Alex Noor

CS506: Data Science Models

Lance Galletti

October 28, 2024

Midterm Writeup and Report

I first want to dive into the processes that were setup in the starter code, and then discuss how I took that setup and improved the model overall to get to my final accuracy score of 0.5525. The starter code used a KNN algorithm to analyze the movie review data points. The main processes I added to the code was feature scaling, dimensionality reduction using PCA, as well as cross validating scores to help analyze the data. In addition, I tweaked the existing code to add more features to review length as a feature and average word length.

Using Professor Galetti's starting code for creating a model to predict movie star ratings based off of reviews, my immediate thought was that in order to increase the accuracy I should alter k from 3. Initially, I observed that a low k-value (k=3) led to high variance, making the model susceptible to noise. By increasing-neighbors to 500, I was able to help KNN fit the curve much better for the data. Next, I wanted to implement Feature Scaling. KNN is sensitive to the scale of data because it relies on distances between points, so implementing feature scaling jumped my accuracy score immensely. Using Min-Max scaling helped normalize feature ranges, ensuring all variables contributed equally to distance calculations. After scaling, I observed an immediate improvement in accuracy, reinforcing the importance of standardizing features in distance-sensitive models like KNN. Something else that I have learned about in this class, as well as my BA305 class is dimensionality reduction using PCA. I did this to manage the high dimensional nature of the data that we were given. This streamlined the KNN algorithm's

performance, allowing it to focus on the most relevant aspects of the data, resulting in faster computation and a more stable accuracy.

We were given an example using KNN so I wanted to see if other models would provide higher accuracy. I tested and played around with both a Random Forest classifier as well as a Gradient Boosting classifier. However when I ran these models, I was producing an accuracy between 0.4 and 0.5, where KNN was much more accurate as long as I increased my number of neighbors. I also played around with text feature extraction using TF-IDF, however it reduced my accuracy score when playing around with different TF-IDF features I noticed when implementing this, I was still getting an accuracy score of 0.53 however it was taking significantly longer to run my code. I decided it was best to scrap this as it caused no improvements to my score and only created crashing issues within my Google Colab.

The final model configuration included a KNN classifier with k=500, scaled features, and key engineered features like review length and average word length. Through these adjustments, I achieved an accuracy of 0.5525. Interestingly, the addition of simple features like review length had a greater impact than more complex TF-IDF vectors, highlighting the importance of interpretable, low-dimensional features in KNN performance.