Final Project Documentation

Nate Smolczyk

Presentation Link : https://youtu.be/43ieUdCO8A4

Prerequisites

In order to run this project, you will need python and node/npm installed on your machine. Python for running the server and training the model, and npm for running the UI written in JavaScript/React.

- Python
- Node/npm

Also, you will need to clone my project from GitHub here

Step One: Training the Model (optional)

Prerequisites

1. You will need to install the following python dependencies using pip
    - sklearn
    - nltk
    - joblib
    - pandas
    - matplotlib
    - "pip install *'package-name'*"

This step is optional since I've included the trained model in my project branch. First, I've provided the data with a link to google drive, since the file was too big to add to GitHub. You can find the data here . Once downloaded, unzip the compressed file, and place it in the project folder at the root level.

You can run the jupyter notebook to train the model by opening it in VSCode and clicking "Run All" at the top of the window.

Step Two: Running the Flask Server

Prerequisites

2. You will need to install the following python dependencies using pip
    - flask
    - nltk
    - flask_cors

In a terminal, navigate to the root folder of the project code. Run this command:

"`export FLASK_APP=server.py`"

Now that the variable is set, you can run the app by running this command:

"`flask run`"

You should see a message in the console like this which tells you which port the server is running on:

Step Three: Running the UI

Prerequisites

1.) To install the dependencies for the UI, open another terminal separate from the server, and navigate to the project root branch.

2.) Navigate to the ui project by running this command:

    i. "`cd videogame-classifier-ui`"

3.) Run this command to install all required dependencies:

    i. "`npm install`"

Now, you can run the UI locally by running this command:

    i. "`npm start`"

Your browser should automatically open a page to the UI once it is ready. From here, you can type in your own reviews to classify.

Summary

In my project I was able to successfully set up a web app to categorize movie reviews as either negative or positive. There were three main challenges: differentiating 1 and 2 stars, differentiating 4 and 5 stars, and classifying 3 star reviews. For the first two issues, I combined 1 and 2 star reviews into the “negative” category. Similarly, I combined 4 and 5-star reviews into a “positive” category, and left 3 star reviews as “negative”

I was not able to get very high accuracy trying to classify each of the 5 categories, as seen here (60%):

```
Overall Accuracy: 0.5922413793103448
              precision    recall  f1-score   support

           1       0.59      0.78      0.67      1441
           2       0.65      0.38      0.48      1497
           4       0.60      0.48      0.53      1440
           5       0.57      0.75      0.65      1422

    accuracy                           0.59      5800
   macro avg       0.60      0.59      0.58      5800
weighted avg       0.60      0.59      0.58      5800
```

However, with the 3 category approach, I was able to achieve about 68% accuracy:

```
Overall Accuracy: 0.6875862068965517
              precision    recall  f1-score   support

    negative       0.66      0.89      0.76      2915
     neutral       0.68      0.01      0.02      1428
    positive       0.72      0.82      0.76      2907

    accuracy                           0.69      7250
   macro avg       0.69      0.57      0.52      7250
weighted avg       0.69      0.69      0.62      7250
```

As you can see, neutral (3-star) reviews are hardest to classify since they can go either way. The language used in 3 star reviews is sometimes positive and sometimes negative. This is evidenced by the fact that neutral reviews have 0.01 recall. So it's rare for the model to classify a 3 star review as neutral, but when it does, it has similar precision to the other categories.

I tried a two category approach, by lumping the 3 star reviews with the positive or negative ones. Both methods gave about the same results: (about 66%), which is further evidence that they are difficult to identify as either

Overall Accuracy: 0.6565517241379311

	precision	recall	f1-score	support
negative	0.97	0.11	0.20	2791
positive	0.64	1.00	0.78	4459
accuracy			0.66	7250
macro avg	0.81	0.55	0.49	7250
weighted avg	0.77	0.66	0.56	7250

Overall Accuracy: 0.6682758620689655

	precision	recall	f1-score	support
negative	0.65	0.99	0.78	4340
positive	0.91	0.19	0.32	2910
accuracy			0.67	7250
macro avg	0.78	0.59	0.55	7250
weighted avg	0.75	0.67	0.60	7250

Finally, I wanted to see what kind of accuracy I could get by not considering 3 star reviews at all. This greatly increased my total accuracy: close to 80%

Overall Accuracy: 0.7910344827586206

	precision	recall	f1-score	support
negative	0.72	0.95	0.82	2882
positive	0.93	0.64	0.75	2918
accuracy			0.79	5800
macro avg	0.82	0.79	0.79	5800
weighted avg	0.82	0.79	0.79	5800