Project 1: Linear Feature Engineering

Group members: Shafizur Rahman Seeam, and Ye Zheng.

Given the training data $X = \{x_1, x_2, \ldots, x_8\}$ and its label Y, there are two methods for solving this fitting problem:

- See X as a whole part and find $P(X)$ to fit $X \to Y$. For example, $P(X) = X^3 + X^2 + \sin X$.
- See $x_1, x_2, \ldots x_8$ as 8-dimensional inputs and find $P(x_1, x_2, \ldots x_8)$ to fit. For example, $P(x_1, x_2, \ldots x_8) = x_1^3 + x_1^2 + x_2^3 + x_2 + x_3 \ldots$.

We choose the second method.

Error

Our training error is:

$$\frac{1}{926} \sum_{i=1}^{926} (y_i - y_i^*)^2 = 46.4455$$

Our prediction for the final test error is:

$$\frac{1}{103} \sum_{i=1}^{103} (y_i - y_i^*)^2 \approx 58.4311881781144$$

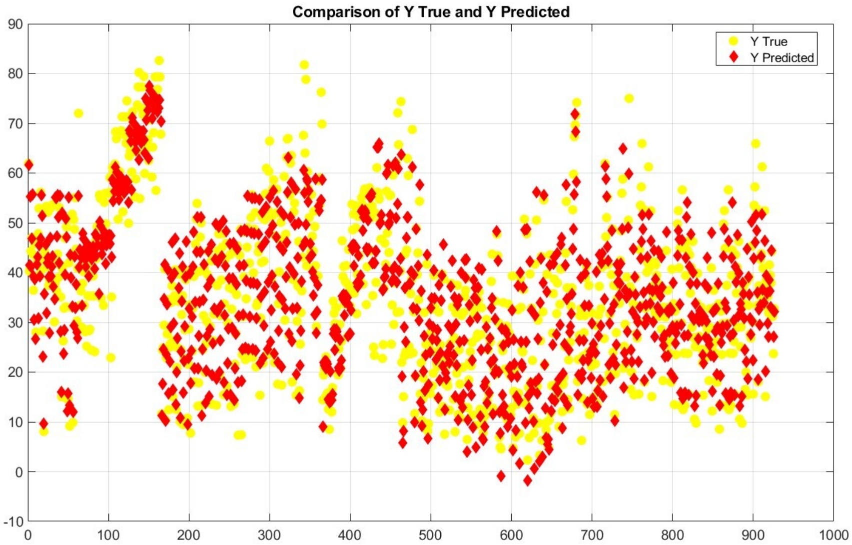
Features

Our method finds a feature for each dimension x_i, so the whole feature may be complex. We assume the system can be approximated by polynomial. To find a suitable polynomial, we conducted the following procedure:

- First try $P = x_1^d + x_2^d + x_3^d + \ldots x_8^d$ (each dimension has the same degree and only the d degree item) to approximate the maximum degree. We found when $d = 3$ or $d = 4$ the fitting errors are lower than other values.
- Set maximum degree $d_{max} = 3$, then use brute force to test the performance of all the polynomials under degree 3 (for x_1, the feature can be $a_1 x_1^3 + a_2 x_1^2 + a_3 x_1^1$).
- Find the polynomial feature having the lowest mean cross validation error. Here we found (3 2 1 3 3 1 1 3) for the eight dimensions respectively, and we also add a constant item. So the whole feature is (3+2+1+3+3+1+1+3)+1=18 dimensional.

Note: This method has more freedom in choosing features than using X as a whole part. For example, we can add feature $x_1 * x_2$ to "capture" the dependence between dimensions.

Following is the scatter figure showing the true value in whole training dataset wrt our predicted value ($y = x$ line means 0 error):



Prediction for the Test Error

We chose different values of K to see the mean value of the K-fold training error:

K = 8	K = 9	K = 10
60.5016	58.4312	60.5295

Our prediction for the test error is the mean value of the K-fold training error (we use 9-fold) in cross validation. That is:

$$\frac{1}{9}\sum_{i=1}^{9}\frac{1}{926/9}\sum_{j=1}^{926/9}(y_j - y_j^*)^2 = 58.4311881781144$$

Dealing with Overfitting

We used cross validation to assign training samples for each potential feature and use the left batch for testing. Specifically, our code structure is:

```
for a given feature
    (cross validation) assign training and testing data batch
        train a local model on the assigned training data
        collect the testing error using the assigned testing data
    collect the mean testing error of the 9 models
from the mean testing error determine which feature to use
```