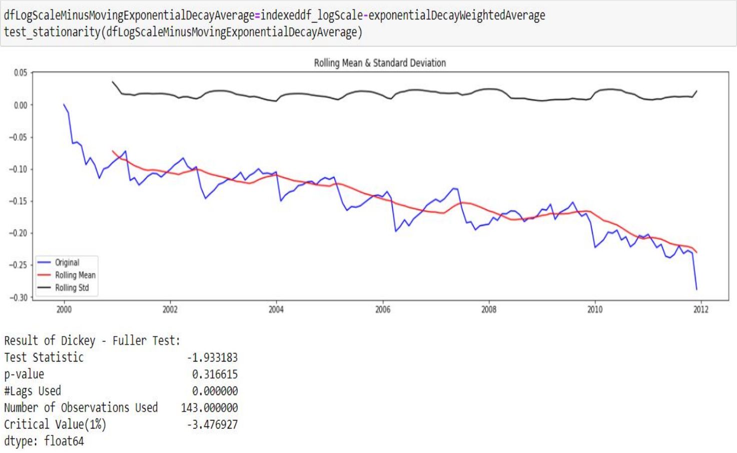
```
[<matplotlib.lines.Line2D at 0x236a773fe08>]
```

**Figure 10:** Transforming to Stationary (Exponential Decay Weighted Average)

Figure 10 shows the plot of exponential decay weighted average in red colour. The plot is smooth compared to log scale plot. However, the plot need to be verified again by evaluate new Dickey – Fuller test.

### 3.8 Phase 8: Evaluate Dickey – Fuller Test Result

This eight phase was carried out to get a better view and proper evaluation on the exponential decay weighted average plot. From Figure 11, the previous rolling mean is similar which was red in colour and black for standard deviation.



```
dfLogScaleMinusMovingExponentialDecayAverage=indexeddf_logScale-exponentialDecayWeightedAverage
test_stationarity(dfLogScaleMinusMovingExponentialDecayAverage)
```

```
Result of Dickey - Fuller Test:
Test Statistic                  -1.933183
p-value                          0.316615
#Lags Used                       0.000000
Number of Observations Used    143.000000
Critical Value(1%)              -3.476927
dtype: float64
```

**Figure 11:** Evaluate New Dickey – Fuller Test

By comparing from Figure 8 the p – value have dropped from 0.33 to 0.31. From this condition proven that the time – series is stationary. Hence, this research proceeds with the plot of Autocorrelation Function (ACF) and Partial Autocorrelation function (PACF).

### 3.9 Phase 9: Plotting ACF and PACF

The purpose of this phase is to obtain the P and Q value to declare the order while building AR, MA and ARIMA models. Figure 12 shows the plot of ACF and PACF.

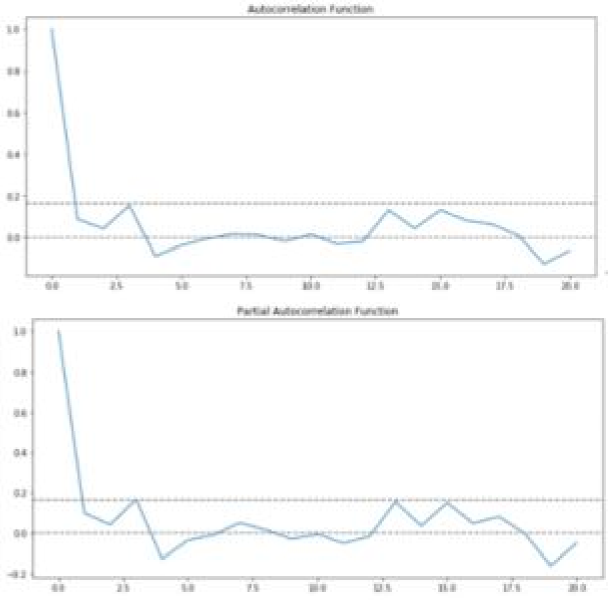
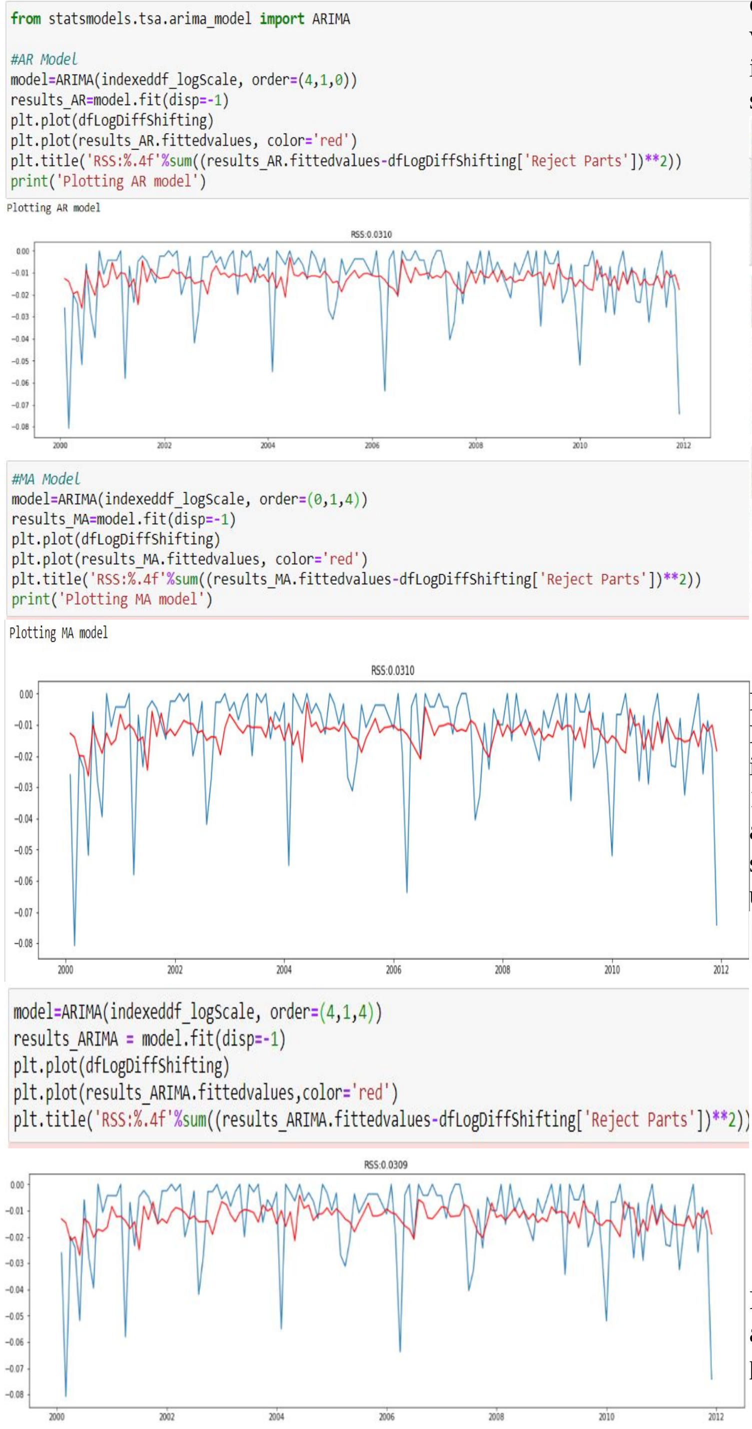


**Figure 12:** Evaluate New Dickey – Fuller Test

Observe from Figure 12, both ACF and PACF gives almost same pattern. At ACF plot, when y equals to zero, x equals to four. This proves the theory Q is equals to four. In PACF, shows when y equals to zero, x equals to four. This also proves the theory that P equals to four. These values were used for next phase of this research.

### 3.10 Phase 10: Build AR, MA and ARIMA Models

The purpose of performing phase 10 is to observe the residual sum of square values. Hence, in order to obtain, proper order need to be declared and few models need to be build. Figure 13 shows the AR, MA and ARIMA models. Observe the programming, the value 4 that obtained in previous phase were used in declaring the order. For AR model, the order is (4,1,0), MA model is (0,1,4) and for ARIMA model combination of both AR and MA models uses the order of (4,1,4).



```
from statsmodels.tsa.arima_model import ARIMA

#AR Model
model=ARIMA(indexeddf_logScale, order=(4,1,0))
results_AR=model.fit(disp=-1)
plt.plot(dfLogDiffShifting)
plt.plot(results_AR.fittedvalues, color='red')
plt.title('RSS:%.4f'%sum((results_AR.fittedvalues-dfLogDiffShifting['Reject Parts'])**2))
print('Plotting AR model')
```

```
Plotting AR model
```

```
#MA Model
model=ARIMA(indexeddf_logScale, order=(0,1,4))
results_MA=model.fit(disp=-1)
plt.plot(dfLogDiffShifting)
plt.plot(results_MA.fittedvalues, color='red')
plt.title('RSS:%.4f'%sum((results_MA.fittedvalues-dfLogDiffShifting['Reject Parts'])**2))
print('Plotting MA model')
```

```
Plotting MA model
```

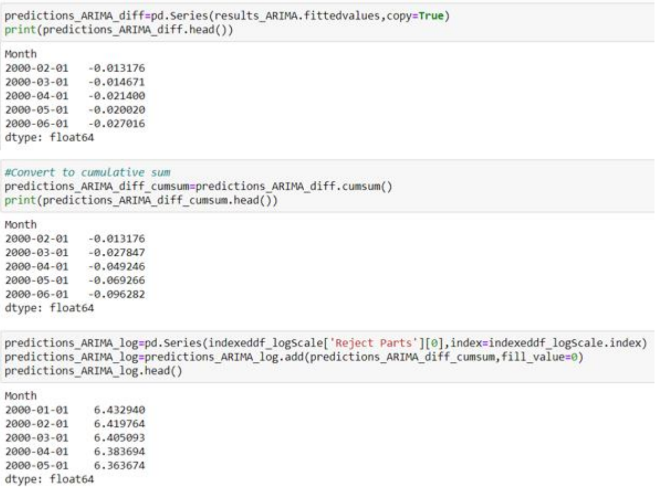
```
model=ARIMA(indexeddf_logScale, order=(4,1,4))
results_ARIMA = model.fit(disp=-1)
plt.plot(dfLogDiffShifting)
plt.plot(results_ARIMA.fittedvalues,color='red')
plt.title('RSS:%.4f'%sum((results_ARIMA.fittedvalues-dfLogDiffShifting['Reject Parts'])**2))
```

**Figure 13:** AR, MA and ARIMA Model

From Figure 13, the Residual Sum of Squares (RSS) is between 0.309 and 0.310. Round off the values, all RSS values show same reading approximately 0.31. Low value gives better AR, MA and ARIMA models. Finally, this research proceeds with last phase, which is performing prediction.

### 3.11 Phase 11: Perform Prediction and Revert Conversion

This is the final phase in result and discussion. From the RSS values from AR, MA and ARIMA model, this research was continued with conversion process. The values were computed for plotting purpose. In order to plot, the values were summed cumulatively. Lastly, the values were converted in log scale in order to remove the negative sign. Figure 14 shows the head printed through python web based.



```
predictions_ARIMA_diff=pd.Series(results_ARIMA.fittedvalues,copy=True)
print(predictions_ARIMA_diff.head())
```

```
Month
2000-02-01   -0.013176
2000-03-01   -0.014671
2000-04-01   -0.021400
2000-05-01   -0.020020
2000-06-01   -0.027016
dtype: float64
```

```
#Convert to cumulative sum
predictions_ARIMA_diff_cumsum=predictions_ARIMA_diff.cumsum()
print(predictions_ARIMA_diff_cumsum.head())
```

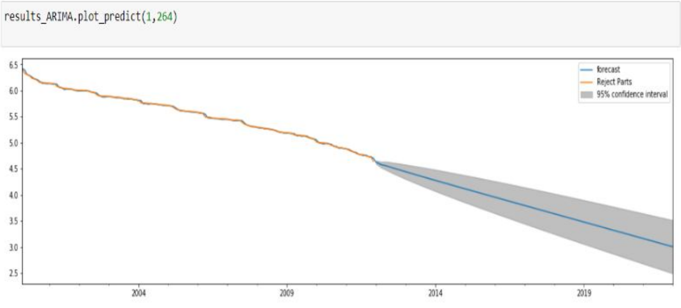
```
Month
2000-02-01   -0.013176
2000-03-01   -0.027847
2000-04-01   -0.049246
2000-05-01   -0.069266
2000-06-01   -0.096282
dtype: float64
```

```
predictions_ARIMA_log=pd.Series(indexeddf_logScale['Reject Parts'][0],index=indexeddf_logScale.index)
predictions_ARIMA_log=predictions_ARIMA_log.add(predictions_ARIMA_diff_cumsum,fill_value=0)
predictions_ARIMA_log.head()
```

```
Month
2000-01-01    6.432940
2000-02-01    6.419764
2000-03-01    6.405093
2000-04-01    6.383694
2000-05-01    6.363674
dtype: float64
```

**Figure 14:** Cumulative Sum of Log Scale Data

Figure 15 shows the data prediction plot in the next 10 years. From the program line, there are 264 steps includes the month in current obtained data. There were 144 months equivalent to 12 years of data. Hence, to extrapolate another 10 years, another 120 steps were add in 144 steps, which sum up to 264 steps. Resulting the plot obtained showing the data prediction until the year 2021.



```
results_ARIMA.plot_predict(1,264)
```

**Figure 15:** Cumulative Sum of Log Scale Data

From Figure 15, shows that the reject numbers decrease across time. The estimate years for the reject to stop being produce may end after the year 2021.

## 4. CONCLUSION

To conclude, ARIMA is a powerful statistical tool that widely used to cater big number of data. There are several algorithm suitable for this application, but ARIMA provides better understanding and clear steps on obtaining data prediction.

The objective of this research is to utilise an algorithm for data prediction of reject produced by manufacturer. Human have emotion and may tend to get fatigue if the same task conducted for long period. Therefore, production of rejects is possible to occur. Proven from Figure 15, there are decreasing in time series trend that the reject may not end, but keep decreasing across time.

## ACKNOWLEDGEMENT

The author wish to express gratitude to those who have given insightful comments and previous advice to improve this paper. The present work is a part of Short Term Research Grant (STR19067). Acknowledge also Universiti Kuala Lumpur for financial support.

## REFERENCES

1. A. Z. Mohamed Noor, M. H. F. Md Fauadi, F. A. Jafar, N. R. Mohamad, and A. S. Mohd Yunos, **A review of techniques to determine alternative selection in design for remanufacturing**, *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 257, no. 1, pp. 1–10, 2017.
2. A. Z. M. Noor, M. H. F. M. Fauadi, F. A. Jafar, and S. F. Zainudin, **Fusion Of Fuzzy AHP in selecting material for drinking water bottle based on customer needs**, *ARPN J. Eng. Appl. Sci.*, vol. 12, no. 14, pp. 4243–4249, 2017.
3. R. Pongto et al., *The Grid-Based Spatial ARIMA Model: An Innovation for Short-Term Predictions of Ocean Current Patterns with Big HF Radar Data*, vol. 936. *Springer International Publishing*, 2020.
4. M. Ji, G. Yi, and J. Jung, **Central Prediction System for Time Series Comparison and Analysis of Water Usage Data**, *IEEE Access*, vol. 8, pp. 10342–10351, 2020.
5. E. Chen, Z. Ye, C. Wang, and M. Xu, **Subway passenger flow prediction for special events using smart card data**, *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1109–1120, 2020.
6. Y. Ye, L. Chen, and F. Xue, **Passenger flow prediction in bus transportation system using ARIMA models with big data**, *Proc. - 2019 Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discov. CyberC* 2019, pp. 436–443, 2019.
7. D. Eldowa, K. Elgazzar, H. S. Hassanein, T. Sharaf, and S. Shah, **Assessing the integrity of traffic data through short term state prediction**, 2019 *IEEE Glob. Commun. Conf. GLOBECOM 2019 - Proc.*, pp. 3–7, 2019.
8. D. Aryo, I. Journal, D. A. Anggoro, P. I. Rahmatullah, and U. M. Surakarta, **The Implementation of Subspace Outlier Detection in K-Nearest Neighbors to Improve Accuracy in Bank Marketing Data**, *Int. J. Emerg. Trends Eng. Res.*, vol. 8, no. February, pp. 545–550, 2020.
9. K. Jose Triny, G. Anjuka, C. Dhanapal, S. Kavibharani, and C. Kowsalya, **A bigdata processing with hadoop map reduce in cloud systems**, *Int. J. Emerg. Trends Eng. Res.*, vol. 8, no. 3, pp. 752–758, 2020.