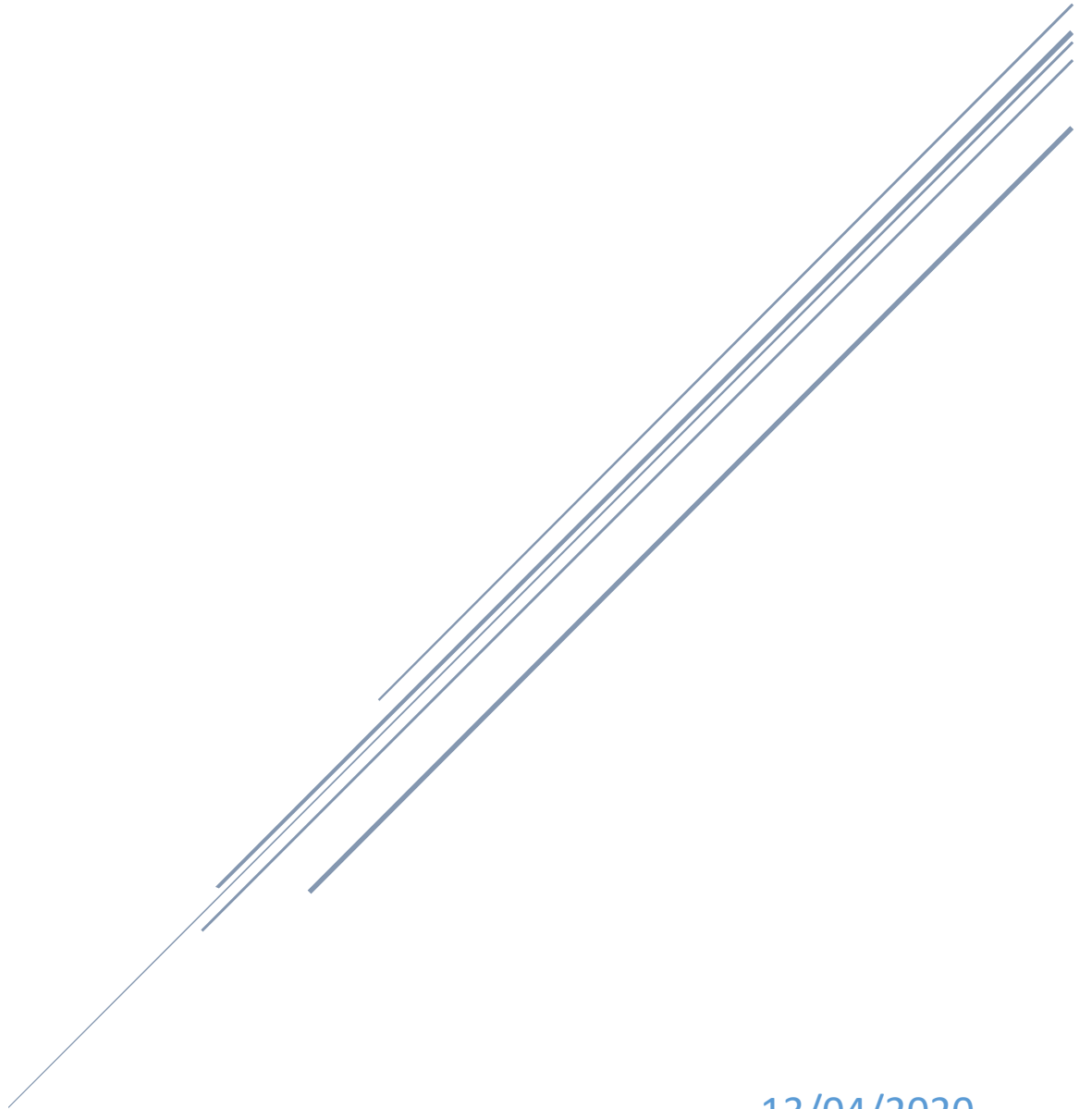


RUMAISA MARYAM

MACHINE LEARNING ASSIGNMENT 3



13/04/2020

Q1: Take 50 startups of any two countries and find out which country is going to provide best profit in future.

Decision Tree

Ans: *For this dataset I have used decision tree regression to solve the problem since it is a classification problem in which we are asked to predict and compare the profit of two cities.*

```

1  #MACHINE LEARNING ASSIGNMENT3
2  #RUMAISA MARYAM
3
4  # Importing the libraries
5  import numpy as np
6  import matplotlib.pyplot as plt
7  import pandas as pd
8
9  dataset = pd.read_csv('50_startups.csv')
10 dataset['Sum']=dataset[['R&D Spend', 'Administration', 'Marketing Spend']].sum(axis=1)
11
12 Xc = dataset.loc[(dataset.State=='California'),'Sum']
13 Yc = dataset.loc[(dataset.State=='California'),'Profit']
14
15
16 # Splitting the dataset into the Training set and Test set
17 """from sklearn.model_selection import train_test_split
18 Xc_train, Xc_test, Yc_train, Yc_test = train_test_split(Xc, Yc, test_size = 0.3, random_state = 0)"""
19 from sklearn.tree import DecisionTreeRegressor
20 regressor = DecisionTreeRegressor(random_state = 0)
21 regressor.fit(Xc, Yc)
22
23 # Visualising the Decision Tree Regression results (higher resolution)
24 plt.scatter(Xc, Yc, color = 'red')
25 plt.plot(Xc, regressor.predict(Xc), color = 'blue')
26 plt.title('California Spending vs Profit (Decision Tree Regression)')
27 plt.xlabel('Spending')
28 plt.ylabel('Profit')
29 plt.show()
30
31 # Predicting a new result with Decision Tree
32 Z = regressor.predict([[9000000]])
33 print ("California =" ,Z)
34
35 Xf = dataset.loc[(dataset.State=='Florida'),'Sum']
36 Yf = dataset.loc[(dataset.State=='Florida'),'Profit']
37

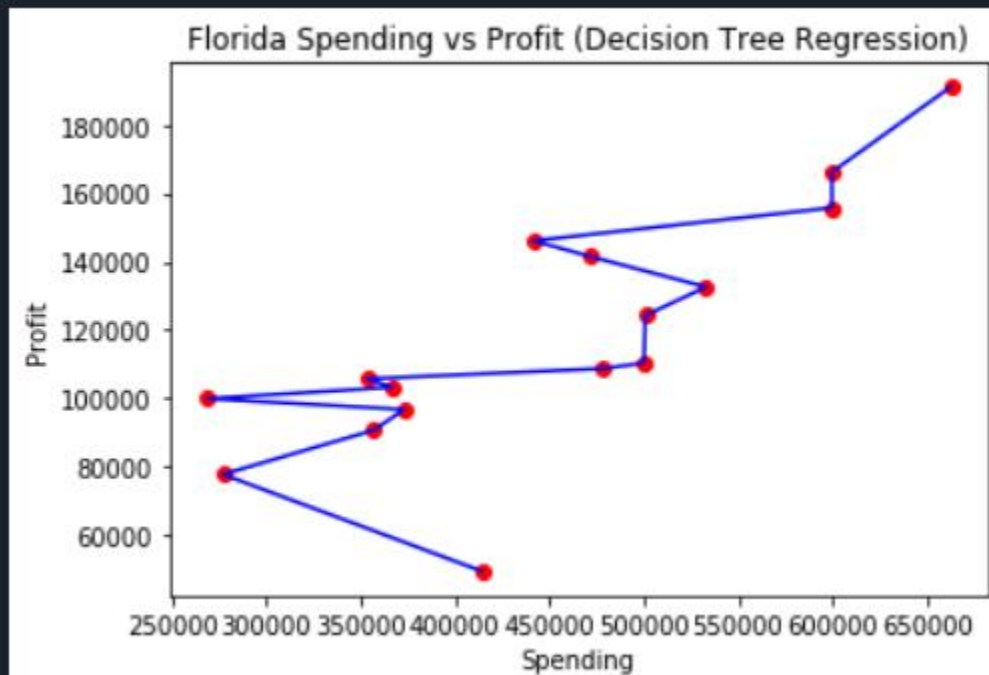
```

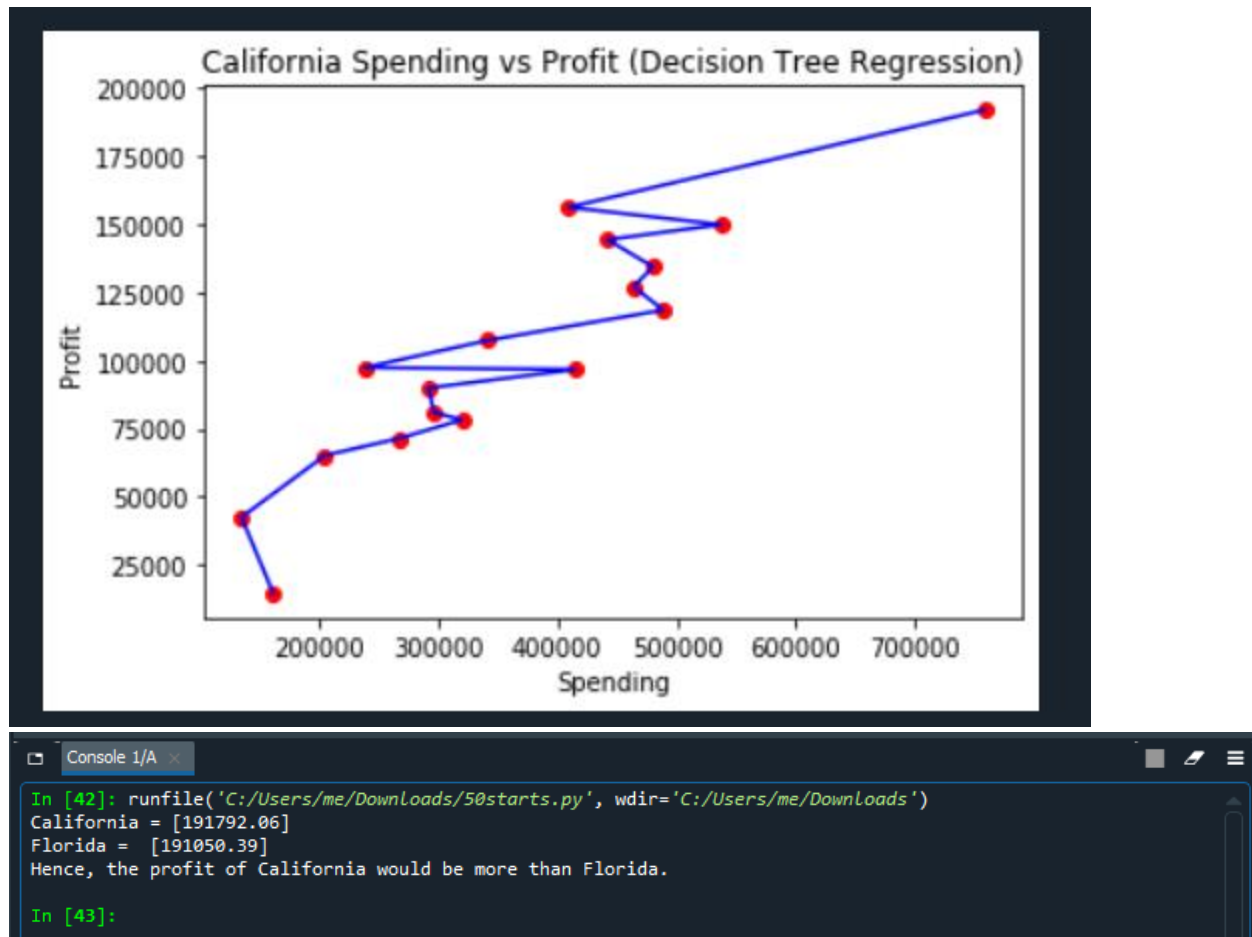
MACHINE LEARNING ASSIGNMENT 3

```

22
23 # Visualising the Decision Tree Regression results (higher resolution)
24 plt.scatter(Xc, Yc, color = 'red')
25 plt.plot(Xc, regressor.predict(Xc), color = 'blue')
26 plt.title('California Spending vs Profit (Decision Tree Regression)')
27 plt.xlabel('Spending')
28 plt.ylabel('Profit')
29 plt.show()
30
31 # Predicting a new result with Decision Tree
32 Z = regressor.predict([[9000000]])
33 print ("California = " ,Z)
34
35 Xf = dataset.loc[(dataset.State=='Florida'),'Sum']
36 Yf = dataset.loc[(dataset.State=='Florida'),'Profit']
37
38 # Splitting the dataset into the Training set and Test set
39 """from sklearn.model_selection import train_test_split
40 Xf_train, Xf_test, Yf_train, Yf_test = train_test_split(Xf, Yf, test_size = 0.3, random_state = 0)"""
41 from sklearn.tree import DecisionTreeRegressor
42 regressor = DecisionTreeRegressor(random_state = 0)
43 regressor.fit(Xf, Yf)
44
45 # Visualising the Decision Tree Regression results (higher resolution)
46 plt.scatter(Xf, Yf, color = 'red')
47 plt.plot(Xf, regressor.predict(Xf), color = 'blue')
48 plt.title('Florida Spending vs Profit (Decision Tree Regression)')
49 plt.xlabel('Spending')
50 plt.ylabel('Profit')
51 plt.show()
52
53 # Predicting a new result with Decision Tree
54 Z1= regressor.predict([[9000000]])
55 print ("Florida = " ,Z1)
56
57 print("Hence, the profit of California would be more than Florida.")

```





Q2: Annual temperature between two industries is given. Predict the temperature in 2016 and 2017 using the past data of both country.

Polynomial Regression

Ans: *For the prediction of annual temperature of two industries I have applied both linear and polynomial regression but if we look carefully its evident that polynomial regression produces the best fit line better than linear regression since the loss is less in it. So the temperature of GISTEMP is more than*

GCAG In 2016 and 2017.

```

1  #MACHINE LEARNING ASSIGNMENT3
2  #RUMAISA MARYAM
3
4  # Importing the libraries
5  import numpy as np
6  import matplotlib.pyplot as plt
7  import pandas as pd
8
9  dataset = pd.read_csv('annual_temp.csv')
10
11  #FOR THE FIRST INDUSTRY GCAG
12  A = dataset.loc[(dataset.Source == 'GCAG'), ['Year']]
13  B = dataset.loc[(dataset.Source == 'GCAG'), ['Mean']]
14
15  # Splitting the dataset into the Training set and Test set
16  """from sklearn.cross_validation import train_test_split
17  A_train, A_test, B_train, B_test = train_test_split(A, B, test_size = 0.2, random_state = 0)"""
18
19  # Fitting Linear Regression to the dataset
20  from sklearn.linear_model import LinearRegression
21  reg = LinearRegression()
22  reg.fit(A, B)
23
24  # Fitting Polynomial Regression to the dataset
25  from sklearn.preprocessing import PolynomialFeatures
26  polyreg = PolynomialFeatures(degree = 4)
27  A_poly = polyreg.fit_transform(A)
28  polyreg.fit(A_poly, B)
29  lin_reg_2 = LinearRegression()
30  lin_reg_2.fit(A_poly, B)
31
32  # Visualising the Linear Regression results
33  plt.scatter(A, B, color = 'red')
34  plt.plot(A, reg.predict(A), color = 'Black')
35  plt.title('Years VS Annual Temperature For GCAG (Linear Regression)')
36  plt.xlabel('Years')
37  plt.ylabel('Temperature')
38
39  """from sklearn.cross_validation import train_test_split
40  A_train, A_test, B_train, B_test = train_test_split(A, B, test_size = 0.2, random_state = 0)"""
41
42  # Fitting Linear Regression to the dataset
43  from sklearn.linear_model import LinearRegression
44  reg = LinearRegression()
45  reg.fit(A, B)
46
47  # Fitting Polynomial Regression to the dataset
48  from sklearn.preprocessing import PolynomialFeatures
49  polyreg = PolynomialFeatures(degree = 4)
50  A_poly = polyreg.fit_transform(A)
51  polyreg.fit(A_poly, B)
52  lin_reg_2 = LinearRegression()
53  lin_reg_2.fit(A_poly, B)
54
55  # Visualising the Linear Regression results
56  plt.scatter(A, B, color = 'red')
57  plt.plot(A, reg.predict(A), color = 'Black')
58  plt.title('Years VS Annual Temperature For GCAG (Linear Regression)')
59  plt.xlabel('Years')
60  plt.ylabel('Temperature')
61  plt.show()
62
63  # Visualising the Polynomial Regression results
64  plt.scatter(A, B, color = 'red')
65  plt.plot(A, lin_reg_2.predict(polyreg.fit_transform(A)), color = 'Black')
66  plt.title('Year VS Annual TemperatureFor GCAG(Polynomial Regression)')
67  plt.xlabel('Year')
68  plt.ylabel('Temperature')
69  plt.show()
70
71  # Predicting a new result with Linear Regression
72  X=reg.predict([[2016]])
73  X1=reg.predict([[2017]])
74  print("The result with linear regression for GCAG in 2016 is" , X)
75  print("The result with linear regression for GCAG in 2017 is" , X1)

```

MACHINE LEARNING ASSIGNMENT 3

```

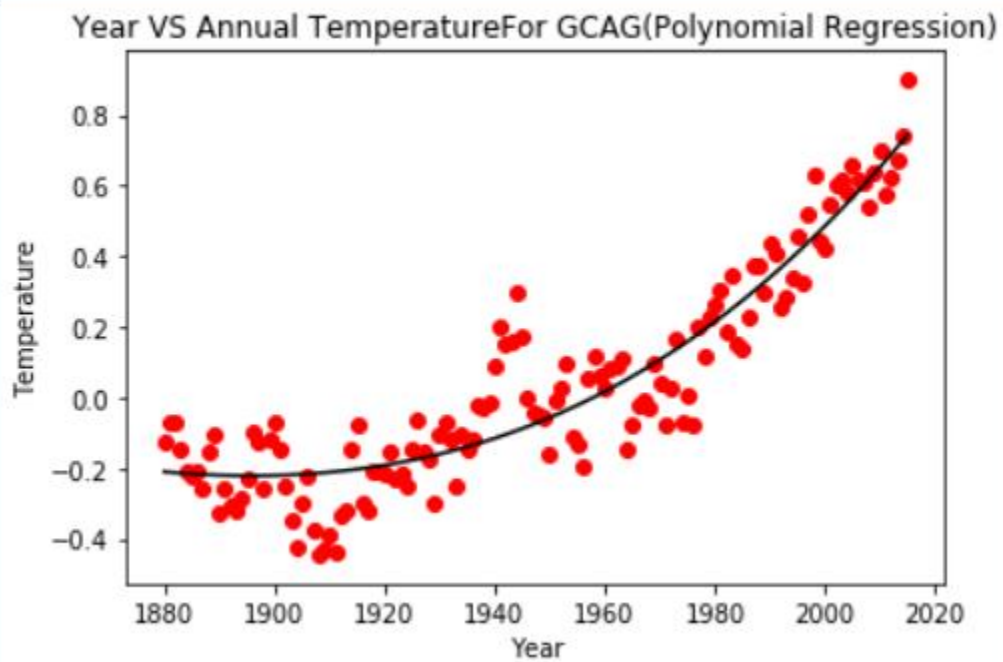
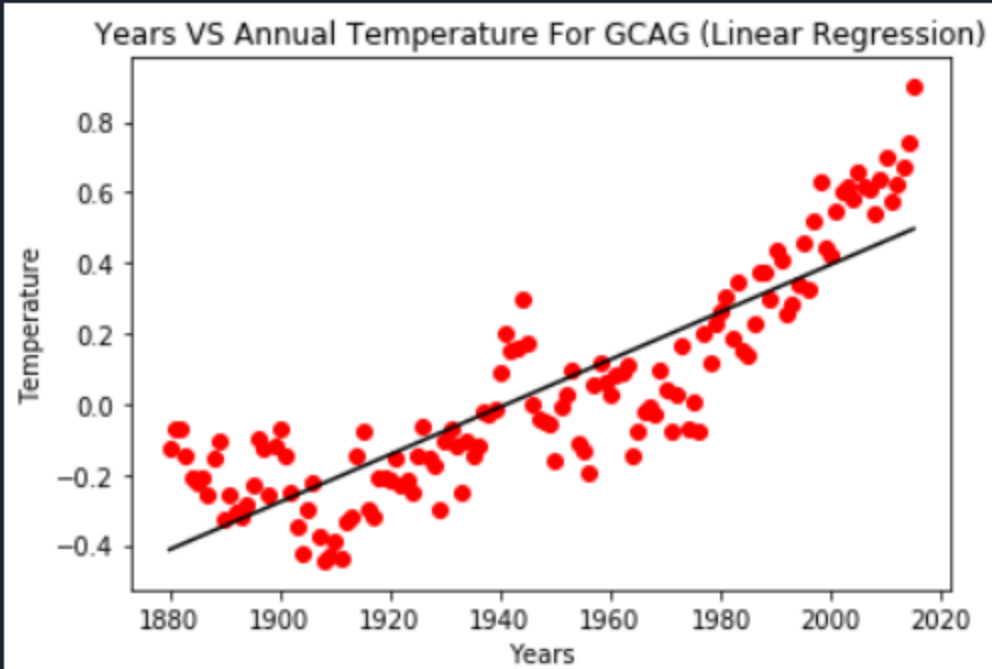
50 X1=reg.predict([[2017]])
51 print("The result with linear regression for GCAG in 2016 is" , X)
52 print("The result with linear regression for GCAG in 2017 is" , X1)
53
54 # Predicting a new result with Polynomial Regression
55 Y=lin_reg_2.predict(polyreg.fit_transform([[2016]]))
56 Y1=lin_reg_2.predict(polyreg.fit_transform([[2017]]))
57 print("The result with polynomial regression for GCAG in 2016 is" , Y)
58 print("The result with polynomial regression for GCAG in 2017 is" , Y1)
59
60
61 #FOR THE SECOND INDUSTRY GISTEMP
62 A1 = dataset.loc[(dataset.Source == 'GISTEMP'), ['Year']]
63 B1 = dataset.loc[(dataset.Source == 'GISTEMP'), ['Mean']]
64
65 # Splitting the dataset into the Training set and Test set
66 """from sklearn.cross_validation import train_test_split
67 A1_train, A1_test, B1_train, B1_test = train_test_split(A1, B1, test_size = 0.2, random_state = 0)"""
68
69 # Fitting Linear Regression to the dataset
70 from sklearn.linear_model import LinearRegression
71 reg = LinearRegression()
72 reg.fit(A1, B1)
73
74 # Fitting Polynomial Regression to the dataset
75 from sklearn.preprocessing import PolynomialFeatures
76 polyreg = PolynomialFeatures(degree = 4)
77 A1_poly = polyreg.fit_transform(A1)
78 polyreg.fit(A1_poly, B1)
79 lin_reg_2 = LinearRegression()
80 lin_reg_2.fit(A1_poly, B1)
81
82 # Visualising the Linear Regression results
83 plt.scatter(A1, B1, color = 'red')
84 plt.plot(A1, reg.predict(A1), color = 'Black')
85 plt.title('Years VS Annual Temperature For GISTEMP (Linear Regression)')
86 plt.xlabel('Years')

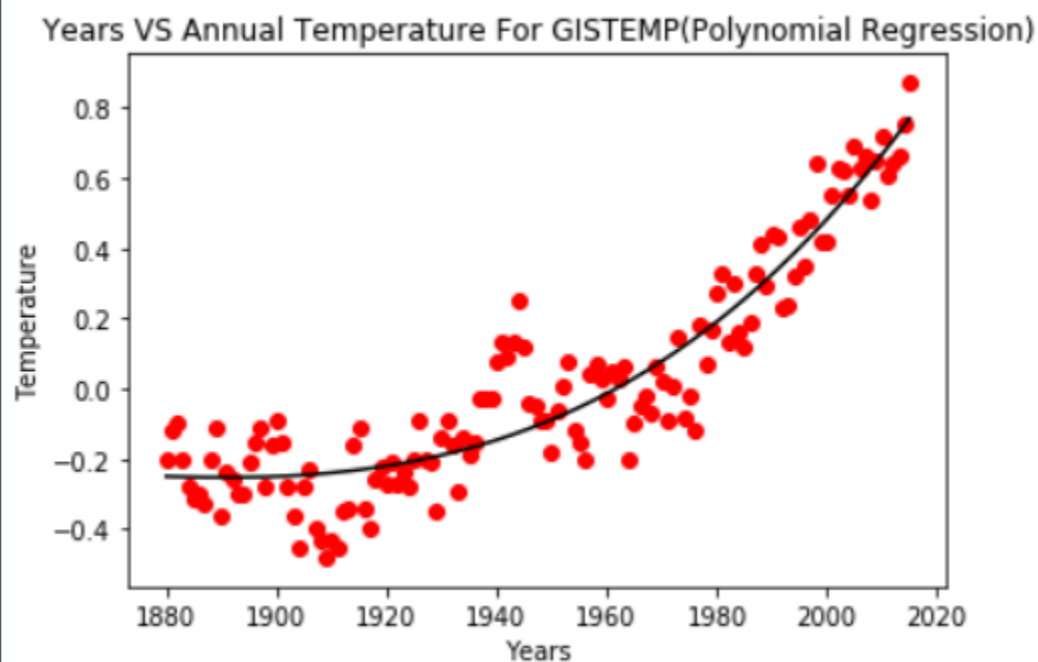
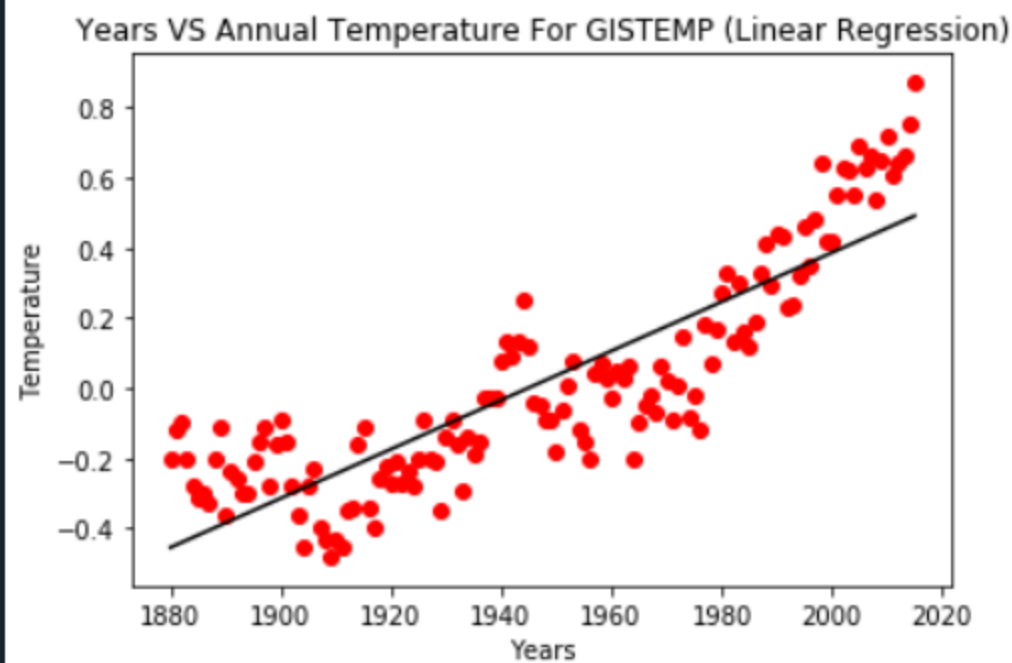
```

```

72 reg.fit(A1, B1)
73
74 # Fitting Polynomial Regression to the dataset
75 from sklearn.preprocessing import PolynomialFeatures
76 polyreg = PolynomialFeatures(degree = 4)
77 A1_poly = polyreg.fit_transform(A1)
78 polyreg.fit(A1_poly, B1)
79 lin_reg_2 = LinearRegression()
80 lin_reg_2.fit(A1_poly, B1)
81
82 # Visualising the Linear Regression results
83 plt.scatter(A1, B1, color = 'red')
84 plt.plot(A1, reg.predict(A1), color = 'Black')
85 plt.title('Years VS Annual Temperature For GISTEMP (Linear Regression)')
86 plt.xlabel('Years')
87 plt.ylabel('Temperature')
88 plt.show()
89
90 # Visualising the Polynomial Regression results
91 plt.scatter(A1, B1, color = 'red')
92 plt.plot(A1, lin_reg_2.predict(polyreg.fit_transform(A1)), color = 'Black')
93 plt.title('Years VS Annual Temperature For GISTEMP(Polynomial Regression)')
94 plt.xlabel('Years')
95 plt.ylabel('Temperature')
96 plt.show()
97
98 # Predicting a new result with Linear Regression
99 X2=reg.predict([[2016]])
100 X3=reg.predict([[2017]])
101 print("The result with linear regression for GCAG in 2016 is" , X2)
102 print("The result with linear regression for GCAG in 2017 is" , X3)
103
104 # Predicting a new result with Polynomial Regression
105 Y2=lin_reg_2.predict(polyreg.fit_transform([[2016]]))
106 Y3=lin_reg_2.predict(polyreg.fit_transform([[2017]]))
107 print("The result with polynomial regression for GCAG in 2016 is" , Y2)
108 print("The result with polynomial regression for GCAG in 2017 is" , Y3)

```





```
In [43]: runfile('C:/Users/me/Downloads/annualtemp.py', wdir='C:/Users/me/Downloads')
The result with linear regression for GCAG in 2016 is [[0.50298425]]
The result with linear regression for GCAG in 2017 is [[0.50972011]]
The result with polynomial regression for GCAG in 2016 is [[0.76231028]]
The result with polynomial regression for GCAG in 2017 is [[0.78149969]]
The result with linear regression for GCAG in 2016 is [[0.49777778]]
The result with linear regression for GCAG in 2017 is [[0.50477625]]
The result with polynomial regression for GCAG in 2016 is [[0.78885745]]
The result with polynomial regression for GCAG in 2017 is [[0.81039365]]
```


Q3: Data of global production of CO2 of a place is given between 1970s to 2010. Predict the CO2 production for the years 2011, 2012 and 2013 using the old data set.

Polynomial Regression

Ans: *For this dataset I have applied all three regression but when seeing the results its evident that **polynomial regression** is the best form of regression here since it produces the best fit line better than linear regression and decision tree. Also the result for decision tree remains the same for different values of prediction.*

```

1  #MACHINE LEARNING ASSIGNMENT3
2  #RUMAISA MARYAM
3
4  # Importing the libraries
5  import numpy as np
6  import matplotlib.pyplot as plt
7  import pandas as pd
8
9  # Importing the dataset
10 dataset = pd.read_csv('global_co2.csv')
11 A = dataset.iloc[:,0:1].values
12 B = dataset.iloc[:, 1].values
13
14
15 # Splitting the dataset into the Training set and Test set
16 from sklearn.model_selection import train_test_split
17 A_train, A_test, B_train, B_test = train_test_split(A, B, test_size = 0.2, random_state = 0)
18
19
20 # Fitting Linear Regression to the dataset
21 from sklearn.linear_model import LinearRegression
22 lin_regressor = LinearRegression()
23 lin_regressor.fit(A, B)
24
25 # Fitting Polynomial Regression to the dataset
26 from sklearn.preprocessing import PolynomialFeatures
27 poly_reg = PolynomialFeatures(degree = 4)
28 A_poly = poly_reg.fit_transform(A)
29 poly_reg.fit(A_poly, B)
30 lin_reg_2 = LinearRegression()
31 lin_reg_2.fit(A_poly, B)
32
33 # Visualising the Linear Regression results
34 plt.scatter(A, B, color = 'red')
35 plt.plot(A, lin_regressor.predict(A), color = 'Black')
36 plt.title('Year VS co2 produced (Linear Regression)')
37 plt.xlabel('Year')

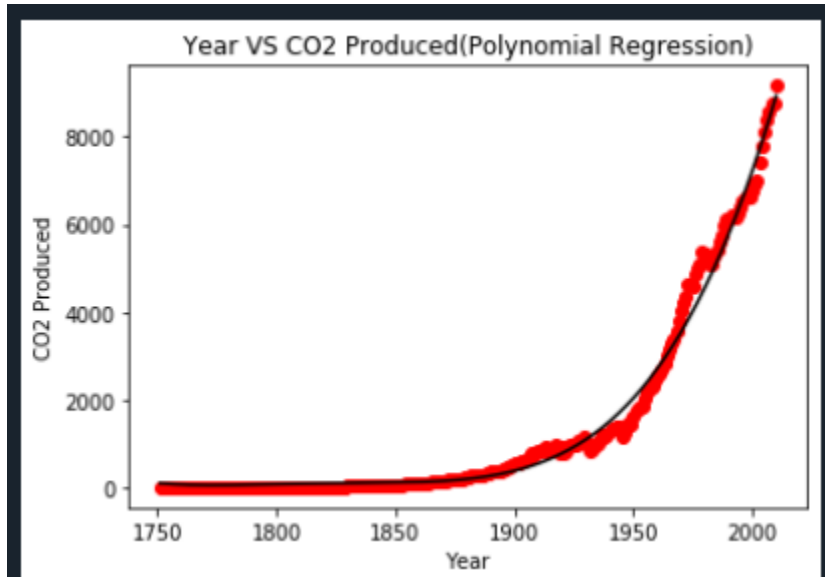
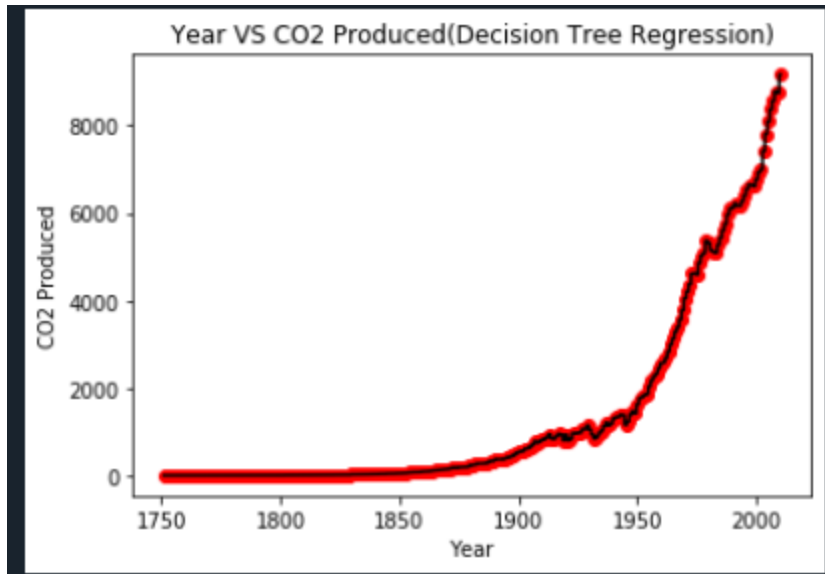
```

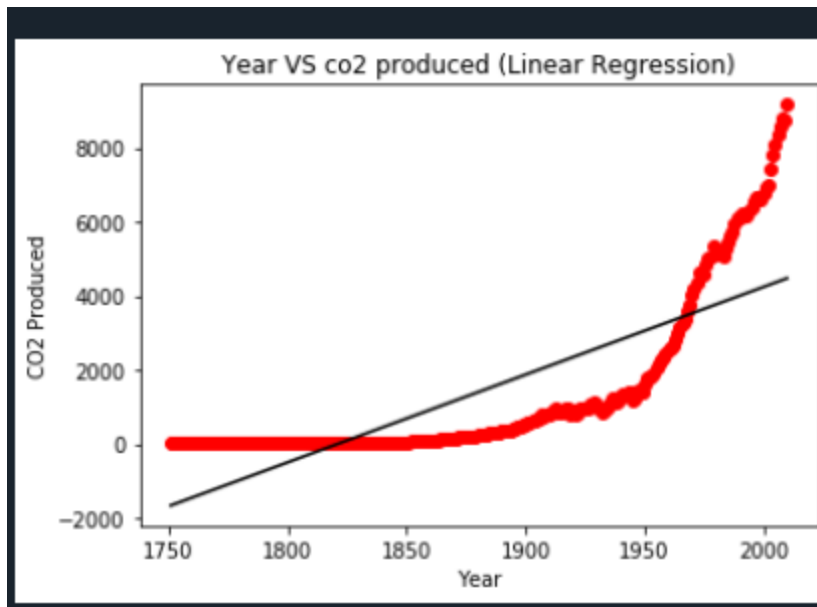
MACHINE LEARNING ASSIGNMENT 3

```

31 lin_reg_2.fit(A_poly, B)
32
33 # Visualising the Linear Regression results
34 plt.scatter(A, B, color = 'red')
35 plt.plot(A, lin_regressor.predict(A), color = 'Black')
36 plt.title('Year VS co2 produced (Linear Regression)')
37 plt.xlabel('Year')
38 plt.ylabel('CO2 Produced')
39 plt.show()
40
41 # Visualising the Polynomial Regression results
42 plt.scatter(A, B, color = 'red')
43 plt.plot(A, lin_reg_2.predict(poly_reg.fit_transform(A)), color = 'Black')
44 plt.title('Year VS co2 produced (Polynomial Regression)')
45 plt.xlabel('Year')
46 plt.ylabel('CO2 Produced')
47 plt.show()
48
49 # Visualising the Polynomial Regression results (for higher resolution and smoother curve)
50 A_grid = np.arange(min(A), max(A), 0.1)
51 A_grid = A_grid.reshape((len(A_grid), 1))
52 plt.scatter(A, B, color = 'red')
53 plt.plot(A_grid, lin_reg_2.predict(poly_reg.fit_transform(A_grid)), color = 'Black')
54 plt.title('Year VS CO2 Produced(Polynomial Regression)')
55 plt.xlabel('Year')
56 plt.ylabel('CO2 Produced')
57 plt.show()
58
59 # Fitting Decision Tree Regression to the dataset
60 from sklearn.tree import DecisionTreeRegressor
61 regressor = DecisionTreeRegressor(random_state = 0)
62 regressor.fit(A, B)
63
64 # Visualising the Decision Tree Regression results (higher resolution)
65 A_grid = np.arange(min(A), max(A), 0.01)
66 A_grid = A_grid.reshape((len(A_grid), 1))
67 plt.scatter(A, B, color = 'red')
68 plt.plot(A_grid, regressor.predict(A_grid), color = 'BLACK')
69 plt.title('Year VS CO2 Produced(Decision Tree Regression)')
70 plt.xlabel('Year')
71 plt.ylabel('CO2 Produced')
72 plt.show()
73
74 # Predicting a new result with Linear Regression
75 X=lin_regressor.predict([[2011]])
76 X1=lin_regressor.predict([[2012]])
77 X2=lin_regressor.predict([[2013]])
78 print("The result with linear regression for co2 produced in 2011 is" , X)
79 print("The result with linear regression for co2 produced in 2012 is" , X1)
80 print("The result with linear regression for co2 produced in 2013 is" , X2)
81
82 # Predicting a new result with Polynomial Regression
83 Y=lin_reg_2.predict(poly_reg.fit_transform([[2011]]))
84 Y1=lin_reg_2.predict(poly_reg.fit_transform([[2012]]))
85 Y2=lin_reg_2.predict(poly_reg.fit_transform([[2013]]))
86 print("The result with polynomial regression for co2 produced in 2011 is" , Y)
87 print("The result with polynomial regression for co2 produced in 2012 is" , Y1)
88 print("The result with polynomial regression for co2 produced in 2013 is" , Y2)
89
90 # Predicting a new result with Decision Tree
91 Z = regressor.predict([[2011]])
92 Z1 = regressor.predict([[2012]])
93 Z2= regressor.predict([[2013]])
94 print("The result with decision tree for co2 produced in 2011 is" , Z)
95 print("The result with decision tree for co2 produced in 2012 is" , Z1)
96 print("The result with decision tree for co2 produced in 2013 is" , Z2)
97

```





```
In [45]: runfile('C:/Users/me/Downloads/globalCO2.py', wdir='C:/Users/me/Downloads')
The result with linear regression for co2 produced in 2011 is [4494.86418176]
The result with linear regression for co2 produced in 2012 is [4518.55824859]
The result with linear regression for co2 produced in 2013 is [4542.25231541]
The result with polynomial regression for co2 produced in 2011 is [9138.92033747]
The result with polynomial regression for co2 produced in 2012 is [9329.39530137]
The result with polynomial regression for co2 produced in 2013 is [9522.85598981]
The result with decision tree for co2 produced in 2011 is [9167.]
The result with decision tree for co2 produced in 2012 is [9167.]
The result with decision tree for co2 produced in 2013 is [9167.]
```

Q4: Housing price according to the ID is assigned to every-house. Perform future analysis where when ID is inserted the housing price is displayed.

Polynomial Regression

Ans: *For this dataset I have used all forms of regression but observing closely its NOTED that **polynomial regression** provides the best and accurate result since it produces the best fit line better than linear regression. As for decision tree its observed that the prediction for different set of IDs remain the*

same so its not useful at all here.

```

1  #MACHINE LEARNING ASSIGNMENT3
2  #RUMAIISA MARYAM
3
4  # Importing the libraries
5  import numpy as np
6  import matplotlib.pyplot as plt
7  import pandas as pd
8
9  # Importing the Annual Temperature dataset
10 dataset = pd.read_csv('housing_price.csv')
11 A = dataset.iloc[:,0:1].values
12 B = dataset.iloc[:, 1].values
13
14
15 # Splitting the dataset into the Training set and Test set
16 from sklearn.model_selection import train_test_split
17 A_train, A_test, B_train, B_test = train_test_split(A, B, test_size = 0.2, random_state = 0)
18
19
20 # Fitting Linear Regression to the dataset
21 from sklearn.linear_model import LinearRegression
22 lin_regressor = LinearRegression()
23 lin_regressor.fit(A, B)
24
25 # Fitting Polynomial Regression to the dataset
26 from sklearn.preprocessing import PolynomialFeatures
27 poly_reg = PolynomialFeatures(degree = 4)
28 A_poly = poly_reg.fit_transform(A)
29 poly_reg.fit(A_poly, B)
30 lin_reg_2 = LinearRegression()
31 lin_reg_2.fit(A_poly, B)
32
33 # Visualising the Linear Regression results
34 plt.scatter(A, B, color = 'red')
35 plt.plot(A, lin_regressor.predict(A), color = 'Black')
36 plt.title('Id VS House Price (Linear Regression)')
37 plt.xlabel('Id')

```

```

40
41 # Visualising the Polynomial Regression results
42 plt.scatter(A, B, color = 'red')
43 plt.plot(A, lin_reg_2.predict(poly_reg.fit_transform(A)), color = 'Black')
44 plt.title('Id VS House Price (Polynomial Regression)')
45 plt.xlabel('Id')
46 plt.ylabel('House Price')
47 plt.show()
48
49 # Visualising the Polynomial Regression results (for higher resolution and smoother curve)
50 A_grid = np.arange(min(A), max(A), 0.1)
51 A_grid = A_grid.reshape((len(A_grid), 1))
52 plt.scatter(A, B, color = 'red')
53 plt.plot(A_grid, lin_reg_2.predict(poly_reg.fit_transform(A_grid)), color = 'Black')
54 plt.title('Id VS House Price (Polynomial Regression)')
55 plt.xlabel('Id')
56 plt.ylabel('House Price')
57 plt.show()
58
59 # Fitting Decision Tree Regression to the dataset
60 from sklearn.tree import DecisionTreeRegressor
61 regressor = DecisionTreeRegressor(random_state = 0)
62 regressor.fit(A, B)
63
64 # Visualising the Decision Tree Regression results (higher resolution)
65 A_grid = np.arange(min(A), max(A), 0.01)
66 A_grid = A_grid.reshape((len(A_grid), 1))
67 plt.scatter(A, B, color = 'red')
68 plt.plot(A_grid, regressor.predict(A_grid), color = 'BLACK')
69 plt.title('Id VS House Price (Decision Tree Regression)')
70 plt.xlabel('Id')
71 plt.ylabel('House Price')
72 plt.show()
73
74 # Predicting a new result with Linear Regression
75 X=lin_regressor.predict([[2920]])

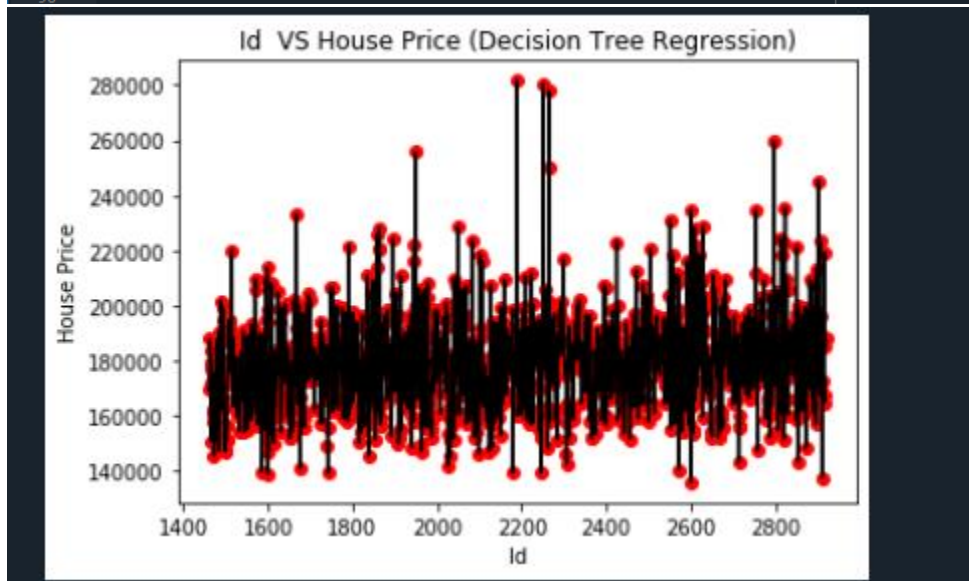
```

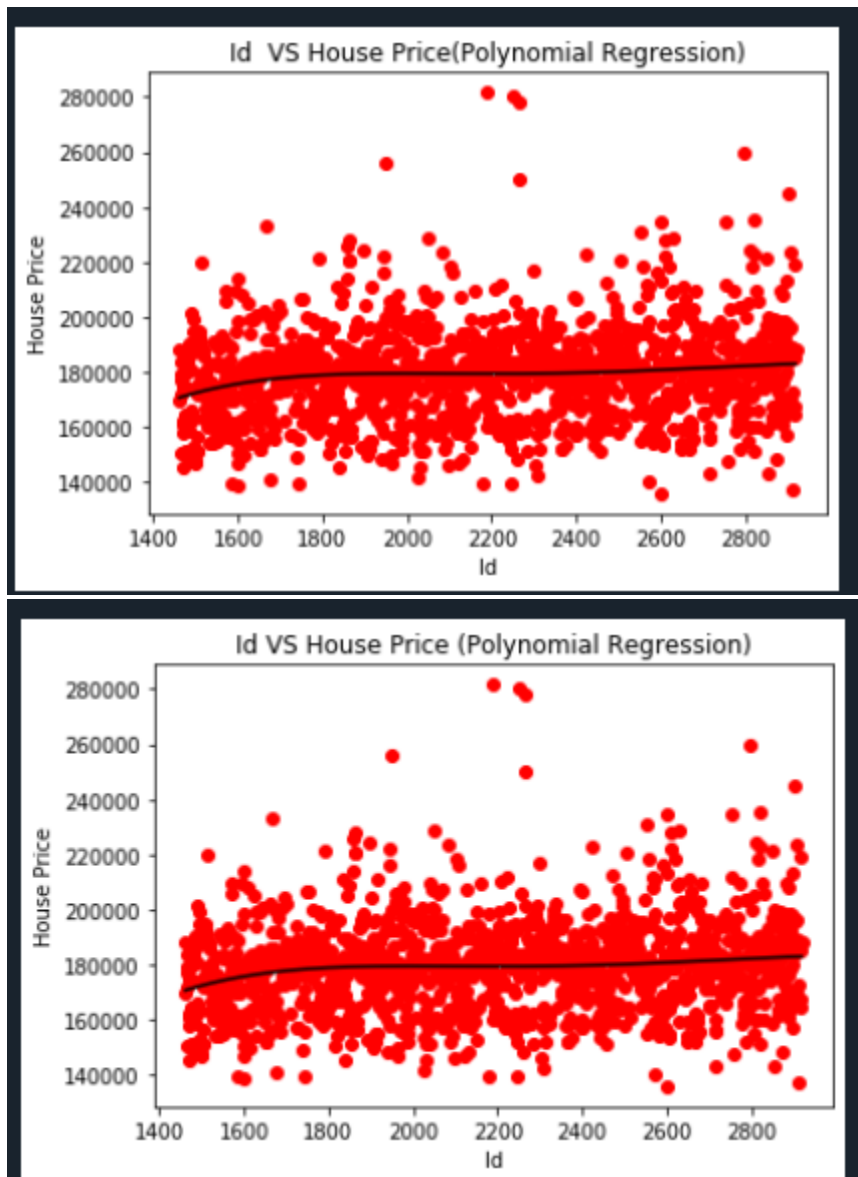
MACHINE LEARNING ASSIGNMENT 3

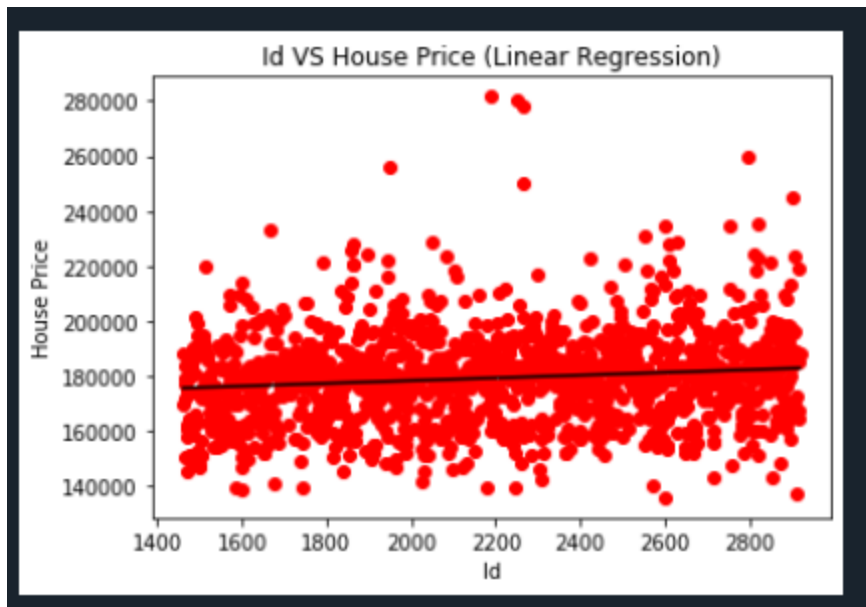
```

nomial_regression_headweightset.py x ann.py x ann_new.py x polynomial_regression.py x simple_linear_regression.py x Annual_temp.py x houseprice.py* x
55 plt.xlabel('Id')
56 plt.ylabel('House Price')
57 plt.show()
58
59 # Fitting Decision Tree Regression to the dataset
60 from sklearn.tree import DecisionTreeRegressor
61 regressor = DecisionTreeRegressor(random_state = 0)
62 regressor.fit(A, B)
63
64 # Visualising the Decision Tree Regression results (higher resolution)
65 A_grid = np.arange(min(A), max(A), 0.01)
66 A_grid = A_grid.reshape((len(A_grid), 1))
67 plt.scatter(A, B, color = 'red')
68 plt.plot(A_grid, regressor.predict(A_grid), color = 'BLACK')
69 plt.title('Id VS House Price (Decision Tree Regression)')
70 plt.xlabel('Id')
71 plt.ylabel('House Price')
72 plt.show()
73
74 # Predicting a new result with Linear Regression
75 X=lin_regressor.predict([[2920]])
76 print("The result with linear regression for house id 2920 is" , X)
77
78
79 # Predicting a new result with Polynomial Regression
80 Y=lin_reg_2.predict(poly_reg.fit_transform([[2920]]))
81 print("The result with polynomial regression for house id 2920 is" , Y)
82
83
84 # Predicting a new result with Decision Tree
85 Z = regressor.predict([[2920]])
86 Z1= regressor.predict([[2929]])
87 print("The result with decision tree for house id 2920 is" , Z)
88 print("The result with decision tree for house id 2929 is" , Z1)
89
90

```







```
In [47]: runfile('C:/Users/me/Downloads/houseprice.py', wdir='C:/Users/me/Downloads')
The result with linear regression for house id 2920 is [182794.79499595]
The result with polynomial regression for house id 2920 is [182886.73493585]
The result with decision tree for house id 2920 is [187741.8667]
The result with decision tree for house id 2929 is [187741.8667]

In [48]: |
```

Q5: Data of monthly experience and income distribution of different employs is given. Perform regression.

Polynomial Regression

Ans: *For this dataset I have used all three regressions but by observing closely its found out that **polynomial regression** is the best type of regression since it produces the best fit line better than linear regression. As for decision tree its found out that it produces the same answers for different values,so it cant be used here.*

MACHINE LEARNING ASSIGNMENT 3

```
1  #MACHINE LEARNING ASSIGNMENT3
2  #RUMAIISA MARYAM
3
4  # Importing the libraries
5  import numpy as np
6  import matplotlib.pyplot as plt
7  import pandas as pd
8
9  # Importing the Annual Temperature dataset
10 dataset = pd.read_csv('expense.csv')
11 A = dataset.iloc[:,0:1].values
12 B = dataset.iloc[:, 1:2].values
13
14
15 # Splitting the dataset into the Training set and Test set
16 from sklearn.model_selection import train_test_split
17 A_train, A_test, B_train, B_test = train_test_split(A, B, test_size = 0.2, random_state = 0)
18
19
20 # Fitting Linear Regression to the dataset
21 from sklearn.linear_model import LinearRegression
22 lin_regressor = LinearRegression()
23 lin_regressor.fit(A, B)
24
25 # Fitting Polynomial Regression to the dataset
26 from sklearn.preprocessing import PolynomialFeatures
27 poly_reg = PolynomialFeatures(degree = 4)
28 A_poly = poly_reg.fit_transform(A)
29 poly_reg.fit(A_poly, B)
30 lin_reg_2 = LinearRegression()
31 lin_reg_2.fit(A_poly, B)
32
33 # Visualising the Linear Regression results
34 plt.scatter(A, B, color = 'red')
35 plt.plot(A, lin_regressor.predict(A), color = 'Black')
36 plt.title('Monthly Expense VS Income (Linear Regression)')
37 plt.xlabel('Monthly Expense')
```

MACHINE LEARNING ASSIGNMENT 3

```

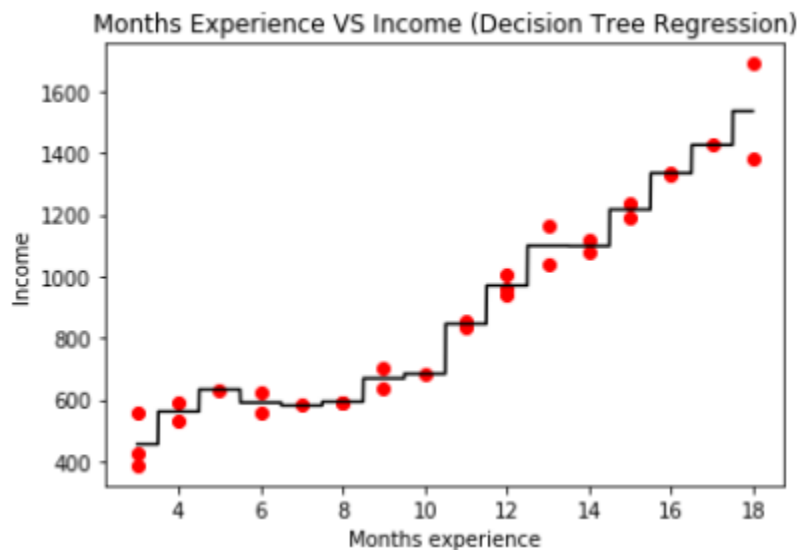
40
41 # Visualising the Polynomial Regression results
42 plt.scatter(A, B, color = 'red')
43 plt.plot(A, lin_reg_2.predict(poly_reg.fit_transform(A)), color = 'Black')
44 plt.title('Months Experience VS Income (Polynomial Regression)')
45 plt.xlabel('Months experience')
46 plt.ylabel('Income')
47 plt.show()
48
49 # Fitting Decision Tree Regression to the dataset
50 from sklearn.tree import DecisionTreeRegressor
51 regressor = DecisionTreeRegressor(random_state = 0)
52 regressor.fit(A, B)
53
54 # Visualising the Decision Tree Regression results (higher resolution)
55 A_grid = np.arange(min(A), max(A), 0.01)
56 A_grid = A_grid.reshape((len(A_grid), 1))
57 plt.scatter(A, B, color = 'red')
58 plt.plot(A_grid, regressor.predict(A_grid), color = 'BLACK')
59 plt.title('Months Experience VS Income (Decision Tree Regression)')
60 plt.xlabel('Months experience')
61 plt.ylabel('Income')
62 plt.show()
63
64 # Predicting a new result with Linear Regression
65 X=lin_regressor.predict([[19]])
66 print("The result with linear regression for 19 months of experince is" , X)
67
68
69 # Predicting a new result with Polynomial Regression
70 Y=lin_reg_2.predict(poly_reg.fit_transform([[19]]))
71 Y1=lin_reg_2.predict(poly_reg.fit_transform([[50]]))
72 print("The result with polynomial regression for 19 months of experince is" , Y)
73 print("The result with polynomial regression for 50 months of experince is" , Y1)
74
75
76 # Predicting a new result with Decision Tree

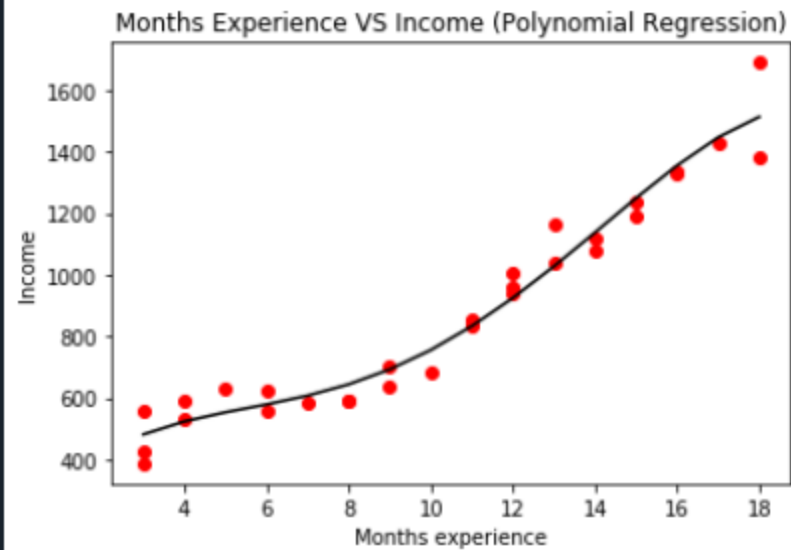
```

```

48
49 # Fitting Decision Tree Regression to the dataset
50 from sklearn.tree import DecisionTreeRegressor
51 regressor = DecisionTreeRegressor(random_state = 0)
52 regressor.fit(A, B)
53
54 # Visualising the Decision Tree Regression results (higher resolution)
55 A_grid = np.arange(min(A), max(A), 0.01)
56 A_grid = A_grid.reshape((len(A_grid), 1))
57 plt.scatter(A, B, color = 'red')
58 plt.plot(A_grid, regressor.predict(A_grid), color = 'BLACK')
59 plt.title('Months Experience VS Income (Decision Tree Regression)')
60 plt.xlabel('Months experience')
61 plt.ylabel('Income')
62 plt.show()
63
64 # Predicting a new result with Linear Regression
65 X=lin_regressor.predict([[19]])
66 print("The result with linear regression for 19 months of experince is" , X)
67
68
69 # Predicting a new result with Polynomial Regression
70 Y=lin_reg_2.predict(poly_reg.fit_transform([[19]]))
71 Y1=lin_reg_2.predict(poly_reg.fit_transform([[50]]))
72 print("The result with polynomial regression for 19 months of experince is" , Y)
73 print("The result with polynomial regression for 50 months of experince is" , Y1)
74
75
76 # Predicting a new result with Decision Tree
77 Z = regressor.predict([[19]])
78 Z1 = regressor.predict([[50]])
79 print("The result with decision tree for 19 months of experience is" , Z)
80 print("The result with decision tree for 50 months of experience is" , Z1)
81

```





```
In [51]: runfile('C:/Users/me/Downloads/experience.py', wdir='C:/Users/me/Downloads')
The result with linear regression for 19 months of experience is [[1486.2554603]]
The result with polynomial regression for 19 months of experience is [[1543.11034559]]
The result with polynomial regression for 50 months of experience is [[-153767.94019964]]
The result with decision tree for 19 months of experience is [1536.]
The result with decision tree for 50 months of experience is [1536.]
```

```
In [52]: |
```