भारतीय प्रौद्योगिकी संस्थान धारवाड
Indian Institute of Technology Dharwad

Statistical Pattern Recognition

Assignment 4: Linear Models

Sawan Singh Mahara

M.Sc Research Scholar
191081003

April 15, 2020

Contents

Chapter 1

Model Overview

1 Pocket Perceptron

1.1 Brief

For the pocket perceptron model, the output y is given by linearly combining the d features of a data sample x

$$y = \sum_{i=1}^{d} w_i x_i + w_0 \tag{1.1}$$

If $y \leq 0$ then the prediction is class 0, else class 1.

1.2 Loss Function

When viewed as an optimisation problem, we try and minimise the following cost

$$J(W) = -\sum_{j:W^\top X_j \leq 0} W^\top X_j \tag{1.2}$$

Here, $W \in \mathcal{R}^{d+1}$ and $X_j \in \mathcal{R}^{d+1}$ where j denotes that sample point which is misclassified. The cost is just the summation of all those points misclassified, dot product product with the classifier W

1.3 Gradient of the Loss Function

The gradient of $J(W)$ with respect to W is found to be

$$\Delta J(W) = -\sum_{j:W^\top X_j \leq 0} X_j \tag{1.3}$$

1.4 Implementation

As we would like to reduce the number of misclassifications, our aim would be to minimise $J(W)$ over all W. The gradient descent method was used to do this. At the k'th iteration;

$$W^{k+1} = W_k + \eta \sum_{j:W^\top X_j \leq 0} X_j \tag{1.4}$$

Here, η is the learning rate, which was fixed at 0.1.

This was run multiple times and the W which gave the least misclassifications was taken as the final weight. This is termed as the pocket perceptron algorithm.

2 Linear Least Squares

2.1 Brief

Similar to the perceptron model, the output y is given by the same equation

$$y = \sum_{i=1}^{d} w_i x_i + w_0 \tag{1.5}$$

If $y \leq 0$ then the prediction is class 0, else class 1.

2.2 Loss Function

The difference between the two arises in the loss function considered. Given a dataset $\mathcal{D} = \{(x_i, y_i) : i \in \{0, .., N-1\}\}$ the loss is given by

$$J(W) = \sum_{j=0}^{N-1} (y_j - W^\top x_j)^2 \tag{1.6}$$

Here again, $W \in \mathcal{R}^{d+1}$ and $X_j \in \mathcal{R}^{d+1}$.

2.3 Implementation

A closed form solution W^* exists that minimises $J(W)$

$$W^* = (X^\top X)^{-1} X^\top Y \tag{1.7}$$

This was used to obtain the weights of the classifer.

3 Logistic Regression

3.1 Brief

The logistic regression model is obtained by computing the posterior probability of an assumed Gaussian class conditional density for a feature vector X. That results in the following form for the classifier.

$$y_i^{pred} = \sigma(W_i X_i) \tag{1.8}$$

Where $\sigma(z) = \frac{1}{1+e^{-z}}$
Again, if $y_i^{pred} > 0$, class 0 is picked, else class 1.

3.2 Loss Function

The logistic loss function for a dataset $\mathcal{D} = \{(x_i, y_i) : i \in \{0, .., N-1\}\}$ with $y \in \{0, 1\}$ is given by

$$J(W) = \frac{1}{N} \sum_{i=0}^{N-1} y_i \log(\sigma(W^\top x_i)) + (1 - y_i) \log(1 - \sigma(W^\top x^i)) \tag{1.9}$$

Here again, $W \in \mathcal{R}^{d+1}$ and $x_i \in \mathcal{R}^{d+1}$.

3.3 Implementation

The gradient of $J(W)$ can be shown to be

$$\Delta J(W) = \sum_{i=0}^{N-1} (y^i - \sigma(W^\top x^{(i)}))x^{(i)} \tag{1.10}$$

Using this gradient, the gradient descent algorithm was used to find the optimal W^*

$$W^* = (X^\top X)^{-1} X^\top Y \tag{1.11}$$

This was used to obtain the weights of the classifer.

4 Fischer's Linear Discriminant Analysis

4.1 Brief

The Fischer's Linear Discriminant Analysis method aims to find the direction that maximises the inter class differences, captured by the **between class scatter matrix $\mathbf{S}_B$** and **within class scatter matrix $\mathbf{S}_W$**

4.2 Optimisation Objective

Unlike in the earlier cases, the objective function is not a cost, but a measure of the differences between classes. We aim to maximise this function $J(W)$, given by

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \tag{1.12}$$

Where

$$S_B = \sum_c (\boldsymbol{\mu}_c - \overline{\mathbf{x}}) (\boldsymbol{\mu}_c - \overline{\mathbf{x}})^T$$

$$S_W = \sum_c \sum_{i \in c} (\mathbf{x}_i - \boldsymbol{\mu}_c) (\mathbf{x}_i - \boldsymbol{\mu}_c)^T$$

c is the set of all classes, each of μ_c is that particular class's estimate of the mean and $\overline{\mathbf{x}}$ is the mean of the class means.

4.3 Implementation

The optimal W^* that maximises equation 1.12 is given by

$$W^* = \mathbf{S}_W^{-1}(\mu_1 - \mu_2) \tag{1.13}$$

for the two class case, which was used in the implementation.

Chapter 2

Classification Task Results

1 Synthetic Dataset

1.1 Variation in Priors

The means for class 0 and class 1 were a 10 dimensional vector of zeros and ones, respectively. The covariance matrices were taken as the identity matrix in both cases.

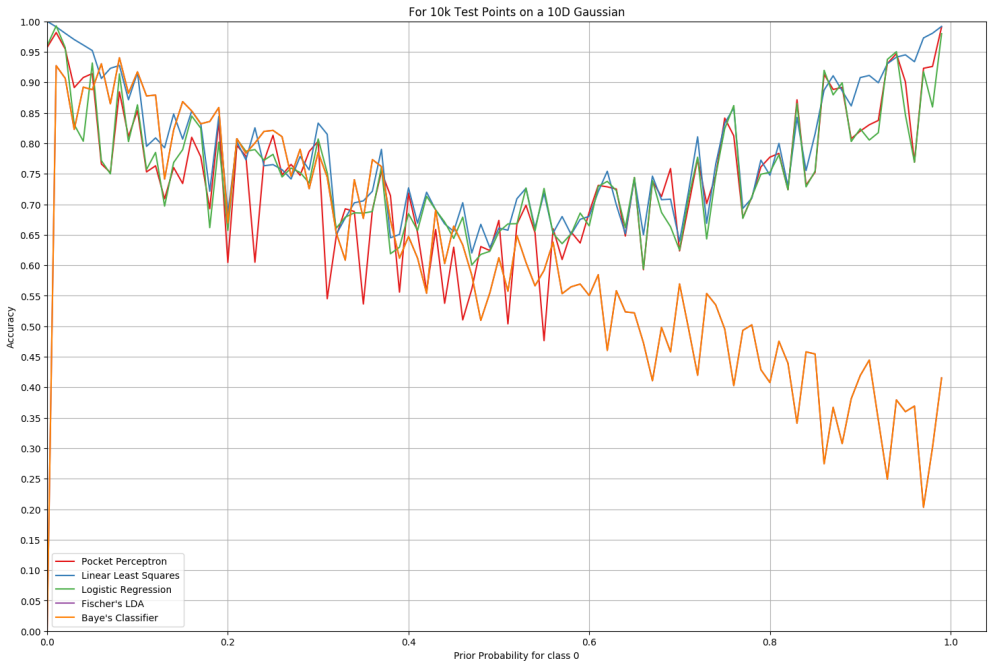


Figure 2.1: The performance of all the classifiers with varying prior probabilities (10-D data)

The same experiment was performed on 2D Gaussians as well.

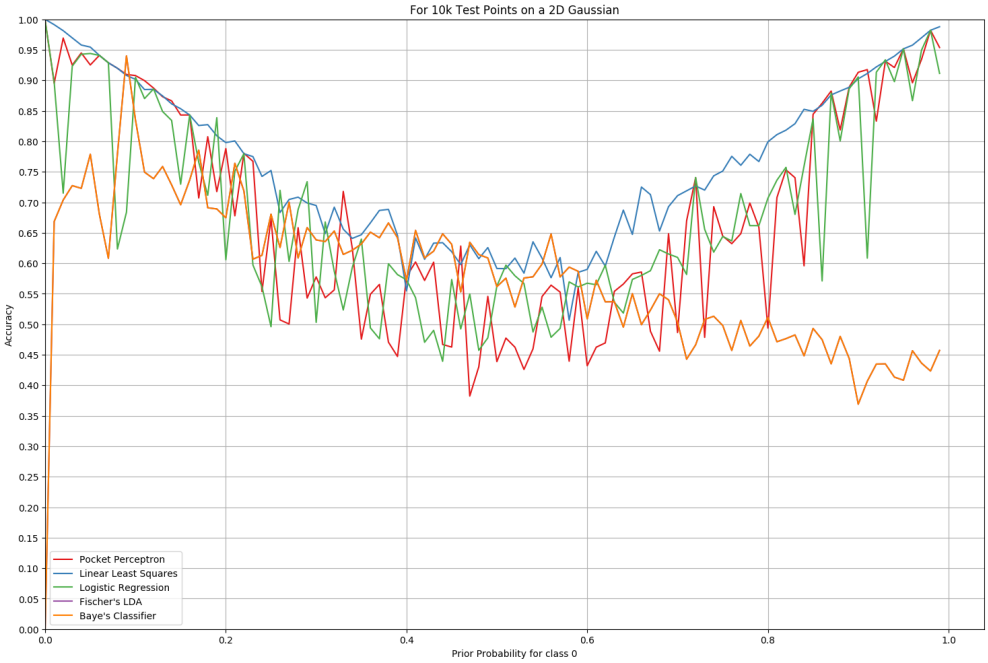


Figure 2.2: The performance of all the classifiers with varying prior probabilities (2-D data)

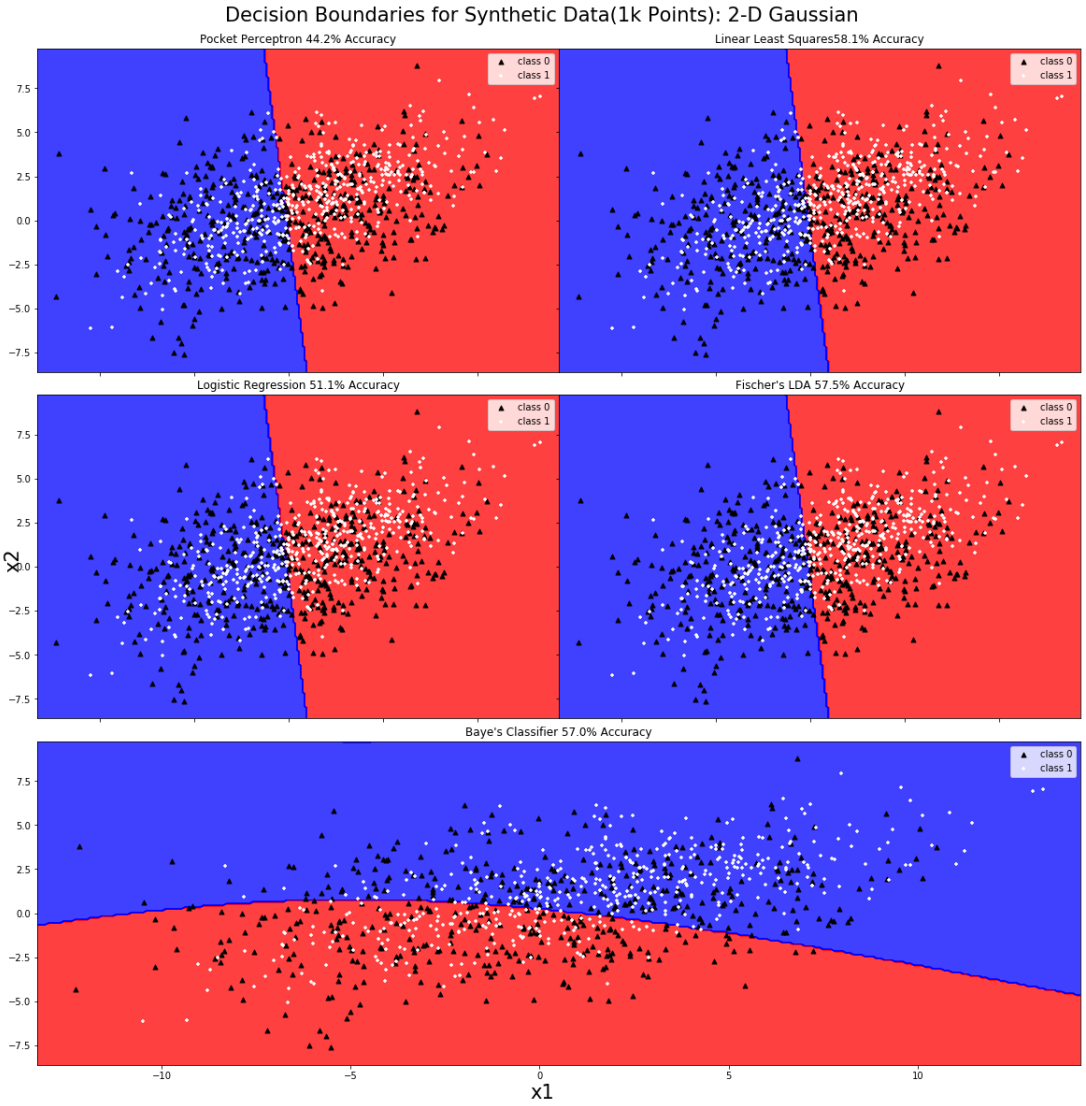


Figure 2.3: The resulting decision boundaries for all classifiers

The experiment was repeated on 10,000 data points as well, and the following decision boundary was obtained.

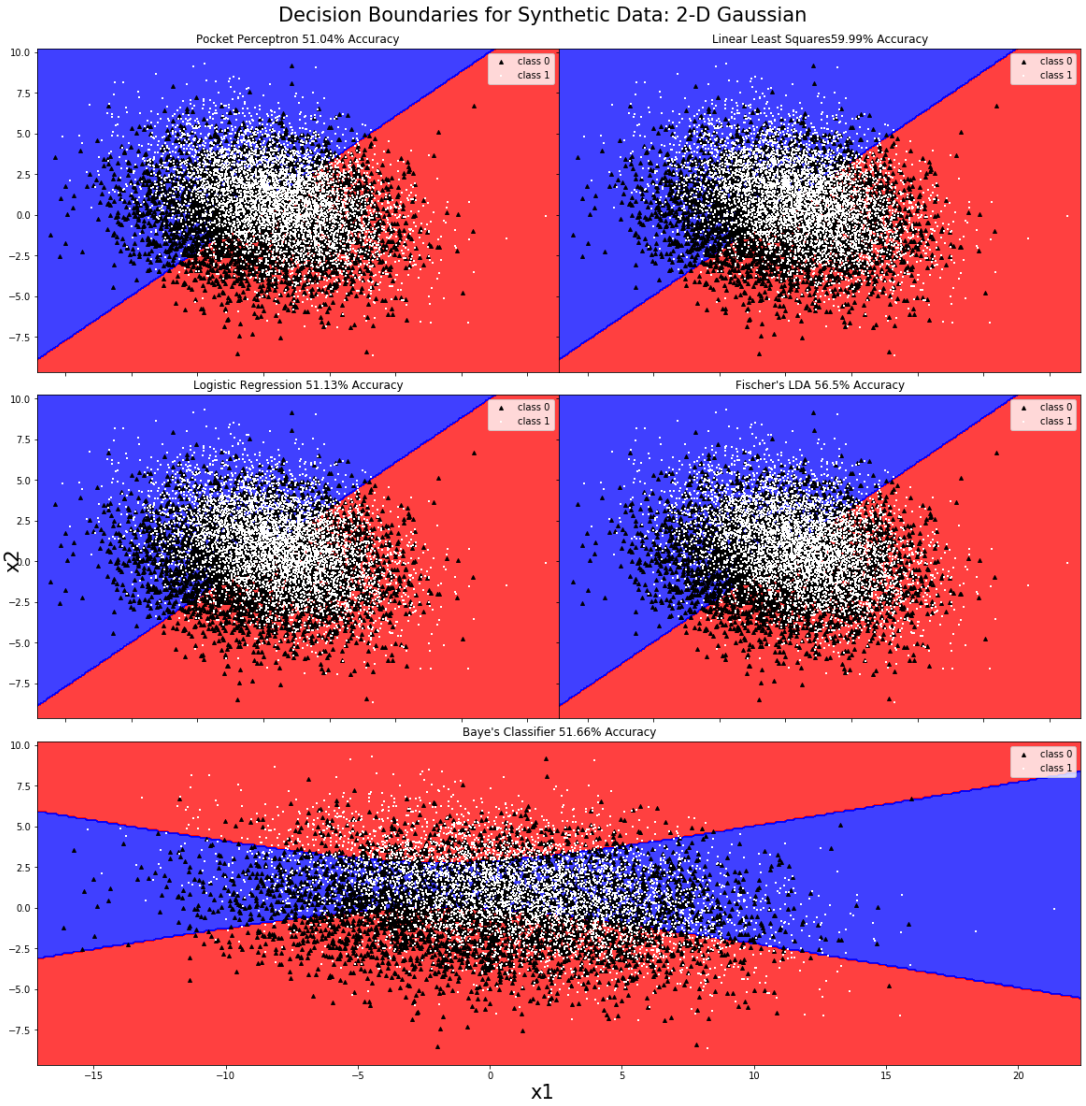


Figure 2.4: The resulting decision boundaries for all classifiers for 10k points

As is evident, the performance drops severely as the data are not that well separated in 2 dimensional space.

1.2 2-D Dimensional Dataset

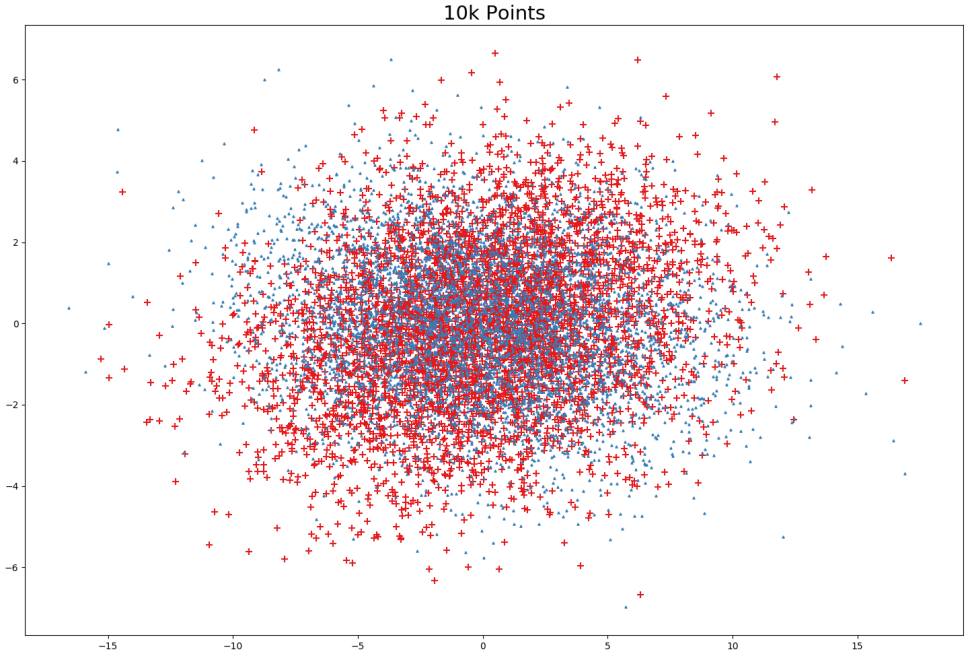


Figure 2.5: The dataset with 0 means and covariance matrices c_0 and c_1

Where,

$$c_0 = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix} c_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The data is not very well separated and class 0 has high correlation between features. The class 1 data, on the other hand, has independent features.

The resulting decision boundaries for all the classifiers were obtained as follows;

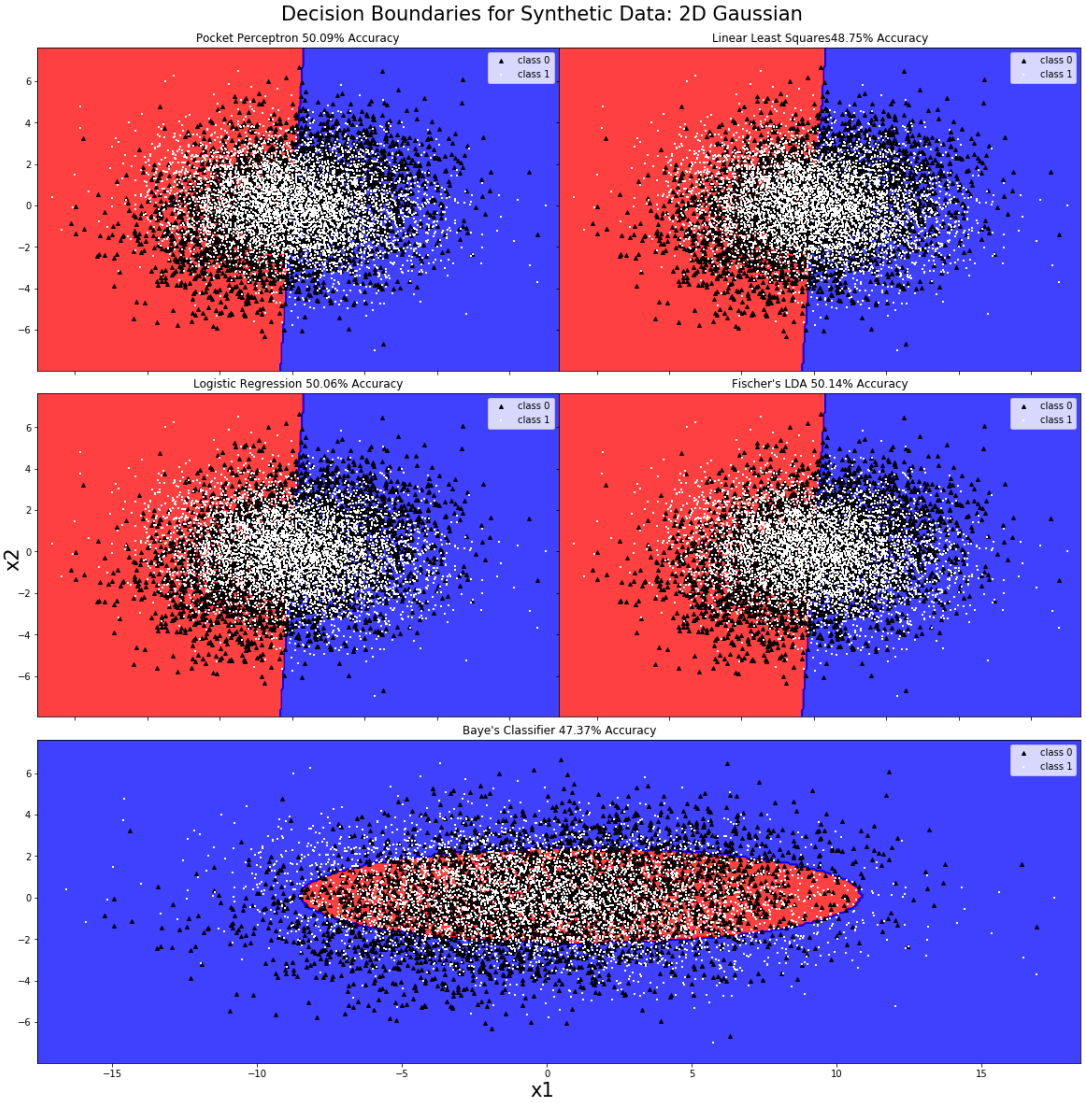


Figure 2.6: As expected, the performance is bad as the classes are clearly not linearly seperable

Clear Clusters

Finally, the analysis was done on the following dataset;

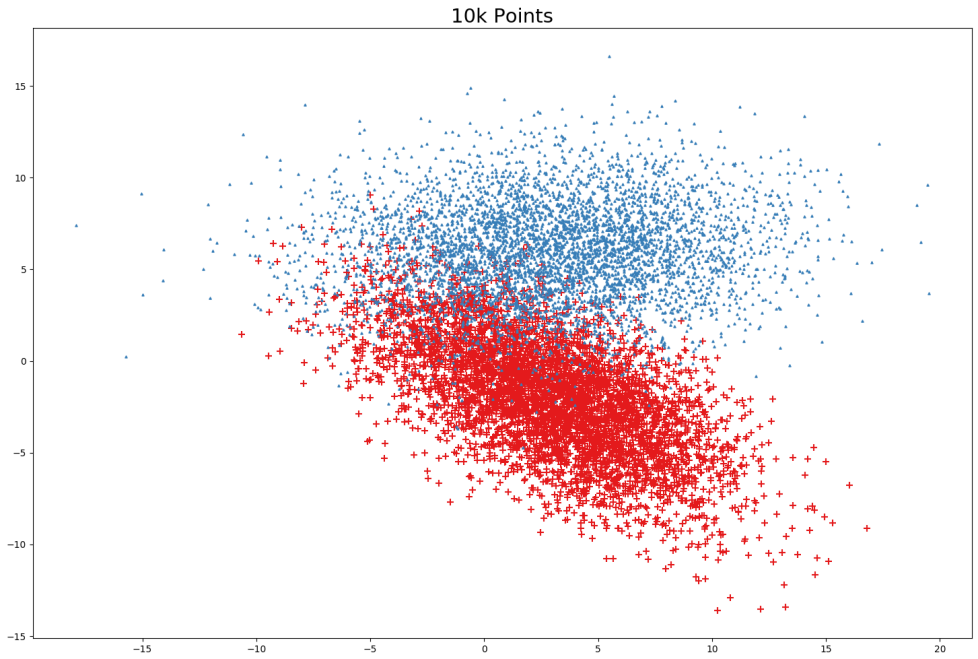


Figure 2.7: The dataset with means m_0 and m_1 and covariance matrices c_0 and c_1

Where,

$$m_0 = \begin{bmatrix} 3 \\ -2 \end{bmatrix} m_1 = \begin{bmatrix} 3 \\ 6 \end{bmatrix} \quad c_0 = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix} c_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The data is somewhat separable and the decision boundaries obtained give fairly good results.

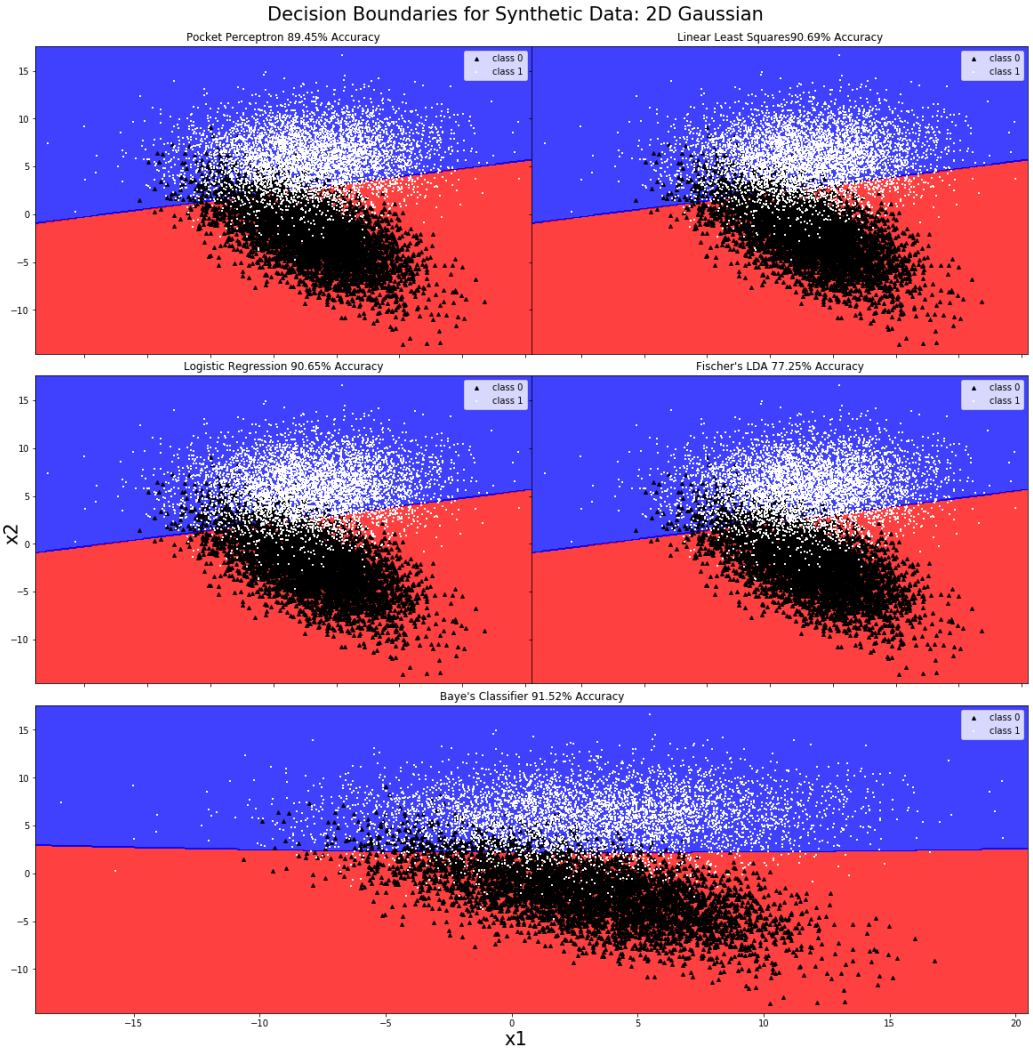


Figure 2.8: Decision boundaries

2 MNIST Handwriting

The multi-class classifier was run on the MNIST digit dataset on digits 5,7,8,9. The entire dataset is sampled in figure 2.9. The algorithms implemented were one vs one and one vs all. They were all hard-coded, except in the case for Fischer's LDA where the sci-kit learn library was used.

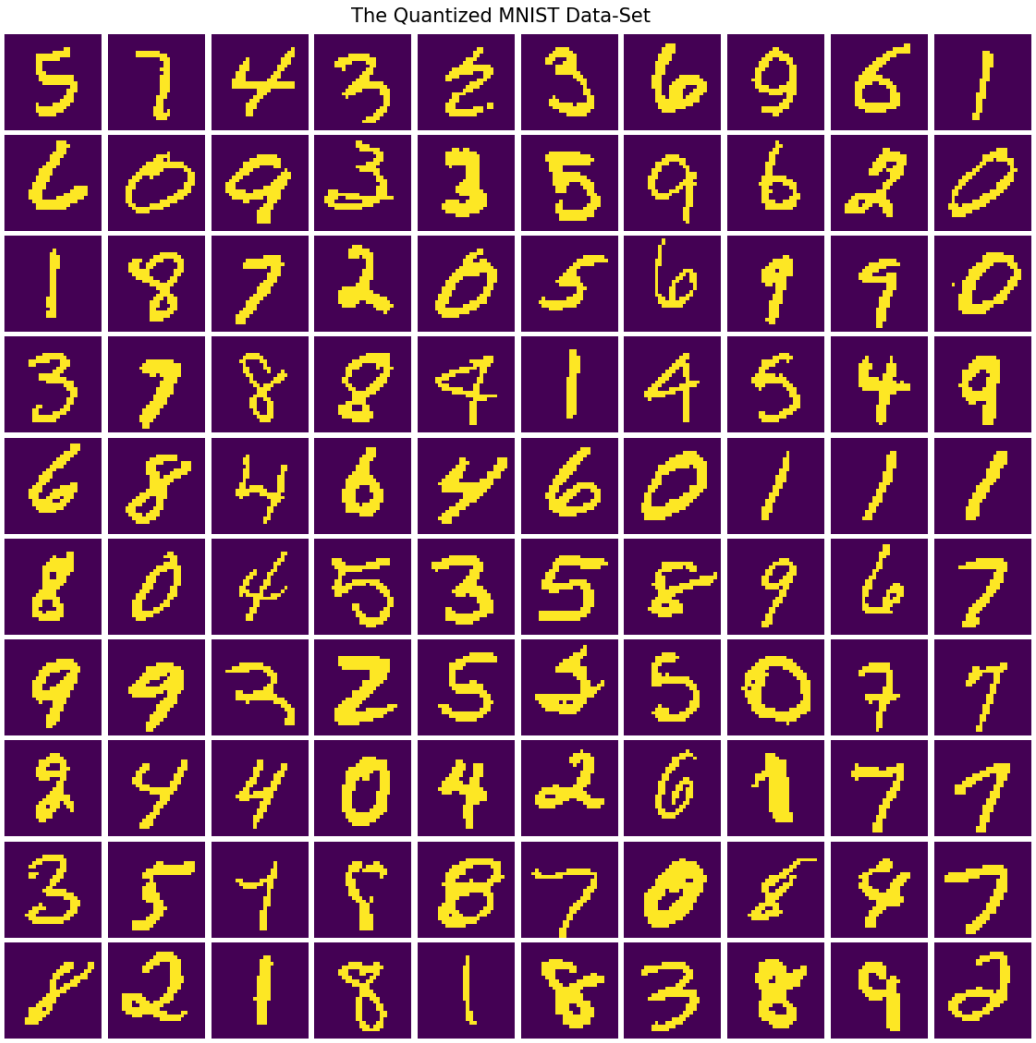
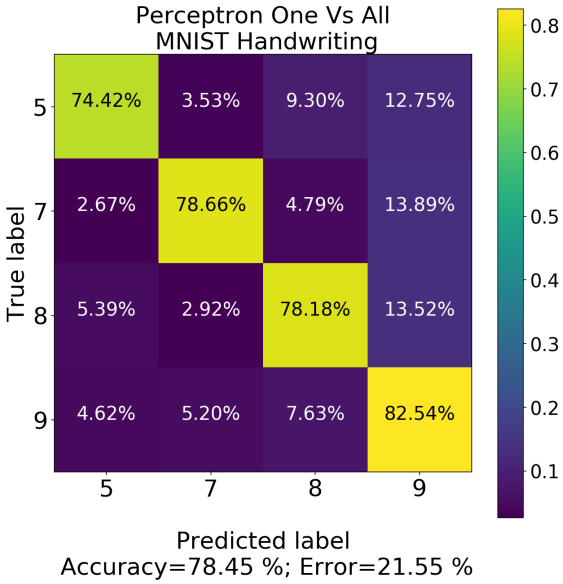
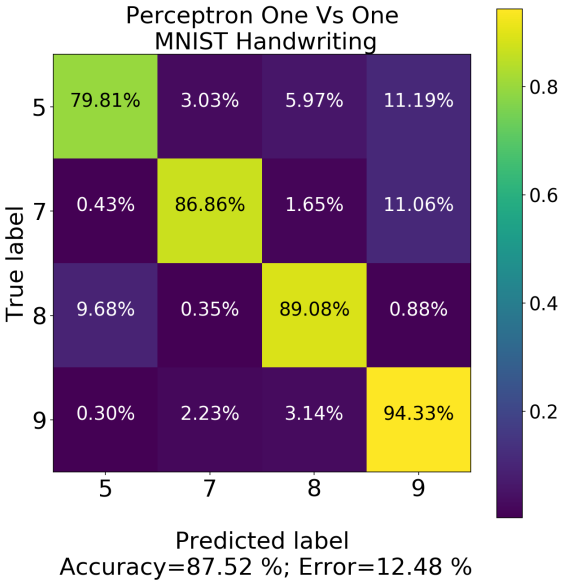
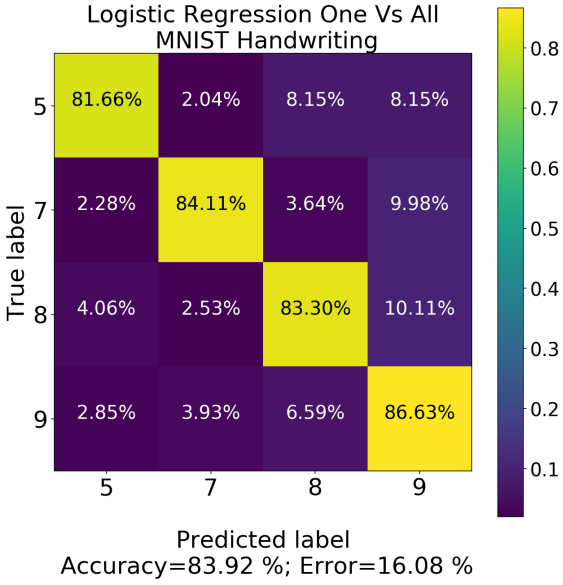


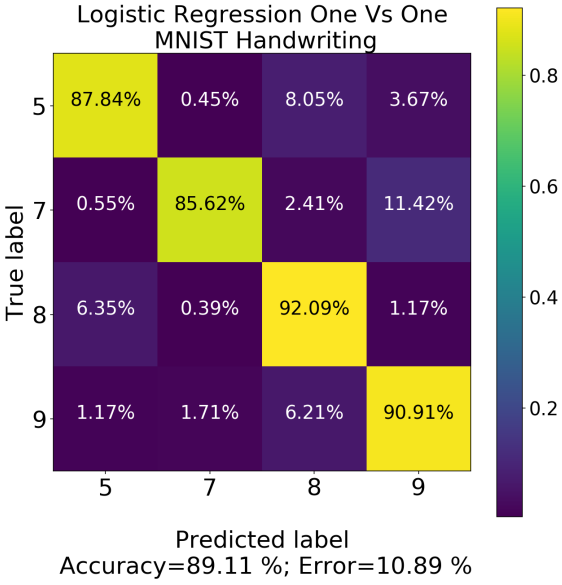
Figure 2.9: MNIST Dataset

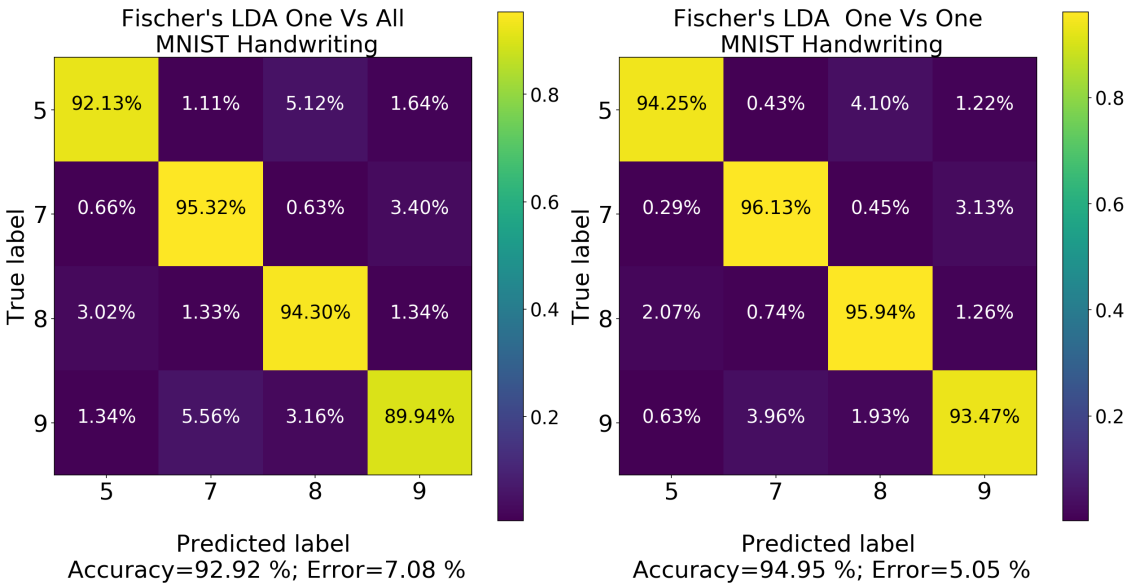
2.1 Confusion Matrices










Fischer's LDA One Vs All
MNIST Handwriting
True label
Predicted label
Accuracy=92.92 %; Error=7.08 %
92.13%
1.11%
5.12%
1.64%
0.66%
95.32%
0.63%
3.40%
3.02%
1.33%
94.30%
1.34%
1.34%
5.56%
3.16%
89.94%
Fischer's LDA One Vs One
MNIST Handwriting
True label
Predicted label
Accuracy=94.95 %; Error=5.05 %
94.25%
0.43%
4.10%
1.22%
0.29%
96.13%
0.45%
3.13%
2.07%
0.74%
95.94%
1.26%
0.63%
3.96%
1.93%
93.47%

3 German Credit Dataset

3.1 Dataset Overview and Pre-Processing

Context

The original dataset contains 1000 entries with 20 categorial and symbolic attributes prepared by Prof. Hofmann. Here, each entry represents someone who requests a loan from a bank. Each person is classified as either a high or low risk, according to the defined set of attributes.

Features Present and the Ones Selected

The dataset contains the following attributes of a person:

- Whether a checking account already exists or not (existingchecking)
- The duration for the load (duration)
- The credit history of the person (credithistory)
- The purpose for the loan(purpose)
- The amount of loan being asked for (creditamount)
- The amount in savings present (savings)
- Duration of employment (employmentsince)
- Installment rate (installmentrate)
- Their sex (sex)
- Other loans in their name (otherdebtors)
- Duration of residency (residencesince)
- Information about property owned (property)
- Their age (age)
- Other EMI's in their name(otherinstallmentplans)
- Type of house owned (housing)
- Existing loans with the current bank (existingcredits)
- Employment type (job)

- How many liabilities are possessed (peopleliable)
- If they have a telephone or not (telephone)
- If they are a foreign worker or not (foreignworker)
- Are the a high risk or not (classification)

The blue features are binary, the pink are integers/floats and the green ones are categorical which were looked at as one-hot encoded features.

The age and residence since features were ignored and the numerical columns were normalised.

Results

The following confusion matrices were obtained when the logistic regression classifier was used to classify the data.

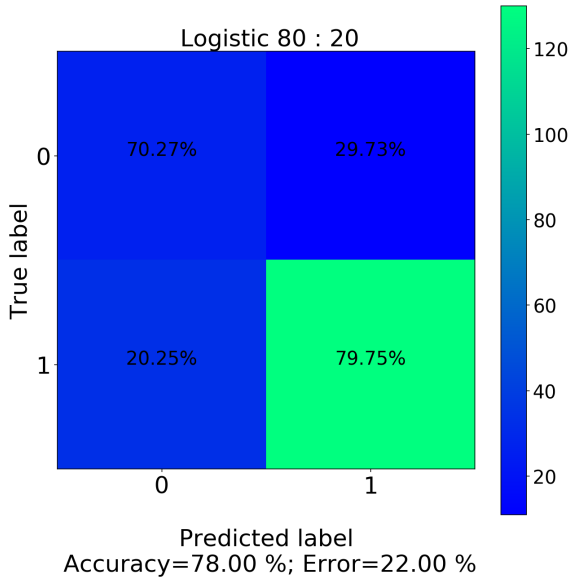


Figure 2.13: Logistic Regression with 80:20 split

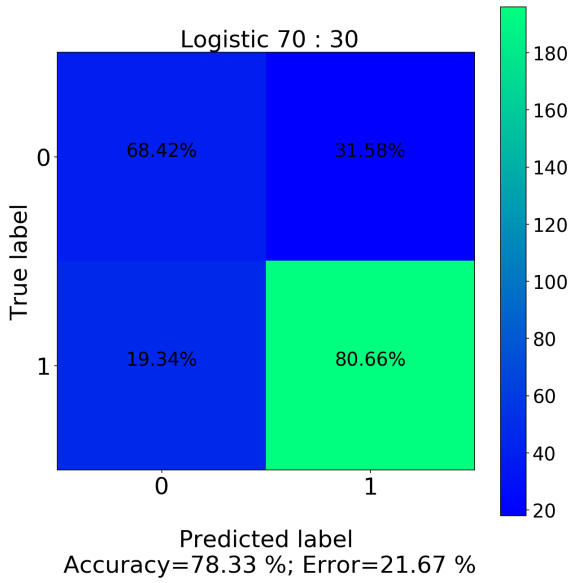


Figure 2.14: Logistic Regression with 70:30 split

Chapter 3

Regression Task

The polynomial is given by $y = 0.25x^3 + 1.25x^2 - 3x - 3$

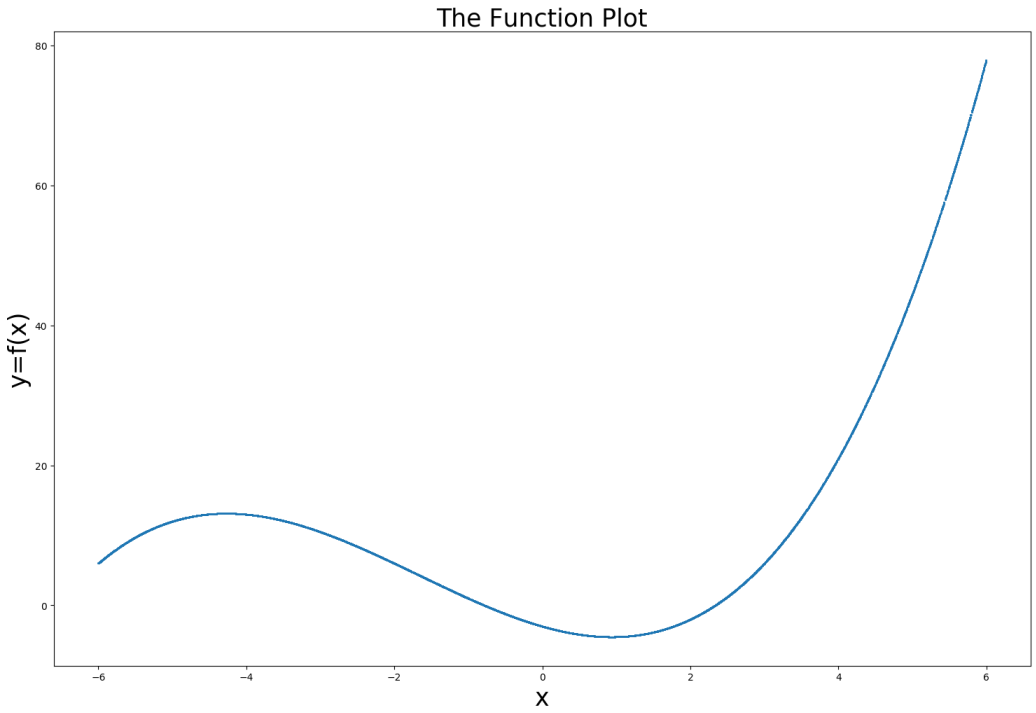


Figure 3.1: Decision boundaries

1 Results

1.1 On the Given Polynomial Function

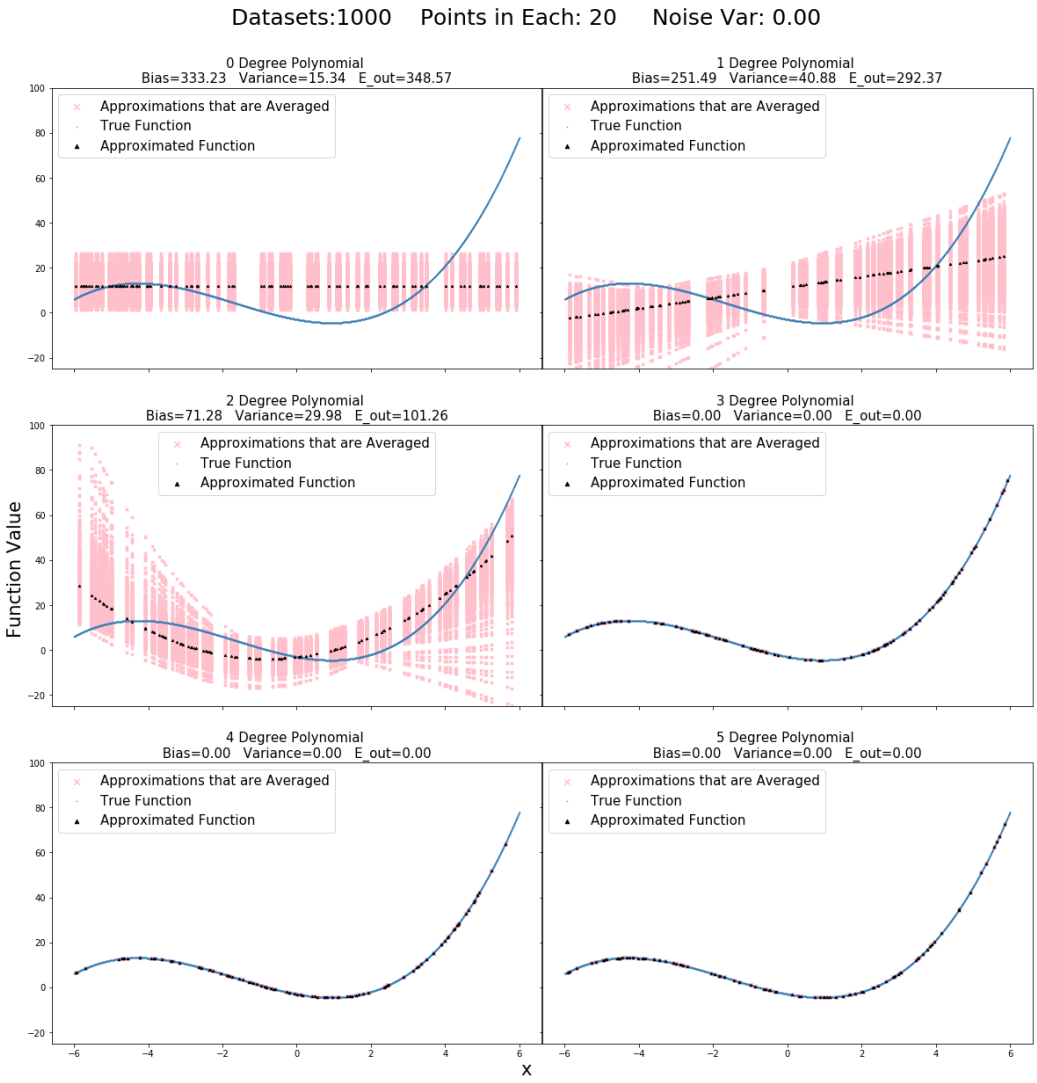


Figure 3.2: The best fit is a 3 degree polynomial

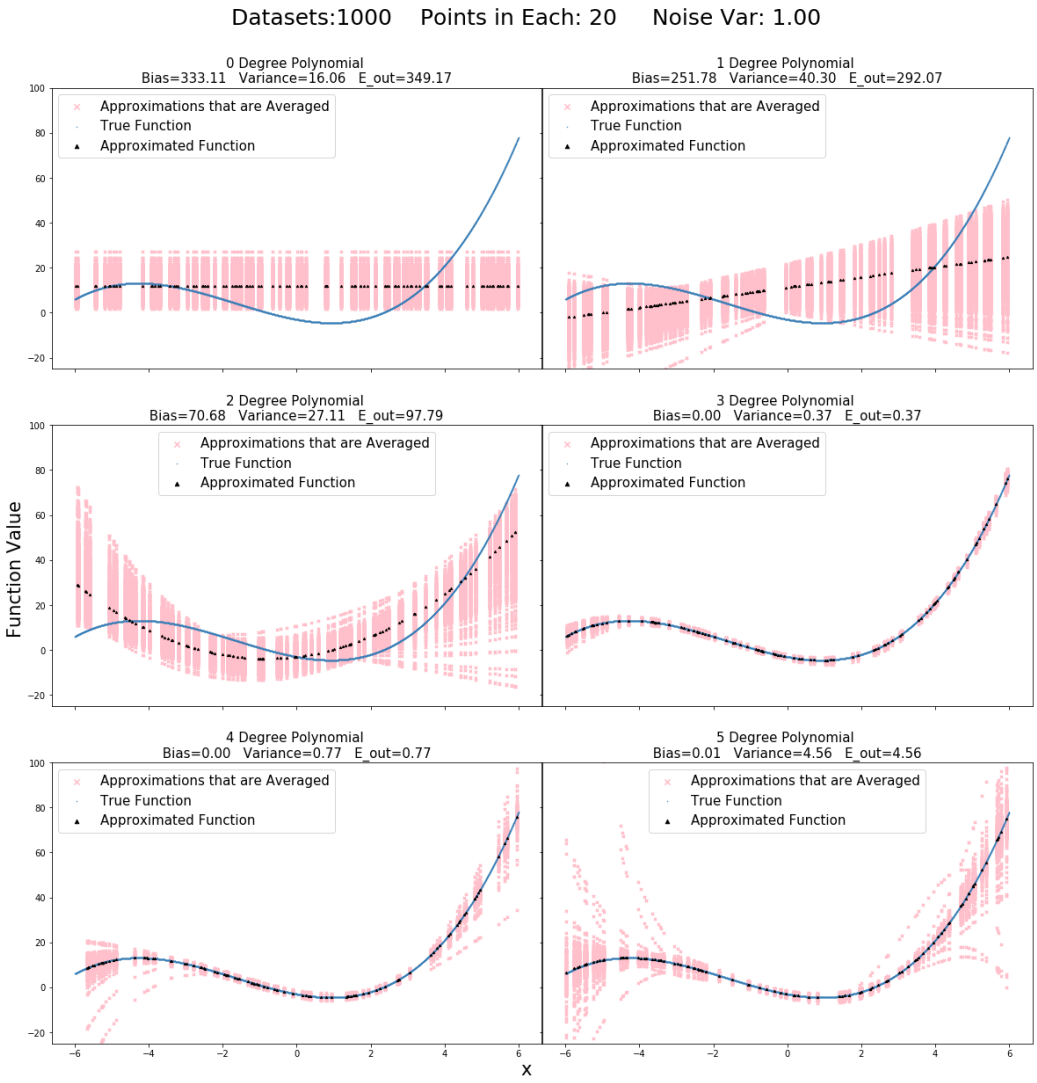


Figure 3.3: The best fit is a 3 degree polynomial

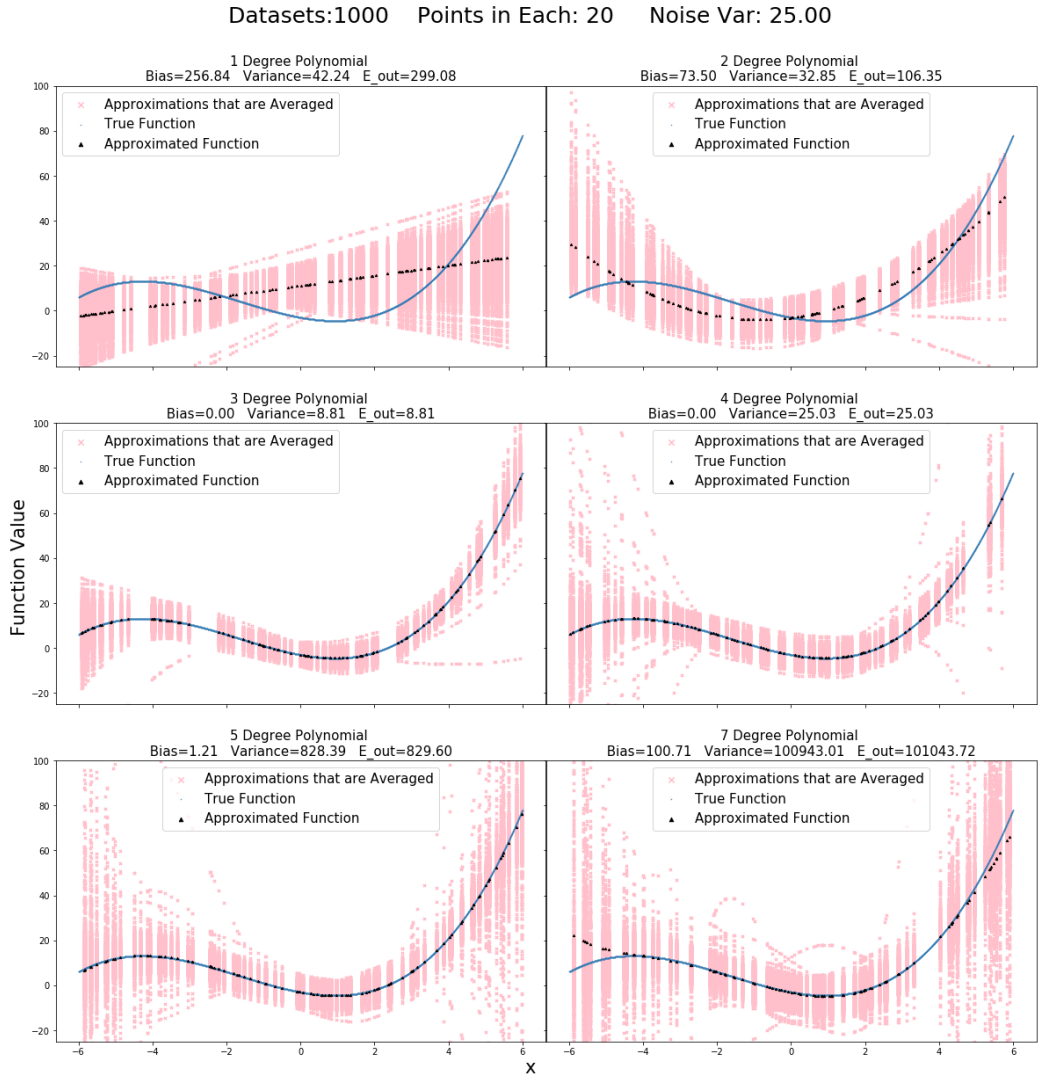


Figure 3.4: The best fit is any 3 degree or higher polynomial

The observation is that as the noise added increases, higher degree polynomials tend to fit the noise more and so lower degree polynomials seem a better fit for any given noisy data.

1.2 On a Sinusoidal Function

Running the algorithm on a sinusoidal function $y = sin(\frac{x\pi}{6})$, the results are as follows

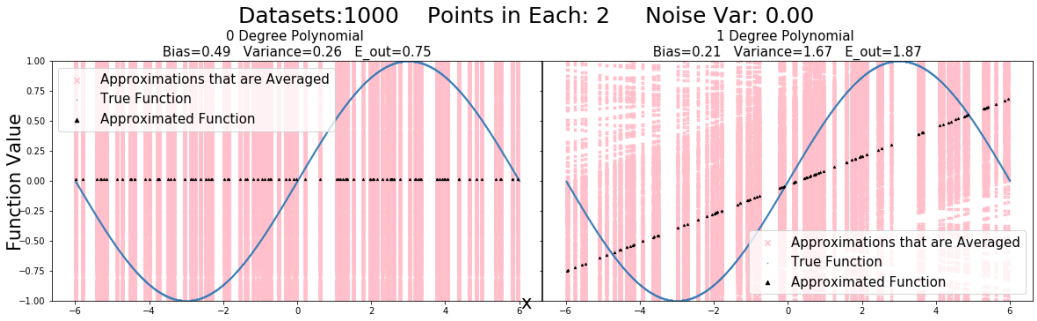


Figure 3.5: The best fit is a 0 degree polynomial

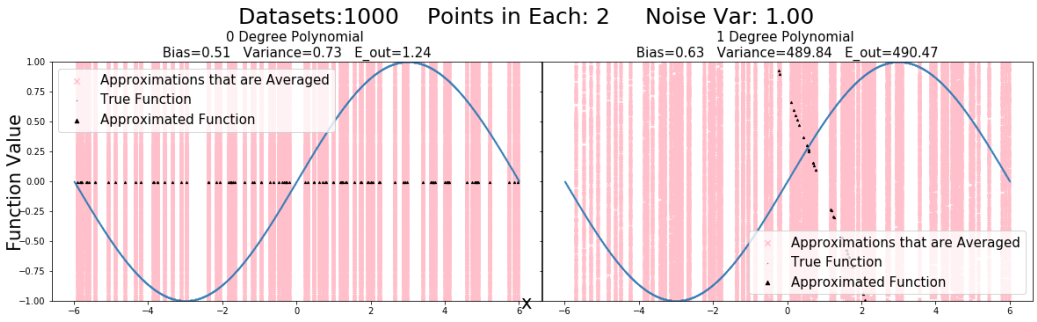


Figure 3.6: The best fit is a 0 degree polynomial

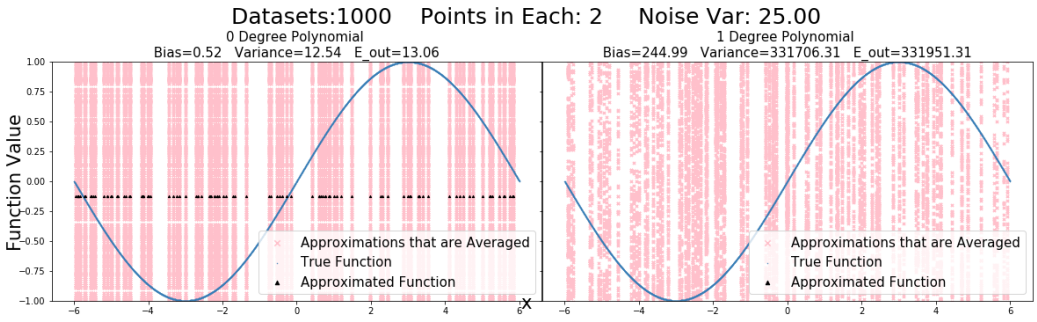


Figure 3.7: The best fit is a 0 degree polynomial

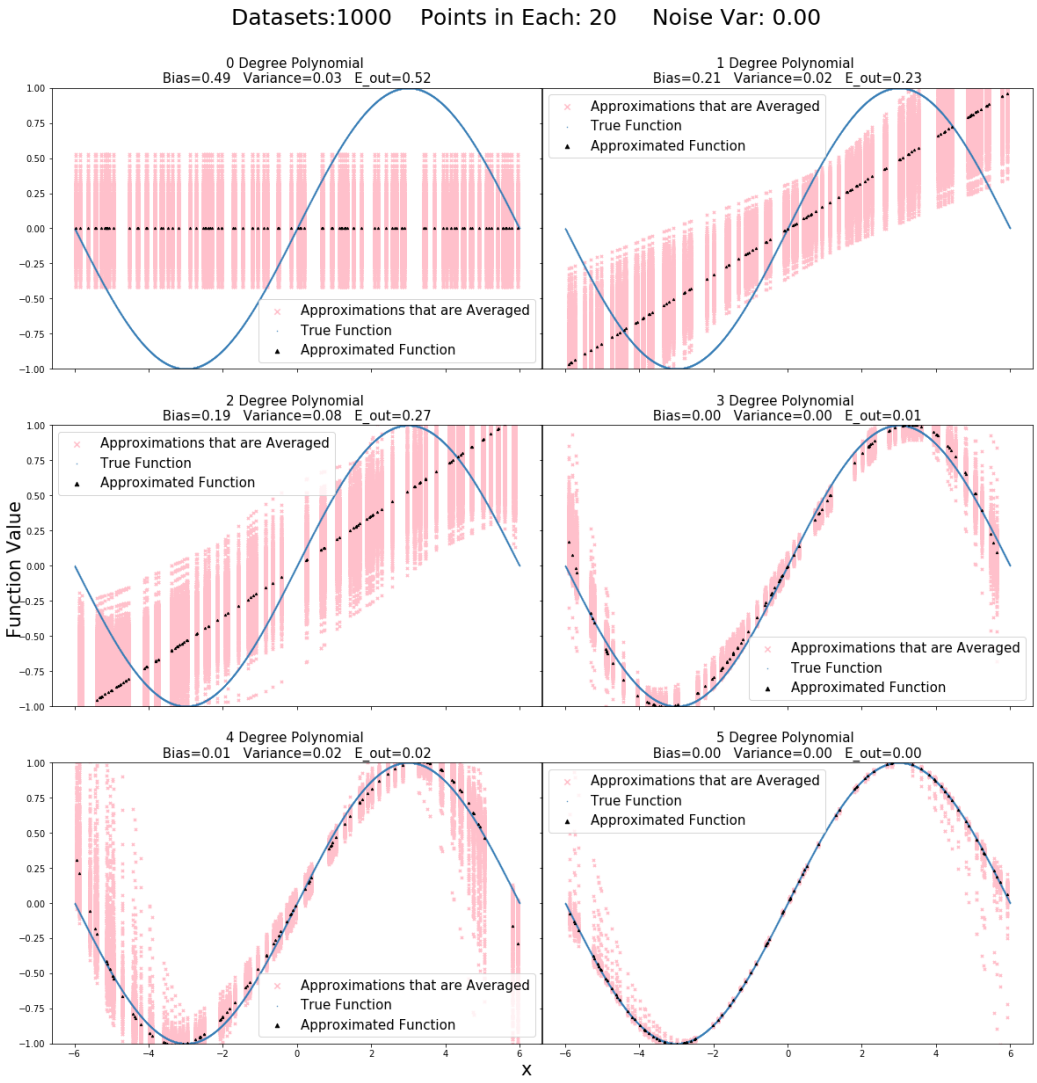


Figure 3.8: The best fit is a 5 degree polynomial

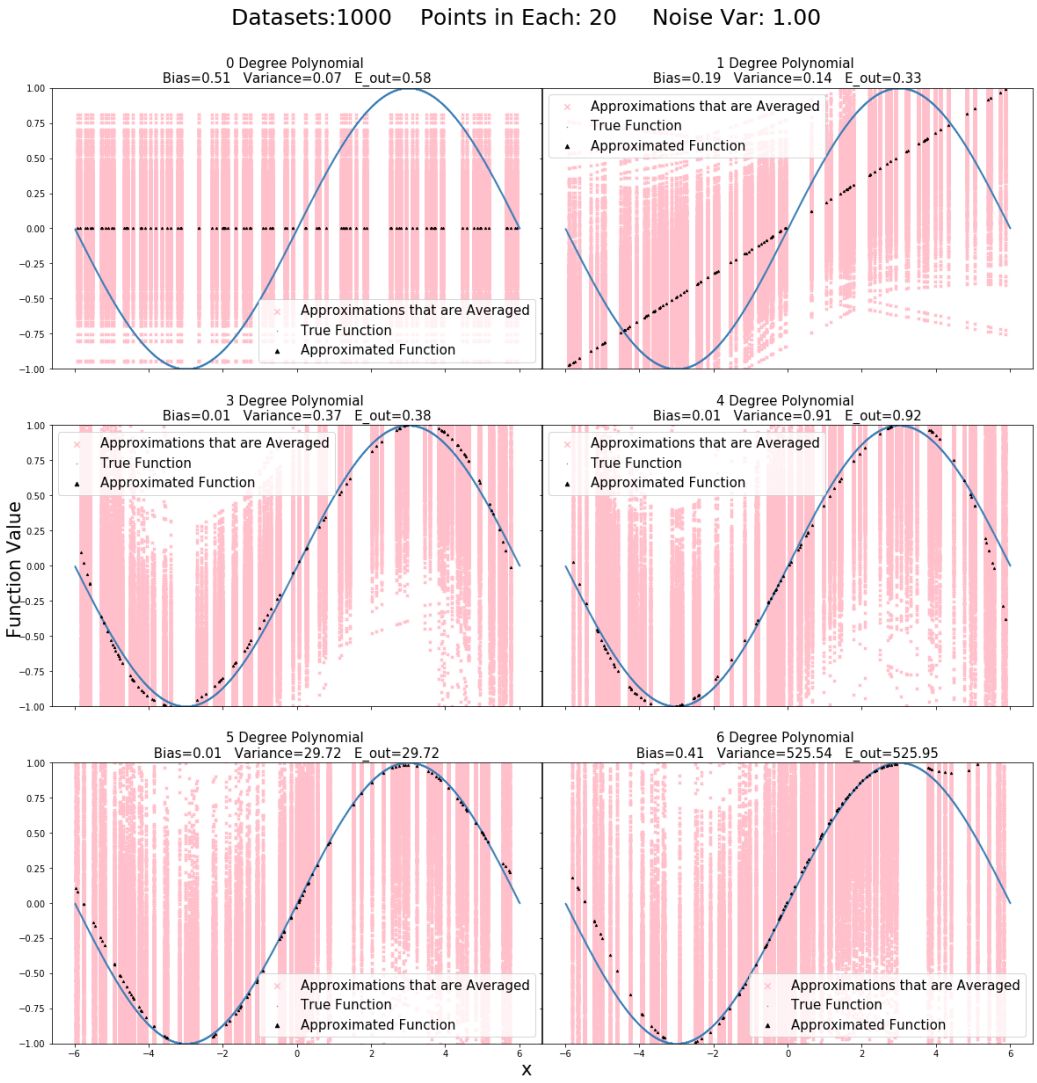


Figure 3.9: The best fit could be either a 2 or 3 degree polynomial.

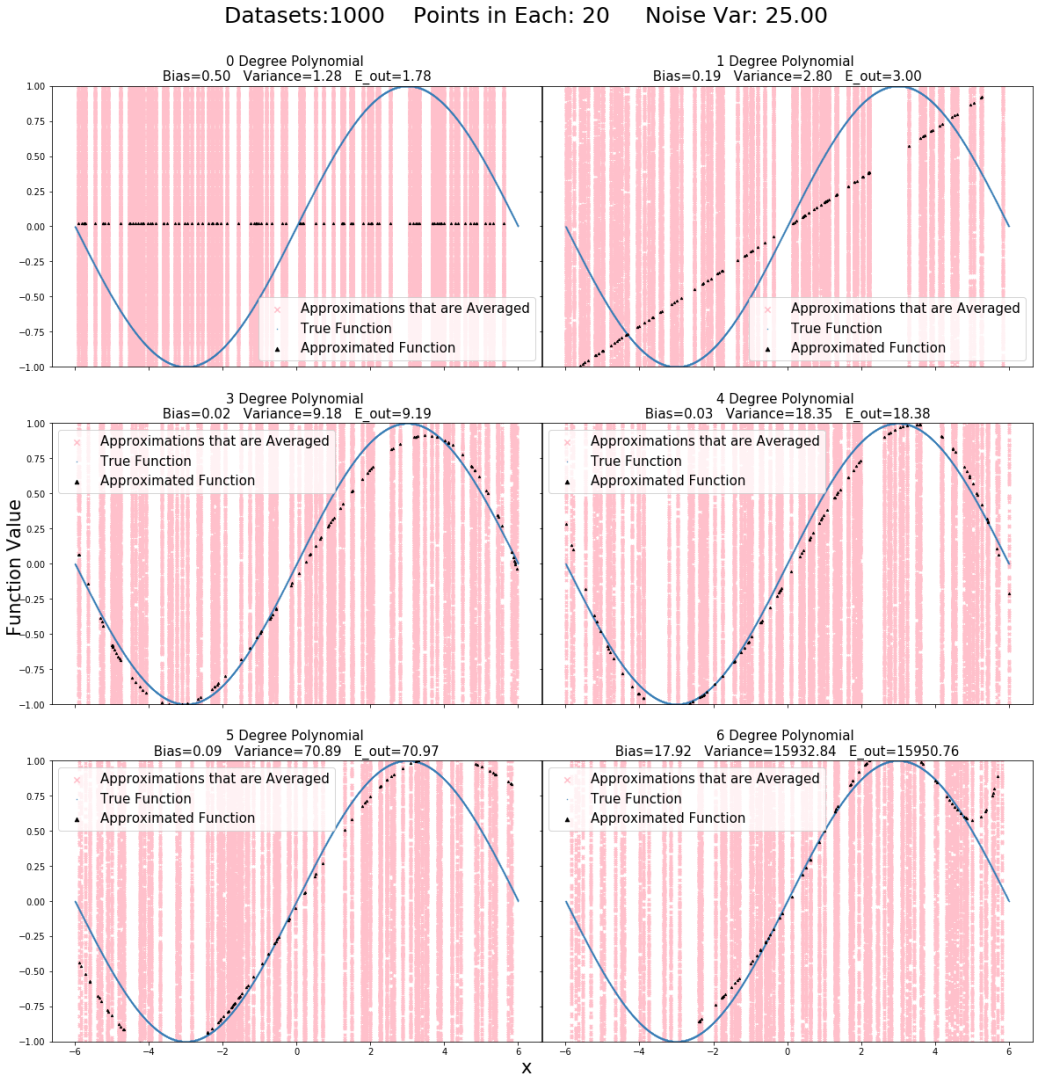


Figure 3.10: The best fit is a 0 degree polynomial

As evident, the best fit polynomial becomes harder to discern at a small noise variance, but settles down at a 0 degree polynomial for very noisy samples. For 0 noise, the best fit is any high degree polynomial, in this case, anything above 5 (figure 3.8).