Logistic Regression Classifier Implementation on the Wine Dataset

Tommy Liu and Pavel Filippov

December 3, 2025

Introduction

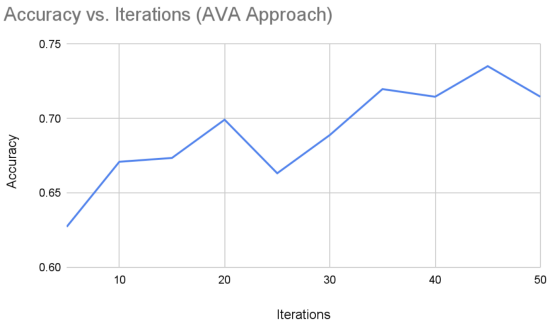
Logistic Regression has applications ranging from risk prediction to text classification. Take life expectancy as an example to demonstrate the application of Logistic Regression in Risk Management: based on features such as smoking, sex, and exercise frequency, Logistic Regression can be used to predict the life expectancy of an individual. It can also be used to make categorical, qualitative predictions: for instance, one can set a cutoff at a certain age, and any life span below that age will be considered a high-risk individual. In text classification, each word is often considered a feature. Using the presence or absence of each word, Logistic Regression can predict, for example, whether a text is sent by a happy person or a sad person. The core design behind Logistic Regression is that each feature is assigned a weight. The prediction is calculated by passing the dot product between the weight and feature vector into the sigmoid function. After each training example, the weight vector is updated using gradient descent. While Logistic Regression is primarily a binary classifier, it could be integrated with techniques such as one-versus-all (OVA) and all-versus-all (AVA) to support multiclass classification.

Project Roadmap

The project team has implemented the binary Logistic Regression (LR) Classifier and has integrated the binary LR classifier with AVA and OVA multiclass classification. The project team attempted to use the binary LR Classifier with AVA to perform multiclass classification on the *wine* dataset from the previous assignment. However, the project team later realized that the training process took an extended amount of time (the reason for this will be discussed later in the report). Therefore, the project team adopted a different implementation - called Softmax LR - that equips the LR Classifier itself with the ability to perform multiclass classification, so it suffices to train only one LR classifier. Without needing multiple classifiers as in AVA or OVA, the runtime was reduced significantly. For the experiment, the project team determined the optimal combination of parameters for the classifier, including the number of iterations and the learning rate. The performance metric was accuracy, which was calculated on the testing set, averaged over 10 trials, obtained from a random 80/20 split of the *wine* dataset.

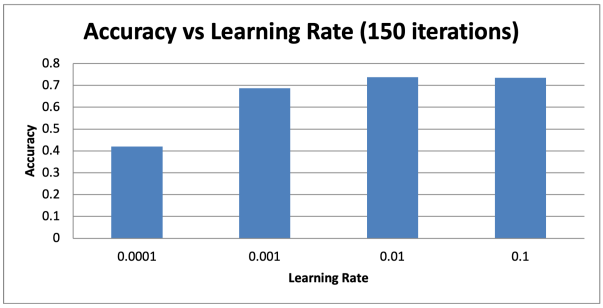
Experimentation

The project team first implemented the binary LR classifier and integrated the classifier with AVA and OVA. The team then fixed the learning rate at $\alpha = 0.01$, and varied the number of iterations to identify the range for the best performance. Below is the graph summarizing the trend in classifier performance as the number of iterations increases.



It appeared that the optimal number of iterations was around 45. However, the decrease in accuracy from 45 to 50 could result from randomness. The natural question that readers might ask is why the project team did not continue running their experiment for a higher number of iterations. It turned out that the LR classifier with AVA is computationally expensive. The project team was unable to train the classifier efficiently because each word is treated as a feature, resulting in thousands of features in the *wine* dataset. To use AVA, a separate classifier is required for every pair of labels in the dataset. The project team conjectured that the high number of classifiers required for AVA is the primary reason why the runtime of the LR Classifier with AVA is so long.

The project team considered an alternative approach to train the LR classifier: instead of predicting the probability that a text belongs to one class rather than another, the model could be trained to output a distribution of probabilities that the text belongs to *each* class. This is known as the Softmax LR Classifier. In this case, the LR Classifier itself becomes a multiclass classifier. To find the best combination of parameters, the project team first fixed the learning rate and varied the number of iterations to find the optimal number of iterations. Next, the team held the number of iterations at its optimality and varied the learning rate to find the optimal learning rate. Lastly, the project team tried multiple combinations of parameters in the neighborhood of the optimal combination to ensure that the optimal combination they found was indeed optimal. Below are the relationships between testing accuracy and learning rate, and between testing accuracy and number of iterations.



Results and Future Work

The project team investigated two versions of the multiclass LR classifier: the AVA approach and the softmax approach. The softmax approach has a much shorter training time than the AVA approach because it only uses a single classifier. The best combination of parameters in the softmax approach is a learning rate of 0.01 and 150 iterations, yielding an accuracy of 75%. Due to time constraints and the late realization that AVA and OVA are not viable approaches for text classification, the project team did not have time to add regularization to the LR classifier. The effect of regularization on the accuracy could be further explored in the future. Meanwhile, the project team would like to compare the performance of the LR classifier and the Naive Bayes (NB) classifier, both under their optimal set of parameters. The optimal parameter combination of the NB classifier was determined in Assignment 7. From the project team's experience, the LR classifier performs better than the NB model, but the hypothesis requires future experimentation.

(reference are attached to the project proposal)