Disease Prediction and Drug Recommendation System

CMPE 255, Group 3

Anurag Upadhyay (014371240)

Khushboo Ekhande (014370837)

Rakshitha Sathyakumar (014511705)

Sughosh Krishnamurthy (014370954)

Abstract

Data Mining is a method that requires analyzing and exploring large blocks of data to extract meaningful trends and patterns. Data mining techniques can be applied to various fields including medical databases. There are thousands of people all over the world facing health and medical diagnosis problems. Hospital Information System (HIS) generates massive data but gaining useful knowledge from the diagnosis case data is a big challenge. Using the methodologies used in this project, patients can easily get information about the disease they are suffering from and the drug helpful for dealing with that disease by just entering the symptoms he/she is showing.

In this project, we recommend drugs to the users based on the diseases and the reviews. The diseases are predicted using four different models. The reviews are analyzed using the Vader tool and NLP based sentiment analysis. And finally the drugs are recommended using probabilistic and weighted average approaches. The details of each model and approach used in this project are explained in detail. The experimental results from this paper can be further utilized for research purposes and for other various medical utilities.

Introduction

One of the most commonly found concerns among patients when confronted with any medical condition is which physician to trust. It is a known fact that the health of an individual significantly affects his/her quality of life. A survey in 2013 by the Pew Internet and American Life Project found that 59% of adults have looked online for health topics and with 35% of respondents focusing on diagnosing a medical condition online. There are more people every day caring about the health and medical diagnosis problem but still many who lose their lives due to medical errors. According to the administration's report, more than 200 thousand people in China and over 100 thousand in the USA, die each year due to medication errors. More than 42% medication errors are caused by doctors because they write prescriptions based on their experience which is quite limited. Hence, finding appropriate physicians to diagnose and treat medical conditions is one of the most important decisions a patient must make.

Advancement in Data mining and Recommender Technologies allow us to explore possibilities of potential knowledge from diagnosis history records and reviews and ratings on drugs to help doctors prescribe the correct medication and decrease the medication errors effectively.

The objective of this Data Mining paper is to design and implement a universal Disease Prediction and Drug Recommendation System that applies various Data Mining technologies to the recommendation system. By combining information from different sources we are using various prediction algorithms along with NLP for sentiment analysis and recommendation. Rest of the report talks about Data Gathering, Pre-processing, Methodology, Results and Conclusion of our project.

Related Work

I. Kamaraj, K.Gomathi & Priyaa, D.Shanmuga. (2016)

In this paper, they have demonstrated results of Decision Tree and Naive Bayes models in predicting three diseases, Heart Disease,Diabetes and Breast Cancer. [1]

II. M.A.Nishara Banu, B Gomathy. (2013)

In this paper, they have predicted heart diseases. By applying K-Mean on the medical dataset, they have clustered the relevant data, upon which MAFIA(Maximal Frequent Itemset Algorithm) is applied to generate rules and identification of frequent pattern which is fed to the C4.5 (Decision Tree) model to classify patterns. [2]

III. H Wang Q Gu J Wei Z Cao Q Liu (2015)

In this paper, determine novel drug indications and side effects in one integrated framework. This strategy provides a complementary method to medical genetics-based drug repositioning, which reduces the occurrence of false positives in medical genetics-based drug repositioning, resulting in a ranked list of new candidate indications and/or side effects with different confidence levels. [3]

IV. Yin Zhang, Daqiang Zhang, Mohammad Mehedi Hassan, Atif Alamri & Limei Peng (2014)

In this paper, we propose a novel cloud-assisted drug recommendation (CADRE), which can recommend users with top-N related medicines according to symptoms. In CADRE, we first cluster the drugs into several groups according to the functional description information, and design a basic personalized drug recommendation based on user collaborative filtering.

V. Youjun Bao ; Xiaohong Jiang. (2016)

In this paper, they design and implement a universal medicine recommender system framework that applies data mining technologies to the recommendation system. The medicine recommender system consists of a database system module, data preparation module, recommendation model module, model evaluation, and data visualization module.

Datasets and Preprocessing

Healthcare information is protected by HIPPA. Sharing of medical records of patients without their knowledge is prohibited. Getting access to government health records and datasets required multiple permissions. Hence, for our project, we are using the datasets that were readily available on the internet and were open to downloads.

Dataset One

Data Gathering

The 10-year medical record dataset was obtained from Medical records 10 yrs - dataset by arvin6 | data.world.

It consists of four CSV files, namely

- encounter.csv
- encounter_dx.csv
- lab_results.csv
- medication_fulfillment.csv

The encounter.csv consists of 1176 rows and 17 columns, encounter_dx.csv consists of 3063 rows and 6 columns, lab_results.csv consists of 7509 rows and 21 columns, and the medical_fulfillment.csv consists of 5447 rows and 28 columns. In order to obtain useful insights and understand if the dataset has required information to deliver the problem statement, we preprocessed and merged the dataset to cater to our needs.

Data Preprocessing

After gathering the raw data and understanding the shape of all the four CSV files, we preprocessed each table and combined the tables to get a single merged dataset with the required columns. To preprocess, we executed certain commands to understand the data. Like, count of each row, the number of unique values, dropped a few columns which were irrelevant to achieve the end goal, dropped columns which had no data values, and combined a few columns to make it more usable.

Once the basic preprocessing was done, we then found out that the four tables were managed using the Star Schema Model and medical_fulfillment.csv is the fact table and ecounter.csv, encounter_dx.csv, and lab_results.csv are dimension tables. The star schema

follows the one-many relationship. We found out the Primary Key and Foreign Key and merged the four tables to form a single table. The medication_fulfillment table has 'Ecounter_ID' as the primary key. We then merged the Medication_fulfillment table with columns 'severity' and 'description' from the encounter_dx table using left join on 'Encounter_ID' by running a SQL query. The resulting table has 1176 rows.

Next, to check if 'order_ID' is the primary key of lab_results.csv, we ran a query to check the count of each row in 'order_ID' column and found out that it's not the primary key as the number of unique values didn't match the number of rows of the table. We then found out that 'Order_ID' and 'Result_LOINC' together make the composite primary key. Since none of the columns from lab_results.csv was useful, we didn't use any columns for the merged dataset.

With encounter.csv remaining, we found that 'Encounter_ID' is the primary key and extracted the CC column and merged it using the left join on 'Encounter_ID'. Now we have a complete merged dataset of the required columns. It consists of 1176 rows. From the merged dataset, we extracted the Drug_Name, description, severity, and CC and grouped them to get the total number of each drug and their associated disease and description. The extracted columns consist of many null values as shown in the figure below.

	Drug_Name	description	severity	CC	cnt
0	OMS 50	Chronic Obstructive Pulmonary Disease	critical	critical shortness of breath	119
1	Ciprofloxacin	None	None	None	105
2	Isotonic Saline (0.9%)	None	None	None	101
3	Lisinopril	None	None	None	83
4	Potassium Chl	Type 1 Diabetes	severe	severe increased thirst	78
...	...	...	...	...	...
76	metoprolol	Hypertension	severe	moderate difficulty walking	1
77	oxycodone-acetaminophen 10-325	None	None	None	1
78	trimethoprim	None	None	None	1
79	trimethoprim	Pyelonephritis	severe	Pyelonephritis	1
80	valsartan	Chronic Congestive Heart Failure	severe	mild palpitations	1

Table 1: Drug name grouped by description, severity, and CC

Hence, we ran another query to get the total number of rows which consist of null values corresponding to a drug name. As a total of 764 rows had null values, we were only left with 416 rows of relevant data to train our classification model. Hence, we concluded that this dataset does not serve the purpose and looked for more datasets that would align with the delivery of our problem statement.

Dataset Two

Data Gathering

For the accurate recommendation of drugs, we first predict the diseases based on symptoms and then recommend the drugs based on the ratings. For this purpose, we have tried to gather information from two main datasets.

1. Symptoms dataset:

- This dataset is used to take symptoms as input and predict the disease as an output.
- Dataset is obtained from the Disease-Symptom Knowledge Database which is a knowledge database of disease-symptom associations generated by an automated method based on information in textual discharge summaries of patients at New York-Presbyterian Hospital admitted during 2004.
- This dataset contains 3 columns:
    1. Disease
    2. Count of Disease Occurrence
    3. Symptom

	Disease	Count of Disease Occurrence	Symptom
0	UMLS:C0020538_hypertensive disease	3363.0	UMLS:C0008031_pain chest
1	NaN	NaN	UMLS:C0392680_shortness of breath
2	NaN	NaN	UMLS:C0012833_dizziness
3	NaN	NaN	UMLS:C0004093_asthenia
4	NaN	NaN	UMLS:C0085639_fall

Table 2: Raw Symptoms dataset

- There are a total of 149 unique diseases in this dataset and 405 symptoms.
- Each disease contains 4-5 symptoms corresponding to it.
- This dataset is sent for pre-processing so that it can be used to train models to classify and predict the disease.

2. Drug Review Dataset:

- This dataset is used in order to take the predicted disease as input and recommend appropriate drugs based on reviews and ratings (Sentiment Analysis).
- The dataset is gathered from the UCI Machine Learning Repository for Drug Review which provides patient reviews on specific drugs along with related conditions and a 10-star patient rating reflecting overall patient satisfaction.
- The Repository had two datasets (Test and Train) which are combined for analysis and visualization purposes as they had the same number of columns.
- It contains 7 columns and 215063 rows:
    1. ID
    2. Drug name
    3. Condition
    4. Review
    5. Rating
    6. Date
    7. Useful count

	uniqueID	drugName	condition	review	rating	date	usefulCount
0	206461	Valsartan	Left Ventricular Dysfunction	"It has no side effect, I take it in combinati...	9.0	20-May-12	27
1	95260	Guanfacine	ADHD	"My son is halfway through his fourth week of ...	8.0	27-Apr-10	192
2	92703	Lybrel	Birth Control	"I used to take another oral contraceptive, wh...	5.0	14-Dec-09	17
3	138000	Ortho Evra	Birth Control	"This is my first time using any form of birth...	8.0	3-Nov-15	10
4	35696	Buprenorphine / naloxone	Opiate Dependence	"Suboxone has completely turned my life around...	9.0	27-Nov-16	37

Table 3: Raw Drug Review Dataset

- There are 3671 unique Drug names and 916 unique Conditions (Disease) in this dataset along with the rating and reviews corresponding with the drug names.
- This dataset is then pre-processed and visualized to gain more information for effective drug recommendation.

3. Side Effects Dataset

- We have successfully included this dataset containing side effects of specific drugs in order to help patients identify the risks involved in the drug that is being recommended.
- This dataset is again gathered from UCI Machine Learning Repository for Side Effects of Drugs along with some raw data gathered from druglib.com.
- This Dataset contains a lot of columns similar to the drug review dataset, but it does not have a lot of rows. Hence only the "Side Effects" column from this dataset will be combined with the other two datasets along with the side effects found from druglib.com

urlDrugName	rating	effectiveness	sideEffects	condition	benefitsReview	sideEffectsReview
enalapril	4	Highly Effective	Mild Side Effects	management of congestive heart failure	slowed the progression of left ventricular dys...	cough, hypotension , proteinuria, impotence , ...
ortho-tri-cyclen	1	Highly Effective	Severe Side Effects	birth prevention	Although this type of birth control has more c...	Heavy Cycle, Cramps, Hot Flashes, Fatigue, Lon...

Table 4: Raw Dataset containing Side effects of drugs

Data Preprocessing

In this project, we have used and worked upon multiple datasets. All the datasets were obtained in raw format. To preprocess all the datasets, a few common steps and measures were carried out. Those were:

- All the datasets were first checked for the count of null and missing values.
- All such values were either handled or dropped from the dataset.
- After that, every column's unique values were found and their frequencies.
- Using standard libraries, the dataset was visualized and outliers, if any, were found out.
- Any irrelevant information was deleted from the dataset.

1. Symptoms Dataset

- This dataset had to be cleaned in order to gain information from it.
- First, the “Count” column was dropped as the information given in it is irrelevant for this project.
- Then, the null values in the Disease column were handled using the drop function.

```
data = df.fillna(method='ffill')
```

- Cleaning of Disease and Symptom columns was done to get only the name and removed unnecessary data.

	Disease	Symptom
0	hypertensive disease	[pain chest, shortness of breath, dizziness, a...
1	diabetes	[polyuria, polydypsia, shortness of breath, pa...
2	depression mental	[feeling suicidal, suicidal, hallucinations au...
3	depressive disorder	[feeling suicidal, suicidal, hallucinations au...
4	coronary arteriosclerosis	[pain chest, angina pectoris, shortness of bre...

Table 5: Symptoms dataset after preprocessing

- For classification of the symptoms based on diseases, we have converted it into a new CSV format file which now has symptoms as the columns and diseases as rows. Using one hot encoding, we have mapped every symptom with all the diseases and adding value 1 if it is present for disease and 0 otherwise. Below is a screenshot of the one-hot encoded dataset. This will help us predict the diseases when symptoms are given as input.

	Disease	Heberden's node	Murphy's sign	Stahli's line	abdomen acute	abdominal bloating	abdominal tenderness	abnormal sensation	abnorma hard consiste
0	Alzheimer's disease	0	0	0	0	0	0	0	0
1	HIV	0	0	0	0	0	0	0	0
2	PneumocystisÂ cariniiÂ pneumonia	0	0	0	0	0	0	0	0
3	accidentÂ cerebrovascular	0	0	0	0	0	0	0	0
4	acquiredÂ immuno-deficiency syndrome	0	0	0	0	0	0	0	0

5 rows × 405 columns

Table 6: Dataset after marking symptoms present for a disease as 1 else 0.

2. Drug Review Dataset

- This dataset contained two sets (Train and Test) which were combined to be able to visualise and analyze the data on a larger dataset. And also because they both had the same columns so could be combined easily.
- The dataset obtained was pretty clean and did not require a lot of pre-processing. Still, a few rows with null values were dropped and columns were renamed.
- The dataset contains a lot of information and visualizing was an interesting task.
- Many different graphs were plotted showing results of drugs with most reviews, most popular drugs, most common diseases, etc.
- One such visualization is shown below which has names of a few most popular drugs:

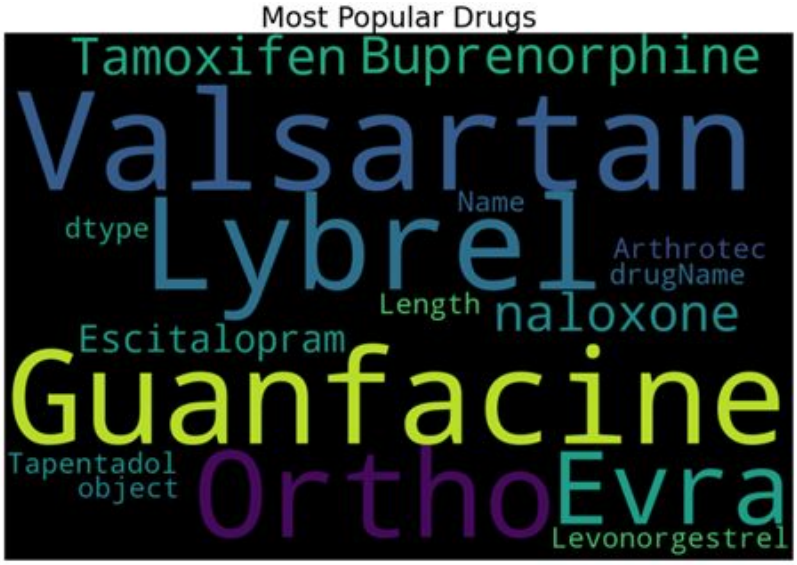

Figure 1: Visualizing the most popular drugs based on the ratings

3. Side Effects Dataset

- This dataset has a lot of information similar to the one in the Drug Review dataset and it was also clean.
- A few null values are handled and irrelevant columns are dropped.
- This dataset is merged with the drug review dataset to map only the side-effects for the specific drugs.

4. Merged Dataset

- The Dataset containing symptoms is merged with the one with reviews for final drug prediction.

```
data_drugs['Disease'] = data_drugs['Disease'].str.lower()
data_symptoms['Disease'] = data_symptoms['Disease'].str.lower()
```

```
merged_data = pd.merge(data_drugs, data_symptoms, on='Disease')
```

```
merged_data.head()
```

	drugName	Disease	review	rating	usefulCount	Symptom
0	Aspirin	transient ischemic attack	"No side effects, easy to take, no more sympt...	10.0	10	['speech slurred', 'dysarthria', 'facial pares...
1	Clopidogrel	transient ischemic attack	"I've been taking this medicine for a lit...	10.0	8	['speech slurred', 'dysarthria', 'facial pares...
2	Clopidogrel	transient ischemic attack	"I took ibuprofen (2 caps at night for severe ...	6.0	13	['speech slurred', 'dysarthria', 'facial pares...
3	Clopidogrel	transient ischemic attack	"After my VAD Stroke I am on plavix. I have a...	5.0	9	['speech slurred', 'dysarthria', 'facial pares...
4	Bayer Children's Aspirin	transient ischemic attack	"No side effects, easy to take, no more sympt...	10.0	10	['speech slurred', 'dysarthria', 'facial pares...

Table 7: Merged Dataset

- Only the relevant columns are kept in the merged dataset and rest are dropped to reduce the dimensionality.

Methodology

The main goal of our project is to recommend a drug to a patient based on the symptoms she/he has. In accordance with our objective to implement a drug recommender system there are two main subcategories which are to be addressed i.e a disease prediction model and a recommendation model and below is the design pipeline and dataflow of our implementation.

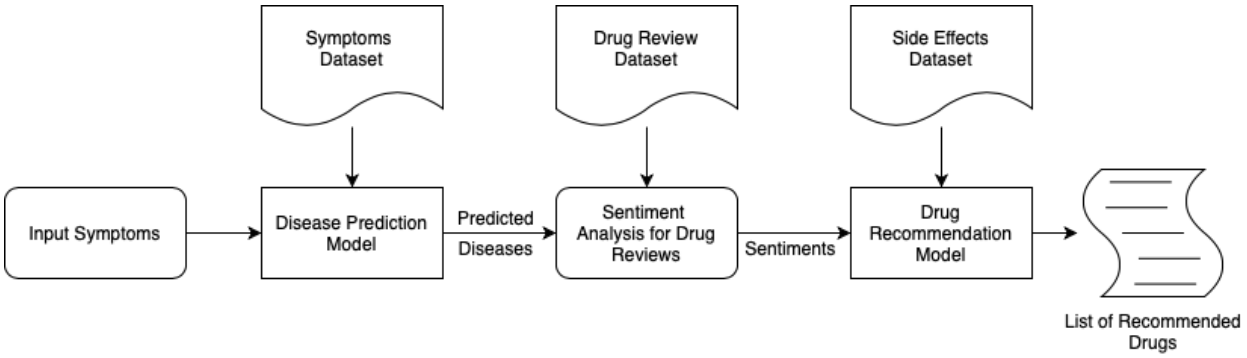

Figure 2: Design implementation pipeline and dataflow of our recommender system. Disease prediction model is a classification model with takes in the symptoms and gives the predicted diseases and the sentiment analysis on the drug reviews and mapped to predicted disease and finally recommended drugs based on the sentiments of the reviews and rank along with possible side effects

Disease Prediction

The disease prediction model is a probabilistic model which will give the predictions based on the symptoms. For this purpose, we are using the Disease-Symptom Knowledge Database which has over 149 unique diseases mapped with 405 symptoms.

For accurately predicting the diseases, we have experimented with multiple approaches in our project. This is because the dataset has only one feature that is the symptoms and hence training any classifier with one data point makes it weak which in turn hampers the prediction rate for different inputs. The solution is to try and create multiple data points for the same disease by taking into account all the symptoms present for that disease, their importance and their occurance. Since we cannot say for sure which method will be well suited for this kind of dataset, we have implemented 4 different approaches and all of them focus on a different way of prediction. In the end, all these approaches are trained on 4 classifiers and the accuracies and prediction results are compared for choosing the best possible way. The approaches are explained in detail below:

Approach 1

In this method, the data has been preprocessed, transformed and mapped to a data frame which consists of columns derived from the list of unique symptoms and the list of unique target diseases as labels. The preprocessed symptoms dataset was used to map the newly created disease prediction data frame by setting the values of symptoms present for a particular disease as 1 and 0 if not. And the rows were grouped by to obtain a single row for each unique disease with the corresponding symptoms column mapped to 1. The data was then trained using Multinomial Probabilistic Model, ExtraTree Classifier Model, Decision Tree Classifier Model, Support Vector Machine Classifier Model, using different Symptoms as the training features and Diseases as the labels. And the trained model is used to predict the diseases by passing one, two, and three features to the classifier models.

Approach 2

In method one, each classifier provided the same accuracy of 89%. Even though the accuracy was 89%, the prediction of the disease using symptoms as input, were not accurate. In order to make the disease predictions precise, we implemented method two. In this method, we have considered the ‘ Count of Disease Occurrence’ column values to increase the number of rows in the dataset and the symptoms are considered to have equal importance. Since Method 1 has just one row for each disease, we couldn’t split the data into train and test datasets. Splitting the data gave us 0% accuracy as the classifier is trained on different diseases and the test data consists of different set of diseases. The preprocessed symptoms dataset was used to map the newly created disease prediction data frame by setting the values of symptoms present for a particular disease as 1 and 0 if not. And the rows are grouped based on the diseases and each row of the disease is multiplied based on the count of disease occurrence. The final data set has 38839 rows. The dataset was then split into training and testing data. The data was then trained using Multinomial Probabilistic Model, ExtraTree Classifier Model, Decision Tree Classifier Model, Support Vector Machine Classifier Model, using different Symptoms as the training features and Diseases as the labels. The trained model is used to predict the diseases by passing one, two, and three features to the classifier models.

Approach 3

Even after augmenting data based on the count of disease occurrence, the classifier produced the same accuracy rates of 86%. With 86% accuracy rate, the disease prediction using method 2 was not accurate enough. Hence we tweaked the data in such a way that the classifiers are able to draw patterns and learn accordingly. In this method, the order of importance of each symptom for a disease is preserved. The symptoms dataset was used to map the newly created disease prediction data frame by setting the values of symptoms present for a particular disease as 1 and 0 if not. The number of rows for each disease is multiplied to increase the number of rows. This will create a data frame with an increased and equal number of rows for each disease. The mapping of 0's and 1's are done in the probabilistic manner. The symptom with the highest importance is mapped to 1 for all the rows of that particular disease. The second most important symptom is mapped to 95% of all the rows and so on. The least important symptom will approximately be mapped to 40% of the total rows of a particular disease. The dataset was then split into training and testing data. The data was then trained using Multinomial Probabilistic Model, ExtraTree Classifier Model, Decision Tree Classifier Model, Support Vector Machine Classifier Model, using different Symptoms as the training features and Diseases as the labels. The trained model is used to predict the diseases by passing one, two, three, and four features to the classifier models.

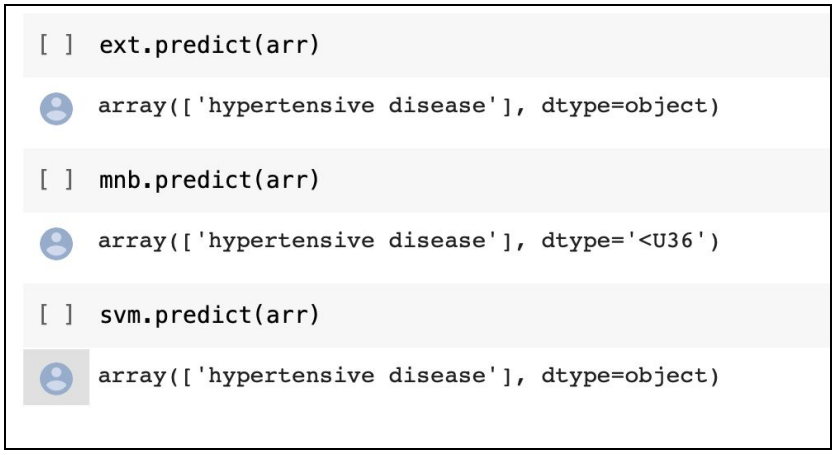

Figure 3: Accurate durg predictions for four input symptoms

Approach 4

In this approach, we have attempted to create a dataset wherein all the symptoms present for a particular disease are randomly distributed among different rows. This gives a dataset which has multiple rows for a single disease along with it's symptoms marked 0 or 1 different in each row.

- To create this dataset, we have created small datasets by using the groupby function for different number of rows each time.

```
df_pivoted_1 = df_pivoted.groupby('Disease').head(3)
df_pivoted_2 = df_pivoted.groupby('Disease').head(5)
df_pivoted_3 = df_pivoted.groupby('Disease').head(7)
df_pivoted_4 = df_pivoted.groupby('Disease').head(10)
df_pivoted_5 = df_pivoted.groupby('Disease').head(12)
df_pivoted_6 = df_pivoted.groupby('Disease').head(15)
df_pivoted_7 = df_pivoted.groupby('Disease').head(17)
df_pivoted_8 = df_pivoted.groupby('Disease').head(20)
df_pivoted_9 = df_pivoted.groupby('Disease').head(24)
df_pivoted_10 = df_pivoted.groupby('Disease').head(28)
```

Figure 4: Code snippet to create small datasets from the original one

- Then performed groupby.sum() to combine all symptoms in one row for each small dataset.

```
df_pivoted_1 = df_pivoted_1.groupby('Disease').sum().reset_index()
df_pivoted_2 = df_pivoted_2.groupby('Disease').sum().reset_index()
df_pivoted_3 = df_pivoted_3.groupby('Disease').sum().reset_index()
df_pivoted_4 = df_pivoted_4.groupby('Disease').sum().reset_index()
df_pivoted_5 = df_pivoted_5.groupby('Disease').sum().reset_index()
df_pivoted_6 = df_pivoted_6.groupby('Disease').sum().reset_index()
df_pivoted_7 = df_pivoted_7.groupby('Disease').sum().reset_index()
df_pivoted_8 = df_pivoted_8.groupby('Disease').sum().reset_index()
df_pivoted_9 = df_pivoted_9.groupby('Disease').sum().reset_index()
df_pivoted_10 = df_pivoted_10.groupby('Disease').sum().reset_index()
```

Figure 5: Code snippet to combine the rows to form one single row

- Finally, the small datasets which have different values of symptoms for each disease are combined together to form one large dataset which has multiple rows for each disease with different data points!

```
train_data = pd.concat([df_pivoted_1,df_pivoted_2,df_pivoted_3,df_pivoted_4,df_pivoted_5,
                        df_pivoted_6,df_pivoted_7,df_pivoted_8, df_pivoted_9, df_pivoted_10], axis=0, sort=True)
```

- This approach helped us to train the models in a better way making the classification more accurate.

Sentiment Analysis of drug reviews

Upon obtaining the most probable diseases the next task is to map the list of drugs that can be prescribed for this particular disease using the Merged Dataset which has been created by preprocessing the symptoms and the drug review dataset. Once we obtain the list of possible drugs the next task is to be able to recommend the best drug for the patient. In accordance we adopted two different approaches for this. First one is a NLP based approach to analyze the sentiments using a neural net model in order to obtain positive or negative sentiment predictions for the reviews. In the second approach we have used the VADER tool. It is a simple rule based model for general sentiment analysis.

Approach 1:

In this approach, we will be adopting sentiment analysis using the Natural Language Processing on the drug review data set to understand the trend in the positive and negative reviews given by the patients. The main rationale behind doing this sentiment analysis was the rating given and the review stated was seen to be inconsistent so likely if the review stated it was good and no side effects it still had a rating below 5 which made it hard to just make use of the rating based on some threshold and as NLP captures the essence of a sentence to predict a sentiment hence it proves to be more reliable. The NLP model was designed based on the word2vec method and a neural network classifier was built to classify the reviews. The model was designed by extracting the probabilistic values of relational occurrence using TfidfVectorizer that convertes the words in the sentences into a vector of probabilities. Upon obtaining the vectors a neural network model using sequential method was trained on this data to obtain the sentiment predictions. The model totally consisted of 3+1 layers with each layer having batch normalization and relu activation and the last fully connected layer has softmax activation function and binary cross entropy loss was used with adam optimizer. The neural network model implementation was completely done using keras API and the model summary is as in Figure 3.

Algorithm: Sentiment Analysis; epochs: 25, l_rate:1e-1, batch_size = 100

Input
X: Reviews
y: Ratings
Output
Drugs, Reviews, Disease and Sentiments
steps:

1. encode the rating with threshold of 7 into two classes (>7 and <7) - input: y
2. get Tfidf vectors for the vectors - input: X
3. define sequential model using keras API
4. for epochs in steps per epoch
    a. train(fit) the model
    b. update gradients
5. get predictions for all the reviews and merge the data to original dataset

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 25)                315550
_________________________________________________________________
batch_normalization (BatchNo (None, 25)                100
_________________________________________________________________
activation (Activation)      (None, 25)                0
_________________________________________________________________
dropout (Dropout)            (None, 25)                0
_________________________________________________________________
dense_1 (Dense)              (None, 25)                650
_________________________________________________________________
batch_normalization_1 (Batch (None, 25)                100
_________________________________________________________________
activation_1 (Activation)    (None, 25)                0
_________________________________________________________________
dropout_1 (Dropout)          (None, 25)                0
_________________________________________________________________
dense_2 (Dense)              (None, 25)                650
_________________________________________________________________
batch_normalization_2 (Batch (None, 25)                100
_________________________________________________________________
activation_2 (Activation)    (None, 25)                0
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 52
=================================================================
Total params: 317,202
Trainable params: 317,052
Non-trainable params: 150
_________________________________________________________________
```

Figure 6:: Neural Network model summary

Approach 2:

The VADER stands for Valence Aware Dictionary and sEntiment Reasoner. It is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. This model does not require any training data as it already uses a combination of qualitative and quantitative methods to produce and then empirically validate a gold-standard sentiment lexicon to evaluate the sentiment of the sentence. This tool will take a string input which will be a drug review from our dataset. This tool not only tells us about the polarity (positive or negative) of the review, but also how much positive or negative the review is. There are many advantages of using this tool. They include[7]:

1. It works exceptionally on the social media style texts which may include emoticons and punctuations.
2. It requires no training data.
3. Can be used with real time streaming data.
4. It does not suffer from a speed-performance tradeoff.

The output generated has 4 components i.e. Positive, Negative, Neutral and Compound. Positive, Neutral and Negative specify how many parts of the sentence have the tone as positive, neutral and negative respectively. It is specified in the form of decimal numbers and the sum of these three components will always be equal to 1. The compound component specifies the overall sentiment of the sentence. Below thresholds help us to interpret results of compound:

1. Positive sentiment : compound score >= 0.05
2. Neutral sentiment : -0.05 < compound score < 0.05
3. Negative sentiment : compound score <= -0.05

You can see the output of a sample review in the below figure. So here in the sentence, there is a bit of positive part in the sentence but the overall sentiment can be computed as negative which is evident by the compound value given by VADER.

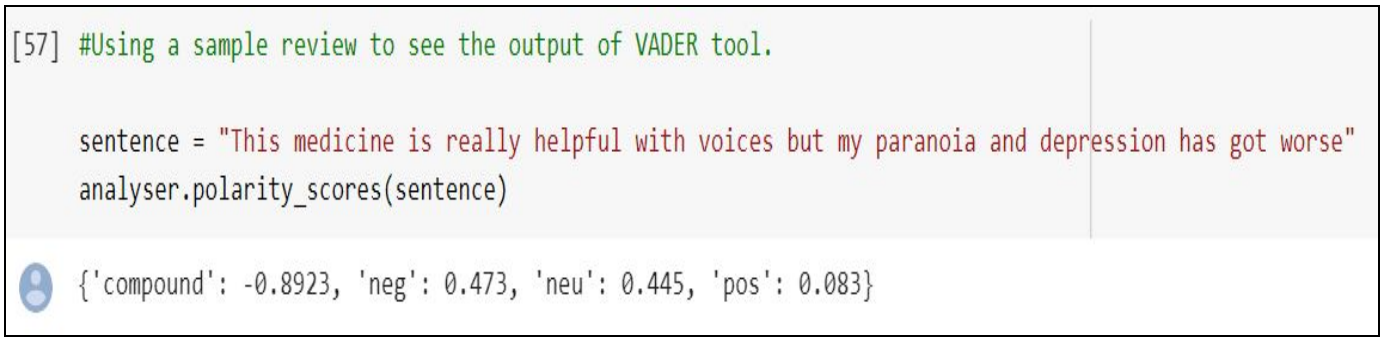

Figure 7: VADER analysis for a sample review from our dataset.

Drug Recommendation

After the disease has been predicted, we need to recommend a drug for that disease. For recommendation of the drug, we have used the UCI Machine Learning Repository for Drug Review dataset which has the disease along with its multiple available drugs, their reviews, ratings and useful count.

This dataset, after preprocessing, is merged with the Disease-Symptom Knowledge Database on the basis of common diseases as shown in the section for preprocessing of merged dataset.

There are 6 features in this merged dataset out of which 3 are the ones that will help us to recommend the best drug. These 3 features and their importance is explained below:

- **Feature 1. Review:**

  The review column is one of the most important columns for drug recommendation as it is a direct feedback from the users after using that particular drug. Hence, by doing Sentiment Analysis, we have taken into consideration only the reviews with a positive sentiment.

- **Feature 2 & 3. Rating & Useful Count:**

  Every drug has a rating associated with it from 1-10 and a useful count which tells us how many users have given that particular rating. This information can be used to find a weighted average rating or a probabilistic value for each drug. Both the approaches are explained below:

Approach 1: Weighted Average Approach:

Weighted average or mean is similar to an ordinary arithmetic mean, except that instead of each of the data points contributing equally to the final average, some data points contribute more than others.

Formula for weighted average:

$$\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i},$$

which expands to:

$$\bar{x} = \frac{w_1 x_1 + w_2 x_2 + \cdots + w_n x_n}{w_1 + w_2 + \cdots + w_n}.$$

Figure 8:

where, data elements with a high weight contribute more to the weighted mean than do elements with a low weight.

In our case, x is the rating and w is the useful count. In this way, we get a weighted average rating which gives more importance to the useful count i.e. the number of people who have actually given that rating.

This is the function created for calculating weighted average:

```
def wavg(group, avg_name, weight_name):
    d = group[avg_name]
    w = group[weight_name]
    try:
        return (d * w).sum() / w.sum()
    except ZeroDivisionError:
        return d.mean()
```

```
data_wavg = data.groupby(["Drug"], as_index=False).apply(wavg, "Rating", "UsefulCount")
```

After calculating , this weighted average column is merged with the dataset which replaces the rating column as shown:

	Disease	Drug	Review	Review_Sentiment	Rating_Wavg	UsefulCount
0	transient ischemic attack	Aspirin	"No side effects, easy to take, no more sympt...	Positive	10.000000	10
1	transient ischemic attack	Bayer Children's Aspirin	"No side effects, easy to take, no more sympt...	Positive	10.000000	10
2	transient ischemic attack	Clopidogrel	"I've been taking this medicine for a lit...	Negative	6.766667	8
3	transient ischemic attack	Clopidogrel	"I took ibuprofen (2 caps at night for severe ...	Negative	6.766667	13
4	transient ischemic attack	Clopidogrel	"After my VAD Stroke I am on plavix. I have a...	Positive	6.766667	9

Table 9: After weighted average calculation

- This merged dataset now has the disease and drug name along with the review and its sentiment and the average rating with useful count.
- For recommending drugs to the user, we have only considered drugs which have a positive sentiment in their reviews. Hence, filtering out unwanted drugs.
- Also, we have taken a total of all the useful counts present for a particular drug to sort the drug that has to be recommended with the highest useful count first and then the highest average rating.
- For this, we have used the groupby.sum() method as shown below:

```
# taking predicted disease as input and recommending drug based on highest weighted average of ratings
groupedByCount = merged_wavg.groupby(['Disease', 'Drug', 'Rating_Wavg'])['UsefulCount'].sum().reset_index
()
```

- This approach allows us to use all the information present in the dataset according to its importance and hence as an output we get the best possible drug recommendation.

We also tried a probabilistic approach which is explained below. Although, the final recommendation was done by combining a little of both the approaches to get accurate results.

Approach 2: Probabilistic Based Recommendation

Upon obtaining the merged dataset, a drug recommendation system was built with disease as input. Initially the drug with positive reviews were filtered out and sorted based on the useful count to recommend the best possible drug as shown in example below.

```
List of recommened drugs based on useful count and sentiment analysis
```

	Disease	Drug
0	osteoporosis	DENOSUMAB
1	osteoporosis	PROLIA
2	osteoporosis	TERIPARATIDE
3	osteoporosis	FORTEO
4	osteoporosis	ZOLEDRONIC ACID

Tabel 10: List of recommended drugs based on sentiment review and useful count

Later, to rank the recommended drug by weighing in the possible side effects we made use of the side effects dataset mention in the Data Gathering section. Unwanted columns were dropped and the string values were mapped into numerical and the side effects for the earlier recommended drug was obtained along with the problistic value for symptoms based on three different ratings and grouped by based on the weights and finally sorted based on the probability. So the final recommended drug consisted of the possible side effects and the probabilistic score associated with it.

Algorithm: Recommendation System - Probabilistic Score

Input
X: Disease
Output

Drugs, Prob. of side effects, Disease and Side Effects
steps:

1. select the rows in the data for the input disease
2. select the disease rows which have only positive reviews
3. sort the values based on the useful count
4. drop duplicates and select top 5
5. encode the ratings on side effects dataset
6. define probabilistic score - based on effectiveness rating, overall rating and side effects rating
7. create the list of drugs recommended
8. for items in drug list
    a. map the corresponding side effect with probabilistic score
    b. concatenate to new dataframe
9. sort the new dataset based on the prob. score

Results

Disease Prediction

Approach 1

For disease prediction using method 1, we have used four models namely, Multinomial Naive Bayes, Extra Tree Classifier, Decision Tree Classifier, and Support Vector Machine. Multinomial Naive Bayes, Extra Tree Classifier, Decision Tree Classifier produces 89.93% accuracy and SVM produces 87.58% accuracy. Disease predictions are wrong as there was insufficient data for the model to learn.

Approach 2

For disease prediction using method 2, we have used four models namely, Multinomial Naive Bayes, Extra Tree Classifier, Decision Tree Classifier, and Support Vector Machine. All the four classifiers produced 86.84% accuracy rate. The disease predictions were incorrect for one symptom. For two and three symptoms, Multinomial Naive Bayes produced the correction prediction.

Approach 3

For disease prediction using method 3, we have used four models namely, Multinomial Naive Bayes, Extra Tree Classifier, Decision Tree Classifier, and Support Vector Machine. The Multinomial Naive Bayes provides 86.90% accuracy. Extra Tree Classifier provides 88.18% accuracy. Decision tree classifier provides 86.95% accuracy for a max depth of 120. And Support Vector Machine classifier provides 86.96%. The disease predictions were incorrect for one and two symptoms. For three symptoms, Multinomial Naive Bayes produced the correction prediction. And for four symptoms, Multinomial Naive Bayes, Extra Tree Classifier, and Support Vector Machine predicted the right disease.

Approach 4

We have done prediction on all 4 models using this approach too and the accuracy achieved is Decision Tree: 88.24%, Multinomial NB: 87.98%, Random Forest: 81.88%, Gaussian NB: 88.34% and SVM: 88.61%. All the classifiers have performed well and the disease predicted after taking 3 symptoms as input features is also correct. We tried predicting the disease by taking symptoms from multiple diseases and the model predicted it accurately 9/10 times.As the results obtained from this approach are quite satisfactory and the computation time is also less, we have used this approach for our final disease prediction.

Comparison of Performance values

Models	Approach 1	Approach 2	Approach 3	Approach 4
Multinomial Naive Bayes Classifier	89.93%	86.84%	86.90%	87.98%
Extra Tree Classifier	89.93%	86.84%	88.18%	-
Decision Tree Classifier	89.93%	86.84%	86.95%	88.24%
Support Vector Machine	89.93%	86.84%	86.99%	88.49%

Table 8: Accuracy values of each approach.

Sentiment Analysis

Approach 1:

For evaluating the NLP model we made use of loss, accuracy, f1 score, recall and precision metrics and the following results were obtained.

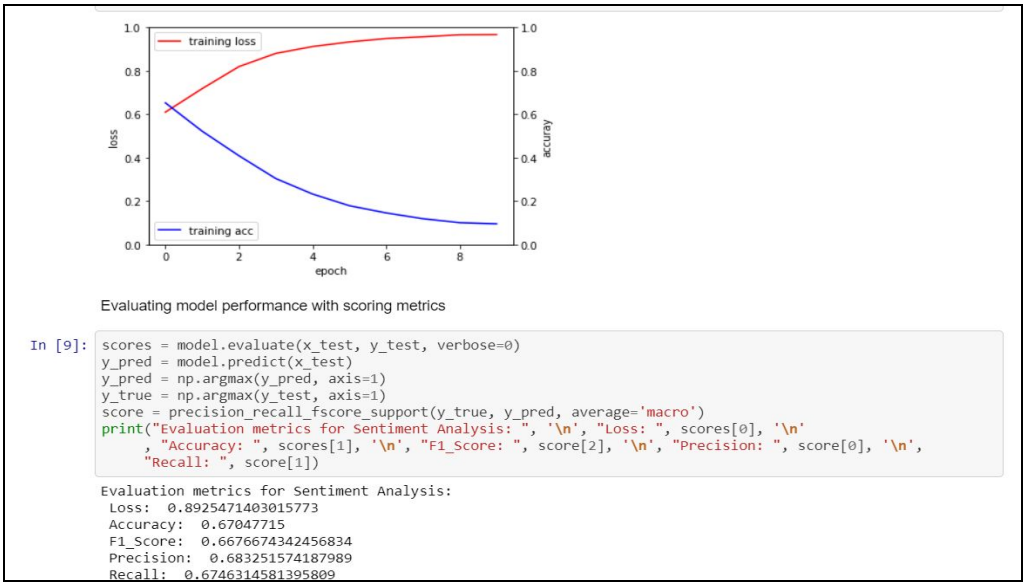

Figure 12: Evaluation metrics for sentiment analysis

It can be noted that the training accuracy reached over 90% but not CV accuracy one reason could be due to inaccurate labelling of actual labels. Below are the few predicted sentiments.

```
Some of the reviews and sentiments using d2v embedding


                                                reviews actual label  \
0  "I am taking Diamox this is the 3rd time I hav...     positive
1                                "Works good for me."     negative
2  "I&#039;ve had glaucoma since I was 9, and I&#...     positive
3  "I have Iritis, so to get rid of the inflamati...     negative
4  "It helped lower my pressure but made my eyes ...     negative

  predicted label
0        negative
1        positive
2        positive
3        negative
4        negative
```

Figure 13: Predicted labels of reviews

In the above figure it can be observed that in review at index 1 says “Works good for me” has a negative review but the predicted label classifies it accurately as a positive review hence giving it a strong reason to use NLP based predicted values than setting threshold for ratings.

Approach 2:

To get the sentiment analysis of each drug review, as a second approach we have used vader analysis. The output given by vader consists of 4 components namely, positive, negative, neutral and compound. After doing the vader analysis of every review, below we have shown results for every review in the dataset.

	Review	Positive	Negative	Neutral	Compound
0	"No side effects, easy to take, no more sympt...	0.206	0.638	0.156	0.1779
1	"I've been taking this medicine for a lit...	0.098	0.762	0.140	-0.4092
2	"I took ibuprofen (2 caps at night for severe ...	0.000	0.721	0.279	-0.8176
3	"After my VAD Stroke I am on plavix. I have a...	0.055	0.945	0.000	0.6757
4	"No side effects, easy to take, no more sympt...	0.206	0.638	0.156	0.1779

Figure 14: Vader Sentiment analysis for all reviews

To classify the overall sentiment of every review, we used the threshold specified earlier. After assigning each review it’s overall sentiment, we can see distribution of positive, negative and neutral reviews in the dataset in the below diagram.

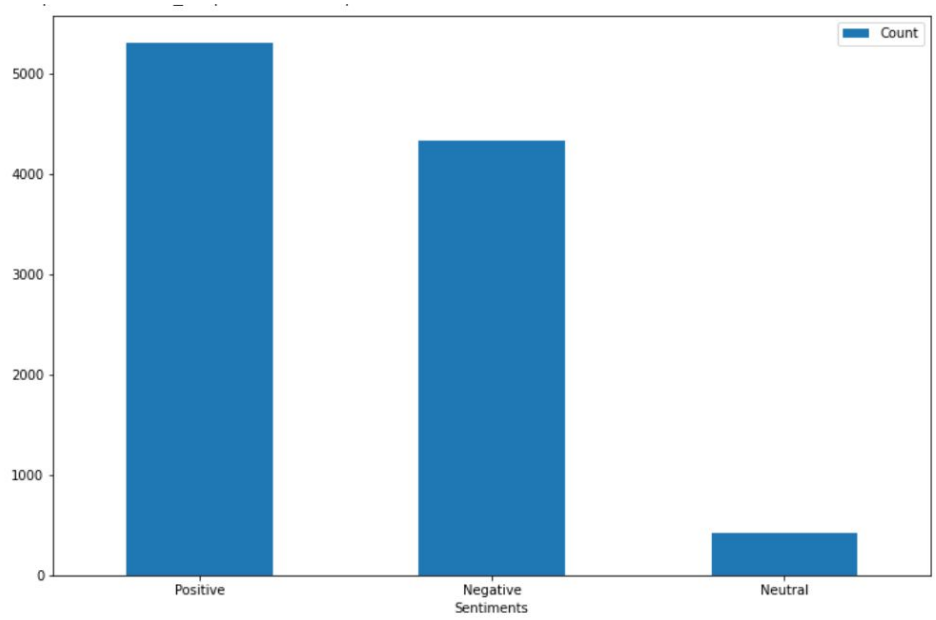

Figure 15: Graph showing sentiment of reviews in the dataset

Drug Recommendation

As discussed above, we have experimented with two approaches for the drug recommendation. But the final recommendation was done using the Weighted Average approach along with the useful count factor from the probabilistic approach.

Few things considered while recommending the drug:

- There are multiple drugs for a single disease. Hence, by using sentiment analysis we were able to filter out the negative and neutral reviews leaving us with only positive ones.

Approach 1: Weighted Average Approach

The results obtained from this approach gave us the correct and highly rated drug for the predicted disease. The disease that we predicted is osteoporosis based on the given and we are recommending top 3 drugs from our dataset as an output. It can be seen below:

```
print("Recommended drugs for this disease are:\n ", predicted_drug["Drug"].unique())
```

```
Recommended drugs for this disease are:
  ['Calcium / vitamin d' 'Caltrate 600+D' 'Calci-Chew']
```

	Disease	Drug	Rating_Wavg	UsefulCount
246	osteoporosis	Calcium / vitamin d	9.0	15
248	osteoporosis	Caltrate 600+D	9.0	15
245	osteoporosis	Calci-Chew	8.0	5

Table 11: Output of weighted average approach for drug recommendation.

- **Verification of results with real world:**

After doing a google search on the treatment of osteoporosis, we found that it requires Vitamins and bone health which is accurately recommended in our 1st drug.

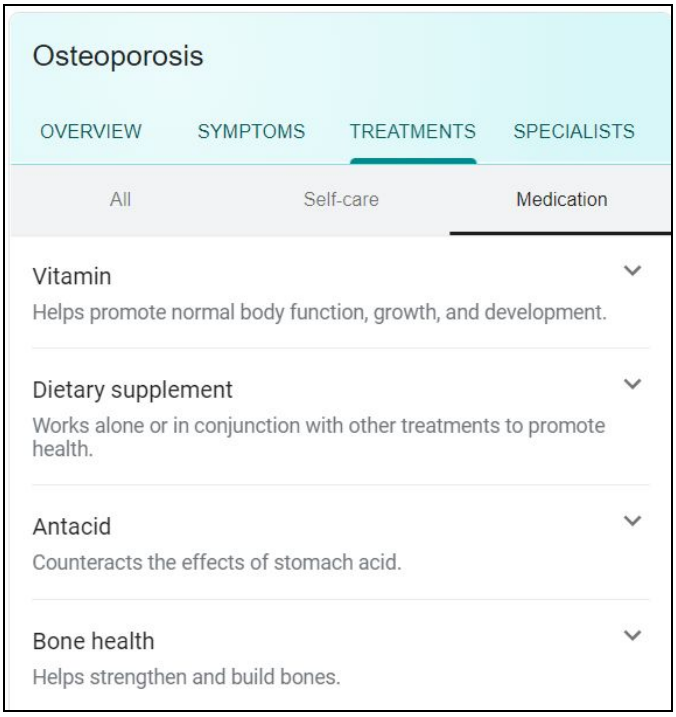

Figure 16: Screenshot of the treatment of osteoporosis

Hence, we can say that our drug recommendation system works as expected and gives the desired results!

- This output is then combined with the side effects dataset to get side effects of the recommended drug if any. This can be seen in the final drug recommendation code:

The recommended Drugs for the given Disease is:

	Drug	Disease	Rating_Wavg	UsefulCount	Prob. of Side Effect	Side Effects
0	GALANTAMINE	alzheimer's disease	8.510638	71	0.1	Chest pain or discomfort, lightheadedness, diz...
2	RIVASTIGMINE	alzheimer's disease	7.481108	95	0.4	Diarrhea, indigestion, loss of appetite, loss ...
1	EXELON	alzheimer's disease	7.526027	80	0.9	an ulcer or stomach bleeding, a seizure, heart...

Figure 17: Recommended drug for Alzheimer's disease with it's possible side effects.

Approach 2: Probabilistic Approach

The results and evaluation obtained from this approach is quite satisfactory. However there is no definite dataset to compare the recommended results as here is the list of recommended drugs along with possible side effects for osteoporosis ranked based on probabilistic score.

The recommended Drugs for the given Disease is:

	Drug	Disease	Prob. of Side Effect	Side Effects
1	PROLIA	osteoporosis	0.1	Back pain, blistering, crusting, cracked, dry ...
0	DENOSUMAB	osteoporosis	0.4	Back pain, blistering, crusting, cracked, dry ...
2	TERIPARATIDE	osteoporosis	0.8	abdominal pain, confusion, constipation, depre...
3	FORTEO	osteoporosis	0.8	abdominal pain, confusion, constipation, depre...
4	ZOLEDRONIC ACID	osteoporosis	0.8	Agitation, blurred vision, cough, depression, ...

Note: The results obtained from this approach can be compared with a list of prescribed drugs for osteoporosis here.

Final Results:

We have combined all the functionalities implemented in this project with the most accurate approaches in one single file. This file takes symptoms as an input and predicts the disease. Which is then passed as an input to the drug recommender from where the recommended drug is taken and its side effects are extracted and given as the output.

Demonstration

Using the ChatterBot, chatbotAI, chatterbot-corpus, flash ngrock library of python we have implemented a basic chat bot which describes the flow and results of our project in a better way. A chatbot a.k.a. chatterbot, is like a computer program or essentially an AI which can conduct a conversation via text based methods. They often simulate human dialogues and conversations. Popular real life examples include SIRI, Cortana. There are various templates freely available for chatbot and one of them has been used here.

- For designing the UI, HTML, CSS and JavaScript technologies are used.
- Using Ngrock, which allows us to host web pages on localhost, we have hosted this chatbot.
- Below is the screenshot for the chatbot UI. The chatbot UI is very basic and primitive. When executing the ChatBot_For_Demonstration.ipynb file, in the last cell of the notebook file, the output will look something like this:

Figure 18: Output for hosting chatbot UI.

To execute this chatbot on your system, there are two techniques:

1. Using JupyterNotebook on a local machine.
   If using this option, just click on the first link shown in the diagram above.

2. Using Google Colab.
   If using Google Colab, or any other medium, use the second link (ngrock.io) as shown above.

However, irrespective of the platform, the ngrock.io link will always work.

After the chatbot has been executed, use the below steps and text for correct output.

1. Firstly, enter this message: “I am suffering from a cough,decreased body weight,dyspnea.”
2. After getting response from chatbot, enter this message: “What medicines should i take for bronchitis”
3. Lastly, after getting the recommended drug, enter this message “Are there any side effects for it ?”.

- Below shown screenshot shows these steps and the output of chatbot:

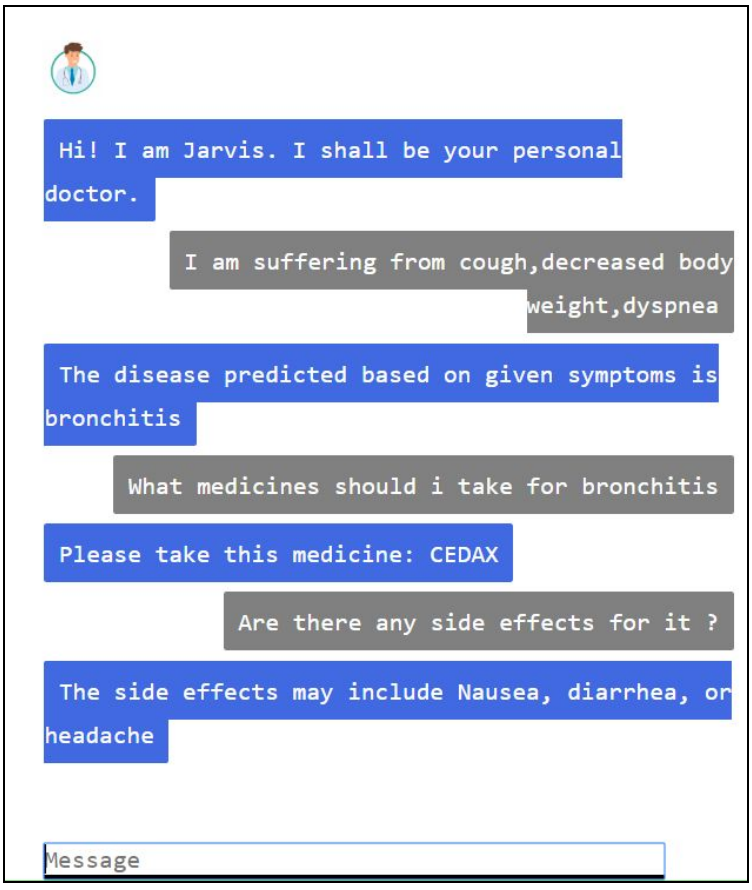

Figure 19: Drug recommendations using a chatbot based on user inputs.

As mentioned earlier, this chatbot is very primitive and only programmed for a fixed set of commands. The only purpose it serves is that it provides the flow of our project. The template for this UI is available freely for use.

Conclusion and Future Scope

Drug Recommendation systems are a prevailing technology in today's online services and with the rise of requirements of these services there is more and more need to automate the processes subsequently we have designed an drug recommendation system which is deployed to a chatbot UI. Below are the key conclusions from our project.

- Successfully built a drug recommendation system and a chat bot that predicts diseases and recommends drugs along with possible side effects based on user symptoms input
- We designed three models for this project implementation. A disease prediction model, Sentiment Analysis model and a recommendation model.
- Experimented with different approaches for each of three models
- Each of the three models gave good accuracies contributing to the overall reliability of the drug recommendation model.
- Demonstration of the project is done using the ChaBot as a UI between the code that we've written and its results for better understanding.

One key future scope can definitely be improving the accuracies of the prediction and recommender model using deep neural networks by using larger data. Also with regard to chatbot UI it can integrate to websites that cater towards online medical services.

References

[1] Kamaraj, K.Gomathi & Priyaa, D.Shanmuga. (2016). Multi Disease Prediction using Data Mining Techniques. International Journal of System and Software Engineering.

[2] M.A.Nishara Banu, B Gomathy. (2013). Disease Predicting System Using Data Mining Techniques. International Journal of Technical Research and Applications.

[3] H Wang Q Gu J Wei Z Cao Q Liu (2015). Mining drug–disease relationships as a complement to medical genetics-based drug repositioning: Where a recommendation system meets genome-wide association studies.

[4] Yin Zhang, Daqiang Zhang, Mohammad Mehedi Hassan, Atif Alamri & Limei Peng (2014). CADRE: Cloud-Assisted Drug REcommendation Service for Online Pharmacies.

[5] Youjun Bao ; Xiaohong Jiang. (2016). An intelligent medicine recommender system framework.

[6] Druglib.com - Drug Information, Research, Clinical Trials, News. http://www.druglib.com/

[7] Hutto, C.J. & Gilbert, Eric. (2015). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014.