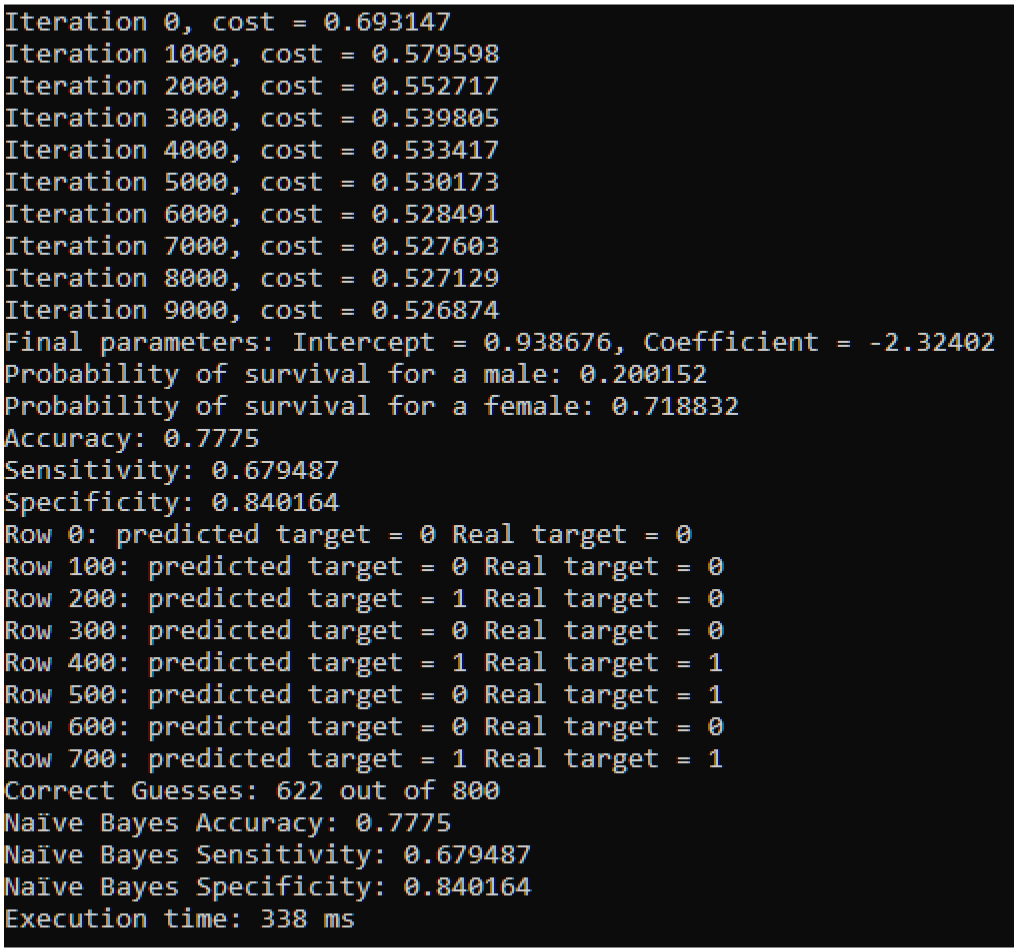
Colton Townsend
3/3/2023
CS 4375.004

C++ Algorithms from Scratch

a. Output to cout:



```
Iteration 0, cost = 0.693147
Iteration 1000, cost = 0.579598
Iteration 2000, cost = 0.552717
Iteration 3000, cost = 0.539805
Iteration 4000, cost = 0.533417
Iteration 5000, cost = 0.530173
Iteration 6000, cost = 0.528491
Iteration 7000, cost = 0.527603
Iteration 8000, cost = 0.527129
Iteration 9000, cost = 0.526874
Final parameters: Intercept = 0.938676, Coefficient = -2.32402
Probability of survival for a male: 0.200152
Probability of survival for a female: 0.718832
Accuracy: 0.7775
Sensitivity: 0.679487
Specificity: 0.840164
Row 0: predicted target = 0 Real target = 0
Row 100: predicted target = 0 Real target = 0
Row 200: predicted target = 1 Real target = 0
Row 300: predicted target = 0 Real target = 0
Row 400: predicted target = 1 Real target = 1
Row 500: predicted target = 0 Real target = 1
Row 600: predicted target = 0 Real target = 0
Row 700: predicted target = 1 Real target = 1
Correct Guesses: 622 out of 800
Naïve Bayes Accuracy: 0.7775
Naïve Bayes Sensitivity: 0.679487
Naïve Bayes Specificity: 0.840164
Execution time: 338 ms
```

b. With 10,000 iterations of logistic regression, the coefficient returned by the program is approximately -2.32, and if we increase the number of iterations, the coefficient shall approach a value of -2.42. The value of this coefficient suggests that being male significantly decreases one's likelihood to survive, which lines up with our calculated probabilities as we can see women are about 2.5 times more likely to survive. Naive Bayes reaffirms our findings as the calculated accuracy, sensitivity, and specificity are the same as our calculated values for logistic regression.

c. Generative classifiers and discriminative classifiers are both strategies in machine learning to tackle classification problems. The primary difference between these two strategies is what information they use in order to make predictions. Generative classifiers are concerned with probabilistic models of the joint distribution of input features and output classes while discriminative classifiers directly model the conditional distribution of the output classes given a set of predictors.

Generative classifiers are suitable for applications where input data is limited and generating new samples is important, such as image or text generation. However, generative classifiers are computationally intense to train since they need to model the entire joint distribution. Discriminative classifiers tend to possess higher accuracy, as they model the boundary between classes directly, and are also computationally easier to train. In essence, generative and discriminative classifiers have different strengths and weaknesses, and the strategy engineers will utilize depends on the specifications and concerns around the problem.

Sources

Goyal, C. (2023, February 14). *Machine learning models: Descriptive & generative ML models*. Analytics Vidhya. Retrieved March 4, 2023, from https://www.analyticsvidhya.com/blog/2021/07/deep-understanding-of-discriminative-and-generative-models-in-machine-learning/

d. Reproducibility is crucial in machine learning as it ensures that the results obtained by an engineer can be studied, verified, and replicated by others. Reproducibility enables researchers and engineers to build upon the work of others, improving the quality and reliability of the work. In addition, reproducibility helps the credibility and transparency of the work, leading others to quickly identify errors and assist in optimization.

   There are many ways to implement reproducibility in machine learning, some of the most significant are:

   1. Experiment tracking, which is concerned with the parameters fed to the machine. Especially with AIs that train in an iterative process, every parameter value changed

and other forms of fine-tuning should be recorded so others can understand the fine process that led to the results.

2. Documentation, which is concerned with the environment of the program including its dependencies, library versions, and hardware. Proper documentation also goes a long way to help others understand the creator's code and engineering process.
3. Version control, which is enabled by programs and systems like Git, which document the changes to code and data over time. Such systems tend to also enable collaboration among programmers, assisting the goal of reproducibility.
4. Data sharing, which is concerned with the data fed to the machine learning AI so others can verify the results. Data sharing also enables researchers to run the data through other methods or AIs so they can affirm or reject the conclusion.
5. Package management systems, which is concerned with dockers and other systems that can reliably handle packages and dependencies so the environment of the program is better controlled.
6. Avoiding non-deterministic algorithms, which is concerned with different results being produced on different runs. Since reproducibility requires others to arrive at the same results as we do, we want to avoid algorithms that could randomly cause significant deviations in the results.

Sources

(Academic Source) LeVeque, R. J., Mitchell, I. M., & Stodden, V. (n.d.). *Reproducible research for scientific computing: Tools and strategies ...* Retrieved March 4, 2023, from https://staff.washington.edu/rjl/pubs/cise12/CiSE12.pdf

Onose, E. (2023, January 26). *How to solve reproducibility in ML*. neptune.ai. Retrieved March 4, 2023, from https://neptune.ai/blog/how-to-solve-reproducibility-in-ml