Optimisation in ML

Lab Assignment 1

```
-Ayush Abrol
```

Question 3

```
R = 2000
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
data = pd.read_excel('3 columns.xls')
data
```

	area	bedrooms	price
0	3050	3	3920000
1	3100	2	2135000
2	4320	3	4690000
3	3060	3	3465000
4	2575	2	3290000
...	...	...	...
95	6600	4	9100000
96	2145	3	4200000
97	5200	4	2852500
98	4410	2	4200000
99	8400	3	5250000

100 rows × 3 columns

```
# We need to find the best fitting plane z = a1x + a2y + a3
# We can use the least squares method to find the best fitting plane
```

```
B = data.values
B
```

```
            [    3240,        2,  2450000],
```

```
       [   5400,       4, 2870000],
       [   6725,       3, 7580000],
       [   5960,       3, 8190000],
       [   3300,       3, 5530000],
       [   3500,       4, 4340000],
       [   4079,       3, 4200000],
       [   4600,       3, 8890000],
       [   4000,       3, 6790000],
       [   7482,       3, 8043000],
       [   2000,       2, 2660000],
       [   3150,       3, 2940000],
       [   2135,       3, 3500000],
       [   4120,       2, 4900000],
       [   3850,       3, 3115000],
       [   2145,       4, 3920000],
       [   4500,       3, 4007500],
       [   4080,       2, 3850000],
       [   6100,       3, 5110000],
       [   4080,       3, 4165000],
       [   6710,       3, 5390000],
       [   5010,       3, 4620000],
       [   3986,       2, 3150000],
       [   7320,       4, 5950000],
       [   8800,       3, 8575000],
       [   3480,       3, 2940000],
       [  10240,       2, 4760000],
       [   6360,       4, 7035000],
       [   3480,       2, 3745000],
       [   4880,       4, 8295000],
       [   3990,       3, 3500000],
       [   8250,       3, 3773000],
       [   8400,       4, 4550000],
       [   6235,       3, 7350000],
       [   4160,       3, 4830000],
       [   5885,       2, 4480000],
       [   5320,       3, 4550000],
       [   9800,       4, 5250000],
       [   3460,       3, 4025000],
       [   6100,       3, 5530000],
       [   4350,       3, 2835000],
       [   3880,       3, 4620000],
       [   5495,       3, 3129000],
       [   3650,       3, 3500000],
       [   6540,       4, 8540000],
       [   6360,       2, 4270000],
       [   3450,       1, 3150000],
       [   9000,       3, 6300000],
       [  11460,       3, 5873000],
       [   3792,       4, 3290000],
       [   3640,       3, 4200000],
       [   3450,       3, 3150000],
       [   3450,       2, 1820000],
       [   6600,       4, 9100000],
       [   2145,       3, 4200000],
       [   5200,       4, 2852500],
       [   4410,       2, 4200000],
       [   8400,       3, 5250000]], dtype=int64)
```

```
x = B[:,0]
```

```
y = B[:,1]
z = B[:,2]


ones = np.ones((len(x),1), dtype=float)
A = np.column_stack((x,y,ones))
A
```

```
       [3.240e+03, 2.000e+00, 1.000e+00],
       [5.400e+03, 4.000e+00, 1.000e+00],
       [6.725e+03, 3.000e+00, 1.000e+00],
       [5.960e+03, 3.000e+00, 1.000e+00],
       [3.300e+03, 3.000e+00, 1.000e+00],
       [3.500e+03, 4.000e+00, 1.000e+00],
       [4.079e+03, 3.000e+00, 1.000e+00],
       [4.600e+03, 3.000e+00, 1.000e+00],
       [4.000e+03, 3.000e+00, 1.000e+00],
       [7.482e+03, 3.000e+00, 1.000e+00],
       [2.000e+03, 2.000e+00, 1.000e+00],
       [3.150e+03, 3.000e+00, 1.000e+00],
       [2.135e+03, 3.000e+00, 1.000e+00],
       [4.120e+03, 2.000e+00, 1.000e+00],
       [3.850e+03, 3.000e+00, 1.000e+00],
       [2.145e+03, 4.000e+00, 1.000e+00],
       [4.500e+03, 3.000e+00, 1.000e+00],
       [4.080e+03, 2.000e+00, 1.000e+00],
       [6.100e+03, 3.000e+00, 1.000e+00],
       [4.080e+03, 3.000e+00, 1.000e+00],
       [6.710e+03, 3.000e+00, 1.000e+00],
       [5.010e+03, 3.000e+00, 1.000e+00],
       [3.986e+03, 2.000e+00, 1.000e+00],
       [7.320e+03, 4.000e+00, 1.000e+00],
       [8.800e+03, 3.000e+00, 1.000e+00],
       [3.480e+03, 3.000e+00, 1.000e+00],
       [1.024e+04, 2.000e+00, 1.000e+00],
       [6.360e+03, 4.000e+00, 1.000e+00],
       [3.480e+03, 2.000e+00, 1.000e+00],
       [4.880e+03, 4.000e+00, 1.000e+00],
       [3.990e+03, 3.000e+00, 1.000e+00],
       [8.250e+03, 3.000e+00, 1.000e+00],
       [8.400e+03, 4.000e+00, 1.000e+00],
       [6.235e+03, 3.000e+00, 1.000e+00],
       [4.160e+03, 3.000e+00, 1.000e+00],
       [5.885e+03, 2.000e+00, 1.000e+00],
       [5.320e+03, 3.000e+00, 1.000e+00],
       [9.800e+03, 4.000e+00, 1.000e+00],
       [3.460e+03, 3.000e+00, 1.000e+00],
       [6.100e+03, 3.000e+00, 1.000e+00],
       [4.350e+03, 3.000e+00, 1.000e+00],
       [3.880e+03, 3.000e+00, 1.000e+00],
       [5.495e+03, 3.000e+00, 1.000e+00],
       [3.650e+03, 3.000e+00, 1.000e+00],
       [6.540e+03, 4.000e+00, 1.000e+00],
       [6.360e+03, 2.000e+00, 1.000e+00],
       [3.450e+03, 1.000e+00, 1.000e+00],
       [9.000e+03, 3.000e+00, 1.000e+00],
       [1.146e+04, 3.000e+00, 1.000e+00],
       [3.792e+03, 4.000e+00, 1.000e+00],
       [3.640e+03, 3.000e+00, 1.000e+00],
       [3.450e+03, 3.000e+00, 1.000e+00],
```

```
       [3.450e+03, 2.000e+00, 1.000e+00],
       [6.600e+03, 4.000e+00, 1.000e+00],
       [2.145e+03, 3.000e+00, 1.000e+00],
       [5.200e+03, 4.000e+00, 1.000e+00],
       [4.410e+03, 2.000e+00, 1.000e+00],
       [8.400e+03, 3.000e+00, 1.000e+00]])
```

```
beta = np.dot(np.linalg.inv(np.dot(A.T,A)),np.dot(A.T,y.T))
print(beta)
```

```
[-4.33680869e-19  1.00000000e+00  0.00000000e+00]
```

```
print("Price of House: ", beta[0]*R + beta[1]*(R+3) + beta[2])
```

```
Price of House:  2002.999999999993
```

Question 4

```
data = pd.read_excel('3 columns.xls')
data
```

	area	bedrooms	price
0	3050	3	3920000
1	3100	2	2135000
2	4320	3	4690000
3	3060	3	3465000
4	2575	2	3290000
...	...	...	...
95	6600	4	9100000
96	2145	3	4200000
97	5200	4	2852500
98	4410	2	4200000
99	8400	3	5250000

100 rows × 3 columns

```
B = data.values
B
```

```
       [   3240,       2,  2450000],
       [   5400,       4,  2870000],
       [   6725,       3,  7580000],
       [   5960,       3,  8190000],
       [   3300,       3,  5530000],
       [   3500,       4,  4340000],
```

```
       [   4079,        3,  4200000],
       [   4600,        3,  8890000],
       [   4000,        3,  6790000],
       [   7482,        3,  8043000],
       [   2000,        2,  2660000],
       [   3150,        3,  2940000],
       [   2135,        3,  3500000],
       [   4120,        2,  4900000],
       [   3850,        3,  3115000],
       [   2145,        4,  3920000],
       [   4500,        3,  4007500],
       [   4080,        2,  3850000],
       [   6100,        3,  5110000],
       [   4080,        3,  4165000],
       [   6710,        3,  5390000],
       [   5010,        3,  4620000],
       [   3986,        2,  3150000],
       [   7320,        4,  5950000],
       [   8800,        3,  8575000],
       [   3480,        3,  2940000],
       [  10240,        2,  4760000],
       [   6360,        4,  7035000],
       [   3480,        2,  3745000],
       [   4880,        4,  8295000],
       [   3990,        3,  3500000],
       [   8250,        3,  3773000],
       [   8400,        4,  4550000],
       [   6235,        3,  7350000],
       [   4160,        3,  4830000],
       [   5885,        2,  4480000],
       [   5320,        3,  4550000],
       [   9800,        4,  5250000],
       [   3460,        3,  4025000],
       [   6100,        3,  5530000],
       [   4350,        3,  2835000],
       [   3880,        3,  4620000],
       [   5495,        3,  3129000],
       [   3650,        3,  3500000],
       [   6540,        4,  8540000],
       [   6360,        2,  4270000],
       [   3450,        1,  3150000],
       [   9000,        3,  6300000],
       [  11460,        3,  5873000],
       [   3792,        4,  3290000],
       [   3640,        3,  4200000],
       [   3450,        3,  3150000],
       [   3450,        2,  1820000],
       [   6600,        4,  9100000],
       [   2145,        3,  4200000],
       [   5200,        4,  2852500],
       [   4410,        2,  4200000],
       [   8400,        3,  5250000]], dtype=int64)
```

```
x,y,z=B[:,0],B[:,1],B[:,2]
```

```
ones=np.ones((len(x),1), dtype=float)
A=np.column_stack((np.power(x,2),np.multiply(x,y),np.power(y,2),x,y,ones))
```

A

```
[2.9160000e+07, 2.1600000e+04, 1.6000000e+01, 5.4000000e+03,
 4.0000000e+00, 1.0000000e+00],
[4.5225625e+07, 2.0175000e+04, 9.0000000e+00, 6.7250000e+03,
 3.0000000e+00, 1.0000000e+00],
[3.5521600e+07, 1.7880000e+04, 9.0000000e+00, 5.9600000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.0890000e+07, 9.9000000e+03, 9.0000000e+00, 3.3000000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.2250000e+07, 1.4000000e+04, 1.6000000e+01, 3.5000000e+03,
 4.0000000e+00, 1.0000000e+00],
[1.6638241e+07, 1.2237000e+04, 9.0000000e+00, 4.0790000e+03,
 3.0000000e+00, 1.0000000e+00],
[2.1160000e+07, 1.3800000e+04, 9.0000000e+00, 4.6000000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.6000000e+07, 1.2000000e+04, 9.0000000e+00, 4.0000000e+03,
 3.0000000e+00, 1.0000000e+00],
[5.5980324e+07, 2.2446000e+04, 9.0000000e+00, 7.4820000e+03,
 3.0000000e+00, 1.0000000e+00],
[4.0000000e+06, 4.0000000e+03, 4.0000000e+00, 2.0000000e+03,
 2.0000000e+00, 1.0000000e+00],
[9.9225000e+06, 9.4500000e+03, 9.0000000e+00, 3.1500000e+03,
 3.0000000e+00, 1.0000000e+00],
[4.5582250e+06, 6.4050000e+03, 9.0000000e+00, 2.1350000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.6974400e+07, 8.2400000e+03, 4.0000000e+00, 4.1200000e+03,
 2.0000000e+00, 1.0000000e+00],
[1.4822500e+07, 1.1550000e+04, 9.0000000e+00, 3.8500000e+03,
 3.0000000e+00, 1.0000000e+00],
[4.6010250e+06, 8.5800000e+03, 1.6000000e+01, 2.1450000e+03,
 4.0000000e+00, 1.0000000e+00],
[2.0250000e+07, 1.3500000e+04, 9.0000000e+00, 4.5000000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.6646400e+07, 8.1600000e+03, 4.0000000e+00, 4.0800000e+03,
 2.0000000e+00, 1.0000000e+00],
[3.7210000e+07, 1.8300000e+04, 9.0000000e+00, 6.1000000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.6646400e+07, 1.2240000e+04, 9.0000000e+00, 4.0800000e+03,
 3.0000000e+00, 1.0000000e+00],
[4.5024100e+07, 2.0130000e+04, 9.0000000e+00, 6.7100000e+03,
 3.0000000e+00, 1.0000000e+00],
[2.5100100e+07, 1.5030000e+04, 9.0000000e+00, 5.0100000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.5888196e+07, 7.9720000e+03, 4.0000000e+00, 3.9860000e+03,
 2.0000000e+00, 1.0000000e+00],
[5.3582400e+07, 2.9280000e+04, 1.6000000e+01, 7.3200000e+03,
 4.0000000e+00, 1.0000000e+00],
[7.7440000e+07, 2.6400000e+04, 9.0000000e+00, 8.8000000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.2110400e+07, 1.0440000e+04, 9.0000000e+00, 3.4800000e+03,
 3.0000000e+00, 1.0000000e+00],
[1.0485760e+08, 2.0480000e+04, 4.0000000e+00, 1.0240000e+04,
 2.0000000e+00, 1.0000000e+00],
[4.0449600e+07, 2.5440000e+04, 1.6000000e+01, 6.3600000e+03,
 4.0000000e+00, 1.0000000e+00],
[1.2110400e+07, 6.9600000e+03, 4.0000000e+00, 3.4800000e+03,
 2.0000000e+00, 1.0000000e+00],
[2.3814400e+07, 1.9520000e+04, 1.6000000e+01, 4.8800000e+03,
 4.0000000e+00, 1.0000000e+00],
```

```
beta=np.dot(np.linalg.inv(np.dot(A.T,A)),np.dot(A.T,y.T))
beta
```

```
array([ 5.50571416e-20, -2.49800181e-16,  1.98951966e-13,  9.71445147e-17,
        1.00000000e+00, -4.54747351e-13])
```

```
print("price of house",beta[0]*pow(R,2)+beta[1]*R*(R+3)+beta[2]
      *pow(R+3,2)+beta[3]*R+beta[4]*(R+3)+beta[5])
```

```
price of house 2003.0000007973103
```

Question 7

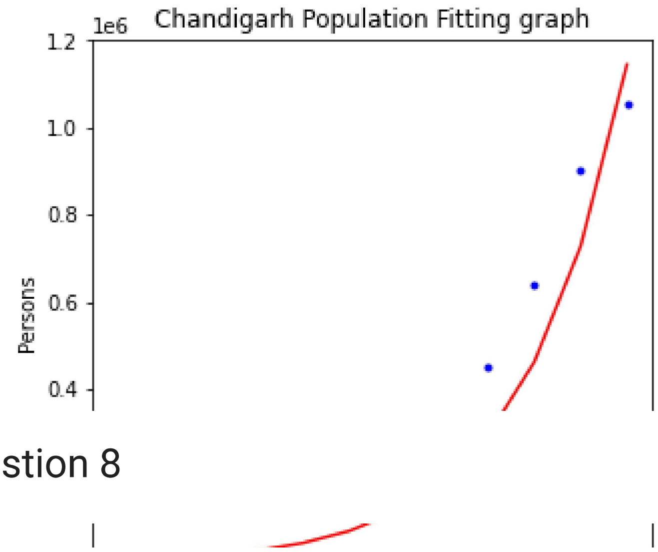
```
data = pd.read_excel('Census data (Chandigarh).xls')
data = data.values
```

```
Year, Persons = data[:,0] ,data[:,1]
#z = log(Persons)
z = np.log(Persons)
ones=np.ones((len(Year),1), dtype=float)
A=np.column_stack((Year, ones))
beta=np.dot(np.linalg.inv(np.dot(A.T,A)),np.dot(A.T,z.T))
```

```
z_2021=beta[0]*2021+beta[1]
Pop_2021 = np.exp(z_2021)
print("Population of Chandigarh in year 2021 using Exponential fittiing will be: ",Pop_202
```

```
Population of Chandigarh in year 2021 using Exponential fittiing will be:  1799523.7
```

```
beta_0=np.exp(beta[1])
beta_1=beta[0]
#Graph
plt.figure(figsize = (5,5))
plt.title("Chandigarh Population Fitting graph")
plt.plot(Year, Persons, 'b.')
plt.plot(Year, beta_0*np.exp(beta_1*Year), 'r')
plt.xlabel('Year')
plt.ylabel('Persons')
plt.show()
```

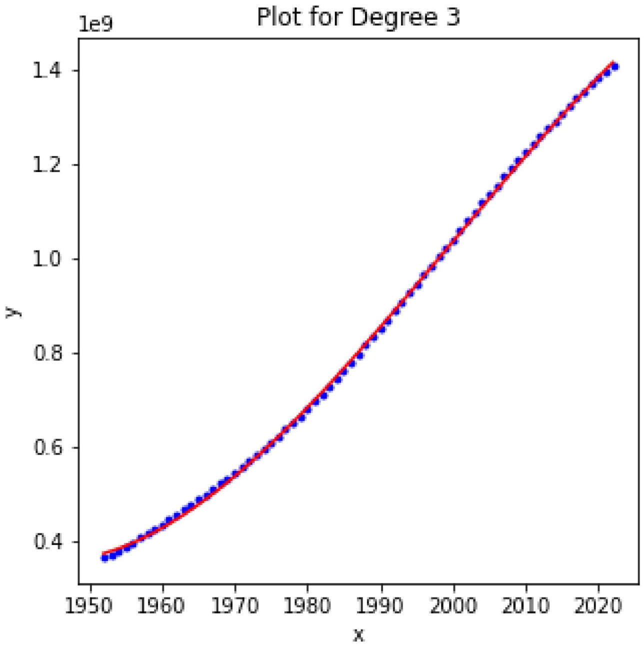


Question 8

```
population = pd.read_excel('population1.xls')
population = population.values


Year, Populations = population[:,0] ,population[:,1]
z=np.ones((len(Year),1), dtype=float)


def polynomial_fit(x,y,p):
  temp = []
  for i in range(p,0,-1):
    temp.append(np.power(x,i))
  temp.append(z)
  temp = tuple(temp)
  A = np.column_stack(temp)
  beta=np.dot(np.linalg.inv(np.dot(A.T,A)),np.dot(A.T,y.T))
  arr = beta[0]*np.power(x,p)
  for i in range(p-1,0,-1):
    arr += (beta[p-i]*np.power(x,i))
  arr += beta[p]
  len_x = len(x)
  res = np.dot(A,beta)-y
  avg_loss = np.dot(res.T,res)/(2*len_x)
  print("Average Loss for Degree ",p," is:",avg_loss)
  #Figure for degree = 3
  if(p==3):
    plt.figure(figsize = (5,5))
    plt.title("Plot for Degree "+str(p))
    plt.plot(x, y, 'b.')
    plt.plot(x, arr, 'r')
    plt.xlabel('x')
    plt.ylabel('y')
    plt.show()
  return avg_loss



min_loss = polynomial_fit(Year,Populations,1)
degree = 1
```

```
for i in range(2,21):
  loss = polynomial_fit(Year,Populations,i)
  if loss<min_loss:
    min_loss = loss
    degree = i

print("Lowest average loss of ", min_loss," is obtained at degree: ", degree)
```

```
Average Loss for Degree  1  is: 523860937459408.3
Average Loss for Degree  2  is: 109243277742810.06
Average Loss for Degree  3  is: 17767906501445.42
```




```
Average Loss for Degree  4  is: 9738430412788124.0
Average Loss for Degree  5  is: 1.0886643273209038e+16
Average Loss for Degree  6  is: 4.7833446584639384e+16
Average Loss for Degree  7  is: 1864605519128191.8
Average Loss for Degree  8  is: 9349668764966870.0
Average Loss for Degree  9  is: 1.1365397575924876e+16
Average Loss for Degree  10  is: 1600056954355572.5
Average Loss for Degree  11  is: 2.2725839884268028e+16
Average Loss for Degree  12  is: 7664308530118347.0
Average Loss for Degree  13  is: 4757888654965746.0
Average Loss for Degree  14  is: 5.432956356516732e+16
Average Loss for Degree  15  is: 1.1452466803916235e+17
Average Loss for Degree  16  is: 1.106633295731981e+16
Average Loss for Degree  17  is: 8.27496792636314e+16
Average Loss for Degree  18  is: 6.422318986556853e+16
Average Loss for Degree  19  is: 1.301322473349929e+16
Average Loss for Degree  20  is: 5081460991010505.0
Lowest average loss of  17767906501445.42  is obtained at degree:   3
```