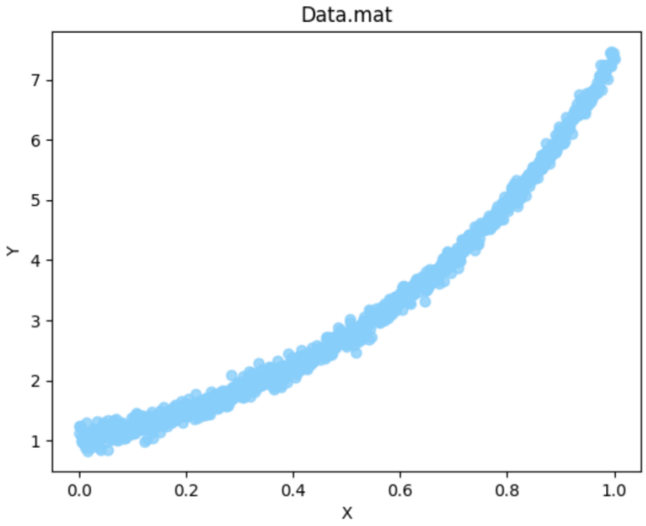
Deep Learning HW1　112753204　陳品伃

Due: 10/20, 2023, 11:59pm

For the following questions, please upload the source code to moodle and explain the results in your report. **If you choose to implement the machine learning models by yourself (no built-in APIs), you will get extra 10% bonus for each question.**

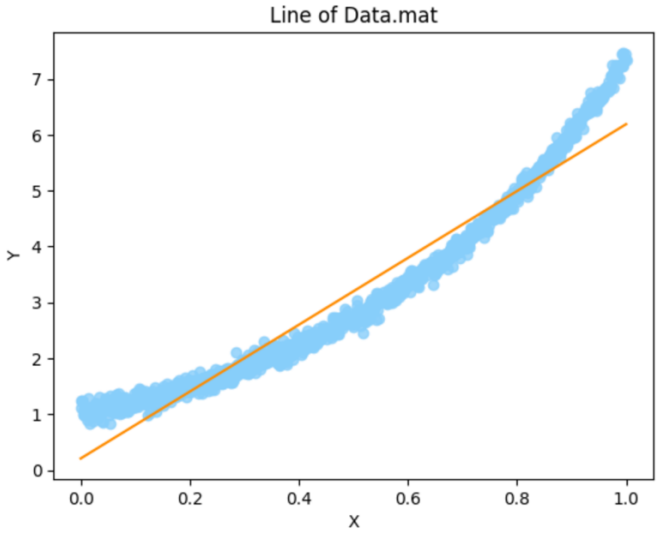
1. Please load 'data.mat' into your Python code, where you will find $x, y \in R^{1001}$. Now do the following procedures.

   1.1. (**5%**) Plot the data using plot function.



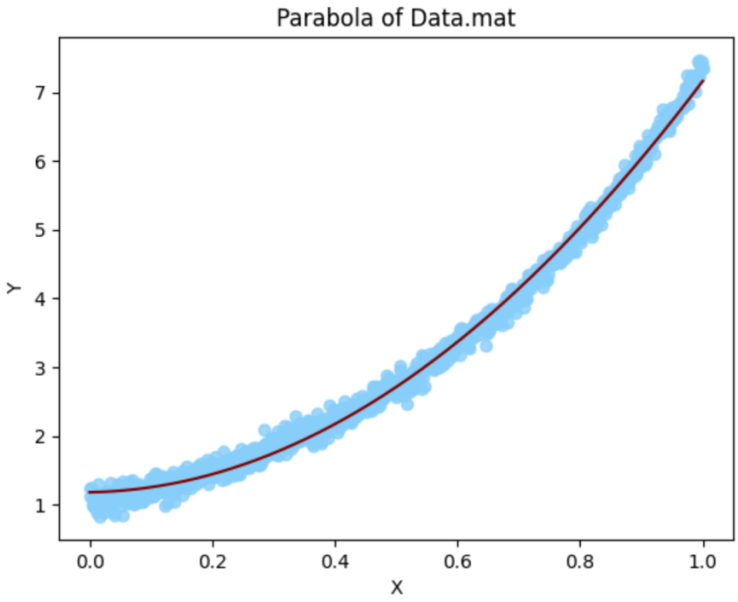
   1.2.(**5%**) Compute the least square line $y = \theta_0 + x\theta_1$ using the given data and overlay the line over the given data.

   Ans：$y = 0.2070272 + 5.98091717x$



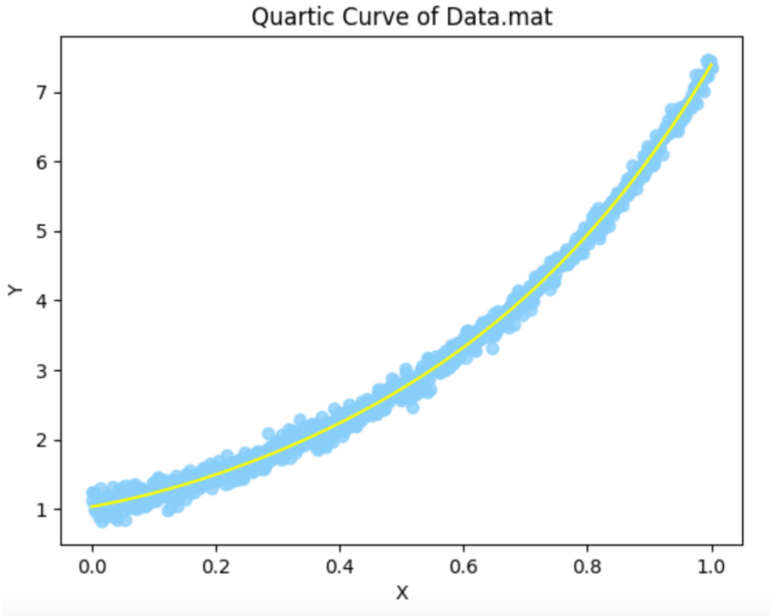
1.3.(**5%**) Compute the least square parabola (i.e. second order polynomial $y = \boldsymbol{\theta_0} + x\boldsymbol{\theta_1} + x^2\boldsymbol{\theta_2}$) to fit the data.

Ans：$y = 1.17894599 + 0.14356709x + 5.83735008x^2$

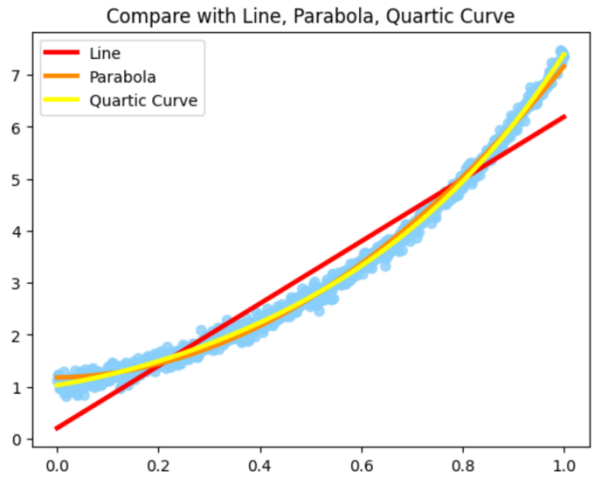


1.4.(**5%**) Compute the least square quartic curve ($y = \boldsymbol{\theta_0} + x\boldsymbol{\theta_1} + x^2\boldsymbol{\theta_2} + x^3\theta_3 + x^4\theta_4$) to fit the data.

Ans：$y = 1.03121867 + 1.59131656x + 3.86161799x^2 + (-1.97292627)x^3 + 2.87810255x^4$



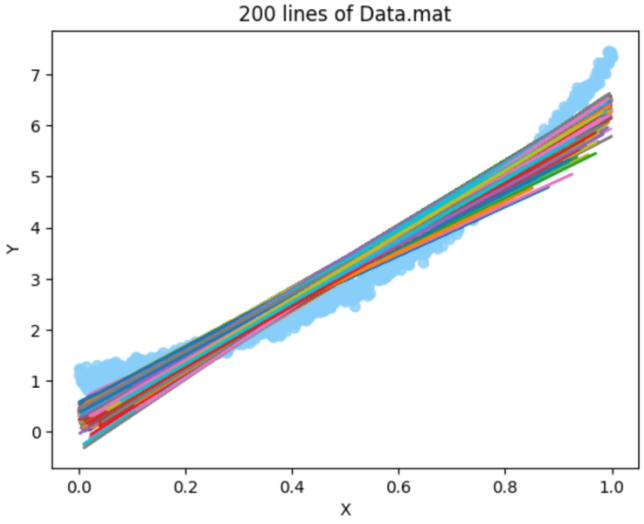
1.5.(**5%**) Explain which formulation (line, parabola, cubic curve) is more suitable for this dataset and why (please calculate the mean square error for these two fitting equations)?

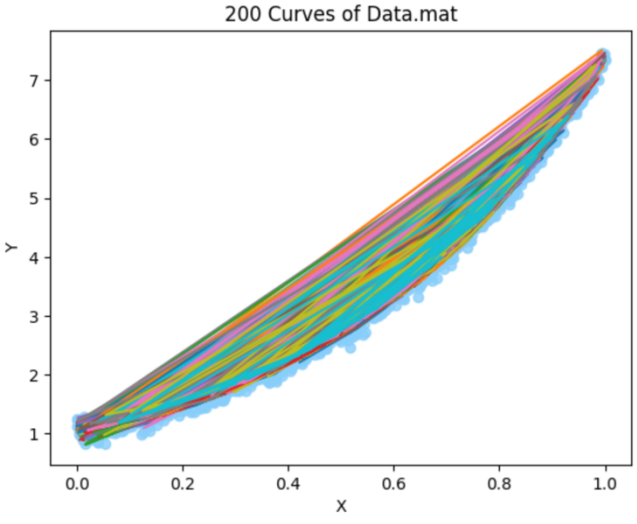


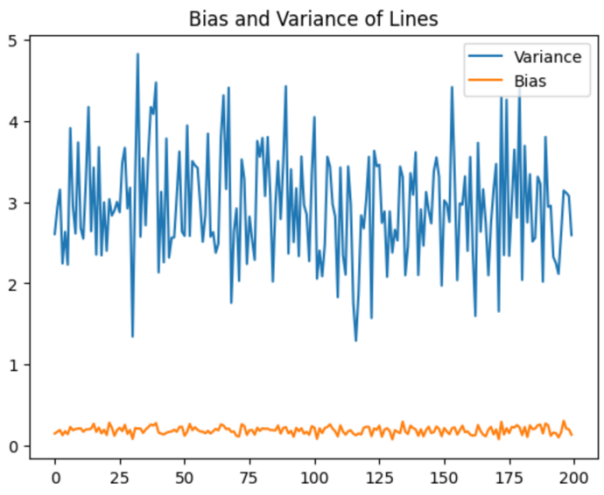

```
MSE of Line :  0.2058
MSE of Parabola :  0.0157
MSE of Quartic Curve:  0.0104
---------------------------------
Min of MSE :  0.0104 [Quartic Curve]
```

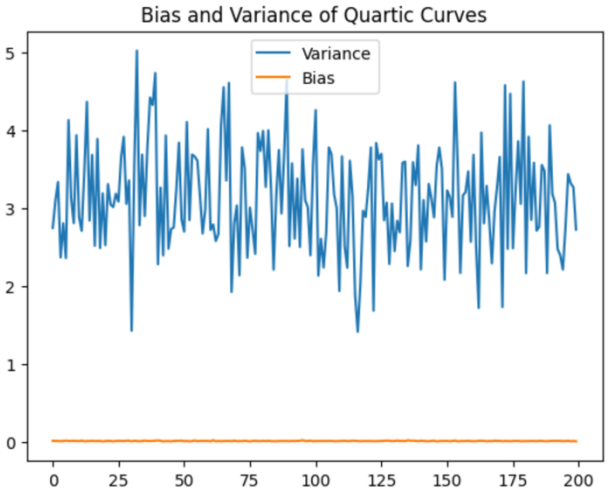
Ans：單純看 MSE 的話，Quartic Curve 的表現是三個之中最好的。
(它預測的值與正確答案誤差最小)

2. (**25%**) Following the previous two questions, please randomly select 30 data samples for 200 times and plot these 200 lines ($y = \theta_0 + x\theta_1$) and quartic curves ($y = \theta_0 + x\theta_1 + x^2\theta_2 + x^3\theta_3 + x^4\theta_4$) in two separate figures, one for lines and the other for quartic curves. Explain these visualizations based on the bias and variance.



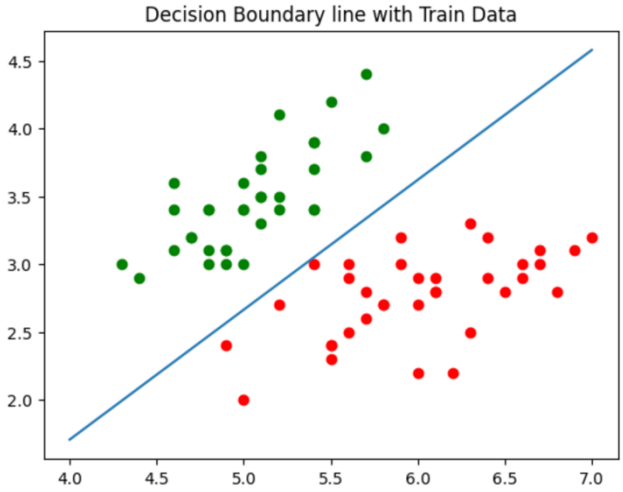






Ans：理想的狀態是找到 Low Bias 和 Low Variance，Bias 越小越接近 Groundtruth，而 Variance 越小代表每個點距離 Decision Boundary 越接近。就這組 dataset 來說，variance 在一次方及四次方的表現上差不多，但 Bias 在四次方明顯的有較好。

3. (**15%**) In 'train.mat,' you can find 2-D points X=[x1, x2] and their corresponding labels Y=y. Please use logistic regression $h(\boldsymbol{\theta}) = \frac{1}{1+e^{-\theta^T x}}$ to find the decision boundary(optimal $\boldsymbol{\theta}^*$) based on 'train.mat." Please report the test error on the test dataset 'test.mat.' (percentage of misclassified test samples)




```
Accuracy : 1.0
Confusion Matrix:
 [[35  0]
 [ 0 35]]
```

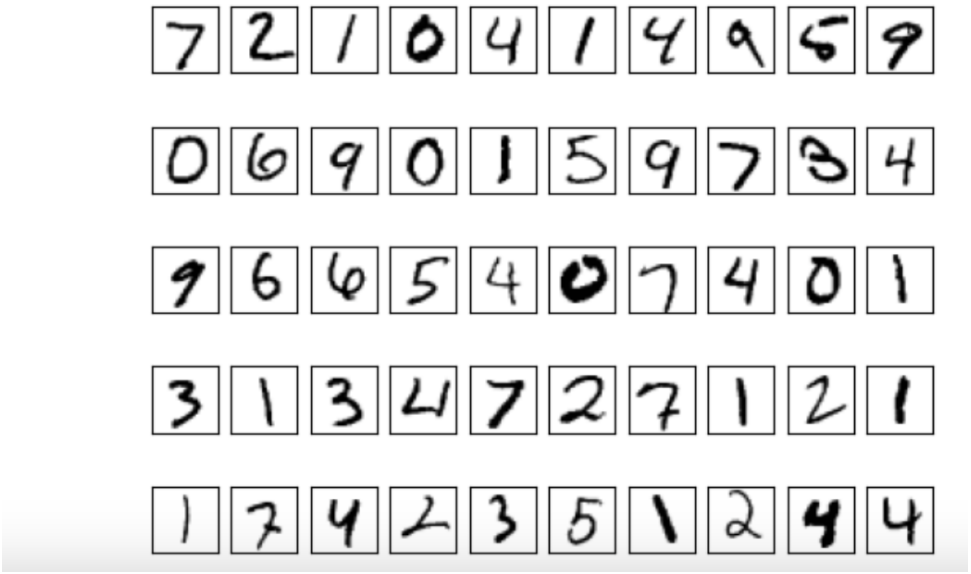

```
Accuracy : 1.0
Confusion Matrix:
 [[15  0]
 [ 0 15]]
```

Ans：這題所找到的 Decision Boundary 為：$y = 5.739957193207684 + (-2.5894065066866045)\,x + 2.7052721961394166\,x^2$，且 test error 為 0.0。(Accuracy 為 1.0)

4. Please randomly choose 5,000 different handwritten images from either the training or the testing dataset to construct your own dataset, where each digit has 500 data samples.

4.1 (**5%**) Use the following code to show 50 images in your own dataset.

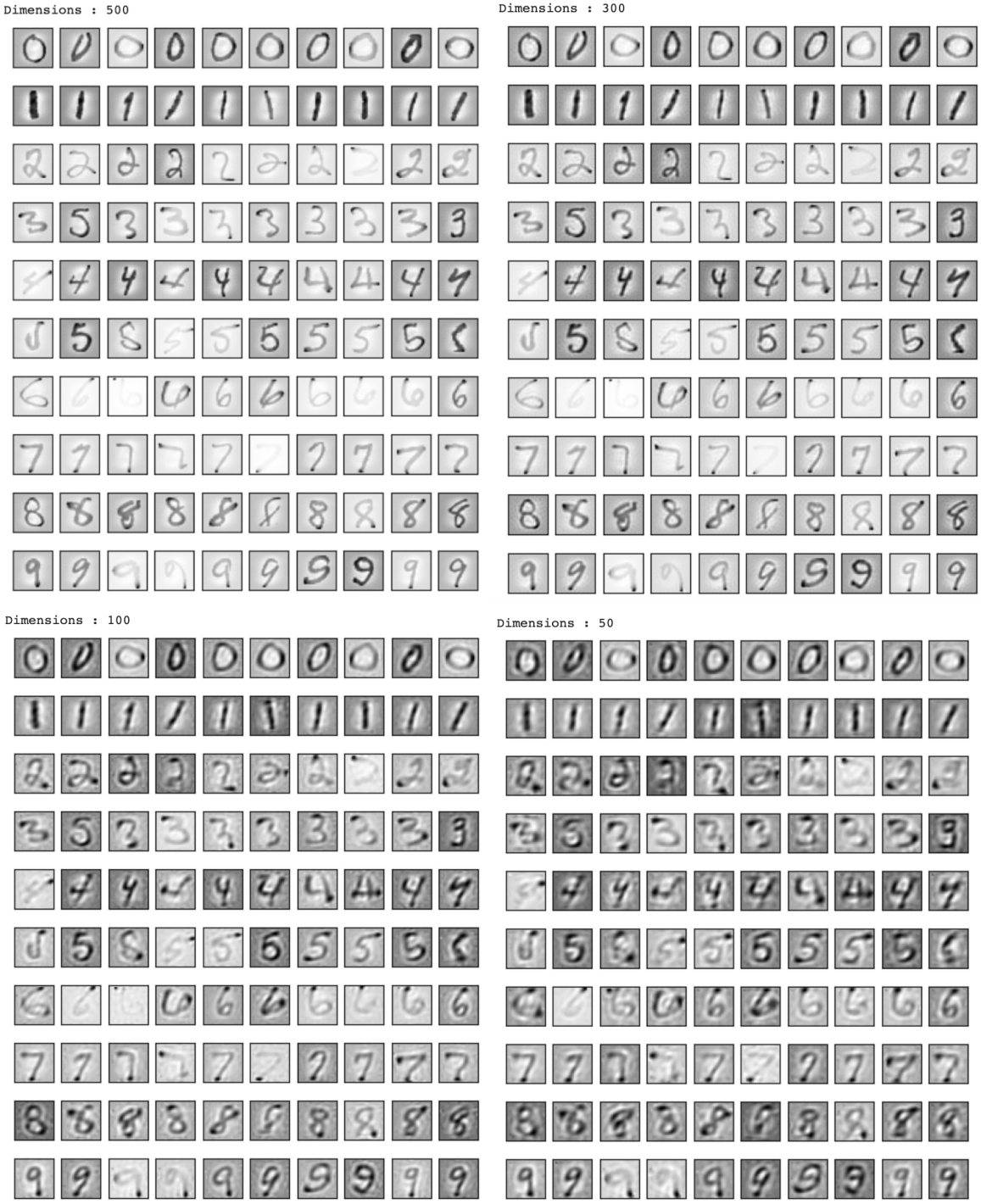
```
In [28]: import numpy as np
         import matplotlib.pyplot as plt

         def show_images(x):
             amount= 50
             lines = 5
             columns = 10
             number = np.zeros(amount)
             for i in range(amount):
                 number[i] = y_test[i]
                 # print(number[0])
             fig = plt.figure()
             for i in range(amount):
                 ax = fig.add_subplot(lines, columns, 1 + i)
                 plt.imshow(x[i,:,:], cmap='binary')
                 plt.sca(ax)
                 ax.set_xticks([], [])
                 ax.set_yticks([], [])
             plt.show()

         show_images(x_test)
```



4.2 (**15%**) Normalize the data (subtracting the mean from it and then dividing it by the standard deviation) and compute the eigenpairs for the covariance of the data (sorted in a descending order based on eigenvalues).

Ans：請參考 ipynb 檔案。

4.3 (**15%**) Please use PCA to reduce the 784 dimensional data to that with 500, 300, 100, and 50 dimensions, and then show 10 decoding results for each digit, respectively. How do you interpret these results?



Ans：PCA 降維是透過 eigenvalue 來判斷 eigenvector 的重要性，通常會取前 k 個 eigenvalue 最大的值所對應到的 eigenvector。以 k=500、300、100、50 為例，k 越小，圖所遺失的資訊會越多，所以當 k=50 時的圖片會最為模糊。