Nathan Glen
MAS4115

Homework 6: Project

Dataset: https://www.kaggle.com/datasets/d4rklucif3r/restaurant-reviews

What I did:

In this project I built 4 models, attempting to classify restaurant reviews from a dataset with 1000 observations and two columns: the review itself and the sentiment associated with it. I wanted to use a supervised learning approach to this problem, so I had to find a dataset with labels already associated with each review. I'll briefly discuss my models and their results in the next paragraphs:

SVM:

Support Vector Machines are very commonly used in classification tasks - they aim to find the optimal hyperplane that separates classes with the goal of getting the largest distance between the boundary and closest data points from each class, with the goal of minimizing classification errors.
I tried two different implementations of SVM, one utilizing the Bag of Words encoding technique and another utilizing the TF-ID vectorizing technique built into sklearn.

Bag of Words SVM Report:

```
0.785
              precision    recall  f1-score   support

           0       0.75      0.82      0.79        96
           1       0.82      0.75      0.78       104

    accuracy                           0.79       200
   macro avg       0.79      0.79      0.78       200
weighted avg       0.79      0.79      0.78       200
```

TFID SVM Report:

```
Accuracy: 0.82
Classification Report:
              precision    recall  f1-score   support

           0       0.77      0.90      0.83        96
           1       0.89      0.75      0.81       104

    accuracy                           0.82       200
   macro avg       0.83      0.82      0.82       200
weighted avg       0.83      0.82      0.82       200
```

Above are the two classification reports for the two separate SVM models. The TFID had a much higher accuracy (total correct observations/ total observations) and f1 score (combines precision and recall). This is expected for me, as TFID is a much more robust algorithm. This is because the bag of words simply counts words and TFID adjusts word counts based on how important a word is.

Logistic Regression:

We didn't really discuss Logistic Regression in this class, but this is another common technique whenever classification problems are talked about. It assigns probabilities to outcomes using the Sigmoid function, and also utilizes a hyperplane to separate two categories.

Report:

```
Accuracy: 0.81

Classification Report:
              precision    recall  f1-score   support

           0       0.76      0.89      0.82        96
           1       0.88      0.74      0.80       104

    accuracy                           0.81       200
   macro avg       0.82      0.81      0.81       200
weighted avg       0.82      0.81      0.81       200
```

These results were very comparable to the previous two, and this doesn't really surprise me. This is because it utilizes the same hyperplane technique to separate decisions, and I imagine under the hood they use a lot of the same algorithms.

CNN:
The Convolutional Neural Network is a model we recently covered in class that is typically used for image classification - it puts inputs through multiple layers of convolution, pooling (downsampling), and fully connected layers to make predictions. I read a medium article that applied CNNs to text classification and wanted to try it out.

Report:

```
Classification Report:
              precision    recall  f1-score   support

           0       0.73      0.81      0.77        96
           1       0.81      0.72      0.76       104

    accuracy                           0.77       200
   macro avg       0.77      0.77      0.76       200
weighted avg       0.77      0.77      0.76       200
```

These results did not really surprise, as like I just stated, CNNs are typically used for **image classification**. So, we ended up with lower accuracy and other metrics for this specific problem.

Testing Models Further:
At the bottom of the *Testing_Classifiers.ipynb* file, I have examples of how the models I trained did on some reviews I wrote. It is interesting to see how the models handle weird phrases, like “This food was fire!”. All of my models classified this as negative, and I think it really goes to show the limitations of these algorithms. The data I included probably did not have any slang like this, and fire is typically a bad thing, so I can easily see how this could be interpreted as so.

Finally, I have an *Interface.py* file that allows a visual interface for you to enter your own reviews and see the classification for the SVM Bag of Words, SVM TFID, and the Logistic Regression models. You can simply run the .py and copy paste the run streamlit command in your command line, and it should open up a localhost in google chrome.