Home Work Assignment 1: Regression

Fatemeh Lotfi
University of Colorado, Colorado Springs
Colorado Springs, CO 80918
flotfi@uccs.edu

Introduction

Regression is a method of modeling that includes estimating a numerical value from a collection of inputs. The standard regression algorithm assumes a linear relationship between the input values and the target variable. Adding penalties to the loss function will shrink the parameters toward the zero, especially that parameters that have lower impact on the prediction. One of the popular regularization models is Ridge Regression. Ridge Regression is a type of regularized linear regression with an L2 penalty that allows the coefficients for input variables that don't add much to the prediction task to diminish. The other type of regularization models is Lasso and Elastic Net regression.
In this paper, we learn how to develop and evaluate linear regression models in Python.

1. Least Squared Linear Regression

Generally speaking, in linear regression problems the goal is to fit a hyperplane to a set of data points. In this section, we consider a dataset D with L examples, N independent features X and a target value or label y that is dependent on X. In this case, we can model y like

$$f_\theta(x) = \beta + \theta_1 x_1 + \cdots + \theta_N x_N, \tag{1}$$

As we said, our goal is to find a line or plane that acceptably follows the data. To do this, we should minimize the error between predicted value $f_\theta(x)$ and actual value in y. The error can be written as

$$\begin{aligned} E(\theta) &= \sum_{i=1}^{L} y^i - f_\theta(x^i) \\ &= \sum_{i=1}^{L} y^i - (\beta + \theta_1 x_1^i + \cdots + \theta_N x_N^i). \end{aligned} \tag{2}$$

Therefore, our objective function can be written as

$$\operatorname*{argmin}_{\boldsymbol{\theta},\beta} \quad E(\boldsymbol{\theta}), \tag{3}$$

$$\text{s.t.,} \quad f_\theta(x) = \beta + \theta_1 x_1 + \cdots + \theta_N x_N. \tag{4}$$

There are many methods to perform the optimization problem (3). In the rest of the report, we are going to investigate some of the more famous methods.

2. Implementation of Least Squared Linear Regression with SGD algorithm

One of the popular methods in solving the optimization problem (3) is Stochastic Gradient Descent (SGD). This method is a modified version of the famous Gradient Descent (GD) method.
To implement the SGD, we used a Python language. The code consists of five sections, including Reading the dataset, feature scaling, define the function for SGD weight update, training function, and testing function. In the following, we will briefly describe each section.

- Reading the dataset : in this section we read the dataset by *pd.read-excel* and *pd.read-csv* from Panda library of Python. Then, we split the dataset based on a fraction factor to the training set and test set. The input of this function is the dataset, and the output is both training and test sets.
- Feature scaling : In this part of the code, we want to normalize the range of independent variables. The input of this function is raw features, and the output is the scaled features.
- Define function for SGD : In this section, we implement the process of the SGD method. The input of this function is features and labels matrices, predefined learning rate, and a predicted value. The main loop of this function updates the weights and the bias for each iteration and measures the error vector and updates the new weights and biases with respect to the learning rate and error vector until the process converges. The output of the function is weights and biases.
- Training function : In this section, the model would be learned by training data and fits the appropriate weights and biases to data.
- Testing function : In the latest part of the code, we evaluate the model's predictive power with Mean Square Error (MSE) and other evaluation metrics to show the achieved model's performance.

3. Test of Least Squared Linear Regression using SGD

In this section, we derive a linear regression solution using the stochastic gradient descent algorithm with a learning rate

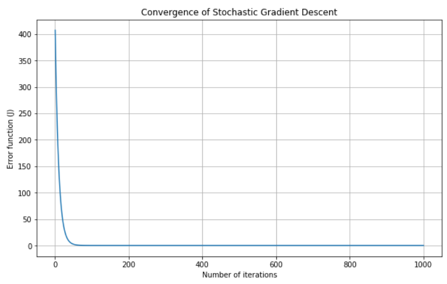

Figure 1: The error function in each iteration of SGD method of Dataset 1

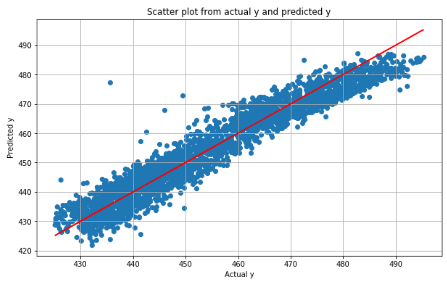

Figure 2: Prediction vs actual labels of SGD method of Dataset 1

of α. In the following, we show the results for two datasets and evaluate the prediction performance.

Dataset 1 : Combined Cycle Power Plant

This dataset contains 9568 data points, four features, and one label. We derive the SGD algorithm with learning rate $\alpha = 0.1$ and maximum iteration 1000. In the following, we present the results.

Fig. 1 depicts the measured error in each iteration of the algorithm. As it turns out, the algorithm is fully converged..

Furthermore, Fig. 2 shows the prediction values followed the actual values to a good extent. In the following, we evaluate the performance of the SGD method for different evaluation metrics. These metrics that we used consist of Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Coefficient of Determination (R^2), Adjusted Coefficient of Determination (AR^2), and F-statistics. Below is a brief description of each.

- Mean Squared Error (MSE):
  One of the popular metrics for the evaluation of regression models is Mean Squared Error (MSE). This metric can be calculated as

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\big(y - f_\theta(x)\big)^2, \tag{5}$$

  where $f_\theta(x)$ is the predicted values for labels.

- Mean Absolute Error (MAE):
  Other powerful evaluation metric is Mean Absolute Error (MAE) defined as

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\big|y^i - f_\theta(x^i)\big|, \tag{6}$$

- Root Mean Squared Error (RMSE):
  This metric is square root of MSE defined as

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\big(y^i - f_\theta(x^i)\big)^2}, \tag{7}$$

- Coefficient of Determination (R^2):
  The other powerful metric to determine how well a regression model fits the data is the R-square metric. This metric defined as follow:

$$\begin{aligned} R^2 =& 1 - \frac{SSE}{SST}, \\ =& 1 - \frac{\sum_{i=1}^{N}\big(y^i - f_\theta(x^i)\big)^2}{\sum_{i=1}^{N}\big(y^i - \bar{y}\big)^2} \end{aligned} \tag{8}$$

  where $\bar{y}$ is a value of predicted label in any given features.

- F-statistic :
  Another metric that show performance of regression models is F-statistic that define as

$$\begin{aligned} F_{statistic} =& \frac{MSR}{MSE}, \\ =& \frac{\frac{1}{N_p}\sum_{i=1}^{N}\big(y^i - \bar{y}\big)^2}{MSE} \end{aligned} \tag{9}$$

  where N_p is number of the independent features.

Table 1 represents the measured values for each metric.

Finally, we find the learning parameters $\theta =$

Table 1: Evaluation metrics for prediction function of Dataset 1

Mean Squared Error (MSE)	21.17
Mean Absolute Error (MAE)	3.71
Root Mean Squared Error (RMSE)	4.60
Coefficient of Determination (R^2)	0.92
Adjusted Coefficient of Determination (R^2)	0.92
F-statistics	4240.027

$\{-13.29, -3.89, 0.59, -2.24\}$ and $\beta = 454.37$. Hence, the predicted model for the linear regression problem can be written as

$$\begin{aligned} f(x) =& -13.29x_1 - 3.89x_2 + 0.59x_3 \\ & - 2.24x_4 + 454.37. \end{aligned} \tag{10}$$

Dataset 2 : Physicochemical Properties of Protein Tertiary Structure Data Set

This data set contains 45730 data points, nine features, and one label. We derive the SGD algorithm with learning rate $\alpha = 0.06$ and maximum iteration 1000. In the following, we present the results.

Fig. 3 represents the measured error in each iteration of the algorithm. As it turns out, the algorithm is fully converged.

Furthermore, Fig. 4 reveals that the prediction value does

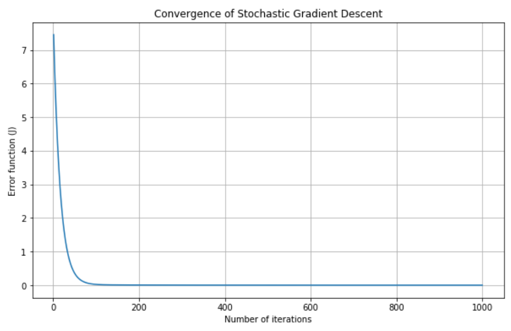

Figure 3: The error function in each iteration of SGD method of Dataset 2

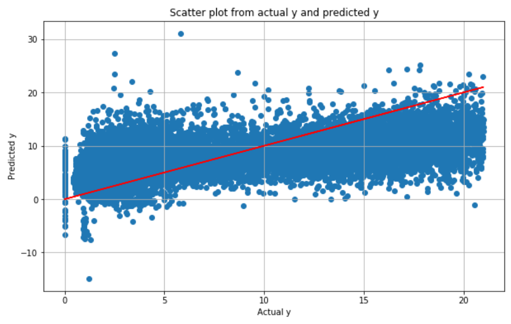

Figure 4: Prediction vs actual labels of SGD method of Dataset 2

not follow the actual values. The reason is that the linear regression can't follow all the data. In the following, for dataset one, we evaluate the SGD method's performance with different evaluation metrics. Table 2 represents the measured values for each metric.

Finally, we find the hyper parameters $\theta = \{0.81, 1.97, 1.61, -6.02, 0.96, -0.67, 0.44, 1.05, -1.49\}$ and $\beta = 7.93$. Hence, the predicted model for the linear regression problem can be written as

Table 2: Evaluation metrics for prediction function of Dataset 2

Mean Squared Error (MSE)	27.48
Mean Absolute Error (MAE)	4.36
Root Mean Squared Error (RMSE)	5.24
Coefficient of Determination (R^2)	0.26
Adjusted Coefficient of Determination (R^2)	0.26
F-statistics	805.96

$$\begin{aligned} f(x) &= 0.81x_1 + 1.97x_2 + 1.61x_3 \\ &- 6.02x_4 + 0.96x_5 - 0.67x_6 \\ &+ 0.44x_7 + 1.05x_8 - 1.49x_9 \\ &+ 454.37. \end{aligned} \tag{11}$$

The effect of learning rate in SGD

We solve the problem for different values of α and then measure the MSE for them. In this way, we can show the importance of proper learning rate selection. Indeed, it will demonstrate how α contributes to the SGD algorithm. Table 3 shows the effect of selecting learning rate on MSE.

Table 3: The effect of different values of α on MSE in SGD algorithm for dataset 1

Learning rate (α)	MSE
0.001	286.77
0.01	21.89
0.1	23.6
0.2	53.7
0.3	98.3
0.35	43880.2

The effect of preprocessing (feature scaling) in SGD

One of the necessary steps in SGD algorithms is preprocessing of data. In some datasets, the range of features is very different and, this issue harms the performance of the SGD algorithm. In this section, we want to compare the performance of the SGD algorithm. First, we perform the algorithm without feature scaling and then with adding feature scaling. Table 4 shows the results.

Table 4: The effect of feature scaling on MSE in SGD algorithm for dataset 1

MSE with feature scaling	20.15
MSE without feature scaling	2.8e+31!

4. Implement and test of Ridge regression

Ridge regression is a kind of regularized regression. This method seeks to alleviate the consequences of multicollinearity, poorly conditioned equations, and overfitting. Ridge regression adds an extra component to the loss function. Therefore, its objective function is:

$$L = \sum_{i=1}^{N} \left(y^i - f_\theta(x^i) \right)^2 + \lambda \sum_{j=1}^{N} \theta_j^2, \tag{12}$$

where the first part is the least squared error and the second part is shrinkage penalty.

In this section, we will implement and examine the Ridge regression problem for various data sets, namely, Combined Cycle Power Plant and Physicochemical Properties of ProteinTertiary Structure. In what follows, we solve the Ridge regression problem for each data set and highlight our main observations accordingly.

Dataset 1 : Combined Cycle Power Plant

This dataset contains 9568 data points, four features, and one label. In this section we derive the Ridge regression problem with SGD algorithm for different amount of

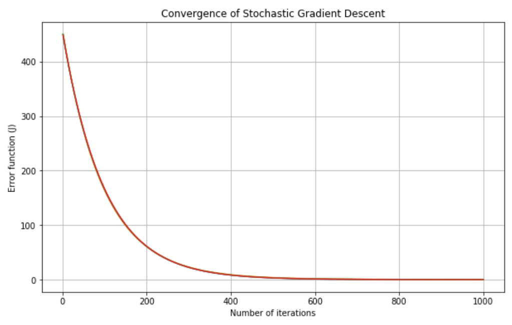

Figure 5: The error function in each iteration of Ridge regression with SGD algorithm of Dataset 1

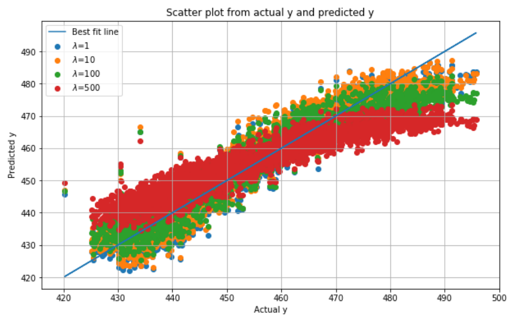

Figure 6: Prediction vs actual labels of Ridge regression with SGD algorithm of Dataset 1

$\lambda = \{1, 10, 100, 500\}$. Fig. 5 represents the measured error in each iteration of algorithm for $\alpha = 0.01$. As it turns out, the algorithm is fully converged.

Also, Fig. 6 shows the prediction value for different amount of λ. As we can see, for some λ values, for instance, $\lambda = \{100, 500\}$, the prediction value is different from the actual value and, it means that the lower value has better performance for prediction values. In the following, we evaluate the performance of the Ridge regression for $lambda = 1$. The algorithm is SGD and, we will try for different evaluation metrics. Table 5 represents the measured values for each metric.

Finally, we find the learning parameters $\theta = \{-14.92, -2.54, 0.70, -2.01\}$ and $\beta = 454.42$. Hence, the predicted model for the linear regression problem can be written as

Table 5: Evaluation metrics for prediction function of Dataset 1 for Ridge regression

Mean Squared Error (MSE)	20.06
Mean Absolute Error (MAE)	3.57
Root Mean Squared Error (RMSE)	4.47
Coefficient of Determination (R^2)	0.93
Adjusted Coefficient of Determination (R^2)	0.93
F-statistics	4940.5495

$$f(x) = -14.92x_1 - 2.54x_2 + 0.70x_3 - 2.01x_4 + 454.42. \quad (13)$$

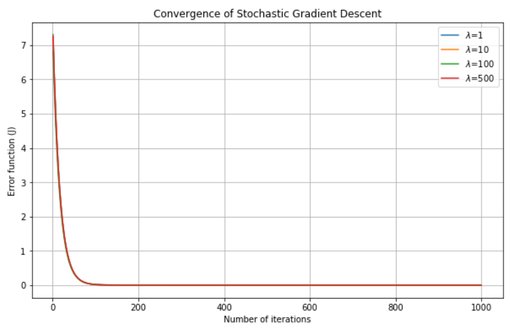

Figure 7: The error function in each iteration of Ridge regression with SGD algorithm of Dataset 2

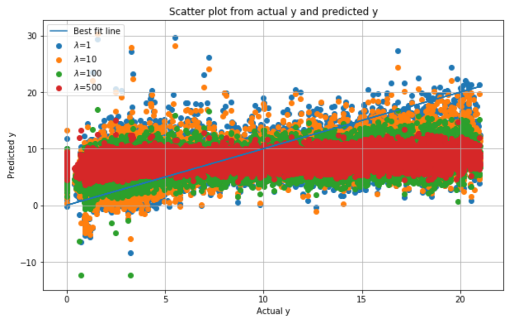

Figure 8: Prediction vs actual labels of Ridge regression with SGD algorithm of Dataset 2

Dataset 2 : Physicochemical Properties of ProteinTertiary Structure Data Set

we derive the Ridge regression problem with SGD algorithm for different amount of $\lambda = \{1, 10, 100, 500\}$. Fig. 7 shows the measured error in each iteration of algorithm for $\alpha = 0.06$ and different value for λ. As we can see the algorithm has reached convergence.

Also, Fig. 8 shows the prediction value for different amount of λ. As we can see, these data are not mapped completely into linear regression, so we can not predict them with the lowest error. However, we can see the effect of different values of λ in our prediction model.

Table 6: Evaluation metrics for prediction function of Dataset 2 for Ridge Regression

Mean Squared Error (MSE)	27.52
Mean Absolute Error (MAE)	4.36
Root Mean Squared Error (RMSE)	5.24
Coefficient of Determination (R^2)	0.27
Adjusted Coefficient of Determination (R^2)	0.27
F-statistics	680.132

In the following, we evaluate the performance of the Ridge regression for $lambda = 1$. The algorithm is SGD and, we try for different evaluation metrics. Table 6 represents the measured values for each metric. Then, we find the learning parameters $\theta = \{1.78, 2.88, 1.05, -5.32, 1.19, -1.23, -0.21, 0.76, -0.07\}$ and $\beta = 7.77$. Hence, the predicted model for the linear regression problem can

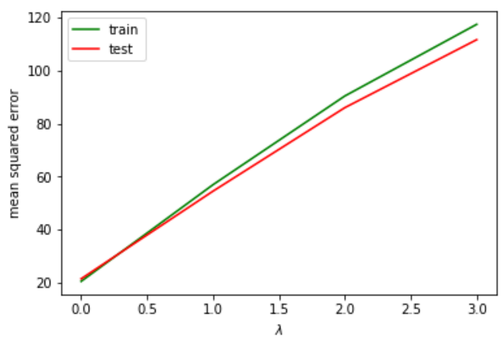

Figure 9: Tuning the hyper-parameter with MSE metric for dataset1

be written as

$$f(x) = 1.78x_1 + 2.88x_2 + 1.05x_3 - 5.32x_4 + 1.19x_5 - 1.23x_6 - 0.21x_7 + 0.76x_8 - 0.07x_9 + 7.77. \quad (14)$$

Tuning Ridge Hyper-parameters

To find the best value and tune the Ridge hyper-parameter, we derive the Ridge model for different values of λ. Fig 9 shows the results for MSE in different values of λ. Therefore, we can find that the minimum values of λ conclude the better performance for our optimization problem for dataset one. It means that the features in this dataset are independent. Hence, by selecting the minimum value for λ, our Ridge regression model would be similar to the LSLR model.

Table 7: Comparison between learning parameters

Regression models	θ_1	θ_2	θ_3	θ_4	β
Our implementation(LSLR)	-13.29	-3.89	0.59	-2.24	454.37
Our implementation(Ridge)	-14.93	-2.54	0.74	-2.02	454.42
Sklearn (LSLR)	-14.82	-2.95	0.38	-2.34	454.31
Sklearn (Ridge)	-10.55	-5.11	1.36	-0.73	454.32
Sklearn (Lasso)	-12.59	-3.45	0.3	-0.46	454.32
Sklearn (Elastic Net)	-10.74	-5.04	1.31	-0.8	454.32

5. LSLR, Ridge, Lasso and ELastic Net Regression problems with Python libraries

In this section, we want to solve our optimization problem with python libraries. To do this, we derive the regression models with the Sklearn library of Python. Tables 7 and 8, respectively, compare the resulting parameters and MSE of Sklearn library models with our implemented model for dataset 1.

As we can see from Table 7, the parameters obtained from our implemented methods are close to the Sklearn library model's parameters. Also, from Table 8 MSE metric calculated by predicted values of our model is close to the MSE calculated from the Sklearn library model. Therefore, our implementation model works properly.

Table 8: Comparison between Mean Squared Error (MSE)

Regression models	MSE
Our implementation (LSLR)	21.17
Our implementation (Ridge)	20.06
Sklearn (LSLR)	20.38
Sklearn (Ridge)	23.4
Sklearn (Lasso)	22.89
Sklearn (Elastic Net)	23.09

Also, the performance of these different models in the SKlearn library depends on their hyper-parameters values. In this work, we find out the minimum of these hyper-parameters result in the best performance for the optimization problem. On the other hand, by selecting the minimum value for hyper-parameters likes α and λ, the Ridge, Lasso, and Elastic Net models will act like LSLR. That is why the LSLR has the best MSE in Table 8.

Conclution

In this report, we discuss several regression models and learn how to solve the optimization problems with them. Also, we find the effect of different parameters in the performance of the regression models. We implement the LSLR and Ridge regression model in Python and test our model with two different datasets. Finally, we compare the results of the implemented models with the Sklearn library models to validate our implementation models.