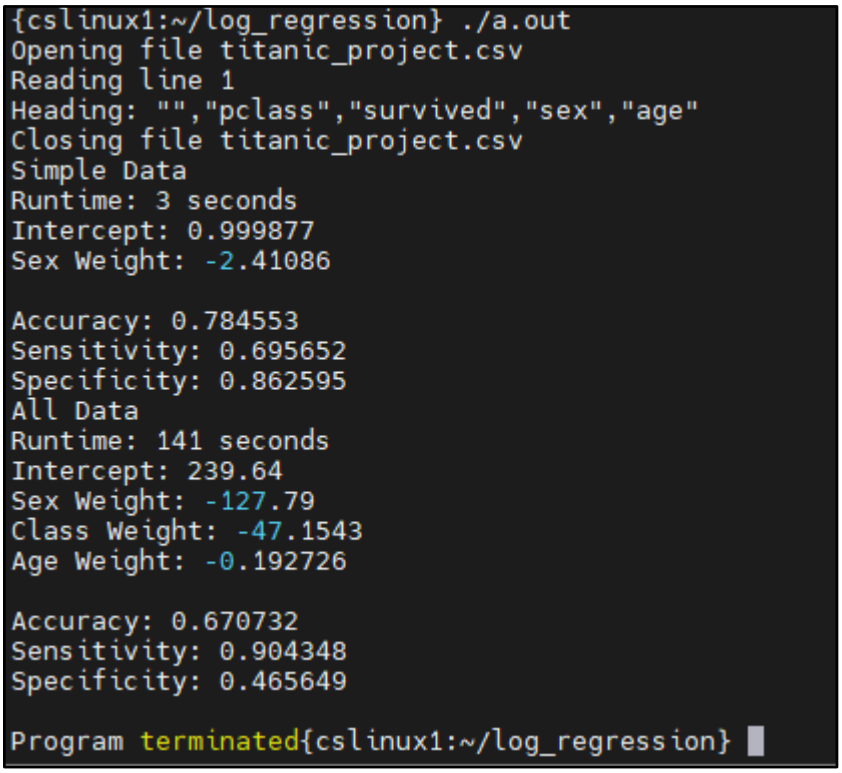
Logistic Regression

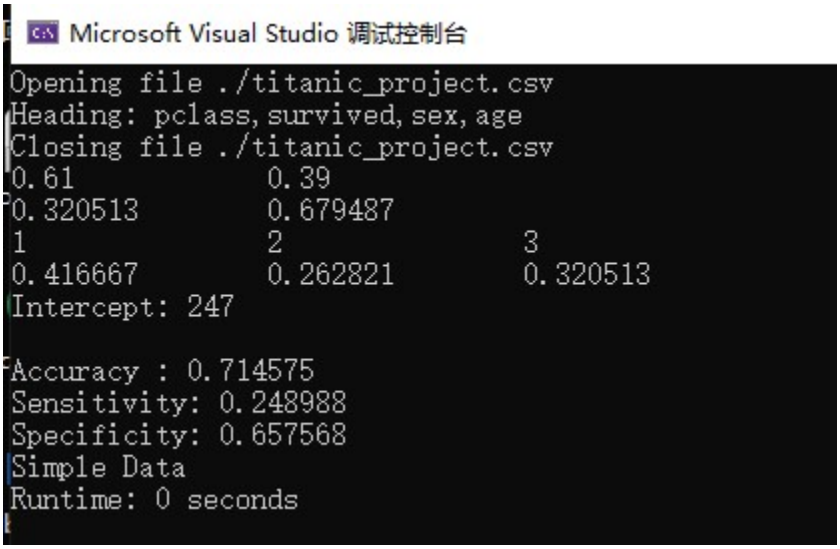
Code Output:



Logistic regression tends to work well on large datasets, and this iteration of the model performed admirably with one predictor, resulting in an accuracy of 0.785. The runtime was fairly short given the amount of matrix operations, namely multiplication, thanks to the check to see if the weights did not change significantly between iterations. The operations themselves took plenty of time, as the code was put together rather hastily, and the matrix operations were done with manually-created functions on nested vectors rather than existing math libraries. The poor runtime of the functions can be seen with the following model, which took several minutes because the weights never quite settled. When the rest of the predictors were added, the accuracy degraded significantly to 0.671. Logistic regression is susceptible to high variance, leading to overfitting on training sets, which would explain why the accuracy became worse with more data. The initial model was more specific than it was sensitive; that is, the model correctly predicted more negative cases than positive ones on the test data.

Naïve Bayes

Code Output:



Naive Bayes performs well on small-scale data and can handle multiple classification tasks. The code first calculates the survival rate and the proportion of different pclass and gender in the survival rate. Because the code is not very concise, it causes some repeated reading. When making predictions on the test data, the accuracy rate is 0.714, which is slightly better than logistic regression. The reason may be because the probability has been calculated in the training set, including the distribution of the sample itself, and each feature is relatively independent. So it is better than logistic regression on small-scale data. The model has low sensitivity and high specificity. The reason may be that it is affected by other variables each time the probability is calculated, resulting in errors in the training data.

Classification

In machine learning, there are two broad categories used to describe classification algorithms, which assign entities to qualitative groups based on input parameters. The first is generative classification; these algorithms focus on what ties data together (Yildirim, 2020). Generative classification is accomplished using joint probability distributions, or the probability that a given feature occurs with a target. The Naive Bayes model is a generative classifier that uses class labels and conditional probability among these classes, or the probability of an unknown label given known data. This form of classification can be used to generate new data, hence the name. Generative classifiers are more susceptible to bias from outliers than the following category.

The other category is discriminative classification; these algorithms focus on what sets data apart (Yildirim, 2020). Discriminative classification makes direct use of the conditional probability distribution, rather than combining conditional probabilities for a joint distribution. Logistic regression is a discriminative classifier, dividing classes using the logarithm of odds and a sigmoid normalization of data. These models cannot create data, but they are also damaged less by outliers; instead of heavily affecting joint probability, an outlier will simply be classified incorrectly.

Reproducibility

Reproducibility is one of the most important metrics in judging the usefulness and trustworthiness of machine learning research. In the context of machine learning, it means that the algorithms can be recreated or copied in such a way that the same results are achieved (Hemant, 2020). Without the ability to recreate the workflow, it can be incredibly difficult and costly to adjust, implement, or optimize the learning system. It may exist in a context that cannot be applied to other use cases, or it may have unintended effects in the place that it is implemented, although that issue overlaps with explainability. Reasons for poor reproducibility include secrecy for military or commercial reasons, in which case reproducibility likely does not concern the authors. In an academic setting, code and training data are provided in research papers covering new findings in machine learning, though errors in the code can bring reproducibility and the contents of the paper as a whole into question (Kapoor & Narayanan, 2022). These errors can be bugs or unintended data leakage, a leading root of problems in machine learning science.

In general, additional documentation is a surefire way to improve reproducibility. The way models are trained should be well-documented, including the choice of features, hyperparameters, and data (Hemant, 2020). Any changes in the previously mentioned attributes should be noted with versions or timestamps. Additionally, meta factors such as the software

environment, packages, and choice of hardware can have impacts on the implementation of a machine learning workflow, and thus these things should be documented. In the scientific community, causes of data leakage include intertwining training and test sets as well as illegitimate features (Kapoor & Narayanan, 2022). These issues have been well documented and should be caught in the review process, though they can be avoided earlier with careful planning and documentation.

References

Hemant, P. (2020, April 7). *Reproducible machine learning*. Towards Data Science. Retrieved from https://towardsdatascience.com/reproducible-machine-learning-cf1841606805

Kapoor, S., & Narayanan, A. (2022). Leakage and the Reproducibility Crisis in ML-based Science. *Princeton*. https://doi.org/10.48550/ARXIV.2207.07048

Yildirim, S. (2020, November 14). *Generative vs Discriminative classifiers in machine learning*. Medium. Retrieved from https://towardsdatascience.com/generative-vs-discriminative-classifiers-in-machine-learning-9ee265be859e