Faculty of Engineering
Cairo University

DSP Assignment #1

Name	Section	BN
Ahmed Radwan GadELRab	1	12
Mohamed Ismail Amer	3	42
Moamen Nasser Saad	3	37

Submitted to: Dr. Mohsen Rashawn

Problem 1.

a, b)

Linear:

R2_SCORE: 12.3%

Equation: $803677.5 - 398.48\mathrm{X}$

Quadratic:

R2_SCORE: 23.3%

Equation: $5.9 * 10^8 - 5.9 * 10^5\mathrm{X} - 1.48 * 10^2 X^2$

Cubic:

R2_SCORE: 24.25%

Equation: $1.44 * 10^{11} - 2.18 * 10^8\mathrm{X} + 07 + 1.1 * 10^5 X^2 - 18.27X^3$

*With these accuracies we can conclude that the **Quadratic model** is the best for the **given** Data.*

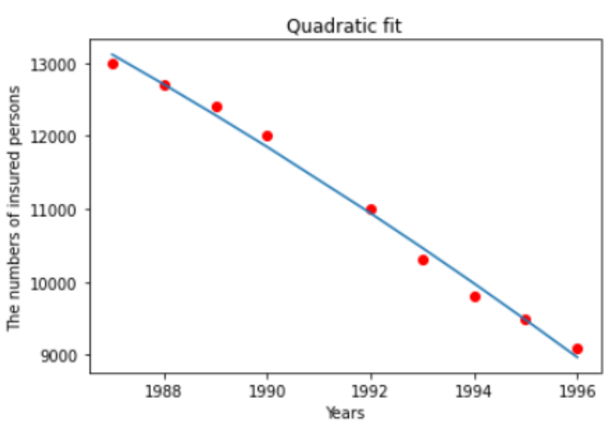
Removing an outlier from the data [**1991**] and recalculating the model:

R2_SCORE: 99.25%

Equation: $-2.47 * 10^7 - 2.5 * 10^4\mathrm{X} - 6.47X^2$

*We can see that the accuracies increased from **23.3%** to **99.25%** by removing the outlier in the data*

c)



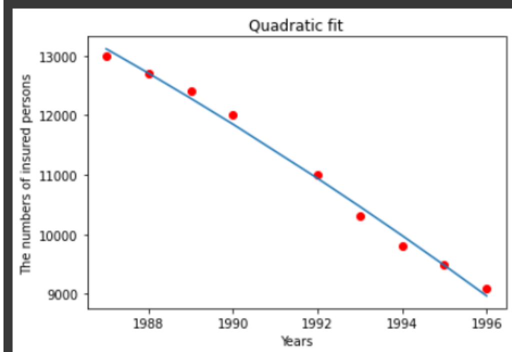
d)Using the Quadratic Model the predicted number of insured persons in 1997 is **8444.38**

Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.preprocessing import PolynomialFeatures

#With the outlier
X = np.array([x for x in range(1987, 1997)]).reshape(-1, 1)
Y = np.array([13000, 12700, 12400, 12000, 1150, 11000, 10300, 9800, 9500, 9100]).reshape(-1, 1)
print("=== Using the given Data ===")
#Linear
model = LinearRegression()
model.fit(X,Y)
Y_hat = model.predict(X)
score=r2_score(Y, Y_hat)
print("Linear Score = ",score.round(decimals=4))

#Quadratic
quad_model = LinearRegression()
quad = PolynomialFeatures(2)
X_quad = quad.fit_transform(X)
quad_model.fit(X_quad, Y)
Y_quad_hat = quad_model.predict(X_quad)
score_quad = r2_score(Y, Y_quad_hat)
print("Quadratic Score = ",score_quad.round(decimals=4))

#Cubic
cubic_model = LinearRegression()
cubic = PolynomialFeatures(3)
X_cubic = cubic.fit_transform(X)
cubic_model.fit(X_cubic, Y)
Y_cubic_hat = cubic_model.predict(X_cubic)
score_cubic = r2_score(Y, Y_cubic_hat)
print("Cubic Score = ",score_cubic.round(decimals=4))
print()


#Removing the outlier
X = np.delete(X, 4).reshape(-1, 1)
Y = np.delete(Y, 4).reshape(-1, 1)
print("=== After Removing the outlier 1991 ===")

#Quadratic
quad_model = LinearRegression()
quad = PolynomialFeatures(2)
X_quad = quad.fit_transform(X)
quad_model.fit(X_quad, Y)
Y_quad_hat = quad_model.predict(X_quad)
score_quad = r2_score(Y, Y_quad_hat)
print("Quadratic Score = ",score_quad.round(decimals=4))

print("Paramters of the best selected Model (Quadratic)
are",quad_model.intercept_[0],quad_model.coef_[0][1],quad_model.coef_[0][2])
print("Average number of insured persons in 1997
=",quad_model.predict(quad.fit_transform([[1997]]))[0][0].round(decimals=2))
print()

plt.scatter(X, Y,color='red')
plt.plot(X, Y_quad_hat)
plt.ylabel("The numbers of insured persons")
plt.xlabel("Years")
plt.title('Quadratic fit')
plt.show()
```

OUTPUT:

```
=== Using the given Data ===
Linear Score =  0.123
Quadratic Score =  0.2329
Cubic Score =  0.2425

=== After Removing the outlier 1991 ===
Quadratic Score =  0.9925
Paramters of the best selected Model (Quadratic) are -24751443.32786534 25328.92755131522 -6.4749053016457765
Average number of insured persons in 1997 = 8444.38
```



Hand Analysis of the Linear Model using Excel:

X	Y	X^2	Y^2	XY
1987	13000	3948169	169000000	25831000
1988	12700	3952144	161290000	25247600
1989	12400	3956121	153760000	24663600
1990	12000	3960100	144000000	23880000
1991	1150	3964081	1322500	2289650
1992	11000	3968064	121000000	21912000
1993	10300	3972049	106090000	20527900
1994	9800	3976036	96040000	19541200
1995	9500	3980025	90250000	18952500
1996	9100	3984016	82810000	18163600
19915	100950	39660805	1125562500	201009050
Beta1	**-398.4848485**			
Beta0	**803677.5758**			

Problem 2.

a, b)

Linear:

R2_SCORE: 89.8%

Equation: $-2090480 + 1060\text{X}$

Quadratic:

R2_SCORE: 99.87%

Equation: $1.19 \; * 10^{9} - 1.19 \; * 10^{6}\text{X} + 300X^{2}$

Cubic:

R2_SCORE: 99.94%

Equation: $-1.95 * 10^{11} + 2.944 * 10^{8}\text{X} - 1.48 * 10^{5}X^{2} + 24.8X^{3}$

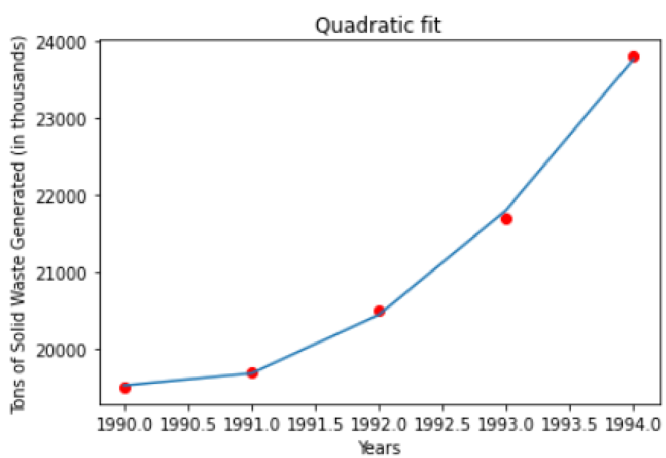
4th Power:

R2_SCORE: 99.94%

Equation: $-4.88 * 10^{10} + 2.7 * 10^{12}\text{X} + 7.4 * 10^{4}X^{2} - 50X^{3} + 9.377 * 10^{-3}X^{4}$

With these accuracies we can conclude that the ***Quadratic model*** *is the best for the* ***given*** *Data.*

c)



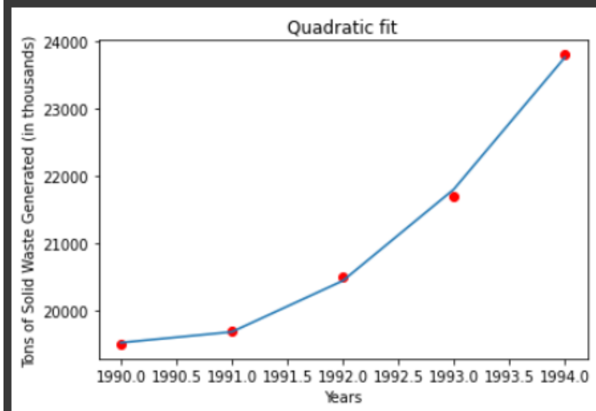
d)Using the Quadratic Model the predicted average tons of waste in 1996 is **29480**

Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.preprocessing import PolynomialFeatures

X = np.array([x for x in range(1990, 1995)]).reshape(-1, 1)
Y = np.array([19500, 19700, 20500, 21700, 23800]).reshape(-1, 1)

#Linear
model = LinearRegression()
model.fit(X,Y)
Y_hat = model.predict(X)
score=r2_score(Y, Y_hat)
print("Linear Score = ",score.round(decimals=4))

#Quadratic
quad_model = LinearRegression()
quad = PolynomialFeatures(2)
X_quad = quad.fit_transform(X)
quad_model.fit(X_quad, Y)
Y_quad_hat = quad_model.predict(X_quad)
score_quad = r2_score(Y, Y_quad_hat)
print("Quadratic Score = ",score_quad.round(decimals=4))

#Cubic
cubic_model = LinearRegression()
cubic = PolynomialFeatures(3)
X_cubic = cubic.fit_transform(X)
cubic_model.fit(X_cubic, Y)
Y_cubic_hat = cubic_model.predict(X_cubic)
score_cubic = r2_score(Y, Y_cubic_hat)
print("Cubic Score = ",score_cubic.round(decimals=4))

#Power
Power_model = LinearRegression()
Power = PolynomialFeatures(4)
X_Power = Power.fit_transform(X)
Power_model.fit(X_Power, Y)
Y_power_hat = Power_model.predict(X_Power)
score_Power = r2_score(Y, Y_power_hat)
print("4th power Score = ",score_Power.round(decimals=4))

#Prediction
print("Paramters of the best selected Model (Quadratic)
are",quad_model.intercept_[0],quad_model.coef_[0][1],quad_model.coef_[0][2])
print("Predicted average tons of waste in 1996
=",quad_model.predict(quad.fit_transform([[1996]]))[0][0].round(decimals=2))
print()
plt.scatter(X, Y,color='red')
plt.plot(X, Y_quad_hat)
plt.ylabel("Tons of Solid Waste Generated (in thousands)")
plt.xlabel("Years")
plt.title('Quadratic fit')
plt.show()
```

OUTPUT:

```
Linear Score =  0.898
Quadratic Score =  0.9987
Cubic Score =  0.9994
4th power Score =  0.9994
Paramters of the best selected Model (Quadratic) are 1188328120.998879 -1194140.001002348 300.0000002514571
Predicted average tons of waste in 1996 = 29480.0
```



Hand Analysis of the Linear Model using Excel:

X	Y	X^2	Y^2	XY
1990	19,500	3960100	380250000	38805000
1991	19,700	3964081	388090000	39222700
1992	20,500	3968064	420250000	40836000
1993	21,700	3972049	470890000	43248100
1994	23,800	3976036	566440000	47457200
9960	105,200	19840330	2225920000	209569000
Beta1	1060			
Beta0	**-2,090,480**			

Problem 3.

a, b)

Linear:

R2_SCORE: 5.1%

Equation: $1.5 - 0.264\text{X}$

Quadratic:

R2_SCORE: 99.93%

Equation: $1.044 + 3.758\text{X} - 4.67X^2$

Cubic:

R2_SCORE: 99.97%

Equation: $1 + 3.9\text{X} - 5.22X^2 + 0.424X^3$

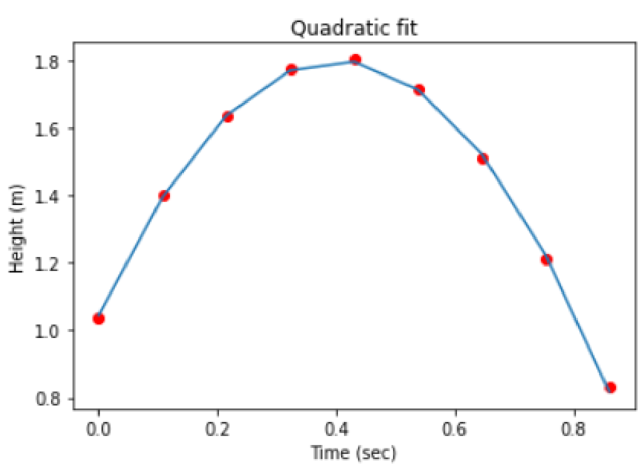
4th Power:

R2_SCORE: 99.99%

Equation: $1 + 3.76\text{X} - 4.2X^2 - 1.5X^3 + 1.12X^4$

*With these accuracies we can conclude that the **Quadratic model** is the best for the **given** Data.*

c)



d) The maximum height of the ball (in meters) is **1.8**
e) The height of the ball is at least 1.5 meters from **0.149** to **0.656**

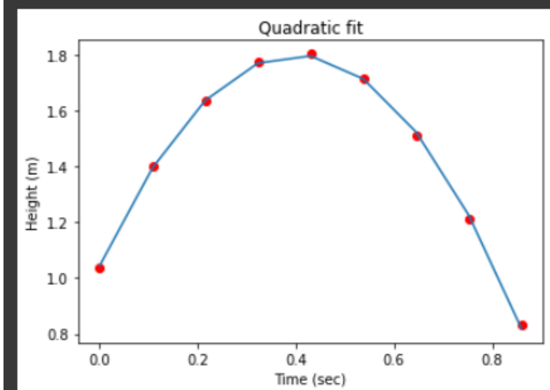
Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.preprocessing import PolynomialFeatures

X = np.array([0.0000, 0.1080, 0.2150, 0.3225, 0.4300, 0.5375, 0.6450, 0.7525, 0.8600]).reshape(-1, 1)
Y = np.array([1.03754,1.40205,1.63806,1.77412,1.80392,1.71522,1.50942,1.21410,0.83173]).reshape(-1, 1)

#Linear
model = LinearRegression()
model.fit(X,Y)
Y_hat = model.predict(X)
score=r2_score(Y, Y_hat)
print("Linear Score = ",score.round(decimals=4))

#Quadratic
quad_model = LinearRegression()
quad = PolynomialFeatures(2)
X_quad = quad.fit_transform(X)
quad_model.fit(X_quad, Y)
Y_quad_hat = quad_model.predict(X_quad)
score_quad = r2_score(Y, Y_quad_hat)
print("Quadratic Score = ",score_quad.round(decimals=4))

#Cubic
cubic_model = LinearRegression()
cubic = PolynomialFeatures(3)
X_cubic = cubic.fit_transform(X)
cubic_model.fit(X_cubic, Y)
Y_cubic_hat = cubic_model.predict(X_cubic)
score_cubic = r2_score(Y, Y_cubic_hat)
print("Cubic Score = ",score_cubic.round(decimals=4))

#Power
Power_model = LinearRegression()
Power = PolynomialFeatures(4)
X_Power = Power.fit_transform(X)
Power_model.fit(X_Power, Y)
Y_power_hat = Power_model.predict(X_Power)
score_Power = r2_score(Y, Y_power_hat)
print("4th power Score = ",score_Power.round(decimals=4))
#Prediction
print("Paramters of the best selected Model (Quadratic)
are",quad_model.intercept_[0],quad_model.coef_[0][1],quad_model.coef_[0][2])
print()
plt.scatter(X, Y,color='red')
plt.plot(X, Y_quad_hat)
plt.ylabel("Height (m)")
plt.xlabel("Time (sec)")
plt.title('Quadratic fit')
plt.show()

lop = np.linspace(0,1,1001)
mx = 0
g=[]
cnt=0
L1=-1
L2=-1
for i in lop:
  g.append(quad_model.intercept_[0]+quad_model.coef_[0][1]*i+quad_model.coef_[0][2]*i*i)
  mx = max(mx,quad_model.intercept_[0]+quad_model.coef_[0][1]*i+quad_model.coef_[0][2]*i*i)
  if quad_model.intercept_[0]+quad_model.coef_[0][1]*i+quad_model.coef_[0][2]*i*i >=1.5 and cnt==0:
    L1=i
    cnt=1
```

```python
    elif quad_model.intercept_[0]+quad_model.coef_[0][1]*i+quad_model.coef_[0][2]*i*i<=1.5 and cnt==1:
        L2=i
        cnt=2
print()
print("The maximum height of the ball (in meters) is",mx.round(decimals=2))
print("The height of the ball is at least 1.5 meters from",L1,"to",L2)
```

OUTPUT:

```
Linear Score =  0.051
Quadratic Score =  0.9993
Cubic Score =  0.9997
4th power Score =  0.9999
Paramters of the best selected Model (Quadratic) are 1.0449726483249402 3.758277636483571 -4.6764031620667925
```




```
The maximum height of the ball (in meters) is 1.8
The height of the ball is at least 1.5 meters from 0.149 to 0.656
```

Hand Analysis of the Linear Model using Excel:

X	Y	X^2	Y^2	XY
0	1.03754	0	0	0
0.108	1.40205	0.011664	0.011664	0.1514214
0.215	1.63806	0.046225	0.046225	0.3521829
0.3225	1.77412	0.10400625	0.10400625	0.5721537
0.43	1.80392	0.1849	0.1849	0.7756856
0.5375	1.71522	0.28890625	0.28890625	0.92193075
0.645	1.50942	0.416025	0.416025	0.9735759
0.7525	1.2141	0.56625625	0.56625625	0.91361025
0.86	0.83173	0.7396	0.7396	0.7152878
3.8705	12.92616	2.35758275	2.35758275	5.3758483
Beta1	**-0.2642203315**			
Beta0	**1.613910599**			

Problem 4.

a, b)

Linear:

R2_SCORE: 78.13%

Equation: $464.2 - 2.44X_1 - 5.53X_2$

Quadratic:

R2_SCORE: 97.6%

Equation: $684.17 - 43.427X_1 - 6.3X_2 + 0.717X_1X_2 + 0.265{X_1}^2 - 0.012{X_2}^2$

*With these accuracies we can conclude that the **Quadratic model** is the best for the **given** Data.*

c)

Prediction of the needed oil if the temperature is 10 Fahrenheit and the insulation is 5 attic insulations inches using the Linear model = **396.63**

Prediction of the needed oil if the temperature is 10 Fahrenheit and the insulation is 5 attic insulations inches using the Quadratic model = **433.75**

d) **Increase** the isolation as the temperature decrease which will decrease the oil consumption

e) After removing the outlier, the accuracies changed as the following:

Linear:

R2_SCORE: 96.03%

Quadratic:

R2_SCORE: 97.98%

Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.preprocessing import PolynomialFeatures

X1 = np.array([4,4,10,6,7,40,6,10,10,4,10,6,4,4,10]).reshape(-1, 1)
X2 = np.array([40,27,40,73,65,35,10,9,24,65,66,41,22,40,60]).reshape(-1, 1)
Y  = np.array([275,360,160,40,90,230,370,300,230,120,30,200,440,323,50]).reshape(-1, 1)
X  = np.dstack((X1,X2)).reshape(15,2)

#With the outlier
print("=== Using the given Data ===")
#Linear
model = LinearRegression()
model.fit(X,Y)
Y_hat = model.predict(X)
score=r2_score(Y, Y_hat)
print("Linear Score = ",score.round(decimals=4))

#Quadratic
quad_model = LinearRegression()
quad = PolynomialFeatures(2)
X_quad = quad.fit_transform(X)
quad_model.fit(X_quad, Y)
Y_quad_hat = quad_model.predict(X_quad)
score_quad = r2_score(Y, Y_quad_hat)
print("Quadratic Score = ",score_quad.round(decimals=4))

print("Paramters of the Linear model are",model.intercept_[0],model.coef_[0][0],model.coef_[0][1])
print("Paramters of the Quadratic model are",quad_model.intercept_[0],

quad_model.coef_[0][1],quad_model.coef_[0][2],quad_model.coef_[0][3],quad_model.coef_[0][4],quad_model.
coef_[0][5])
print("Prediction of the needed oil if the temperature is 10 Fahrenheit and the insulation is 5 attic
insulations inches using the Linear model =",
      model.predict([[5,10]])[0][0].round(decimals=2))
print("Prediction of the needed oil if the temperature is 10 Fahrenheit and the insulation is 5 attic
insulations inches using the Quadratic model =",
      quad_model.predict(quad.fit_transform([[5,10]]))[0][0].round(decimals=2))

print()

#Removing the outlier
print("=== After Removing the outlier 230 ===")
X1 = np.array([4,4,10,6,7,6,10,10,4,10,6,4,4,10]).reshape(-1, 1)
X2 = np.array([40,27,40,73,65,10,9,24,65,66,41,22,40,60]).reshape(-1, 1)
Y  = np.array([275,360,160,40,90,370,300,230,120,30,200,440,323,50]).reshape(-1, 1)
X  = np.dstack((X1,X2)).reshape(14,2)
#Linear
model = LinearRegression()
model.fit(X,Y)
Y_hat = model.predict(X)
score=r2_score(Y, Y_hat)
print("Linear Score = ",score.round(decimals=4))

#Quadratic
quad_model = LinearRegression()
quad = PolynomialFeatures(2)
X_quad = quad.fit_transform(X)
quad_model.fit(X_quad, Y)
Y_quad_hat = quad_model.predict(X_quad)
score_quad = r2_score(Y, Y_quad_hat)
print("Quadratic Score = ",score_quad.round(decimals=4))
```

OUTPUT:

```
=== Using the given Data ===
Linear Score =  0.7813
Quadratic Score =  0.976
Paramters of the Linear model are 464.19406247601114 -2.4426022236096383 -5.535104760053267
Paramters of the Quadratic model are 684.1703286034714 -43.42764558800189 -6.328428174285615 0.7176277689506456 0.2652028380318576 -0.012005090994115 62
Prediction of the needed oil if the temperature is 10 Fahrenheit and the insulation is 5 attic insulations inches using the Linear model = 396.63
Prediction of the needed oil if the temperature is 10 Fahrenheit and the insulation is 5 attic insulations inches using the Quadratic model = 433.75

=== After Removing the outlier 230 ===
Linear Score =  0.9603
Quadratic Score =  0.9798
```

Problem 5.

a, b)

Linear:

R2_SCORE: 92.72%

Equation: $-15.36 + 1.37X$

Quadratic:

R2_SCORE:93.22%

Equation: $-10.6 + \mathrm{X} + 0.003X^2$

Logistic:

R2_SCORE:99.97%

Equation:$\frac{100}{1+e^{-(-6.151+0.126\mathrm{X})}}$

*With these accuracies we can conclude that the **Logistic model** is the best for the **given** Data.*

c)

Prediction of % Who Bought using the Linear model = 122.2 %
Prediction of % Who Bought using the Quadratic model = 130.93 %
Prediction of % Who Bought using the Logistic model = 99.86%

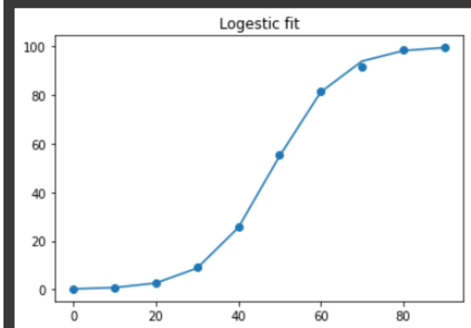
d) Don’t increase the number of ads more than 100

Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import r2_score
from sklearn.preprocessing import PolynomialFeatures
X = np.array([x for x in range(0,100,10)]).reshape(-1, 1)
Y  = np.array([0.2,0.8,2.6,9.1,26,55.6,81.3,91.8,98.5,99.5]).reshape(-1, 1)
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
#Linear
model = LinearRegression()
model.fit(X,Y)
Y_hat = model.predict(X)
score=r2_score(Y, Y_hat)
print("Linear Score = ",score.round(decimals=4))

#Quadratic
quad_model = LinearRegression()
quad = PolynomialFeatures(2)
X_quad = quad.fit_transform(X)
quad_model.fit(X_quad, Y)
Y_quad_hat = quad_model.predict(X_quad)
score_quad = r2_score(Y, Y_quad_hat)
print("Quadratic Score = ",score_quad.round(decimals=4))

#Logestic
def predict_log(model,input):
    return 100*1/(1+np.exp(-model.predict([[input]])[0]))[0].round(decimals=4)

X = np.array([x for x in range(0,100,10)]).reshape(-1, 1)
Y_original  = np.array([0.2,0.8,2.6,9.1,26,55.6,81.3,91.8,98.5,99.5]).reshape(-1, 1)
Y = np.log((Y_original/100)/(1-Y_original/100))
logestic = LinearRegression()
logestic.fit(X,Y)
Y_log_hat=logestic.predict(X)
Y_log_hat=100*1/(1+np.exp(-Y_log_hat))
score_log = r2_score(Y_original, Y_log_hat)

print("Logestic Score = ",score_log.round(decimals=4))
print("Paramters of the Linear model are",model.intercept_[0],model.coef_[0][0])
print("Paramters of the Quadratic model
are",quad_model.intercept_[0],quad_model.coef_[0][1],quad_model.coef_[0][2])
print("Paramters of the Logestic model are",logestic.intercept_[0],logestic.coef_[0][0])

print("% Who Bought using the Linear model =",model.predict([[100]])[0][0].round(decimals=4))
print("% Who Bought using the Quadratic model
=",quad_model.predict(quad.fit_transform([[100]]))[0][0].round(decimals=4))
print("% Who Bought using the Logestic model =",predict_log(logestic,100))
print()
plt.scatter(X, np.array([0.2,0.8,2.6,9.1,26,55.6,81.3,91.8,98.5,99.5]).reshape(-1, 1))
plt.plot(X, Y_log_hat)
plt.title("Logestic fit")
plt.show()
```

OUTPUT:

```
Linear Score =  0.9272
Quadratic Score =  0.9322
Logestic Score =  0.9997
Paramters of the Linear model are -15.363636363636367 1.3756363636363638
Paramters of the Quadratic model are -10.600000000000527 1.0183636363636363663 0.003969696969696734
Paramters of the Logestic model are -6.151540059087443 0.12688818811936822
% Who Bought using the Linear model = 122.2
% Who Bought using the Quadratic model = 130.9333
% Who Bought using the Logestic model = 99.86019572598362
```



Hand Analysis of the Linear Model using Excel:

X	Y	X^2	Y^2	XY
0	0.2	0	0.04	0
10	0.8	100	0.64	8
20	2.6	400	6.76	52
30	9.1	900	82.81	273
40	26	1600	676	1040
50	55.6	2500	3091.36	2780
60	81.3	3600	6609.69	4878
70	91.8	4900	8427.24	6426
80	98.5	6400	9702.25	7880
90	99.5	8100	9900.25	8955
450	465.4	28500	38497.04	32292
Beta1	**1.375636364**			
Beta0	**-17.61**			

Hand Analysis of the Logistic Model using Excel:

X	Y	Y_log	X^2	Y^2	XY
0	0.2	-6.212606096	0	38.5964745	0
10	0.8	-4.820281566	100	23.23511437	-48.20281566
20	2.6	-3.623314766	400	13.12840989	-72.46629531
30	9.1	-2.301485588	900	5.29683591	-69.04456763
40	26	-1.045968555	1600	1.094050218	-41.83874221
50	55.6	0.2249437318	2500	0.05059968248	11.24718659
60	81.3	1.469622493	3600	2.159790271	88.17734956
70	91.8	2.415478143	4900	5.834534661	169.08347
80	98.5	4.18459144	6400	17.51080552	334.7673152
90	99.5	5.293304825	8100	28.01907597	476.3974342
450	465.4	-4.415715937	28500	134.925691	848.1203348
Beta1	**0.1268881881**				
Beta0	**-6.151540059**				

The following are Google Colab link for the codes and the excel file used for hand analysis

https://colab.research.google.com/drive/1iU7TCocQT3FuDDWZWhhw2BbS3wrGf7jE?usp=sharing

https://docs.google.com/spreadsheets/d/1IR7qr68aGdVKz7Rge5suKOArEjbcJtN3KBhKUMZH4_g