ML Algorithms from Scratch
By Aleezah Athar and Umaid Ahmed

In this assignment, we built logistic regression and Naive Bayes models from scratch in C++ instead of using the functions commonly found in R. The data we used was from titanic.csv.
This was the output from a run of the Logistic Regression model:

```
Starting to build the logistic regression model
Opening titanic.csv
The training time taken was 1994 milliseconds.

Coefficients:
intercept: 0.999877 slope:  -2.41086

The metrics are as follows:
Accuracy: 0.784553
Sensitivity: 0.695652
Specificity: 0.862595
```

As seen in the output, the equation is y=0.999877-2.41086x. This model has an accuracy of 0.785, a sensitivity of 0.696 and a specificity of 0.862. This information indicates that it's a good model.

This was the output from a run of the Naive Bayes model:

```
Starting to build Naive Bayes Model
The training for Naive Bayes Model have ended!

The training time taken was 0.27 milliseconds.

     0             1
0.420866       0.579134
0.79383 0.20617
0.871238       0.128762
0.226168       0.773832
0.145842       0.854158
0.165216       0.834784

The metrics are as follows:
Accuracy: 0.784553
Sensitivity: 0.695652
Specificity: 0.862595
```

The output shows the training time, the first 5 probabilities output from the testing model, and lastly the metrics. This model has an accuracy of 0.785, a sensitivity of 0.696, and a specificity of 0.862 indicating that it is a good model. The rows under column name 0 (zero) show the probability that the

passenger did not survive and the rows under column name 1 show the probability of those who survived. The training time as compared to the logistic regression model was 7,385 times faster, while both of them had the same level of accuracy, sensitivity, and specificity. The Naive Bayes model tends to fall off for larger sets of data, in that case, the logistic regression model is preferred. Naive Bayes makes the naive assumption that each predictor is independent of the other and Naive Bayes tries to make guesses for the data it was not trained for.

Generative vs Discriminative Classifiers

Generative classifiers try to model how a particular class would generate input data. When these classifiers are presented with a new observation, it makes an attempt to determine which class would have most likely produced the observation. Naive Bayes, which we did in this assignment, is actually an example of generative classifiers. As is Gaussian Mixture Model and various others.
Discriminative classifiers, on the other hand, learn which input features are most helpful in differentiating between the various potential classes. For example, to distinguish between dogs and cats, it would realize that all dog images have a collar and would learn that having a collar means the image is that of a dog, as stated on medium.com. Logistic regression, which we also modeled in this assignment, is an example of a discriminative classifier. As are Random Forest and Neural Networks.

Both approaches utilize conditional probability to categorize, but to produce conditional probability, they must first learn different kinds of probabilities. Discriminative classifiers are also considered to be more accurate according to Akanksha Malhotra because she says it tries to directly solve the classification task, rather than trying to solve a general problem as an intermediate step as generative models do.

Reproducible Research in Machine Learning

Reproducible research in machine learning refers to being able to get the same results when you run your algorithm repeatedly on a dataset. The goal is to create research that is transparent and verifiable so that others can build upon your work and verify your results.
“Reproducible Research for Scientific Computing: Tools and Strategies for Changing the Culture” from https://staff.washington.edu/rjl/pubs/cise12/CiSE12.pdf tells us that the time it takes to clean data and document it for release and reuse is the “single biggest barrier to sharing code and data”. This could explain why people have trouble making their research reproducible. Furthermore, the article states that “reproducible computational science must be recognized as standard practice” for many reasons.
According to decisiveedge.com, reproducibility adds value to any continuous integration or continuous delivery cycle, allowing these activities to proceed smoothly so that in-house changes become routine. Additionally, reproducibility helps reduce errors and ambiguity, ensures data consistency, and creates trust and credibility with the ML project. Reproducibility can be achieved by using version control, keeping detailed documentation, sharing data, using standardized evaluation metrics, and making your code open source.