MCGILL UNIVERSITY

# Assignment 1

Nabil Chowdhury

ID: 260622155

COMP 551

January 26, 2018

## 1 MODEL SELECTION

### 1.1 FITTING 20-DEGREE POLYNOMIAL TO THE DATASET

Dataset-1 consists of a real-valued scalar as input, and a real-valued scalar as output. In order to fit a 20 degree polynomial, we must create features $x^2$, $x^3$,..., $x^{20}$, since $x^1$ is already given to us. We must also include $x^0 = 1$ in the input (the bias term). We must find $w_0$, $w_1$,..., $w_{20}$ such that $\hat{y} = w_0 + w_1 x^1 + ... + w_{20} x^{20}$, where ŷ is the prediction.

Applying the closed form solution $w = (X^T X)^{-1} X^T y$ yields the least-squares weights $w$ for the polynomial. Using these weights, the following mean-squared-errors for the training, validation and test sets were calculated:

| **Set** | **MSE** |
|---|---|
| Train | 7.15259 |
| Valid | 458.64632 |
| Test | 17.25163 |

Here is the fit on the training, validation, and test sets:

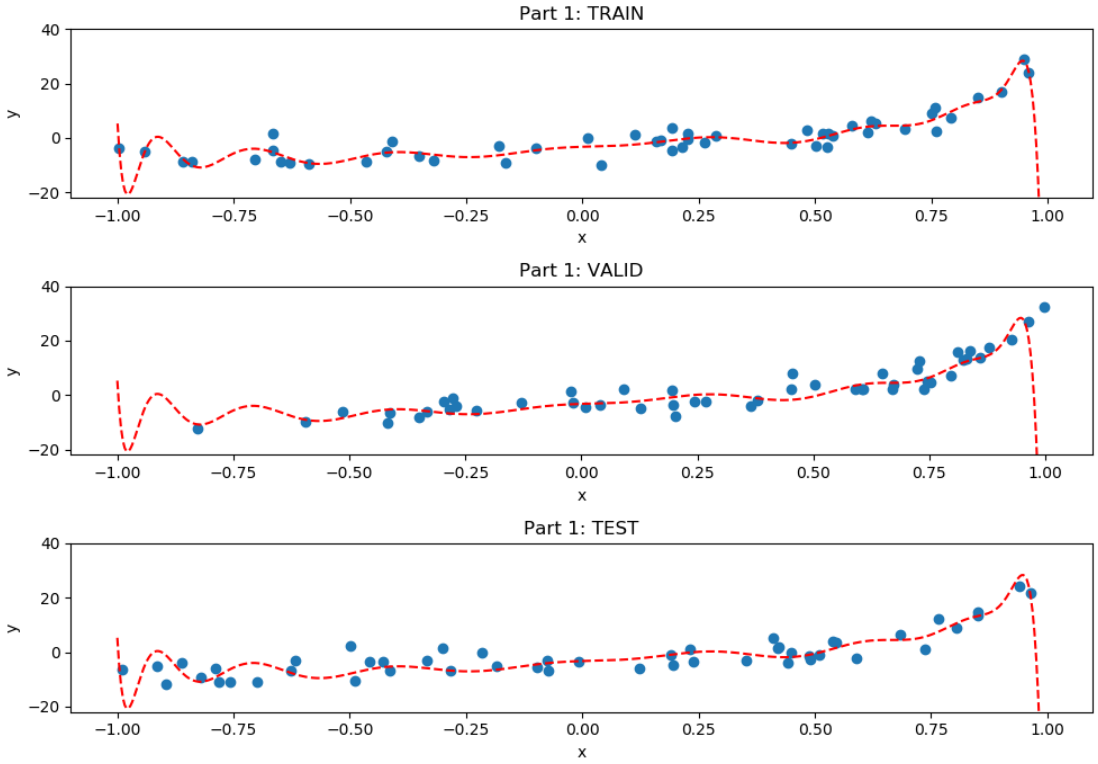


Figure 1.1: Fit of 20-degree polynomial without regularization

A closer look at the training plot shows oscillations in the fit (i.e. it is trying very hard to fit the training data). This observation, along with a high validation MSE of 458.64632 indicates that the 20-degree polynomial overfits the dataset. In other words, while the training MSE is low, the curve does not generalize well to the validation set.

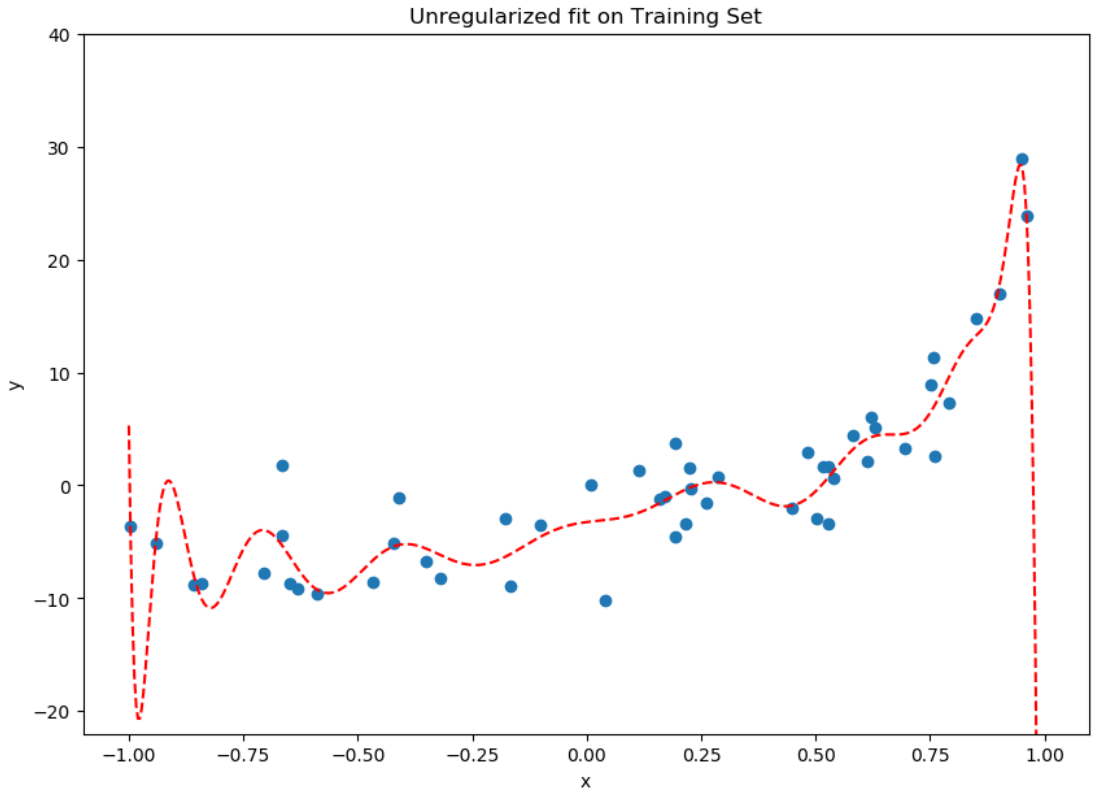


Figure 1.2: Fits training set well

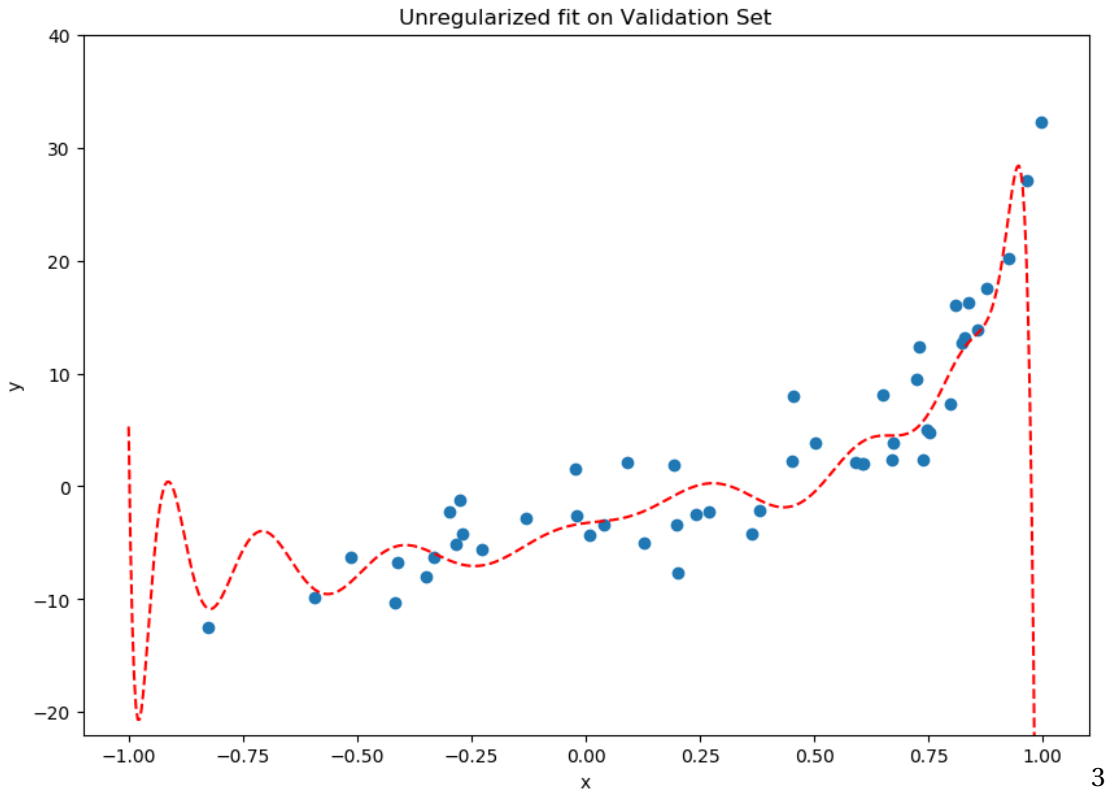




Figure 1.3: Fit does not generalize well to validation set

## 1.2 Adding L2 regularization to the model

Adding L2 regularization to our model requires changing the closed form solution by adding a $\lambda I$ term to $X^T X$ as follows:

$$w = (X^T X + \lambda I)^{-1} X^T y$$

Varying $\lambda$ from 0 to 1 with increments of .0001 yields the following results for the training and validation sets:

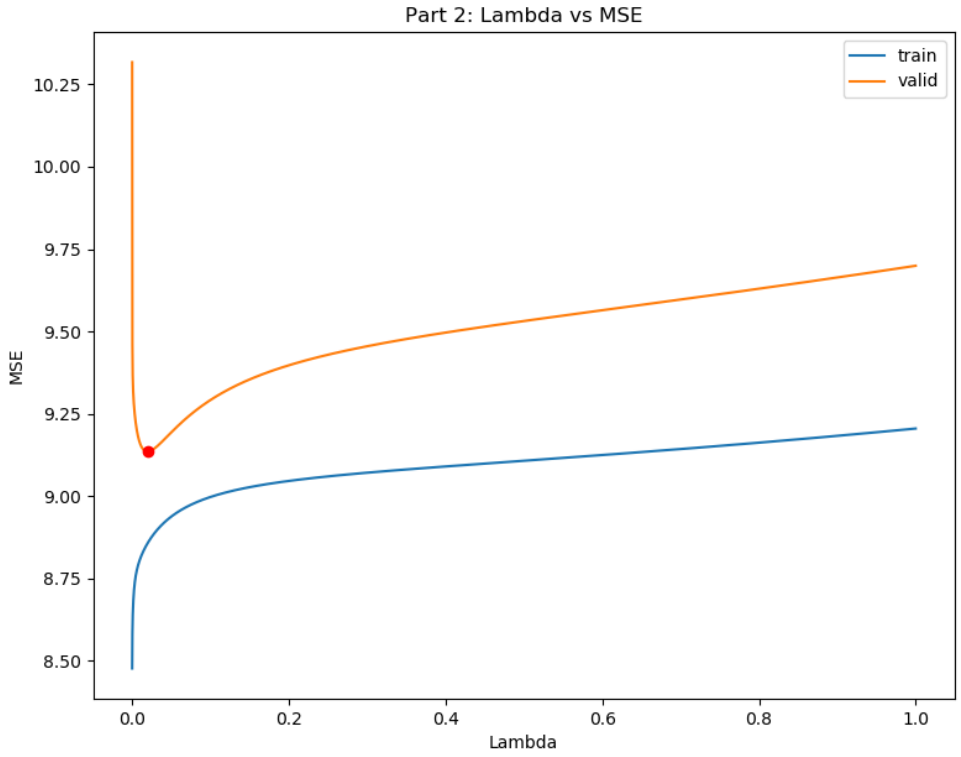


Figure 1.4: $\lambda$ vs MSE for training and validation sets

The optimum $\lambda$, based on the lowest MSE of the validation set is 0.0197. Retraining the model with this chosen lambda yields much better MSEs for the validation and test sets:

| **Set** | **MSE with** $\lambda = 0.0197$ |
|---|---|
| Train | 8.85645 |
| Valid | 9.13508 |
| Test | 10.73230 |

We can also visualize this regularized fit on the validation and test sets:

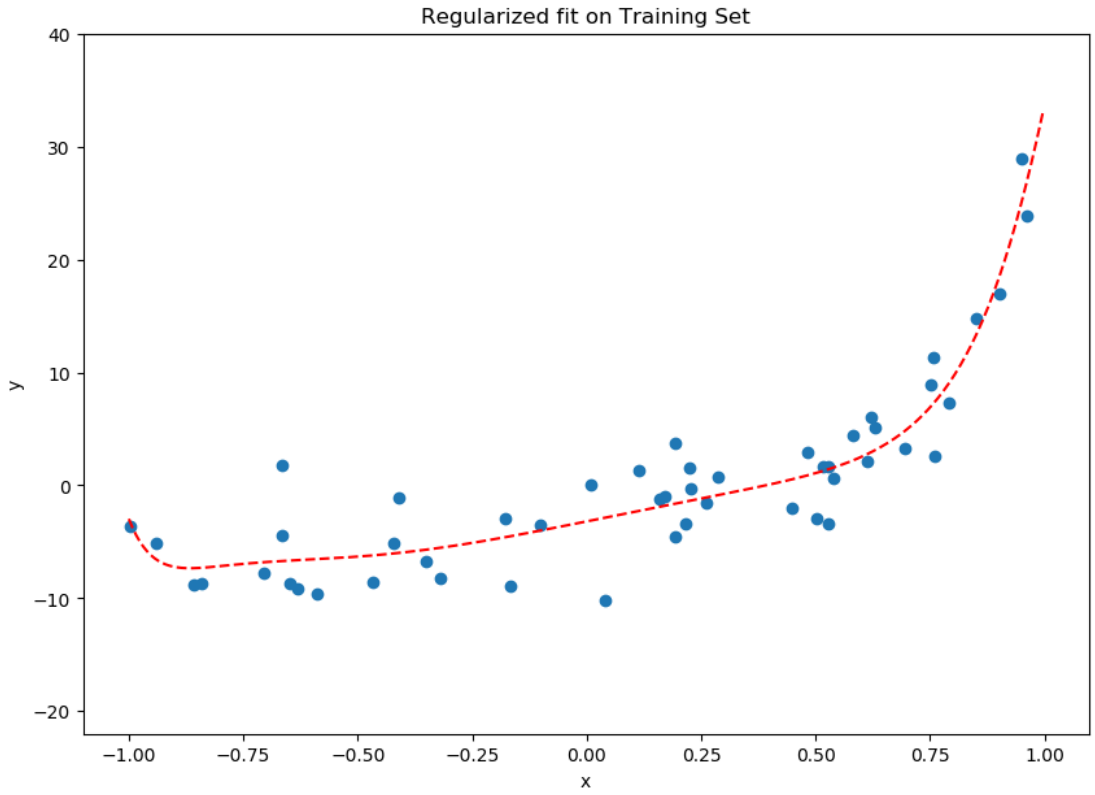


Figure 1.5: Fit of 20-degree polynomial on training set with optimal lambda

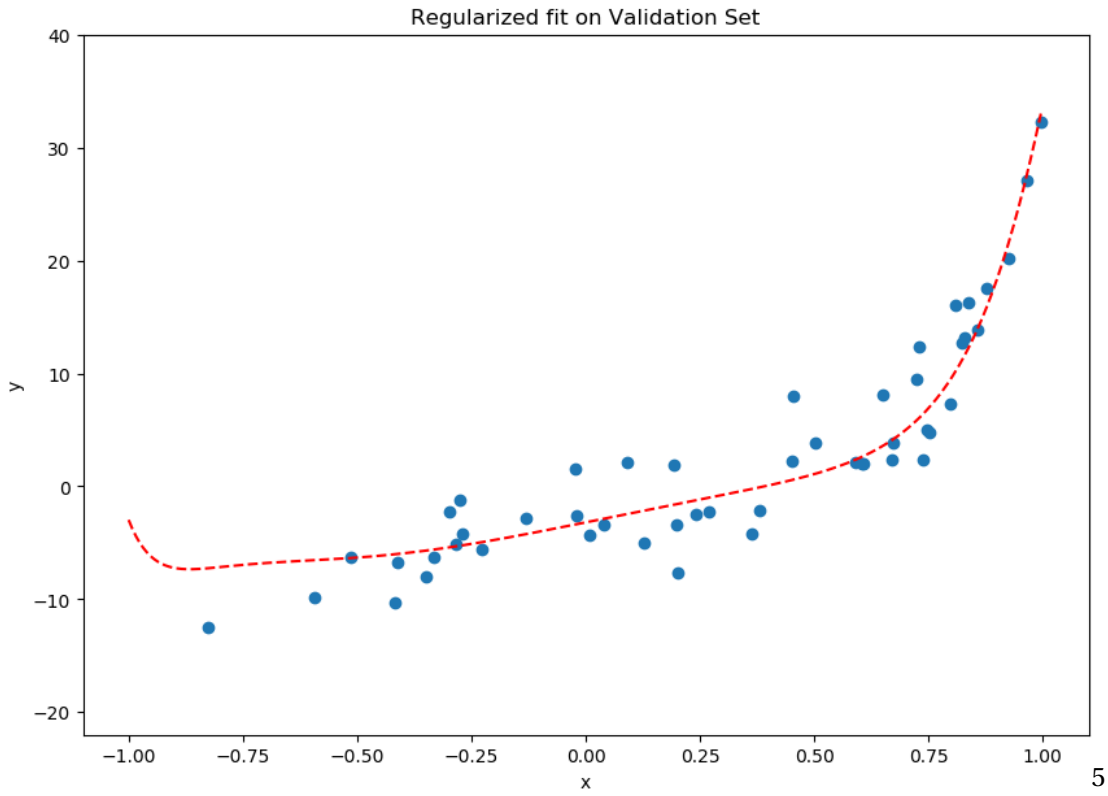




Figure 1.6: Fit of 20-degree polynomial on validation set with optimal lambda

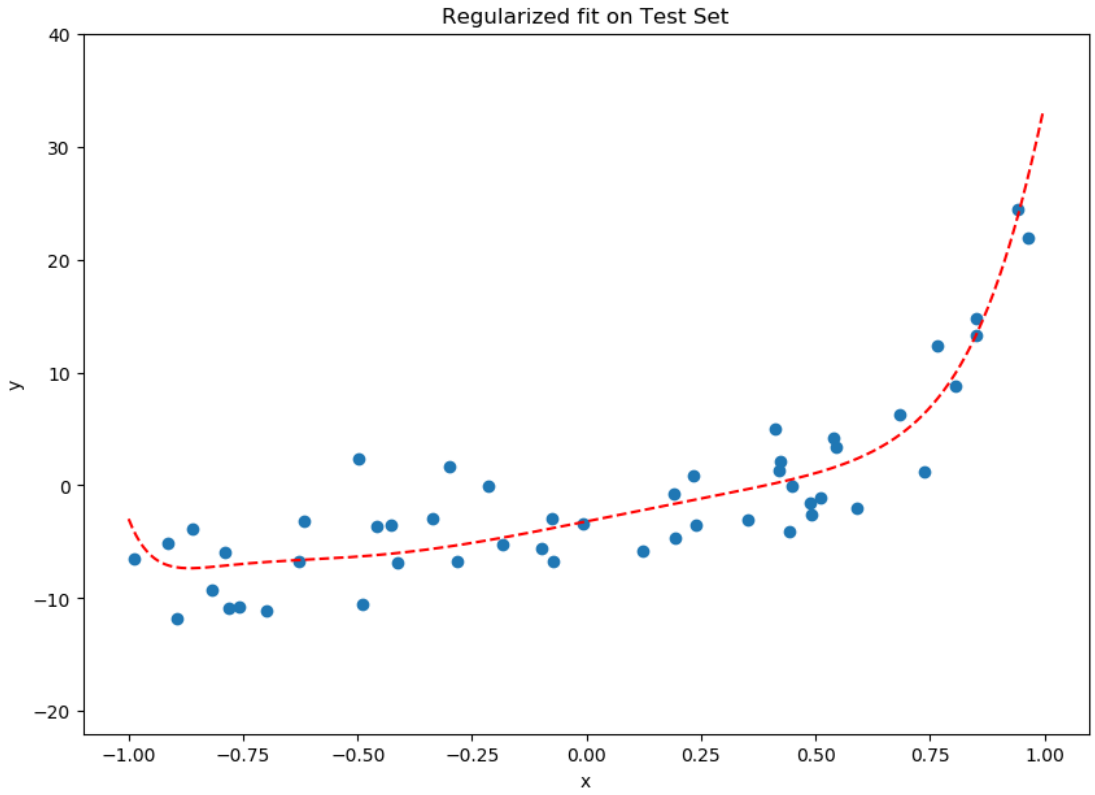


Figure 1.7: Fit of 20-degree polynomial on test set with optimal lambda

### 1.3 Estimation of degree of source polynomial

Observing the regularized fit of the model indicates two sharp bends (at the ends), with approximately 3 slight bends in between. This indicates that a $6^{th}$ degree polynomial is a good estimate of the source polynomial.

## 2 Gradient Descent For Regression

### 2.1 Fitting a linear regression model using Stochastic Gradient Descent

Dataset-2 provides us with a real-valued scalar as input and a real-valued scalar as output. Using Stochastic Gradient Descent (SGD) with a step size $\alpha$ of $10^{-6}$ yields the following learning curve (Note: since the weights are randomly initialized, this will not be the exact curve for every run of the code in the Jupyter Notebook):

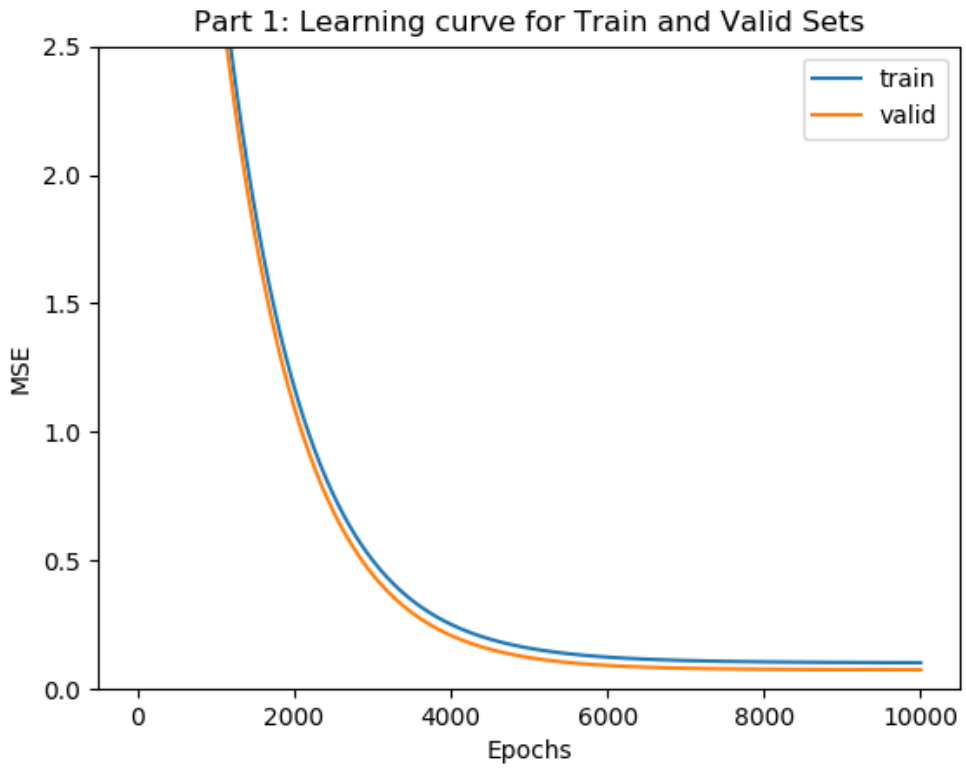


Figure 2.1: Epoch vs MSE at $\alpha = 10^{-6}$

| **Set** | **Final MSE at** $\alpha = 10^{-6}$ |
|---|---|
| Train | 0.10277 |
| Valid | 0.07558 |
| Test | 0.08120 |

## 2.2 Finding the best step size

The following step sizes were tested to determine which step size yielded the best results for the validation set: $10^{-6}, 5\times10^{-6}, 10^{-5}, 5\times10^{-5}, 10^{-4}, 5\times10^{-4}, 10^{-3}, 5\times10^{-3}, 10^{-2}, 5\times10^{-2}$. Based on these values and the initial random weights used in 2.1, the best $\alpha$ was found to be $5\times10^{-6}$.

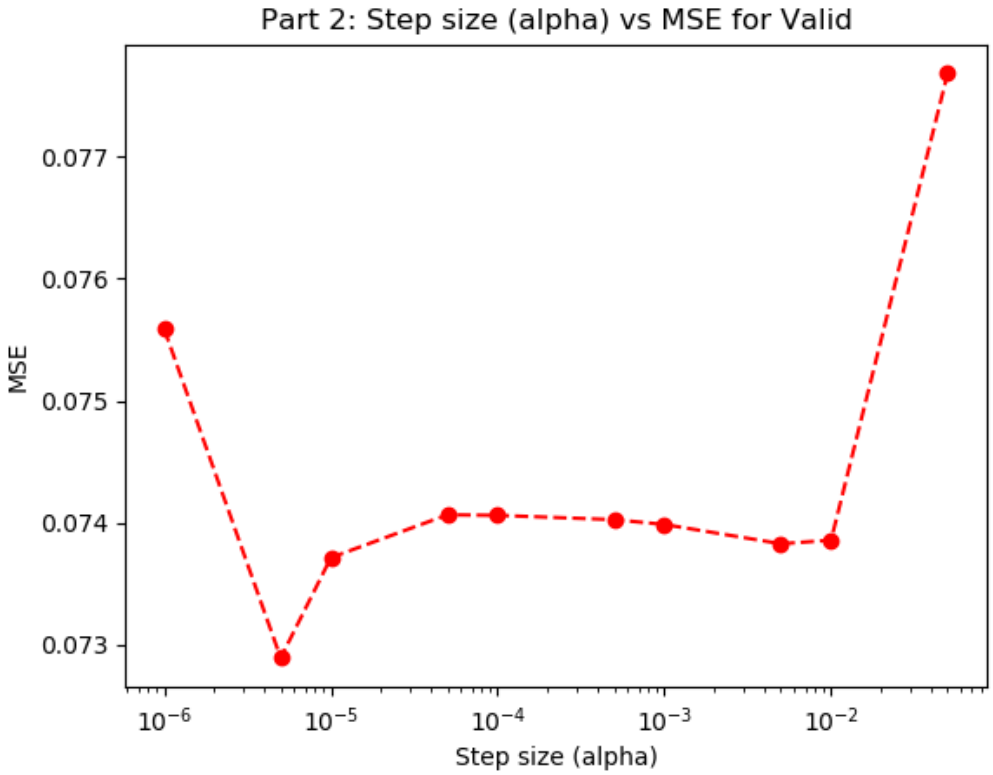


Figure 2.2: $\alpha$ vs MSE for the Validation set

The MSE for all three sets using the optimal $\alpha$ value of $5 \times 10^{-6}$ is:

| **Set** | **Final MSE at** $\alpha = 5 \times 10^{-6}$ |
|---|---|
| Train | 0.09606 |
| Valid | 0.07289 |
| Test | 0.07099 |

As we can see, the MSEs at $\alpha = 5 \times 10^{-6}$ are lower than at $\alpha = 10^{-6}$

## 2.3 Evolution of the regression fit during training

The following figures show the progression of SGD at 0, 800, 1600, 2400, 3200 and 4000 epochs (using $\alpha = 5 \times 10^{-6}$).

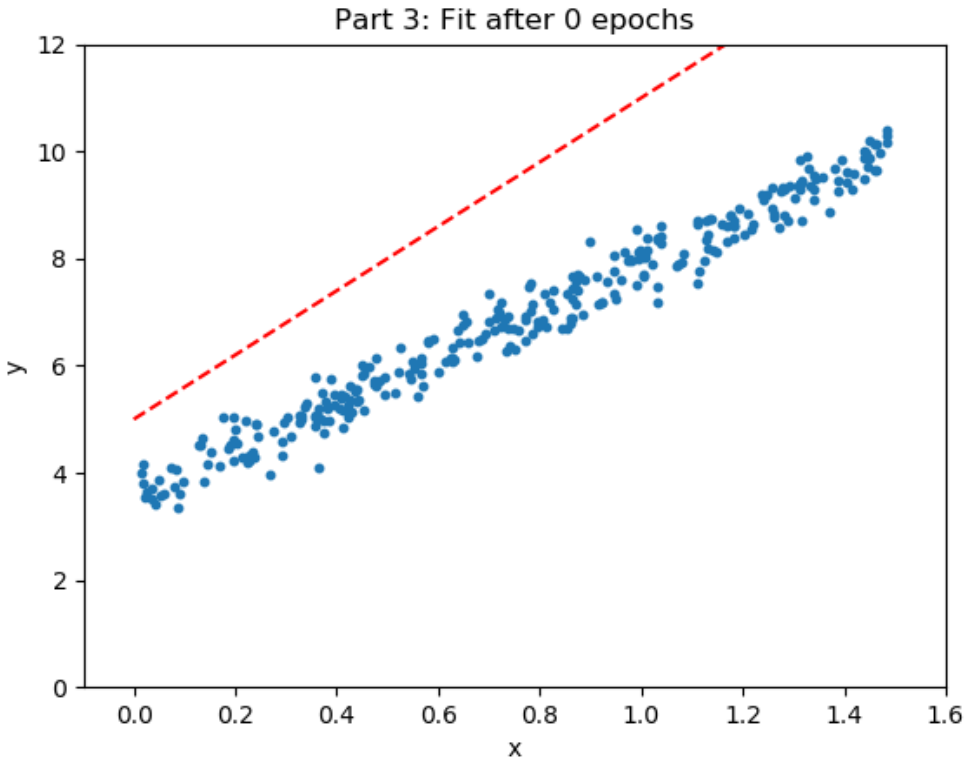


Figure 2.3: Fit with randomly initialized weights

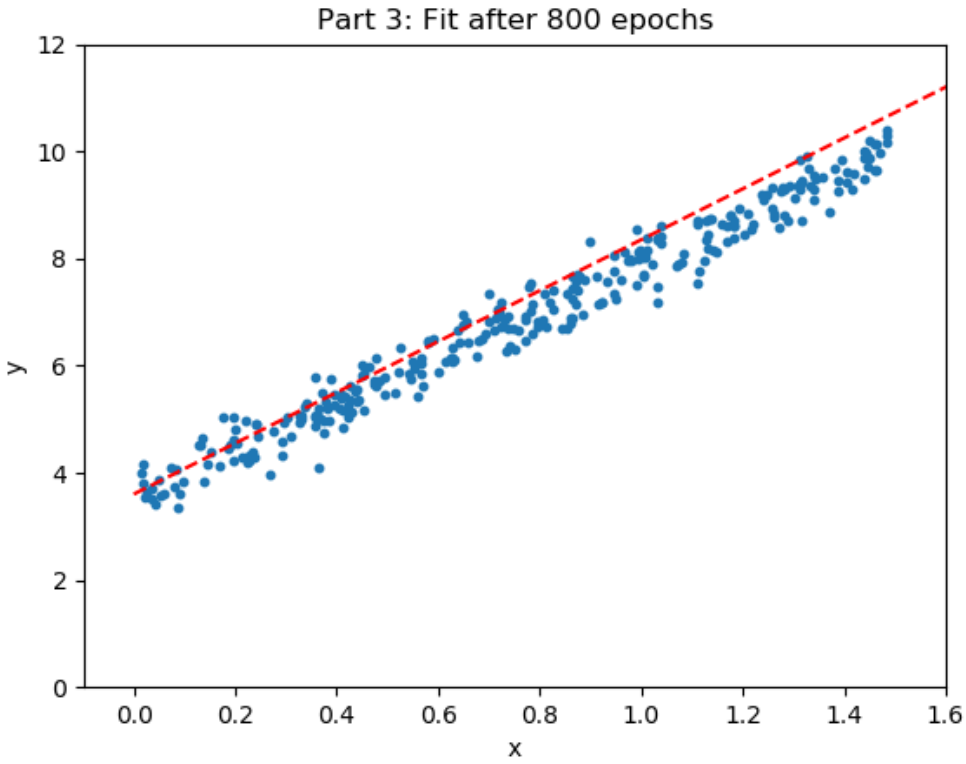




Figure 2.4: SGD makes large jump towards optimum weights

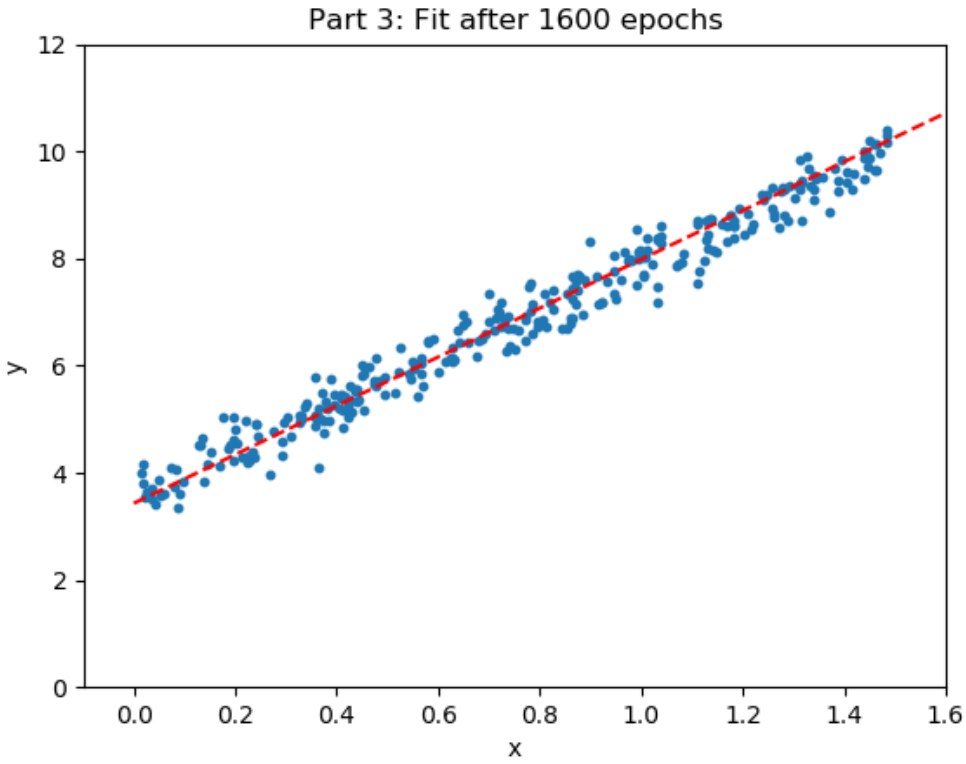


Figure 2.5: SGD starts taking smaller steps towards optimum

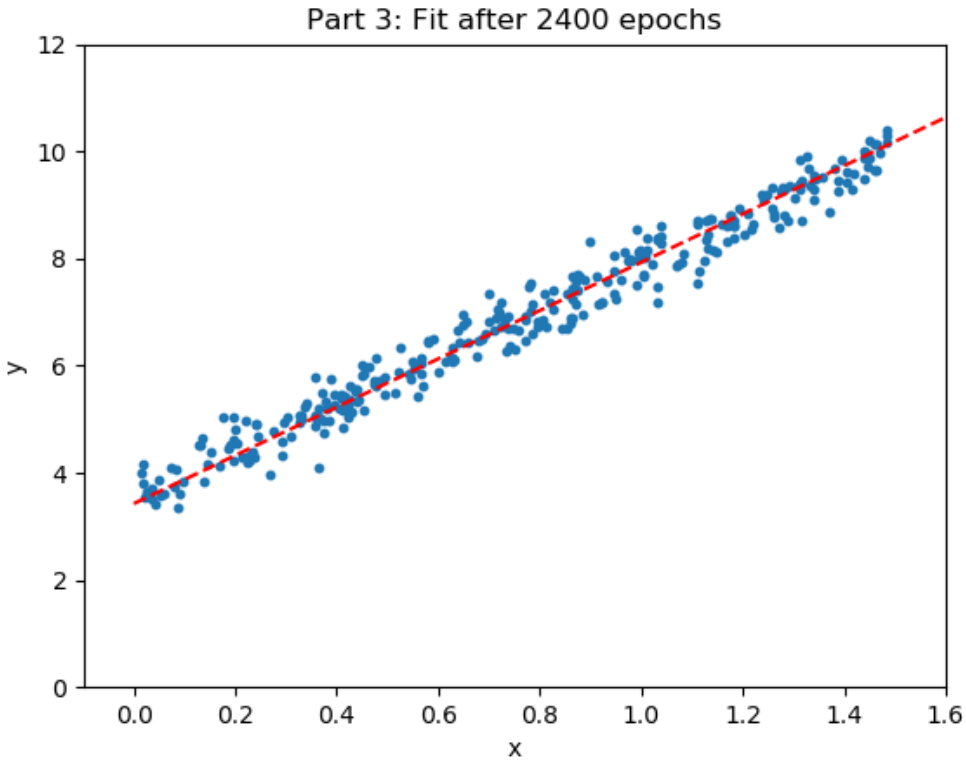




Figure 2.6: At 6000 epochs

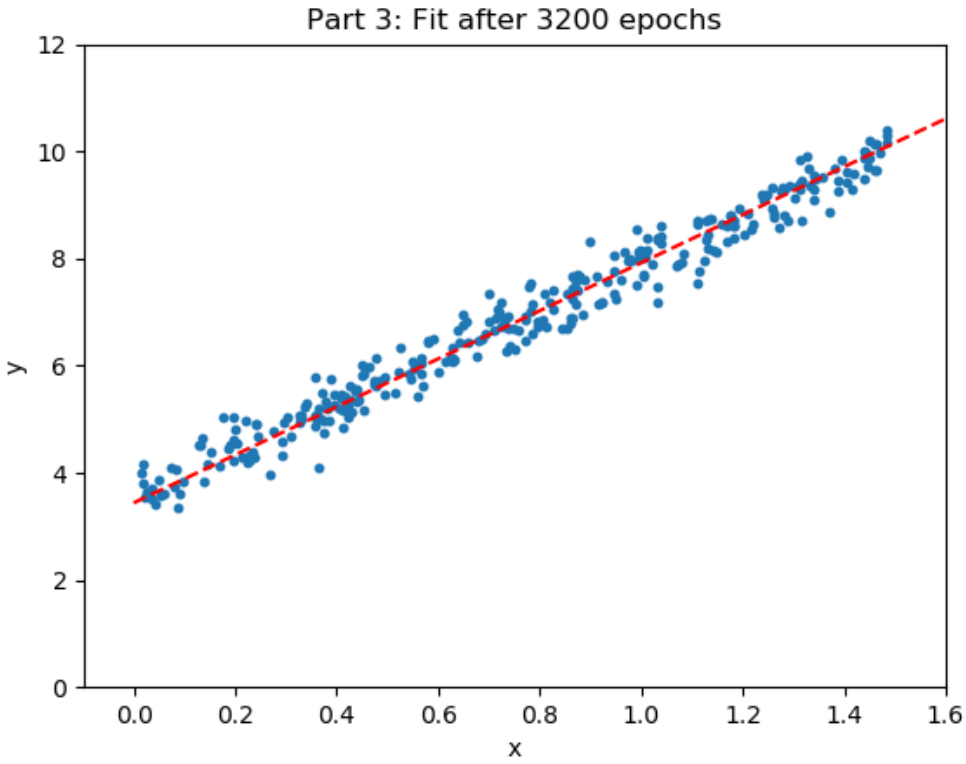


Figure 2.7: Near the end of training

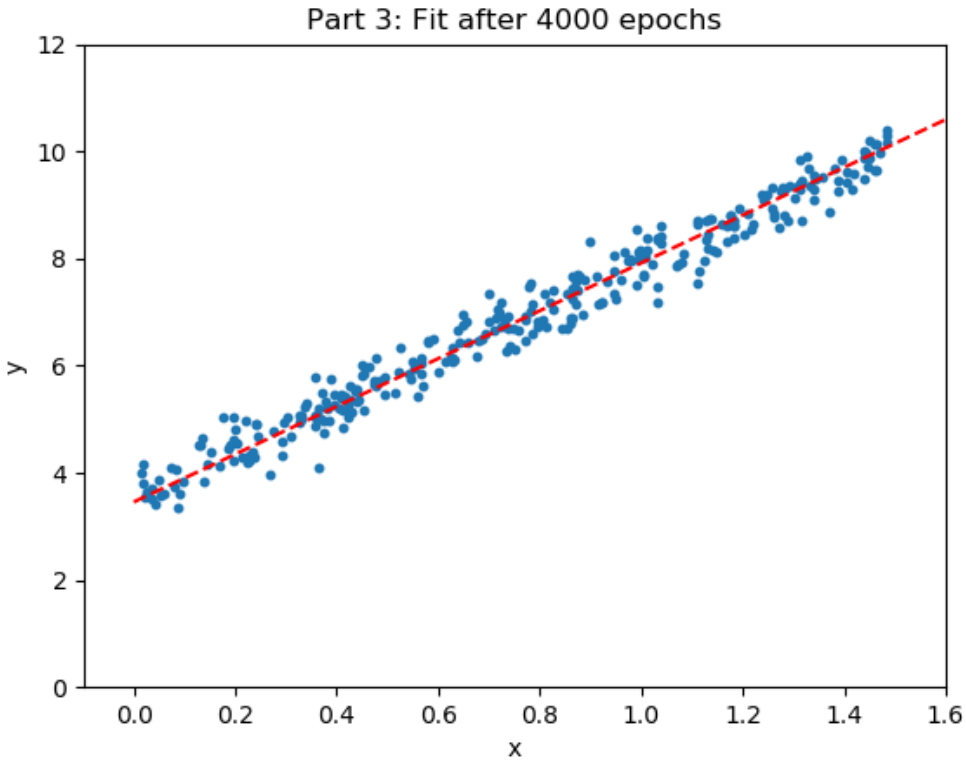




Figure 2.8: End of training

# 3 REAL-LIFE DATASET

## 3.1 FILL IN MISSING VALUES

The mean is a good filler for missing values if that particular attribute is interval or ratio data. Interval data is data that has a definite order and an exact difference, such as temperature. Ratio data is data with an absolutely defined zero, such as the Kelvin scale for temperature. In our case, the attributes are numerical data and thus is a good fit for the mean. The median and mode can also be used on such data.

The mean will not work categorical data since categories may not have an exact order (nominal data), and if they do, the distance between subsequent categories may not be definite (ordinal data). For nominal data, the mode is a good candidate for filling in missing values whereas for ordinal data, both the mode and median can be used, since there is notion of order and distance, even though the distance is not exactly known.

For this dataset, the mean of each feature (over all examples) was used to fill in missing values.

## 3.2 FITTING THE MODEL

The data contained 127 attributes, of which the first five were "not predictive" according to the source and so were omitted. Including the bias column added to the dataset, a total of 123 weights were used in the model. The data was shuffled and split into five 80-20 train-test splits. Using both the closed form solution and gradient descent separately on each of the splits, the following MSEs averaged over 5 splits were found:

Note: The MSE depends on the random shuffling of the data. This is one result, but all values are similar.

| **Algorithm** | **MSE on test** |
|---|---|
| Closed form solution | 0.11375 |
| Gradient descent | 0.45150 |

The weights for each fold calculated by both the closed form and gradient descent are attached in the csv files `weights_for_closed_form` and `weights_for_gradient_descent` respectively. Each column represents the weights calculated for each fold.

## 3.3 RIDGE REGRESSION

For this part, only the closed form solution was used, but the code in the Jupyter Notebook supports Ridge Regression for both the closed form and gradient descent.

Values of $\lambda$ between 0 and 5, with increments of 0.05 were tested to find the best $\lambda$. Again, since the examples are shuffled initially, this value will change from one execution of the code to another. For the shuffle order used for 3.2, the best $\lambda$ is 1.45, with an average MSE across

the five folds (on test set) of 0.01841. This is an improvement from the test MSEs in 3.2.

Here is a graph of $\lambda$ vs MSE:

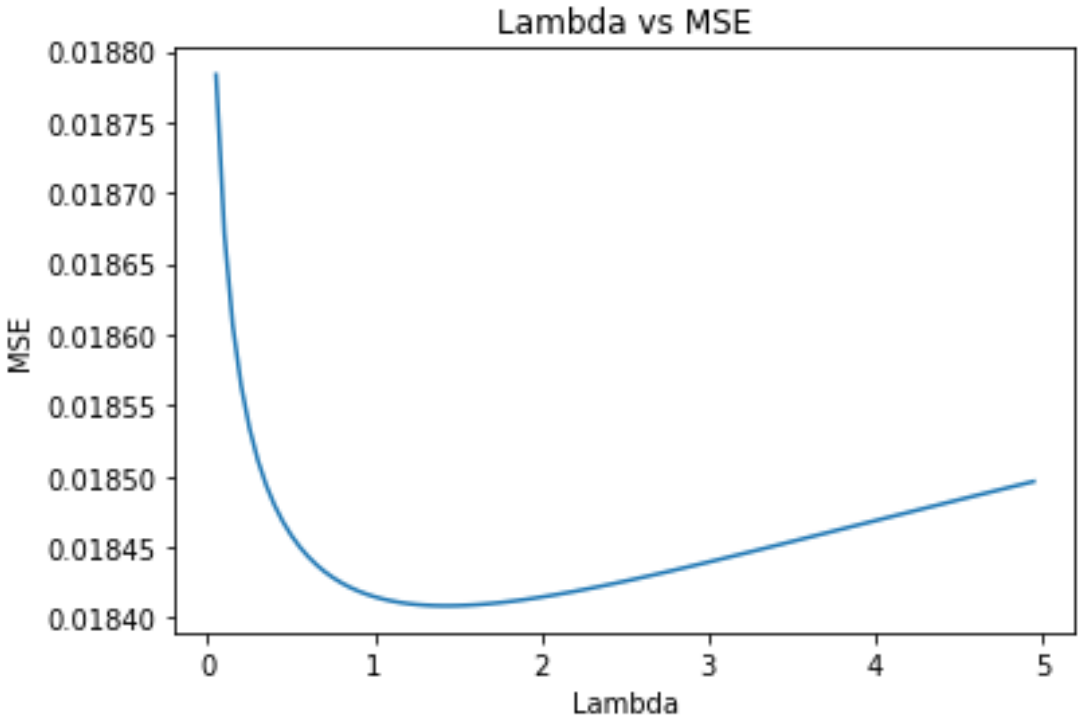


Figure 3.1: $\lambda$ vs average MSE on test set

Using the optimal weights produced by $\lambda = 1.45$, we can relate Ridge Regression and the importance of certain weights. Weights with magnitudes close to zero contribute less to the prediction than weights with larger magnitudes. By retraining the model with regularization using $\lambda = 1.45$ and sorting the weights by magnitude, we can observe which weights contribute the least (this was done for each of the five folds and then aggregated). Please see the Jupyter Notebook for implementation details. For this particular run, the number of features dropped by the algorithm was 51, producing better MSE of 0.01792:

| **No. of features** | **MSE on test, with** $\lambda = 1.45$ |
|---|---|
| All | 0.01841 |
| **72** | **0.01792** |