Persona-Based Drug Recommender System using Online Reviews

Experiment Findings · August 2021

DOI: 10.13140/RG.2.2.29049.19048

2 authors, including:

Pranshu Tiwari
IBM
5 PUBLICATIONS **0** CITATIONS

SEE PROFILE

Persona-Based Drug Recommender System using Online Reviews

August Posch[1] and Pranshu Tiwari[2]
Northeastern University[1,2]
posch.au@northeastern.edu[1], tiwari.pra@northeastern.edu[2]

Abstract

This project explored an online drug reviews dataset, with the goals of understanding the dataset and creating a novel drug recommender system. To understand the dataset, we used text processing, dimensionality reduction, and clustering techniques. We found insights on the best ways to process this dataset to find helpful clusters . We created a drug recommendation system where we used personas as features to recommend the best drug based on predicted rating.We found that rating of medical post based on neural network feed forward architecture worked better for 600-dimension TFIDF data set than a 16-dimension Doc2Vec data model . We also found that t-SNE on TFIDF revealed separation of a "taken drug more than once" persona . Although we have a functioning recommender system for a subset of our data , there is a future opportunity to work with wider variety of diseases and include domain expertise on drugs . The study gives an opportunity for health care settings to develop a deeper data set which tracks previous drugs tried by multiple patients that would pave way for a more robust research on drug recommendation system.

Introduction

People suffer from health many health problems. Often there is a lack of access to doctors, and even doctors may have inexperience with complex situations involving multiple diseases and multiple drugs already tried. We hope to provide a way so that patients and doctors can make more informed decisions about which drugs to try next.

We are creating a first-of-a-kind persona-based drug recommendation system. There is no prior work done in Kaggle on a drug recommendation system of this type. Our methodology creates a unique model for each disease but can be used across all different kinds of diseases. It will improve quality of life for patients and give patients and doctors additional information to navigate various comorbidities and side effect profiles.

Our comprehensive data set allows us to implement many kinds of machine learning, both unsupervised and supervised.

Dataset

Our raw dataset consists of online drug reviews, obtained by scraping pharmaceutical review sites (Gräßer et al., 2018). There are 161297 observations with 7 features. The features are review ID, drug name, condition (the condition or disease experienced by the patient), review (the text of the review), rating (from 1 to 10 stars), date (when the review was submitted), and usefulCount (number of site users who found the review useful). For clarity, in the rest of this paper, we will refer to all conditions as "diseases" throughout this paper. There are 884 unique diseases and 3436 unique drugs in the dataset. The dataset contains no information about repeat patients – we do not know whether any two of these reviews were given by the same person. The dates range from February 24, 2008 to December 12, 2017.

This is a training dataset from a Kaggle competition (Li, 2018). In this paper, we apply unsupervised machine learning techniques to this training dataset. There is also a companion test dataset with the same features and 53766 observations, which we use later on for testing our recommender system.

Objectives

We pursued two tracks of inquiry in parallel. One one track, we looked at the complete dataset from a high

level, in order to understand the structure of the dataset better and extract features for our recommender system. This track consisted firstly of exploratory data analysis and data cleaning. Then we performed clustering, in order compare preprocessing techniques, compare evaluation methods, and find clusters that reveal information about how the reviews relate to drugs and diseases.

On the other track we started with a reduced dataset corresponeding to just one disease, ADHD. We looked at the reviews manually and generated personas reflecting different types of people that review ADHD drugs. Then we predicted ratings and evaluated the effectiveness of preprocessing architecture for predicting ratings. Finally, we created a recommender system for ADHD drugs, leveraging our personas as users.

Approach (Entire Dataset Track)

Step 1: Exploratory Data Analysis

We created histograms and boxplots exploring various features. To explore the text data, we read through samples of the diseases, drugs, and reviews. See the Results section for findings.

Step 2: Data Cleaning

Looking at the text data revealed that we had data quality issues in reviews and diseases. For reviews, we had HTML issues, e.g. `"I&#039;ve tried this drug"`, so we corrected these into, e.g., `"I've tried this drug"`.

For diseases, we had poorly scraped entries. Some (900) refer to the wrong part of the page, e.g. `'61</span> users found this comment helpful.'`. Some (216) were other scraping errors that couldn't be identified as a disease, e.g. `'mist ('`, or `'min / sitagliptin)'`. Some (899) were nulls . We opted to remove all 2015 of these observations from the dataset

We left in various other disease spelling errors, such as `'Cance'`, `'Bipolar Disorde'`, and `'emale Infertility'`. These clearly refer to cancer, bipolar disorder, and female infertility. Furthermore, these errors were consistent across the dataset – there were no other entries referring to female infertility except those that were spelled as 'emale Infertility', and this was true for all the spelling errors. Thus, we opted to leave these as-is, because we know which diseases they refer to.

Step 3: Text Processing

We removed stop words, performed stemming, and created both a Word Frequency dataset and This was achieved using the NLTK's SnowballStemmer in conjunction with Scikit-learn's CountVectorizer and TfidfVectorizer, both of which have built-in ability to remove stop words.

```
import nltk.stem

english_stemmer = nltk.stem.SnowballStemmer('english')

class StemmedCountVectorizer(CountVectorizer):
    def build_analyzer(self):
        analyzer = super(StemmedCountVectorizer, self).build_analyzer()
        return lambda doc: ([english_stemmer.stem(w) for w in analyzer(doc)])

class StemmedTfidfVectorizer(TfidfVectorizer):
    def build_analyzer(self):
        analyzer = super(StemmedTfidfVectorizer, self).build_analyzer()
        return lambda doc: ([english_stemmer.stem(w) for w in analyzer(doc)])
```

Above is the definition of the vectorizers used.

To get a word-frequency dataset, we divided the word counts by the total number of words in each document.

Step 4: Dimensionality Reduction

Dimensionality reduction was performed on both datasets. We used TruncatedSVD, which similar to PCA, except that the data are not centered before finding the components. The mathematical effect is that the datapoints are projected to an affine space, rather than a subspace, of the original data space. TruncatedSVD has memory advantages over PCA, making it more appropriate for sparse datasets like ours, and it is common in text analysis. (Scikit-learn developers, 2020)

We extracted 200 LSA components from each dataset. See Results section for findings.

Step 5: K-means Clustering and Evaluation

K-means clustering with plusplus initialization was performed on various versions of the data using various parameters, and for each run we recorded evaluation criteria. This can be thought of as a gridsearch over: which dataset (word-frequency LSA or TFIDF LSA), number of clusters K (integers from 2

through 10), number of LSA components d (d = 1, 2, 4, 7, 10, 20, 40, 70, 100, or 200), how to evauate internal criteria (evaluate on the d-component embedding or evaluate on the 200-component embedding). There were 180 runs of K-means in total. For each run, we recorded sum of squared error (SSE), Calinski-Harabasz scores (CH), Silhouette coefficients (SC), Normalized Mutual Information (NMI) with drugs, and NMI with diseases.

Due to encountering runtime issues early in the project, mini-batch K-means (also known as web-scale K-means) was attempted. We used the Scikit-learn implementation and compared results using Normalized Mutual Information (NMI) with a traditional K-means clustering. We got the best results with batch_size=5000, max_no_improvement=50, and n_init=100, which seemed to be the most generous we could be to allow best results. These are also considered reasonable parameters in the literature (Sculley, 2010). However, we still had poor evaluation results and not enough improvement in runtime performance, so we switched back to using the Elkan K-means algorithm from Scikit-learn.

The biggest runtime performance boost was found by using random samples of 1000 to calculate Silhouette coefficient, rather than the entire dataset. This dramatically cuts down on the number of pairwise distance computations, which was the biggest cause of runtime slowness in our initial implementation.

In the end we decided on K-means algorithm parameters of max_iter=100 and n_init=10 based on trial and error and seeing where the NMI between clusterings stopped changing. These parameters lean on the side of longer runtime but higher probability of achieving best results.

We used the elbow method to find the best K by sum-of-squared-error (SSE), evaluated in the input dimension as SSE is designed to do. To choose K this way, the standard practice is to look for an "elbow" in the plot of SSE vs K.

Part of our approach with the clustering was to find the parameters that give the best evaluation score. Another part of our approach was to see, among the best clusters we found for different evaluation approaches, which evaluation approach gave us the most helpful clusters. Please read more about this iterative process in the Results section.

Approach (ADHD Dataset Track)

The following construct has been defined for a smaller sub-set for setting up of drug recommendation system based on machine learning.

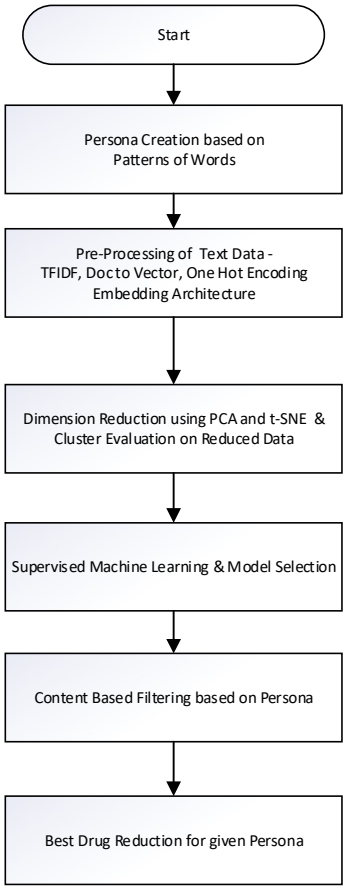


Figure 1: Approach for ADHD subset of data

Step 1: Persona Creation

Since we were working on smaller sub-set of data of size about 3000, we wanted to create 4 real life personas for patients. Each persona is set up by finding presence of certain pattern of words:

Persona	Comorbidity	>1 Trials
1	No	No
2	Yes	No
3	No	Yes
4	Yes	Yes

Table:1 Persona Features

Comorbidity is found by presence of words which are not caused by side effect of medicine. The following patterns of words have been used for persona creation

Feature	Status	Pattern of words
Comorbidity	+	"Cholestrol","High Blood Pressure","Diabetes","Migraine"
>1 trial	+	"Second"," second chance","third","multiple"

Table:2 Feature Pattern Identification

Step 2: Transforming text Reviews to numeric data

We removed the following stop words, converted words to lower case and performed lemmatization. We followed the same process as given in the complete data set.

1. We used TFIDF and word vectorization formulation to convert text to feature vectors as per mechanics below:

$$tf(t,r) = \sum_{i=0}^{V} f(x,r)\ ;\ x\ is\ index; r\ is\ review; t - word$$

$$tf(t,r) = \{1\ if\ x = t\ else\ 0\}$$

$$TF\_IDF = tf(t,r) * idf(t);\ where\ idf(t)\ is\ as\ below$$

$$idf(t_i|D) = log\frac{(D)}{(tf(t_i,|D)+1)}; ..eq(1)$$

$$D = Total\ Review\ Posts; t_i - term\ or\ word$$

2. Doc2Vec model extends the Word2vec model, by concatenating multiple word vectors into a sentence vector
to encapsulate the context (Lau & Baldwin, 2016). In the
project, each drug review is a document, and each document is represented by a n-dimension vector; n $\epsilon[16,32,64]$ with a word that have minimum count of 40 in the corpus. Since we are training set is about 3000 documents, we initially postulated that 16 dimensions would be sufficient to represent each document. A window size of 2 was used to control the distance between the word vectors in the concatenation and the word to be predicted. This way using Neural Networks, we identify the vector representation for a document by predicting the next 2 words.

3. One Hot Encoding: One hot encoding is a representation of categorical variables as binary vectors. This first requires that the categorical values be mapped to integer values. Then, each integer value is represented as a binary vector that is all zero values except the index of the integer, which is marked with a 1. However, since each document will have many words the position of 1 is written in the data frame representing the hot encoding. In this project, although we have converted the text to one hot coding as well, the prediction on this architecture is not scoped in the project.

Step 3: Dimension Reduction: PCA & T-SNE

Since TFIDF architecture had a Feature Vector size of 1000, we reduced the features so as to ensure that 80% of variance is captured. We have used Principal Component Analysis. [Jolliffe, 2002]. Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. Principal Component analysis can be arrived at through Singular Value Decomposition if the initial Matrix is normalized. Since Doc 2 Vector is a normalized vector we can use SVD equation as below

$$X = U * \Sigma * V^T \quad eq(2)$$

$$where\ Cov = X^T X$$

$$and\ X\ is\ Normalized\ TFIDF\ Matrix$$

U,V are left and right singular vectors of X. We can tune the number of Principal components to preserve the information as per equation below:

$$\lambda_i = \Sigma_{ii..}^2 \quad eq(3)$$

Reconstructed Matrix can be computed by x*V

Similarly, we also perform t-stochastic neighborhood embedding (T-SNE). This algorithm calculates a

similarity measure between pairs of instances in the high dimensional space and in the low dimensional space by calculating p(j|i) by centering on x_i. This gives us a set of probabilities for all points. Those probabilities are proportional to the similarities. The Gaussian distribution or circle can be manipulated using what is called perplexity, which influences the variance of the distribution (circle size) and essentially the number of nearest neighbors. After this we map the probability to Cauchy distribution and create KL divergence to optimize it.

$$p(j|i) = \frac{exp(-\|x_i - x_{\bar{j}}\|^2 / 2\sigma_i^2}{\sum_{k \neq i} exp(-\|x_i - x_{\bar{j}}\|^2 / 2\sigma_i^2}$$

$$\sigma_i^2 : Variation\ at\ i; j\ is\ a\ point\ other\ than\ i$$

$$q(j|i) = \frac{exp(-\|y_i - y_{\bar{j}}\|^2}{\sum_{k \neq i} exp(-\|y_i - y_{\bar{j}}\|^2}$$

$$Cost\ function = \sum_i \sum_{j \neq i} log\ (\frac{p_{j|i}}{qj|i}) \quad eq(4)$$

Then we arrive at optimum y so as to ensure the distance between two probability spaces is minimal by computing the gradient ∇c w.r.t to y_i

We then use Gaussian Mixture Model and k clustering technique to arrive at Clusters. Here we compare the Euclidean distance or L2 distance between vectors of drug reviews with vectors value of initial clusters and then iterate over until the mean value of clusters do not change over successive iteration. Once this is done, we calculate the performance of clusters with external labels. We iterate the clusters as per cost function

$$L(\mu, z) = \sum_{k=1}^{K} \sum_{n=1}^{n=N} I(z = k) * \|x_i - \mu_k\|^2 \quad eq(5)$$

K represents the clusters of medical reviews and we want to minimize the loss function. To do this we differentiate eq (5) to minimize the SSE loss function to arrive at the centroid value x represents the document vector position in reduced dimension space.

$$\mu_k(l) = \frac{1}{m} * \sum x_k \quad eq(5.1)$$

m refers to number of points in the cluster for that iteration

We also leveraged Gaussian mixture model and DBSCAN algorithm to find the best clustering over linear dimensional reduced space.

We then create a Gradient Vector which computes loss function as a function of different weight vectors and arrive at best weight for which $L(\theta)$ is minimized.

We also evaluate these clusters to rating labels and persona labels using Mutual Information and SC score on the complete training data set.

Step 4: Predict Rating of Drug Reviews

Considering the test dataset does not have ratings, we wanted to leverage the training data over reduced dimension data obtained from TFIDF architecture to obtain the rating. We also modeled the ratings from training data on Doc 2 Vector (16 feature vector) as

For linear regression we just optimize the cost function:

$$\overrightarrow{\theta_i} = \left|\left|f(\vec{x}) - \mathrm{y}\right|\right|^2 \quad eq(6)$$

Since X= [$\overrightarrow{X_1}, \overrightarrow{X_2}, \overrightarrow{X_3}, \overrightarrow{X_n}$], which are vectors for n medical reviews, and y is a scalar which represents the rating provided by system, then we can calculate:

$$\vec{\theta} = (\lambda I + X^T X)^{-1} X^T Y \quad eq(7)$$

We wanted to use training data to aptly run the model. In this case since, rating are numerical numbers between 1 to 10, we considered linear regression and Neural Network with ReLu activation function using stochastic gradient optimizer or coordinate gradient method to optimize the weights. Since we are using a stochastic function, we will be using batch size to train the data and update $\vec{\theta_i}$ for i^{th} iteration.

Similarly, we can use Neural Network algorithm by changing the weights across different layers of Perceptrons to arrive at rating. The X refers to the training set matrix with medical reviews and feature

vectors. However, we add an $\phi(x,\theta)^T * w$. We have a parameter θ that map ϕ to desired output. Rectified linear function are used on top of affine function.Since neural network consist of various dense layers it is concatenation of various functions. $y = f_1(f_2(f_3(X))$

Step 5: Recommendation System

The way the data is set up, we only have one unique drug per user. Although technically the number of unique users is more than number of drugs, we would naturally think item-item collaborative filtering would be better model. However, since there is one unique user per drug, we would not be able to get average rating of drug for each user. Thus we tried a user-user collaborative model. However, since the matrix was sparse we were not able to get the full model up and running. Instead, we leveraged content filtering with each persona acting as a feature. We then ran a prediction model using the closed form of the Linear Regression formulation:

$$\vec{\theta} = (\lambda I + X^T X)^{-1} X^T Y \quad Eq(8)$$

X is the matrix of features which in this case reflects personas, which are explained in Step 1.We then predict the missing ratings for each persona, as we know its features to predict the rating of each drug given the person $\vec{x}$.

$$y\_pred = \vec{\theta}\,\vec{x} \quad Eq\,(9)$$

We now predict the drug score for a given persona for every drug. The drug with the maximum predicted score will be interpreted as the best drug for this persona.

Empirical Results (Entire Dataset Track)

Exploratory Data Analysis

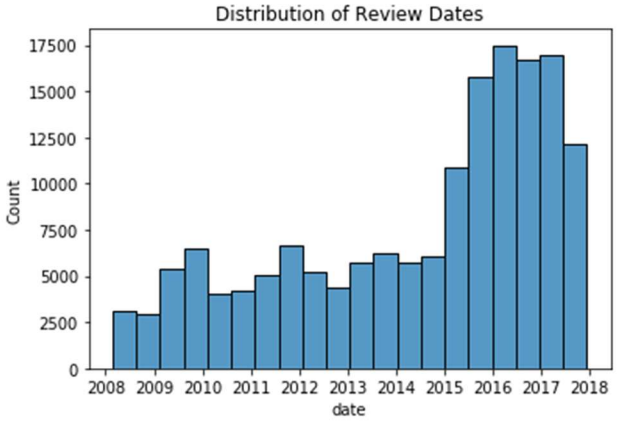


Figure 2: Distribution of Review Dates

Above we see that reviews are 2-3 times more frequent in the time between 2015 and 2018 than in the rest of the timeframe.

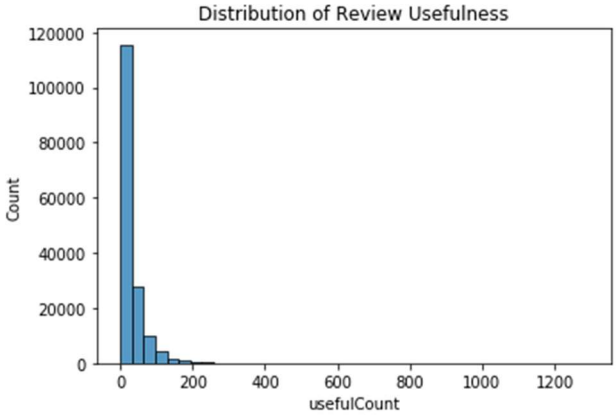


Figure 3: Distribution of Review Usefulness

Above we see that the usefulCount feature has a roughly exponentially decreasing distribution – i.e., as we get into more and more useful reviews, there are fewer and fewer reviews that are so useful.

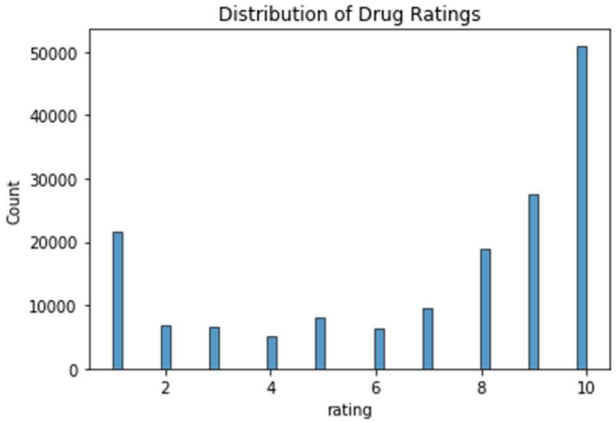


Figure 4: Distribution of Ratings

Above we see that drug ratings have a U-shaped distribution. More people give either a 1 or a 10 than give a 5 or a 6.

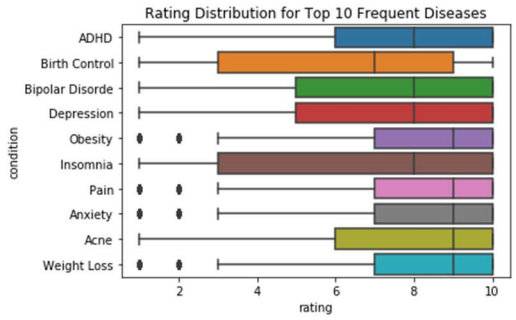


Figure 5: Rating Box Plots

Above, we first grouped the dataset by disease and filtered to the 10 most frequent diseases. Then we plotted a boxplot of rating for these 10 diseases. We see that Birth Control has the lowest Q1, median, and Q3 values – because these ratings are lower than for most other diseases, there is great opportunity for future study to help these people. Likewise, Insomnia has the largest interquartile range, suggesting that a good drug recommender system might make a big difference in the patient's experience. ADHD has an interquartile range that is not the broadest, but also not the narrowest, among these 10 diseases. Its Q1 and median are also not the highest nor the lowest. Due to these characteristics, we could think of ADHD as an "average" disease with respect to ratings, among these top diseases. Therefore ADHD is a good candidate for developing our first drug recommender system.

```
'Sedation',
'Vaginal Yeast Infection',
'Constipation',
'Pain',
'Anxiety',
'Birth Control',
'Ankylosing Spondylitis',
'Chronic Fatigue Syndrome',
'High Blood Pressure',
'Post Traumatic Stress Disorde',
'Multiple Sclerosis',
'Weight Loss',
'Diaper Rash',
```

Above is a selection of some diseases. Note that some have spelling errors – we address this in the next section.

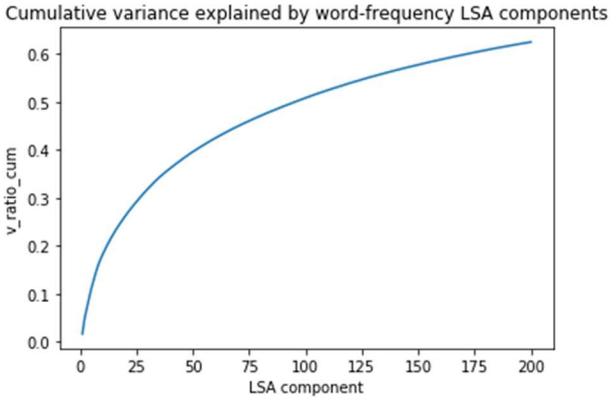
```
'Desvenlafaxine',
'Bisacodyl',
'Ethinyl estradiol / norethindrone',
'Percocet',
'Sertraline',
'Testosterone',
'Desonide',
'Lo Loestrin Fe',
'Etonogestrel',
'Celexa',
'Ethinyl estradiol / norethindrone',
'Polyethylene glycol 3350 with electrolytes',
'Estarylla',
```

Above is a selection of some drugs. Note that some are combinations of multiple chemicals, separated by a slash. There are 19235 of these. We left them as-is, because our reseach into these cases found that they are prescribed this way and come in the same pill.

Dimensionality Reduction

We extracted 200 LSA components from the word-frequency dataset; these components collectively capture 60% of the variance of the word-frequency dataset.

We extracted 200 LSA components from the TFIDF dataset; these components collectively capture 30% of the variance of the TFIDF dataset.



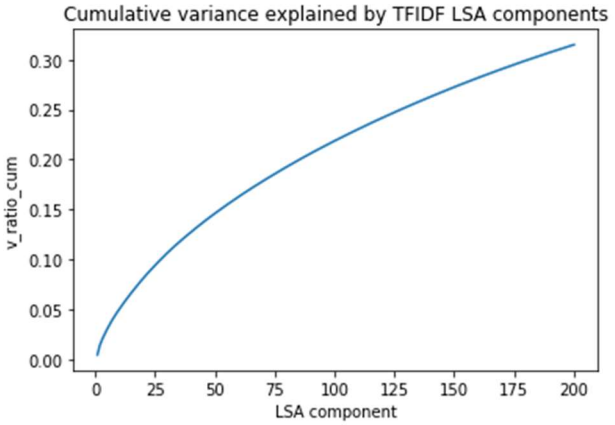


Figure 6a: Cumulative Variance explained by word frequency based LSA components
Figure 6b: Cumulative variance explained by TFIDF LSA components

Clustering – Internal Evaluation

See appendix for complete K-means results and selected plots.

For internal evaluation on the 200-dimension dataset, the word-frequency LSA dataset always performed better than the TFIDF LSA dataset, holding any of the other variables constant. The best CH and SC were found with K=2 and any d≥2 with the word-frequency LSA dataset. We used NMI to compare the clusterings of the tied-for-best values of d, and found that these clusterings were exactly the same.

For internal evaluation on the d-dimensional dataset, CH and SC always decreased with increasing d. Due to the curse of dimensionality and the fact that CH and SC are computed using distance, we expect this trend in general. Because the curse of dimensionality trend occurred monotonically in our results, we cannot easily compare the effectiveness of different d.

In the plot of SSE vs K, for d ≥ 7, the trend is linear, so there is no best K by the elbow method. This could indicate K=2 is the best, or that these are bad clustering s. In the plot of SSE vs K, for d = 1, 2, 4, the trend curves gently along K = 2, 3, 4, and 5, so any of those is a reasonable best K by the elbow method.

The above findings mean that, for the purpose of clustering text like ours, evaluation on a 200-dimension dataset is better than evaluation on a d-dimensional dataset. These are the reasons: our goal is to compare best clustering's across input embeddings of different d (and avoid accounting for the curse of dimensionality); the best K found using SC and CH of the 200-dimension dataset agrees with SSE elbow method that the best K may be K=2; and 200 dimensions cumulatively covers 60% (a proportion that is heuristically pretty good) of the variance of the word-frequency dataset, which was the best dataset for internal evaulation criteria.

Clustering – External Evaluation

For external evaluation using NMI with drug labels and NMI with disease, we found some overarching insights. The TFIDF LSA dataset was always better than the word-frequency LSA dataset. The NMI with diseases was always higher than the NMI with drugs. None of the NMI scores were especially good – the best was around 0.29.

For NMI with drugs, the best scores were around 0.22 and they all had K = 8, 9, or 10. For NMI with drugs, the best scores were around 0.29 and they all had K = 8, 9, or 10. Higher K allows the drug partition or disease partition to match the cluster partition by chance, resulting in higher NMI even though the clusters are not the most helpful. Thus we needed to overcome this with a workaround.

We measured NMI/K for both drugs and diseases, and using this heuristic we found a helpful clustering. This clustering had the best NMI/K with drugs at 0.053 as well as the best NMI/K with diseases, at 0.074. Its parameters were K=2 and 200 components of the TFIDF LSA dataset. A look at the most frequent diseases in each cluster revealed helpful clusters. Cluster 0 can be called "mental health" as the top diseases are Depression, Pain, and Anxiety. Cluster 1 can be called "reproductive health" as the top diseases are Birth Control, Emergency Contraception, and Abnormal Uterine Bleeding.

Thus, the above clustering was the most helpful one that we found, and we found that the best evaluation method to find helpful clusters was the heuristic NMI/K.

Empirical Results (ADHD Dataset Track)

The clustering results showed clear distinction for ADHD data set after T-SNE processing for the persona of individuals who tried a drug a second time. We also saw somewhat of a pattern for the comorbidity persona.

Clustering Results on TFIDF Architecture on t-SNE space

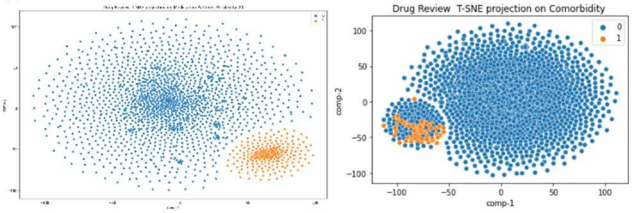


Figure7a(left) Persona of "has tried more than one drug" colored in orange, Figure 7b(right): Persona of "has comorbidity" colored in orange. Both were run on the same TFIDF architecture for ADHD dataset.

Clustering Metric evaluation on t-SNE space

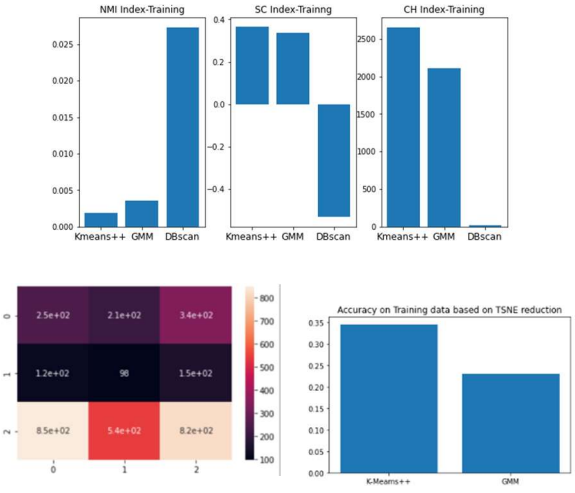


Figure Cluster Evaluation: 8a. (upper left) NMI Indexing 8b. (upper middle) SC indexing .8c (Upper Right)CH Index 8d (Bottom Left) Confusion Matrix for KNN Clustering with External Label Rating, Performance Evaluation of 3 clustering criterion (Silhouette Score, DBSCAN and Gaussian Mixture Model).8d Bottom Right Diagram shows the accuracy of right label prediction on GMM and k-means++ clustering

Although all three clustering perform poorly in these evaluation criteria, DBSCAN performs somewhat better when we compare Mutual Information based on external labels. However, GMM and K-means++ perform well on CH Index for training Data Set.

For external evaluation, we evaluated clusters using 3 bins of rating. Ratings less than 5 got label 0, ratings between 5-7 inclusive got label 1, and rating greater than 7 got label 2.

Confusion Matrix is based on reduced TFIDF Transformation on TSNE space. The K-means++ clustering performs better as compared to GMM clustering.

Rating Evaluation

We then divide the training data set to a train set and a validation set (33% of Training set) to validate performance of our model. The objective of the model is to predict the rating of Drug /medicine taken by the patient and usefulness of it. We have primarily used Neural Network with multi-layers where each input sends output function which is used as input in next layer. This is equivalent to multiple functions cascaded to each other. We have used ReLU activation function and used means square error as optimizing function. When then extended the similar neural architecture to TFIDF architecture. The only change with the TFIDF architecture is the input data has 600 features which represents 80% of variance from original TFIDF Architecture. We can see validation test results for TFIDF architecture is showing better results in rating of medications. This could be because the feature vectors are 600 in TFIDF architecture as compared to 16 vectors in Doc 2 Vector Architecture.

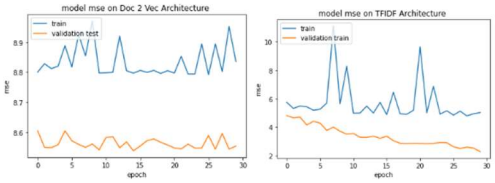


Figure 9a(Left) represents the MSE error between the actual rating and predicted rating for Doc-2-Vec architecture
Figure 9b (Right)Model MSE TFIDF Architecture

The boxplot shows the neural network model based on 600-feature TFIDF architecture with 3 dense layers

had lower mean square error in validation test as compared to a similar neural network model on Doc 2vector architecture. Hence, we will use the Neural network-based model on TFIDF architecture on test data set as it had better results in validation set.

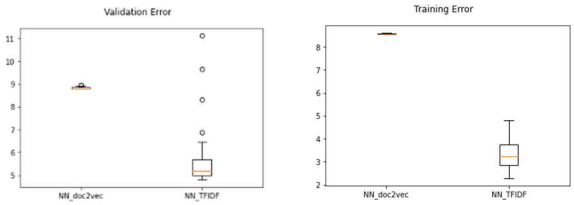


Figure 10. Box Plot on Validation & Training subset : Comparison of MSE error in various epochs on 2 different architecture -Doc2 Vec(16 features) vs TFIDF(600 Features)

Recommendation Engine

Once ratings are created from review, we were left with 3 major options to run recommendation engine. The three options are given below:

Models	Initial Heuristics	Findings
Item-Item Collaborative Filtering	Since users were more than items we initially preferred this approach. However we later realized that sparsity was an issue with this recommendation system	We can calculate drug similarity index but the average rating of user would be same as rated for drug.
User-User Collaborative Filtering	Though items are less than users we felt this would not be the best approach	We computed similarity matrix of each user which was either 1 or 0 in our case. If similarities were found we recommend the highly rated drug to customer. If similarity was not found
Content Based Filtering based on persona of each patient.	This seems a logical approach as we had got personas as we had got in section 1	We were able to predict scores of each drug for a given persona and then computed the maximum rated drug which could be used by patient through linear regression.

Table 3: Recommendation Systems

Hence our recommendation engine was able to successfully able to predict rating of each drug given the user features. We then found the drug which had the highest rating given the persona. This highest rated drug was then recommended as part of our recommendation. Following was the test example of our target segment. We selected this patient as he had comorbidity and had taken multiple drugs before reaching final drug.

Patient id	Persona Type	Drug	Predicted Rating
250	4 Comorbid +Prior Meds	Lisdextameftin	6.98

Table 4: Example of a Target Patient of Our Recommendation System

Based on our Content based filtering (Personas), we were able to get $\vec{\theta}$ and were able to predict the best drug for the patient . Our recommendation system thus predicts that ingestion of “Dexedrine” could be beneficial for this patient as it had the highest rating for the user with these features among all the drugs. We assume in this case that Dexedrine does not cause any drug reaction.

Conclusions and Future Directions

Conclusions

1. We found that word-frequency LSA-processed data with K-means K=2 produces the clusterings with the best internal clustering criteria.
2. TFIDF LSA-processed data with high number of components with K-means K=2 gives the most helpful clustering with respect to true disease labels and true drug labels.
3. We found that Neural Network based TFIDF based architecture had better prediction on rating of post than Document to Vector architecture. This was looks slightly surprising because Document 2 Vector architecture has contextual meaning of document. However, if we look at it in deeply we had used only 16 feature Document 2 vector model as compared to 600 feature TFIDF model. The accuracy of 32,64 feature vector models can be compared as a future study with existing TFIDF architecture .
4. Linear regression of TFIDF architecture also had better performance than 16 vector - Document 2 Vector Model
5. Content based filtering is pretty robust in a very sparse recommendation matrix as compared to user-user based filtering mechanism. This result was as expected.
6. We were successfully able to identify the audience who could benefit from our recommendation system.
7. We were successful to identify the best drug suitable to our targe patients-one with comorbidities and who had tried multiple drugs. The rating of new target drug was much higher.

Future Directions

The dataset was challenging as every unique user had only 1 drug which made collaborative filtering inconsequential. Data on previous used drugs could help in expanding our recommendation system to collaborative filtering.

While our drug recommendation system was successful in predicting the rating of score, the model was based on 4 personas. We could make disease-specific personas and could make new models for diseases other than ADHD, and we would try making models for groups of diseases to see if that improves results as compared to single diseases.

While we focused on people who have tried multiple drugs and have comorbidities, we may be able to target other demographics as well and see if demographics has some role in it. Future work can include use of topic modeling to generate personas and see if these personas led to a better recommender system. Given our success with content-based modelling, future work can be done with group of medical domain experts to add more features at user or drug level like personas with allergies or drug type to bring more robustness to industrial application of our system. Domain experts and pharmacists can provide additional variable like drug type as a feature that could help in content-based filtering based on drug features as well.

Future work can also incorporate cluster labels and network community labels in creating our personas. more ways of clustering, such as DBSCAN and spectral clustering, and insights can be gained about network analysis (using reviews as nodes, connected by common drugs or common diseases) finding communities through spectral clustering.

References

Felix Gräßer, Surya Kallumadi, Hagen Malberg, and Sebastian Zaunseder, 2018. “Aspect-Based Sentiment Analysis of Drug Reviews Applying Cross-Domain and Cross-Data Learning.” In Proceedings of the 2018 International Conference on Digital Health (DH '18). ACM, New York, NY, USA, 121-125.

Ian T. Jolliffe, 2002 “Principal Component Analysis, 2nd edition.”

Jessica Li, 2018. “UCI ML Drug Review Dataset.” Kaggle. https://www.kaggle.com/jessicali9530/kuc-hackathon-winter-2018

D. Sculley, 2010. “Web-Scale K-Means Clustering.”.

Scikit-learn developers, 2020. “TruncatedSVD.”

View publication stats