Airline Passenger Satisfaction Classification

Machine Learning Final Project Report

Anish Kurani

ITCS-3156

December 9rd, 2024

1 Introduction

1.1 Problem Statement

I am looking to solve the problem of negative passenger flight experiences. The objective is to develop a model to help classify whether a passenger will have a satisfactory flight experience using various factors such as delays, legroom, etc. Developing a successful model for this purpose can enable airlines to optimize their flight configurations to maximize customer satisfaction while minimizing costs. In other words, this model can identify which details are most relevant to passengers and enable airlines to cover them. The implementation of my solution can be found at https://github.com/QuantumCubed/ml-final-project.

1.2 Motivation and Challenges

I am a big aviation enthusiast and plan to get a pilot's license eventually. As someone who flies (as a passenger) relatively often, I have experienced less than satisfactory service on my flights. As this is an industry-wide issue, I wanted to use the machine learning algorithms we learned this semester to provide a potential solution for this problem. One challenge with this project is the highly subjective nature of the problem. It is very difficult to identify why someone may or may not enjoy their flight experience. I am trying to solve a binary classification problem by using a few factors. This alone creates a bias because it is impossible to account for why someone may either like or dislike their experience. Another challenge is that even if the model does work as intended, some solutions to this problem might not be easy to implement. For example, if it is found that passengers find legroom to be a massive factor in their flight experience, it is a variable that airlines can change quickly and inexpensively.

1.3 Approach

I chose to approach this problem as a binary classification problem. This is the best way to develop this model for two reasons. Firstly, the dataset I used provides the classes 'neutral or not

satisfied' and 'satisfied'; the dataset is already optimized for binary classification. Secondly, passengers will either enjoy or dislike their flight experience, so it would be best to focus on ensuring that they fully enjoy their experience; realistically, there is no need to have other classes such as 'partially enjoyed' or 'somewhat.' The two algorithms I used to solve this binary classification problem were a logistic regression and a Deep Neural Network. I chose logistic regression for its simplicity, especially because the algorithm excels at binary classification. I chose a DNN as my second algorithm because I wanted to see if the hidden layers could capture more complex relationships between the input features and see if that would improve the accuracy.

2 Data/Environment

I am using the Airline Passenger Satisfaction dataset, which contains 24 features and over 100,000 samples. The dataset includes categorical features such as ***class***, ***gender***, ***customer type*** and **numerical** features such as ***age***, ***flight distance***, ***cleanliness,*** etc. I used a train split of ***0.80%***, leaving ***0.20%*** for validation and testing. For the data preprocessing, I split the features into categorical and numerical columns to determine which ones I needed to use in the actual **X** value. I did this with SKlearn's preprocessor transformer, enabling me to use the standard scaler. Specifically, I determined that ***['Gender,' 'Customer Type', 'Type of Travel', 'Class']*** are categorical features and ***['Age', 'Flight Distance', 'Departure Delay', 'Arrival Delay'] and ['Service', 'Convenience']*** are numerical features. I then manually added bias to the post-processed **X** input. For the data-processing visualizations, please look at GitHub, as the images must be enhanced for clear viewing.

3 Method

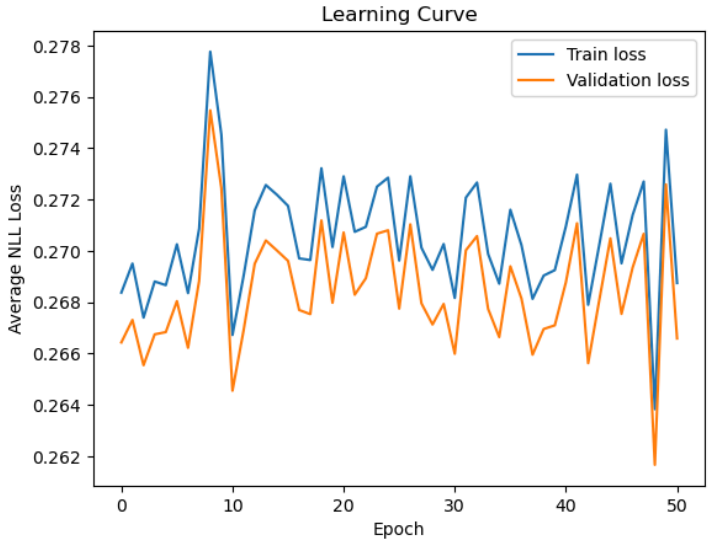
As mentioned above, the algorithms that I used are Logistic Regression and a Deep Neural Network using TensorFlow. I used the binary cross-entropy loss function for the logistic regression solution with the sigmoid activation function. Initially, I tried to use the softmax regression extension of logistic regression; however, I was having issues with the output (most

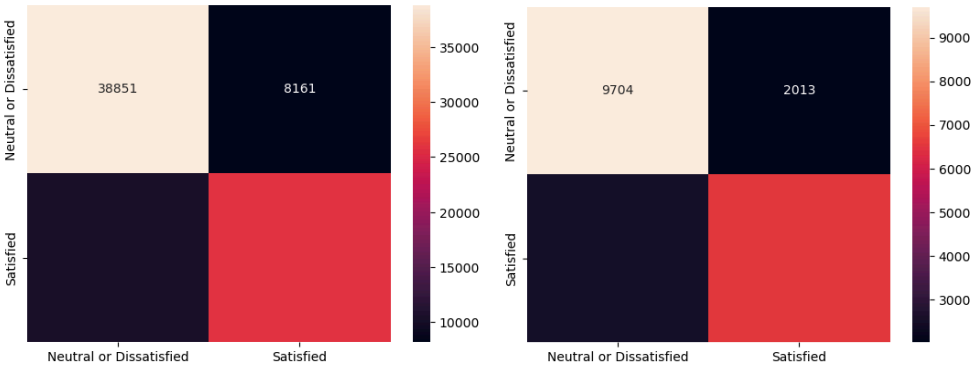
likely because Softmax is better for multiclassification problems), so I designed a simpler binary classifier. I modified my initial Softmax solution; the primary difference is how the weights are updated per epoch. Instead of using the Softmax function to determine the class probabilities, I use the Sigmoid function directly. This difference is so consequential to the solution because the Signmoid function squashes the linear combination of ***X_b*** (the current batch of X) and the randomly initialized weights ***W*** between 0 and 1. When implementing the prediction function, I also had to change the return value of the prediction; instead of returning the arg max of the linear combination, I had to return the Sigmoid output of the linear combination. To return the classification prediction, I applied a threshold of 0.5 to the Sigmoid output to determine the class. To implement the Neural Network, I used TensorFlow with 3 dense layers (16 units, 8 units, 4 units), a dropout layer per layer, and a final output layer (1 unit). For each dense layer, I used the ReLu activation function because this activation function is non-linear and can be used to capture complex relationships between features; this function is also known for being consistent with handling the vanishing gradient problem. The final output layer uses the sigmoid activation function; like logistic regression, this squashes the probabilities between 0-1. Like the logistic regression, I also had to apply the 0.5 threshold to the probabilities to return the class labels.

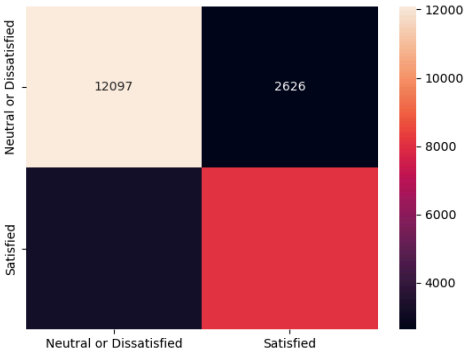
4 Results

The experimental setup (hyperparameters) and data split I used for the logistic regression and neural network are identical. I used the following setup: ***alpha=0.03, batch_size=64, epochs=51, and seed=0***. The rationale for using the same hyperparameters is to see if there is a noticeable accuracy difference between the two solution implementations with the same hyperparameters. The logistic regression algorithm produced the following results: ***Train Accuracy: 77.70%, Validation Accuracy: 78.20%, and Testing Accuracy: 77.59%***. The learning curve and

confusion matrix visualizations can be seen below.

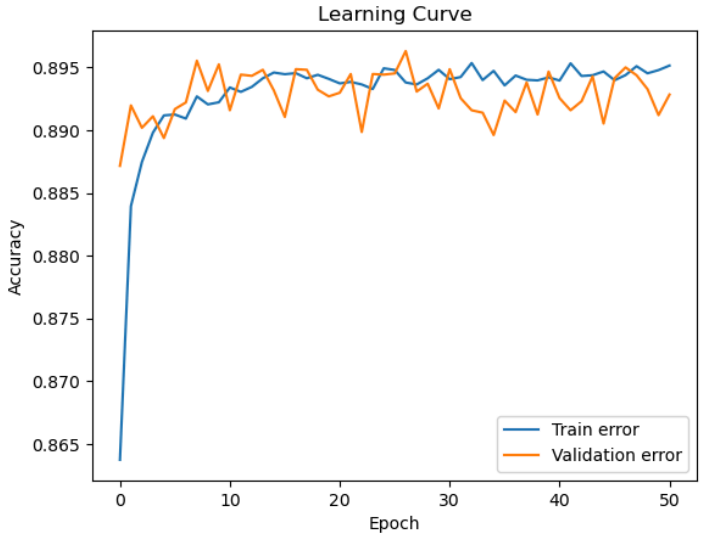


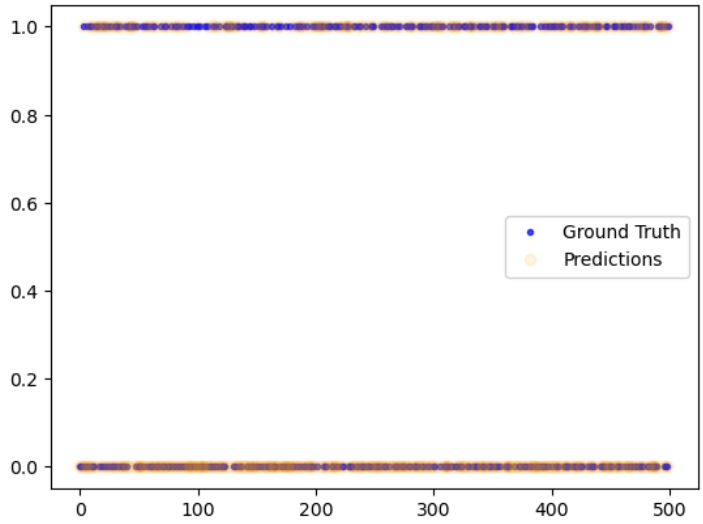


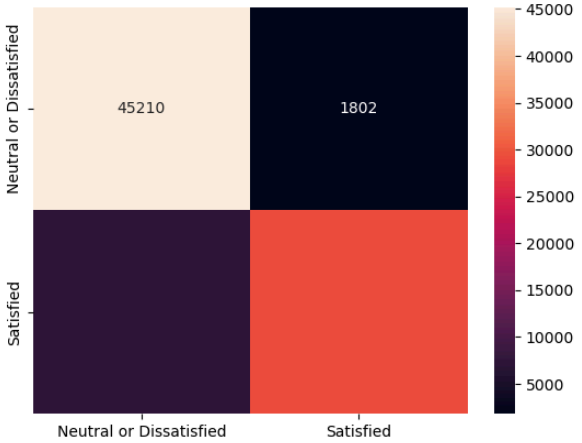


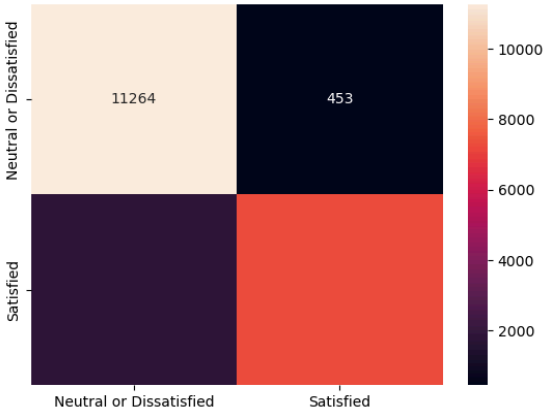
The logistic regression algorithm produced the following results: ***Train Accuracy: 89.41%, Validation Accuracy: 89.30%, and Testing Accuracy: 89.13%***. The learning curve, prediction,

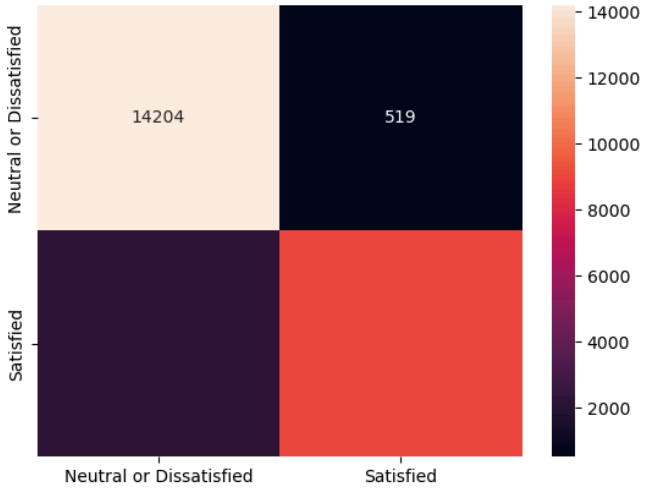
and confusion matrix visualizations are below.











With the same hyperparameters, there is about a 12% accuracy increase with the neural network solution. This is likely because the ReLU activation function is used for the different layers instead of directly using the sigmoid activation function. I also tried adjusting the hyperparameters for the neural network itself, such as **batch_size**, **epochs,** and **alpha**. This is the optimal setup I found for the neural network and disabling the dropout layers. Although my network design can use dropout layers (to prevent overfitting), the lack of enough feature data to take advantage of the dropout layers causes a loss in accuracy. Attempts to change the hyperparameters decreased overall accuracy; For example, using the configuration ***alpha=0.01, batch_size=32, epochs=31, and seed=0*** decreased the accuracy by ***4%*** to ***84%,*** and the processing time increased by **2** minutes.

Conclusion

After developing both solutions and having an accuracy of ***89%***, I think the project is a success and can be used to complete the objective that I set out, identifying ways to improve the passenger experience. I have learned a lot more in-depth about these algorithms. For example, I learned that softmax regression is not the most optimal solution for a binary classification problem and learned how to modify my code. I learned why ensuring that the neural network architecture is balanced is important so that the layer-to-feature ratio is not unbalanced. One major challenge that I faced was getting the data preprocessing to work. I had challenges figuring out which features to drop or label as a categorical variable. It was important to figure out how to solve this issue because the data preprocessing step is the biggest step of the machine learning pipeline. If the input data is not processed correctly, the output data may be useless.

Acknowledgments & References:

I used ChatGPT to assist with debugging code I created myself and for general-purpose machine-learning questions. I also used notes and code from previous homework assignments, specifically Homework 9 - Logistic Regression and Homework 10 - Neural Networks.

- Reda, sohila. “Airline Passenger Satisfaction.” *Kaggle*, 23 Sept. 2024, www.kaggle.com/datasets/sohilaredaeldally/airline-passenger-satisfaction/data.
- OpenAI GPT model mini4o (Unable to cite link with images)