ECSE 551 Machine Learning for Engineers
Mini-project 1 Report

Fei Peng
260712440
fei.peng@mail.mcgill.ca

Yukai Zhang
260710915
yukai.zhang@mail.mcgill.ca

Yi Zhu
260716006
yi.zhu6@mail.mcgill.ca

Abstract

Logistic regression is one of the most popular linear classification techniques in machine learning and it has been widely used in binary classification and predictions. In this mini-project, we investigated and evaluated the performance of logistic regression by implementing it using two benchmark datasets: *Hepatitis* and *Bankruptcy*. Both two datasets were pre-processed to explore their characteristics, thereby achieving the best outcome in the following training. Fitting, predicting, and testing were then performed to train and compare different models. The accuracy evaluations were analysed, and detailed results will be presented in the following parts of this report. Among them, we found that the accuracy of the logistic regression classifier could be improved by normalizing the datasets, changing order of the model, or removing certain features of the numerical datasets.

1 Introduction

In recent years, machine learning has been adapted to many different fields, including medical research and financial analysis [1]. As one of the most feasible and efficient classification methods, logistic regression is widely used to predict binary outcomes. The main objective of this project is to implement a logistic regression classifier and investigate its accuracy using k-fold cross validation, as well as analysis the impact of data preprocessing on the validation accuracy. We took the probabilistic views to process the binary classification. In probabilistic approaches, we focused on discriminative learning which directly estimates the probability of y (i.e., output class/label) given x (i.e., input data). Using Bayes' Rule, the logistic function (also known as sigmoid function) could be obtained, which takes a linear function of x as independent variable $w^T x$, and produces the conditional probability $P(y|x)$.

The most critical task of implementing the logistic regression algorithm was to find the linear term w (i.e., weights) that produces the minimum error/maximum accuracy in prediction while ensuring the efficiency of this algorithm. This was achieved using gradient descent with momentum [2], and tweaking the hyperparameters (e.g., learning rate, momentum term, and stopping criteria) of the fit function. It was found that, with a carefully chosen set of hyperparameters, accuracy could be increased while remaining satisfying training efficiency.

During data pre-processing, normalization was performed and proved to increase the accuracy. While analysing the characteristics of the datasets, it was noticeable that some features of input x satisfies the null hypothesis [3], which means that these features are not helping discriminate between two classes and should be removed to improve performance. Moreover, through the distribution of the two classes of the two datasets, it was discovered that the entropy of classes could affect the accuracy of the model, resulting in the model accuracy for hepatitis data (with higher entropy) being higher

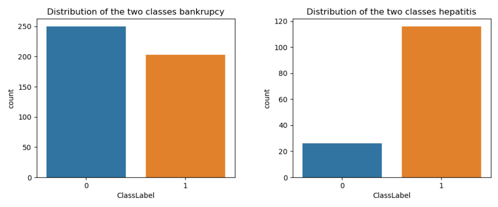

Figure 1: *Bankruptcy* data class distribution (left) and *Hepatitis* data class distribution (right)

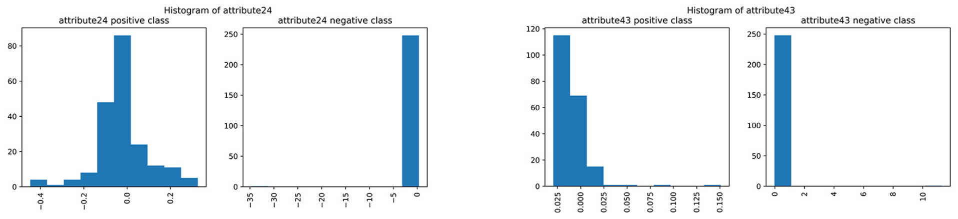

Figure 2: Attribute 24 (left) and attribute 43 (right) feature distribution for *Bankruptcy* data

than that of *bankruptcy* data. Last but not least, properly increasing the order of the model (order 3 for *bankruptcy* data) would also help increase accuracy. These aspects will be further discussed.

2 Datasets

Data preprocessing usually lays the groundwork for the later data analysis and helps it yield a higher accuracy efficiently. In this project, the following data preprocessing had been performed.

2.1 Entropy Analysis

Both datasets *Bankruptcy* and *Hepatitis* are examples of binary classification. The distributions of each class are shown in Figure 1. The uncertainty in prediction can be quantified by the entropy of each dataset with equation 1.

$$H(X) = -\sum_{x \epsilon X} p(x) log_2 p(x) \tag{1}$$

2.2 Features Analysis

For the purpose of gaining a better understanding toward the dataset, the distributions of each feature in both classes are plotted. Some of the plots are shown in Figure 2.

Note that both pairs of plots come from *Bankruptcy* dataset and in each pair, the distribution of positive class is shown on the left, otherwise right. When comparing positive and negative classes, it is noticeable that the data distribution differs greatly in attribute 24 but slightly in attribute 43. This indicates that features with similar distribution to attribute 43 will have little contribution in the prediction process and should be considered as null distribution. Kolmogorov-Smirnov method [4] was performed to efficiently find all null distributions. Attributes 43 and 60 in the *Bankruptcy* dataset were identified. The removal of these two features may improve the performance of the machine learning process. The Kolmogorov-Smirnov method, however, failed to yield a promising result in the *Hepatitis* dataset because of the existence of multiple binary features. The effect of removing features with null distribution will be further discussed in the result section.

2.3 Data Shuffling

It was found that the *Bankruptcy* dataset is ordered by class label. Proceeding without shuffling the data would result in some validation/training sets all filled by data from one class. Thus, rows will be shuffled every time after importation. Doing so will also help reduce the variance and generalize the algorithm, therefore the *Hepatitis* dataset should also be shuffled.

2.4 Data Normalization

Normalizing the data into z-score can centralize the value distribution of the data while keeping its precision [5]. When a calculation involves large float numbers, they can easily trigger overflow/underflow and lose precision without normalization. Calculating the gradient with high order data is an example of that. Luckily, using z-score should help prevent introducing such errors and thus increase the model accuracy.

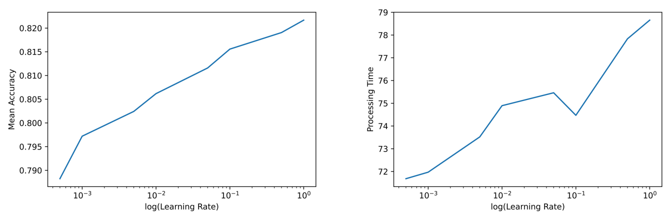

Figure 3: Mean accuracy (left) and processing time (right) vs. logarithm of learning rate

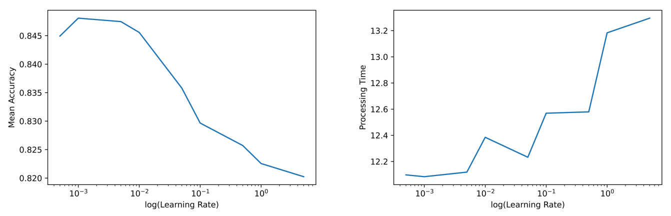

Figure 4: Mean accuracy (left) and processing time (right) vs. logarithm of learning rate

3 Results

Multiple experiments have been conducted to evaluate the performance of the logistic regression with respect to different hyperparameters and data preprocessing techniques. The detailed results are shown below.

3.1 Learning Rate

Learning rate is a parameter that determines the step size on the way approaching a (local) minimum. A large step size can accelerate the convergence but can also result in jumping over the desired solution (oscillate forever). A small step size can avoid skipping the solution but is very time consuming. To find a learning rate best fit for each model, values between 0.0005 and 5 are tested. The performance/processing time vs. logarithm of learning rate are shown in Figure 3 and Figure 4.

3.2 Stopping Criteria:

Theoretically, if a model suits the data distribution, the training accuracy will keep increasing with more iteration calculated. However, failing to set up a proper stopping criteria will not only waste the computational power but also lead to overfitting [6]. In this case, max iteration and epsilon(minimum gradient) are introduced as the upper limit of the iterations. In this experiment, the max iteration term is sampled from 500 to 25000 and the epsilon term is sampled from $1e^{-3}$ to 0.5. The plots of accuracy/processing time versus max iteration/epsilon are shown in Figure 5 and Figure 6.

Note that all the data displayed come from the *Bankruptcy* dataset. The *Hepatitis* dataset has a similar trend and will not be displayed. The plots confirm that in a reasonable iteration range, accuracy tends to rise as the maximum iteration increases and the minimum acceptable gradient decreases. For the trade-off between accuracy and efficiency, 25000 iterations with an epsilon of $1e^{-3}$ is selected for the *Bankruptcy* dataset and $(10000, 5e^{-3})$ is selected for the *Hepatitis* dataset.

3.3 Momentum Gradient Descent Constant - Beta

When a regular gradient descent method is performed, it is very unlikely that the algorithm can move directly from the starting point to the (local) minimum. Instead, the path usually oscillates back

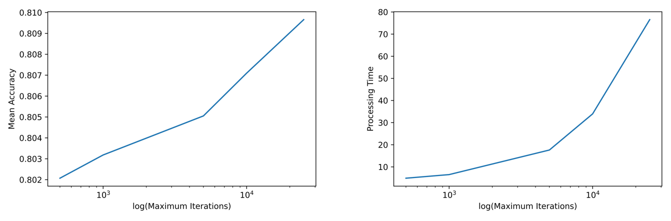

Figure 5: Mean accuracy (left) and Processing time (right) with respect to log of Max. Iteration

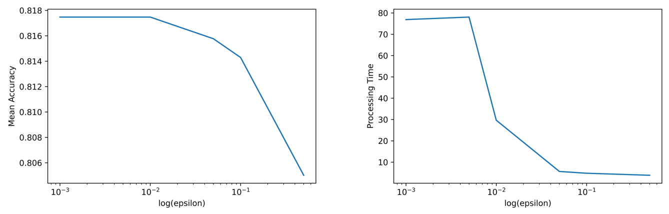

Figure 6: Mean accuracy (left) and processing time (right) with respect to log of epsilon

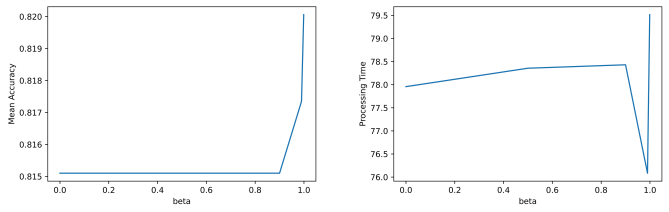

Figure 7: Mean accuracy (left) and processing time (right) with respect to beta

and forth around that optimum shortcut. As described in part 3.1, this oscillation will have a larger amplitude if the learning rate is high. A momentum gradient descent method can be used to flatten this oscillation and take a rather straightforward approach in the search of the minimum point. It does so by calculating the weighted sum of the current gradient and the previous step size and then using it as the new step size. β value is in charge of controlling the weight of the previous step size, if β is set to 0, it becomes equivalent to a regular gradient descent method [2]. In this experiment, value β is tested in the range of 0 to 0.999 and its results are shown in Figure 7.

The data displayed all come from *Bankruptcy* dataset. The trends of that in the *Hepatitis* dataset is also very similar to those above. According to the figures, a β of 0.99 yields the best overall performance regarding the accurateness and efficiency.

3.4 Normalization

During data preprocessing, it was discovered that most of the features in *Bankruptcy* data are floating points. Therefore, calculation may result in precision loss due to overflow/underflow. To improve model accuracy, data should be normalized before performing training or validation. The normalization technique used in this project is called z-score [5] which calculates the mean (j) and standard deviation (μ_j) for each feature (σ_j) of the training data, and perform $\frac{x_j - \mu_j}{\sigma_j}$ for both the training and validation data. Therefore the model accuracy could be improved by bringing more precision to the calculation of gradient. During the testing, it was found that the overflow warning indeed disappeared after normalization. It is shown in Figure 8 on the left that compared to that of without normalization, the validation accuracy of the normalized original dataset is higher. This advantage is even pronounced when more features (higher orders) are introduced, which will be discussed later.

3.5 Removing Features

As discussed previously in datasets, null distributions in the *Bankruptcy* dataset were found using Kolmogorov-Smirnov method [4]. This means that the distribution of some attributes in two classes

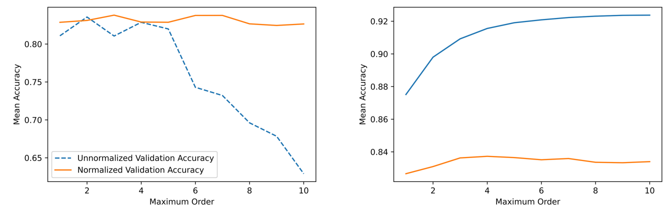

Figure 8: Comparison between with and without normalization (left) and the training and validation accuracy vs. order of *Bankruptcy* data (right)

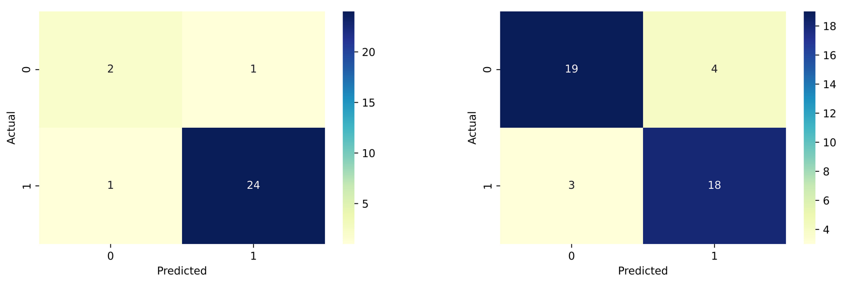

Figure 9: The Confusion Matrix. for *Hepatitis* data (left) and *Bankruptcy* data (right)

are similar and may not be useful for training and class prediction. Attribute 43 and 60 of *Bankruptcy* dataset were identified to be unhelpful, therefore were removed during training and validation. However, it came out that the mean validation accuracy and training efficiency did not vary significantly from before. One of the assumptions that could explain this result is that the linear term, w, of the log-odds ratio (i.e., the independent variable, $w^T x$, of the logistic function) as already taken care of the weight of each attribute, therefore removing any attribute would not have much impact on the accuracy.

3.6 Adding features

The approach of adding more features to the dataset was also adopted to improve model accuracy. The order of each attribute in the *Bankruptcy* dataset was increased and added to the original dataset. As shown in Figure 8 on the right, the blue curve indicates the training accuracy, while the orange curve represents the validation accuracy. It clearly shows that when the order is increased to around 3, the maximum validation accuracy is achieved. However, continuing increasing the order could cause overfitting [6], as the training accuracy remains increasing but the validation accuracy starts to decrease. Therefore, order 3 was chosen to be the most suitable model for training *Bankruptcy* dataset. This approach, however, was tested to be ineffective for *Hepatitis* dataset due to the fact that most of its features are binary.

4 Discussion and Conclusion

In conclusion, the logistic regression classifier implemented in this project performed as expected overall on the two datasets, *Bankruptcy* and *Hepatitis*. Here are the two representative confusion matrices, shown in Figure 9, calculated during testing to measure the accuracy, precision and sensitivity of the models. The selected model used for calculating the confusion matrix was the one that performed the best during testing (as explained in the previous section). The accuracy for the best *Hepatitis* model could achieve 92.86%, with 96% specificity. However the number of instances in this dataset is too small (only 142 total) to take any common measurements, therefore 5-fold validation was performed to obtain this confusion matrix. Moreover, this is one of the best case scenarios, thus in the future, more testing should be performed, ideally with more instances, to obtain a convincing average accuracy. For the best *Bankruptcy* model, the accuracy reached 84.09%, while the precision was 82.61% and the sensitivity was 86.36%. These measurements confirmed the previous hypothesis, that lower entropy can yield a higher accuracy.

Possible directions for future investigation could include choosing other types of model (e.g., with log, exponential, or interaction terms), exploring more efficient but complex gradient descent algorithms, and comparing logistic regression with genetic learning as *Hepatitis* is a relatively small dataset. These aspects should help not only improve model accuracy, but also accelerate the training process.

5 Statement of Contributions

The workload of this project is distributed equally between the team members (i.e., the three authors of this report). Yi Zhu implemented the fit function for training datasets, and explored various gradient descent algorithms to improve the computation speed. Fei Peng was responsible for developing the k-fold partition, predict and accuracy evaluation functions for predicting the classes/labels of the input data and evaluating the model accuracy, as well as performing thorough testing of the models. Yukai was in charge of data pre-processing and model selection, which helped to significantly improve the model accuracy.

References

[1] T. Peña, S. Martínez, and B. Abudu, "Bankruptcy Prediction: A Comparison of Some Statistical and Machine Learning Techniques," in *Computational Methods in Economic Dynamics,* H. Dawid and W. Semmler Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 109-131.

[2] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747,* 2016.

[3] R. W. Vitral, M. J. Campos, and M. R. Fraga, "The null hypothesis," *American journal of orthodontics and dentofacial orthopedics : official publication of the American Association of Orthodontists, its constituent societies, and the American Board of Orthodontics,* vol. 144, no. 4, pp. 498-9, 2013, doi: 10.1016/j.ajodo.2013.08.010.

[4] T. Liu, "A Kolmogorov-Smirnov type test for two inter-dependent random variables," *arXiv preprint arXiv:1802.09899,* 2018.

[5] A. E. Curtis, T. A. Smith, B. A. Ziganshin, and J. A. Elefteriades, "The Mystery of the Z-Score," *(in eng), Aorta (Stamford),* vol. 4, no. 4, pp. 124-130, 2016, doi: 10.12945/j.aorta.2016.16.014.

[6] AH. Allamy, "METHODS TO AVOID OVER-FITTING AND UNDER-FITTING IN SUPERVISED MACHINE LEARNING (COMPARATIVE STUDY)," 12/27 2014.

Appendix

(Please see following pages for code implementations attached)