**Drug side-effects detection for PubMed data**

KAVYA mANDLI, 800984226, University of North Carolina at Charlotte

1. **Introduction**

This document is an overview of the project that deals with digging for drug/substance side effects or adverse effects from the PubMed Dataset. PubMed comprises of more than 28 million citations for biomedical literature from MEDLINE, life science journals, and online books. Citations may include links to full-text content from PubMed Central and publisher web sites. The sample file from this dataset has been used for the searching and extraction of the list of drugs and their major side effects. This project requires a thorough understanding of data, its structure and the ways to extract and parse it. Hence, We will first dig into the structure of the data, its complexity and ways to acquire them from the XML file.

1. **Structure of the Data**

With the public having access to create and maintain this kind of open platform, the founders have made sure that they provide an easy access to their data dumps. The MEDLINE PubMed data can be accessed from the following link : <https://www.nlm.nih.gov/databases/download/pubmed_medline.html>. In the above mentioned link, PubMed also has APIs that allow access to their content using Software programs. It is important to know and understand all the PubMed XML elements to further be able to extract the required information. The structure goes as follows in order to get to the drug list:

* 1. **<Title>**

The full journal title taken from NLM cataloging data following NLM rules for how to compile a serial name is exported in this element. Some characters that are not part of the NLM [MEDLINE/PubMed Character Set](https://www.nlm.nih.gov/databases/dtd/medline_characters.html) reside in a relatively small number of full journal titles.

* 1. **<PubmedArticle>**  
     <PubmedArticle> is one of the top document level elements in the PubmedArticleSet (PubmedBookArticle and DeleteCitation being the other two) and contains one entire record. <PubMedArticle> is an envelop element that contains the <MedlineCitation> and <PubmedData> elements.
  2. **<MedLineCitation>**

The party responsible for creating and validating the citation is recorded as the MedlineCitation Owner attribute. Each citation has only one MedlineCitation Owner and there are eight possible values for this attribute.

* 1. **<ChemicalList>**

This element contains one or more **<Chemical>** elements that, in turn, contain <RegistryNumber> and **<NameOfSubstance>**. **<ChemicalList>** is always complete. <NameOfSubstance> is the tag which is needed for the drug name extraction.

* 1. **Tree Structure**

<PubMedArticleSet >

<PubMedArticle>

<MedlineCitation>

<ChemicalList>

<Chemical>

<NameofSubstance>

1. **Implemented Methods:**

**3.1 XML Parser**

An XML parser/scrapper has been implemented to crawl through the above hierarchy and fetch all the drugs and names list from the tag <NameOfSubstance>. Total number of drugs extracted from the XML sample file are 342.

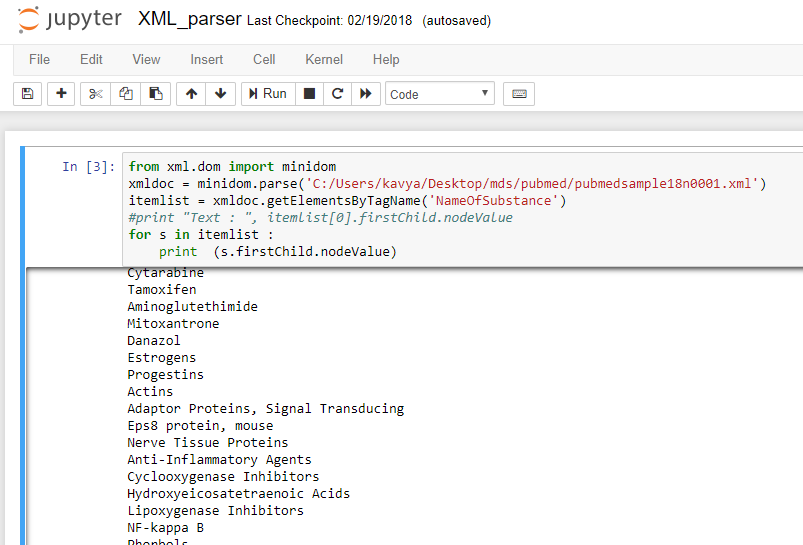
****

Fig 1 : Screenshot of the XML parser to extract the “list of substances”.

**3.2. Related articles extraction**

Now, that we have the list of drugs from the source dataset, it is important to scrap all the information related to these drugs from the internet. The major source of this is Wikimedia dumps. As mentioned before, Data for Wikipedia can be found in different versions and formats using the link;  [https://dumps.wikimedia.org/enwiki/20171001](https://dumps.wikimedia.org/enwiki/20171001/). This link has data related to all the XML and related dumps of the data. This link has data, metadata, indexes, geotags, SQL files and much more useful dumps. The articles are present in different versions and different formats. Given the size and complexity of Wikipedia dumps as mentioned above, I adopted an approach of finding a comprehensive list of drug articles from Wikipedia. The link: <https://en.wikipedia.org/wiki/Special:Export>, extracts the Wiki articles of interest in its native XML format based on the query used for search. The query “Lists of Drugs” lists all categories of drugs and the export button hence provides the XML dump of all the matched articles. This XML dump has been used as the source of “Medical Related Articles” for side effects extraction.

**3.3 Regex Parser**

Some of the tasks that need to be accomplished in this are finding all the terms that are related to medical field. This in turn includes scraping all the lists, using medical related websites or even using drugs.com. The regex parser starts off my searching for the title page or the title of the drug/substance exists in the XML dump or not. If the corresponding XML article or page exists, then all the content from the page is grabbed for further scraping.

The structure of the parser is a single python program that would iterate over each regex matched side-effects section and pull the text under that section. The regex is intelligently modified to make sure if a section or a subsection is selected, entire text is extracted within its respective block, i.e., until the next section is encountered. The input file for this regex parser would be the XML file download from the web containing all the related articles for the drugs. Furthermore, all the data that has been enclosed within [[ ]] tags and sub sections | [[]].

All the data/lines around the following keywords has been pulled out as they’re the effective could-be sections containing the side-effects. The list of words are as below:

* Effects
* Adverse effects
* Side Effects
* Hazards
* Causes
* Increases
* Decreases

**3.3.1 What the Parser can do**

Described below are some of the additional functions performed by the parser apart from the ones mentioned above.

1. The function of the regex would be to match specific sections in the bz2 file OR the extracted medical article XML file.
2. For an article, if there’s no “effects” section or a subsection, 15 lines after or before keywords such as “effects” or “increases” or “decreases” or “causes” are extracted.
3. Each extracted section is placed under it’s article’s title.
4. For large text files, the regex will support working in chunks, taking a performance effective route rather than loading up all the text at once.

**3.3.2 Advantages:**

* Very easy to implement
* High Precision

**3.3.3 Disadvantages:**

* List will be very big
* Just doing Regex would be very expensive operation
* An index is available for all the wikipedia articles which can be used to make this quicker. It contains exact byte position of where that article is within the dump
* Recall may suffer on how extensive the list is
* Circular-lists formation.

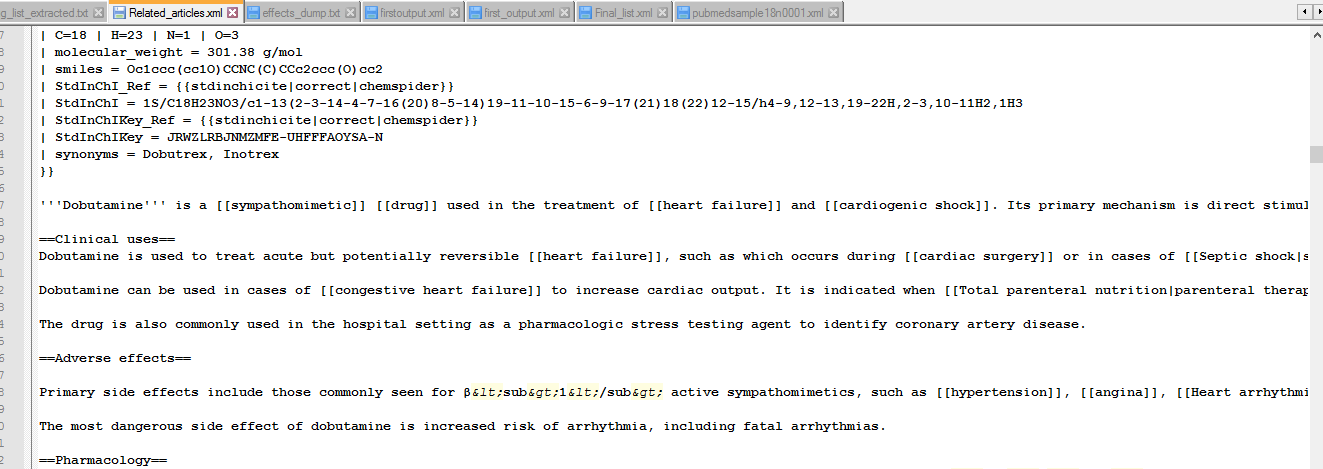
****

Fig 2 : Regex Parser extracting the related sections and sub-sections around the topic-effects

**3.4 Similarity Check for Drug Side-effects detection**

The latest XML dump containing the articles surrounding the words are about 2k articles out of which 200 articles have been eliminated for being too small (since the number of lines in each article are less than 100 ), indicating that the content is too small because they contain the XML tags and other XLST native style content. In order to pull the drug side effects, this parser has been divided to perform two major tasks which are as below:

**3.4.1 Pull hyperlinks in each section**

Most words with hyperlinks, found under a title are related to an adverse effect, so all these are extracted to form a comma separated list of adverse effects. This scraper can pull out all the links between the start page and the end page of Wikipedia. All the hyperlinks, ebedded text and the specific categories between these two points are extracted. This can be used for web scrapping of any related articles, not just medical articles.

**3.4.2 POS tagging (nouns and adjectives )**

This method does the following;

1. Tokenize each sentence under a title
2. Pull all adjectives and nouns
3. Load Wikipedia pretrained GloVe embeddings using Gensim model
4. Compute cosine similarity between each word found in step b with words in the list “nausea, itching, inflammation, diarrhea, glaucoma, constipation, fever and metastasis”. These words in general represent a side effect for a different human body part.
5. Consider the results with a cosine similarity of more than 55%.
6. Append the list of words found in step e to for the side effects of the mentioned article.

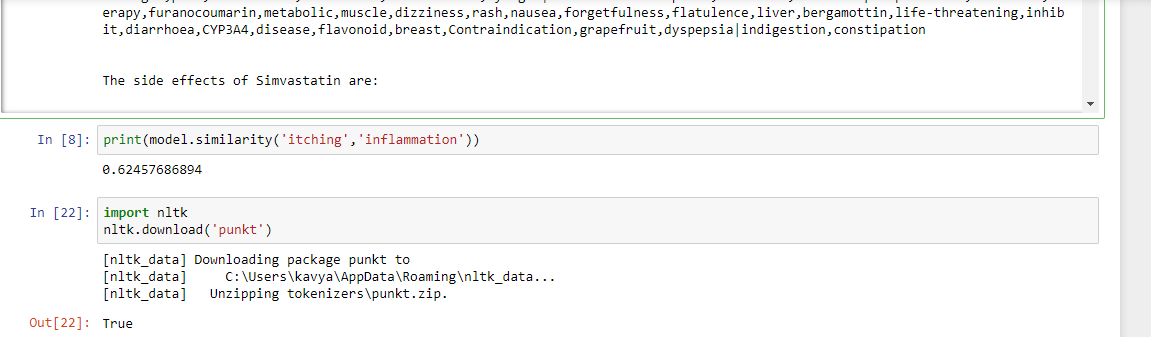


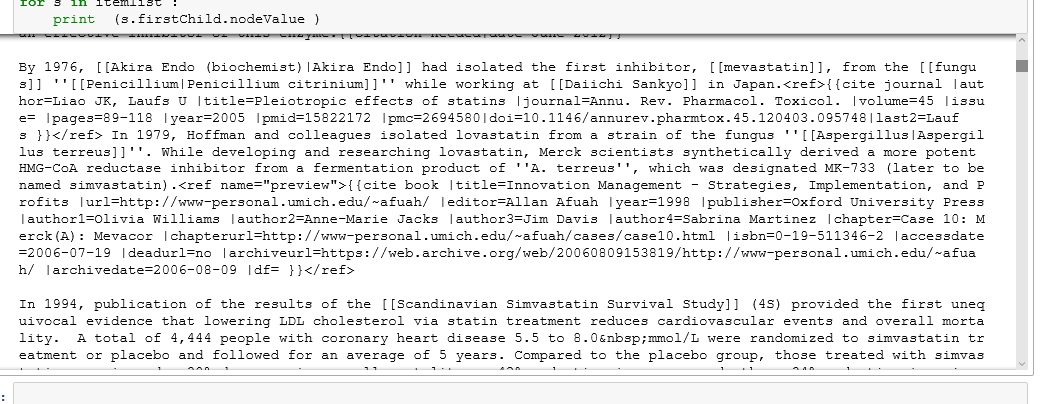
Fig 3 : Screenshot of Drug side-effects extraction and similarity check between words.

**3.5 Graph Method**

The wikipedia category graph can be used to get names of all the pages we need to travel. We will restrict our depth to a fixed value (Explained in earlier). Once the names are collected, we use dumps to extract all the pages. Page extraction can be done using index file already available which would speed-up this process.

* Code to extract names is complete but it cannot be paused and continued. Making code dynamic is needed
* DFS gave a depth of around 50ish but we need to restrict it to an extent
* BFS to level 7 was done but since code does not save the progress.

Using this method, since we already have the list of drugs and list of side-effects extracted already, we can now pull out all the medical related articles or the drug related articles by using both BFS and DFS and the related articles extracted are as shown below:



1. **Other Related approaches**

**4.1 Bottom up approach eliminating the list Information**

This approach is an improvement on Regex. It tries to overcome one of the major disadvantages of Regex: List formation.

Here, the idea is to get initial seed list and then expand this list using the attributes of wikipedia data.

Since, each page in wikipedia belongs to one or more categories. We can use this to find other pages linked to same categories and expand our list. This initial list will help us get more medical terms and thereby expand our list and finally our corpus. The depth of how many pages to categories and categories to pages exploration is subjective and it can be done by some hit and trial.

One of the main reason why this can’t be let to run till it reaches the end is because graph structure of wikipedia is very complex and many pages might be connected to lot of non related pages via these categories. So our precision will suffer.

Getting pages in a category can be done in two ways:

* Finding those category pages in the dump
* Using wikipedia API

Latter is very simple to implement, whereas in first case there will be lot of handling required. But the index file will be very helpful

**4.1.1 Advantages:**

* Easy to implement
* Gives good precision
* List creation is easy

**4.1.2 Disadvantages:**

* Also takes long time
* Doesn’t guarantee exploration wikipedia structure
* Can get quite complex and difficult to understand
* Performance Metrics would be difficult to measure

**4.2 SQL and CSV files**

This method is very similar to the approach that has been used for the extraction but only difference is e use sql files provided for categories instead of using the XML dump category. In dumps, there are 2 main sql files: one which has category information and other has relations between categories. These two files can be used to form a query and collect all the pages we need.

**4.2.1 Advantages:**

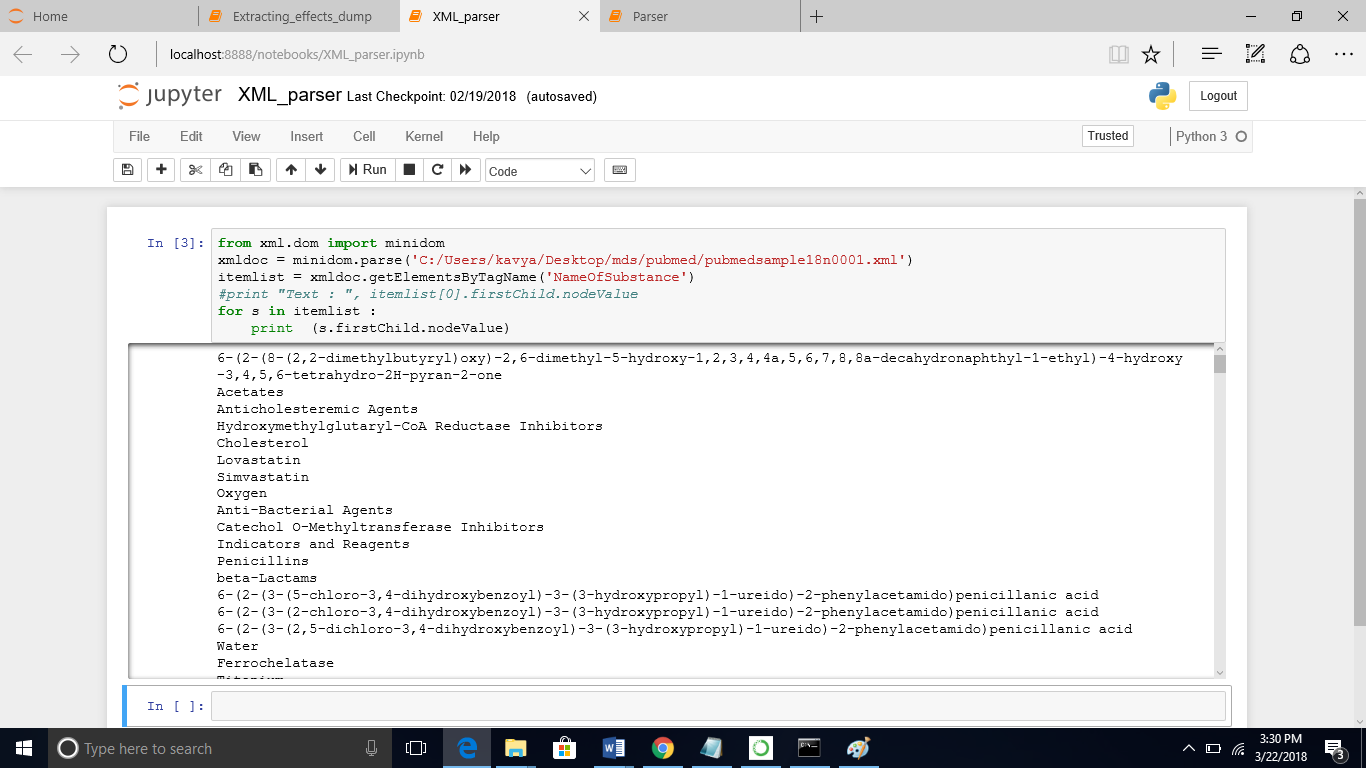
* Very less work
* Intuitive
* Very fast

**4.2.2 Disadvantages:**

* Complex query
* Size of files is huge (Takes lot of time to load in a SQL engine)

1. **Screenshots and results**

**Extraction of List of drugs**



**Sample XML dump extracted from Wikimedia.**

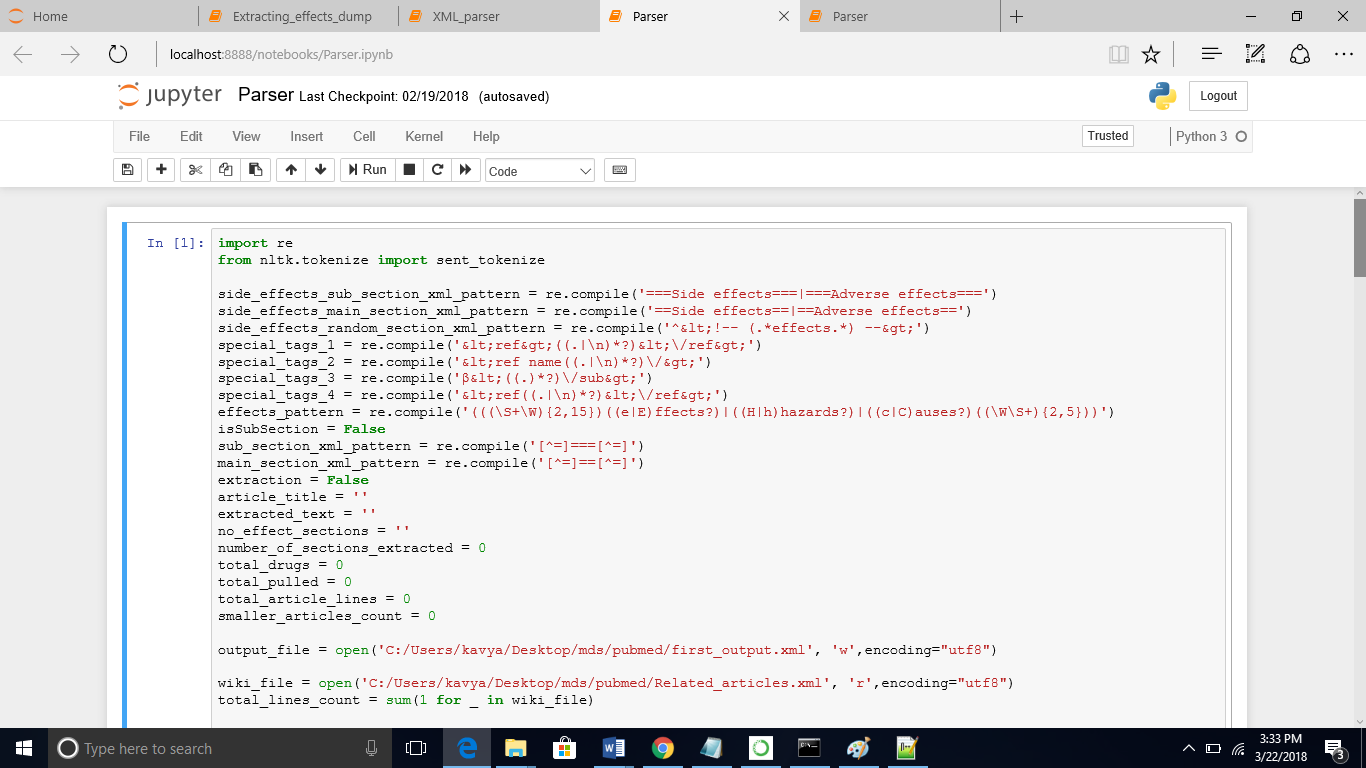
*“==Chemical and physical properties==*

*{{Main article|Properties of water}}*

*{{see also||Water (data page)|Water model}}*

*Water ({{chem|[[hydrogen|H]]|2|[[oxygen|O]]}}) is a [[Chemical polarity|polar]] [[inorganic compound]] that is at [[room temperature]] a [[taste]]less and [[odor]]less liquid, nearly [[Transparency and translucency|colorless]] with a [[Color of water|hint of blue]].<!--please read the article before considering removing it.--> This simplest [[hydrogen chalcogenide]] is by far the most studied chemical compound and is described as the "universal solvent" for its ability to dissolve many substances.<ref>{{Greenwood&Earnshaw2nd|page=620}}</ref><ref>{{cite web |title=Water, the Universal Solvent |url=http://water.usgs.gov/edu/solvent.html |work=[[USGS]]}}</ref> This allows it to be the "[[solvent]] of life".<ref>{{Cite book |title=Campbell Biology |last=Reece |first=Jane B. |date=31 October 2013 |publisher=[[Pearson Education|Pearson]] |year= |isbn=9780321775658 |edition=10 |location=|publication-date= |page=48 |pages= |via= |author 2=}}</ref> It is the only common substance to exist as a [[ice|solid]], liquid, and [[water vapor|gas]] in normal terrestrial conditions.<ref>{{Cite book |title=Campbell Biology |last=Reece |first=Jane B. |date=31 October 2013 |publisher=[[Pearson Education|Pearson]] |year= |isbn=9780321775658 |edition=10 |location=|publication-date= |page=44 |pages= |via= |author 2=}}</ref>”*

**Regex Parser**



**Sample articles extracted:**

*“Common side effects (&gt;1% incidence) may include indigestion and eczema. Rare side effects include joint pain, memory loss, and muscle cramps. Cholestatic hepatitis, hepatic cirrhosis, rhabdomyolysis (destruction of muscles and blockade of renal system), and myositis have been reported in patients receiving the drug chronically. Serious allergic reactions to simvastatin are rare.*

*A type of [[DNA]] variant known as a [[single nucleotide polymorphism]] (SNP) may help predict individuals prone to developing [[myopathy]] when taking simvastatin; a study ultimately including 32,000 patients concluded the carriers of one or two risk [[allele]]s of a particular SNP, rs4149056, were at a five-fold or 16-fold increased risk, respectively. In 2012, the Clinical Pharmacogenetics Implementation Consortium has released guidelines regarding the use of rs4149056 genotype in guiding dosing of simvastatin and updated the guideline in 2014.*

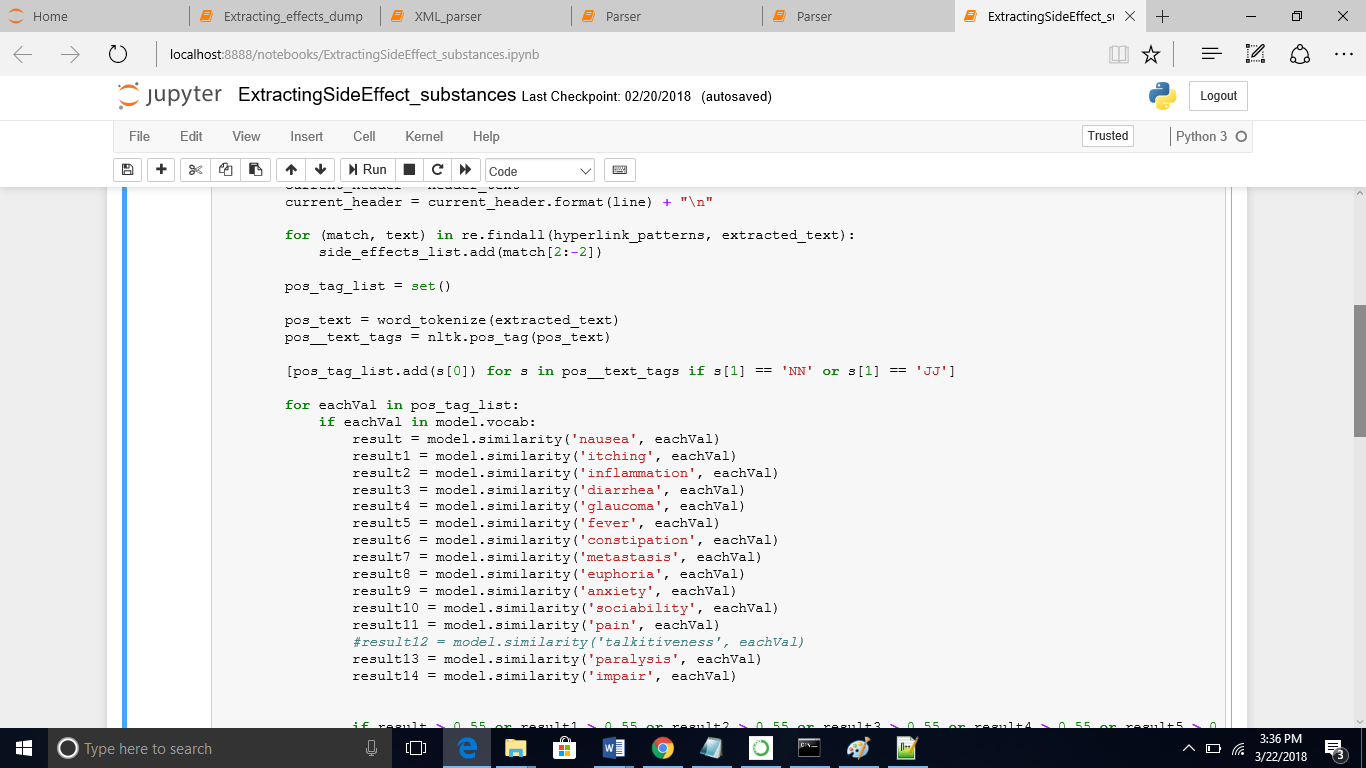
*In March 2012, the FDA updated its guidance for statin users to address reports of memory loss, liver damage, increased blood sugar, development of [[type 2 diabetes]], and muscle injury. The new guidance indicates:*

*\* FDA has found that liver injury associated with statin use is rare but can occur.*

*\* The reports about memory loss, forgetfulness, and confusion span all statin products and all age groups. The FDA says these experiences are rare but that those affected often report feeling “fuzzy” or unfocused in their thinking.*

*\* A small increased risk of raised blood sugar levels and the development of type 2 diabetes have been reported with the use of statins.”*

**Extracting hyperlinks and final List**



**Sample list of drug Side effects detected extracted**

*“The side effects of Lovastatin are:*

*depression,intracellular,atherosclerosis,inflammatory,mortality,heart,epidermal,ingestion,receptor,atheroma,cerebral,cause,cardiovascular,hemorrhage,depression (mood)|depression,lipoprotein,severe,sterol regulatory element*

*na ot sdael esatcuder AoC-GMH saerehw ,maertsdoolb eht morf LDL gnitalucric segnevacs rotpecer LDL ehT .,cerebral hemorrhage,blood,disease,endoplasmic reticulum,incidence,chronic,cardiovascular disease,cancer,coronary,coronary heart disease,protein*

*The side effects of Simvastatin are:*

*naringin,pain,simvastatin,metabolism,atorvastatin,myalgia|muscle aches or pains,abdominal,creatine phosphokinase,treatment,therapy,furanocoumarin,metabolic,muscle,dizziness,rash,nausea,forgetfulness,flatulence,liver,bergamottin,life-threatening,inhibit,diarrhoea,CYP3A4,disease,flavonoid,breast,Contraindication,grapefruit,dyspepsia|indigestion,constipation”*

1. **Applications**

The following are some of the application of this project

* Extraction of all the medical related data will help in the detection of the adverse effects/side effects.
* It further helps in detecting major drug-drug interactions.
* It can be used to send and receive reports and orders for other medical related purposes.
* Research of new drugs.

1. **Challenges faced**

* Structure of the data (All text Represented as XML with native Wiki-XSLT)
* Cross listing of an article within other related/unrelated articles
* Choosing the best algorithm(s)
* Determining metrics (precision and F score)
* Finding relations between articles (links or similarity scores)
* Dealing with the synonyms while finding cosine similarity between words

1. **Conclusion**

This project makes a good effort at projecting an approach at digging into a large corpus in search of finding specific set of documents, and further extract information of interest from them. Generic programs capable of doing this could prove worth a lot of value in many interesting fields of study. This project has been a good case study at extracting a specific bag of words related to a particular topic. Given the fact that the bag of words is extracted from a document set, which is maintained in a specific format made it very challenging. Many topics learnt in the class such as the indexing, tf-idf scores, word embeddings, cosine similarities and contextual phrases and words helped approach this project with ease.