Kyle Nicely
EE291
4/28/2022

Project 2

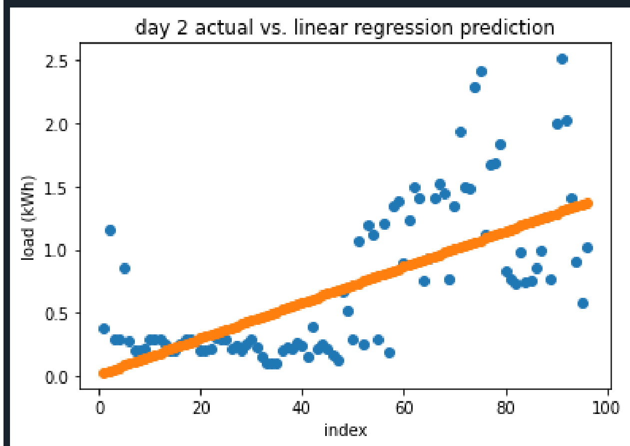
Part 1: Linear Regression Model

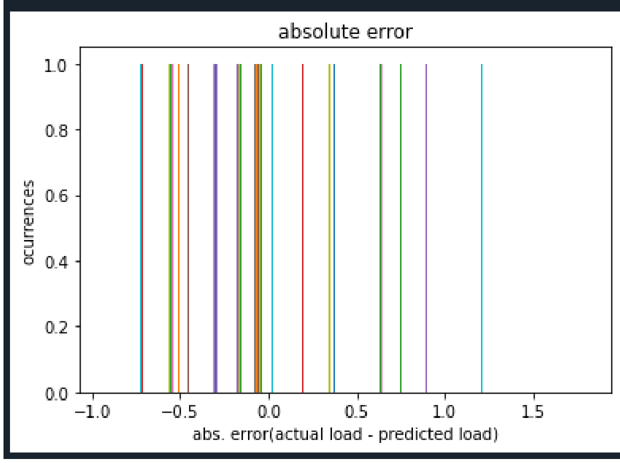
Based on the given excel spreadsheet electricity consumption data for one day, create a linear regression model that predicts the following day's electricity consumption.

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Apr 13 14:47:28 2022

@author: knicely
"""

import pandas as pd
from sklearn.linear_model import LinearRegression as LR
import matplotlib.pyplot as plt

df = pd.read_excel('Electricty Consumption.xlsx', sheet_name = 'Day_1')
df2 = pd.read_excel('Electricty Consumption.xlsx', sheet_name = 'Day_2')

x = df['Index']
x2 = df2['Index']
y = df['KWh']
y2 = df2['KWh']

x_train = x.values.reshape(-1,1)
x_pred = x2.values.reshape(-1,1)
y_train = y.values.reshape(-1,1)
y2_train = y.values.reshape(-1,1)


reg = LR()
reg.fit(x_train,y_train)
print('linearConst = ',reg.intercept_)
print('linearCoeff = ', reg.coef_)

y_pred = reg.predict(x_pred)

plt.scatter(x2, y2)
plt.title("day 2 actual")
plt.xlabel("index")
plt.ylabel("load kWh")

plt.scatter(x2, y_pred)
plt.title("day 2 actual vs. linear regression prediction")
plt.xlabel("index")
plt.ylabel("load (kWh)")

err = y2_train - y_pred
errList = err.tolist()


fig, ax = plt.subplots()
ax.hist(errList)
ax.set_title('absolute error')
ax.set(xlabel='abs. error(actual load - predicted load)', ylabel='ocurrences')
```

Python code for linear regression

```
IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/Kyle/Documents/EE291Project2/Project2.py', wdir='C:/Users/Kyle/Documents/EE291Project2')
linearConst =  [0.01472807]
linearCoeff =  [[0.01415939]]
```





Linear Regression

The scatter plot (blue dots) shows the actual data for electricity consumption for day 2. The linear regression (orange line) is the predicted consumption of day 2 based on the data from day 1.

The histogram x-axis is the absolute error between the actual load and predicted load. The y-axis is the number of occurrences (each thin colored bar is one data point).

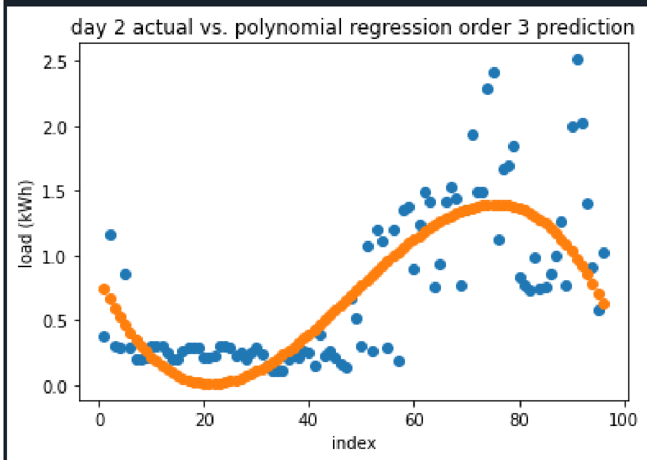
Part 2: Polynomial Regression Model

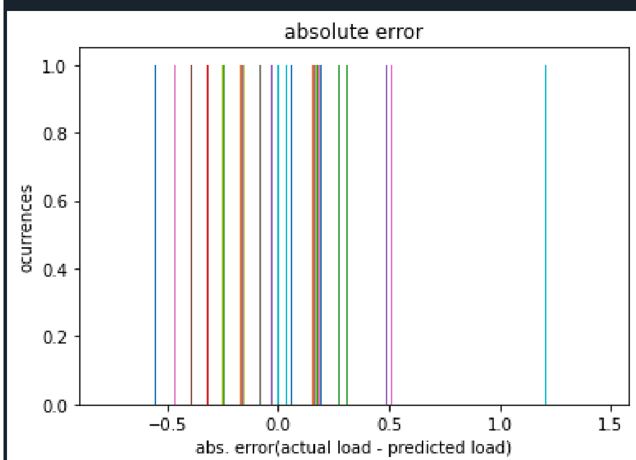
Based on the given excel spreadsheet electricity consumption data for one day, create a $3^{rd},\ 5^{th},\ and\ 7^{th}$ order polynomial regression model that predicts the following day's electricity consumption. Compare the three models based on their total squared error of prediction.

```python
Created on Thu Apr 14 14:01:08 2022

@author: Kyle
"""

import pandas as pd
from sklearn.linear_model import LinearRegression as LR
from sklearn.preprocessing import PolynomialFeatures as PF
import matplotlib.pyplot as plt
import numpy as np

df = pd.read_excel('Electricty Consumption.xlsx', sheet_name = 'Day_1')
df2 = pd.read_excel('Electricty Consumption.xlsx', sheet_name = 'Day_2')

x = df['Index']
x2 = df2['Index']
y = df['KWh']
y2 = df2['KWh']

x_train = x.values.reshape(-1,1)
x_pred = x2.values.reshape(-1,1)
y_train = y.values.reshape(-1,1)
y2_train = y.values.reshape(-1,1)

Poly = PF(3)
x_poly = Poly.fit_transform(x_train)
x_poly_2 = Poly.fit_transform(x_pred)
reg2 = LR()
reg2.fit(x_poly,y_train)

print('poly3Const = ', reg2.intercept_)
print('poly3Coef = ', reg2.coef_)
y_pred2 = reg2.predict(x_poly_2)

err = y2_train - y_pred2
errList = err.tolist()
squared_error = err**2

total_squared_error = np.sum(squared_error)
print('Total squared error of prediction = ', total_squared_error)

plt.scatter(x2, y2)
plt.title("day 2 actual")
plt.xlabel("index")
plt.ylabel("load kWh")

plt.scatter(x2,y_pred2)
plt.title("day 2 actual vs. polynomial regression order 3 prediction")
plt.xlabel("index")
plt.ylabel("load (kWh)")


fig, ax = plt.subplots()
ax.hist(errList)
ax.set_title('absolute error')
ax.set(xlabel='abs. error(actual load - predicted load)', ylabel='ocurrences')
```

Python code for 3^{rd} order polynomial regression

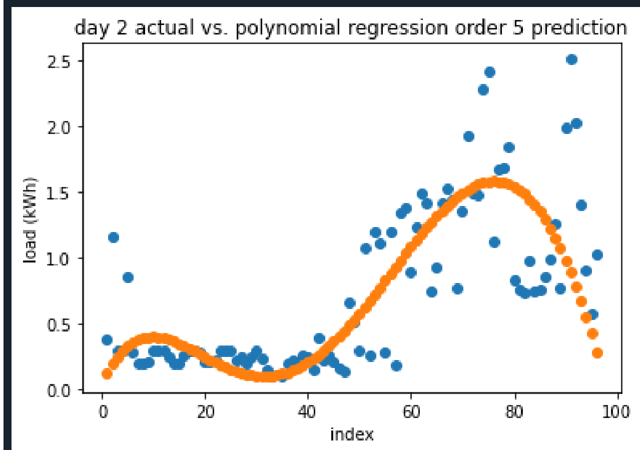
```
In [11]: runfile('C:/Users/Kyle/Documents/EE291Project2/Project2.1.py', wdir='C:/Users/Kyle/Documents/
EE291Project2')
poly3Const =  [0.82528354]
poly3Coef =  [[ 0.00000000e+00 -8.45768021e-02  2.54717750e-03 -1.75783320e-05]]
Total squared error of prediction =  16.918429415728856
```

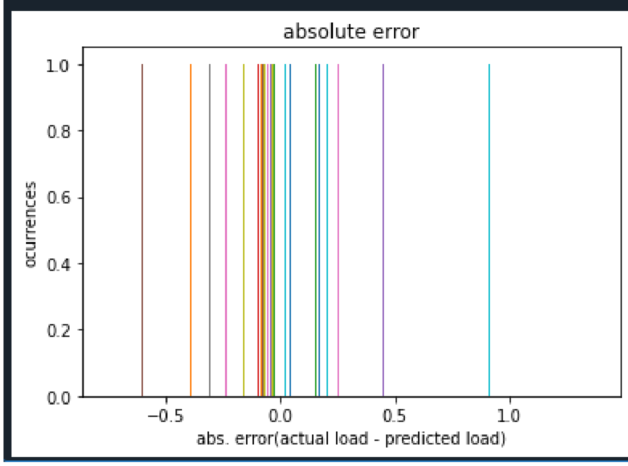




3^{rd} order Polynomial Regression

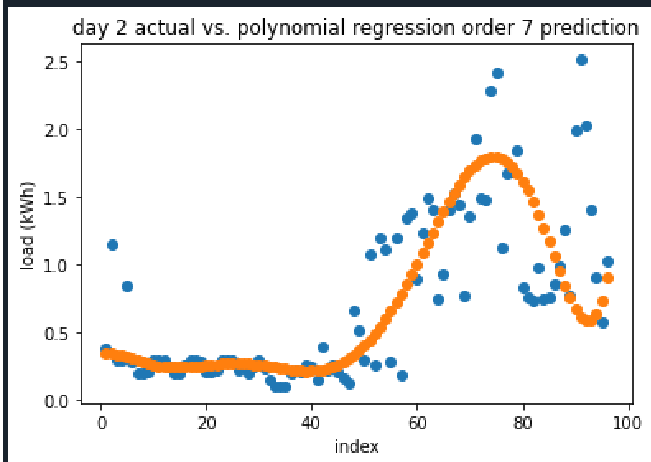
```
In [10]: runfile('C:/Users/Kyle/Documents/EE291Project2/Project2.2.py', wdir='C:/Users/Kyle/Documents/
EE291Project2')
poly5Const =  [0.0476388]
poly5Coef =  [[ 0.00000000e+00  8.58716239e-02 -6.61748684e-03  1.72336232e-04
  -1.65191703e-06  5.00592631e-09]]
Total squared error of prediction =  13.688242086620118
```

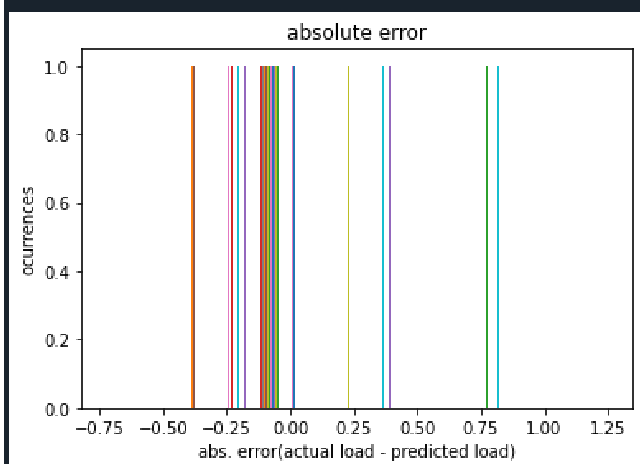




5^{th} order Polynomial Regression

```
In [9]: runfile('C:/Users/Kyle/Documents/EE291Project2/Project2.3.py', wdir='C:/Users/Kyle/Documents/
EE291Project2')
poly7Const =  [0.3364964]
poly7Coef =  [[ 0.00000000e+00  1.32289209e-02 -5.37708877e-03  4.93978062e-04
  -1.98030840e-05  3.86424746e-07 -3.55042841e-09  1.22710511e-11]]
Total squared error of prediction =  11.24466580886131
```





7^{th} order Polynomial Regression

The scatter plot (blue dots) shows the actual data for electricity consumption for day 2. The polynomial regression (orange line) is the predicted consumption of day 2 based on the data from day 1.

The histogram x-axis is the absolute error between the actual load and predicted load. The y-axis is the number of occurrences (each thin colored bar is one data point).

Total squared error of prediction:

The total squared error of prediction is represented by $E = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} (y_i - \hat{y})^2$ where y is the actual load and ŷ is the predicted load. For the 3^{rd} order polynomial the total squared error of prediction is 16.918. For the 5^{th} order it is 13.688, and for the 7^{th} order, 11.245. As the order of the polynomial regression increases the total squared error of prediction decreases. We see this in the histograms of absolute error. As the order increases the distribution of error for each data point becomes closer to zero.