Project 1: Report
Machine Learning CS-433

Fateme Jamshidi, Ehsan Mokhtarian, Sadegh Khorasani
EPFL Lausanne, Switzerland

***Abstract*—We aim to tackle one of the prevalent machine learning challenges (finding the Higgs boson) with the original dataset from CERN. For this classification task, we compare 1) linear regression, 2) least square regression, 3) ridge regression, and 4) logistic regression. After a preprocessing step and adding higher-order features to the set of features, we achieve the best result by the least square method, with an accuracy of 0.817 on test data.**

I. Introduction

In this project, we work on the *Higgs boson* dataset from CERN. The labels are -1 and $+1$, and the feature vector size is 30. The train dataset contains 568,238 samples, and the goal is to train a model for predicting the labels using the feature vector. To this end, we randomly select 20% of the samples as for validation data, and remove them from train data. Then, we apply various classification algorithms to train a predictive model. Finally, we use the validation data and the test data to tune the hyper-parameters and check whether the trained model is over-fitting, respectively.

In Section II, we first apply two preprocessing steps to normalize and clean the dataset. Then, we add higher-order features to the feature vector, since the available dataset is relatively large compared to the number of features. In Section III, we introduce the implemented classification methods, and discuss how we choose the hyper-parameters in these methods. In Section IV, we evaluate and compare the performance of these methods.

II. Preprocessing step

In this section, we propose three preprocessing steps that we apply to the dataset in order to make the data clean and also to increase the performance of the classification methods. Moreover, we discuss other preprocessing steps that we checked, but did not increase (or even decreased) the accuracy of the classification methods.

NaN entries: After checking the feature vectors, a lot of samples for some features are NaN that are shown by -999 in the dataset. To deal with this problem, we tested two ideas instead of keeping the -999 entries: I) Delete the features that contain a lot of NaN entries. II) Replace the NaN entries by the average of the corresponding feature. As we witnessed in our experiments, in most of the cases, the first idea decreases the accuracy while the second idea increases the accuracy. Hence, we replace the the NaN entries by the average of the corresponding feature.

Normalization: Normalization is a crucial step for some classification methods, e.g., linear regression using gradient descent. Hence, for each feature S and for each sample of it like X, we replace X by $\frac{X-\mu_S}{\sigma_S}$, where μ_S and σ_S are the average and the standard deviation of the samples of S.

Higher-order features: In order to increase the accuracy by training a more complex classification model, we add higher-order features to the feature vector. Formally, for each feature X, we add $\eta - 1$ new features with values $X^2, X^3, \dots, X^\eta$. In the next section, we discuss how to select η to increase accuracy while avoiding over-fitting.

III. Classification methods

In this section, we discuss various classification methods that we use in this project. Moreover, in order to choose the hyper-parameters of these methods, we present a set of experiments.

Linear regression using normal equation, GD, and SGD: Loss function in linear regression is as follows, where N is the number of samples.

$$L(w) = \frac{1}{2N}||Xw - y||^2. \tag{1}$$

To minimize this function, we use the following closed form solution as provided in the lecture notes.

$$w^* = (X^T X)^{-1} X^T y. \tag{2}$$

Table I shows the loss function of trained models using normal equation for different values of η. From this table we conclude that over-fitting happens for $\eta > 10$. Hence, we select η to be 9. Next, we implement GD and SGD methods to minimize the cost function in Equation 1.

For GD method, we used different values of the learning rate (lr), denoted by γ, and observe that the large values of γ, e.g., 0.1, makes the gradient to explode. Figure 1 depicts the loss function for $\gamma = 0.01$ and $\gamma = 0.001$ for this method. In our submitted version, we set $\gamma = 0.01$ and the number of iteration to be 5000.

Similarly for SGD, we repeat the same experiment and observe that for $\gamma = 0.01$, the gradient does not converge. Figure 2 shows the loss function for $\gamma = 0.001$ and $\gamma = 0.0001$. In our submitted version, we set $\gamma = 0.0001$ and the number of iteration to be 20000.

Table I: Loss function for different values of η in linear regression using normal equations.

η	3	4	5	6	7	8	9	10	11
validation loss function	0.786	0.795	0.798	0.800	0.808	0.814	0.817	0.817	0.815

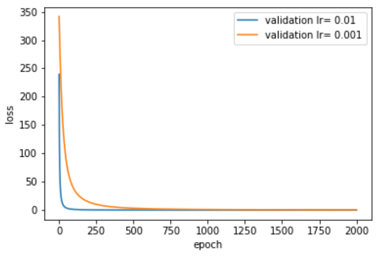

Figure 1: Validation loss in linear regression using GD.

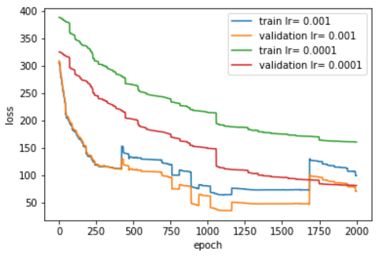

Figure 2: Training and validation loss in linear regression using SGD.

Ridge Regression: Loss function in ridge regression is

$$L(w) = \frac{1}{2N}||Xw - y||^2 + \lambda||w||^2, \quad (3)$$

where λ is the hyper-parameter in this method. To minimize this function, we use the following closed form solution.

$$w^* = (X^T X + 2N\lambda I)^{-1} X^T y. \quad (4)$$

Table II: Loss function for different values of λ in ridge regression.

λ	train loss	validation loss
0	0.069	0.071
0.1	0.095	0.095
0.2	0.104	0.105

Table II illustrates the loss function for different values of the regularization term. According to this table, by increasing λ, the loss function for both the training and validation set increases.

Logistic Regression: Loss function in logistic regression without regularization term is as follows.

$$L(w) = \frac{1}{N}(\sum_{n=1}^{N} ln[1 + exp(x_n^T w)] - y_n x_n^T w). \quad (5)$$

We use GD to minimize this lost function. After checking different learning rates in GD method, we select $\gamma = 0.1$. As shown in Figure 3, train and validation loss converge after 1000 iterations. In the final model, training and validation loss are 0.480 and 0.496, respectively.

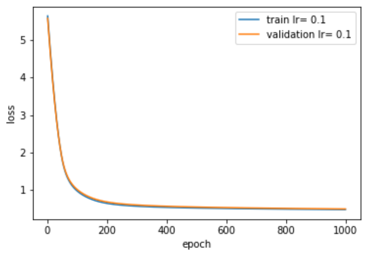

Figure 3: Training and validation loss in logistic regression using GD.

In logistic regression with regularization term, we need to tune the regularization term λ. Our experiment data shows that for $\lambda = 0.1, 0.3, 0.5$ the loss of validation are 0.488, 0.492 and 0.489, respectively. Also for larger λ we witness that the validation loss increases. Therefore, we select $\lambda = 0.1$.

IV. Comparing the Performance of the Methods

In this section, we evaluate and compare the performance of the aforementioned methods with the tuned hyper-parameters selected in Section III. Table III shows the accuracy of these methods on the test dataset. As this table suggests, the least square method that uses the normal equation achieves the best accuracy.

Table III: Accuracy of various methods on the test data.

Method	Accuracy
Least squares GD	0.755
Least squares SGD	0.649
Least squares	0.817
Ridge regression	0.817
Logistic regression	0.779
Regularized logistic regression	0.778

V. Conclusion

In this project, we used machine learning methods to make predictions in Higgs boson dataset. The major steps in this process were dealing with missing values, normalization, high-order features and testing/tuning the models. By comparing the result of these methods, we conclude that least squares with normal equations has the best performance for this task.