Review Sentiment Analysis

1. Introduction

In this project, we have been supplied with several thousand single-sentence reviews, collected from Amazon, Yelp, and IMDB. With the provided data, we developed binary classifiers that can generate sentiment labels for new sentences, automating the assessment process.

2. Pre-processing

Before we train the dataset into the model, I have to transform the dataset to a better format, so the dataset can clearly recognize by the given model. To do that, I used natural language processing and vectorization to transform the given dataset.

First, I disregarded all punctuations from the given text and changed them to lowercase. Then, I split the given text into small units by using tokenize from Natural Language Toolkit. I iterated through tokenized text and removed “stop words” from the list. For the final step, we used PorterStemmer to remove various suffixes and prefixes. So, the given text changed to its word stem. For clarification, check the below example:

Waste of 13 bucks. (Given)

waste of 13 bucks (Remove punctuation and change to lowercase)

[waste, of, 13, bucks] (Tokenize)

[waste, 13, bucks] (Stop words)

[wast, 13, buck] (PorterStemmer)

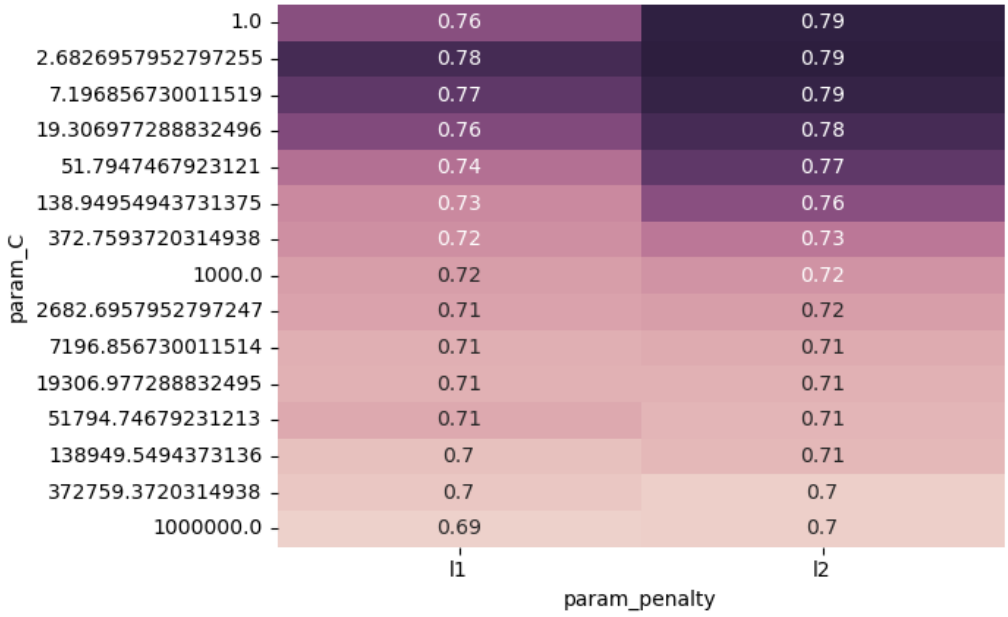
wast 13 buck (return value)

After the pre-processing, I utilized TF-IDF features to reform sentence datasets as a matrix. This is important because it is important to consider the group of words together

rather than individual words during sentiment analysis. Therefore, the model clearly classifies the keyword from the vectorized data.

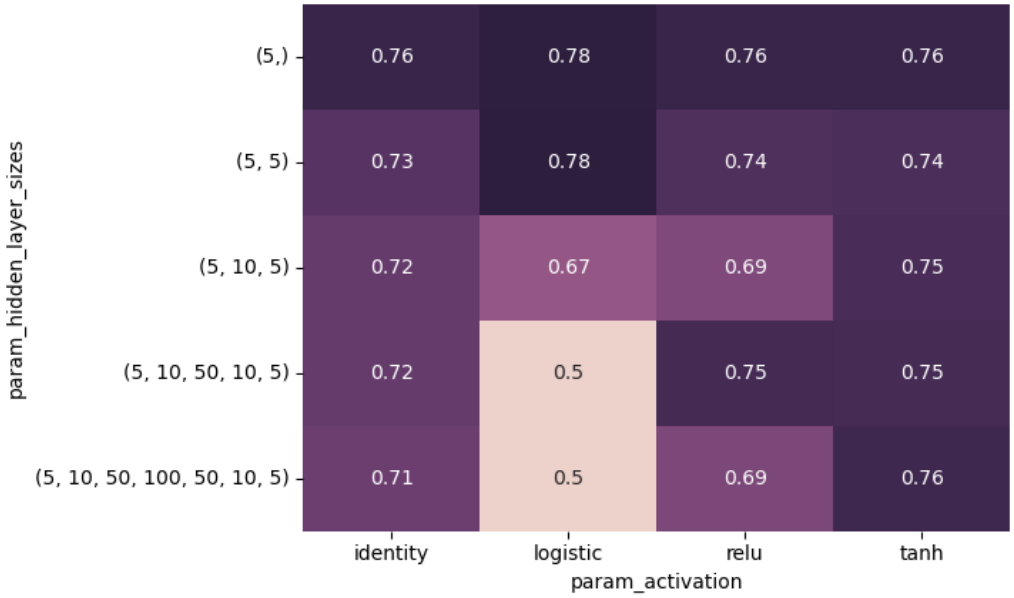
3. Logistic Regression

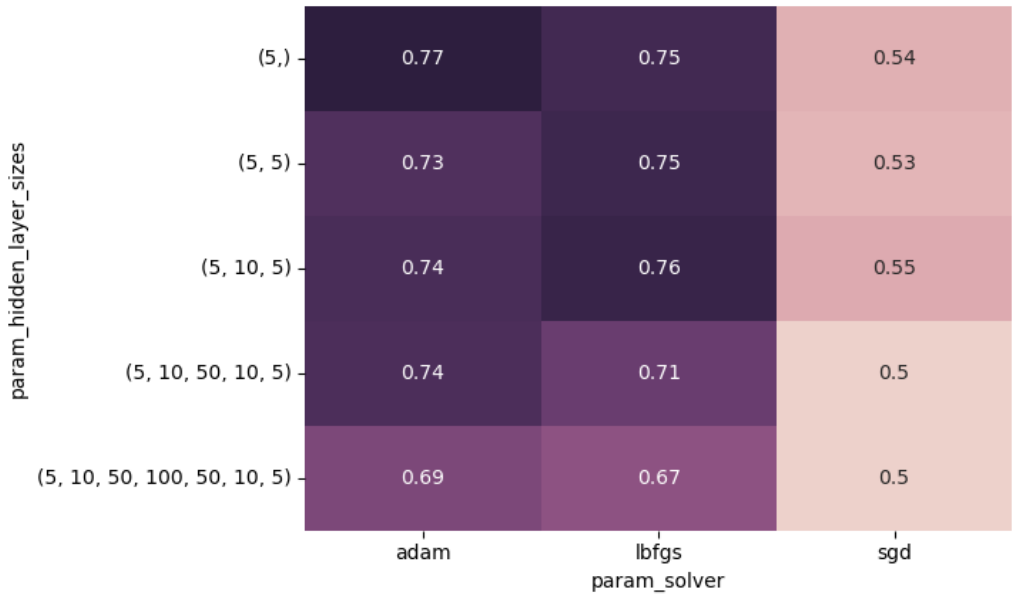
For the Logistic Regression model, I applied various regularization penalties and regularization methods as hyperparameters. I created 15 different samples in the logspace that vary from 1.0 to 1000000.0 for the regularization penalties. Then, I used L1, which is a Lasso regression, and L2, which is a ridge regression, for the regularization methods. While I was creating the model, I used liblinear as a solver. Moreover, I applied GridSearchCV to implement 5 folds cross-validation with various hyperparameters. After I fit the model into the training data, the L2 model performs better than L1 according to the heatmap. The best model that was proven by the heatmap was the L2 model with 1.0 as a regularization penalty. The performance of the model was an accuracy of 0.794 and a standard deviation of 0.0183 while the error rate is 0.195, and AUROC is 0.882.



4. Neural Network (MLP)

For the Neural Network Model (MLP), I used various hidden layers and different types of activation and solver to train the model. I built two different models one using four different activations and the other using three different solvers while both models trained on various hidden layers. The hidden layers that I used to train data varied from (5,) to (5, 10, 50, 100, 50, 10, 5). For the model with four different activations, which were identity, logistic, relu, and tanh, the logistic performed best with the smaller number of hidden layers while it performed worse with a large number of hidden layers. However, in the case of tanh activation, it performed equally well.





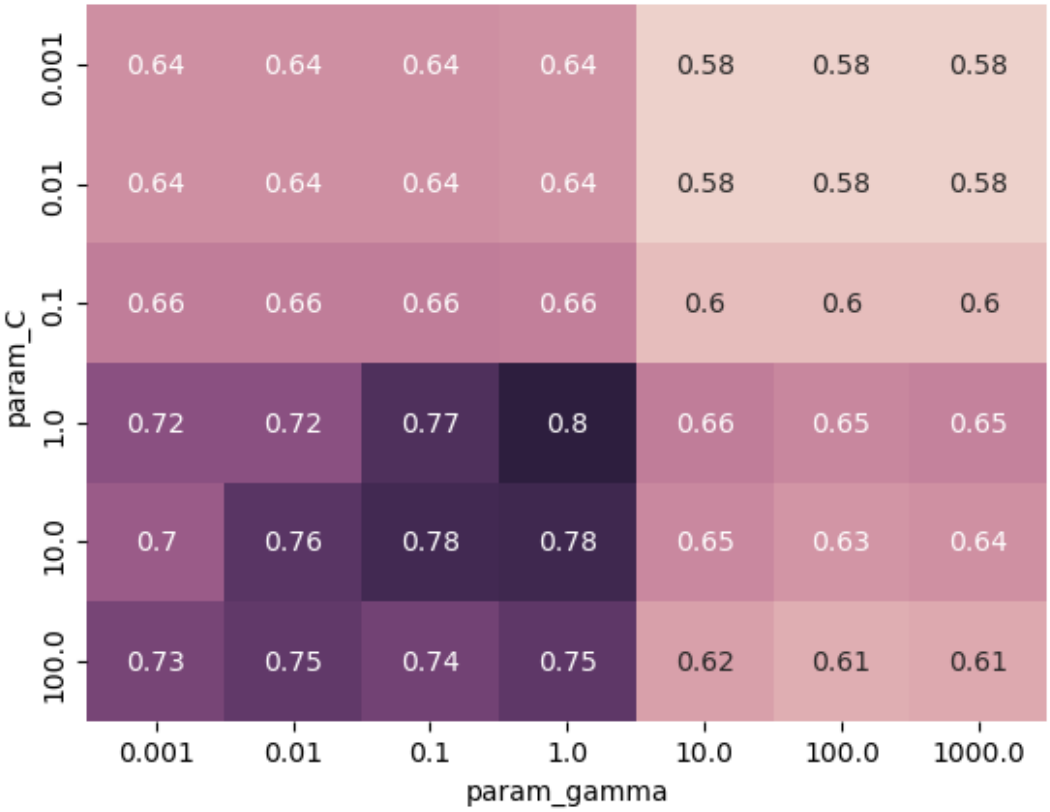
For the model with three different solvers, which were adam, lbfgs, and sgd, the sgd performed worst in all hidden layers. However, adam and lbfgs performed equally in each hidden layer.

To implement 5 folds cross-validation on these models with hyperparameters, I used GridSearchCV as I did in Logistic Regression. By training data, the best model was found with small hidden layers and logistic activation. The performance of the model has an accuracy of 0.775 and a standard deviation of 0.00664 while the error rate is 0.21 and AUROC is 0.866.

5. SVM

For the SVM model, I used a supported vector machine from the scikit-learn library to clarify data because the SVM classifier effectively classifies cluster data and sentiment analysis. The two hyperparameters I applied to build models were C and gamma values. The C parameter varies from 0.001 to 100.0 while the gamma parameter varies from 0.001 to 1000.0. According to the heatmap, the best-performing model has an accuracy of

0.801 and a standard deviation of 0.0784. Overall, the model has an AUROC of 0.883 and an error rate of 0.175.



6. Performance

Among the three models, the SVM has the best performance. Since the SVM classifier has the highest accuracy with the lowest error rate, the risk of overfitting in the SVM classifier is slightly lower than in the Logistic Regression classifier. However, the SVM classifier seems like it has some trouble recognizing combined negative words according to the false positive values. If we resolve this issue by training data in a better format, I think the SVM classifier can provide a better result.