**Methodology:**

The methodology of this project is separated into several parts.

**Acquiring the dataset:**

Acquiring the dataset was made possible by creating a data crawler using Java JSoap library using the following steps.

* Choosing the Drugs for this research: For the purpose of this research, drugs using for chronic diseases had to be chosen. After some consideration, drugs used for hypertension were chosen, and they are Lisinopril, Nadolol, Amlodipine, Diltiazem, Hydrochlorothiazide and Atenolol.
* Choosing the medical forums for mining: The websites chosen for this purpose are MedHelp and AskAPatient, which was chosen since most of its posting members are more committed to share their personal data, such as age and gender. The dataset acquired from both forums will be used comparing results and quality of the datasets.
* Finding the correct links and tags: JSoup establishes a connection to the server and returns the HTML script as text to a variable, and from that script tags can be chosen based on IDs or classes, given a universal search query link (example: <https://www.medhelp.org/search/expanded?cat=posts&page=2&query=Nadolol>), the web can easily be navigated through JSoup, and given the correct tags from each the given posts (example: subject\_msg), data can extracted from each page and it’s HTML script.
* Store Data: The chosen data storage is on CSV file which can be accessed using MS excel, they can also be used later using Pandas library in Python, and later saved as excel files (which proved even more convenient than CSV files).

**Data retrieval:**

Several natural language processing techniques were implemented using Python to extract the data necessary, using NLTK (Natural Language Tool Kit) library

* Tokenize data: Turn the words into separate tokens.
* Remove stop-words: Stop-words like (and, a, or) were removed to decrease the size of data.
* Stemming data: Porter stemmer was used to turn words into their roots (exhaustion, exhaustive, exhausted= exhaust), both the original and the stemmed tokens were kept into separate csv columns.

**Build Dictionary:**

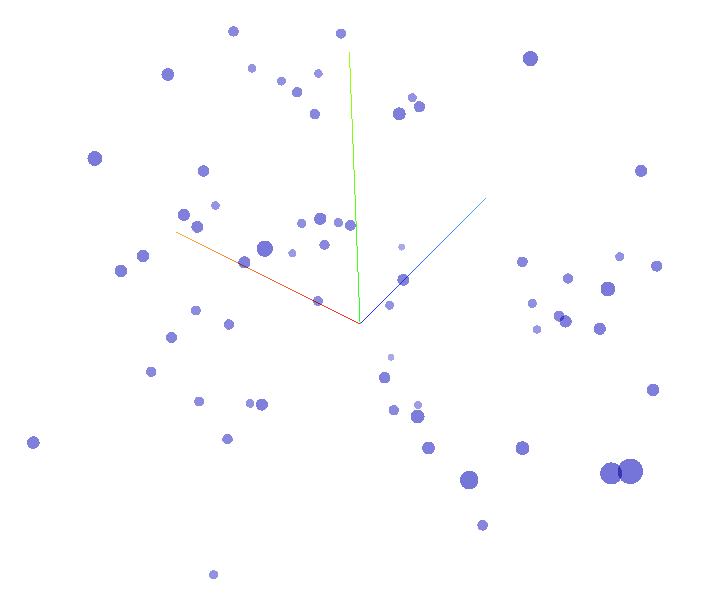
A dictionary filled with concepts like ADR, Disease and Mental issues were needed to narrow down the search premise into the UMLS.

* Find term frequency: This in itself is not necessary for building a dictionary, but the TfidfVectorizer from Sklearn library can double as a retrieving method for all unique words in the text, term frequency will be used later, but for now all unique words are stored in two term frequency files, one for stemmed words and one without stemming.
* Send the files to MetaMap: MetaMap as mentioned before is used to extract UMLS concepts, by sending both files, we can be certain that every single concepts mentioned in the dataset will be tested.
* Extract the concepts from MetaMap: MetaMap output needed is classified into three categories ([Signs and Symptoms] = ADR, [Disease or Syndrome] , and [Mental or Behavioral Dysfunction]), a script was made to handle MetaMap output and extract these concepts into files containing the concept in Python Dictionary Format.
* Extract The concepts per post: The dictionary was copied and pasted into a python script (After some minimal manual revision) to be used for concept extraction from the dataset, every post had was scanned for any token that match any concept in the dictionary, and these concepts were than aligned it’s related meaning ('cramp': ['Muscle Cramp ', 'Cramping sensation quality ']) or meanings, this arrangement is made so that concepts with similar meanings don’t get repeated. Age was also extracted by applying a moving window of three tokens that searches for some limited age related words couple with numbers (I am 28 years old). As well as tension (150/70).

The size of the dictionary was further limited by the use of stemmed words which limited similar words (confusion, confused = confus), however it also removed the meaning from certain words and was therefore undiscovered by MetaMap (“Acne” became “acn” which means nothing and was therefore undiscovered), so a combination of stemmed and un-stemmed dataset were used.

**Association with GloVe:**

GloVe: Global Vectors for Word Representation [1], is a very popular library created by Stanford to be used for word embedding. It is used here to make some indications which can be observed later in the machine learning phase. Word embedding shows the association between all the words in the corpus.



* Both stemmed and un-stemmed datasets were used to create a word vector models, the effectiveness of either can be tested later on.
* After the models were saved into several formats, they were trimmed to only include ADRs and Mental issues, as they are the most relevant.
* The reduced models were saved as 2D array for vectors in .txt format.
* The text file was loaded into tensorflow projector [2] [3]to represent the points of the model into an intelligible 3D (in truth 100D) plot (pictured above).
* This plot allows the observation of related ADRs and Mental issues, each dot representing one of the model’s label. The closer two dots are to each other, the higher their association, and therefore the probability of co-occurrence.
* By looking at the closest vectors to a certain drug, it was possible to find which ADRs and Mental issues have the highest chance to occur when using the drug. For example, amlodipine was found more related to hoarseness, frenzy and hallucinations than it is related to pain, nervousness and alcohol abuse. It is therefore more expected for a patient to encounter hoarseness- for example- than to encounter pain while taking the drug.

**Machine Learning:**

**Preparing the dataset:** After building the dictionary, it is now possible to identify the diseases, ADRs, and mental issues that were mentioned in the user posts.

* **Pandas** library was used to access and manipulate the data.
* A scanner was made to iterate on every token in every record in the stemmed version of the dataset, matching each token with an equivalent in the dictionary.
* When a token matches the dictionary, it marks the meaning of the concept as existing if another token with same meaning, such as (‘ache’: ‘pain’, ‘pain’: ‘pain’), it is ignored to limit repetition.
* The number of concepts (diseases, ADRs and mental issues), were counted for each record.
* Two types of classes are created using this process.
  1. Concept exists, where for each concept, a Boolean value is given to determine the existence of the concept in the record.
  2. Concept count, where the number of specific concepts in a particular range is drawn, meaning the number of distinct ADRs in a record for example is 5, this number is recorded and then assorted in the following ranges as a class.
     + 0 for 0 concept.
     + 1 for range [1,3].
     + 2 for anything more than 3.

In the previous example, 5 will be in the 2 class.

* As for the features, user information harvested is used, such as age, gender, blood pressure, weight and height. Unfortunately, the MedHelp dataset has a lot of missing data, so not all user information could be used at the same time.
* To solve this issue, the dataset was divided into three groups.
  1. Age + Gender only
  2. Age + Gender + Weight + Height
  3. Age + Gender + Blood Pressure
* Now that missing data is in an acceptable level, the remaining missing values (Age and Gender) were imputed using the SciKit learn library, SimpleImputer. Gender was imputed based on most frequent strategy, while age was imputed based on median strategy.
* The data sizes for the three datasets respectively is: 1557, 130, 462. With dataset 2 ⊆ dataset 1 and dataset 3 ⊆ dataset 1.
* The classes labels were stored in separate excel files, from which they can be extracted later and used in the classifier.
* The same procedure was applied for AskAPatient dataset, however the dataset did not include anything other than age and gender, therefore as a whole the size of the dataset is 757 with no divided parts between them

**Preparing the classifiers:**