

McGill University

Course COMP-551 Fall 2019

Applied Machine Learning

Instructor
William Hamilton

PROJECT REPORT

AUTHORS:

Hamza Rizwan
260816900

Ashray Malleshachari
260838256

Logan Ralston
260851093

Abstract

This project involved building and investigating the performance of two linear classification models without using machine learning libraries like sklearn. The two models were Logistic Regression and Linear Discriminant Analysis (LDA) and they were trained on two open-source datasets: Wine Quality and Breast Cancer Diagnosis. The project also included data cleaning and pre-processing to remove missing values, feature scaling, and labelling examples based on certain conditions. We also implemented 5-Fold Cross Validation for our experiments. We found that the LDA Model was significantly faster to train than the Logistic Regression model on both datasets and we achieved a better accuracy on the Breast Cancer Dataset when training with both models. We also explored the relationship between learning rate and training accuracy using Logistic Regression to find an optimum learning rate for our model.

1 Introduction

The objective of the project was to implement two linear classification models (Linear Regression and Linear Discriminant Analysis) on two public datasets: Red Wine Quality Dataset and Breast Cancer Diagnosis dataset. For the red wine dataset, we have built a classifier to predict the wine quality based on its chemical properties. For the second dataset, we built a logistic regression classifier using gradient descent to predict the occurrence of breast cancer.

2 Dataset

Our goal for the red wine quality dataset was to predict the quality of red wine based on its chemical properties. This dataset contained 1599 training examples with 11 feature columns (not counting the quality column) and did not have any missing features. The last column of the dataset was the quality of wine from 1-10. We converted this quality column to binary values by thresholding the quality at 6 (Wines with a quality of 6 or higher were labelled 1 or Good Quality, and wines with 5 or lower were labelled 0 or Bad Quality). No other changes were made to the wine dataset as we didn't have any missing values. The Red Wine dataset contained 855 rows that belonged to class 1 (positive) and 744 rows that belonged to class 0 (negative).

Features of the red wine dataset:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	qua
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	0.658149	10.422983	5.636
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	0.169507	1.065668	0.807
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	0.550000	9.500000	5.000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	0.620000	10.200000	6.000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	0.730000	11.100000	6.000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000	2.000000	14.900000	8.000

Table 1: Features of the red wine dataset

After this, we created a feature correlation heatmap using seaborn and found that some features were correlated with others which could give us some information about any redundant features that could be linear combinations of other features.

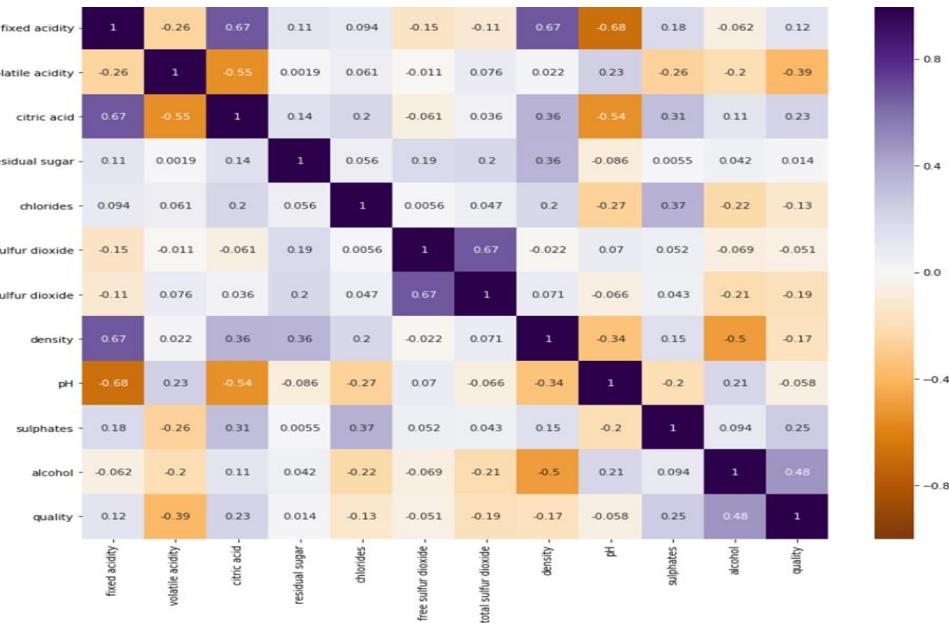


Figure 2: Features Correlation Heatmap using Seaborn

We found that we could improve the accuracy of the logistic regression model by deleting three features: "Citic Acid" because it was positively correlated with "Fixed Acidity" and negatively correlated with "Volatile Acidity" and "pH". We deleted "Fixed Acidity" because it was highly correlated with "Density" and negatively correlated with "pH". We also deleted the "Free Sulphur Dioxide" because it correlated with "Total Sulphur Dioxide". The second dataset i.e. the Breast Cancer Diagnosis Wisconsin had 16 missing values. We deleted these rows from the dataset before training the model. We also deleted the column "sample code number" because ID numbers do not contribute to predicting the occurrence of breast cancer. This step also possibly eliminates any ethical or privacy concerns because the modified training dataset does not contain ID numbers that could possibly be used to identify patients. The dataset also had values 2 and 4 for the labels benign and malignant which needed to be converted into binary labels. The Breast Cancer dataset contained 283 rows that belonged to class 1 (positive) and 444 rows that belonged to class 0 (negative). Features of Breast Cancer Diagnosis Wisconsin dataset:

Out[495]:

	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bland Chromatin	Normal Nucleoli	Mitoses	Class
count	683.000000	683.000000	683.000000	683.000000	683.000000	683.000000	683.000000	683.000000	683.000000
mean	4.442167	3.150805	3.215227	2.830161	3.234261	3.445095	2.869693	1.603221	2.699854
std	2.820761	3.065145	2.988581	2.864562	2.223085	2.449697	3.052666	1.732674	0.954592
min	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	2.000000
25%	2.000000	1.000000	1.000000	1.000000	2.000000	2.000000	1.000000	1.000000	2.000000
50%	4.000000	1.000000	1.000000	1.000000	2.000000	3.000000	1.000000	1.000000	2.000000
75%	6.000000	5.000000	5.000000	4.000000	4.000000	5.000000	4.000000	1.000000	4.000000
max	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	4.000000

Table 2: Features of the breast cancer dataset

We performed feature scaling (normalization) on both datasets by dividing each column by the maximum value of the column. No other changes were made to the datasets before training.

3 Implementation

The Logistic Regression Classifier was implemented using Gradient Descent. We implemented a numpy vectorized version (1) of the gradient descent algorithm by using the matrix form of the Gradient Descent update rule. This vectorized implementation runs much faster than an implementation that uses surface level for-loops in Python because all the computation is handled by numpy. The gradient descent update rule presented in the slides includes a summation term of multiplying row vectors of X by the weights vector W. We performed a matrix multiplication of the feature matrix X (dim: $m \times n$) and the weights vector W (dim: $m \times 1$) which produces a $m \times 1$ vector that will be passed into the sigmoid function to output a $m \times 1$ probability vector of each example belonging to class 1. This matrix form implementation was adapted from the matrix form of the cost function for Linear Regression. For the second model, LDA, we achieved faster runtime by using built-in numpy functions to calculate column-wise means and covariance matrices. The shared covariance matrix Σ was calculated by splitting the feature matrix X into rows that belong to class 1 and 0 and calculating the covariance matrices for each part X_n using $\text{np.cov}(X_n.T)$.

The sum of these 2 covariance matrices divided by (total number of examples - 2) gives us our covariance matrices without using any for-loops for calculating sums.

4 Results

We varied the learning rate α for logistic regression between 0.000001 and 1 in order to see the effect that it had on accuracy:

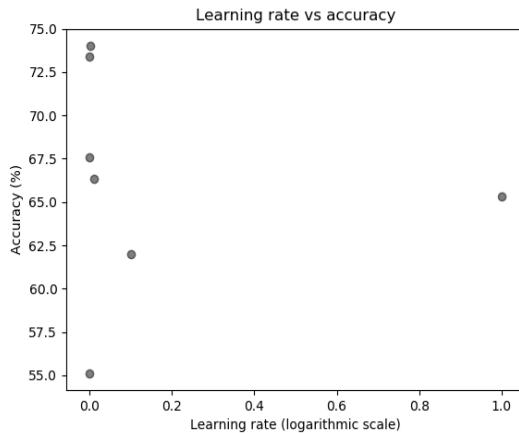


Figure 2: Learning Rate and Accuracy for Logistic Regression (for the range 0.000001 and 1)

From this experiment, we hypothesized that the optimal learning rate for gradient descent convergence must be between 0.0001 and 0.01. So we performed another test, varying the learning rate α from values between 0.0001 and 0.001 and testing the 5-fold accuracy on each value.

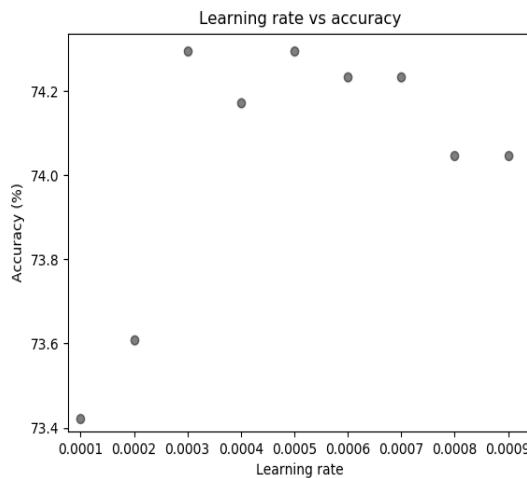


Figure 3: Learning Rate and Accuracy for Logistic Regression (for the range 0.0001 and 0.001)

It looks like 0.0002 is the learning rate that produces the best accuracy for our implementation of Logistic Regression on the wine dataset.

All algorithms were run with k-fold cross validation using 5 folds, training on 80% of the data and testing on 20% of the data 5 times and averaging the results. We built a script that performs 5-Fold Cross Validation on the entire dataset and returns the average accuracy across 5 folds. We found that our Linear Discriminant Analysis model was significantly faster to train than the Logistic Regression model for both datasets. Both models performed better on the Breast Cancer dataset.

The exact runtimes and accuracies for both models are listed below:

	Red Wine Quality		Breast Cancer Diagnosis	
	Logistic Regression	Linear Discriminant Analysis	Logistic Regression	Linear Discriminant Analysis
Accuracy	73.25%	73.29%	85.21%	96.05%
Run Time	27.93 s	0.0092 s	4.008 s	0.008 s

Table 3: Run Time and Accuracy for Logistic Regression and Linear Discriminant Analysis

In order to improve logistic regression's accuracy on the wine dataset we attempted to add the square of every feature as new features. However this only resulted in a 0.35% improvement in accuracy while doubling the size of the feature matrix, so we decided this was not a good approach. We also attempted to find subsets of the wine dataset that were more accurate than the entire dataset. We did this by taking every possible combination of 1 to 11 features and fitting a logistic regression model to those columns and comparing the prediction accuracy achieved to the accuracy achieved by fitting all 11 feature columns. However, the most accurate subset (using just the columns fixed acidity, volatile acidity, residual sugar, chlorides, total sulfur dioxide, density and pH) only achieved an accuracy of 71.79% versus 73.25% when all features were included. We also attempted to remove outliers from the data before training but it did not affect accuracy. We also tried to test the accuracy for different values of gradient descent iterations ranging from 10 to 1 million, we found that 10,000 steps was the optimal number of steps that produced the greatest accuracy. 30,000 steps produces a slightly better accuracy but we chose to keep 10,000 steps as the default value for Logistic Regression because the extra cost of computation wasn't worth a tiny boost in accuracy.

5 Discussion and Conclusion

LDA performed better than logistic regression both in terms of accuracy and in terms of runtime. We could not achieve the same kind of accuracy on the wine dataset that we could on the breast cancer dataset, even with taking feature subsets or adding new features to the breast cancer dataset. This supports the hypothesis that the more complicated the feature set, the harder it tends to be to fit a model to that data. In the future, in order to improve accuracy we could try to implement better stopping criteria for the gradient descent on logistic regression, as right now we are just having it perform a set number of iterations. A better implementation would likely be to have gradient descent run until it reaches a point where the magnitude of the update it is applying to the model's weights fall below a certain value (or it reaches a maximum number of iterations, to prevent edge cases).

6 Statement of contribution

All of us worked on the tasks together.

7 References

1. Xing, E. (September 10, 2014). Lecture 2: Advanced Introduction to Machine Learning [PDF]. Retrieved from <https://www.cs.cmu.edu/~epxing/Class/10715/lectures/lecture2-LR-annotated.pdf>