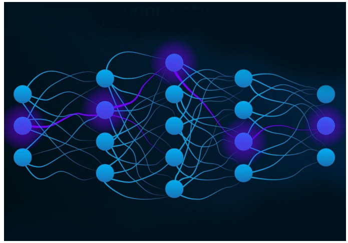
Predicting Planetterp Professors’ Performances



By: Chris Wu
CMSC320 - Intro to Data Science
December 12, 2025

General Overview

ASSOCIATED REPO LINK: ***https://github.com/cheesuschris/320Homeworks/tree/main/Homework4***

- **Objective**: The goal for this project was to build a tool that accepts a text-based review of a UMD professor and predict the professor's associated star rating
- **Data Collection**: Five UMD professors' ratings and reviews were pulled from Planetterp's API – specifically through their associated Python wrapper
- **Fine-tuning**: The data was used alongside Pytorch to fine-tune a model to more accurately predict the correct ratings a professor received based on their reviews
- **Testing**: The transformer was then used to guess how many stars correlated to each review from the data collection step, and these results were compared to the actual star ratings the reviews gave the professors

Data Collection

The dataset consisted of solely reviews and ratings ingested from the following 5 UMD professors:

- Justin Wyss-Gallifent
- Kevin Daniels
- Christopher Kauffman
- Nelson Padua-Perez
- David Mount

706 total reviews and ratings were ingested, and were packed into tuples/columns (look in repo) to be used during fine-tuning

High-Level Approach: Part 1

The problem statement originally informed us NOT to setup the assignment as a supervised learning problem. However, when fine-tuning “distilgpt2” with only reviews-rating strings through unsupervised learning, many hallucinations appeared:

- The professor was acting so badly. Rating: Rating (1-5 stars): 5-5 stars (5 stars): 5-5 stars (5 stars): 5-5 stars (5 stars): 5-5 stars (5 stars): 5
- ÀÀÀÀÀÀÀÀÀÀÀÀÀÀÀÀÀÀÀÀÀÀ
- ?? irks …
- Etc.

Trying to extract the prompted star ratings resulted in using regex resulted in highly inaccurate answers and also MOSTLY didn’t even contain the rating. Thus the next step I took was to treat it as a supervised learning problem even after trying to tune the query prompt & hyperparameters (such as stopping criteria) & changing the pre-trained model all couldn’t seem to make the accuracy (~30% cap) go up.

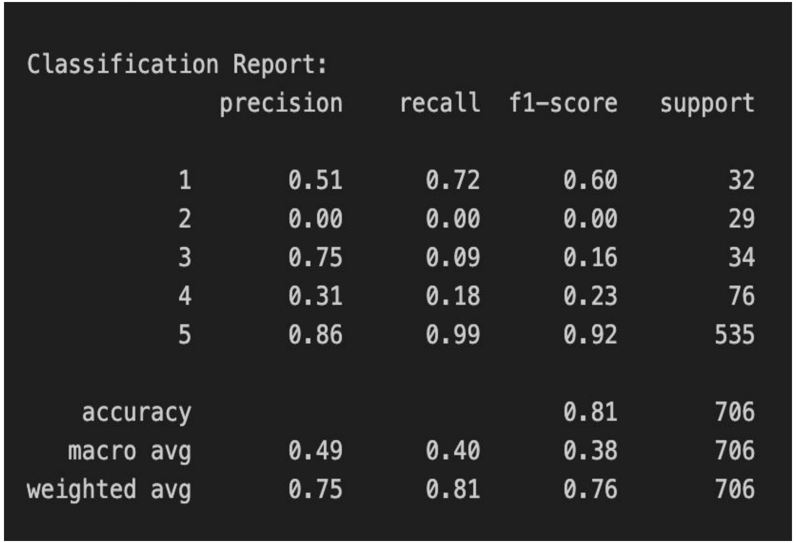
High-Level Approach: Part 2

Using the data as a DataFrame and actually training/testing a “distilbert-base-uncased”model resulted in much better results than unsupervised learning:

- The best accuracy shot up to 76.76%
- Actual ratings (non-gibberish) were outputted
    - No hallucinations with supervised training
- Comparing predictions to actual star ratings were accurate

Overall Success

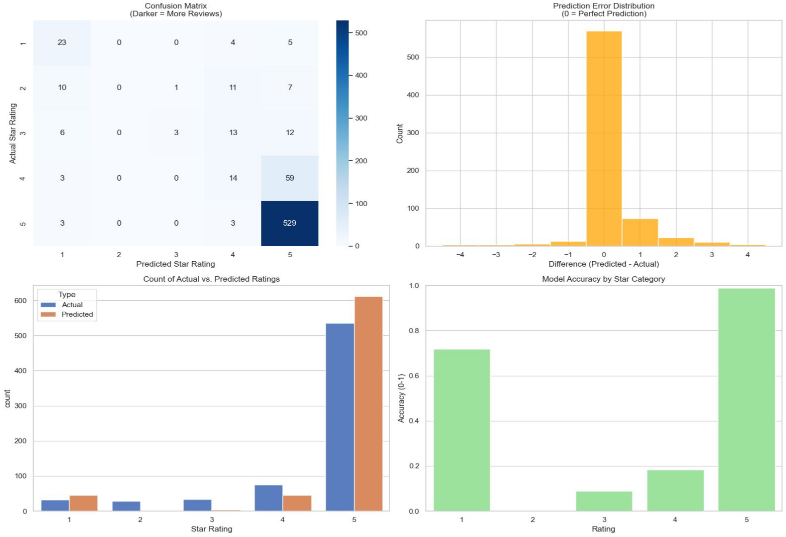
Across the 706 reviews ingested from Planetterp API, a supervised-learning problem to fine-tune the “distilbert-base-uncased” model rather than an unsupervised-learning problem produced the following results:



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

           1       0.51      0.72      0.60        32
           2       0.00      0.00      0.00        29
           3       0.75      0.09      0.16        34
           4       0.31      0.18      0.23        76
           5       0.86      0.99      0.92       535

    accuracy                           0.81       706
   macro avg       0.49      0.40      0.38       706
weighted avg       0.75      0.81      0.76       706
```


Confusion Matrix
(Darker = More Reviews)
Actual Star Rating
Predicted Star Rating
1: 23 0 0 4 5
2: 10 0 1 11 7
3: 6 0 3 13 12
4: 3 0 0 14 59
5: 3 0 0 3 529
Prediction Error Distribution
(0 = Perfect Prediction)
Count
Difference (Predicted - Actual)
Count of Actual vs. Predicted Ratings
Type
Actual
Predicted
count
Star Rating
Model Accuracy by Star Category
Accuracy (0-1)
Rating