| كلية الحوسبة والمعلوماتية
College of Computing and Informatics

الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

Semester 1 – 2021/2022

Course Code	DS550
Course Name	Machine Learning
Assignment type	Critical Thinking
Module	05

Student ID	G200007615
Student Name	Abdulaziz Alqumayzi
CRN	15062

Solutions:

Critical Thinking Assignment 2

Introduction

In this activity, we will define and explain Polynomial Regression analysis algorithm. Provide a python programming example on how to apply the algorithm on a dataset. The dataset used is **Synchronous Machine Data Set Data Set** from UCI Machine Learning Repository.

Data Set Information: Synchronous motors are constant speed AC motors. A real experimental set is provided for a Synchronous Motor Data Set. The experimental operation environment produced synchronous machine data in real-time.

Attribute information in Table 1.

Table 1

Attribute Information

Column	Data type
Iy (Load Current)	Integer
PF (Power factor)	Integer
e (Power factor error)	Integer
dIf (Changing of excitation current of synchronous machine)	Integer
If (Excitation current of synchronous machine)	Integer

Polynomial Regression analysis

The Polynomial Regression is a regression technique which shapes a relationship of dependent(y) to independent (x) variable as a polynomial of the nth degree. It is a linear model that has been altered to enhance accuracy. The data set utilized for training polynomial regression is non-linear. If a linear model is used to a linear dataset, it gives us with a decent outcome as demonstrated in Simple Linear Regression, but if we apply the same model with no change in a non-linear dataset, a dramatic consequence would be produced. As the loss function increases, the error rate is large, and accuracy decreases. Therefore, we need the Polynomial Regression model for such instances in which data points are not organized in linear way. This can be better understood with the following comparison graphic of the linear and nonlinear data sets.

Equation of the Polynomial Regression Model:

Simple Linear Regression equation: $y = b_0 + b_1 x$

Multiple Linear Regression equation: $y = b_0 + b_1 x + \ b_2 x_2 + b_3 x_3 + \ldots + b_n x_n$

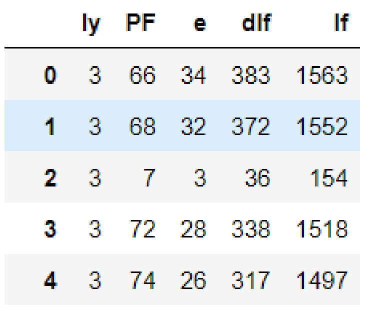
Polynomial Regression equation: $y = b_0 + b_1 x + \ b_2 x^2 + b_3 x^3 + \ldots + b_n x^n$

Polynomial Regression Steps

First step we imported the needed libraries to the polynomial regression. Then imported the dataset as a dataframe as shown in figure 1.

Figure 1

Dataframe

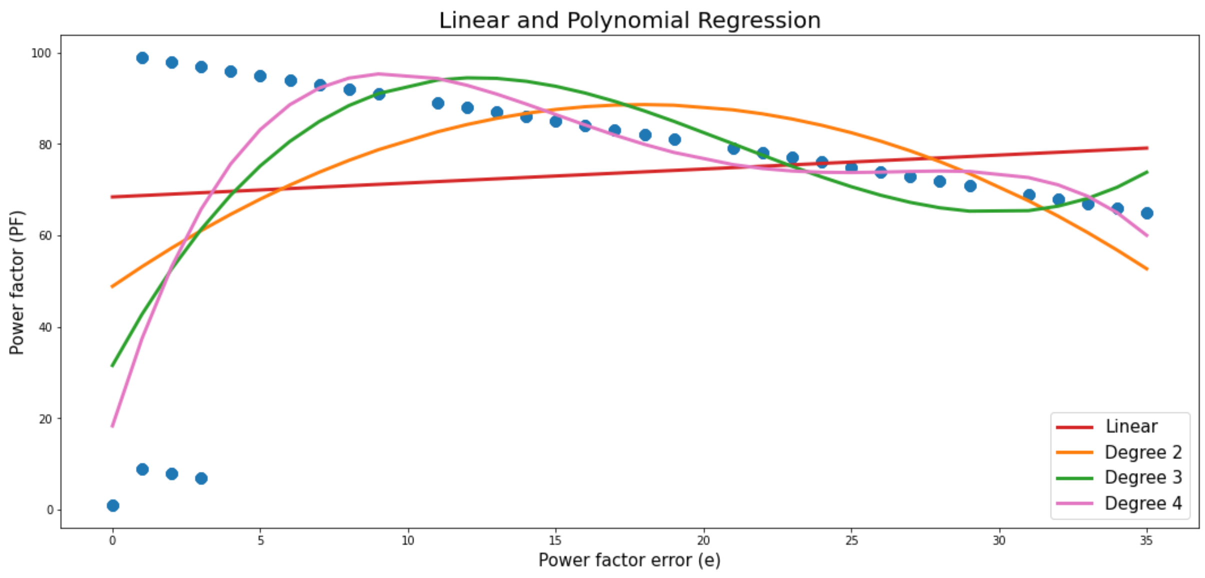


	ly	PF	e	dlf	lf
0	3	66	34	383	1563
1	3	68	32	372	1552
2	3	7	3	36	154
3	3	72	28	338	1518
4	3	74	26	317	1497

After that we extracted the x axis and y axis. x axis is “e” column which is **power factor error** and y is the **power factor**. Next, we created a linear and three polynomials regression models. Lastly, we visualized the model as shown in figure 2.

Figure 2

Linear and Polynomial chart



Python Code

```
# importing needed packages

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

#importing datasets
df = pd.read_csv('synchronous machine.csv')

#Extracting Independent and dependent Variable
x= df.iloc[:, [2]]
y= df.iloc[:,[1]]

i= x.values.ravel().argsort()
x= x.values.ravel()[i].reshape(-1,1)
y= y.values[i]

#Fitting the Linear and Polynomial Regression to the dataset
lin_reg= LinearRegression()
lin_reg.fit(x,y)

poly_regs_d2= PolynomialFeatures(degree= 2)
x_poly_d2= poly_regs_d2.fit_transform(x)
lin_reg_2 =LinearRegression()
lin_reg_2.fit(x_poly_d2, y)
```

```
poly_regs_d3 = PolynomialFeatures(degree= 3)
x_poly_d3= poly_regs_d3.fit_transform(x)
lin_reg_3 =LinearRegression()
lin_reg_3.fit(x_poly_d3, y)

poly_regs_d4 = PolynomialFeatures(degree= 4)
x_poly_d4= poly_regs_d4.fit_transform(x)
lin_reg_4 =LinearRegression()
lin_reg_4.fit(x_poly_d4, y)

#Visulaizing the result for Linear and Polynomial Regression
fig = plt.figure(figsize = (18,8))
plt.scatter(x,y,color="tab:blue",linewidth=4)
plt.plot(x,lin_reg.predict(x), color="tab:red",linewidth=3, label='Linear')
plt.plot(x, lin_reg_2.predict(x_poly_d2),color="tab:orange",linewidth=3,
label='Degree 2')
plt.plot(x, lin_reg_3.predict(x_poly_d3),color="tab:green",linewidth=3,
label='Degree 3')
plt.plot(x, lin_reg_4.predict(x_poly_d4),color="tab:pink",linewidth=3,
label='Degree 4' )
plt.title("Linear and Polynomial Regression",fontsize = 20)
plt.xlabel("Power factor error (e)", fontsize = 15)
plt.ylabel("Power factor (PF)", fontsize = 15)
plt.legend(loc='lower right',fontsize=15)
plt.show()
```

References

Kumar, V. (2018). *Scikit learn; cannot create plot for polynomial regression correctly*. Stack Overflow. Retrieved October 15, 2021, from https://stackoverflow.com/questions/50127606/scikit-learn-cannot-create-plot-for-polynomial-regression-correctly.

Machine learning polynomial regression - javatpoint. www.javatpoint.com. (n.d.). Retrieved October 15, 2021, from https://www.javatpoint.com/machine-learning-polynomial-regression.

BAYINDIR , R., & KAHRAMAN , H. T. (2021). *Synchronous Machine Data Set Data Set*. UCI Machine Learning Repository: Synchronous Machine Data Set Data Set. Retrieved October 15, 2021, from https://archive.ics.uci.edu/ml/datasets/Synchronous+Machine+Data+Set#.

Bonaccorso, G. (2018). *Machine learning algorithms: Popular algorithms for Data Science and Machine Learning*. Packt.