COMP 551 - Applied Machine Learning
Assignment 1

Tiffany Wang 260684152

January 28, 2018

ACKNOWLEDGEMENT

This assignment was fully completely by myself, Tiffany Wang. However, I discussed answers and methodologies with Frank Ye, John Wu and Nabil Chowdhury.

1 MODEL SELECTION

1.1 Fit a 20-degree polynomial to the data. Report the training and validation MSE (Mean- Square Error). Do not use any regularization. Visualize the fit. Comment about the quality of the fit.

Linear regression is applied to solve this problem. The first step is to construct a matrix of the input features, which is the following:

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & ... & x_1^20 \\ 1 & x_2 & x_2^2 & ... & x_2^20 \\ . & & . & & . \\ . & & & . & . \\ 1 & x_n & x_n^2 & ... & x_n^20 \end{bmatrix}$$

The weight parameters of the model are then trained using : $W = (X^T X)^{-1} X^T y$
where X and y are the training inputs and outputs respectively. Once trained, the weights are utilized to predict the outputs of the validation set. $\hat{y}_{valid} = W^T * X_{valid}$

The Mean Square Error (MSE) of the training model is found to be 7.15 for the training set and 457.99 for the validation set.
We can remark that the predicted training output is more accurate, lower MSE, than that of the validation set. This is expected, since the weights parameters of the model are trained from the training set, so they have already *seen* the data, whereas the validation set is new.

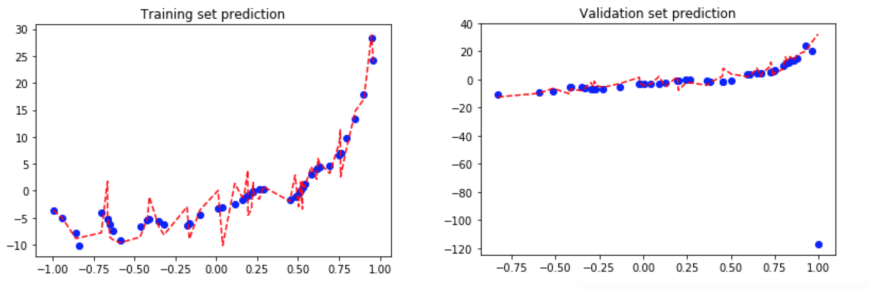

Figure 1: Comparison of predicted output(blue) and true dataset output (red) of training and validation sets

1.2 Now add L2 regularization to your model. Vary the value of λ from 0 to 1. For different values of λ, plot the training MSE and the validation MSE. Pick the best value of and report the test performance for the corresponding model. Also visualize the fit for the chosen model. Comment about the quality of the fit

In this section, the weights parameters formula is modified to: $W = (X^T X - \lambda \cdot I)^{-1} X^T y$
The regularization helps increase the accuracy of the model, and thus decrease the MSE.

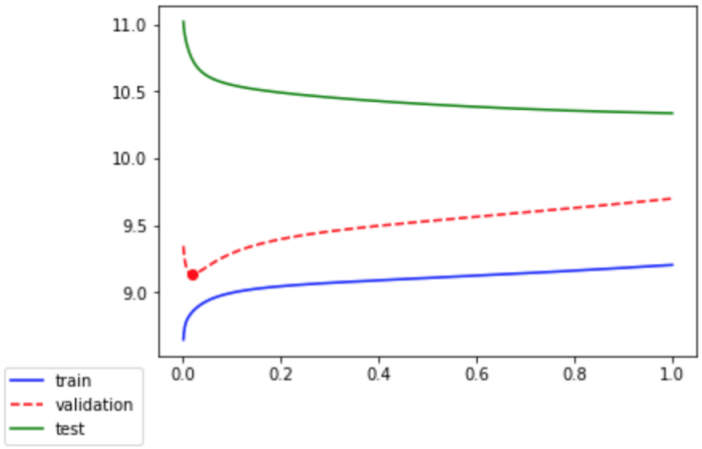

Figure 2: Mean Square Error on training, validation and test sets with varying regularization λ value

The best regularization lambda is 0.02, for which the MSE of the training, validation and testing sets are respectively

smallest train MSE:	8.85765667763104
smallest valid MSE:	9.135098784694554
smallest test MSE:	10.730218400927397

The regularization introduces a bias in the training model in order to minimize the error function. The prediction of the validation set fits better with the true validation set output once regularization is added. The MSE dropped from 457.99 to 9.13. However, we notice that the regularization has shifted the bias away from the training set. The MSE increased (relatively minimally).

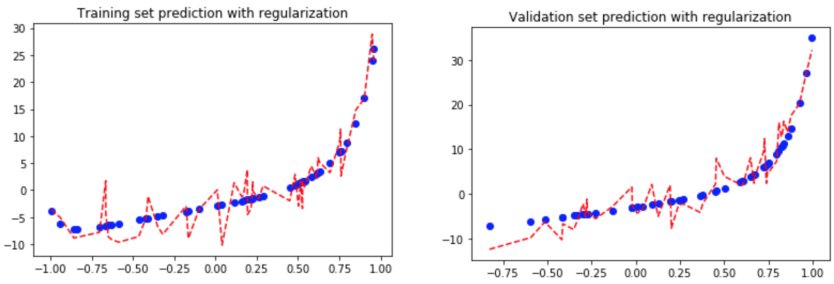

Figure 3: Comparison of predicted output(blue) and true dataset output (red) of training and validation sets

1.3 What do you think is the degree of the source polynomial? Can you infer that from the visualization produced in the previous question?

We introduced a 20-degree polynomial into the dataset. However, as the training set error is not zero, and the validation set error is large, we can conclude that the the polynomial is larger than 20-degree. Further testing is needed to find out the exact degree of the source polynomial.
One solution is to iteratively increase the polynomial degree and train the model, until the training MSE reaches 0, say at degree = N. The source polynomial would then be of degree N-1.

2 Gradient Descent for Regression

2.1 Fit a linear regression model to this dataset by using stochastic gradient descent. You will do online-SGD (with one example at a time). Use the step size of 1e-6. Compute the MSE on validation set for every epoch. Plot the learning curve i.e. training and validation MSE for every epoch.

This is a single input / single output mapping model. The stochastic gradient descent linear regression model is calculated using.

$$w_0' = w_0 + \alpha(\hat{y}^{(i)} - y^{(i)})$$
$$w_1' = w_1 + \alpha(\hat{y}^{(i)} - y^{(i)}) \cdot x^{(i)}$$

where α is predefined to be 10^{-6}.
To decrease the bias in the training, the first w_0 and w_1 are randomly generated to be between 0 and 10. Then, they are iteratively computed until the $|w' - w| < 10^{-4}$. The best fitted parameters are:

$$(w_0, w_1) = (3.549, 4.352)$$

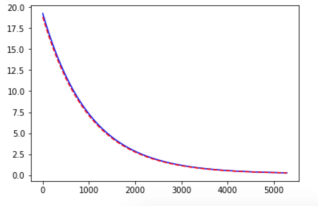

Figure 4: Trend of variation of training (blue) and validation (red) MSE vs. epoch

From the training curves, we can see that MSE of both validation and training set decrease and converges as the number of epochs iterations increase.

2.2 Try different step sizes and choose the best step size by using validation data. Report the test MSE of the chosen model.

The goal of this part is find the best step λ for this training model. However, it is overkill to train the models until convergence of the parameters, so I trained the parameters for a set number of iterations of 1000. The model is validated by running it on the validation set. We can assume that the step λ rendering the lowest MSE is the best step for this model.

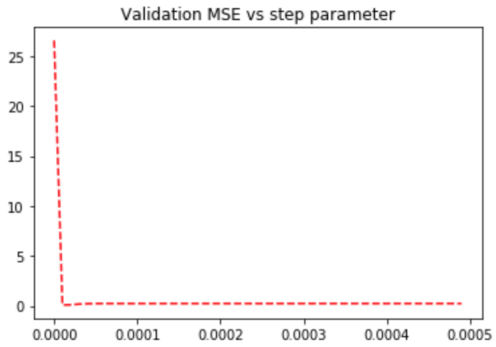

Figure 5: MSE of validation vs. lambda step

50 different steps are tested from 10^{-6} to $2 \cdot 10^{-3}$, with intervals of $2 \cdot 10^{-5}$. The best step is 0.00001, for which the MSE is 0.0975.

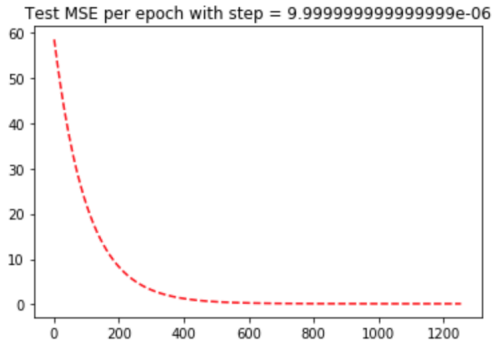

Figure 6: Test MSE of the training model with $\lambda = 1 \cdot 10^{-5}$ vs epoch

The final test MSE of this model is 0.1352.

2.3 Visualize the fit for every epoch and report 5 visualizations which shows how the regression fit evolves during the training process.

As the model trains, and the parameters w_0, w_1 converge, the prediction's fit to the true output becomes more accurate. At this stage, the model is trained with the step found in 2.2 (0.00002), and is trained until convergence of parameters. These visualizations were selected because they were empirically observed to be most representative of this model's learning process.

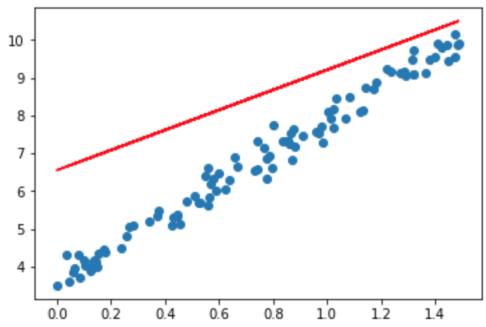

Figure 7: Training curve of the model: fit of the prediction(red) on the true output (blue) after 1 epoch

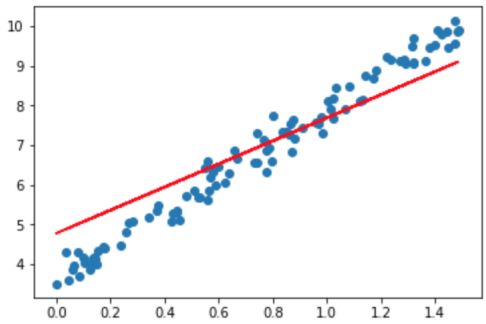

Figure 8: Training curve of the model: fit of the prediction(red) on the true output (blue) after 20% of epochs

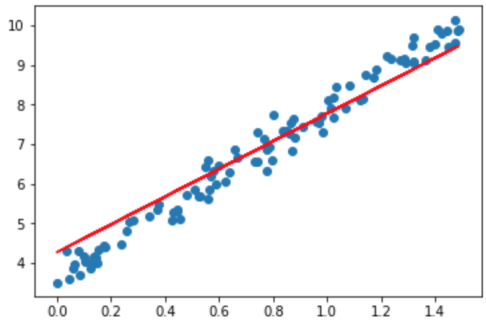

Figure 9: Training curve of the model: fit of the prediction(red) on the true output (blue) after 40% of epochs

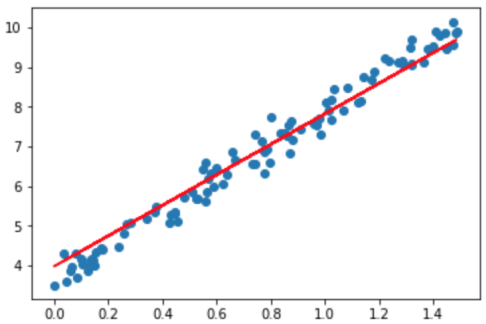

Figure 10: Training curve of the model: fit of the prediction(red) on the true output (blue) after 60% of epochs

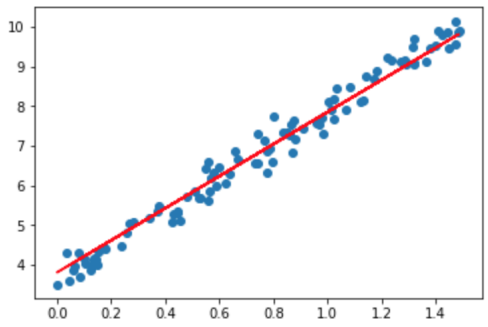

Figure 11: Training curve of the model: fit of the prediction(red) on the true output (blue) at convergence

3 Real Life Dataset

3.1 This is a real-life data set and as such would not have the nice properties that we expect. Your first job is to make this dataset usable, by filling in all the missing values. Use the sample mean of each column to fill in the missing attribute. Is this is a good choice? What else might you use? If you have a better method, describe it, and you may use it for filling in the missing data. Turn in the completed data set.

The dataset comes with 127 features. However, the first 5 columns (state, county, community, communityname, fold) are irrelevant and do not contribute to the prediction of the violent crimes. Therefore, they are dropped, rendering a input with 122 features only.

There are several ways of filling in missing data. The common ones are replacement by mean and replacement by median. In our case, the values have been normalized using an unsupervised equal-interval binning method. Therefore, we can assume that the data has no great outliers. The replacement by average method will be applied to fill the missing data.

3.2 Fit the above data using linear regression. Report the 5-fold cross-validation error: The MSE of the best fit achieved on test data, averaged over 5 different 80-20 splits, along with the parameters learned for each of the five models.

We will employ the least square classification approach to approximate the missing violent crimes outputs. Similar to part 1, the weights are calculated using $W = (X^T X)^{-1} X^T y$

The best average MSE achieved on the test data is 0.6015. The weights parameters obtained for the five datasets can be found in $3\_2\_weights < num > \_noReg.csv$. This MSE is subject to change since the train / test sets are randomly generated.

3.3 Use Ridge-regression on the above data. Repeat the experiment for different values of λ. Report the MSE for each value, on test data, averaged over 5 different 80-20 splits, along with the parameters learned. Which value of gives the best fit? Is it possible to use the information you obtained during this experiment for feature selection? If so, what is the best fit you achieve with a reduced set of features?

The Ridge-regression, also known as L2 regression, helps reducing the MSE error. This is the same method applied in 1.2.

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

The best λ is 1.76. The parameters learned can be found in $3\_3\_weights < num > \_reg.csv$.

$$\text{MSE} = \begin{matrix} 0.01919 \\ 0.01833 \\ 0.01945 \\ 0.01676 \\ 0.02199 \end{matrix}$$

The overall MSE is 0.01914. Once again, these values depend on the randomly generated datasets, and may differ from run to run.

When studying the learned weights parameters of the model, we notice that certain parameters have a larger weight than others, and thus have a larger 'effect' in the model prediction. We can select the top features based on the top weighed features.

In this case, the model is retrained (without regularization) with the top 20 features, and the resulting overall MSE is 0.01958 which is 187.359% lower than the MSE of the model without feature selection.