UNIVERSITY OF VICTORIA

Department of Electrical and Computer Engineering

ECE 503 Optimization for Machine Learning

PROJECT REPORT

Title: Banknote Authentication via Logistic Regression

Report Submitted on: December 18, 2021

To: Lei Zhao

Name: Siddharth Chadda (V00947906)

Table of Contents

1. Objective

Our goal in this project is to build a binary classification based Logistic Regression model to automatically to predict whether a bank note is Genuine or Counterfeit. In this project we are using the Banknote Authentication Data Set provided by UCI Machine Learning Repository. Further in this project we will optimise the parameters of the Logistic Regression model by minimising the SoftMax cost function.

2. Introduction

Even though we are living in the age of digital banking and credit cards, however the use of paper money remains one of the main options for the exchange of products and services. One of the main problems with using physical money is the detection of counterfeit banknotes, these counterfeit notes have over the years have become increasingly close to resemble the genuine notes.

Although there are machines for detecting counterfeit banknotes, however, these machines are very expensive, so the identification of counterfeits can only be done by financial and government institutions.

Through our project we wish to harness the power of machine learning and optimization algorithms to develop a cheap and cost-effective solution for detecting counterfeit notes.

2.1 Dataset Description

The dataset used in this project is the Banknotes Authentication dataset taken from the UCI Machine Learning Repository. The data set was built by first extracting data from images of genuine and forged banknote specimens using an industrial camera. The final images obtained were of 400x 400 pixels in size with a resolution of about 660 dpi. Post this, Wavelet Transform tool was applied to extract features from these images.

The final dataset has 1,372 data rows with 5 numerical variables. The dataset presents a binary classification problem with two classes namely, Genuine and Fake banknotes. Where 0 represents Genuine banknotes and 1 represents Fake banknotes.

The five variables in the dataset are:

- variance of Wavelet Transformed image (continuous).
- skewness of Wavelet Transformed image (continuous).
- kurtosis of Wavelet Transformed image (continuous).
- entropy of image (continuous).
- class (integer).

3. Formulating the problem as an Optimization Problem

From the nature of the dataset, we can clearly see that it is a binary classification problem that can be solved using Logistic Regression. However, given the sensitivity of Logistic Regression classifier towards outliers and unscaled data makes Logistic Regression on its own a very weak classifier. Thus, in order to increase overall accuracy of our classifier we will implement the below optimization techniques to come up with a much more smooth and efficient solution of the problem at hand.

I. **Standard Normalization:**
Data normalization is a necessary step especially when numerical ranges of different features in the raw data are wildly different. Through data normalization we can scale down the features in the dataset down to a similar scale while maintaining the ratio between them. This scaling down of features help us eliminate magnitude bias from our dataset since now all of the features are similar in magnitude and hence all the features become equally important for the classification model. Thus removing the bias from our dataset, the performance of the classification model increases.

II. **SoftMax Regularization:**
Using SoftMax cost function improves the learning of our model in terms of solution uniqueness and better classification accuracy. On its own Logistic Regression is a very weak classification model however by applying SoftMax Regression on top of logistic regression we can make the decision boundary of the logistic regression-based classification model to fit more *tighter* to our dataset.

4. Implementation and Solution Description

Summary of the steps implemented:

1) The 'banknote_authentication.csv' data file is loaded.
2) The dataset is divided into Training and Testing datasets.
3) Standard Normalization is applied on the training and testing datasets
4) We create a SoftMax cost function under the name 'f_wdbc.mat'.
5) Then we create a gradient of the SoftMax cost function under the name 'g_wdbc.mat'.
6) We minimise the cost function using the Gradient Descent algorithm.
7) We try different values for μ and K, to obtain different solution parameters.
8) We evaluate the model on training set by calculating Confusion Matrix, Accuracy Precision, Recall, F1-Score.
9) We evaluate the model on testing set by calculating Confusion Matrix, Accuracy Precision, Recall, F1-Score.

4.1 Step 1: Preparing the Training and Testing datasets

The dataset is of size 1373 x 5 in size. We build the training and testing datasets by splitting the dataset in a 70:30 ratio, meaning, the first 961 rows in the dataset (representing approximately 70 percent of the dataset) are used to build the training et to train our classification model and the next 412 rows (representing approximately 30 percent of the dataset) are used to test performance of our classification model.

We further split the datasets according to their columns by placing the Feature columns in *X_train* and *X_test* matrices and the Label or Target vector columns in *y_train* and *y_test* vectors. Further for easier processing we take the transpose of each these matrices.

Thus, we are able to get the below matrices and vectors:

Matrix/ Vector	**Final Shape**
Original_dataset	5 x 1372
X_train	4 x 961
X_test	4 x 411
y_train	1 x 961
Y_test	1 x 411

4.2 Step 2: Pre-Processing the Data by Standard Normalization

Before constructing a useful cost function, it is necessary that we pre-process the original data set especially when the numerical ranges of different features in our dataset varying wildly. This is true in our case also, the scale of features in our dataset are varying greatly having values between -10 to 15. Hence it is essential for us to normalize our dataset inorder to remove any bias from the dataset.

Further by normalising the input data, each feature on the data set becomes mean-cantered with unit variance, and the contours of the cost function built with normalised data become much more circular, resulting in quicker gradient-descent iterations.

Further in practical machine learning application, we must only normalize the training dataset (ie. *X_train)* first and forget about the testing dataset (ie. *X_test*). This is done so because during the training phase where model parameters are being optimized by minimizing a loss

function, only the training dataset is available to the model and the testing data set is not available until the training phase is complete.

Hence for real-world ML applications the data normalization step should be performed as two separate sub-steps:

i. First the training dataset should be normalized
ii. Statistical information like mean and variance should be calculated using the training data (this can be done through code by using functions like *mean(xi)* and *var(xi)*)
iii. Now only we will use the statistical information calculated above to normalize the testing dataset.

4.3 Step 3: Applying Logistic Regression Based Binary Classification

We will now apply the Logistic regression based binary classification on top of the normalized training datasets.

Steps:

i. For the logistic regression we use a variant of the SoftMax cost function given by:

$$E_{LR}(\hat{\boldsymbol{w}}) = \frac{1}{P}\sum_{p=1}^{P}\log\left(1+e^{-y_p\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_p}\right)+\frac{\mu}{2}\|\hat{\boldsymbol{w}}\|_2^2$$

We create this function programmatically in the code file 'f_wdbc.mat'.

ii. To apply the gradient descent (GD) algorithm, gradient of the above cost function is required. the gradient of the SoftMax cost unction is given by:

$$\nabla E_{LR}(\hat{\boldsymbol{w}}) = \mu\hat{\boldsymbol{w}} - \frac{1}{P}\sum_{p=1}^{P}\frac{y_p\hat{\boldsymbol{x}}_p}{\left(1+e^{y_p\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_p}\right)}$$

Thus, now we programmatically create a gradient of the SoftMax cost function under the name 'g_wdbc.mat'.

iii. We will now minimise the cost function using the Gradient Descent algorithm.
iv. We try different values for μ and K, to obtain different solution parameters.

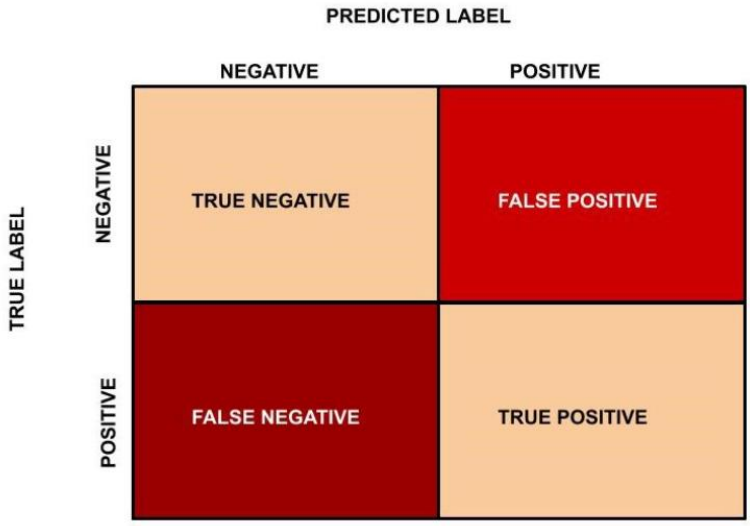
4.4 Step 4: Evaluating Model Performance

To get better understanding of the performance of our model, we evaluate the model on training dataset as well as on the testing dataset by calculating Confusion Matrix, Accuracy Precision, Recall, F1-Score.

I. Confusion Matrix:

Confusion Matrix in case of binary classification problem is 2 x 2 matrix that is used to infer the performance of a classification-based machine learning algorithm.

A confusion Matrix is extremely useful in calculating the Accuracy, Precision, Recall, and F1 scores of classification-based ML model.



Definitions:

a) **True Positives (TP):** These are cases in which we predicted fake note, and the note is actually fake.
b) **True Negatives (TN):** We predicted not fake, and the note is not fake.
c) **False Positives (FP):** We predicted fake, but the note is not fake. (Also known as a "Type I error.")
d) **False Negatives (FN):** We predicted not fake, but the note is actually fake. (Also known as a "Type II error.")

II. Accuracy:

Accuracy represents the number of correctly classified data instances over the total number of data instances.

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN}$$

III. Precision:

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

$$Precision = \frac{TP}{TP + FP}$$

IV. Recall:

Recall is the ratio of correctly predicted positive observations to the all observations in actual class.

$$Recall = \frac{TP}{TP + FN}$$

V. F1 Score:

F1 Score is the weighted average of Precision and Recall.

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

5. Results and Simulations

We will try different values for μ and K, to obtain different solution parameters and then evaluate the model on training dataset as well as on the testing dataset by calculating Confusion Matrix, Accuracy Precision, Recall, F1-Score.

I. For μ = 0 and K= 10:

Training Dataset	Testing Dataset
CONFUSION_MATRIX_test = 2×2 182 70 0 159 ACCCURACY_test = 0.829683698296837 PRECISION_test = 0.694323144104803 RECALL_test = 1 F1_SCORE_test = 0.834605597964377	CONFUSION_MATRIX_train = 2×2 428 154 0 379 ACCCURACY_train = 0.839750260145682 PRECISION_train = 0.711069418386492 RECALL_train = 1 F1 SCORE train = 0.848812095032397

II. For μ = 0.1 and k= 10:

Training Dataset	Testing Dataset
CONFUSION_MATRIX_test = 2×2 182 70 0 159 ACCCURACY_test = 0.829683698296837 PRECISION_test = 0.694323144104803 RECALL_test = 1 F1_SCORE_test = 0.819587628865979	CONFUSION_MATRIX_train = 2×2 428 158 0 375 ACCCURACY_train = 0.835587929240375 PRECISION_train = 0.703564727954972 RECALL_train = 1 F1_SCORE_train = 0.825991189427313

III. For μ = 0 and k= 30:

Training Dataset	Testing Dataset
CONFUSION_MATRIX_test = 2×2 182 63 0 166 ACCCURACY_test = 0.846715328467153 PRECISION_test = 0.724890829694323 RECALL_test = 1 F1_SCORE_test = 0.840506329113924	CONFUSION_MATRIX_train = 2×2 428 134 0 399 ACCCURACY_train = 0.860561914672216 PRECISION_train = 0.748592870544090 RECALL_train = 1 F1_SCORE_train = 0.856223175965665

IV. For μ = 0.05 and k = 30:

Training Dataset	Testing Dataset
CONFUSION_MATRIX_test = 2×2 182 65 0 164 ACCCURACY_test = 0.841849148418491 PRECISION_test = 0.716157205240175 RECALL_test = 1 F1_SCORE_test = 0.834605597964377	CONFUSION_MATRIX_train = 2×2 428 140 0 393 ACCCURACY_train = 0.854318418314256 PRECISION_train = 0.737335834896811 RECALL_train = 1 F1_SCORE_train = 0.848812095032397

6. Conclusion

We were able to successfully use Logistic Regression to predict the authenticity of Bank notes. To increase the model accuracy and to remove bias from the model, the training and testing datasets were normalized and the model the model parameters were optimised by minimising the SoftMax cost function. Model evaluation for different values of μ and K done by calculating the Confusion Matrix and Accuracy score and best values of μ (μ = 0) and K (K =30) were selected. By selecting these parameters, we were able to get an accuracy of 84.67% on the training set and 86.05% on the testing set.

7. References

i. https://archive.ics.uci.edu/ml/datasets/banknote+authentication
ii. https://studentweb.uvic.ca/~leizhao/403-2021/LabManual2021.pdf
iii. https://www.mathworks.com/help/matlab/referencelist.html?type=function

Appendix: MATLAB Code