

6-13-2018

Symptom Search: Predicting Symptom and Product Correlations using FDA Adverse Effects Reports

Isabela Figueira

Santa Clara University, ifigueira@scu.edu

Neesha Godbole

Santa Clara University, ngodbole@scu.edu

Angelina Poole

Santa Clara University, apoole@scu.edu

Kelly Wesley

Santa Clara University, kwesley@scu.edu

Follow this and additional works at: https://scholarcommons.scu.edu/cseng_senior



Part of the [Computer Engineering Commons](#)

Recommended Citation

Figueira, Isabela; Godbole, Neesha; Poole, Angelina; and Wesley, Kelly, "Symptom Search: Predicting Symptom and Product Correlations using FDA Adverse Effects Reports" (2018). *Computer Engineering Senior Theses*. 124.

https://scholarcommons.scu.edu/cseng_senior/124

This Thesis is brought to you for free and open access by the Engineering Senior Theses at Scholar Commons. It has been accepted for inclusion in Computer Engineering Senior Theses by an authorized administrator of Scholar Commons. For more information, please contact rsccroggin@scu.edu.

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Date: June 8, 2018

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Isabela Figueira
Neesha Godbole
Angelina Poole
Kelly Wesley

ENTITLED

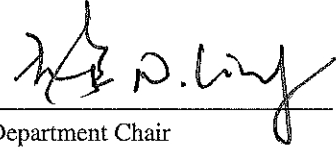
**Symptom Search: Predicting Symptom and Product Correlations using FDA
Adverse Effects Reports**

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING



Thesis Advisor



Department Chair

Symptom Search: Predicting Symptom and Product Correlations using FDA Adverse Effects Reports

by

Isabela Figueira
Neesha Godbole
Angelina Poole
Kelly Wesley

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 13, 2018

Symptom Search: Predicting Symptom and Product Correlations using FDA Adverse Effects Reports

Isabela Figueira
Neesha Godbole
Angelina Poole
Kelly Wesley

Department of Computer Engineering
Santa Clara University
June 13, 2018

ABSTRACT

Prescription and over-the-counter drugs are abundant now more than ever, and many people use them on a regular basis. These drugs come with a variety of side effects, ranging from common to very rare. However, the symptoms listed on drug packaging might not be the only symptoms that a consumer might experience when taking a specific drug. When experiencing a symptom, consumers might think they have a disease, turn to the Web for answers, and attempt to diagnose themselves when a disease is not necessarily the cause. Instead, these symptoms might be reactions to medications; however, the first thought that people have when experiencing a symptom is that they most likely have a disease. This often hinders us from thinking of alternative solutions. In addition, people often take multiple drugs simultaneously, which makes pinpointing the source of a bad reaction or unexpected symptom increasingly difficult. These factors make it challenging to locate the true source of a symptom. Our product, Symptom Search, is a tool that assists in this search for answers. Symptom Search uses FDA Adverse Effects Report data and machine learning to provide users with a method to search for potential root causes of their symptoms. Our system conveys how likely given drugs are correlated with given symptoms, and it suggests other products that could be triggering symptoms or reactions based on other users' interactions with the products.

Table of Contents

1	Introduction	1
2	Requirements	2
2.1	Functional Requirements	2
2.2	Non-Functional Requirements	3
2.3	Design Constraints	3
3	Use Cases	4
3.1	Login/Create user profile	5
3.2	Input symptoms	5
3.3	Input currently used products and drugs	6
3.4	View possible causes of symptoms	6
4	Activity Diagrams	7
5	Architectural Diagram	8
6	Technologies Used	9
7	Description of System Implemented	11
7.1	An Overview of the System	11
7.2	The Dataset	12
7.3	The Results Page: The Ranking Algorithm	12
7.3.1	Table of Likelihoods	12
7.3.2	Suggested Products	13
8	Design Rationale	20
9	Test Plan	21
9.1	Alpha Testing	21
9.2	Beta Testing	21
10	Ethical Analysis	23
10.1	Team and Organization	23
10.1.1	Team	23
10.1.2	Organizational	23
10.2	Social	24
10.3	Product	25
10.4	Science, Technology, and Society	26

11 Lessons Learned	27
11.1 Angelina	27
11.2 Isabela	27
11.3 Kelly	28
11.4 Neesha	28
12 Conclusions	29
12.1 Suggested Additions to the Implemented System and Future Work	29
13 Acknowledgements	31
A User Manual	32
A.1 User Functionality	32
A.1.1 Account Creation and Login	32
A.1.2 Symptom Input and Product Input	32
A.1.3 Results	33

List of Figures

3.1	Use Case Diagram	4
4.1	Activity Diagram	7
5.1	Architectural Diagram	8
7.1	Start Screen	14
7.2	Register or Sign In	15
7.3	Disclaimer With Registration	16
7.4	Input Drugs and Symptoms	17
7.5	Initial Results: A Table of Likelihoods and an Accompanying Key	18
7.6	Suggested Drugs	19

Chapter 1

Introduction

In our lives, we can experience strange symptoms with causes that we cannot pinpoint. For example, we get rashes or experience stomach troubles but are unable to figure out the cause. Since we use a large variety of products in our day to day lives, such as food and drugs, it can be hard to isolate what is causing a reaction or if the reaction is caused by multiple products that are interacting with one another. If there was a way to easily search for connections between symptoms and products we use regularly, we might more easily find the source of our symptoms.

Currently, when we want to discover the cause of our symptoms, we might use Google search or check health related sites such as WebMD or healthline.com. These websites only predict the diseases that could be causing the symptoms but not necessarily the products that could be causing adverse reactions. There is clearly a demand for a way to search for other causes.

Our product, Symptom Search, utilizes FDA Adverse Effects data to make the search process possible. When a user inputs specific drugs and symptoms, the system finds correlations between them based on the number of complaints within the dataset. Symptom Search makes it easy for the user to quickly receive information about what could be the root of their symptoms. Once they have an idea of what might be the problem, they should consult their doctor with this information in mind.

There is a need for a tool that draws connections between the day to day products we use and the symptoms or reactions we might experience that are unlikely to be caused by a disease. Currently, everything is disease-based and there is not an easy way to search for answers. Symptom Search aims to make the search for answers easy and accurate.

Chapter 2

Requirements

This section details our project requirements including functional and non-functional requirements and design constraints. Functional requirements dictate what the system will do. Non-functional requirements dictate how the system will be. Lastly, design constraints detail limitations and criteria that must be taken into account when constructing the solution. Along with each requirement is its priority level, ranging from suggested to critical. A critical requirement must be implemented for the project to be complete. Recommended requirements are not essential but greatly improve the quality of the system and provide a better user experience. Finally, suggested requirements are the lowest priority and should only be implemented once the others have been properly executed to add additional features or improve ease of use.

2.1 Functional Requirements

- Critical
 - The system will allow users to input their current health symptoms.
 - The system will allow users to input any products or medications they recently started using.
 - The system will display potential products, medications, or foods that might be causing the user's symptoms.
 - The system will display correlations to the user based on previous FDA complaints relating symptoms to goods and products and the current user's similarities to these cases.
 - The system will implement an algorithm to generate correlations between products and user-inputted symptoms.

- Recommended
 - The system will allow users to create an account before searching.
 - The system will suggest next steps for the user to address the problem indicated by the search.
- Suggested
 - The system will temporarily save a user’s basic information for a subsequent search.
 - The system will allow users to save searches and their results for future reference.
 - The system will allow the user to generate a printable report of the search results.

2.2 Non-Functional Requirements

- Critical
 - The system will be free of major bugs.
 - The system will be responsive.
- Recommended
 - The system will be intuitive to use.
 - The system will have a simple and clean user interface.

2.3 Design Constraints

- Critical
 - The system will be web-based.
 - The system will run on Google Chrome, Firefox, Safari, and Internet Explorer.
- Recommended
 - The system will be hosted locally.
- Suggested
 - The system will be a mobile application.

Chapter 3

Use Cases

The following use case diagram (Figure 3.1) and task descriptions provide a visual representation of the actions that can be taken by a user in this solution. The figure represents a user actor, with connected lines attaching the actor to all possible use cases. These use cases are visualized in ovals describing their specific action. Each use case has a particular goal, actor, preconditions, postconditions, steps to execute, and exceptions. Each use case is summarized below with descriptions of these attributes.

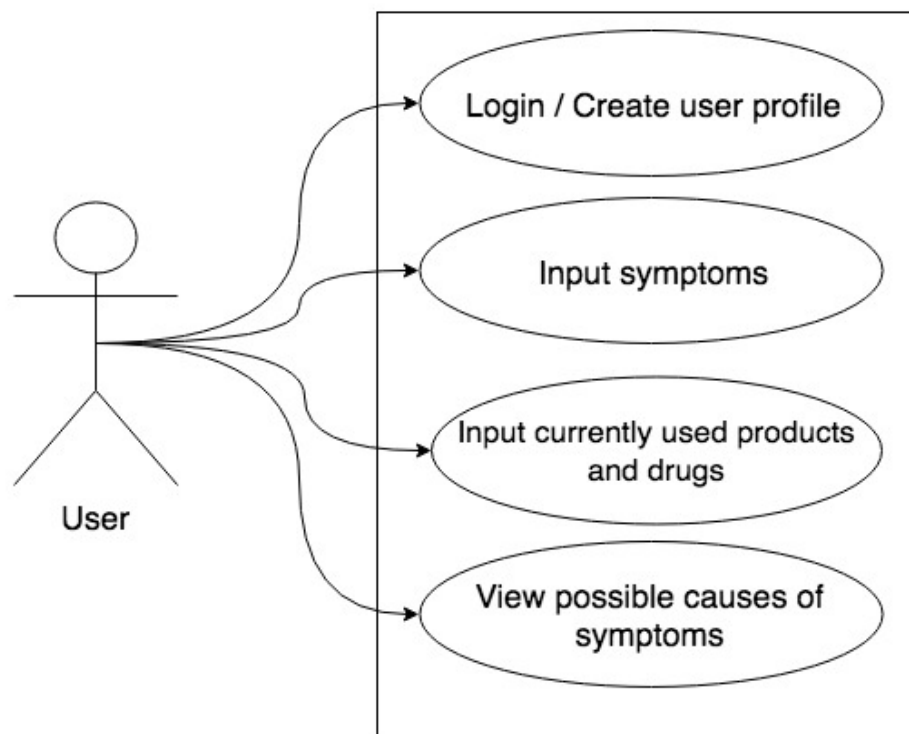


Figure 3.1: Use Case Diagram

3.1 Login/Create user profile

- Login
 - **Goals:** Enter the system.
 - **Actors:** User.
 - **Pre-Conditions:** User must have an account.
 - **Steps:** User inputs email and password then clicks submit to enter the system.
 - **Post-Conditions:** Application displays user information for confirmation purposes.
 - **Exceptions:** Checking for correct password, valid email, etc.
- Create User Profile
 - **Goals:** Provide general information about a user for a more accurate set of results. Access the system.
 - **Actors:** User.
 - **Pre-Conditions:** None.
 - **Steps:** User selects between drop down list options, text boxes, and radio buttons to provide basic information to Symptom Search when registering.
 - **Post-Conditions:** Application displays user information for confirmation purposes. The account is created when the user inputs valid information and clicks the register button.
 - **Exceptions:** Checking for valid string entries, valid email, valid password, etc.

3.2 Input symptoms

- **Goals:** Provide information about symptoms a user experiences for the machine learning algorithm.
- **Actors:** User.
- **Pre-Conditions:** User must have a profile set up with basic information such as name and email.
- **Steps:** User selects areas on a human model or chooses words from a list to select and input the symptom(s) they have been experiencing.
- **Post-Conditions:** Application displays list of symptoms that have been selected by the user.
- **Exceptions:** None.

3.3 Input currently used products and drugs

- **Goals:** Provide information about products a user consumes for the machine learning algorithm.
- **Actors:** User.
- **Pre-Conditions:** User must have a profile set up with the necessary basic personal information.
- **Steps:** User chooses from a list to select and input a product they have been consuming.
- **Post-Conditions:** Application displays list of products that have been selected by the user.
- **Exceptions:** None.

3.4 View possible causes of symptoms

- **Goals:** Provide information about drug/product interactions for users so they can see which products are most likely causing their symptoms.
- **Actors:** User.
- **Pre-Conditions:** User must have a profile set up with the necessary personal information such as age, sex, pregnancy status, etc. and the user must have already selected at least one symptom and at least one product to check for interactions.
- **Steps:** None.
- **Post-Conditions:** User sees a table of likelihoods that the drug(s) given are correlated to the symptoms(s) given. The user also sees a list of suggested drugs to cause the given symptom(s).
- **Exceptions:** None.

Chapter 4

Activity Diagrams

The Activity Diagram (Figure 4.1) illustrates the flow of possible activities that can be performed by the user in this solution. The diagram does not include the system actions, such as processing symptom and product information.

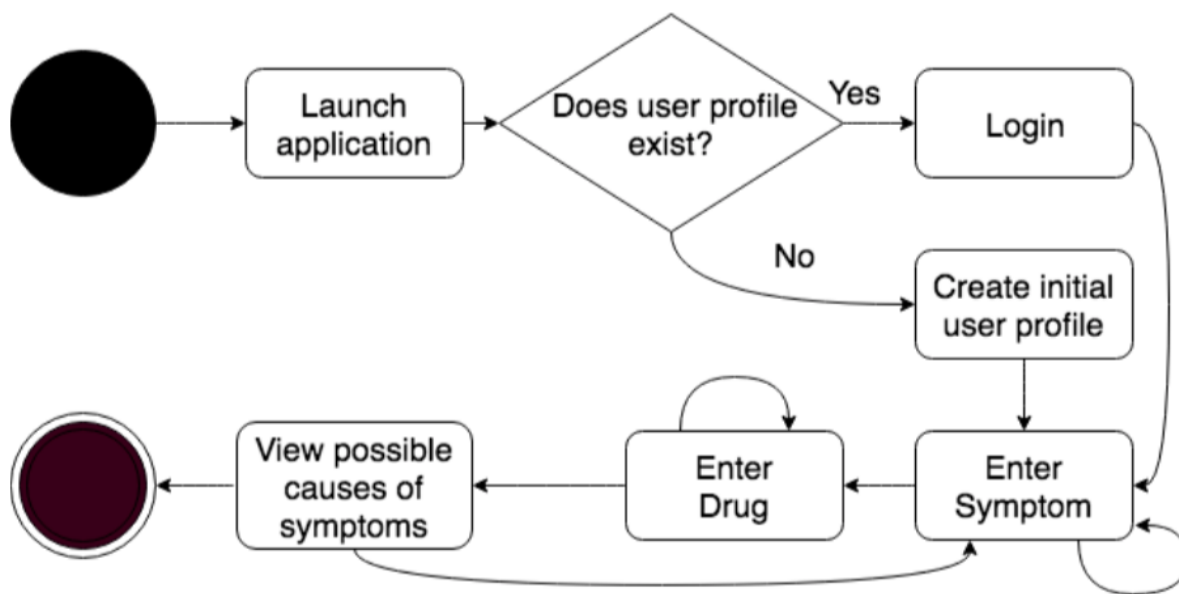


Figure 4.1: Activity Diagram

Chapter 5

Architectural Diagram

We used a client-server architecture to build our project as shown in (Figure 5.1). The client, a user, will interact with our web page and input basic information and symptoms. This information will be sent to the server, and the server will compare the given data with the data stored in the database. The client's screen will then display the results of the prediction. We used this architecture specifically because the server handles the requests from the clients and generates results using information stored in the database.

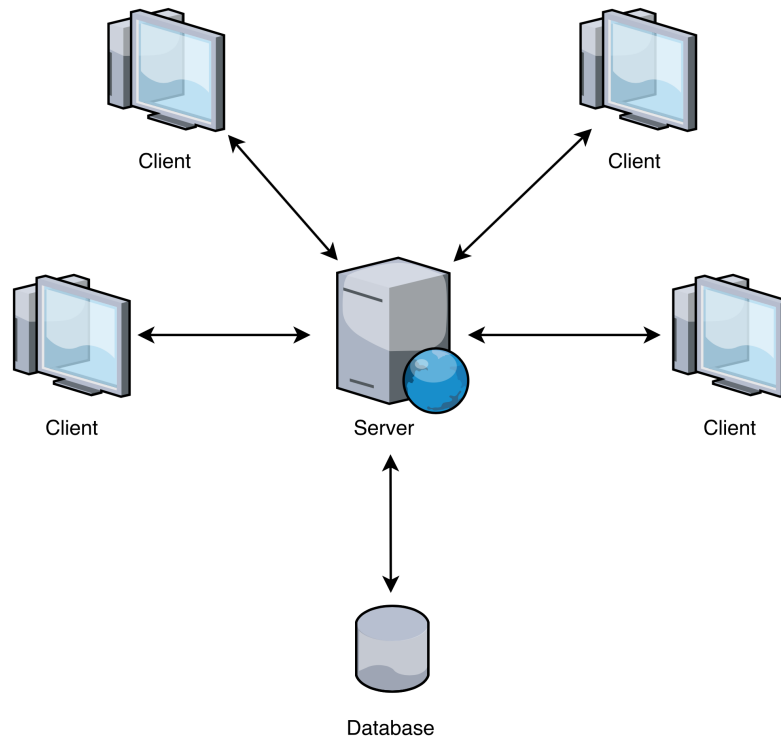


Figure 5.1: Architectural Diagram

Chapter 6

Technologies Used

- HTML: A language used for web page structure and form fields.
- CSS: A language used for web page styling.
 - Skeleton: A CSS library used for uniformity of style as well as responsiveness of webpages.
- MEAN.js: A full-stack JavaScript tool that assists in the development of web applications using MongoDB, Express, AngularJS, and Node.js.
 - MongoDB: A database used because of its ability to quickly process unstructured data, which is needed for machine learning and data mining.
 - * Mongoose: A tool to model MongoDB objects in an asynchronous environment like Node.js.
 - Express: A web framework used because of its robust set of features, compatibility with Node.js, and simplicity of use for efficient development.
 - * Connect-mongo: A package used to store database sessions in Express.
 - * Express-session: Express middleware that implements sessions in encrypted tamper-free cookies.
 - Node.js: A server framework used because of its ability to scale easily, write quickly, and process data efficiently.
 - * Bcrypt: A Node.js package used to process and salt passwords for security purposes.
 - * Body-parser: A Node.js middleware used to parse response bodies from the server.
- Python: a scripting language that is commonly used in machine learning and data science. We used it for the development of our ranking algorithm. During development, we also used a variety of Python libraries such as Numpy, Json, Pymongo, and Scikit-learn. Listed below are the descriptions of the Python libraries we utilized in production and in the final product.

- Numpy: used for matrix manipulation.
- JSON library: easily read JSON files and create JSON objects for ease of processing.
- Pymongo: a library used for accessing and manipulating data in a Mongo database.
- Scikit-learn: popular machine learning library for Python.
- Matplotlib: used to plot diagrams of the processed data and clustering.
- SciPy: library providing tools for science and data processing. I used dendrogram, linkage, clustering, for hierarchical clustering.
- editdistance: provides a C-based Levenshtein distance algorithm that is faster than python implementations.

Chapter 7

Description of System Implemented

7.1 An Overview of the System

Symptom Search is a web application built using HTML, CSS, MongoDB, Express, and Node.js. Python is used for the ranking algorithm component. Users are able to create an account and then input their symptoms and the products that they are using. The system will return a list of products that are most likely causing their symptoms, as well as a list of suggested products.

Our web application is comprised of a home page (Figure 7.1), a registration and login page (Figure 7.2), a symptom and product input page (Figure 7.4), and a results page (Figures 7.5 and 7.6).

The registration and login page uses the bcrypt Node.js package to hash and store passwords. We use an asynchronous approach to create and verify a hashed password. We store our hashed password in the database. When creating an account, the user agrees to Symptom Search's terms. A disclaimer is linked above the "Register" button on the registration page. The disclaimer text that pops up upon clicking that link is shown in Figure 7.3.

The symptom and product input page is comprised of a form for symptom and product input. The user is able to begin typing the brand name of the product that they are using, and he or she may choose an option from the smart drop-down list that auto-completes with the desired product. Similarly, for the symptom input, the user is able to type the name of their symptom, and he or she is able to choose an option from the smart drop-down list.

After a user has inputted their symptoms and products, they may press the "Submit" button and be directed to the results page. This page will display the results of the ranking algorithm that uses the FDA Adverse Effects Report data to determine how likely the drugs are to be correlated to the symptoms provided in the input screen.

7.2 The Dataset

Our dataset is a specially curated subset of the FDA Adverse Effects Report database. We did not need all of the data from the FDA for this system, so we stored only what we needed. The data is organized by each drug. Within each drug scope is a collection of all of the symptoms that a user reported experiencing with that drug. Within each symptom is the frequency of reports to the FDA of that symptom with that drug. In other words, this is the number of times users reported experiencing that symptom with that specific drug. An example of this tree-like structure is shown below:

- Drug 1
 - Symptom A
 - * Frequency of Symptom A with Drug 1
- Drug 2
 - Symptom A
 - * Frequency of Symptom A with Drug 2

When curating our database, we counted the number of complains of symptoms with certain drugs and grouped them in the above structure. These frequencies are counted and stored beforehand so when prompted by the user, Symptom Search generates and displays results quickly since it does not need to compute anything. These frequencies are only recomputed if we update the database with new data from the FDA Adverse Effects Reports dataset. We do not add user input data to the database. This way we can make sure the information is as accurate as possible. Information in the database comes only from the FDA where legitimate complaints about product adverse effects are submitted.

7.3 The Results Page: The Ranking Algorithm

The results page displays two kinds of results.

7.3.1 Table of Likelihoods

First, the results page displays a table of likelihoods of the given drugs and symptoms (Figure 7.5). Each likelihood is calculated using the following algorithm. Given a drug and symptom, the algorithm finds that drug in the database, and for each symptom stored within that drug scope in the database we retrieve the stored frequency. We then find

the maximum frequency within that symptom scope. We calculate the likelihood of each given symptom using the following equation:

$$\text{Likelihood of a given drug and symptom} = \frac{\text{Given symptom frequency}}{\text{Maximum frequency of a symptom in the given drug scope}} * 100\%$$

For example, in the tree structure above, if we are given “Drug 2” and “Symptom A,” we would retrieve the “Frequency of Symptom A with Drug 2.”

This table is accompanied by a key to help the user interpret the results (Figure 7.5). If the symptom is not listed under that drug, the frequency of the symptom, and therefore the likelihood, is zero. If the likelihood is 100, this means that the given symptom was the most reported symptom for that drug in the FDA database. This does not mean that the given drug is necessarily causing the symptom or reaction. This means that the given symptom is the most likely to be correlated with that drug compared to all the other symptoms reported for that drug.

7.3.2 Suggested Products

The results page also displays one or more lists of suggested products that are ranked in order, beginning with the most likely to be correlated with the inputted symptoms. The number of lists is equal to the number of symptoms the user input to the system. In each list, each of the products’ likelihoods is calculated using the following equation:

$$\text{Likelihood of each item in the ranked list} = \frac{\text{Given symptom frequency}}{\text{Total number of complaints in the given drug scope}} * 100\%$$

The system then orders the top ten likelihoods found from largest to smallest. The system only displays ten results for readability. This does not mean there are only ten products that are likely to be correlated with the given symptom(s). The system might also output less than ten products in the case that there are few likelihoods. Additionally, the products might not necessarily be highly correlated with the symptoms at all. This list is just a suggestion of other products that might be correlated. The greatest likelihood could potentially not be very high at all, but it happens to be the greatest of all the likelihoods calculated.

This result is useful because it provides users with information on other products that might also be correlated with the symptoms they input (if not more correlated). This way, users can see a list of other drugs that may be more likely to be correlated with their symptoms than the combination they originally input.

Symptom Search

Symptom Search helps you discover possible correlations between symptoms you are experiencing and any drugs and products you are currently using.

BEGIN

Figure 7.1: Start Screen



Login

E-mail:

Password:

Register

First Name:

Last Name:

E-mail:

Password:

Confirm Password:

[By clicking register, I agree to your terms.](#)

Figure 7.2: Register or Sign In

Do not rely on Symptom Search to make decisions regarding medical care. While we make every effort to ensure that data is accurate, you should assume all results are unvalidated. Symptom Search should not be used for medical diagnosis and does not replace a conversation with your doctor. By agreeing to our terms, you authorize Symptom Search to store your private data. Your data will not be shared with others.

BACK

Figure 7.3: Disclaimer With Registration

Symptom Search

Please indicate one or two symptoms you are experiencing and one or two drugs you are using. Based on past user complaints, the results will indicate the likelihood of either drug causing your symptom(s).

Symptom Details

Symptom 1 Experienced:

e.g. CHILLS

Symptom 2 Experienced:

e.g. HEADACHE

Medical Details

Medicine 1:

e.g. IBUPROFEN

Medicine 2:

e.g. MUCINEX

RESET

SUBMIT

LOGOUT

Figure 7.4: Input Drugs and Symptoms



Symptom Search

The table below describes the likelihood that one or more symptoms are correlated to the inputted drug(s). Please refer to the Key for an additional description of this likelihood.

Drug Vs. Symptom Likelihood			Key	
	chills	headache	Range	Likelihood
MUCINEX	0	33.33	0 - 25	Very Unlikely
IBUPROFEN	16.67	61.11	25 - 50	Somewhat Unlikely
			50 - 75	Somewhat Likely
			75 - 100	Very Likely

Figure 7.5: Initial Results: A Table of Likelihoods and an Accompanying Key

*The list(s) below show up to ten drugs that are most likely to cause the given symptom(s).
The drugs are ranked in order, beginning with the highest likelihood.*

**Drugs Most Likely to Cause:
CHILLS**

1. HYDROCHLORIDE B1
2. HEXETIDINE
3. MULTI VIT
4. FLU VACCINE VII
5. FLURAZEPAM HYDROCHLORIDE
6. THIETHYLPERAZINE
7. DOXEPINE
8. OXYGESIC AKUT
9. GLAGIN
10. SOFLAX (CANADA)

**Drugs Most Likely to Cause:
HEADACHE**

1. VASTINAN
2. SILLICA
3. DIUREX
4. GABA
5. L-5 HTP
6. MICORNOR
7. CLARITHROMYC
8. ATRIPLA
9. ALISKIREN
10. HYDROCODONE/PARACETAMOL

BACK

LOGOUT

Figure 7.6: Suggested Drugs

Chapter 8

Design Rationale

To build the product, we decided to use MEAN.js, which is a full-stack JavaScript solution that assists in the development of web applications using MongoDB, Express, AngularJS, and Node.js. This tool makes web development easy on the front end and has a simple back end.

We chose MEAN.js specifically because it uses MongoDB as its data storage. Out of the MEAN stack, we used MongoDB, Express, and Node.js. According to the developer, Ryan Dahl, “the aim of Node.js was to create real-time websites with push capability.” Node is useful for building fast, scalable network applications. The NPM tool allows users to install a set of publicly available packages. Some of the packages that we installed were bcrypt, express, express-session, mongoose, and mongodb. We used the express-session package to set up sessions. Since HTTP is a stateless protocol, the web servers do not keep track of who is visiting a page. In order to keep track of a user after they have registered or logged in, implementing sessions and cookies was necessary.

MongoDB is an open source NoSQL database that stores JSON-like documents that provides scalability and flexibility in querying and indexing data. Due to the fact that we are using the FDA Adverse Effects Report data, which is kept in JSON files, we wished to find a way to easily store and quickly access this unstructured data. Since MongoDB uses JSON formatting and we have a lot of JSON-formatted data, we decided to use MongoDB as our database. MEAN.js uses MongoDB and makes front end easy to develop, so we decided to use it to develop our product.

We used Python to process all of our data and compute our results to display on the web page. Python is a scripting language that is simple, and it can be used server-side for web development. We chose Python to write our ranking algorithms due to its simplicity and the variety of libraries available to Python for data processing. We used libraries such as Numpy, Json, Pandas, and Mongoose to manipulate, process, and handle the data.

Chapter 9

Test Plan

Our test plan has focused on testing the accuracy of the suggested symptom and product interaction that is generated by our prediction system. We have also gained feedback on the usability of the system and during Beta testing. We made adjustments to our system based on the results of both phases of testing.

9.1 Alpha Testing

During Alpha Testing, we tested the implementation of our system using both iterative and regression testing. In iterative testing, we designed our system, build it out, then repeated the process based on how the system operated. With regression testing, we ensured that our system continued to work with all the additional pieces as the design built upon itself. Using these tests, we continued to verify that the system worked correctly and make any necessary adjustments before they became large-scale problems.

9.2 Beta Testing

During Beta Testing, we have acquired input from real users by having them try out our prediction engine. We have reached out to peers and had them test our system and give us feedback on its accuracy as well as the site's ease of use and user interface. Through this level of testing, we have and will continue to gain feedback and improve our system.

Table 9.1: All Test Cases

Feature	Test Procedure	Expected Outcome	Result
First user creation	Sign up for a new account with the database empty	A user account is created	Success
Returning user sign-in	Login with user information already in database	Validate login data and redirect to symptom input page	Success
User input symptoms	Input 1 or 2 symptom(s) into form fields and click "Add Information"	Symptom information is added to the database associated with correct user	Success
User input medicine taken	Input 1 or 2 medicine(s) into form fields and click "Add Information"	Medicine information is added to the database associated with correct user	Success
Select symptom or medicine from drop down	Click the arrow in a form field, and select a symptom or medicine from the list	The symptom or medicine is shown in the form field	Success
View results	Click "View Results"	A page of symptom causes and likelihoods is displayed	Success
Logout	Click the logout button	The user is returned to the login page and cannot access input page until they log in again.	Success

Chapter 10

Ethical Analysis

Symptom Search can be classified both as medical technology and as machine learning technology. As a result, the team working to design, implement and produce this piece of software has an ethical imperative to ensure that sound ethical practices are employed in all technical areas of our project and throughout the life cycle of our product.

10.1 Team and Organization

10.1.1 Team

The team designing and implementing this solution consists of four students, Isabela Figueira, Neesha Godbole, Angelina Poole, and Kelly Wesley, along with their faculty advisor, Dr. Yi Fang. All four students maintain open and frequent lines of communication, ensuring that any inter-team ethical issues will be identified early and dealt with swiftly. All team members have a long-time relationship with each other, and with the faculty advisor, Dr. Yi Fang. This will ensure that the team remains open and honest about progress, design decisions, obstacles, and ethical quandaries.

10.1.2 Organizational

Though all student group members are individual contributors to this project, our advisor Dr. Yi Fang is an employee of Santa Clara University. Santa Clara University is also our primary sponsor of this project. Santa Clara University, as well as the individual contributors to this project, maintains high ethical standards for all projects and partnerships it plays a role in. For these reasons, we believe that our organizational ethics are sound with respect to this project.

10.2 Social

Symptom Search aims to offer helpful and accurate information to its users about products and medicines that may be contributing to symptoms they experience. As a result, the product risks misinforming its users if the machine learning algorithm isn't perfect or if a disease is causing symptoms instead of an FDA-regulated product. To maintain an ethical product, the team behind Symptom Search broadly displays a disclaimer (Figure 7.3) warning users that any conclusions or suggestions made by our product should be taken under a doctor's advisement. The team discussed the details of the disclaimer with Dr. Yi Fang and other officials at Santa Clara University to guarantee that the disclaimer conforms to established legal and ethical guidelines.

Every person is unique in their reaction to different products and medications: just because a few other people got a rash after taking a certain pill does not necessarily mean the same pill is the cause of your rash as well perhaps a new lotion is the culprit or an illness. Symptom Search can only return to the user correlations based on the number of people who reported their symptoms and the drug that caused the symptoms through the FDA complaint system. The tool analyzes this data and returns which drugs have a high likelihood of causing the inputted symptoms based on previous user experience and the optional input of any drugs or medications the user started taking recently. However, there is a large amount of personalization and additional factors such as medical history, environmental factors, etc. that contribute to an accurate medical diagnosis. Symptom Search as a tool has a large room for error; if users take the results at face value and alter their lifestyle accordingly, there is a large chance they will be doing so in error. The tool is intended to be a resource, not a recommendation system.

The potential to misinform a user introduces the possibility of harm to the user and her health and well-being. To prevent this harm, as previously mentioned, our system website includes visible disclaimers informing that the results indicated on this site are not to be taken as recommendations or causations and that Symptom Search does not replace a conversation with one's doctor. In this way, the system reminds any visitor that Symptom Search is a tool based only on numbers and does not consider an individual's situation the way a medical professional would. By providing users this information, our design reduces the potential for harm and supports the health and well-being of users. The decision to include this disclaimer is justified through the perspective of the Rights Approach, in that it respects the moral rights of the individual to know the truth about the knowledge they are receiving and make their own decisions accordingly.

10.3 Product

Symptom Search takes in user health information and uses it to generate predicted symptom and product interactions. This information is a crucial aspect of the product's functionality. One ethical consideration with our product is determining if user health information needs to be stored and, if so, how to best securely store sensitive health information. Another consideration is the machine learning model and its ability to predict relationships between symptoms and products. In order to be ethically mindful and ensure accurate predictions, Angelina, Isabela, Kelly, and Neesha checked in with their advisor, Dr. Yi Fang, and heavily tested the model. Since this project is based in software, there are no physical safety risks that are involved.

As mentioned, user privacy is a key ethical concern. When someone uses the Symptom Search platform, they have the option to create a login with a user name and password, which will be stored in our database. The next page prompts the user to input their symptoms and any medications or products they recently started taking or using, information which will also enter the database as inputs tied to that users login. Because the user input data is stored in our database in relation to a users personal login information, it is imperative that our project take careful consideration of its privacy. The ACM Code of Ethics obligates computing professionals to Respect the privacy of others. By this tenet, our system must ensure that user data is protected from attacks by malicious parties.

Part of this is accomplished by our system only storing the user data that is relevant to the program. In our websites original design, the user was asked to input a variety of details including age, location, and gender. These details were stored along with the symptoms the user input and her login information. If the system was hacked, these health details associated with that person via her login would be available to external parties. As we continued working with the machine learning component of this project, we realized that due to insufficient data from the FDA complaints, it was not possible to incorporate the age, gender, and location factors when returning possible problematic medications or products to the user. Hence, these inputs were irrelevant. Rather than leaving them on the website, we elected to remove them as inputs only storing the necessary personal information and thereby reducing the potential severity of damage in the event of a security breach. In addition to limiting user inputs, our system also incorporates some security measures to decrease the potential of a breach. Particularly, the website uses secure sessions and encrypted passwords, which provide built-in security and secret keys to mitigate the risk of an external party stealing user data.

Another piece of user privacy involves the accuracy of data. It is imperative that the system include procedures that allow users to review their inputs and ensure they are accurate. On our input page, this is accomplished by providing a Clear button for the user that will reset all inputs if they need to be corrected. With these features, our website promotes ACMs ethical component of respecting user privacy and ensuring the accuracy of inputs and therefore results.

10.4 Science, Technology, and Society

Like all engineering innovations, Symptom Search has the potential to impact society in various ways. First, as a Web platform, it supports society's reliance on the Internet and Web browsers for information accessibility. While this continues to be the norm, it is debated that most (if not all) individuals have access to these resources, potentially making our platform only usable by those with computers, smart phones, tablets, and Wifi. In this way, however, the platform also contributes to the rise in information accessibility by making data that should be available to the public actually available through a useful tool.

Chapter 11

Lessons Learned

11.1 Angelina

While I have taken an introductory web programming course and done basic web programming to build a functional multi-page website before, I have not worked with the MEAN stack. The extent of my previous knowledge was working with HTML, CSS, PHP, and a MySQL database. Before we decided to use the MEAN stack, we tested a variety of options, such as using LAMP or creating a web application from scratch using Flask. However, due to the constraints of this project, we chose to use most of the MEAN stack, which included MongoDB, a NoSQL database that allowed us to store our JSON files, Express, and Node.js. I improved my programming skills in working on the front end with Javascript, as well as working on a clustering algorithm for the symptoms and products on the back end. Thankfully, building a web application with Node.js was well documented. However, we ran into some obstacles trying to combine the knowledge that we learned with the different tutorials, as there was often more than one way to do the same thing.

11.2 Isabela

I have worked with datasets before, but machine learning using health data is different to say the least. During the development process, I couldn't just think about ranking drugs and symptoms and outputting the likelihoods of their correlations. I had to consider how the user would interpret the results, the fact that our data is limited and potentially incomplete or skewed, and how it might affect the user's health and wellbeing. In machine learning, we often use specific features to describe data. In our case, I thought to use age, sex, and location to represent a user. However, due to the fact that severe cases are more often reported in this specific dataset, not every complaint is logged in this dataset, and not every report includes age, sex, and location, it would be unwise to use these data points to truly

describe a user. If I were to describe a user this way, I would have to ignore the incomplete user complaints, but this could change the data completely and change the resulting recommendations, which could lead to inaccurate results. Our product, while seemingly simple, has the potential to affect people's health. I consequently had to be weary of this danger while we developed our product.

In regards to teamwork, I learned a great deal about project management and team communication. We are a team of four friends, and without clearly defined leadership, it proved difficult to assign work to different team members. We kept open lines of communication about the project and made sure we updated the rest of the team when certain parts of the project were finished. In the end, we just knew what had to get accomplished and made sure the tasks we assigned ourselves were completed.

11.3 Kelly

Throughout the course of this project, I learned that communication is the single biggest factor in determining whether or not a project will be a success. Although I was unfamiliar with all of the technologies we subsequently used in Symptom Search, communicating my questions, concerns, and clarifications to our advisor and soliciting information from friends who have had experience with web applications gave me the confidence to make decisions about which technologies would yield the best result. In addition, communicating with my group mates was at times pivotal to our success because schedules, work-division, and integration issues could not have been solved if we hadn't take the effort throughout the project to keep every group member up to date about each teammate's progress.

11.4 Neesha

Despite having taken almost all of my programming courses at this point in my academic career at Santa Clara, I struggled immensely when our group attempted (at first) to utilize the MEAN stack. I also struggled to use the various tools we incorporated throughout the project such as Github, MongoDB, and Node JS. These tools are all unfamiliar to me and at no point was I taught them in a classroom setting. As a result, I learned that the Internet is a great resource for knowledge about technological tools. I also learned the frustration of trying to accomplish a task one way for a large amount of time only to learn that your way is incorrect. In sum, many of my lessons came from trial and error, as that is often the zigzag path to success in the world of engineering.

Chapter 12

Conclusions

Symptom Search is a Web application for individuals experiencing untraceable symptoms. The product generates predictions of drugs that might be the source of the user's ailments. This system uses a machine learning algorithm with FDA Adverse Effects Reports data from various user complaints to make educated recommendations of possible correlations between products and inputted symptoms. The technologies we used - including the MEAN Stack and Python data processing tools - allowed us to develop a responsive and useful Web application to address the need. The design decisions and documentation in this report allowed us to make informed and purposeful choices regarding our structure and implementation.

12.1 Suggested Additions to the Implemented System and Future Work

If allotted more time, we would add the following features:

- Using clustering to group the symptoms and products would improve the auto-fill displayed in the smart drop-down menu. Since the FDA dataset contained a wide variety of products and symptoms, it would make it easier for the user to find the symptom or product that they want to input. For example, the system has symptoms of "blood glucose decreased," "blood glucose increased," and "blood glucose." It would improve usability to cluster these similar phrases.
- We solely used a ranking algorithm so that we knew what the results meant. Since we are not specialists with health data, we did not want to make any predictions using features that were not immediately connected to the drug and reactions that could create results that were potentially not accurate. We aimed to create the best product possible despite the fact that we did not have much direction on the health aspect of the data. Much of the data such as gender, age, and location was not filled in for many of the complaints. We chose to not use

those features in the machine learning process since we did not want to leave any data out of our system due to incomplete entries. In the future, we would somehow incorporate this data to more completely analyze the data and hopefully return different predictions than just a ranking algorithm would.

- We only used a subset of the large amount of data in the FDA database. To make this product live on the Web, we would add all the rest of the data from the database and put the website on a server to make it accessible. Since this is a senior design project, we decided to build the system on local host so we would not be hindered by space limitations on rented servers.
- Adding a user profile page that keeps track of past uses of the system would allow the user to see a profile page after logging in. Their personal information, as well as previous searches would be shown. The user would then be able to update their personal information. The user would also be able to view previous searches without having to complete the form again.
- Instead of a fixed number of input boxes for symptoms and products in the form, we would add the ability to change the number of inputs. Adding this functionality would give the user the ability to customize how many symptoms and products he or she is able to input. This would give the user the ability to compare more drugs and symptoms simultaneously on the results page rather than completing multiple searches.
- To make the product more accessible for those without computers, we wanted to turn the system into a mobile application. We did not implement this during this project because we were constrained by the framework we chose to build the system in on the Web. With more time, we could create a mobile application using the necessary frameworks as well as the Web application we focused on.

Chapter 13

Acknowledgements

We would like to thank our Advisor, Dr. Yi Fang, for his guidance and mentoring throughout the duration of this project, especially in the areas of data mining and machine learning. Without his vision and encouragement, our product would not be what it is today. We would also like to thank our thesis readers, Dr. Silvia Figueira and Dr. Darren Atkinson, for their input and advice throughout our process. Finally, we would like to thank Santa Clara University's School of Engineering for its continued support of our scholastic endeavors throughout our years as students here.

Appendix A

User Manual

A.1 User Functionality

It is a known fact that almost all drugs that are used to treat any kind of ailment or disease have the possibility of causing side effects. When experiencing symptoms with seemingly no cause, a patient often turns to online resources to seek answers before they decide to meet with a medical professional. The first thought that a patient typically has when experiencing a strange symptom is not that they are experiencing their symptom due to a side effect of medication, but rather that they are experiencing this symptom because they are ill.

In order to make this distinction, our web application, Symptom Search, uses the symptoms that the user is experiencing and the products that they are taking and displays a likelihood metric of how likely it is that the product is causing that symptom, as well as a suggested list of other products that are likely to cause the user's inputted symptoms.

A.1.1 Account Creation and Login

1. On the home page, the user clicks "Begin" and is directed to the registration and login page.
2. On this page, the user creates an account by inputting their name, email, and password.
3. To login, the user inputs their email and password into the login section.

A.1.2 Symptom Input and Product Input

1. After the user has either registered or logged in, they are directed to the symptom and product input page.
2. At this page, the user inputs one or two symptoms that they are currently experiencing.

3. Then, the user inputs one or two products that they are currently taking that they suspect might have contributed to their symptoms.
4. For ease of use, the user can click “Reset” if they made a mistake in their symptom or product input to reset the input fields.
5. After inputting this information, the user submits their information and is directed to the results page.

A.1.3 Results

1. On the results page, the user is able to see a likelihood table of their symptoms and products, as well as a list of suggested products that are most likely to cause the user’s inputted symptoms.
2. The user can log out of the system by clicking the “Logout” button or click the “Back” button to return to the symptom and product input page to check other symptoms and products.