- First dataset is created with original datapoints, where the label data is moved 10 steps ahead. Our idea is to find failure 10 minutes in advance.

```
indexData_org=pd.DataFrame(columns=['Index', 'sensor_01', 'sensor_02', 'sensor_03', 'sensor_04', 'sensor_06',
                                    'sensor_10', 'sensor_11', 'sensor_12', 'sensor_38', 'sensor_40', 'machine_status'])
```

```
indexData_org['Index']=range(indexData.shape[0])
```

```
  indexData_org['sensor_01'] = indexData['sensor_01']
  indexData_org['sensor_02'] = indexData['sensor_02']
  indexData_org['sensor_03'] = indexData['sensor_03']
  indexData_org['sensor_04'] = indexData['sensor_04']
  indexData_org['sensor_06'] = indexData['sensor_06']
  indexData_org['sensor_10'] = indexData['sensor_10']
  indexData_org['sensor_11'] = indexData['sensor_11']
  indexData_org['sensor_12'] = indexData['sensor_12']
  indexData_org['sensor_38'] = indexData['sensor_38']
  indexData_org['sensor_40'] = indexData['sensor_40']
```

```
for i in tqdm(range(indexData.shape[0]-10)):
  indexData_org['machine_status'][i] = indexData['machine_status'][i+10]
```

```
  0%|          | 0/220310 [00:00<?, ?it/s]/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

100%|██████████| 220310/220310 [03:13<00:00, 1141.09it/s]
```

```
indexData_org.dropna(inplace=True)
```

```
indexData_org=indexData_org.set_index('Index')
```

- Second dataset is created with average datapoints of last 10 values.
- Average datapoints includes mean, median and standard deviation.
- Here also label data moved 10 steps ahead to find prediction 10 minutes in advance.

```
# Create dataframe to store mean and median values of sensor data

indexData_avg=pd.DataFrame(columns=['Index', 'mean_sensor_01', 'mean_sensor_02', 'mean_sensor_03', 'mean_sensor_04', 'mean_sensor_06',
                                   'mean_sensor_10', 'mean_sensor_11', 'mean_sensor_12', 'mean_sensor_38', 'mean_sensor_40', 'median_sensor_01',
                                    'median_sensor_02', 'median_sensor_03', 'median_sensor_04', 'median_sensor_06', 'median_sensor_10',
                                    'median_sensor_11', 'median_sensor_12', 'median_sensor_38', 'median_sensor_40', 'std_sensor_01',
                                    'std_sensor_02', 'std_sensor_03', 'std_sensor_04', 'std_sensor_06', 'std_sensor_10',
                                    'std_sensor_11', 'std_sensor_12', 'std_sensor_38', 'std_sensor_40','machine_status'])
```

```
indexData_avg['Index']=range(10,indexData.shape[0])
```

```
for i in tqdm(range(10,indexData.shape[0])):
  indexData_avg['mean_sensor_01'][i] = indexData['sensor_01'][i-10:i].mean()
  indexData_avg['mean_sensor_02'][i] = indexData['sensor_02'][i-10:i].mean()
  indexData_avg['mean_sensor_03'][i] = indexData['sensor_03'][i-10:i].mean()
  indexData_avg['mean_sensor_04'][i] = indexData['sensor_04'][i-10:i].mean()
  indexData_avg['mean_sensor_06'][i] = indexData['sensor_06'][i-10:i].mean()
  indexData_avg['mean_sensor_10'][i] = indexData['sensor_10'][i-10:i].mean()
  indexData_avg['mean_sensor_11'][i] = indexData['sensor_11'][i-10:i].mean()
  indexData_avg['mean_sensor_12'][i] = indexData['sensor_12'][i-10:i].mean()
  indexData_avg['mean_sensor_38'][i] = indexData['sensor_38'][i-10:i].mean()
  indexData_avg['mean_sensor_40'][i] = indexData['sensor_40'][i-10:i].mean()
```

- Similarly create datapoints with median, std also
- Do as we did it for original datapoints.