Predicting Clinical Trial Phase Through Gene-Disease Relationships

Andrew Wang, Bhavani Shekhawat, Charlie Flanagan, Derek Kinzo, Gerald Ding, Ramandeep Harjai

Abstract

Disease outbreak detection systems have existed for decades. However, many of them are underutilized or are rarely modernized and thus are incapable of handling massive amounts required for robust classification, which is necessary for detecting outbreaks. Abundantly available information through sources such as NCBI's PubMed gives us an excellent opportunity to leverage Big Data analytics and techniques to glean insights and guide drug research. The vast majority of knowledge is in the form of raw unstructured natural language text, such as medical literature, journals, clinical trials, which dwarfs the structured and classified data. The challenge for most of the BioPharma companies is striking a balance between research domains and available funds. Although there isn't a deficiency of drug research engines in the market, most of them are incapable of keeping up with gene evolution and lack market sense. Our research effort aims to employ indicators such as drug trials, citations, and biological research papers to determine trending gene-disease relationships to guide the industry leaders in the most impactful research direction. Although the core focus of this paper is on detecting disease trends, however, the extensibility of the proposed system allows developers to add other sources and classifiers for deriving trends.

Keywords: trends, data, detection, drug, trials, gene, diseases, research, classification, Pharmaceutical

1. Introduction

Biopharma companies are constantly trying to bring effective treatments to patients whilst being limited by research funds. Research and development is a considerable investment made by these companies. Therefore, leveraging data to gain insights into treatment options could result in a significant competitive advantage (Copping 2016). From 1980 to 2006, the overall R&D investment in the industry went from $2B to $43B; however, the average yearly number of drugs approved by the FDA remained the same. Today, discovering a new medicine takes 12 years on average, and the average cost is more than $1B, which is more than what it takes NASA to send a rocket to the moon (Garnier 2008). As a result, increasing the probability of research dollars spent going towards a successfully deployable therapy has become increasingly important.

Not only is it important to maximize the research success rate, but the first-to-market advantage is especially noticeable in the biopharmaceutical industry. First-to-market companies have a 6 percent market-share advantage over others (Cha 2014); in addition to enjoying exclusivity for the subsequent 20 years after patent filing (Gupta, Himanshu et al. “Patent protection strategies), allowing the company to establish eminence in the realm (FDA 2020). Therefore, starting research earlier than competitors has significant financial and competitive advantages in this industry.

Although there are several challenges in R&D, the rule of success in business is elementary, and that is to make better decisions than bad ones over time. Despite that there is tons of data out there, many leaders are not taking full advantage of it and rely heavily on their gut-feelings. Historically, the pharmaceutical industry has recruited statistical analysts who have performed well-defined analyses of clinical trials in an organized, effective manner. However, real-world data frequently appears in a highly unstructured format and is widespread with missing values. It is untidy data, congested with discrepancies, possible biases, and noise. The need for creativity and innovation becomes even bigger to answer critical research questions to support drug research and development and, ultimately, to provide patients with access to the right therapies. Genentech has been developing such a capability for two years. In addition to investing in data organizations and analytics tools, it has built a big-data infrastructure — a platform that can examine billions of patient records in seconds. A recent example of the kind of work it carries is the creation of a database on a historical group of real-world patients previously diagnosed with cancer. The team examined its data to analyze different patient subtypes and therapy regimens to help Genentech understand the association between biomarker alterations, treatment models, and clinical results. This knowledge would ultimately promote critical drug development choices.

Genie is composed of a Clinical Trial Phase classifier that aims to provide predictions for promising gene-disease relationships. To do this, the framework uses PubMed publication metadata related to gene-disease pairs in order to build features for training the classifier. Features include citation count, author affiliations, chemicals used, and journals themselves. The labels for the classifier are booleans which represent if a gene-disease relationship is involved in a clinical trial which reaches at least Phase II - negative labels are applied to all other gene-disease relationships. The data used to connect the clinical trials to relevant PubMed articles are the genes and diseases that are parsed from the abstracts and detailed descriptions in the PubMed and clinicaltrials.gov records.

2. Prior Work

A number of tools already exist with the purpose of mining data from major journals in order to parse out gene-disease relationships. However, the focus of these tools is to collect data, and the onus of interpretation is left to the user. One such system is *Digsee* that allows the user to search MEDLINE abstracts for evidence sentences describing that 'genes' are involved in the development of 'cancer' through 'biological events'. From a functional perspective, the limitation of this system is that the data is often not current. Moreover, it targets only certain diseases whereas our system is disease agnostic and relies on the data obtained publicly.

Similarly, *PolySearch2* is an online text-mining system that searches for associations against comprehensive collections of free-text collections, including local versions of MEDLINE abstracts, PubMed Central full-text articles, Wikipedia full-text articles, and US Patent application abstracts. PolySearch2 also searches 14 widely used, text-rich biological databases such as UniProt, DrugBank, and Human Metabolome Database to improve its accuracy and coverage. PolySearch2 maintains an extensive thesaurus of biological terms and exploits the latest search engine technology to rapidly retrieve relevant articles and database records. PolySearch2 also generates, ranks, and annotates associative candidates and presents results with relevancy statistics and highlighted key sentences to facilitate user interpretation. However, at the core, it is yet another search engine that remains at the mercy of the user when it comes to usage. As far as the data quality goes, it updates its data every six months which makes any statistical predictions obsolete. On the contrary, our system can be configured to crawl data at provided intervals. Moreover, we utilize views to maintain our data which allows us to deliver the results faster to the customer front-end. By utilizing views, we have also opened possibilities to only looking for updated data. This would save us space, time, and processing power.

Genies is yet another natural-language processing system for the extraction of molecular pathways from journal articles. In this project, authors (Friedman, Kra, Yu, Krauthammer, & Rzhetsky 2001) have designed a system called GENIES (GENomics Information Extraction System), using an existing medical natural language processing system – MedLEE. GENIES extracts and structures information about cellular pathways from various biological and medical literature.

The system has three main components:

1. **Term Tagger** identifies and tags genes and proteins in the target literature,

1. **Preprocessor** is a MedLEE component, which determines sentences, words, and phrases, and performs a lexical lookup

3. **Parser** is also a MedLEE component, identifies relationships, and specifies target output forms.

To test the system, authors compared GENIES output for given literature with an analysis done manually by an expert. The test literature contained 7,790 words, which took 1.3 minutes for GENIES to parse and process on a 500 MHZ PC with 128 MB RAM. The expert identified 51 binary relations; GENIES correctly extracted 27 (53%). Later it was found that many of the 51 binary relations were redundant, and only 19 were unique. GENIES precision was thus found to be 100%. The idea conceptually very closely aligns with what we are trying to do with our Trend Detection project, which is to detect and highlight trends in gene-disease relationships by processing publicly available medical literature, however, it doesn't combine any of clinical trials data to provide any sort of biological pathways to drug research companies.

The authors of the paper *Application of an automated natural language processing (NLP) workflow to enable federated search of external biomedical content in drug discovery and development* take an approach of crawling the data available via the internet to create an integrated information workflow. However, the sheer focus is on the extraction of data and providing that data to the end customer. Eventually, it is up to the customer to interpret any information or derive relationships.

3. System Design

The system proposed by this paper is on Trend Detection by leveraging gene-disease relationships. The construction of the gene-disease relationship will require data from PubMed and ClinicalTrials.gov. The core of the system is the Clinical Trial Predictive Classifier which will take into account several variables that are deemed important. The end goal is to provide a measurable index such as a probability indicating which gene-disease relationships are involved in clinical trials that make it to Phase II or later stages five years prior to the start of the clinical trial. ***Figure 3*** shows the system at a very high level.

The system is comprised of 5 main sub-systems:

1. ***Scraping component*** deals with crawling various data sources (like PubMed, ClinicalTrials.gov), scraping the raw data, and parsing the required data attributes.

2. ***Parsing component*** deals with parsing the crawled data into a format(s) that is backed by our domain models and persists the scoring model aspect of the data into the database by invoking BigQuery APIs.

3. ***BigQuery Storage Table*** deals with data persistence, availability, and retrieval. The Crawling & Parsing sub-system exposes functionality to push new data into the underlying tables, and data be pulled out for training the machine learning (classifier) models, and presenting the data and related analysis into views and then pushed to the user interface. At the moment these tables are hosted on GCP however, the code is database agnostic.

4. ***Classification Manager*** deals with building the classifier models for - The Clinical Trial Phase predictor which can help in predicting gene-disease relationships that are most likely to reach a Phase 2 or above the stage in clinical trials.

5. ***User Interface sub-system*** Presents the data and classifier analysis in a dashboard format. It also provides an export feature through which a user can export the data presented on the screen.

The GeniePy package provides the BigQuery database storage on Google Cloud Platform with lightly processed raw data from various sources. The core sources are outlined below while ***figure 1*** depicts a high-level overview of its internal working w.r.t the data. Also, ***figure 2*** shows that BigQuery queries the storage tables and puts the results into the views.

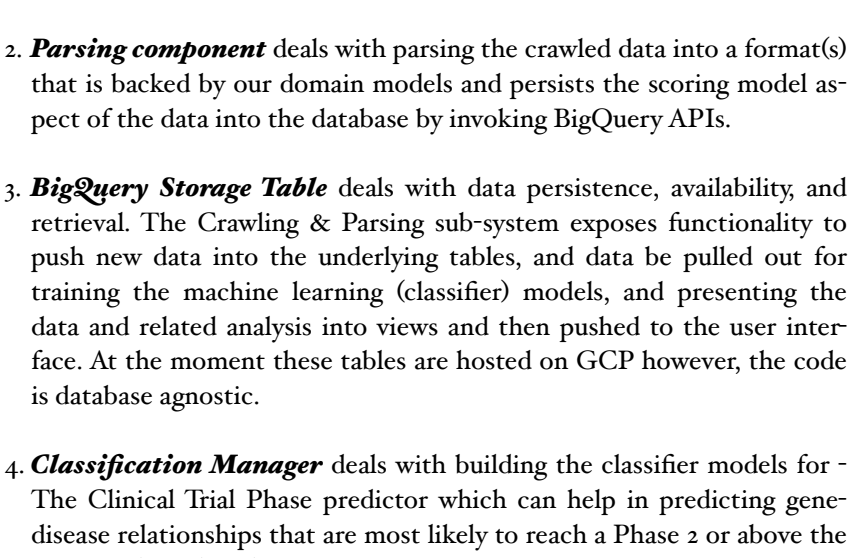

Figure 1: The relation between core tables and the views

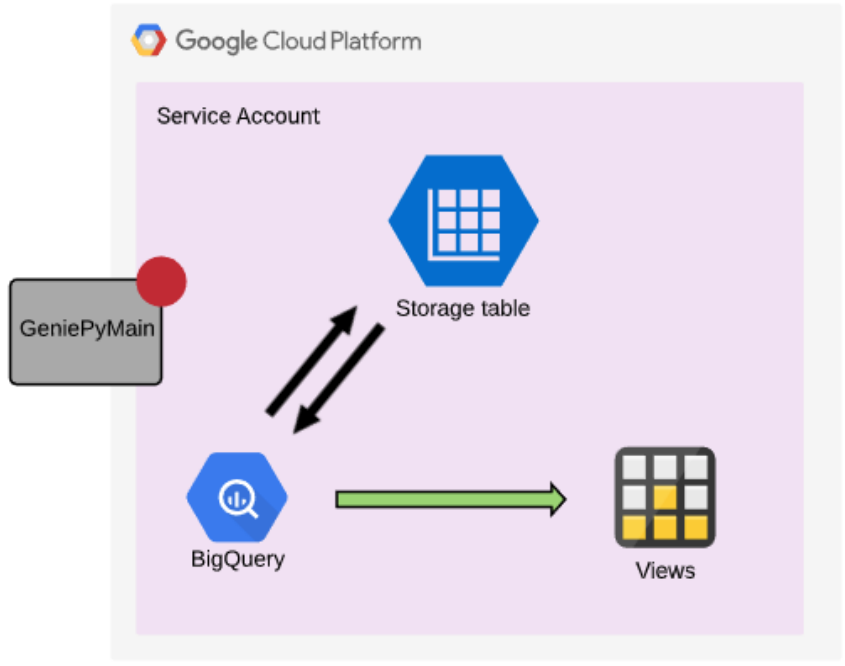

Figure 2: Interaction between storage tables, BigQuery, and the views

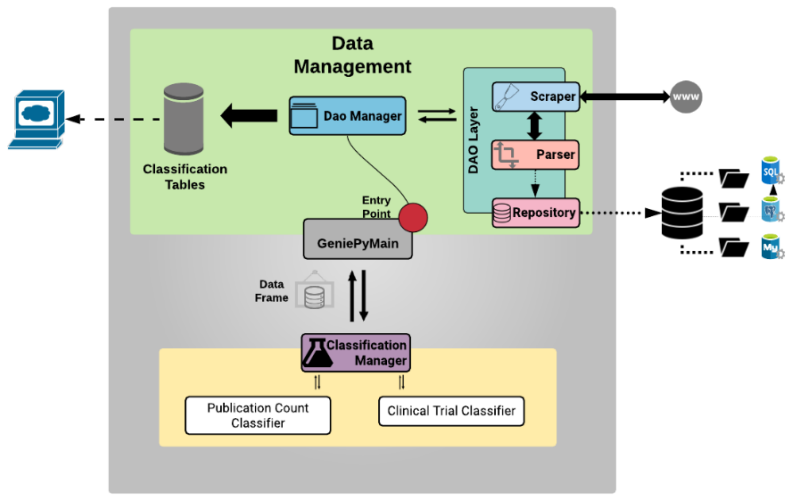

Figure 3: An overview of Components at a high level. The figure shows the placement of components such as a scraper, parser, and the repository.

4. Clinical Trial Classifier

Before any machine learning work can be done, proper data acquisition and data wrangling routines must be in place. The data loader accesses the databases stored in the Google Cloud Platform using BigQuery calls. We have two primary databases: clinical trial data, PubMed article data. In order to produce usable features and labels for machine learning, we must associate the proper clinical trial data with its corresponding PubMed article data if any. The data preprocessor must first identify keywords from the free text of the descriptions of the clinical trial. After processing for equivalent terms, we can pair this data with the appropriate articles. From these associations, features can be constructed and associated with truth labels for training. The classifier algorithms we need are already implemented in the SciPy sci-kit-learn library (sklearn). By using these pre-built algorithms in a fast prototyping development environment like a Python Jupyter notebook, it frees us much more resources to work on classifier tuning and feature engineering. The primary classifier model we are using is Gradient Boosting which performed the best among the techniques we tried as discussed later in this document. This library employs gradient boosting which is a powerful technique for building ensemble classifiers. The Gradient Boosting technique has a good reputation in the online community for general usage and is likely worth it's overhead. With this framework, we can easily add new classifiers into the ensemble to improve our performance. For example, in our initial exploration, we noticed that publication count was an extremely good feature for detecting if a gene-disease relationship is in the Phase II clinical trial. However, our goal is to predict the Phase II clinical trial 8 years in advance, so we built a separate predictor optimized for the prediction of the order of magnitude of publications in the future. Other classification and regression algorithms will be incorporated like random forests, ada boost, and logistic regression.

In order to train a classifier that can perform this predictive identification, a mapping between gene-disease relationships and their success label of achieving pertinence in Phase II clinical trial was assembled. Feature data was then associated with these mappings in order to provide the classifier actual training data. This data was then fed strategically to the classifier to ensure that the model has predictive power.

The truth labels were derived from all clinical trials which were curated from the Aggregate Analysis of ClincalTrials.gov (AACT) database. Through this source, events such as the announcement of a trial beginning along with whether the trial has achieved Phase II or not was determined. The AACT database also contained information to derive any articles associated with each clinical trial. From these articles, it could be determined if the clinical trial has any pertinent gene-disease relationships to the study. This is how we obtain our mapping from gene-disease to relationship to our label of making it to a Phase II clinical trial.

To provide the data for the classifier to train on, features were constructed from PubMed publication metadata and journal prestige metrics. As noted earlier, the goal is to be able to predict the truth label from five years in the past. Therefore it is important that the raw data used to construct the features is only obtained from data sources that existed five years or more before the first announcement of a clinical trial. As such, all of the features are either windowed features or booleans. Below are the features we currently have deployed on the full system below:

The number of publications (num_publications): the feature is the cumulative number of publications associated with a gene-disease relationship up to the data cutoff date. For example, if the clinical trial for a particular gene-disease relationship were announced in 2010, then this feature would be the sum of all publications pertaining to this gene-disease relationship on and before 2005.

The number of citations (citations_cum_sum): the feature is the cumulative number of citations associated with a gene-disease relationship up to the data cutoff date. A citation is counted towards a gene-disease relationship if it cites a publication associated with the gene-disease relationship. For example, if the clinical trial for a particular gene-disease relationship were announced in 2010, then this feature would be the sum of all citations which cite an article pertaining to this gene-disease relationship on and before 2005.

h-index average (h_index): the feature is a metric constructed by the h-index(es) associated with an article's publisher. The h-index is a count metric that describes the impact of an author based on how often their publications are cited. It is computed by finding the maximum integer h for which h publications are cited at least h times. For example, if an author has an h-index of 10, we would know that they have 10 articles that have been cited at least 10 times. This is one of the more popular author impact score metrics because it is easy to compute. However, it disfavors authors who publish fewer but very impactful papers. The metric also has no normalization factor for the credibility of the publication. For our purposes, we have compiled an average h-index score for each publisher. Our feature is computed by finding the average h-index of every publisher which published an article associated with the gene-disease relationship. We then take the weighted average of this average publisher h-index relative to the number of articles contributed by each publisher. In other words, each publisher has an average h-index composed of the h-indexes of its contributing authors. We then associate the articles associated with the gene-disease relationship with the average h-indexes of their publishers. We then take the straight average of these values for this feature. As per all other features, only articles published before the date cutoff are counted into the metric.

Scimago Journal Rank average (sjr): the feature is a metric constructed by the journal ranking associated with an article's publisher. The SJR score is a measure of a publishers' prestige based on what publishers cite it. The formula is involved, but it is considered more robust due to there being both a reputation component and a time component to this score with recent citations being weighted more. Each article is associated with the SJR of its publisher, and this feature is computed by taking the average of these SJR scores.

The number of unique authors who contributed publication to the gene-disease relationship (authors_cum_sum): As noted earlier, only publications before the set cut-off date are counted toward this sum.

USA and UK published boolean (us_uk_published): this feature is a boolean which denotes whether there has been a publication by both a US publisher and a UK publisher on or before the cut-off date.

The USA published boolean (us_published): this feature is a boolean which denotes whether there has been a publication by a US publisher on or before the cut-off date.

The number of published languages (num_languages): this feature is the number of distinct languages that an article has been published in on or before the cut-off date. For example, if there have been articles published in English and in German, this feature would be 2.

Publisher count (cum_sum_journals): this feature is the number of distinct journals that have released publications pertaining to this gene-disease relationship. Only publications before the cut-off date are considered for this feature.

Chemical count (chemicals_cum_sum): This metric is a count of the unique chemicals as defined in the keyword section of an article. Only publications before the cut-off date are considered for this feature.

These features associated with the gene-disease to truth label pairings form the training data that can further be used to train the classifier model. However, only a subset of clinical trials has an associated gene-disease relationship, which greatly reduces the number of positive labels that we have. This is because a clinical trial may not necessarily have an article with a gene-disease relationship mentioned. Furthermore, an even smaller number of gene-disease relationships make it to a clinical trial in the first place. As a result, there is an extreme imbalance where gene-disease relationships that make it to Phase II clinical trials are extremely rare relative to those that do not.

To alleviate this imbalance, the focus is on gene-disease relationships that were already represented in the clinical trial data. However, this could potentially create a training set bias as the training is only performed on the data which has achieved at least a clinical trial announcement. However, with this subset, the rarity of successes to failures ratio is much lower (approximately 1:2). This was balanced enough for the purposes of our training. While all of our features are currently numeric, we found little success using the easily trained linear classifier models. Out of the models we attempted, we had the most success with sklearn's random forest and gradient boosting trees. Neural networks would certainly have been a topic of interest, but we did not have sufficient resources to pursue the significantly greater model tuning requirements of training a neural network. After some testing, we decided to settle on gradient boosting which performed slightly better by our metrics. ***(See Figure 4)***

To reiterate, the classifier ingests the formed feature data and outputs a predicted probability of a gene-disease relationship being a part of a clinical trial that achieves the Phase II study. In order to judge how well our model performed, we decided on using the Area Under the Receiver Operating Characteristic curve (AUC) score as our primary metric. While a straight-forward accuracy metric is more intuitive, the AUC metric is more useful when the outcome distribution is imbalanced. For our purposes, there are fewer gene-disease relationships that make it to Phase II trials than those that do. A straight accuracy score could potentially be maximized by a naive classifier which simply guesses the most likely outcome every time. Since it is our goal to detect the uncommon case (Phase II trials) the AUC metric is a measure of how high we can make the true positive probability while minimizing the false positive rate.

5. Classifier Model Validation

To ensure that our models were not overfitting, every step in the process we performed k-fold cross-validation (usually between 5 and 10). By ensuring that the validation scores were similar across all folds, we can reasonably believe that we were not accidentally overtraining the model.

A very crucial validation step was withholding later time periods from the training data. It may be confusing why this is necessary at first as we have already ensured that feature construction only uses raw data collected from five years before a clinical trial's announcement. However, in order to show that the model has predictive power, we need to validate that it can make predictions on time epochs which it has not had any training data for at all. To this end, we needed to add new validation sets that withhold later time periods in order to ensure that our model performance remained consistent. An example of this is training the model on clinical trials announced on and before 2015 and then validating clinical trials announced during 2017.

Validation sets were also partitioned based on genes and diseases themselves. This way we could ensure that the model had efficacy when encountering unseen genes or diseases. For example, we would withhold a subset of data that pertained to a certain gene from the training set and then validate it on the withheld set. These validation sets were less important to us because the rate at which completely novel genes and diseases are added to the literature is relatively slow. However, we deemed these sets worthy to include in our studies.

By using grid-search among other sklearn tools, we were able to tune the parameters in a way that maximized our AUC score with respect to the aforementioned validation sets. We achieved an AUC value of 0.71 when training on clinical trials from 2015 and earlier and validating on 2017 data and later. While this result was significantly worse than validating on a ran-

domly withheld subset of the total dataset, this was still much better than expected.

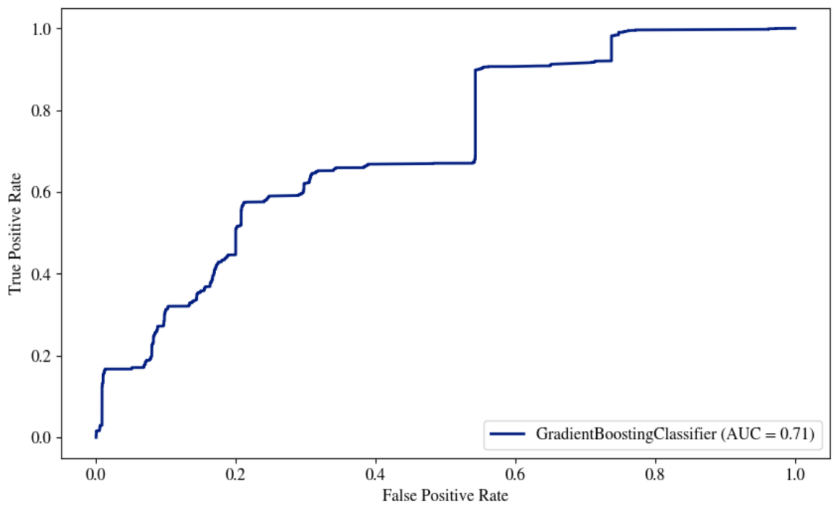

Figure 4: Gradient Boosting Classifier Results

The finalized model relies on citation counts the most. This makes sense as this is a metric of how interested users are in a particular topic. ***(See Figure 5)*** The two publication reliability metrics were the second most important. The third most important feature was the number of unique authors who had contributed a publication to that gene-disease relationship. This makes sense as it is another metric of how much attention the topic is receiving. On the other hand, we were surprised to see the publication count had almost no predictive power.

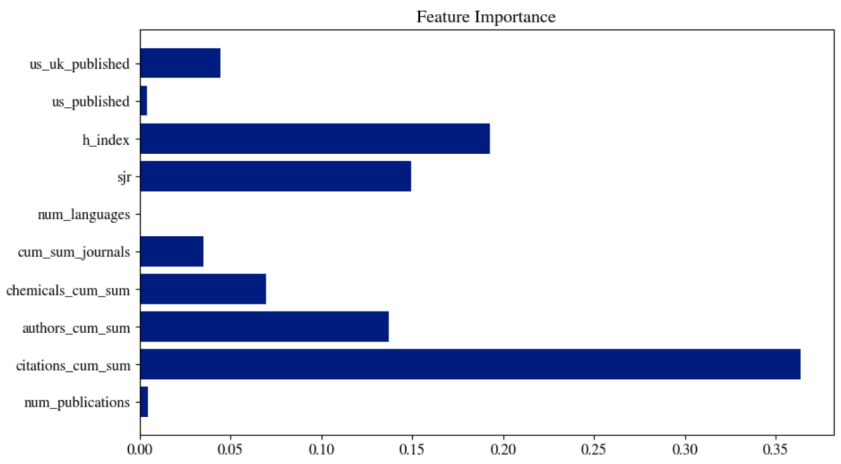

Figure 5: Feature Importance

6. Software Design

The ***figure 6*** provides a high-level description of how the application is architected. The application is subdivided into two main sub-packages: Data and Classification Management.

The data management sub-package is responsible for gathering and composing the data available online, creating local databases, and serving this data as a pandas data-frame back to the main function.

The classification sub-package handles the classifiers and interacts with the main application by receiving a data-frame with classifier features and returning a data frame containing the corresponding predictions.

The entry-point of the application starts by generating the necessary tables by scraping online sources. The main application can then request the gathered data from the data management sub-package and feed it into the classification manager to calculate the predictions. Once the predictions have been calculated, the resulting data-frames are handed back to the data management to store the predictions into the classification tables which are then queried by the user interface to display the results.

The DAO Manager and Classification Manager expose the functionality of their layers whereas GeniePyMain controls the entire application.

From an extensibility perspective, it is to be noted that the **BaseRepository**, **BaseParser**, and **BaseScraper** class adhere to the open-closed principle and provide a way to create custom functionality for future development. For example, developers can use the **BaseRepository** class to create implementations for other database dialects. Similarly, **BaseScraper** can be utilized to create more web-crawlers and likewise, **BaseParser** can be used to provide any custom logic for data transformation.

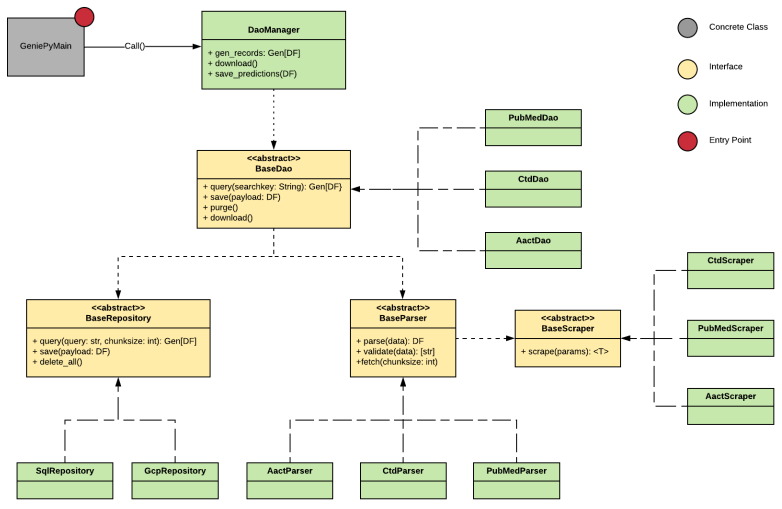

Figure 6: The diagram below depicts the class diagram for the Data Management sub-package

7. Software Architecture

1. GeniePy application gets invoked from the command line as seen in ***figure 8***
2. The application facilitates the base table creation in Google Cloud.
3. The scrapers get initialized and go out to crawl the data from specified resources.
4. The crawled data is then parsed by the parsers so that the resultant structure conforms to our cloud storage tables.
5. Once the data is set into the cloud storage tables, we run BigQuery commands to lookup the populated tables and build the source table views. **NOTE:** As of now the BigQuery commands are scheduled to run weekly due to the nature of the data and to save costs.
6. The classification manager invokes BigQuery commands to create these views and place the results of joining the base tables into those views.
7. At this point, two views are created i.e. ***Gene-Diseases connected*** and **Scoring Features view**
8. A table called ***Scoring features table*** from ***Scoring features view*** gets created.
9. The scoring features table is then fed into a Classifier (Contains SQL code) and two tables are created by the name of ***Classifier Output Scores*** and ***Diffs Table***.
10. The front-end queries Google Cloud using service account credentials and loads a subset of scoring-features view.

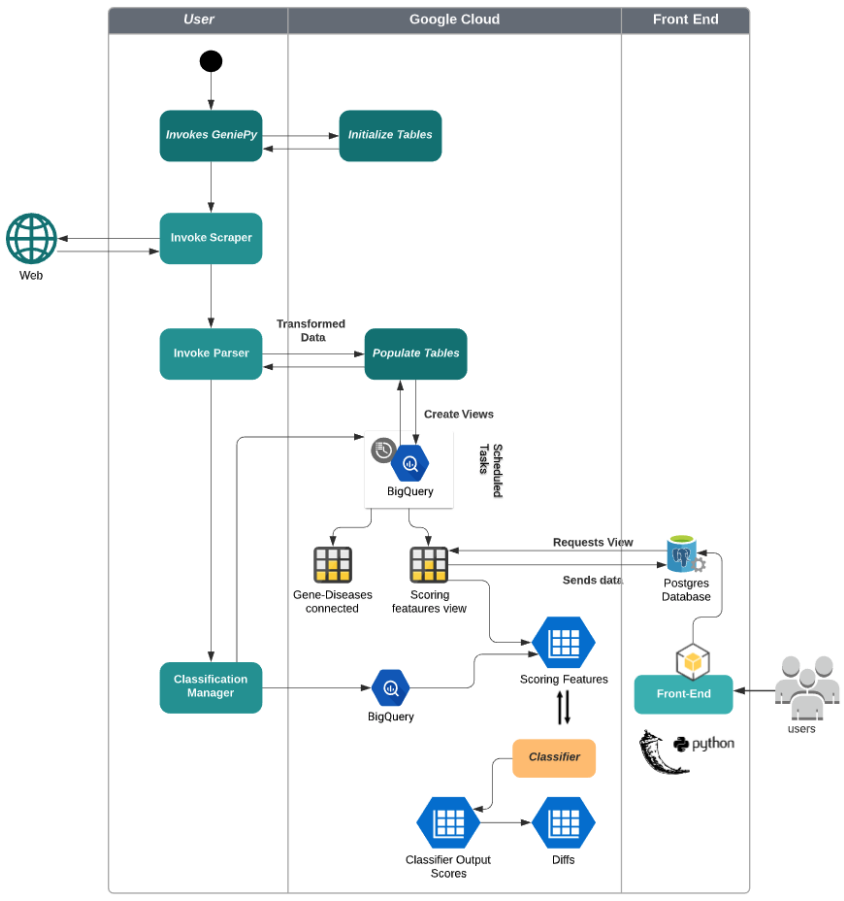

Figure 8: An end-to-end workflow from user to GeniePy

8. Conclusion

Our demonstration has shown the potential and applicability of this framework to collect and wrangle biomedical data to train and validate models that predict trends in the biopharmaceuticals space. In this project, we have assembled an end-to-end pipeline for scraping and parsing

PubMed articles, clinical trial data, and other sources for the purpose of creating feature data to train classifiers. We achieved full implementation of this framework to train and test a classifier model for predicting which gene-disease relationships will be involved in clinical trials that make it to Phase II. Our classifier achieved a validation AUC score of 0.71 when trained on the year 2015 data and earlier

clinical trial data and validated on clinical trial data from 2017 and later. This is a promising show of robustness for the model's predictive power which ideally can be bolstered by adding more features. While we have only provided the implementation of one classifier pipeline, our work can be extended to accommodate any desired classifier variations which can leverage our data processing. Moreover, additional sources can be added for crawling.

9. Future Work

The constructed classifier, while potent, is a relatively simple ensemble model trained on a small dataset. Given more time and resources, a more sophisticated model could have been built. For example, we could have tuned a secondary classifier that predicts future citation counts and used its output as an input to the primary classifier in a larger ensemble. We also could have explored neural networks, which is a huge area of untapped potential for our project.

Our dataset consists only of gene-disease relationships which had been associated with a clinical trial at some point in time. In order to train a more comprehensive predictive model, we would need to input more gene-disease relationships that never achieved clinical trial status as negative labels. This venture was gated by computational costs.

Many other features that likely had predictive value are also omitted due to being unable to be incorporated into our data pipeline in a timely manner. For example, extracting the contact information of authors is throttled. Other interesting time series features like moving averages or goodness of fit tests are not included due to being unable to parallelize their computation or transcribe their calculation into SQL commands.

Citations

1. Friedman, C., Kra, P., Yu, H., Krauthammer, M., & Rzhetsky, A. (2001, June). Genesis: a natural language processing system for the extraction of molecular pathways from journal articles. Bioinformatics, 17, S74-S78

2. McEntire, R., Szalkowski, D., Butler, J., Kuo, M. S., Chang, M., Chang, M., Xu, J. J. (2016, May). Application of an automated natural language processing (NLP) workflow to enable federated search of external biomedical content in drug discovery and development. Drug Discovery Today, 21, 826-835.

3. Garnier JP. Rebuilding the R&D engine in big pharma. Harv Bus Rev. 2008;86(5):68–128.

4. Cha, M. and Yu, F. "Pharma's first-to-market advantage," McKinsey & Company: Pharmaceuticals & Medical Products, 2014. Online. Available: https://www.mckinsey.com/industries/pharmaceuticals-and-medical-products/our-insights/pharmas-first-to-market-advantage.

5. FDA, "Frequently Asked Questions on Patents and Exclusivity," U.S. Food & Drug Administration, 2020. Online. Available:

6. Ryan Copping-Michael Li, The Promise and Challenge Of Big Data For Pharma, - https://hbr.org/2016/11/the-promise-and-challenge-of-big-data-for-pharma

7. Global protein therapeutics market 2016-2020 - key vendors are AbbVie, Amgen, F.hoffmann-la Roche, johnson & johnson, Merck & Novo Nordisk. (2015, Dec 11). PR Newswire Europe Including UK Disclose Retrieved from http:/search.proquest.com.ezpprod1.hul.harvard.edu/docview/1747898023?accountid=11311

8. Nadkarni, P. M., Chapman, W. W., & Ohno-Machado, L. (2011, September 01). Natural language processing: an introduction. Journal of the American Medical Informatics Association, 18(5), 544–551

9. Yandell, M. D., & Majoros, W. H. (2002, August 01). Genomics and natural language processing. Nature Reviews Genetics, 3, 601–610.

10. Kim J, So S, Lee HJ, Park JC, Kim JJ, Lee H. DigSee: Disease gene search engine with evidence sentences (version cancer). Nucleic Acids Res. 2013;41(Web Server issue):W510–W517. doi:10.1093/nar/gkt531

11. Liu Y, Liang Y, Wishart D. PolySearch2: a significantly improved text-mining system for discovering associations between human diseases, genes, drugs, metabolites, toxins and more. Nucleic Acids Res. 2015;43(W1):W535–W542. doi:10.1093/nar/gkv383

12. Gupta, Himanshu et al. "Patent protection strategies." Journal of pharmacy & bioallied sciences vol. 2,1 (2010): 2-7. doi:10.4103/0975-7406.62694