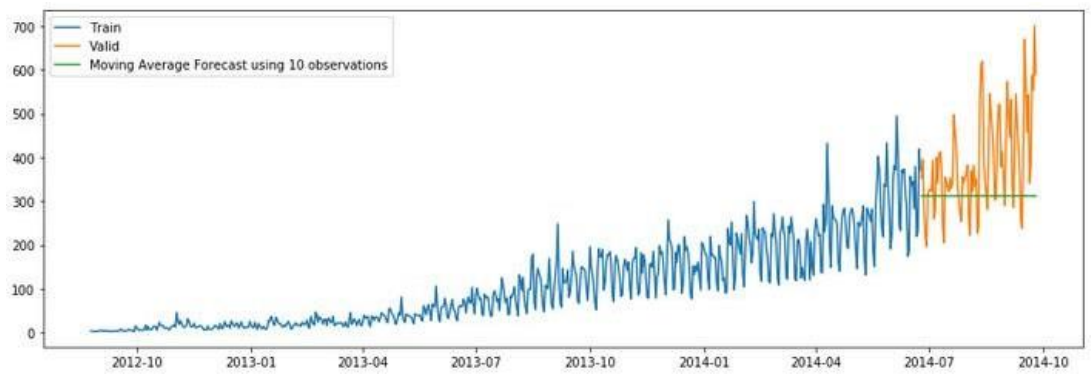
Practice problem on time series forecasting

Problem 1:-

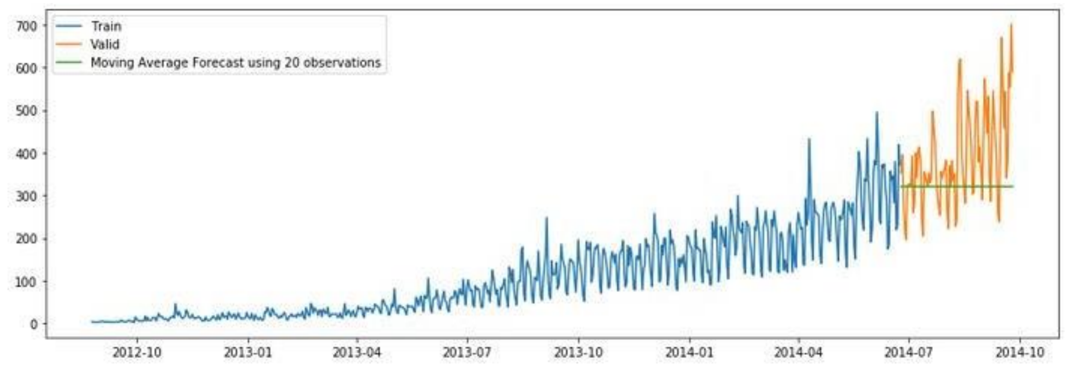
Here the predictions are made on the basis of the average of the last few points instead of taking all the previously known values.

Let's try the rolling mean for the last 10, 20, and 50 days and visualize the results.

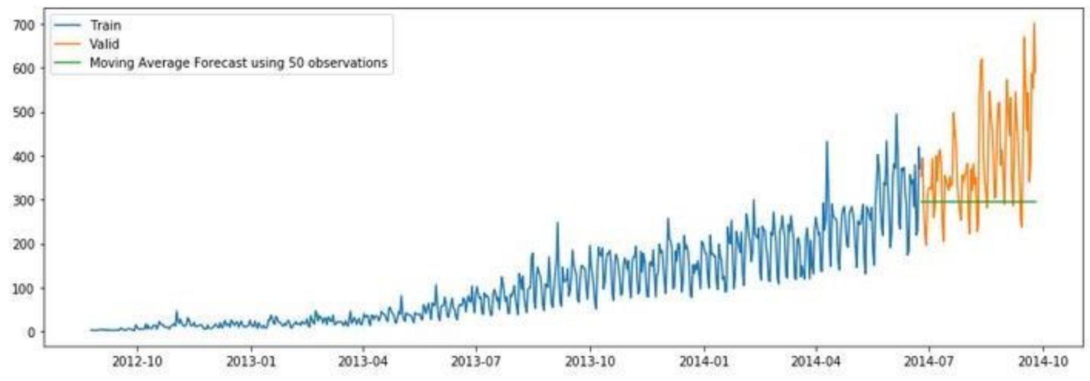
```
y_hat_avg = valid.copy()
y_hat_avg['moving_avg_forecast'] = Train['Count'].rolling(10).mean().iloc[-1]
plt.figure(figsize=(15,5))
plt.plot(Train['Count'], label='Train')
plt.plot(valid['Count'], label='Valid')
plt.plot(y_hat_avg['moving_avg_forecast'],
            label='Moving Average Forecast using 10 observations')
plt.legend(loc='best')
plt.show()
```



```
y_hat_avg = valid.copy()
y_hat_avg['moving_avg_forecast'] = Train['Count'].rolling(20).mean().iloc[-1]
plt.figure(figsize=(15,5))
plt.plot(Train['Count'], label='Train')
plt.plot(valid['Count'], label='Valid')
plt.plot(y_hat_avg['moving_avg_forecast'],
            label='Moving Average Forecast using 20 observations')
plt.legend(loc='best')
plt.show()
```



```
y_hat_avg = valid.copy()
y_hat_avg['moving_avg_forecast'] = Train['Count'].rolling(50).mean().iloc[-1]
plt.figure(figsize=(15,5))
plt.plot(Train['Count'], label='Train')
plt.plot(valid['Count'], label='Valid')
plt.plot(y_hat_avg['moving_avg_forecast'],
                  label='Moving Average Forecast using 50 observations')
plt.legend(loc='best')
plt.show()
```



We took the average of the last 10, 20, and 50 observations and predicted based on that. This value can be changed in the above code in .rolling().mean() part. We can see that the predictions are getting weaker as we increase the number of observations.

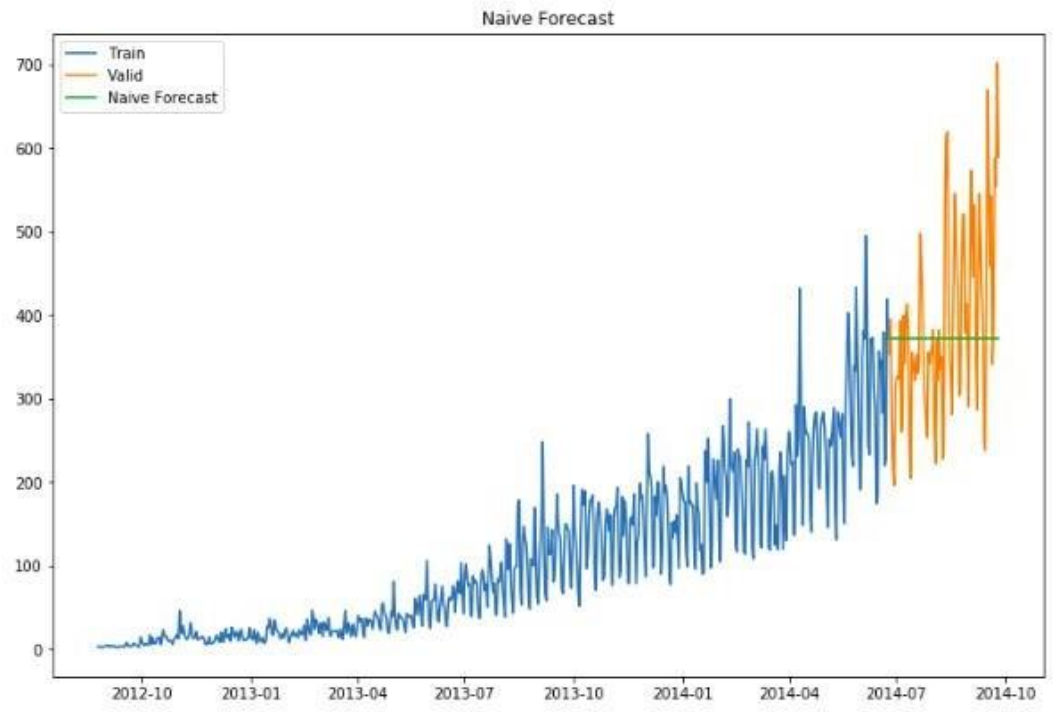
```
rms = sqrt(mean_squared_error(valid.Count, y_hat_avg.moving_avg_forecast))
print(rms)
```

Problem 2:-

The blue line is the prediction here. All the predictions are equal to the last observed point.

Let’s make predictions using the naive approach for the validation set.

```
dd= np.asarray(Train.Count)
y_hat = valid.copy()
y_hat['naive'] = dd[len(dd)-1]
plt.figure(figsize=(12,8))
plt.plot(Train.index, Train['Count'], label='Train')
plt.plot(valid.index,valid['Count'], label='Valid')
plt.plot(y_hat.index,y_hat['naive'], label='Naive Forecast')
plt.legend(loc='best')
plt.title("Naive Forecast")
plt.show()
```



- We can calculate how accurate our predictions are using rmse(Root Mean Square Error).
- rmse is the standard deviation of the residuals.
- Residuals are a measure of how far from the regression line data points are.

- The formula for rmse is:

rmse=sqrt∑i=1N1N(p−a)2

We will now calculate RMSE to check the accuracy of our model on the validation data set.

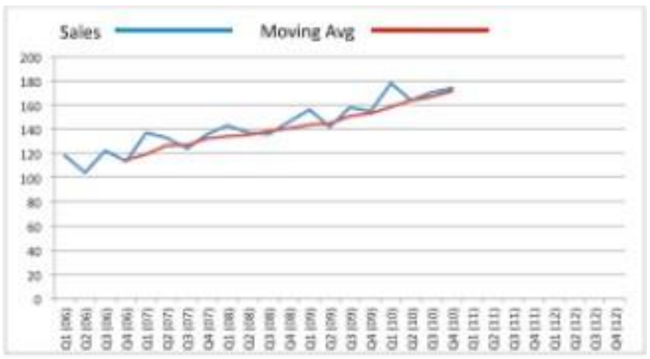
```
from sklearn.metrics import mean_squared_error
from math import sqrt
rms = sqrt(mean_squared_error(valid.Count, y_hat.naive))
print(rms)
```

We can infer that this method is not suitable for datasets with high variability. We can reduce the rmse value by adopting different techniques.

2) Moving Average

- In this technique, we will take the average of the passenger counts for the last few time periods only.

Let’s take an example to understand it:


Sales
Moving Avg
Q1 (06)
Q2 (06)
Q3 (06)
Q4 (06)
Q1 (07)
Q2 (07)
Q3 (07)
Q4 (07)
Q1 (08)
Q2 (08)
Q3 (08)
Q4 (08)
Q1 (09)
Q2 (09)
Q3 (09)
Q4 (09)
Q1 (10)
Q2 (10)
Q3 (10)
Q4 (10)
Q1 (11)
Q2 (11)
Q3 (11)
Q4 (11)
Q1 (12)
Q2 (12)
Q3 (12)
Q4 (12)