```
import pandas as pd
import numpy as np
import tensorflow as tf
from pandas import datetime

from statsmodels.tsa.stattools import adfuller

import matplotlib.pyplot as plt
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 18, 5


def parser(x):
  return datetime.strptime(x, '%Y-%m-%d %H:%M:%S')


bgd = pd.read_csv('/content/drive/My Drive/blood-glucose-data.csv', index_col=1, pa
bgd.head()
```

point_timestamp	point_value(mg/dL)	timezone_offset
2017-05-15 07:51:22	142	-700
2017-05-15 07:56:23	140	-700
2017-05-15 08:01:22	138	-700
2017-05-15 08:06:22	136	-700
2017-05-15 08:11:23	130	-700

```
# del bgd['timezone_offset']
plt.xlabel('Date')
plt.ylabel('point value mg/dl')
plt.plot(bgd)
```

```
[<matplotlib.lines.Line2D at 0x7f8a95a36668>]
```

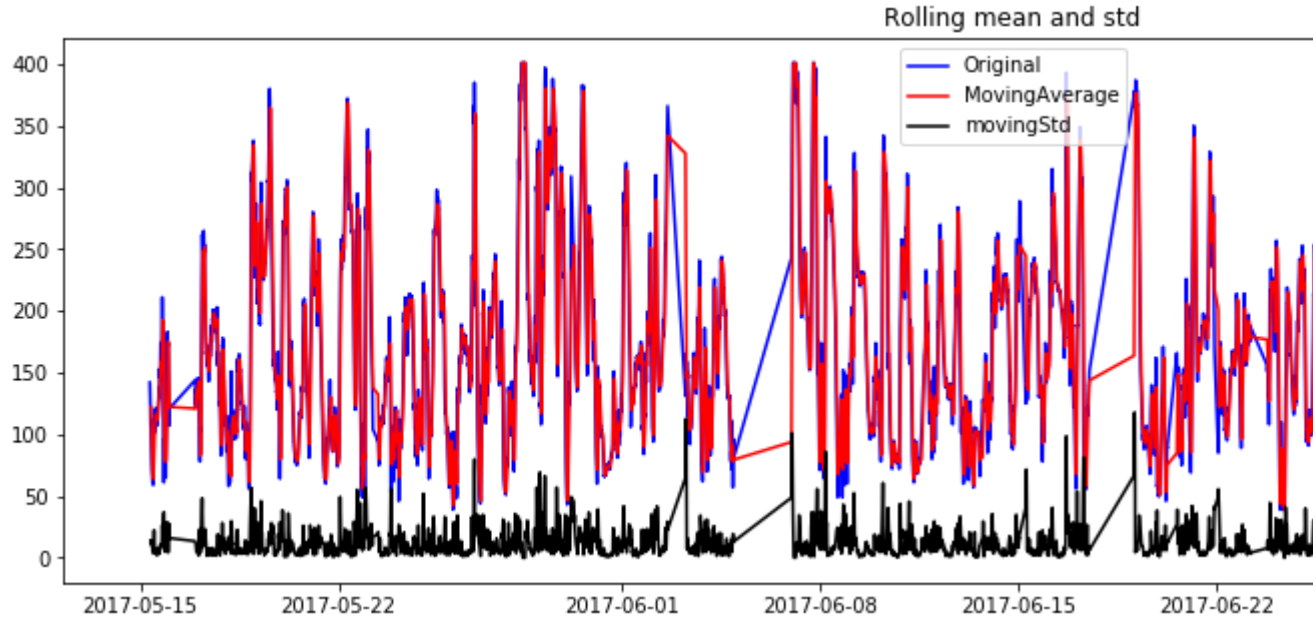
```
# detrrmine the rolling mean
rolmean = bgd.rolling(window=12).mean()
rolstd = bgd.rolling(window=12).std()
print(rolmean, rolstd)
```

```
                     point_value(mg/dL)
point_timestamp
2017-05-15 07:51:22                 NaN
2017-05-15 07:56:23                 NaN
2017-05-15 08:01:22                 NaN
2017-05-15 08:06:22                 NaN
2017-05-15 08:11:23                 NaN
...                                 ...
2017-07-15 07:29:50          216.166667
2017-07-15 07:34:51          217.250000
2017-07-15 07:39:50          218.000000
2017-07-15 07:44:50          218.500000
2017-07-15 07:49:50          219.000000

[14702 rows x 1 columns]                      point_value(mg/dL)
point_timestamp
2017-05-15 07:51:22                 NaN
2017-05-15 07:56:23                 NaN
2017-05-15 08:01:22                 NaN
2017-05-15 08:06:22                 NaN
2017-05-15 08:11:23                 NaN
...                                 ...
2017-07-15 07:29:50            3.761850
2017-07-15 07:34:51            2.562846
2017-07-15 07:39:50            1.758098
2017-07-15 07:44:50            2.236068
2017-07-15 07:49:50            2.923261

[14702 rows x 1 columns]
```

```
# plot rolling
orig = plt.plot(bgd, color='blue', label='Original')
mean = plt.plot(rolmean, color='red', label='MovingAverage')
std = plt.plot(rolstd, color='black', label='movingStd')
plt.legend(loc='best')
plt.title('Rolling mean and std')
plt.show(block=False)
```



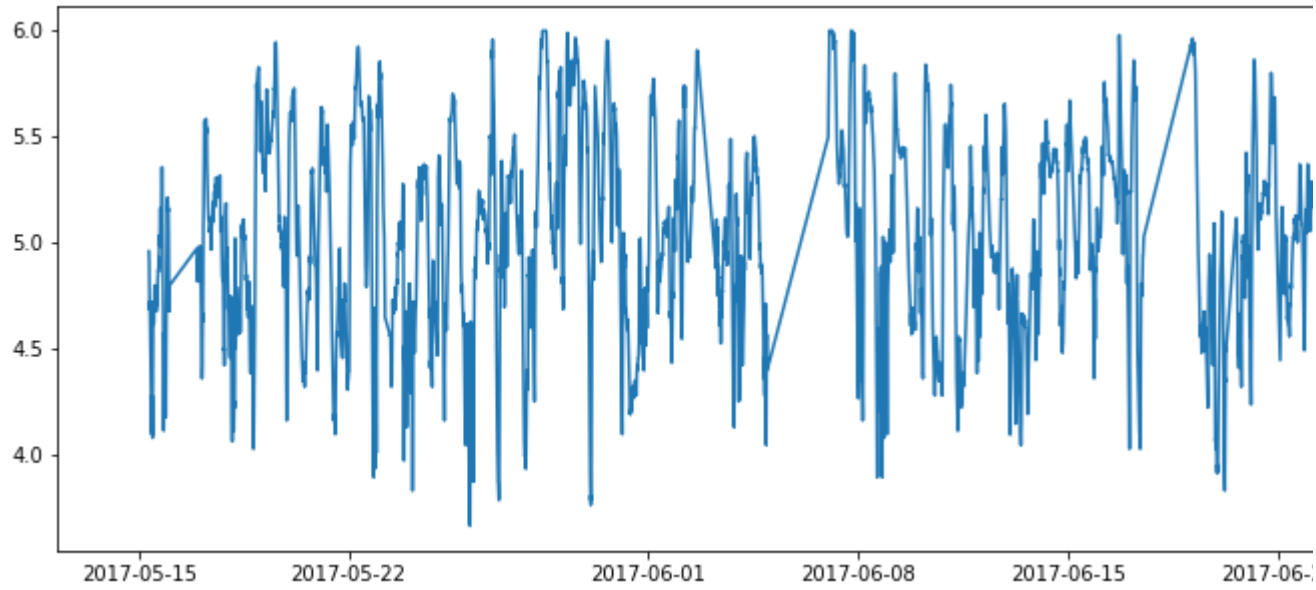

```
# Dickey Fuller results
bgd_test = adfuller(data['point_value(mg/dL)'], autolag='AIC')
bgd_output = pd.Series(bgd_test[0:4], index=['Test Statistic', 'p-value', '#Lags-Us
                                             'Number of observations used'])
for key, value in bgd_test[4].items():
  bgd_output['Critical Value (%s)'%key] = value

print(bgd_output)
```

```
Test Statistic                -1.338734e+01
p-value                        4.843618e-25
#Lags-Used                     1.200000e+01
Number of observations used    1.468900e+04
Critical Value (1%)           -3.430795e+00
Critical Value (5%)           -2.861737e+00
Critical Value (10%)          -2.566875e+00
dtype: float64
```

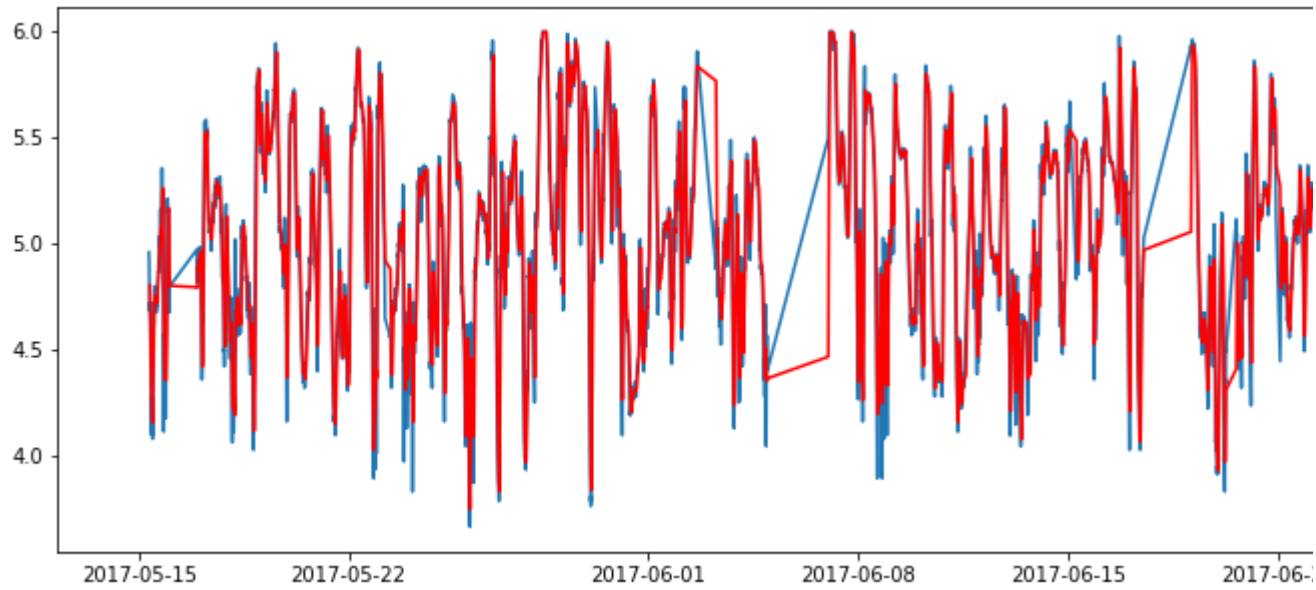
```
bgd_logscale = np.log(bgd)
plt.plot(bgd_logscale)
```

```
[<matplotlib.lines.Line2D at 0x7f8a90f13828>]
```



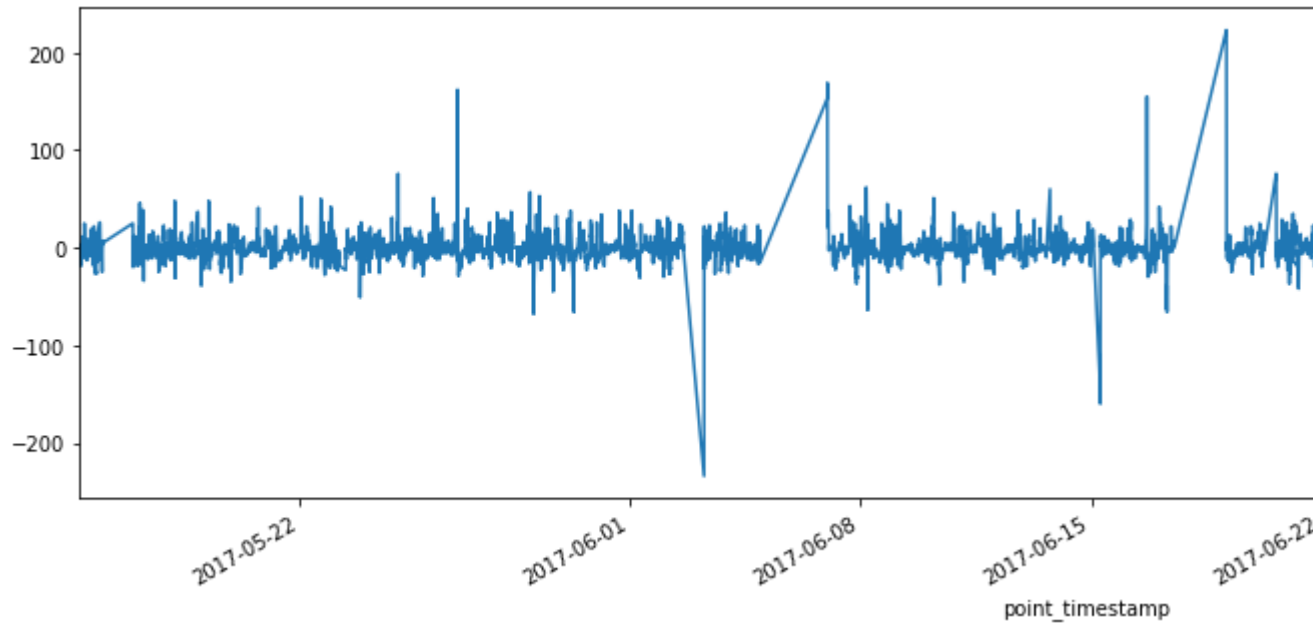

```
rolmean = bgd_logscale.rolling(window=12).mean()
rolstd = bgd_logscale.rolling(window=12).std()
plt.plot(bgd_logscale)
plt.plot(rolmean, color='red')
```

```
[<matplotlib.lines.Line2D at 0x7f8a90e7c278>]
```




```
bgd_diff = bgd.diff(periods=2)
bgd_diff.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8a8dc379e8>
```



check for stationary