Clinical Trial Prediction via Natural Language Processing and Graph Mining

A Major Qualifying Project Report: Submitted to the Faculty of

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

By

Calvin Rambacher

Submitted:

March 24, 2023

Approved by Professor Xiaozhong Liu, Project Advisor and Fabricio Murai, Project Co-Advisor

Table Of Contents

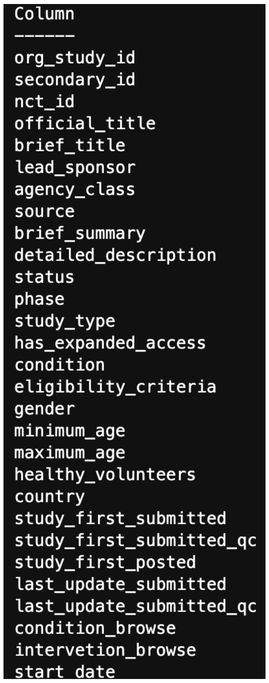
Table Of Figures

Background Information

The Clinical Trial Prediction dataset that is used in this MQP can be found on Kaggle. The dataset used consists of 103K XML files, with data on 338K+ clinical trials. This dataset was extracted from ClinicalTrials.gov.

Extracting The Files

Each of the files in the dataset are XML files of varying size. To effectively extract the necessary data from the files, the Python package ElementTree was used. Once the file system was extracted from Kaggle, the file structure was such that each XML file was enclosed in a folder with up to 1000 other XML files. Additionally, there were 133 of those folders holding XML files. In order to extract the data from each of those files, for loops were set up to go through each folder in the dataset and each file in that folder. Within each XML file, there were tags for



```
Column
------
org_study_id
secondary_id
nct_id
official_title
brief_title
lead_sponsor
agency_class
source
brief_summary
detailed_description
status
phase
study_type
has_expanded_access
condition
eligibility_criteria
gender
minimum_age
maximum_age
healthy_volunteers
country
study_first_submitted
study_first_submitted_qc
study_first_posted
last_update_submitted
last_update_submitted_qc
condition_browse
intervetion_browse
start_date
```

Figure 1: XML Columns

different parts of the clinical trial, as seen in Figure 1. A complication from this is that not all XML files have all columns, which throws errors when the Python program is trying to reach that column. To solve this, all of the column requests used try-except statements to ensure that all of the data was being added with minimal losses. This also results in many rows having null values, though those will be dealt with when formatting the database. In order to store the data received from the XML files, each file is formatted into a one-dimensional array, which is then added to a Pandas DataFrame. That DataFrame is also sent to a CSV file for easier editing.

Formatting The Database

Once all of the data was extracted from the XML files into a Pandas DataFrame, the data needed to be cleaned. Firstly, as the “phase” column is the column we want to predict, all rows that do not have any data in the “phase” column are dropped. Additionally, the “phase” column is made up of strings, such as “Phase 1” and “Phase 3,” so all values in that column were changed to be integers for easier analysis. This change decreased the number of rows down to ~65K from over 100K, which made computation time significantly shorter when analyzing the data. The “nct_id” column was also modified in a similar way to get rid of “NCT” at the beginning of each value to change each value to integers. The “minimum_age” and “maximum_age” columns were converted to integer values of years, as values were formatted as strings, such as “6 Months” to “85 Years.” All columns with integer values filled null values with zeros and columns with string values filled null values with “X.” This change was needed to be able to run the specific model, as only integer and string values were allowed (not even null values). The “has_extended_access” column was also changed from “No” and “Yes” values to 0 and 1, respectively, to make more integer values for easier analysis. The “gender” column was changed into "has_male" and "has_female" binomial integer (0 or 1) columns, as the "gender" column had "Male," "Female," and "All" options.

Analyzing The Data

Because of the nature of the dataset consisting mostly of textual data, CatBoost was chosen as the model framework, as opposed to XGBoost or other computational frameworks. Firstly, the data was split into training and testing data using the train_test_split() method from the sci-kit-learn package. Then, the "phase" column was moved to a separate DataFrame from each of the subsequent training and testing DataFrames. It was also then dropped from the training and testing DataFrames, as it cannot be a predictor for itself.

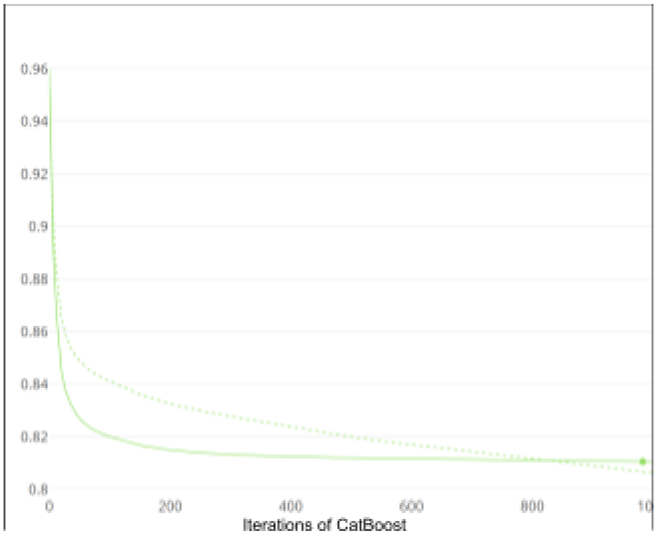


Figure 2: CatBoost Model Accuracy

The model had a final accuracy of 81.06% for predicting "phase" from all other columns. This accuracy is due to the various tuning and maximization algorithms that CatBoost offers. I did try to change values to see if I could get a better fit than the program, but any value I changed in any way made the algorithm's accuracy worse. Additionally, I ran the CatBoost Feature Importance tool, which outputted how much each feature contributed to the final prediction of the "phase"

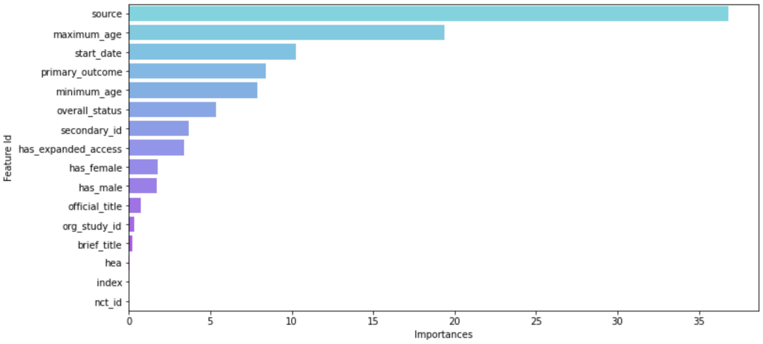


Figure 3: CatBoost Column Weight

column. According to CatBoost, “source,” the column with where the clinical trial originated from, is by far the most important factor in determining how far a drug will reach in clinical trials.