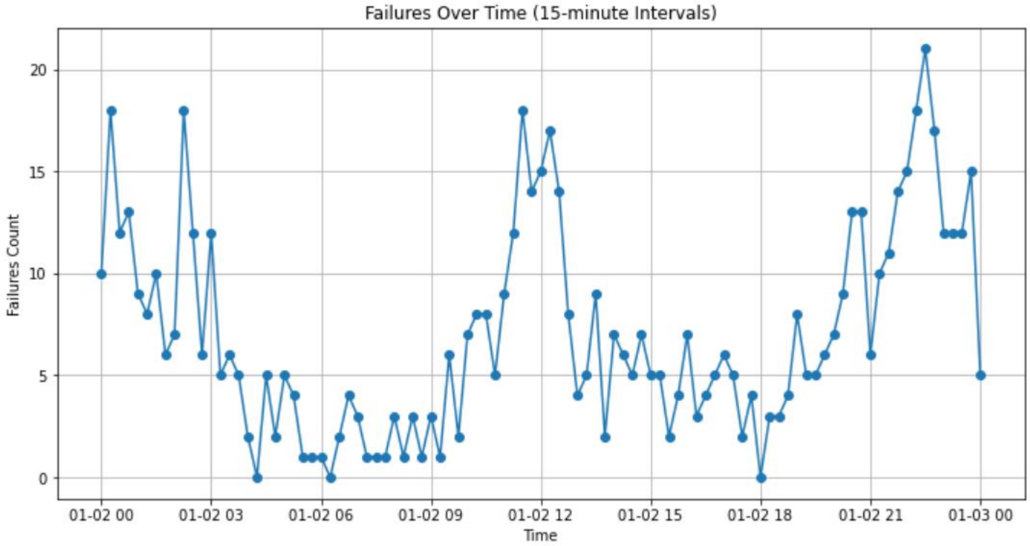
Predicting number of failures

Based on aggregated data (15 min time interval), we get aggregated count of failures across 3 month window for every 15 mins.

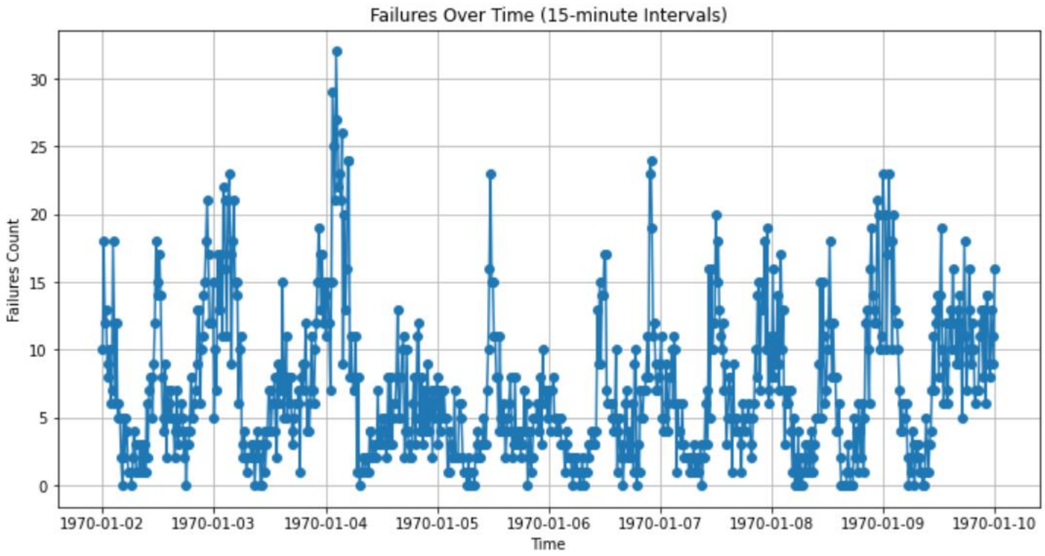
Sample Aggregated Data

timestamp	Failure Count
1970-01-01 20:00:00	2
1970-01-01 20:15:00	6
1970-01-01 20:30:00	9
1970-01-01 20:45:00	7

Based on time series plot, data looks like below (Over 1 day)



Over 10 day Period



Key Initial Observations:

1) **Seasonality** - Data seems to have seasonality as there is spike after every few time intervals.
2) **Mean Stationary data** – Data seems to be mean stationary as means seems to be constant across every cycle.
3) **Variance Non stationary** – Data seems to be nonstationary as variance is not constant. Differencing might be required.

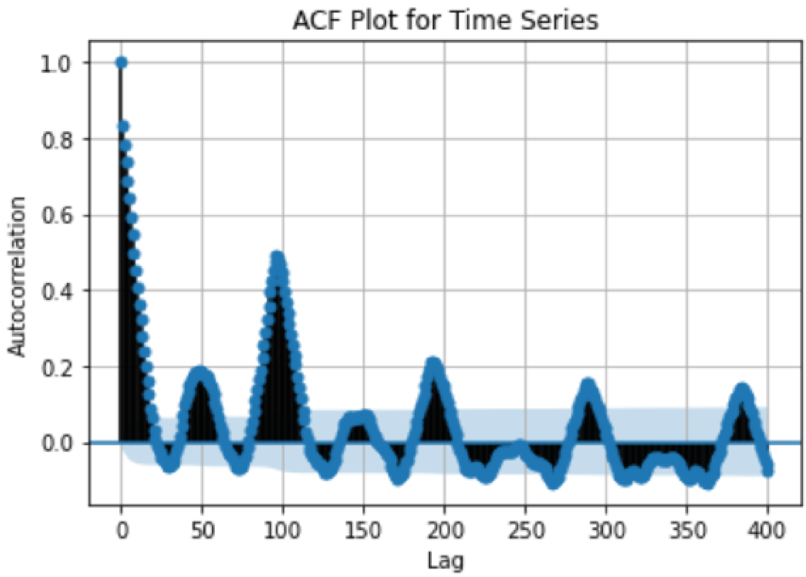
With this hypothesis, we check the following:

1) **Stationarity of Data** – Using KPSS test , where in null hypothesis is data is stationary,
   We get data to be non stationary.
   Thus differencing is required.

Without differencing, ACF plot shows seasonality in data.

For knowing AR, MA component of time series, we use ACF and PACF plots.

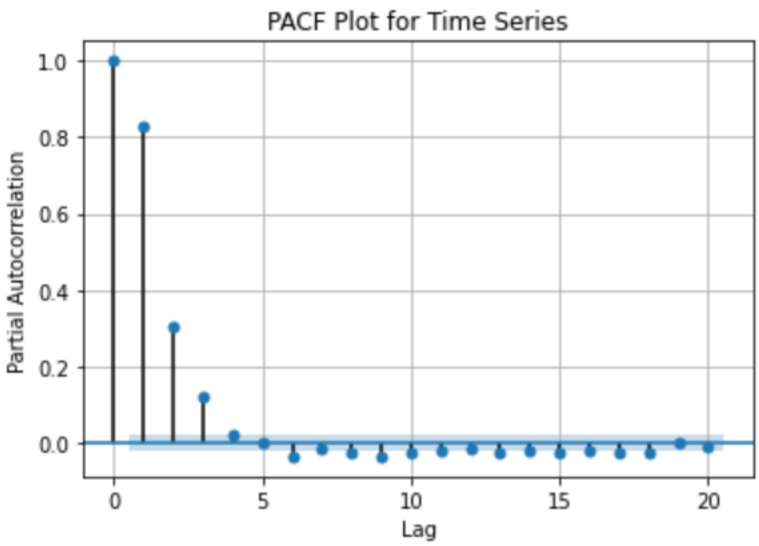
ACF PLOT



- Through the ACF plot, we observe seasonality in the timeseries, data is after every 50 cycle, peak is observed.
- Along with it **moving average order (q)** seems to be 18, post which correlation seems to be within limit.

PACF plot

<Figure size 720x432 with 0 Axes>

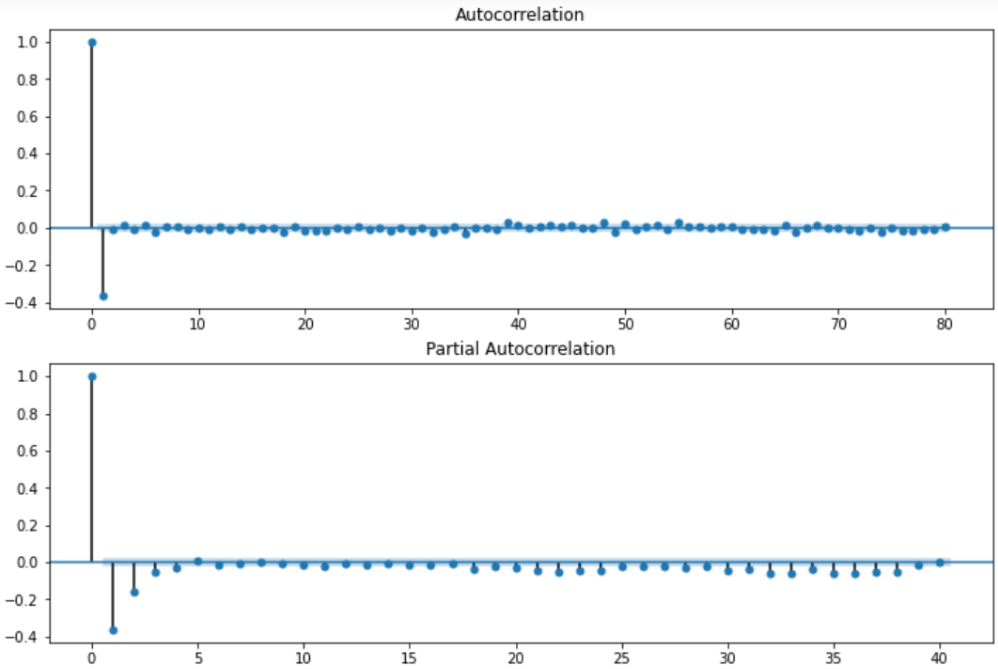


It shows the autoregressive order(P) = 4, rest lag components have minimal correlation with current estimate.

We do first order differencing in the data to make the data stationary.

```
KPSS Test Results:
KPSS Statistic: 0.0030984841843458614
p-value: 0.1
Lags Used: 38
Critical Values:
   10%: 0.347
   5%: 0.463
   2.5%: 0.574
   1%: 0.739
Stationary (Fail to reject the null hypothesis)
```

With first order differencing, we check the ACF and PACF plots to determine significant lags.



Through this we take ,

autoregressive order(P) = 4

moving average order (q) =1

Model building

Basis above observations , we model our prediction using SARIMA model (arima with seasonality) with values p,d,q = (4,1,1) and seasonality parameter to be 50.

Due to below memory constraints our model was not working.

```
In [57]:  1 mod = sm.tsa.statespace.SARIMAX(subset_df.time, trend='n', order=(0,1,0), seasonal_order=(0,1,1,50))
           2 results = mod.fit()
           3 print(results.summary())
           4

---------------------------------------------------------------------------
MemoryError                               Traceback (most recent call last)
```

```
---> 61         return bound(*args, **kwds)
     62     except TypeError:
     63         # A TypeError occurs if the object does have such a method in its

MemoryError: Unable to allocate 61.0 MiB for an array with shape (769, 102, 102) and data type float64
```

Alternative Path : ARIMA model with (p,d,q = 4,1,1) parameters we built the model.

```
                             ARIMA Model Results
==============================================================================
Dep. Variable:                 D.time   No. Observations:                 9787
Model:                 ARIMA(4, 1, 1)   Log Likelihood              -28354.520
Method:                       css-mle   S.D. of innovations              4.384
Date:                Thu, 26 Oct 2023   AIC                          56723.040
Time:                        05:35:11   BIC                          56773.361
Sample:                    01-01-1970   HQIC                         56740.091
                         - 04-13-1970
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
const            0.0006      0.000      3.839      0.000       0.000       0.001
ar.L1.D.time     0.5393      0.010     53.359      0.000       0.519       0.559
ar.L2.D.time     0.2284      0.011     19.975      0.000       0.206       0.251
ar.L3.D.time     0.1055      0.011      9.223      0.000       0.083       0.128
ar.L4.D.time     0.0207      0.010      2.046      0.041       0.001       0.040
ma.L1.D.time    -1.0000      0.000  -3140.736      0.000      -1.001      -0.999
                                    Roots
```

We get the above coefficients for lagged components.

Prediction:

For prediction across next 4, 15 minute time intervals , using same ARIMA model we get the following estimates of failures

```
array([6.45382743, 6.93548553, 7.29684217, 7.62894752])
```