In [ ]:

First we import all the necessary libraries

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
```

After that, we import the data from the csv files and store it in the dataframe using panda library.

In [ ]:
```
data=pd.read_csv('Untitled1.csv')
df = pd.DataFrame(data)
df['Day']=df['Timestamp'].str.slice(0,2).astype(int)
df['Month']=df['Timestamp'].str.slice(3,5).astype(int)
df['Year']=df['Timestamp'].str.slice(6,11).astype(int)
df['Time']=(df['Timestamp'].str.slice(11,13)+df['Timestamp'].str.slice(14
pd.set_option('display.max_rows',None)
```

Then, we divide our data into three categories on the basis of the current values - Good, Bad and Missing

In [ ]:
```
good=[]
bad=[]
missing=[]
for j in range(283):
    data0=[]
    data1=[]
    data2=[]
    for i in range(96,192):
        data0.append([df['Timestamp'][j*288+i-96],df['HT R Phase Current'
        data1.append([df['Timestamp'][j*288+i],df['HT R Phase Current'][j
        data2.append([df['Timestamp'][j*288+i+96],df['HT R Phase Current'
    mean = np.mean([x[1] for x in data1])
    std = np.std([x[1] for x in data1])
    counter = 0
    missed_values = 0
    for k in range (2,len(data1)-2):
        std_t = np.std([data1[k-2][1],data1[k-1][1],data1[k][1],data1[k+1
        if(std_t>10):
            counter = counter + 1
    for l in range (0,len(data1)-1):
        if (data1[l][1]==0):
            missed_values = missed_values+1

    if(missed_values>30) :
        for i in range(96):
            missing.append(data0[i])
        for i in range(96):
            missing.append(data1[i])
        for i in range(96):
            missing.append(data2[i])
    elif(counter < 10):
        for i in range(96):
            good.append(data0[i])
        for i in range(96):
            good.append(data1[i])
        for i in range(96):
            good.append(data2[i])
    else:
        for i in range(96):
            bad.append(data0[i])
        for i in range(96):
            bad.append(data1[i])
        for i in range(96):
            bad.append(data2[i])
```

We make the good data better by denoising the good signal and replacing the noisy signal by the local mean of the signals

In [ ]:
```
for i in range(287,len(good),288):
        for j in range(i-190,i-94):
         good[k][1]= [good[k-2][1],good[k-1][1],good[k][1],good[k+1][1],g
```

We then create a new dataframe with only improved good data

In [ ]:
```
gooddf=pd.DataFrame(good)
column_names = ["Timestamp",'HT R Phase Current', 'Day', 'Month', 'Year',
gooddf.columns = column_names
print(gooddf)
```

Here, we have trained our model using Multiple Linear Regression with parameters as Time index and its multiple powers.

In [ ]:
```
gooddf['Time'] = gooddf['Time'].astype(int)
index=[]
for i in range(287,len(gooddf['Time']),288):
        for j in range(0,288):
         index.append(j)

gooddf['Time'] = index
gooddf_train=gooddf.head(23040)
gooddf_test=gooddf.tail(6335)
gooddf_train["T2"]=gooddf_train["Time"]**2
gooddf_train["T4"]=gooddf_train["Time"]**4
gooddf_train["T3"]=gooddf_train["Time"]**3
gooddf_train["T5"]=gooddf_train["Time"]**5
gooddf_train["T6"]=gooddf_train["Time"]**6
gooddf_train["T7"]=gooddf_train["Time"]**7
X_train = gooddf_train[['Time',"T2","T4","T3","T5","T6"]]
y_train = gooddf_train['HT R Phase Current']
X_test = gooddf_test[['Day', 'Month', 'Time']]
y_test = gooddf_test['HT R Phase Current']
```

In [ ]:
```
t=168
X_train = sm.add_constant(X_train)
model = sm.OLS(y_train, X_train).fit()
print(model.summary())


prediction=model.predict(X_train).tolist()
gooddf_train['prediction']=prediction

for i in range(287,len(gooddf['Time']),288):
        for j in range(i-287,i-211):
         gooddf_train['prediction'][j]=0.1
        for j in range(i-49,i+1):
         gooddf_train['prediction'][j]=0.1

gooddf_train['accuracy']=(abs(gooddf_train['prediction']-gooddf_train['HT

gooddf_train['accuracy']=gooddf_train['accuracy']/gooddf_train['HT R Phas
accuracy=[]
for j in range(96,192):
 accuracy.append(gooddf_train['accuracy'][j])

print(np.mean(accuracy))

example=gooddf_train[gooddf['Month']==2]
plt.plot(example['Timestamp'],example['HT R Phase Current'],label='Actual
plt.plot(example['Timestamp'],example['prediction'],label='Pred')
plt.legend()
```