NEW JERSEY INSTITUTE OF TECHNOLOGY
CAPSTONE PROJECT FALL 2020

Prof. O. Eljabiri | Prof. P. Vaish

December 12, 2020

EVE'S APPETITE FOR CURE

FINAL REPORT

GEORGE VARGHESE	(CS-491)
KETKI AMBEKAR	(YWCC-691)
SIRISHA BOJIREDDY	(YWCC-691)
YASSMIN ALI	(CS-491)

Table of Contents

Chapter 1 Introduction

Project Definition

Eve's Appetite for Cure is a non-profit organization created with the intention to create solutions to those affected by cancer. The name is a tribute to the founder, Lisa Marks-Canty's mother who had lost a battle with stomach cancer. Having various expertise in the technology sector through her various business ventures, she hopes to use this experience to guide her through this passion project.

Mission

Our focus is on providing as much comfort and support to the fighters and caregivers of fighters who are going through the battle of fighting gastric cancer. Our platform helps us affect change by providing people with an outlet to speak to each other and to relate to each other in a way no one else can understand because they are not living through this nasty disease.

Problem Definition

In today's world everyone knows someone that has been affected by cancer. It's an extremely difficult and delicate situation to be in. Whether you yourself are suffering from the disease or you are taking care of someone that is affected by the disease. Although patients have access to information through their hospital and doctors, oftentimes, caregivers are burdened with seeking out medical information and guidance in order to properly take care of their loved ones.

Although many patients have access to their doctors, information on the web and many different cancer forums it does not provide a unique social networking experience. The goal is to create an application that helps our users to create a personal connection with each other and be able to keep up with one another.

Cancer forums have the limit of just the interaction in one forum post and users will not be able to tell if they receive accurate information. As a result the Need to Speak platform was built but the application still needed some more functionality in order to become a user friendly social networking application.

Our task as the data science team was to add more functionality to the existing app to create a better user experience. We were responsible for creating three modules Sentiment Analysis, Semantic Search Engine, and Topic Classification. It allows the user to search for topics that would be the most relevant to them, which makes it easier for them to find what they need.

Glossary

Topic Classification: The identification of the topics of conversation in a Cancer Forum and classifying user posts into these topics. We selected ten different topics in order to categorize the posts that were web scraped.

Pre trained model: a model that was trained on a large benchmark dataset to solve a similar problem (*towardsdatascience*). In this project, pre trained models were used to compare accuracy with the trained models.

Topic Modelling: An unsupervised machine learning technique that's capable of scanning a set of documents, detecting word and phrase patterns within them.

LinearSVC (Support Vector Classifier): Utilized to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data.

Word Embeddings: Is any of a set of language modeling and feature learning techniques in natural language processing.

SBERT: A so-called twin network which allows it to process two sentences in the same way, simultaneously.

Restful API: A so-called twin network which allows it to process two sentences in the same way, simultaneously.

Sentiment: It is an attitude, thought prompt or judgement prompted by feeling. (*Merriam Webster*)
Sentiment Analysis: It a natural language processing technique used to interpret and classify emotions in a subjective area. (*monkeylearn*)

Spark NLP: It is an open source natural language processing library, built on top of Apache Spark and Spark ML. (*towardsdatascience*)

Deep Learning: It is a subfield of Machine Learning concerned with algorithms inspired by the structure of the brain. (*machinelearningmastery*)

StandfordNLP: is a Python natural language analysis package. It contains tools, which can be used in a pipeline, to convert a string containing human language text into lists of sentences and words, to generate base forms of those words (*StandfordNLP*).

Iteration/Revision Updates

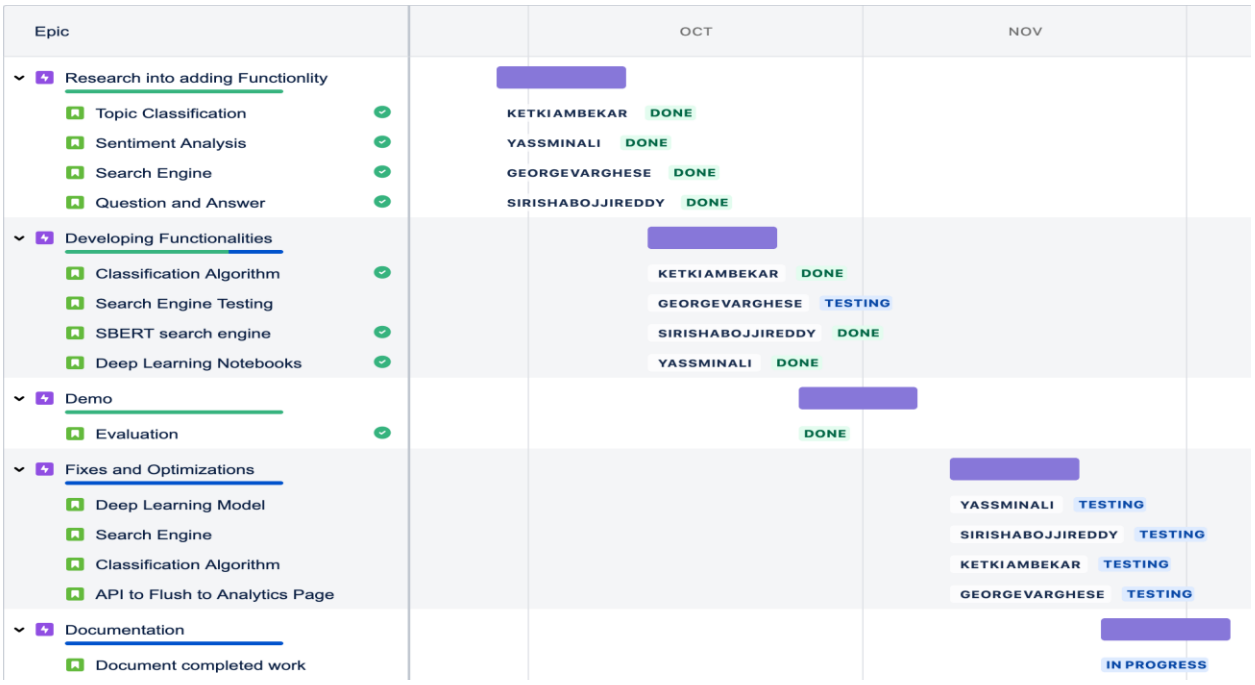
For the first phase of the project we were focused on trying to find the best algorithms for our modules. We split up our tasks into three different modules, Topic Classification, Sentiment Analysis and Semantic Search. Initially we were all focused on finding different algorithms that would provide the best results. As a result, we spent a decent amount of time looking for algorithms that would prove to be the fastest. So we lost time for some of our modules and were unable to create APIs for Topic Classification and the Elasticsearch portion. However, we were able to produce rest APIs for the Sentiment Analysis and SBERT Semantic Search modules.

Chapter 2 Project Management

Task Analysis

At the beginning of the project we had established three deliverables which were the modules we previously discussed. As the project went on, the actual output of that topic area became clearer and thus guided the task breakdown. In the initial Gantt chart we scheduled each person to focus on research for the first sprint and then start to gear towards development. As the semester progressed everyone was working on their individual modules. At the end of each sprint, we checked in with our sponsor during which we presented our deliverable updates and received feedback.

WBS/GANTT Chart



Epic		
Research into adding Functionlity		
Topic Classification ✓	KETKIAMBEKAR	DONE
Sentiment Analysis ✓	YASSMINALI	DONE
Search Engine ✓	GEORGEVARGHESE	DONE
Question and Answer ✓	SIRISHABOJJIREDDY	DONE
Developing Functionalities		
Classification Algorithm ✓	KETKIAMBEKAR	DONE
Search Engine Testing	GEORGEVARGHESE	TESTING
SBERT search engine ✓	SIRISHABOJJIREDDY	DONE
Deep Learning Notebooks ✓	YASSMINALI	DONE
Demo		
Evaluation ✓		DONE
Fixes and Optimizations		
Deep Learning Model	YASSMINALI	TESTING
Search Engine	SIRISHABOJJIREDDY	TESTING
Classification Algorithm	KETKIAMBEKAR	TESTING
API to Flush to Analytics Page	GEORGEVARGHESE	TESTING
Documentation		
Document completed work		IN PROGRESS

Roles

Name	Role	Module
George Varghese	Project Manager	Open Source Search Engine

Ketki Ambekar	Data Scientist	Topic Classification
Sirisha Bojireddy	Data Scientist	Semantic Search Engine
Yassmin Ali	Data Scientist	Sentiment Analysis

In terms of roles the team consisted of one project manager and the remaining the rest were data scientists. We had 4 members in this team including myself who were responsible for our own modules. Ketki was responsible for creating the topic classification model. Sirisha was responsible for creating the semantic search engine model. Yassmin was responsible for creating the sentiment analysis module. I was also responsible for creating a search engine module using an open source API. However, if any of us had any issues, we would update each other throughout the week. For instance, we realized that we needed more data entries in our database because our models were not able to produce great accuracy. So we all spent more than a week labelling more data helping us to increase our accuracy.

Risk Analysis

Throughout this project there have been many risks that we have had to mitigate. While we were able to solve certain issues we also faced other unexpected issues.

Risk	**Probability**	**Mitigation**
Unfamiliarity with subject area	medium	Accounted for research periods within project management in order to ensure significant time to become familiarized with technologies/ concepts.
Hardware Performance	low	Utilized AWS in order to run our modules which helped reduce stress on machines.
Adhere to changes in the timeline	medium	Prioritizing tasks at hand to maximize efficiency and value delivered
Lack of labelled data	high	We spent a sprint trying to label as much data as possible which significantly improved accuracy.

One of the main risks that we encountered in this project was a lack of familiarity with the subject area. Half the team was doing their masters in data science so they were knowledgeable about the task at hand. However, I only took one intro to data science course so everything was fresh and we were at a beginner level. Most of the team did not have any practical real world experience in this area of data analysis. Otherwise, we were very committed to learning about our modules and ended up learning enough to complete our modules.

Another issue that we ran into was using our hardware to test Elasticsearch. I was responsible for testing an open source search engine but I was unable to fully test the capabilities of the search engine. However, I was able to resolve this matter by using a virtual machine on AWS and was able successfully test out Elasticsearch in that manner.

One of the major issues we had this semester was the amount of unlabeled data that we had. This was important to address because all of our modules performance is dependent on the amount of data we had access to. So we ended up dedicating a sprint to try to label more data for our modules. As a team effort we managed to label over 700 entries of data which drastically improved the accuracy of our modules.

Additionally, we were unable to follow the original timeline as our research extended beyond the scope of the first sprint. Our topic classification ended up taking a lot more time to develop than expected due the unforeseen difficulties that had occurred. As a result, we had to discuss with our sponsor to have a functional module instead of having its API ready.

Chapter 3 Define

Project Stakeholders

Project Founder/Sponsor
The Eve's Appetite for Cure (EAFC) app is a vision of Ms. Lisa Marks-Canty who is the founder of Nest Global Solutions (our sponsor company). She was inspired to create the Eve's Appetite for Cure platform after losing her own mother to the cancer disease. She thoroughly understands the need for such a platform and has the vision to take it forward. She graciously advised us on which topics to finalise for topic classification, she suggested some important additions that were not highlighted by our topic detection algorithm.

Project Mentor
Smit Purohit is a Data Scientist at Nest Global Solutions and also our mentor for the project. He is highly skilled in Machine Learning, Natural Language Processing and Deep Learning subjects. He also happens to be an NJIT alumnus (M.S. Data Science). Having worked on this project previously via the NJIT Capstone program, he has a deep understanding of project requirements and his insights and guidance were deeply valuable to us. He was responsible for communicating the requirements of the projects with both the front-end and back-end teams and coordinating between them. We regularly met Smit after each sprint and shared our progress with him. We also approached Smit when we had any decisions to make or were facing difficulties. We ensured that our project outcomes were as per Smit's expectations.

Development Team Members
Our team of students of NJIT Capstone Program are major stakeholders in the project. They are divided into two sub-teams: the front-end team and the backend team, each consisting of 4 student developers. Each of these teams are led by one project manager (from among the four team members). The teams are responsible for coming up with a solution to the defined problem and developing the components of the projects with their technical expertise, with help and direction from their mentor Smit.

Users
The users of the EAFC app are an important stakeholder as their opinions and User Experience will be a huge shaping factor in the direction of the project. The Users can be divided into following categories:
<u>Fighters</u>: These are the users who are experiencing stomach cancer themselves. They can reach out for emotional support, medical advice, etc on the EAFC app.
<u>Survivalists</u>: These are the users who have previously experienced stomach cancer but have successfully defeated the disease. Their experience would be valuable to other users.
<u>Care-Givers</u>: These are the individuals who are friends/family of a patient fighting stomach cancer, who may want to understand how to help their loved-one better.
<u>Medical Professionals</u>: This group of users involves medical professionals who are doctors, nurses, etc who may be able to offer medical advice to other users.

NJIT Capstone Executive Team

Professor P. Vaish: Professor Vaish was our academic advisor for the project. We got to learn about various project management techniques under him. We also learned about SCRUM, MicroService Architecture and about error handling. Thanks to professor Vaish, we were able to study and implement industry best practices in our project.

Professor O. Eljabiri and the NJIT Capstone Executive team: Thanks to Professor Eljabiri and the executive team of NJIT Capstone Fall 2020, we got a clear understanding on how to present our project. We got expert tips on best practices in speaking, organizing our content, dressing well, etc to make our oral as well as written report presentations better. They made the process fun by conducting a lively quiz about various project management practices and UML diagrams.

Project Scope

Functional Requirements

Module	**Requirement**
Topic Classification	The module should identify broad topics being talked about in a public forum related to cancer.
Topic Classification	The microservice should detect the topic of all posts in the forum
Search Engine	The microservices should take input from the front-end.
Search Engine	The microservice should return relevant search terms typed in the search box.
Sentiment Analysis	The microservice should detect the sentiment (positive, negative) of all posts posted in the forum

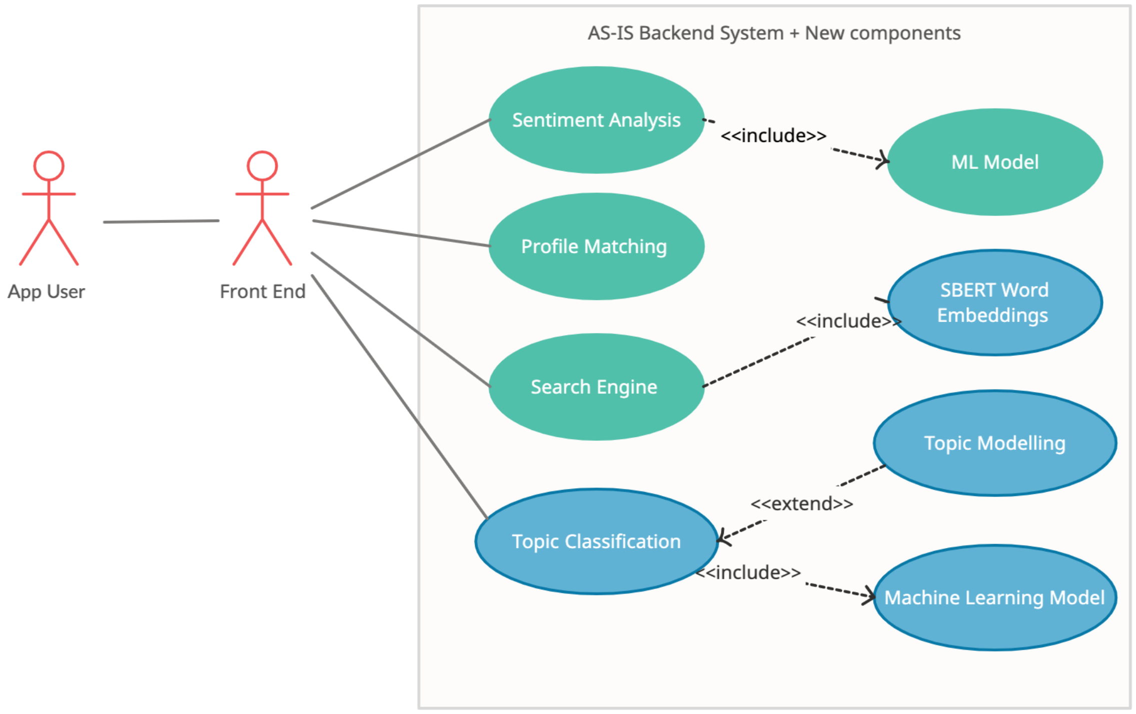
Non - Functional Requirements

Module	**Requirement**
Topic Classification, Sentiment Analysis	The system responses from micro service should be instantaneous.
Topic Classification, Sentiment Analysis	The sentiment and topic detection should be as accurate as possible
Topic Classification	The system should be able to detect wide variety of topics accurately

Search Engine	The search results should be displayed in real time
Miscellaneous	User Data storage should be HIPAA Compliant
Miscellaneous	The system responses should be considerate, courteous and polite towards users who are going through hard times. It should make users feel safe and supported.

AS-IS processes Use Case Diagram

The following is the Use case diagram that shows the AS-IS backend system components (green) and the new System Components (blue).



Requirement Gathering Methods

1) **One-on-one interviews:**
We had some one-on-one meetings with Smit (our mentor) in addition to the regular group meetings, to understand how the different modules talk to one another. What exactly are the inputs and what would be the format of the outputs. How do the modules fit in with the existing modules, and what does our end product look like. This took some effort on our

part. It took us surprisingly longer than expected to chalk up our sequence diagram. The interviews were especially helpful not only for envisioning the UML diagrams, but also to define tangible expected goals. We also met with Lisa, to get her inputs and understand her vision and requirements when finalising the classes for topic classification.

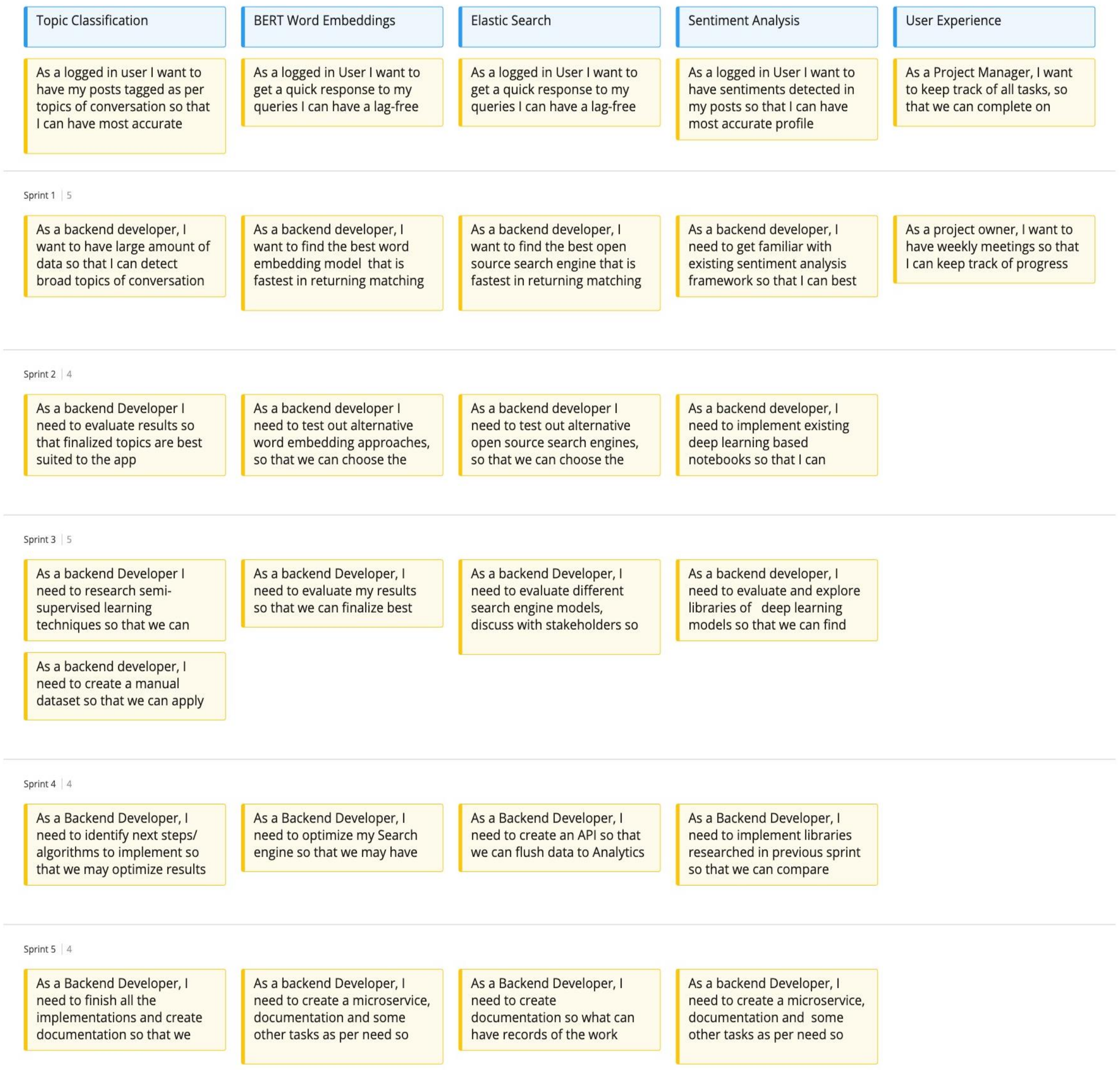
2) **Analysing Existing Documents:**
The documentation from the previous team was pivotal in helping us understand the finer details of the project. They brought us up to speed on some of the work that had already been done. This allowed us to build up on previous work and understand where to start from scratch in some modules. We also referred to some research papers that had solved similar problems, this gave us a lot of insight on how to tackle some of the issues, and helped us anticipate some issues which we hadn't thought of before.

3) **Prototyping:**
We also used the prototyping method for requirement gathering in our project. This was done in the form of the previous iteration of the project where the students created a sentiment analysis model. It was a good model, but better accuracy was needed. Based on observations from that model, we were able to create our improved deep learning sentiment analysis model.

4) **Regular Text Communication:**
Finally, when we had any questions, our sponsors and mentors were available to answer questions via our Slack channel. We would ping then with our questions and they would revert to our queries within the same day.

User Stories (SCRUM Format)

Following are our user stories created using Miro Board software. These are helpful in keeping us on track during each sprint, and are helpful for each member to know exactly what to do at all times. (The user storyboard software truncated some of the stories while exporting for print. The but the entire board can be viewed at https://miro.com/app/board/o9J_lbxudXs=/)



	Topic Classification	BERT Word Embeddings	Elastic Search	Sentiment Analysis	User Experience
	As a logged in user I want to have my posts tagged as per topics of conversation so that I can have most accurate	As a logged in User I want to get a quick response to my queries I can have a lag-free	As a logged in User I want to get a quick response to my queries I can have a lag-free	As a logged in User I want to have sentiments detected in my posts so that I can have most accurate profile	As a Project Manager, I want to keep track of all tasks, so that we can complete on
Sprint 1 | 5	As a backend developer, I want to have large amount of data so that I can detect broad topics of conversation	As a backend developer, I want to find the best word embedding model that is fastest in returning matching	As a backend developer, I want to find the best open source search engine that is fastest in returning matching	As a backend developer, I need to get familiar with existing sentiment analysis framework so that I can best	As a project owner, I want to have weekly meetings so that I can keep track of progress
Sprint 2 | 4	As a backend Developer I need to evaluate results so that finalized topics are best suited to the app	As a backend developer I need to test out alternative word embedding approaches, so that we can choose the	As a backend developer I need to test out alternative open source search engines, so that we can choose the	As a backend developer, I need to implement existing deep learning based notebooks so that I can	
Sprint 3 | 5	As a backend Developer I need to research semi-supervised learning techniques so that we can As a backend developer, I need to create a manual dataset so that we can apply	As a backend Developer, I need to evaluate my results so that we can finalize best	As a backend Developer, I need to evaluate different search engine models, discuss with stakeholders so	As a backend developer, I need to evaluate and explore libraries of deep learning models so that we can find	
Sprint 4 | 4	As a Backend Developer, I need to identify next steps/ algorithms to implement so that we may optimize results	As a Backend Developer, I need to optimize my Search engine so that we may have	As a Backend Developer, I need to create an API so that we can flush data to Analytics	As a Backend Developer, I need to implement libraries researched in previous sprint so that we can compare	
Sprint 5 | 4	As a Backend Developer, I need to finish all the implementations and create documentation so that we	As a backend Developer, I need to create a microservice, documentation and some other tasks as per need so	As a Backend Developer, I need to create documentation so what can have records of the work	As a backend Developer, I need to create a microservice, documentation and some other tasks as per need so	

Feature Driven Development (FDD)

Given the long term nature of our project, it is a perfect candidate for a Feature Driven Development (FDD) methodology of development. We have a high number of instances of newer members joining the team and needing to come up to speed with project needs and current development. FDD makes this on-boarding process easier.

FDD Requirements Grouping

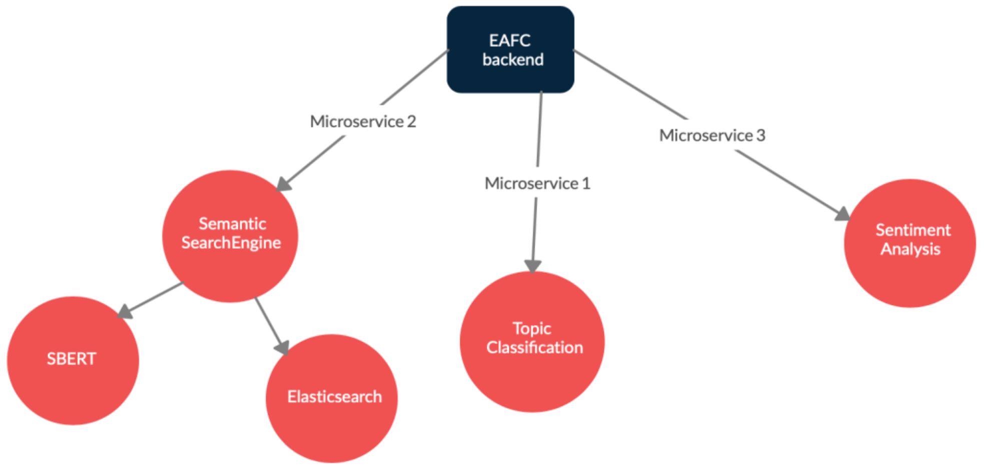
The term ‘user’ hereafter refers to the front end (and not the app user), as depicted in the use case diagram. The term ‘End user’ refers to the person using the app.

Features List
Developers need to detect and decide broad topics in a forum
User should be able to accurately detect topics in forum posts
User should experience minimum delay in fetching the results
Framework should be light weight and resource efficient.
User is given pertinent search query results only.
User is given search result as quickly as possible
User is given accurate result of whether message is of positive or negative sentiment
User data should be handled in HIPAA compliant manner
End User should feel safe and supported in the app

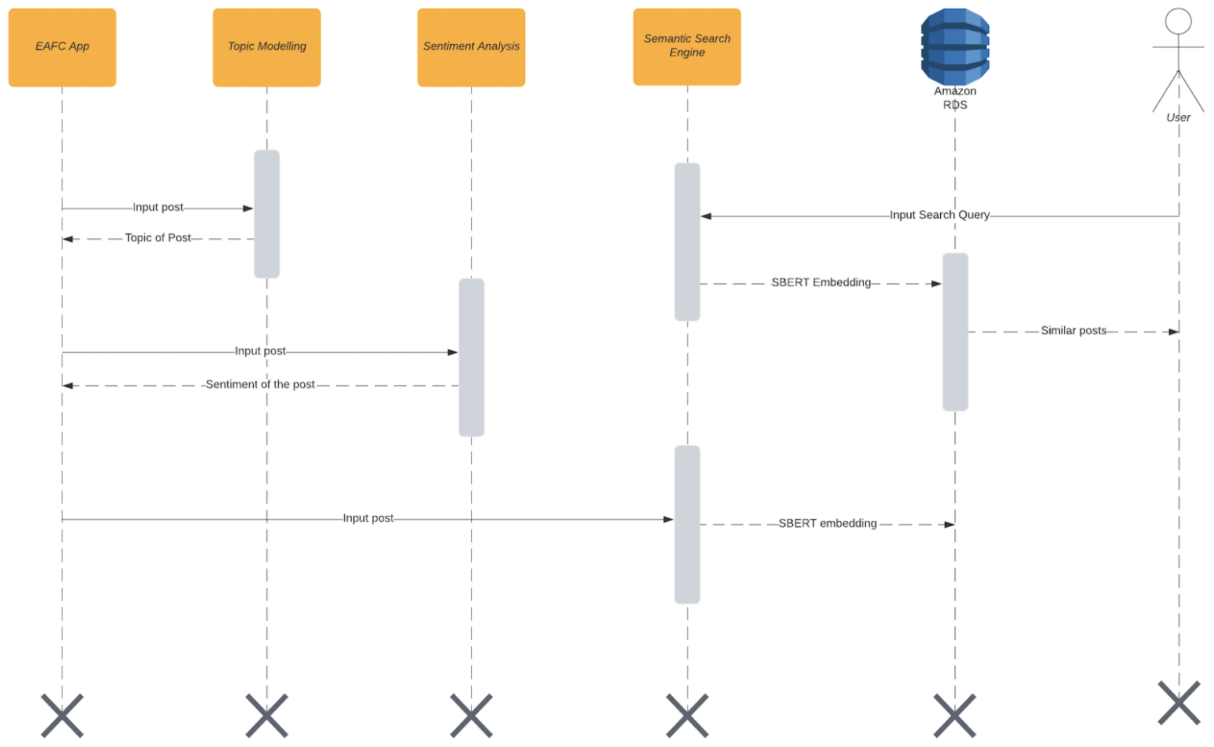
<u>Modules Color key:</u>
Green - Topic Classification
Blue: Search Engine
Yellow: Sentiment Analysis

Chapter 4 Design

Our ER Workflow diagram looks somewhat like this where our overall task was divided into three main microservice modules



Here is the sequence diagram which is an overview of how each of our modules work with the Application and the database. This diagram gives us a more detailed explanation of the internal working of our different modules.



Implementation Details

Module 1 - Topic classification

It is the identification of the topics of conversation in our Cancer Forum and classifying user posts into those broad topics

This module is further divided into three sub modules:

1) Topic Modelling

Before we could identify the topics of conversation, we had to identify what topics are generally discussed in the cancer forums. For this purpose, we scrapped data from Reddit Cancer Forums, approximately 2346 posts. We performed Topic Modelling to determine ten broad topics of conversation from the forum. We used the Latent Dirichlet Allocation (LDA) algorithm for this purpose.

Latent Dirichlet Allocation (LDA) is a generative statistical model. It allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. Topic examples: Chemo Treatment, Post treatment care, Surgery etc.

2) Dataset creation

We first performed Exploratory Data Analysis (EDA) and Data Cleaning to form a better dataset and improve our overall accuracy. We removed punctuations, word contractions such as isn't, I've, etc. We also converted all the text to lowercase to avoid confusion and case-sensitivity such as 'hello', 'Hello' and 'HELLO'. We also removed stop words such as 'a', 'an', 'the', 'I', etc., which did not actually contribute or add value to our vocabulary and analysis.

We then had to convert the sentences into smaller chunks called 'tokens' with the process of Tokenization. This is done to create a vocabulary of the different words from the entire dataset. We tried to identify the different parts of speech and grouped together the inflected parts of a word so that they can be analyzed using Lemmatization. For example, 'flying', 'fly' are considered different when they actually have the same meaning or root word that is 'fly'. We had to group such words together with this process.

We initially tried to perform semi-supervised learning methods, but since it took a lot of research, we decided to move forward with the supervised learning methods by utilizing manual labor for almost one sprint. We were able to create a manually labelled dataset of 700 forum messages, from the scraped data from reddit forum with the combined effort from the entire team.

3) Machine learning

The Machine learning algorithms don't understand words, they understand numbers. We had to represent words in our dataset numerically using the process of Vectorization for which we used various methods including, count vectorization, TFIDF vectorization, Uni-gram, Bi-gram and Tri-gram representations.

Alternative solutions and a comparison between them:

We applied six different Machine learning algorithms, such as Multinomial Naive Bayes, Logistics Regression, Random Forest, K-Nearest Neighbors (KNN), Support Vector Machine (SVM) and Linear SVM to our dataset. The Linear SVC performed better than the other models and gave us an accuracy of 62%.

We had another alternative solution with the semi-supervised learning models, but since we didn't have enough documentation and time, we had to move on with supervised learning which was done manually by the team. Even though the manual labelling took quite some time, we were sure of the accurate results rather than the new semi-supervised learning models.

Module 2 - Semantic Search Engine

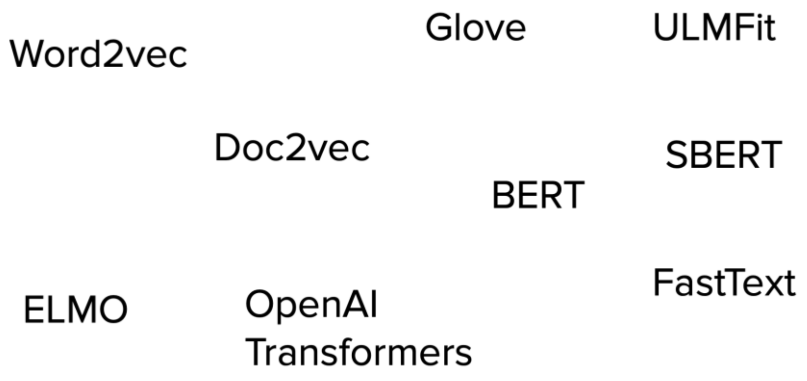
There are two independent modules in this semantic search engineer module.

1) SBERT:

Text data cannot be used as such to find similarity, so we need to visualize and find relationships between the text using plots and algebraic equations for which we used word embeddings. What are Word Embeddings? The process of converting a word into an array of numbers called vectors is called a word embedding. It is the state of an Art.

Alternative solutions and a comparison between them:

Here is the list of different language models that we looked-up in order to find the best model for our problem.



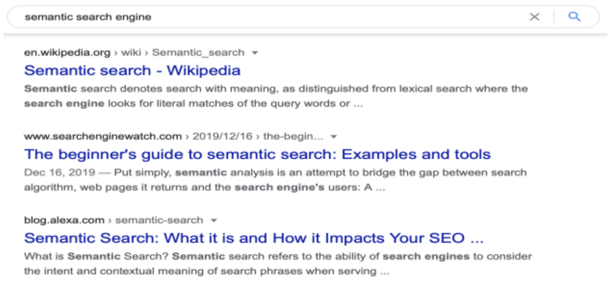
Word2vec, which converts every word to a vector of numbers, doc2vec which converts sentences, paragraphs or documents to vectors instead of just words. In w2v, if we search for bank, riverbank and bankdeposit are considered similar no matter what the context is. Glove Embeddings takes into account the context of the word according to the sentence it belongs to. Fast Text language model creates sub vectors for better understanding and produces embedding for rare words that are not available in the training corpus. These are all the sequential learning models, given 1,2,3 predicts 4 and 1,2,3,4 predicts 5 in sequential order.

We also have bidirectional language models like ELMO, which has the sense of previous and next words, Universal Language model with fine-tuning (ULMFit) which uses Multilayer bi-LSTM. This model includes pre-training and fine-tuning for good performance and accuracy. There are OpenAI transformers, also called decoder models with no attention sublayers. ELMO and ULMFit were bidirectional but this model was forward only, so it was a failure model in the improvements.

Bidirectional encoder language model (BERT) performs various functions on large amounts of data. There are many variations of the BERT model like ALBERT, DistillBERt, ROBERTa, etc. And we decided to work with Sentence BERT for our context based semantic search engine.

When you give 10000 sentences as an input to the BERT or ROBERTa model, it requires approx 50million combinations to compute and takes 65hours. Sentence transformer, which is the modification of BERT, is used to find the similarity between sentences within 5seconds. This information is provided in the below ACL 2019 paper "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, by Nils Reimers and Iryna Gurevych from UKP-TUDA.

Our plan for the semantic search engine looks like a google search engine. When we type a search query, it will give us the similar posts like below:



semantic search engine

en.wikipedia.org › wiki › Semantic_search
Semantic search - Wikipedia
Semantic search denotes search with meaning, as distinguished from lexical search where the **search engine** looks for literal matches of the query words or ...

www.searchenginewatch.com › 2019/12/16 › the-begin...
The beginner's guide to semantic search: Examples and tools
Dec 16, 2019 — Put simply, **semantic** analysis is an attempt to bridge the gap between search algorithm, web pages it returns and the **search engine's** users: A ...

blog.alexa.com › semantic-search
Semantic Search: What it is and How it Impacts Your SEO ...
What is **Semantic** Search? **Semantic** search refers to the ability of **search engines** to consider the intent and contextual meaning of search phrases when serving ...

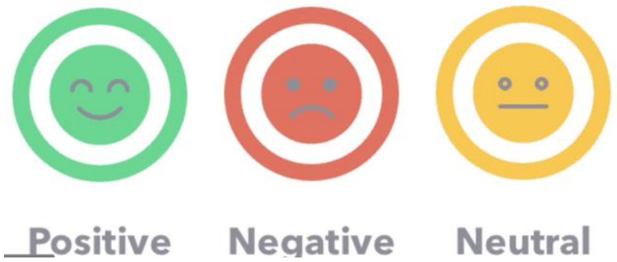
2) Elasticsearch:

For this module, we initially had to connect to the AWS RDS database, upon update of the database, we had to insert the record into the elasticsearch package, which in-turn gave us the output of the search as the most similar posts.

The elasticsearch package in python uses apache lucene. It is a powerful search software package that performs full text search by indexing the data from the database according to the search query. It was faster compared to the other traditional search methods, but we were not able to complete this task, because it was completely a new topic and our requirements were not clear and we had very little time for the end of the semester. Although we were not able to complete this model, we created documentation from our research and handed it to the sponsor for future reference.

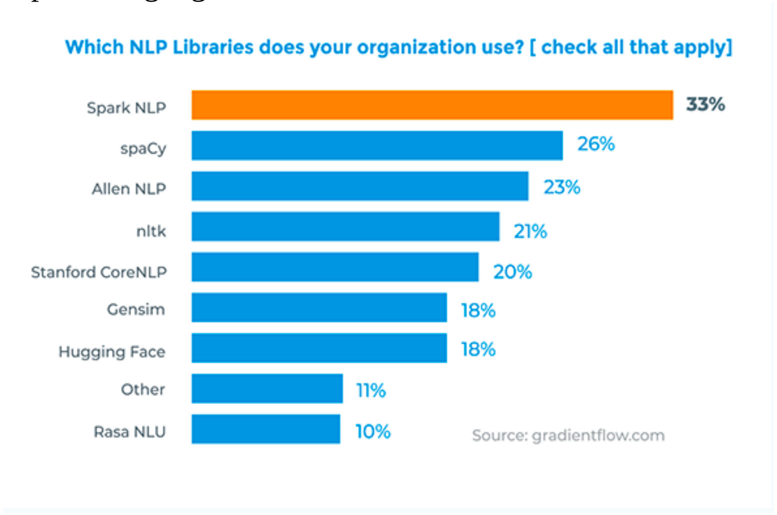
Module 3 - Sentiment Analysis

Sentiment Analysis is the process of analyzing people's opinions, sentiments, attitudes, and emotions towards a product, service, or an event. There are basically three main sentiments, positive, negative and neutral.



We will be using this sentiment analysis for profile matching. We will be able to integrate this module with other modules in the project, such as semantic search, in order to provide a unique experience for each user. Our goal for this module is to build a model that predicts sentiment of a user based on his/her post or comment.

Our previous capstone team was able to find the sentiment analysis of around 500 posts using various machine learning algorithms. Our aim is to improve the accuracy of finding sentiments using deep learning algorithms.



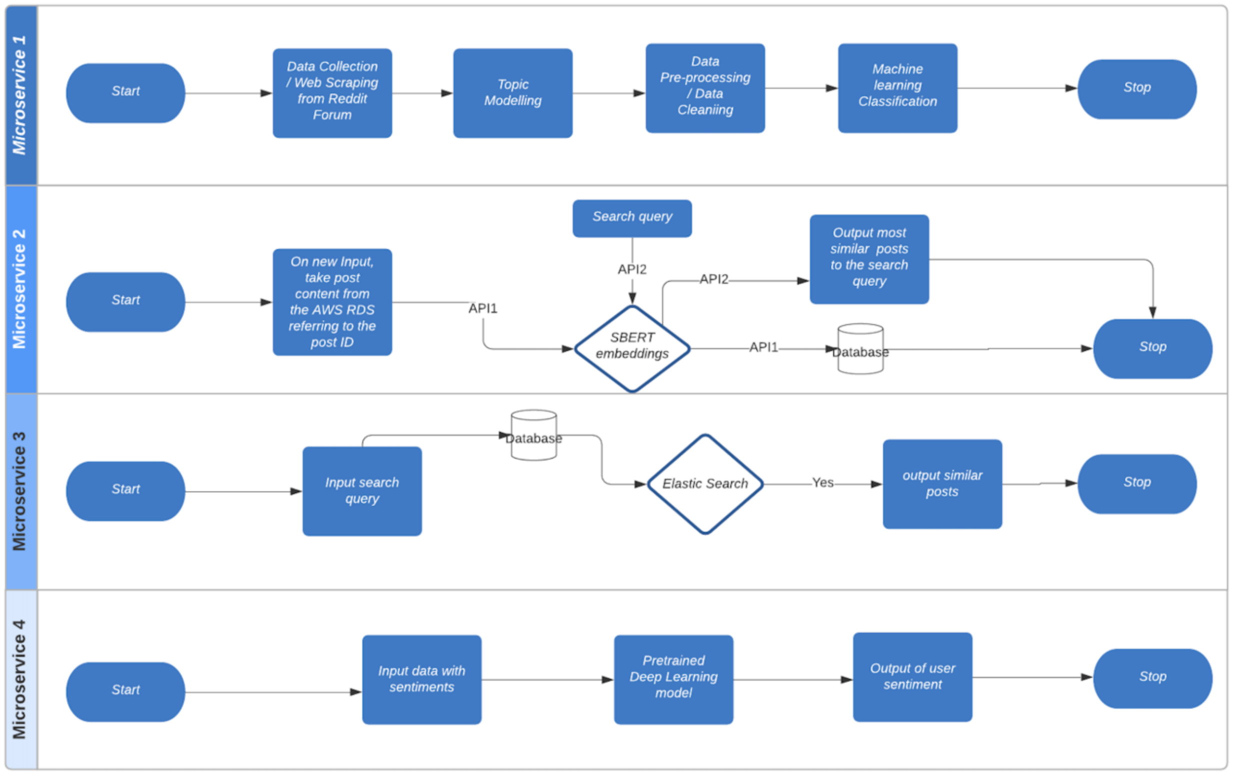
There were various deep learning algorithms that we looked up, but decided to proceed with SparkNLP model, because of its wide usage in the industry and high accuracy compared to other models with less computation time.

Alternative solutions and a comparison between them:

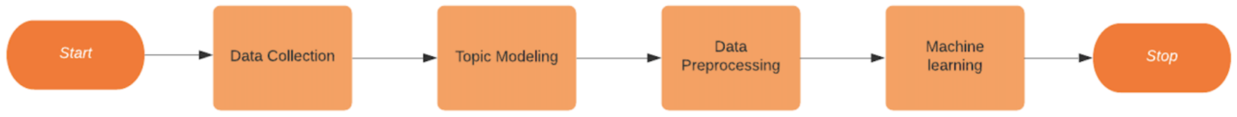
We used a pre-trained sparkNLP deep learning model which gave us an accuracy of 67% and fine-tuned it to achieve a higher accuracy of 80%. We then used StandfordCoreNLP pre-trained deep learning model which gave us an accuracy of 63%. We finally created a RESTful API that took the user’s input post and gave us the output of the user’s sentiment.

Chapter 5 Development

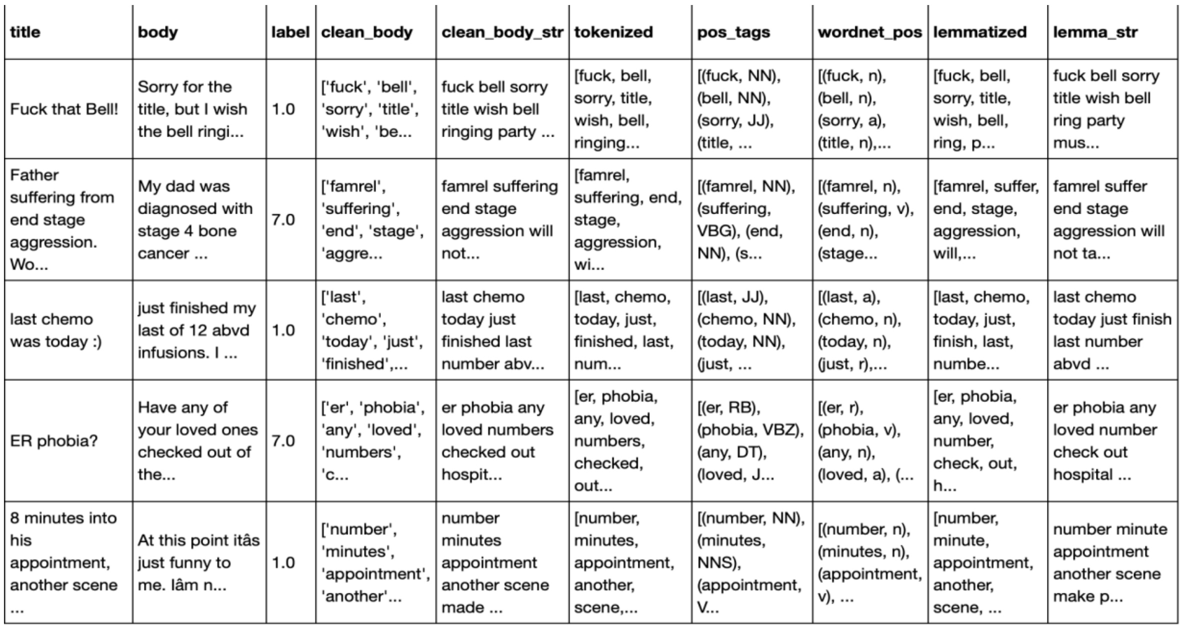
Our project was divided into multiple microservice modules which are explained below in detail with the following swimlane diagram.


Microservice 1
Start
Data Collection / Web Scraping from Reddit Forum
Topic Modelling
Data Pre-processing / Data Cleaniing
Machine learning Classification
Stop
Microservice 2
Start
On new Input, take post content from the AWS RDS referring to the post ID
API1
Search query
API2
SBERT embeddings
API2
Output most similar posts to the search query
API1
Database
Stop
Microservice 3
Start
Input search query
Database
Elastic Search
Yes
output similar posts
Stop
Microservice 4
Start
Input data with sentiments
Pretrained Deep Learning model
Output of user sentiment
Stop

Module 1 - Topic classification

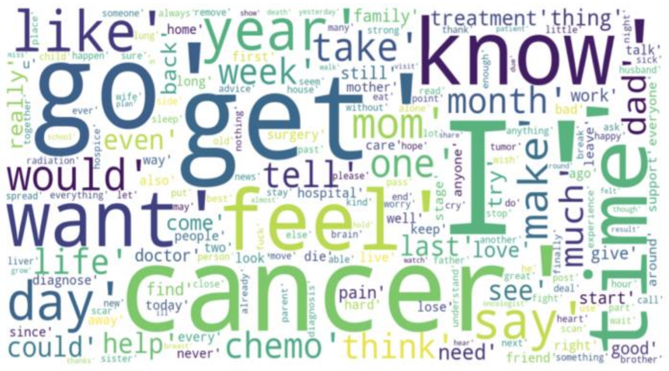


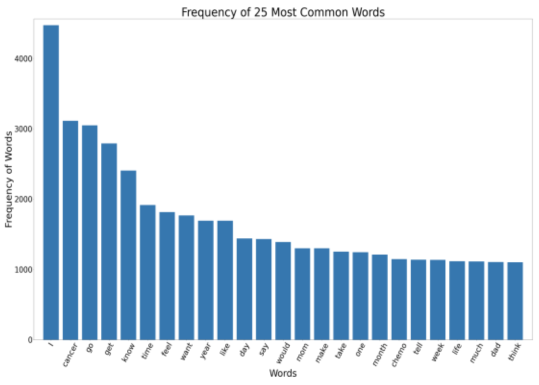
Our input data for this module, which was scrapped from the reddit cancer forum looks like below. We performed data cleaning and data preprocessing and created tokenized, pos-tags, lemmatized columns before building the machine learning algorithm models for classification.

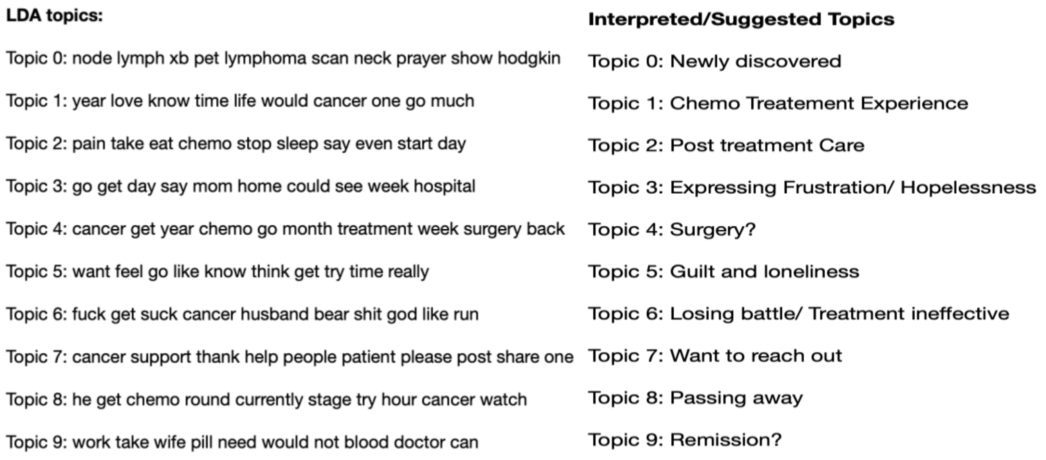


title	body	label	clean_body	clean_body_str	tokenized	pos_tags	wordnet_pos	lemmatized	lemma_str
Fuck that Bell!	Sorry for the title, but I wish the bell ringi...	1.0	['fuck', 'bell', 'sorry', 'title', 'wish', 'be...	fuck bell sorry title wish bell ringing party ...	[fuck, bell, sorry, title, wish, bell, ringing...	[(fuck, NN), (bell, NN), (sorry, JJ), (title, ...	[(fuck, n), (bell, n), (sorry, a), (title, n),...	[fuck, bell, sorry, title, wish, bell, ring, p...	fuck bell sorry title wish bell ring party mus...
Father suffering from end stage aggression. Wo...	My dad was diagnosed with stage 4 bone cancer ...	7.0	['famrel', 'suffering', 'end', 'stage', 'aggre...	famrel suffering end stage aggression will not...	[famrel, suffering, end, stage, aggression, wi...	[(famrel, NN), (suffering, VBG), (end, NN), (s...	[(famrel, n), (suffering, v), (end, n), (stage...	[famrel, suffer, end, stage, aggression, will,...	famrel suffer end stage aggression will not ta...
last chemo was today :)	just finished my last of 12 abvd infusions. I ...	1.0	['last', 'chemo', 'today', 'just', 'finished',...	last chemo today just finished last number abv...	[last, chemo, today, just, finished, last, num...	[(last, JJ), (chemo, NN), (today, NN), (just, ...	[(last, a), (chemo, n), (today, n), (just, r),...	[last, chemo, today, just, finish, last, numbe...	last chemo today just finish last number abvd ...
ER phobia?	Have any of your loved ones checked out of the...	7.0	['er', 'phobia', 'any', 'loved', 'numbers', 'c...	er phobia any loved numbers checked out hospit...	[er, phobia, any, loved, numbers, checked, out...	[(er, RB), (phobia, VBZ), (any, DT), (loved, J...	[(er, r), (phobia, v), (any, n), (loved, a), (...	[er, phobia, any, loved, number, check, out, h...	er phobia any loved number check out hospital ...
8 minutes into his appointment, another scene ...	At this point itâs just funny to me. lâm n...	1.0	['number', 'minutes', 'appointment', 'another'...	number minutes appointment another scene made ...	[number, minutes, appointment, another, scene,...	[(number, NN), (minutes, NNS), (appointment, V...	[(number, n), (minutes, n), (appointment, v), ...	[number, minute, appointment, another, scene, ...	number minute appointment another scene make p...

We were able to do EDA, analyze and visualize the topmost common words according to the frequency counts of each word. We also had to remove stop words such as ‘like’, ‘go’, etc which did not add value to our classification.







LDA topics:	Interpreted/Suggested Topics
Topic 0: node lymph xb pet lymphoma scan neck prayer show hodgkin	Topic 0: Newly discovered
Topic 1: year love know time life would cancer one go much	Topic 1: Chemo Treatement Experience
Topic 2: pain take eat chemo stop sleep say even start day	Topic 2: Post treatment Care
Topic 3: go get day say mom home could see week hospital	Topic 3: Expressing Frustration/ Hopelessness
Topic 4: cancer get year chemo go month treatment week surgery back	Topic 4: Surgery?
Topic 5: want feel go like know think get try time really	Topic 5: Guilt and loneliness
Topic 6: fuck get suck cancer husband bear shit god like run	Topic 6: Losing battle/ Treatment ineffective
Topic 7: cancer support thank help people patient please post share one	Topic 7: Want to reach out
Topic 8: he get chemo round currently stage try hour cancer watch	Topic 8: Passing away
Topic 9: work take wife pill need would not blood doctor can	Topic 9: Remission?

This dataset was then given to six different machine learning models to classify the topic of the post. The Linear SVC model with TFIDF vectorization performed better than the other models. Our accuracy comparison of the different models is shown below.

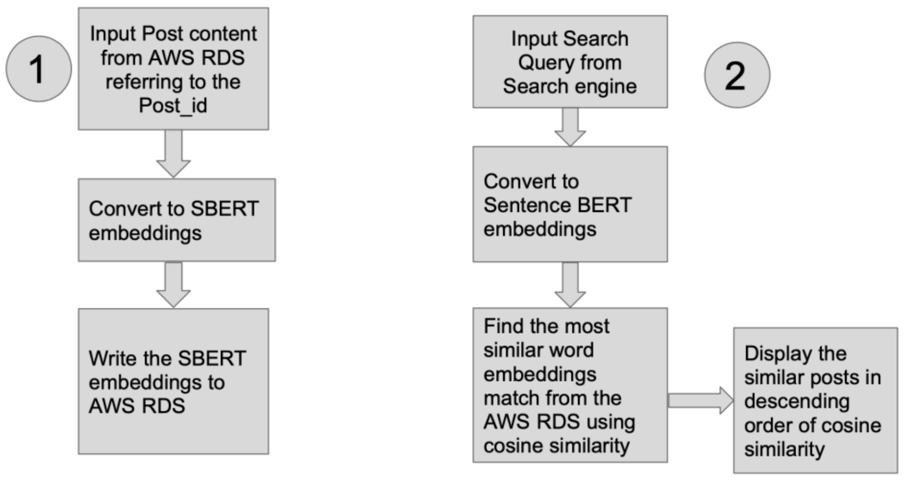
```
model_name
LinearSVC                 62.522124
LogisticRegression        57.368837
MultinomialNB             55.772756
RandomForestClassifier    55.772756
SVC                       61.281606
```

Module 2 - Semantic Search Engine

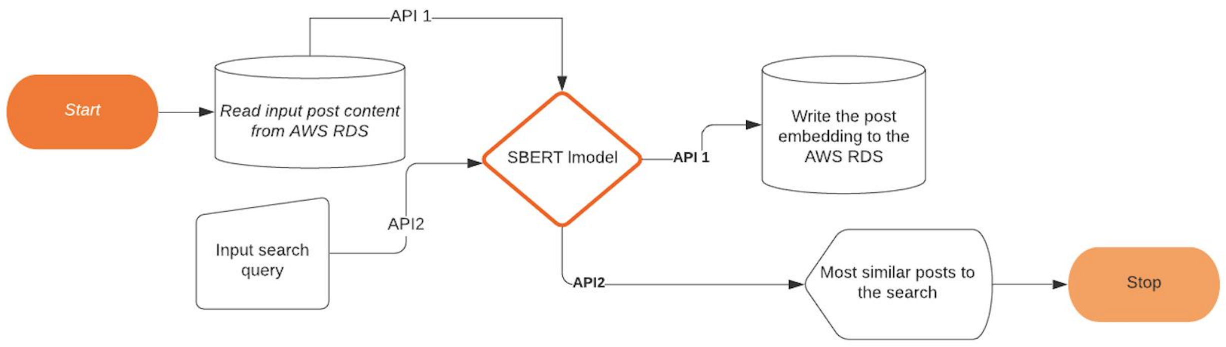
Since the SBERT model was successful and the Elasticsearch model had some complications and required more research and analysis before development, we had to finalize with the SBERT model due to time constraints.

Sentence BERT or SBERT model:

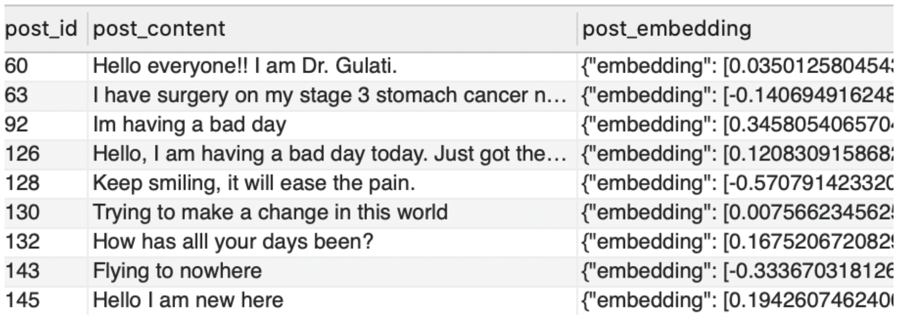
Our plan for the SBERT microservice module is to create two RESTFul APIs as belows:


1
Input Post content from AWS RDS referring to the Post_id
Convert to SBERT embeddings
Write the SBERT embeddings to AWS RDS
2
Input Search Query from Search engine
Convert to Sentence BERT embeddings
Find the most similar word embeddings match from the AWS RDS using cosine similarity
Display the similar posts in descending order of cosine similarity

A detailed flowchart of the module and the two APIs is below:

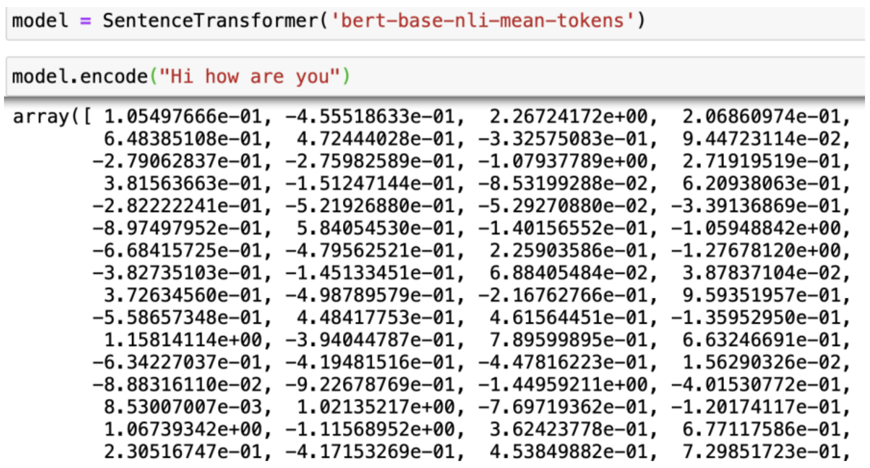

Start
Read input post content from AWS RDS
API 1
SBERT lmodel
API 1
Write the post embedding to the AWS RDS
Input search query
API2
API2
Most similar posts to the search
Stop

We were able to connect to the AWS RDS database using mysql connector python library. We then had to convert every input post. After calling our API 1, we will be able to generate a post_embedding column for each post.



post_id	post_content	post_embedding
60	Hello everyone!! I am Dr. Gulati.	{"embedding": [0.03501258045
63	I have surgery on my stage 3 stomach cancer n…	{"embedding": [-0.14069491624
92	Im having a bad day	{"embedding": [0.34580540657
126	Hello, I am having a bad day today. Just got the…	{"embedding": [0.12083091586
128	Keep smiling, it will ease the pain.	{"embedding": [-0.57079142332
130	Trying to make a change in this world	{"embedding": [0.00756623456
132	How has alll your days been?	{"embedding": [0.16752067208
143	Flying to nowhere	{"embedding": [-0.33367031812
145	Hello I am new here	{"embedding": [0.19426074624

The SBERT embedding for a query looks something like this on applying the Sentence Transformer pre-trained model. The input sentence is converted to an array of numbers called a vector.

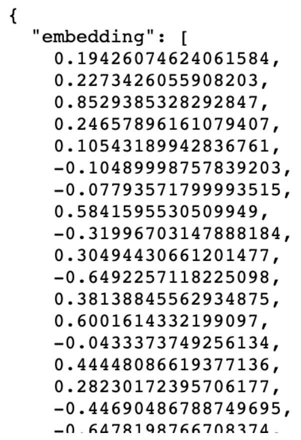


```
model = SentenceTransformer('bert-base-nli-mean-tokens')
```

```
model.encode("Hi how are you")
```

```
array([ 1.05497666e-01, -4.55518633e-01,  2.26724172e+00,  2.06860974e-01,
        6.48385108e-01,  4.72444028e-01, -3.32575083e-01,  9.44723114e-02,
       -2.79062837e-01, -2.75982589e-01, -1.07937789e+00,  2.71919519e-01,
        3.81563663e-01, -1.51247144e-01, -8.53199288e-02,  6.20938063e-01,
       -2.82222241e-01, -5.21926880e-01, -5.29270880e-02, -3.39136869e-01,
       -8.97497952e-01,  5.84054530e-01, -1.40156552e-01, -1.05948842e+00,
       -6.68415725e-01, -4.79562521e-01,  2.25903586e-01, -1.27678120e+00,
       -3.82735103e-01, -1.45133451e-01,  6.88405484e-02,  3.87837104e-02,
        3.72634560e-01, -4.98789579e-01, -2.16762766e-01,  9.59351957e-01,
       -5.58657348e-01,  4.48417753e-01,  4.61564451e-01, -1.35952950e-01,
        1.15814114e+00, -3.94044787e-01,  7.89599895e-01,  6.63246691e-01,
       -6.34227037e-01, -4.19481516e-01, -4.47816223e-01,  1.56290326e-02,
       -8.88316110e-02, -9.22678769e-01, -1.44959211e+00, -4.01530772e-01,
        8.53007007e-03,  1.02135217e+00, -7.69719362e-01, -1.20174117e-01,
        1.06739342e+00, -1.11568952e+00,  3.62423778e-01,  6.77117586e-01,
        2.30516747e-01, -4.17153269e-01,  4.53849882e-01,  7.29851723e-01,
```

Output of API 1:

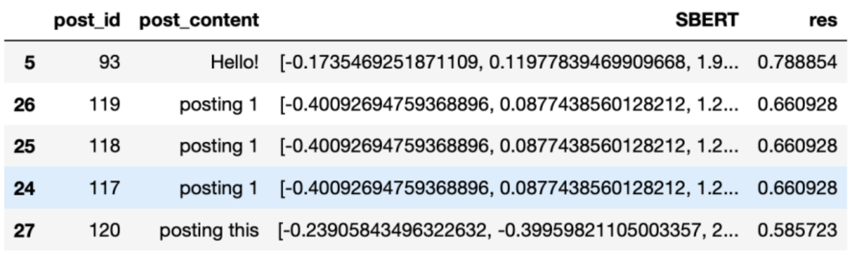


```
{
  "embedding": [
    0.19426074624061584,
    0.2273426055908203,
    0.8529385328292847,
    0.24657896161079407,
    0.10543189942836761,
    -0.10489998757839203,
    -0.07793571799993515,
    0.5841595530509949,
    -0.31996703147888184,
    0.30494430661201477,
    -0.6492257118225098,
    0.38138845562934875,
    0.6001614332199097,
    -0.0433373749256134,
    0.44448086619377136,
    0.28230172395706177,
    -0.44690486788749695,
```

The SBERT embeddings of the input post from the user is created as a result of calling our First RESTful API. These json formatted embeddings which is the output of API 1 are stored in the ‘post_embedding’ column of the AWS RDS database to use when comparing with the search query. This is done to avoid run-time complexity.

Output of API 2:

The second API compares the SBERT embeddings of the input search query whose similar posts are to be found with the ‘post_embedding’ column in the AWS RDS database and creates a ‘res’ column which has the cosine similarity. It then sorts the ‘res’ column in descending order.

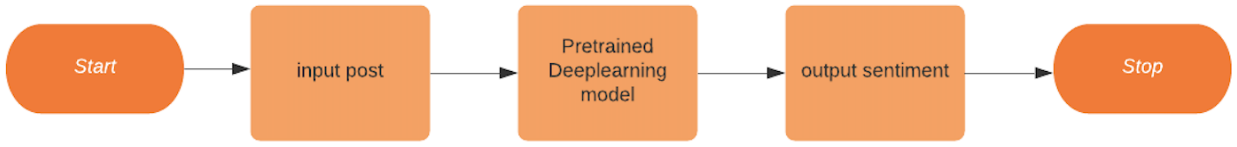


	post_id	post_content	SBERT	res
5	93	Hello!	[-0.1735469251871109, 0.11977839469909668, 1.9...	0.788854
26	119	posting 1	[-0.40092694759368896, 0.0877438560128212, 1.2...	0.660928
25	118	posting 1	[-0.40092694759368896, 0.0877438560128212, 1.2...	0.660928
24	117	posting 1	[-0.40092694759368896, 0.0877438560128212, 1.2...	0.660928
27	120	posting this	[-0.23905843496322632, -0.39959821105003357, 2...	0.585723

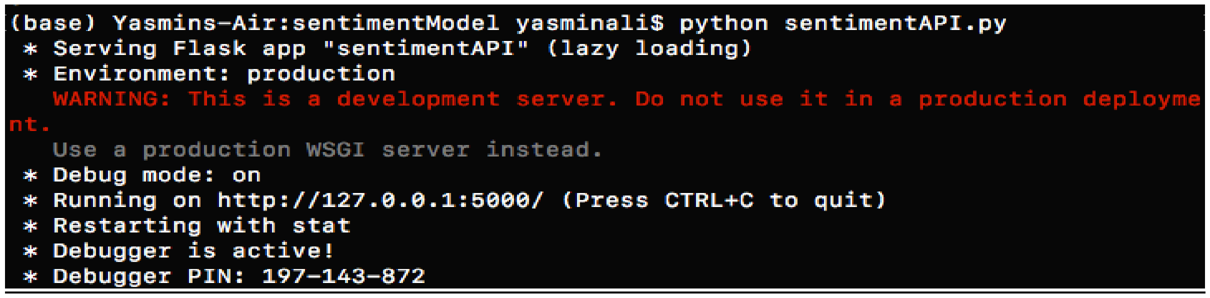
We then use the ‘res’ column from the temporary data frame created to sort the posts in descending order of similarity and display the results to the user. The JSON format of the results looks somewhat like the below.

```
{
  "posts": [
    "How has alll your days been?",
    "Hello I am new here",
    "Hello everyone!! I am Dr. Gulati.",
    "Trying to make a change in this world",
    "Im having a bad day",
    "Keep smiling, it will ease the pain. ",
    "Flying to nowhere",
    "Hello, I am having a bad day today. Just got the news of my relative havigng cancer. ",
    "I have surgery on my stage 3 stomach cancer next week. Feeling very anxious. Anyone have any advice, or any comment on how it will be?"
  ]
}
```

Module 3 - Sentiment Analysis

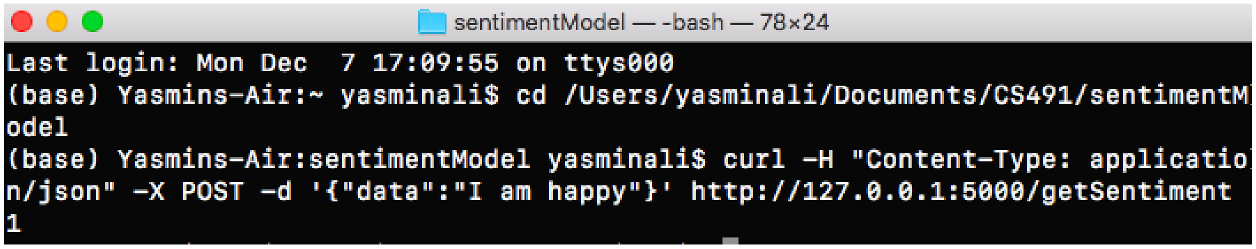


We created a RESTful API for this process and was able to successfully run the API using python command as below:



```
(base) Yasmins-Air:sentimentModel yasminali$ python sentimentAPI.py
 * Serving Flask app "sentimentAPI" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 197-143-872
```

We were able to verify the API route using curl function as below and manually entered our user post into the command to see our sentiment result as ‘1’.



```
Last login: Mon Dec  7 17:09:55 on ttys000
(base) Yasmins-Air:~ yasminali$ cd /Users/yasminali/Documents/CS491/sentimentM
odel
(base) Yasmins-Air:sentimentModel yasminali$ curl -H "Content-Type: applicatio
n/json" -X POST -d '{"data":"I am happy"}' http://127.0.0.1:5000/getSentiment
1
```

Challenges and problems encountered:

- Lack of enough resources, references and clear documentation available for development, because most of our requirements and ideas were to use new models.
- Most of us were not familiar with Deep learning and NLP concepts, so we required a lot of research and analysis before developing functionalities which took most of our time.
- We faced some technical difficulties with our laptops because we were working with Deep learning models, but were able to overcome them by using Google collab and AWS EC2 machines provided by our sponsor.

Chapter 6 Evaluation and Conclusion

Solution Testing

Our work is divided into four modules: topic classification, semantic search, Elastic search, and sentiment analysis. We used various test methods to assess whether our work functions as intended. For topic classification and sentiment analysis, we used accuracy measures. Specifically, after training the two models on training datasets, we ran the models on test datasets and measured accuracy of prediction. We achieved accuracy of 80% and 62% for sentiment model and topic classification, respectively. Accuracy measures helped us decide whether the model would yield accurate classification for user's posts. For the semantic search model, we used manual testing. We tested the results of our model by calculating the cosine similarity between the key words in our search query and the samples in our dataset. The model seems to perform very well. For elastic search we could not build the model due to some reasons that will be discussed later.

Verification and Validation

Our deliverables include three models, topic classification model, semantic search model and sentiment analysis model. Additionally, we created two API, a sentiment API and a semantic search API. The design and implementation of the models and the APIs meet the requirements mentioned by our sponsor.

Ideas for Improvement

We have ideas to improve each module. Firstly, performance of the topic classification module can be improved by using deep learning instead of machine learning, increasing the size of our training dataset, and labeling the data using semi-supervised learning instead of manual labelling providing more consistent labeling. Moreover, the sentiment analysis model can be better trained by adding more data to the training dataset. Finally, the semantic search module can be improved by adding more data since our test dataset included only 20 samples of data.

Team Conclusion

What did we learn?

- **Technical Skills**:
  We Learned techniques for preprocessing text data, while preparing them for the topic classification model. We also Learned how to work with pre-trained deep learning models for sentiment analysis in natural language processing libraries, such as sparkNLP and stanfordNLP. Our work on the semantic search module enabled us to learn new word embedding techniques, such as SBERT. We also did research and learned about elastic search.

In addition, we learned how to create REST API for our models. Finally, we learned to use the Jira tool for project management purposes.

- **Management and leadership Skills**:

We have learned leadership qualities that are just as important; teamwork, time management, communication and accountability are some examples.

1. Teamwork:

Each member handled a clearly defined task, but we felt responsible for the whole project. We frequently assisted each other by exchanging ideas and sharing experiences. Some tasks required collaboration among all team members. For instance, as a team, we manually labelled data needed for both topic classification and sentiment analysis models.

2. Time management and communication:

Team members met regularly to assure satisfactory progress of the project. We met weekly via Google Meet video calls to discuss milestones, achievements and next steps. We also used Slack to facilitate communications and exchange files. We used Jira time management tool to create the Gantt chart and track our tasks' timeline. Furthermore, we had bi-weekly meetings with our sponsor to review our results, discuss any shortcomings, and inspire ourselves to work harder

3. Accountability:

Even though we worked on this project during the COVID-19 pandemic, which caused lack of face-to-face communications among other obstacles, the team members made sure to communicate on a regular basis and persistently submitted the required work on time.

What would we do differently?

Although we achieved a big part of the work that was required from us, and that our sponsor is satisfied with our work and deliverables, there is always room for improvement. For instance, having had even a stronger understanding of the problem that we tried to solve and the goal we tried to achieve would have helped us do better especially pertaining to the elastic search model. This could have been done by communicating and asking more questions to our sponsor.

Were we successful?

We consider ourselves successful. We delivered three models and two APIs, a sentiment API and a semantic Search API, which are ready to be used immediately. We did the most tedious and exhaustive part in the classification model – that is labeling data, preprocessing lots of text data, and experimenting with several machine learning models. We learned a lot, and our collected effort paid back in our final presentation as we secured second place and received a certificate of accomplishment

References

1. B. Qiu et al., "Get Online Support, Feel Better -- Sentiment Analysis and Dynamics in an Online Cancer Survivor Community," 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing, Boston, MA, 2011, pp. 274-281, doi: 10.1109/PASSAT/SocialCom.2011.127
2. Richard Socher, Alex Perelygin, JeanWu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing, pages 1631–1642.
3. Nils Reimers and Iryna Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”, ACL, 2019, https://arxiv.org/abs/1908.10084v1
4. https://medium.com/@evergreenllc2020/semantic-search-engine-with-s-abbfb3cd9377
5. SparkNLP documentation: https://nlp.johnsnowlabs.com/docs/en/annotators#sentimentdetector-sentiment-analysis
6. Unknown, A. (2020, October 27). Functional vs Non-Functional Requirements: The Definitive Guide. Retrieved December 03, 2020, from https://qracorp.com/functional-vs-non-functional-requirements/
7. Smith, L. W. (2000). Stakeholder analysis: a pivotal practice of successful projects. Paper presented at Project Management Institute Annual Seminars & Symposium, Houston, TX. Newtown Square, PA: Project Management Institute.
8. Whizible Editor. (2020, October 29). *Whizible Blog | Agile Scrum User Story Tips and Template*. Whizible | Integrated Project Management | With Guarantee. https://www.whizible.com/how-to-write-effective-agile-scrum-user-story/
9. Technical University Munich, & Goyal, S. (2007, August). *Feature Driven Development Agile Techniques for Project Management and Software Engineering*. https://csis.pace.edu/~marchese/CS616/Agile/FDD/fdd.pdf