DS 4400

Project Report

Project Title: Predicting drug ratings from user reviews

TA: Jake Horban

Team Members: Mirah Gordon, Caterina Wang, Genny Jawor

Github Repository (Code)

Video Recording (Presentation)

Problem Description:

User reviews are extremely important to consumers when trying new products. These reviews typically contain two elements: a numerical rating system, and the option to write comments. People generally find comments much more insightful than a single number because they provide information that helps contextualize a good or bad review. This process of looking at reviews is incredibly common for those who have been prescribed a new drug by their physician for a certain condition, or people looking into different drugs to treat a condition they may have. Drugs can have intense side effects on people's bodies, so these reviews help lots of people look at other's experiences with certain drugs to help them evaluate if that drug is beneficial for them, or if it's actually hurting them. However, it is extremely time-consuming to look through and read hundreds or thousands of reviews, so a great solution to help save a consumer's time could be to have an overall rating number which accurately reflects the sentiments of a single review. This number could then be averaged among all reviews to summarize the data.

Our team is attempting to use text drug reviews in order to predict the rating that user gave the drug. This is a classification problem. We have split the problem into binary classification and multi-class classification to compare the results of both approaches. The dataset we are using contains user ratings of the drugs they reviewed on a scale of 1-10. For the multi-class classification problem, we attempted to classify the rating between 1-10 the user who wrote the review gave the drug alongside their review. For the binary classification problem, if a drug was given a rating of 1-5, we classified that as a bad review, giving it a rating value of 0. If the drug was given a rating between 6-10, we classified that as a good review, giving it a rating

value of 1. Then, we attempted to predict the class of the drug review from the other features of the reviews.

Through our research, we've seen that it's possible that the sentiment of a drug review may not necessarily align with the numerical rating given alongside that review. It is important to not only look at the numerical value given, but also the sentiment of the text in the review since a misalignment could make it seem like some drugs work well, even if they give users debilitating side effects, or effective drugs could appear to be less effective than they truly are. It's also important for patients to have a thorough knowledge of their medications. Studies show that patients who exhibit a better understanding of their conditions and medications are able to better adhere to their chronic medications. Many people may be looking at drug reviews on drugs.com, the site the drug reviews in our dataset are from, to gather information and better understand their current medications. Our project idea would consolidate review information and help people looking for insight synthesize sentiment and efficacy of certain drugs quickly and accurately, helping educate them on the very drugs that could potentially be going into their bodies.

References:

1. https://colab.research.google.com/github/BrittonWinterrose/Drug_Review_NLP/blob/master/01_Drug_Review_Dataset_Exploration.ipynb#scrollTo=95pgbDWU2ZDM
2. https://www.medrxiv.org/content/10.1101/2021.04.15.21255573v1.full
3. https://journals.sagepub.com/doi/pdf/10.1177/20101058140230020 7#:~:text=Medication%20review%20provides%20a%20platform,issues%20related%20to%20their%20health

Dataset and Exploratory Data Analysis:

The dataset we are using is the UCI Drug Review Dataset. There were 6 features in the original dataset containing text, numerical, and date feature types. There were 215,063 rows in the original dataset.

Feature Definition:

- Date: representing the date the drug review was written,

- drugName: the drug the user was reviewing
- Condition: the condition the drug was taken for
- Review: the text of the review of the drug
- Rating: the rating the user gave the drug
- usefulCount: the number of users who marked that they thought the review was useful

We needed to clean some of the text reviews as a lot of them contained the HTML for special characters such as &, ‘, <, >. We converted the HTML code for these characters back to text so our natural language processing of the reviews would be more effective and more accurately reflect the true sentiment of the review.

We also found a few columns that were missing the condition feature in the row. We decided to get rid of any rows missing the condition since there were only around 1,000 - 2,000 rows that were missing the condition feature, which was fairly negligible within the size of our dataset.

After looking at the data, we found that the great variation of both drugs and conditions was leading to the data being clouded and might affect the ability of our models to successfully classify (i.e. there might be drugs and/or conditions that only appear once). To account for this, we filtered our data to include only rows with the top 10 most prevalent conditions, which still accounted for almost half of the dataset– with over 98,000 rows. In reviewing the drugs associated with the top 10 conditions, there were still 121 single occurrence drugs; we decided to remove these drugs from the dataset to narrow the data before splitting into training and testing sets.

Exploratory Data Analysis:

We inspected the data and found that there were 837 unique conditions and 3658 unique drugs in the initial dataset.

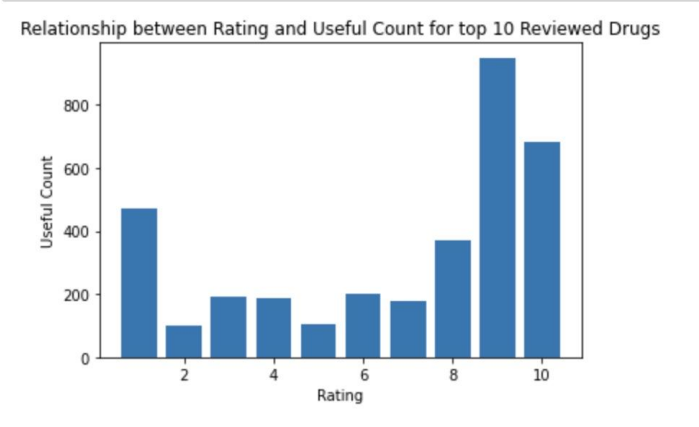
```
In [3]: # find the number of unique conditions
        unique_con = len(pd.unique(drugs['condition']))
        print("Number of unique conditions: " + str(unique_con))

        Number of unique conditions: 837

In [4]: # find the number of unique drugs
        unique_drug = len(pd.unique(drugs['drugName']))
        print("Number of unique drugs: " + str(unique_drug))

        Number of unique drugs: 3658
```

We also did a preliminary inspection of the relationship between a user rating and the useful count recorded of that rating, as seen in the bar graph below. The graph indicates that the more polarizing a rating is (i.e. on the lowest or highest end), the more useful that review was found– which could play a role in the effectiveness of a drug.



Approach and Methodology:

Our approach to solving this problem was to perform sentiment analysis on the reviews in order to define more features representing the sentiment of the review. After taking a look at the most common medical conditions and the most commonly prescribed drugs, we decided to take the 10 most common medical conditions and perform sentiment analysis on their reviews to try to accurately predict their rating from their review.

To perform this sentiment analysis, we used the python NLTK package and filtered out the most common English stop words from the drug reviews. From these filtered reviews, we used the sentiment analyzer from NLTK, which outputs a compound polarity score normalized between -1 and 1. The compound polarity score is interpreted as a score between -1 and -0.05 having an overall negative sentiment, a score between -0.05 and 0.05 being overall neutral, and a score between 0.05 and 1 being overall positive. Using the polarity scores we obtained, we assigned negative scores with a value of 0, neutral scores with a value of 0.5, and positive scores with a value of 1.

We performed further feature engineering by taking the word count and character count of each review and creating new columns for those features. Afterwards, we performed data normalization with the numerical features: rating, usefulCount, word_count, and character_count.

In order to compare a binary classification and multi-class classification model, we needed to create separate datasets for evaluation. For the binary dataset, we converted the 'rating' column from the original 1-10 rating scale to 1 indicating a positive rating if the rating was between 6-10, or 0 indicating a negative rating if the rating was between 1-5. The multi-class classification dataset retained the original data in the 'rating' column. The continuous features such as usefulCount, word_count, and char_count were standardized to have a mean of 0 and a standard deviation of 1.

Since we were using Naive Bayes (NB) for one of our models– and our data contained categorical and continuous data features– we created a new subset of training and testing data for the NB model specifically. Further, we found a mixed NB model online, in order to use the full dataset with both types of features. This required some additional feature engineering but was effective in creating a meaningful NB model.

Machine Learning Models:

We decided on using the following 6 Machine Learning Classifiers:

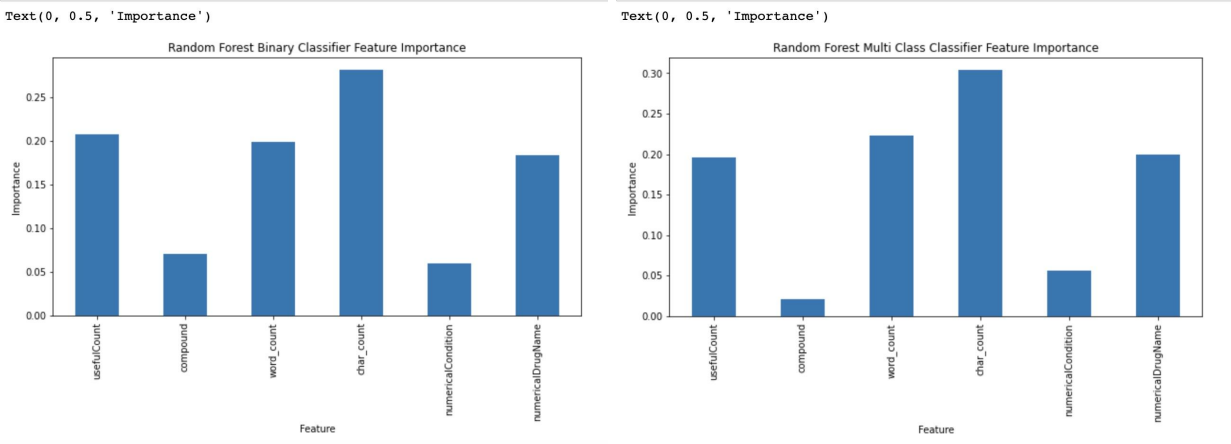
- Logistic Regression
    - Binary Hyperparameters: max_iter=10,000; solver='sag'
    - Multiclass Hyperparameters: max_iter=10,000; solver='sag'; multi_class='multinomial'
- Naive Bayes
    - Mixed Naive Bayes Model
- Decision Tree
    - For both models, tried using gini and entropy split criterion
- Random Forest
    - For both models, tested on 50, 100, 250, and 500 trees
- AdaBoost
    - For both models, tested on 10, 50, 100, and 500 classifiers
- Multi-Layer Perceptron (MLP)
    - For both models, random_state=1 and max_iter=300

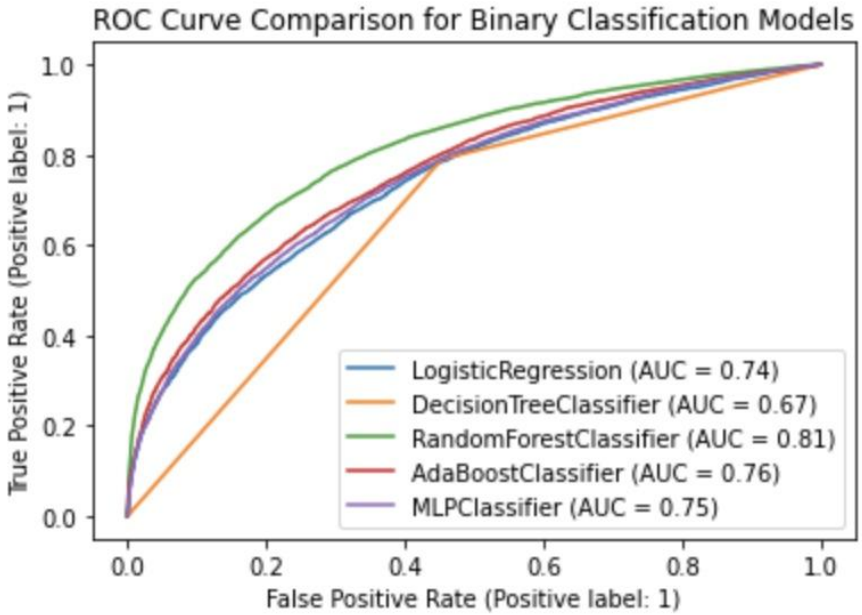
Discussion and Result Interpretation:

Our 6 models generated a lot of different results. The best performing model was the Random Forest Ensemble used on the Binary Classification Data with an accuracy of about 76% - with the number of trees being at least 100. In the feature importance analysis of the Random Forest models, we found that the most important feature was character count, with word count and useful count coming in close behind in importance. From the feature importance graphs, we can also see that the compound score from the sentiment analysis using the NLTK package was not helpful in classifying the models and was consistently a less important feature.

Our multiclass models did not perform well at all. The 6 multi-class models had an overall accuracy of around 30%, although the decision tree and random forest models saw this bump up closer to 45%. We think the multiclass models performed much more poorly than the binary classification models because of a lack of feature richness in the data (only having 6 features throughout) and the high number of possible classes for the multi-class classification problem (10 classes). The number of features in our data was not enough for accurate multi-class classification with so many possible bins. Our binary classification model was able to achieve results from between 70%-76% with some hyperparameter tuning; given the nature of the raw data and the lack of features, this felt successful in the scope of the classification problem at hand. The binary classification task is much easier as there are only 2 classes to differentiate from, and the features in our data were helpful in accomplishing this task.

Overall, this presented itself as quite a challenging task for a number of reasons, including but not limited to: a lack of features and difficulty in engineering new meaningful features, the size of the dataset (even when it was filtered to only 10 conditions, the run-times on some cleaning tasks were completely outsized), and the sparsity of variance within the drug names (there were over 100 drugs that only appeared once in the filtered dataset). Given these limitations of our data, the resulting metrics from our 6 models make sense; working with real world data is often messy and even with hours spent cleaning the dataset and performing various feature extraction, it was difficult to come up with high accuracy across our models. However, we believe that our models show promise in this classification problem– with a more robust dataset and a more powerful computing backdrop, many improvements could be made.





Conclusion:

In conclusion, we do not recommend the multi-class classification task for predicting the rating of a drug from extracting sentiment analysis features from a review. However, we think that if more features were extracted from the review and added to the dataset, the binary classification problem of determining if the sentiment towards a drug is positive or negative could reach a higher accuracy. These features could further implement NLP properties, such as TF-IDF score or counting the frequency of significantly weighted words. This project presented a great jumping off point for classifying drug efficacy based on user reviews, and while

improvements could be made, the binary classifiers sufficiently performed throughout the project.

Team Member Contribution:

Mirah

- Exploratory data analysis
- Wrangling with data and NLTK - emailed TA to find solutions to speed up run time
- Trained the Naive Bayes and Random Forest Classifiers
- Created presentation

Genny

- Data exploration
- Compound polarity score interpretation
- Trained the Decision Tree and AdaBoost Classifiers

Caterina

- Problem Description Background Research & Problem Description Paragraph
- Normalized numerical features in the dataset
- Trained the Logistic Regression and MLP Classifiers
- Wrote report