Topic-wise Temporal Sentiment Analysis Of NHS COVID-19 Application's User Reviews

By

Amritya Pradeep

Student ID: 2319659

Supervisor: Dr Phillip Smith

A thesis submitted to the University of Birmingham

For the degree of MSc in Data Science

School of Computer Science

University of Birmingham

Birmingham, UK

September 2022

Table of Contents

Table of Figures

Abstract

The covid-19 virus has caused one of the worst pandemic in history. Various countries have taken the route of Contact Tracing to control the infection rate of Covid-19. To achieve this aim of contact tracing, several mobile applications have been developed in different countries. However, as these applications are downloaded by citizens of different countries and everyone carries their own perspective on the application, it becomes necessary to understand how people feel and think about using these contact tracing applications. Various topics can pop into the minds of people using these apps, such as performance, privacy, and the purpose of the applications. Also, people will carry either a positive, negative, or neutral sentiment associated with the topics they have in their minds regarding the contact tracing application they are using. To understand what topics people have in mind and how the sentiments associated with those topics changes with time, this paper will introduce a topic-wise temporal sentiment classification method. The focus of this paper will be on the NHS COVID-19 App, which is a contact tracing app used in the UK. To achieve this task, we will first train and test a sentiment classification model on a large-scale manually annotated dataset consisting of reviews of contact tracing apps from 46 countries. Then, we will scrape the user reviews of the NHS COVID-19 application from the Google Play Store app to create another dataset and extract topics from it. And then finally, we will conclude on the topic-wise temporal sentiment classification of the NHS COVD-19 app user reviews where, for user reviews under each topic, we create two trendlines, one trendline for the sum of positive sentiments per month and the other trendline shows the sum negative sentiment per month. The accuracy of 97.31% on the validation dataset and 94.06 % on testing confirm the feasibility of the sentiment classification model. Moreover, high accuracies of the sentiment classifier for different topics of the NHS COVID-19 APP user reviews will prove that this pipeline will give us a deeper insight into the minds of the people reviewing the NHS COVID-19 APP. For those topics where the sentiment classifier achieves low accuracy, the reasons will be discussed, and we will understand why those topics are more important than the ones for which we have achieved higher accuracies.

Keywords: NLP, Sentiment Analysis, Topic Extraction, Machine Learning, Deep Learning, BERT, BERTopic

Acknowledgement

I would like to thank and express my sincere gratitude to my supervisor Dr Phillip Smith for his guidance and constant support throughout the project. His mentorship and necessary advice gave the right direction to this project, and this project would not have been completed without him. I would also like to thank my parents and brother for their continued support, encouragement, and motivation. They are the source of all my inspiration and enthusiasm in my life, and I would not have been able to succeed without them.

I would also like to show my utmost respect to the University of Birmingham for giving me a chance to pursue MSc in Data Science and providing me with the necessary resources and support throughout my journey. I feel proud to be a student at this university, where I got to experience a world-class education and meet such intelligent and amazing people. I will always cherish this educational experience all my life.

CHAPTER 1: Overview

1.1 Introduction

The coronavirus pandemic has been one of the most life-changing events in people's life. Covid-19 virus' ability to spread so quickly and at the same time be life-threatening has negatively impacted countries worldwide financially. The UK has been one of the worst hit countries due to Corona Virus, among other countries. The whole style of going to the office for work was replaced by working from home. The ease with which coronavirus spreads forced people to stay at home and wear masks. Lockdowns were implemented to reduce the spread of coronavirus by the countries worldwide, which was a big blow to their economies.

All these protective measures, like lockdown, masks, etc., were quite successful in reducing the spread, but contact tracing was one of the most essential tool to control the virus's spread. Contact tracing means finding all the people a person infected by the coronavirus came in contact with. Without knowing who has coronavirus, treating the patient or giving proper medical attention is impossible. Because most people these days have smartphones, leveraging this technology was one of the best ideas to trace the contacts of Covid-19 Patients. Different countries implemented their own mobile applications for the purpose of contact tracing, and these applications were loaded with other features depending on the government policies of these countries. In the UK, the name of the application is the NHS COVID-19 app. As this application is going to be downloaded by many people, the risks associated with the application must be addressed. Also, it will be beneficial to understand the perspective of the person downloading the applications and consider it when developing such contact tracing applications. For instance, as governments promote these applications, people might think about government-related topics while using this application. To understand the topics and what people think, we should read the reviews of the people about these applications. But reading many reviews by a human can take a lot of time, and saving time is very crucial when it comes to life-threatening diseases like covid-19.

Today machine learning tools can help understand the mind of people using any smartphone application. Natural Language Processing, a subfield of machine learning, can be used to understand the common topics people are writing about in the reviews. NHS COVID-19 App is available on the Google Play store and App store, and many people have given their reviews on the application. The reviews carry not only common topics and themes but also a sentiment that might be positive, negative, or neutral. The concept of finding topics that a body of the text is discussing is known as Topic Extraction, and the concept of finding the kind of emotional tone of a text is called Sentiment classification/analysis. These topics and sentiments can be very useful in not only improving the features of the app. Also, not only do these topics and sentiments help in improving the NHS Covid-19 app, but it can help build similar applications in the future which might have the purpose of providing some medical service. To get a deeper understanding of the reviews of the people about the NHS COVID-19 App, finding the topic-wise temporal sentiment classification can provide a deep understanding of what the people are thinking about the app. For topic-wise temporal sentiment analysis, for user reviews under each topic, we create two trendlines (one for positive sentiment and the other for negative sentiment) where one trendline shows the sum of positive sentiments per month, and the other trendline shows the sum of negative sentiment per month. Lastly, the method that our paper will use to find the topics and sentiments can be utilized by other non-medical products to get very useful insights and better understand their customers.

As Topic Extraction and Sentiment Classification are independent fields in which a large amount of research is going on, most research is focused on either Topic Extraction or Sentiment Analysis without considering the potential of combining them. Although both concepts are very good at analyzing textual data and serve different purposes, combining Topic Extraction and Sentiment Analysis will give a much deeper understanding.

1.2 Motivation

This research will use Natural Language Processing (NLP) models to predict the Sentiments per Topic of the user reviews of the NHS COVID-19 App on the Google Play store. The reason the motivated to develop this project is as follows:

- There is very less information available on the user experience and the difficulties that people faced while using the NHS Covid-19 App. Finding the topics, sentiments and Sentiment per topic of the NHS COVID-19 App will help in analyzing the performance of the app and the perspective of the users of this app at a very deeper level. As this App is used in the UK, this project can give us an understanding of what citizens of the UK and people travelling in the UK think about this App.

- Most research treats Topic Extraction and Sentiment Analysis Separately. Topic extraction is a field of NLP focused on extracting topics that the text or a document focuses on. Sentiment analysis/classification is a field in which the goal is to find and identify the emotional tone of a body of a text. The Fields of Topic Extraction and Sentiment Analysis are independent fields in themselves and contain different methods to achieve their purpose. But very less research is available on combining the topic extraction and sentiment analysis fields to get the sentiments of a text per topic that is being discussed in the body of text. Although topic extraction and sentiment analysis serve different purposes, combining both will give a much deeper understanding of the text rather than directly applying the sentiment analysis.
- As the topics and the sentiments of the reviews of the NHS COVID-19 App will help in understanding the minds of its user, the results of this app can be used as a baseline for developing similar Apps in the future which will serve some medical needs of the people living in the UK.
- The combing topic extraction and sentiment analysis to perform topic-wise temporal sentiment analysis can be implemented on any textual data to understand that data which is much deeper and more insightful than performing topic extraction and sentiment analysis separately. This method can help companies in getting an excellent understanding of their customers.

1.3 Research Question

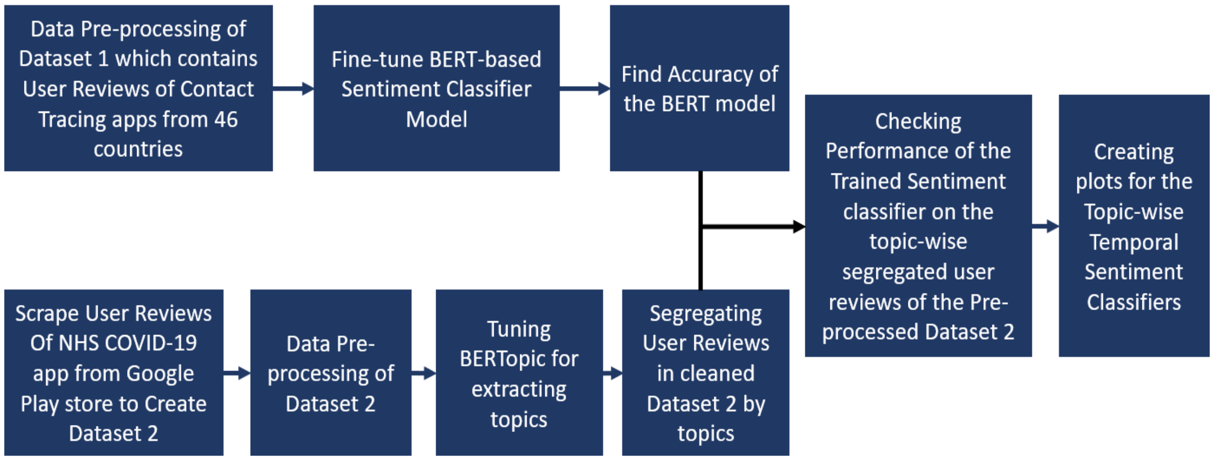
Natural Language Processing techniques, especially topic extraction and sentiment analysis, can help better understand the topics being discussed and the emotional tone of the user reviews of NHS Covid-19 data or any other text body in a better way. This project uses a topic extraction model and a sentiment analysis model to extract common topics, cluster the reviews according to their topics, predict whether the sentiment of the reviews is positive or negative, and create a trend of positive and negative sentiment for each topic per month. To analyse the reviews of the NHS COVID-19 app, the following research questions were considered:

- What are the commonly found topics in the reviews written for the NHS COVID-19 App in the Google Play Store?
- What hyperparameters will be good in finding commonly discussed topics?
- What should be the pipeline to create a Natural Language Processing (NLP) model that extracts all the important commonly discussed topics and the sentiments per topic from the NHS COVID-19 App?
- Is the sentiment analysis model achieving a good accuracy compared to other models?
- What are the trends of positive and negative sentiments of the user reviews of the NHS COVID-19 App available on the Google Play store for each topic that we extract?

1.4 Methodology

Figure 1 shows the flow chart of this project. This project is divided into two parts. The whole project has used python (programming language) and its libraries/packages to achieve all the tasks. In the first part, we train and test a sentiment classification model on a huge manually annotated data set that contains reviews of Covid-19 Contact tracing applications from 46 countries and a label that tells whether the review is positive or negative and then checks its performance by using a metric called as accuracy. Out of many available models, this project will focus on the state-of-the-art model called BERT (Bidirectional Encoder Representations from Transformers), which considers the context of the texts while finding the emotional tone of the text. The sentiment classification model will involve two steps. The first step will be to convert the text into a form the computer can understand. To achieve this, we convert the text into embedding vectors. In the second step, these embedding vectors will be used to train the model for sentiment classification. The Sentiment classification model will use BERT (Devlin, et al., 2019), a pre-trained language model from Natural Language Processing tasks such as sentiment classification, creating chatbots to answer your questions, predicting trailing words, summarizing the text etc. BERT has proven to be better than previously used models such as SVM and Naïve Bayes. In chapter 3, we will discuss how the BERT models take the embedding vectors and get trained. We will also learn how Bert models perform Sentiment classification. We will also learn about tokenizers that are used to create embedding vectors. We will also discuss why BERT is one of the best available models and is appropriate for this project.

The second part of this project starts with scraping the reviews of the NHS COVID-19 app (Coronavirus contact tracing application in the UK) along with other data related to the reviews such as ID number, Date (when the user review was posted), name of the reviewer, Rating (out of 5 where five is the highest and one is the lowest) given by the reviewer. Then scraped data goes through the Data-Preprocessing, which involves cleaning the data and converting the user reviews into a vector representation. After the data pre-processing, the next step is Topic Extraction. In topic Extraction, we use the state-of-the-art topic extraction model known as BERTopic (Grootendorst, 2022), which has shown to perform better than its predecessors such as LDA. BERTopic will be explained in much more detail in chapter 3. After finding the appropriate hyperparameters and tuning the BERTopic model, the next step is to divide the reviews according to the topic they belong. Then finally, we can load the trained sentiment classification model and perform topic-wise temporal sentiment classification. To achieve this, we calculate the monthly sum of positive and negative reviews for each topic and create a line plot for each topic where each graph has two trendlines, one representing the sum of positive sentiments per month and the other being the sum of negative sentiments per month. The X-axis is the months, and the Y-axis represents the sum of positive sentiments or negative sentiments. These plots will be discussed in detail in chapter 4.



Data Pre-processing of Dataset 1 which contains User Reviews of Contact Tracing apps from 46 countries
Fine-tune BERT-based Sentiment Classifier Model
Find Accuracy of the BERT model
Checking Performance of the Trained Sentiment classifier on the topic-wise segregated user reviews of the Pre-processed Dataset 2
Creating plots for the Topic-wise Temporal Sentiment Classifiers
Scrape User Reviews Of NHS COVID-19 app from Google Play store to Create Dataset 2
Data Pre-processing of Dataset 2
Tuning BERTopic for extracting topics
Segregating User Reviews in cleaned Dataset 2 by topics

Figure 1: Project flow Chart

CHAPTER 2: Literature Review

This section of the paper will discuss various machine learning and neural network-based algorithms used in previous studies related to Covid-19, Sentiment classification, and Topic Extraction.

Since the beginning of Covid-19, a lot of textual data has been generated from various sources such as social media, research papers and other sources of texts related to the coronavirus pandemic. This Generated data can provide various insights if explored properly and efficiently, and a great amount of research has been done in the field of NLP to explore the texts and documents concerned with COVID-19. To fight against the coronavirus pandemic, all the research communities in various fields such as NLP, Computer Vision, and healthcare have played a very big role. As a result, many great and interesting solutions focusing on vivid aspects of the pandemic have been proposed (Latif, et al., 2020). A lot of effort has been aimed at automatic diagnosis, prediction, and treatment (Gharavi, et al., 2020) (Qayyum, et al., 2021). Also, a lot of effort has been made to detect COVID-19 early to help prepare for emergency response (Gharavi, et al., 2020).

Next, the research community put a lot of focus on detecting fake news, assessing the risk, planning the logistics and understanding the social intervention, primarily monitoring social distancing (Latif, et al., 2020) (Bang, et al., 2021). In this pandemic era, social media is playing a very big role in keeping people connected and well-informed (Nabity-Grover, et al., 2020). Many mainstream social media platforms saw a sudden spike amongst English speakers since the beginning of this Covid-19 Pandemic (Wiederhold, 2020). Sentient Analysis analysis or opinion mining using social media gives the ability to find insights and trends in public discussion. Twitter, which provides a user-friendly API and small, focused messages known as Tweets, are often used for such tasks (Medhat, et al., 2014) (Giachanou & Crestani, 2016).

The researcher in (Antypas, et al., 2021) took the challenge of finding covid-19 misinformation-related tweets from the social media platform called Twitter. As told by the researcher, “The corpus was derived from two sources of Twitter data for the English language with a misinformation-related corpus collected via the Social Media analysis platform Sentinel (Preece, et al., 2018) and a corpus of random tweets (‘generic’) for the same period. The tweets were tracked and selected using a list of keywords related to the pandemic1”. Both the sets (‘misinformation related and ‘generic’) were balanced following the same distribution: 8911 tweets from January, 596 from Feb, 411,412 from March and 20,434 from April. Then they tried to find differences in the Generic corpus's lexical features and misinformation-related corpus. After that, they tried to test whether models could retrieve misinformation-related content using the lexical feature only. They used three lexical features: 1) Frequency Features based on TF-IDF;2) Semantic-based on the average of word embeddings within the tweet; and 3) the extra-linguistic features, namely tokens, hashtags, emojis, punctuations, @ and exclamations. They compared a simple linear model, namely Naïve Bayes and SVM (Support Vector Machine) (Hearst, et al., 1998), with transformer (Vaswani, et al., 2017) based models such as BERT (Devlin, et al., 2019) and deep learning-based models, namely CNN. They found BERT to outperform Naïve Bayes and SVM for the misinformation corpus and the generic corpus on a single year of data with an accuracy of 91%. But when the models were compared every month, BERT-based models outperformed the SVM with a small margin where the accuracy of Bert was 89 %, and SVM had an accuracy of 84 %, which indicates that transformer-based models are more robust but, in some cases, SVM model might be more beneficial due to low processing power requirements. Additionally, they were able to demonstrate that there is a distinct difference in the general language used in each type of corpus and that misinformation-related tweets can be accurately retrieved using a straightforward linear classifier based on lexical features. They also provided evidence that misinformation focuses on the kind of vocabulary that does not reflect the general distribution of what can be found on the public social media content for a particular topic.

The researchers in (Babu & Eswari, 2020) presented models for WNUT 2020 shared task2. As written in the paper, “The shared task2 involves identification of COVID-19 related informative tweets.” They focused on a binary text classification problem and experimented with the pretrained language models CT-BERT (Covid-Twitter-BERT) and RoBERTa (Liu, et al., 2019). They also said that before 2018, most text classification models were based on CNN (Convolutional Neural Network) and RNN (Recurrent Neural Network), and such models were shallow and could not learn more informative features from the input. Also, these models had to be trained from scratch, which required a greater number of training instances (Kalyan & Sangeetha, 2021) (Kalyan & Sangeetha, 2020). But Pre-trained language models have shown great success for classification tasks. In (Babu & Eswari, 2020), researchers created a dataset of 20,000 Tweets from the social media platform called Twitter, and each tweet was labelled 0 (uninformative tweet) or 1 (informative tweet). They divided this data set into train, validation, and test sets. Their model 1, which included the model CT-BERT, is initialized from BERT-Large weights and is then further pre-trained on 160 million tweets related to Corona Virus. They used this task to perform a Binary classification task and showed that pretrained language models such as CT-BERT and RoBERTA are excellent for the purpose of binary classification tasks. Our project will be performing sentiment classification to predict sentiment if the user reviews are positive or negative, which is a Binary classification task, and Pre-trained language models work very well on it.

During the COVID-19 pandemic, contact tracing has also become widely explored in different research communities. For example, (Lash, et al., 2020) analysed and explored the mechanism and final results of contact tracing in Two countries. The authors of this paper found that if an efficient mechanism of contact tracing is introduced, it can reduce the infection rate of the virus significantly. But, as we know, this contact tracing has several challenges and difficulties related to the timely and accurate tracing of the contacts of COVID-19-positive patients. So to resolve this issue, a combined effort from the research community and the use of more state-of-the-art and advanced methods that use currently available technologies such as Wi-Fi, Bluetooth, GPS (Global Positioning System), Cell phone tracking etc., could help a lot (Mbunge, 2020) (Lalmuanawma, et al., 2020). People carry devices with them daily; mobile phones, which already have most of these technologies embedded in them, seem like an ideal platform for creating solutions for contact tracing. That is why many countries have developed and introduced mobile applications for contact tracing. In addition to these mobile applications, other features have also been implemented based on the COVID-19 policies of a particular country (Lalmuanawma, et al., 2020). For example, applications have been used in countries such as Australia and Qatar to assess different facilities.

Although these mobile applications are a feasible and reasonable solution to battle the pandemic, they have also gathered much criticism due to their multiple issues. There are a lot of issues that have been reported which include data privacy, government control, increased power consumption, wastage of tax money, and irritating mobile notifications and alerts. In the paper (Bengio, et al., 2021), the author reported various privacy issues linked with the COVID-19 Contact tracing applications. To ensure user privacy, some researchers and authors also proposed a design for contact tracing apps that involve decentralisation by optimizing the trade-offs between user privacy and the application's utility. In (Bengio, et al., 2021), the author explored and analysed the privacy-related concerns of the applications and proposed a mechanism which is dependent on the privacy-preserving protocol known as the MPC (multi-party computation) (Reichert, et al., 2020), which will help in privacy preservation. Mobile phone batteries' high power consumption is another key issue associated with these COVID-19 contact tracing applications.

There are several impressive works where the viability of the contact tracing application is checked by analysing the feedback of the people using these applications (Ahmed, et al., 2020) (Li & Guo, 2020) (Cho, et al., 2020). For example, to get people’s feedback on HSE (Contact Tracing application in Ireland), an online survey was done (O’Callaghan, et al., 2020). In this survey, a decent percentage of people showed interest in using the contact tracing application. However, the survey’s focus was primarily on analysing and identifying the different obstacles in using such applications, which excluded the user’s experience of the apps.

To understand the user’s experience in a better way on these COVID-19 contact tracing applications, a detailed and thorough analysis of the public user reviews must be done, which is available on the Apple Store (Apple, 2008) and Google play store (Google, 2012). In the literature, there are already some efforts toward this path. For example, (Rekanar, et al., 2020) provided a thorough analysis of the users’ feedback on HSE, focusing on usability, functionality, and performance. But the authors performed manual analysis only, which is a tedious and time-consuming process and prone to more human errors. Also, (Garousi, et al., 2020) is another fantastic work where the authors have done an exploratory analysis of the user’s feedback in nine COVID-19 contact tracing applications. But they relied solely on the commercial app-reviewing analytics tool known as the Appbot. They used Appbot to extract the user reviews on these COVID-19 contact tracing applications. But they didn’t create any benchmark dataset or use advanced state-of-the-art methods such as machine learning to make the reviewing process automatic. Most of the available literature focused on a single or few COVID-19 applications focusing on specific regions, which is helpful to give insights about that region but does not provide a general overview of the matter. But the researcher in the paper (Antypas, et al., 2021) analysed and explored data from 46 countries.

Authors in (Antypas, et al., 2021) gathered user reviews of COVID-19 contact tracing applications from 46 countries and used various models to classify the review's sentiment. Authors in (Antypas, et al., 2021) managed to create a benchmark Data set for Reviews of Covid-19 Contact Tracing Apps and Perform Sentiment Analysis using different machine learning models and compare the performance. This Data set comprised 34,534 user reviews of Contact tracing applications from 46 countries and their sentiment value. The reviews were categorised into four categories, positive, negative, neutral, and technical. This data was intended to cover three different tasks (Antypas, et al., 2021). And for these three tasks, to facilitate the potential users of the dataset, they also provided the task-wise distribution of reviews for each task in a separate folder (Ahmad, et al., 2021). They used classical machine learning algorithms as well as deep learning algorithms. In classical algorithms, the authors used Multinomial Naive Bayes (MNB), SVM and Random Forest. For feature representation with these algorithms, the authors utilised bag-of-n-grams. In Deep Learning, the authors fastText (an NLP library aiming at efficient word embedding and text classification at a higher speed than the traditional deep learning) and Transformers. In the Transformers category, the authors used various types such as BERT, RoBERTa, XLM-RoBERTa (Conneau, et al., 2020) and DistilBERT (Sanh, et al., 2020). This paper also showed that Transformer based models (like BERT) perform better than classical machine learning models when performing sentiment analysis-related tasks. But it didn’t perform topical or temporal analyses and left it as a critical aspect to be analysed in the future.

(Sanders, et al., 2021) is a good paper in which the authors scrutinized a database of over 1 million tweets collected from march 2020 to July 2020, when the infection rates spiked in the USA, Europe and other parts of the world, to analyse and explore the public attitude towards the usage of mask in the COVID-19 pandemic. To achieve this goal, researchers’ gathered posts from Twitter (Tweets) related to Covid-19. Then the researchers used Natural Language Processing methods, namely Topic Modelling and Sentiment classification, to organise tweets relating to mask-wearing into high-level themes and then give a narrative of each theme using automatic text summarisation. Topical modelling can be understood as the extraction of topics from the text. They performed Topical Modelling using k-means clustering to Segregate tweets into 15 high-level themes and 15 specific topics within each theme and performed sentiment analysis on the entire corpus of data and on each theme and topic in the five months. To perform sentiment analysis, the researchers converted the tweets into a vector using the transformer implementation of Google’s Universal Encoder (Cer, et al., 2018). The vectors representing each tweet were given by a sum of contextual word representations at each position of the output of the transformer encoder. Tweets that are similar in terms of their semantics are grouped in the resulting embedding space; there, the cosine similarity tells the metric of how close the tweets are. Then to get the sentiment, they used a rule-based method called VADER (Hutto & Gilbert, 2014) (Valence Aware Dictionary for sentiment reasoning is a social media-focused, lexicon-based characterisation approach) which, although very good in detecting sentiments, ignores the context of the text. To cluster and create sub-clusters of themes and topics, the researcher applied k-means in the embedding space to create a 2-level hierarchy - the corpus is grouped into 15 primary clusters, and each primary cluster is further grouped into 15 sub-clusters. Finally, the researcher applied the abstractive text summarization model using NLP to interpret and explain the subject of the conversation within each theme and cluster. Then the researcher created insightful data visualisations and statistical analyses to examine sentiment trends.

In the papers discussed till now in the literature, we have already got a glimpse of topic modelling to extract topics related to COVID-19-related terms using clustering methods, like K-means Clustering (Sanders, et al., 2021). Researchers from many communities, especially in the field of NLP, applied different topic modelling techniques and algorithms to analyse and explore texts and documents. In General, Latent Dirichlet (LDA) and Non-Negative Matrix Factorization are two of the most popular topic modelling techniques. LDA uses a statistical model which uses a probabilistic approach. In comparison, NMF uses a matrix factorization approach. Most researchers focused on a statistical method called LDA (Latent Dirichlet Allocation) to uncover the topics discussed around the coronavirus pandemic. For instance, in (Oyebode, et al., 2022), the researchers analysed the COVID-19-related comments collected from six social media platforms using NLP Techniques. They found relevant key phrases and their sentiment polarity (negative/positive) from more than 1 million randomly selected comments. Then they categorised them into a broader theme using thematic analysis. The researcher successfully uncovered 34 negative themes related to economic, political, and educational issues. They also identified 20 positive themes. Finally, they suggested interventions to tackle them based on the positive themes and evidence. To extract themes, the authors utilised a context-aware NLP approach. This method was a development of the one used by Dave and Varma (Dave & , 2010) and included enhanced POS (Part-of-Speech) patterns adapted to their goal, followed by chunking (in conjunction with CoNLL IOB tagging (Sang & Meulder, 2003)), transformation, and sentiment scoring stages. To extract the sentiment of the key phrase, the authors have used a lexicon-based algorithm called VADER. In the literature, we found another paper that explored social media and focused on neutral and controversial terms for the Covid-19 pandemic (Chen, et al., 2020). (Xue, et al., 2020) is another excellent paper in the literature where the authors used the LDA for topic modelling on the Twitter data to study the users' discussion and psychological reaction to COVID-19. They collected about 1.9 million English-written Tweets related to the coronavirus pandemic. They used LDA to extract topics and themes and performed sentiment analysis to find the emotion in the tweets: fear, anger, anticipation, disgust, joy, sadness, surprise, and trust. To see which emotions were prevalent in a certain topic, the researchers used a statistical approach called the z-test and assessed if each of the eight emotions were statistically significant across topics and found exciting results.

Topic modelling is not limited to research coronavirus-related text and documents but other medical fields as well. A lexicon-based Natural Language Processing (NLP) technique was used to find the prevalence of essential keywords suggesting public interest in e-cigarettes, marijuana, Ebola and influenza using the data from social media, and LDA was used to extract topics from it (Park & Conway, 2017). Also, a machine learning-based NLP technique was utilised to analyse clinical notes to predict hospital readmissions for COPD patients (Agarwal, et al., 2018) and perform sentiment analysis on mental health applications' user comments (Oyebode, et al., 2020).

In recent years more research has been focused on using neural topic models, which have shown high success in leveraging neural networks to improve topic modelling techniques (Terragni, et al., 2021). The incorporation of word embeddings into classical models like LDA has demonstrated a possible use of these powerful representations (Nguyen, et al., 2015). Pre-trained language models have become more widely popular. CTM demonstrates the benefit of relying on pre-trained language models, suggesting improvements in the language models for better topic modelling (Bianchi, et al., n.d.). Several approaches have started using clustering word embeddings and document embedding, simplifying the topic-building process (Sia, et al., 2020). However, a new technique, BERTopic (Grootendorst, 2022), builds on the clustering embeddings approach and extends it by using a Pre-trained Language model to create embedding representation and incorporating a class-based variant of TF-IDF for creating topic representations. The author of the paper (Grootendorst, 2022), Maarten Grootendorst, introduced BERTopic (a Neural topic modelling with a class-based TF-IDF procedure) which takes the context of the text into account. The BERTopic has shown great potential and has shown to perform better than Classical techniques such as Latent Dirichlet allocation (LDA) and Non-Negative Matrix Factorization (NMF). In our paper, we will utilise the power of BERTopic, and Bert-based sentiment classifier to create a topic-wise sentiment analysis model on the NHS-COVID-19 app, which is a contact tracing app for tracing contacts of Covid-19 patients in the UK.

CHAPTER 3: Background

Introduction

This section will introduce some essential algorithms and terminologies that have been considered while building this project. First, we will begin discussing pre-processing techniques to clean and prepare our data for further analysis. The following section will explain feature representation and weighing approaches that convert words into computable vectors. After that, we will discuss popular classification models, their working, and their benefits/limitations. Then we will list some validation techniques that help avoid overfitting and discuss some methods for evaluating the performance.

3.1 Data Pre-processing

Data pre-processing steps are essential for sentiment classification and topic extraction to ensure that only the relevant part of the data is used in the process. Some of the widely used techniques for cleaning the data and pre-processing are as follows:

- Tokenisation: As explained by the authors in (Trieschnigg, et al., 2007), "Tokenisation is the process of converting a stream of characters into a stream of words or tokens". Document tokenization also notes whether these tokens are punctuation, letters, or numbers.
- Remove stop words and punctuation: Usually, punctuation and a list of stop words are deleted from the documents as these occur in high frequency and have no sentimental meaning. These words can harm the calculations of the terms' weight which will be explained in the following sections. But the new models based on Transformers, such as BERT, don't require the removal of stop words and have shown a better performance than the previously used models using classical machine learning models such as Naïve Bayes.
- Porter Stemmer: It is a method introduced by (Porter, 1980), and it is used to remove suffixes from words. For example, "running" will be stemmed to "run", "having" will be stemmed to "have", and words like "generous" and "general" have the same stem "gener". As seen from the example, these generated stem does not need to be correct in English.

3.2 Feature Representation Methods

Text documents are a sequence of text having varied lengths, and such data needs to be represented as vectors (features). This process is also known as feature extraction. The following sub-sections will introduce some of the word representation, feature extraction and weighing methods that were considered when building this project and will decide which way will be good for this project.

3.2.1 N-gram

The N in the n-gram represents the number of words to be linked together in a text document (Jurafsky & Martin, 2014). N=1 represents unigram, n=2 represents bi-gram, and n=3 represents a trigram. For instance, for n=2, two words linked could be "New York" or "The new", and for n=3, the 3 words will be grouped together, such as "The New York". This method can be used to calculate the probabilities of words likely to be together and words which are unlikely to be linked.

3.2.2 Bag of Words

One of the simplest techniques for feature representation is the bag-of-words model. A text (such a sentence or a document) is modelled in this theory as a bag (or multiset) of its words. Although the word order and grammar are ignored, the multiplicity is preserved. The frequency of each word is employed as a feature in this approach, which is frequently used in document categorization tasks (Jurafsky & Martin, 2014). The Bag of words involves two steps. To understand them, we will try to create a feature or vector representation of the following:

- the bird sat

- the bird sat on the tree
- the bird sat near the tree

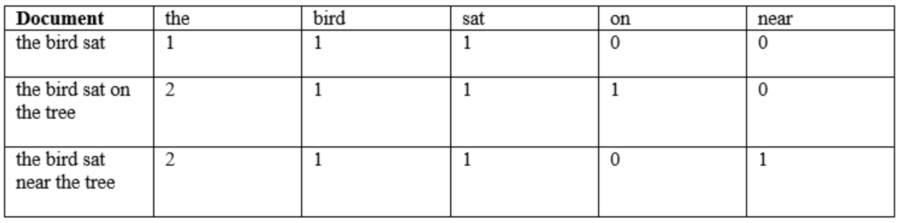
Each of the above sentences will be referred to as a text document

Step 1: Determine the Vocabulary

The vocabulary is nothing but the words found in the document set. The words found in the three documents above are: the bird, sat, on, tree, and near.

Step 2: Count

To vectorize the documents, the only thing we need to do is count the number of times each word appears:



Document	the	bird	sat	on	near
the bird sat	1	1	1	0	0
the bird sat on the tree	2	1	1	1	0
the bird sat near the tree	2	1	1	0	1

Figure 2: Bag Of Words Representation of documents

So, the vectors for each document are as follows:

- the bird sat: [1,1,1,0,0]
- the bird sat on the tree: [2,1,1,1,0]
- the bird sat near the tree: [2,1,1,0,1]

3.3 Feature weighting methods

We learned how to represent terms without giving their terms more weight in the previous section. This section will be devoted to talking about strategies for giving the key terms in the papers more significance. Assume that you have a collection of documents D = {d^1, d^2, …., d^{m-1}, d^m} that include words T = {t_1, t_2, …, t_{n-1}, t_n}, and you need to organise the terms into a collection of classes or labels L = {l^1, l^2, …, l^{p-1}, l^p}. The approaches that we thought about using for this project to convert text documents to weighted vectors will be covered in the next subsection. The importance or weight of the words will be shown as w_k^i, where i is the document and k is the word (López, et al., 2007).

3.3.1 Term Frequency

This method helps in a vector representation, where each item represents the weight equal to the number of times a term t_k was used in a document (Paltoglou & Thelwall, 2010).

$$w_k^i = tf_k^i \tag{3.1}$$

where,

tf_k^i = (Number of times term t appears in a document) / (Total number of terms in the document)

3.3.2 Term Frequency Inverse Document Frequency (TF-IDF)

This method was given by (Salton, et al., 1975)**,** and it is a widely used method. It is used to calculate the weight of the word/term t_k in the document d^i using the formula

$$w_k^i = tf_k^i \times log\frac{N}{df_k} \tag{3.2}$$

Where N = Total number of documents and df_k = number of documents containing the term t_k .

The benefit of this method is that it ranks the words by their importance in the document. But this method ignores the context in which the word is being used in the text.

3.3.3 Word2Vec

Word2vec (Mikolov, et al., 2013), as quoted on Wikipedia, "uses a neural network model to learn word associations from a corpus of text" (Wikipedia, n.d.). They effectiveness of word2vec is shown by its ability to group vectors of similar words. The word2vec model is explained wonderfully in (P., 2021). But one limitation of word2vec is that it ignores the context in which a particular term is used. Its vector representation of a word will always be the same regardless of where the word is in the documents and has a different meaning in different contexts. BERT Embeddings resolve this issue.

3.3.4 BERT Embeddings

A pre-trained language model called BERT (Bidirectional Encoder Representations from Transformers) seeks to comprehend ambiguous semantic relations in the text by inferring the context from the surrounding text. This model can be refined and fine-tuned with much less effort on new datasets to be used for various applications because it was pre-trained on Wikipedia Text. The process of pretraining and fine-tuning is shown in figure 6 and will be discussed in section 3.4.1.2.3. It is a Transformer-based model developed by Devlin and colleagues (Devlin, et al., 2019)**.** Before the BERT model is trained, BERT has its own approach, which will be covered in this part, for turning text into a vector representation.

3.3.4.1 Input and Output Embeddings

BERT embeds input tokens in a vector representation using a token embedding layer, segment embeddings, and position embeddings. Figure 3 below shows the illustration of the three embedding layers.

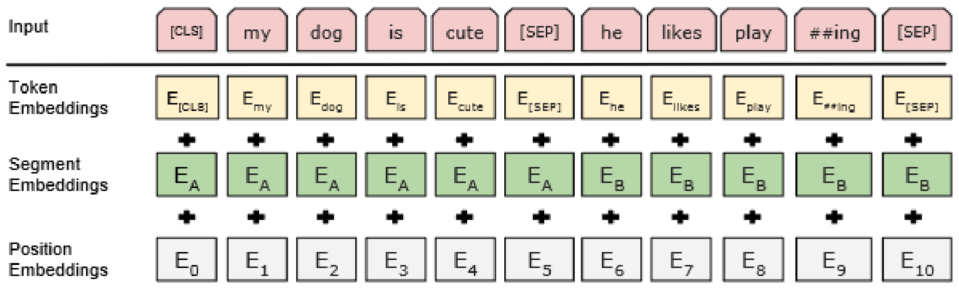

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 3: Embedding Layer structure of BERT to convert Input text to vectors (Devlin, et al., 2019)

To make an input embedding, these three layers are added together. Before the input is processed by the embedding layers, WordPiece is used to tokenize the input text.

3.3.4.2 WordPiece

BERT uses WordPiece tokenization method proposed by (Wu, et al., 2016). WordPiece was intended to improve the handling of uncommon (rare words) words. To achieve this, words were sliced into sub-words called as WordPieces by using a pre-defined Vocabulary dataset. The size of the BERT paper, its vocabulary set is 30,000-word pieces. When the rarity of a word increases, it is divided into more minor individual characters. And a prefix ## is added to the sub words. For instance, "hugs" might be broken down into "hug" and "##s", and "hugging" might be broken to "hug" and "##ing". "Hug" is closely related to "hugs" and "hugging", so both words are also linked together.

3.4 Machine Learning

As mentioned in (Dangeti, 2017), “Machine learning is a branch of study in which a model can learn automatically from the experiences based on data without exclusively being modelled like in statistical models. Over a period and with more data, model predictions will become better.” We use machine learning to predict the future using past data. Machine learning works by detecting similar patterns and classifying them, thus enhancing the process of decision-making (Dangeti, 2017). We will also talk about the machine learning subfield known as deep learning (LeCun, et al., 2015). Deep Learning algorithms are an extension of machine learning that is mathematically more difficult. Deep learning's goal is to develop mathematical models of the human brain that will enable computers to reason quickly and accurately, much as humans do.

Several different types of machine learning models can be grouped according to scenarios, where these scenarios might differ in the type of training data that the learner has access to, the method for training on the available data, and the test data used to assess the learning algorithm. Learning methods include active learning, reinforcement learning, transductive inference, supervised learning, unsupervised learning, and semi-supervised learning. A detailed explanation of these types of learning is available in (Mohri, et al., 2018). Based on the Data Available for this project and the nature of this project, our focus was on the Supervised and Unsupervised Learning models.

In Supervised learning, the machine learning model is given a set of labelled training data. Each Record in the data has a bunch of features and is assigned a label (or class). Once the modelled is trained to predict, we can input new features into the model, and the model will output one of the labels as the expected output, which was present in the training set. Supervised learning can be used for both classification and regression problems (Mohri, et al., 2018). In classification problems, we have discrete categorical labels such as ‘High’ and ‘Low’. As sentiment classification is a classification problem, supervised learning methods will be the best for this project. In our project, for the purpose of sentiment classification, we will consider supervised learning methods. Models that come under supervised learning are Naïve Bayes, SVM, RNN, BERT, etc.

In Unsupervised learning, the training dataset has input features for each record, but there is no label (or class) for each record. Labels are not there because of unavailability, or it is too costly to retrieve them (Mohri, et al., 2018). The dataset we will be training on for sentiment classification will have labels, so we don’t need unsupervised learning. But for Topic Extraction, Unsupervised learning can be very useful when the labels are unavailable. The explanation of the Dataset we will use for the project will be given in detail in the next section. Models that come under unsupervised learning are K-means Clustering, HDBSCAN, etc.

Before we discuss some models that were considered for building this project, let’s look at the steps involved in developing a machine learning model based on the paper (Mohri, et al., 2018).

Finding the issue is the first step. This step entails comprehending the problem at hand and formulating the pertinent queries that the machine learning model will need to respond to. Data gathering and exploration are the next two steps. At this stage, our goal is to gather the information that will aid in resolving the identified issue. Data might be from one source or many, and it can also be in different formats (e.g., text, numbers, video, images etc.). It will be necessary to transform the collected data into a format that the machine learning model can use. If the problem to be dealt with is related to NLP (Natural Language Processing), the text data in NLP is always transformed into a feature representation (vectors). Some of the methods to convert text into vectors are already shown in sections 3.2 and 3.3. To do so, data must be thoroughly examined to discover patterns. Data pre-processing is the third step that we take after collecting and analysing the data. At this stage, we clean the data and eliminate any data points (or noise) that won't aid in solving or providing the solution to the problem that has been found. The creation and training of the machine learning model is the fourth phase. Here, we choose the best model to train on a subset of the cleaned and gathered data known as the training data. The fifth phase, testing the model, comes after the model has been trained. Here, we test the model with test data that hasn't been seen before, and we use a variety of evaluation techniques to assess how well the model performed. Checking the accuracy to verify how many values the model can correctly categorise and how many it can't is an example of an evaluation technique. The final stage is to use the model to predict or categorise further data that will be received in the future.

3.4.1 Sentiment Classification models

Sentiment Analysis or classification is used to understand the emotional tone of the text document and understand the thoughts of the people about a particular subject. It can be used to classify the sentiment of a text as positive, neutral, or negative and to assign sentiments. Now we will discuss some machine learning models that were considered while building this project.

3.4.1.1 Classical Machine Learning Model

The most popular machine learning models are Logistic Regression, Naïve Bayes, SVM (Support Vector Machine), etc. In this subsection, we will summarise a few of these models and conclude why these methods are not the best for our project.

One of the supervised ML models used for classification tasks is logistic regression. It can be used for various classification tasks, including identifying spam emails and detecting fraudulent credit card transactions, among others. It is a binary classifier that will determine a probability function for the sigmoid function S, which ranges from 0 to 1.

$$S(\theta x) = \frac{1}{1 + e^{\theta x}} \tag{3.3}$$

Where θ is the weight parameter and S is the sigmoid function. The weight parameter can be set according to the dataset we have using a loss function. For Binary Classification, a threshold value of S is decided. For instance, in an email spam classification task, if a threshold value is 0.5, for S>0.5, the email can be considered spam, and for S<0.5, the threshold value is not spam (Han, et al., 2009). A perfect explanation of the Logistic Regression can be found in the paper (Han, et al., 2009), in which the author has created an improved version of Logistic Regression for email classification.

The next model we will discuss is Random Forest. To understand this model, we need the understanding how the decision tree algorithms. The decision tree algorithm is also a supervised learning model used for solving classification and regression tasks. Decision trees derive branch rules from features to predict the class labels. Splitting decisions are found using Gini Impurity (Singh, 2021). It is calculated as follows:

$$\text{Gini Impurity} = 1 - \text{Gini} \tag{3.4}$$

Where,

$$\text{Gini} - 1 - \sum_{i=1}^{C} P_i^2 \tag{3.5}$$

Where C = number of classes. Also, P_i denotes the fraction of points at P that belongs to class C.

A supervised learning algorithm that makes use of the Bayes Theorem is called Naive Bayes. It operates under the presumption that all the data's properties are unrelated to one another. Fitting the model to the data is likewise an iterative process and a loss function. Naive Bayes has an excellent explanation in (Rogers & Girolami, 2016), where the whole process of how the Bayes theorem works are explained in detail and derivation of Naïve Bayes is given.

All the classical machine learning models can give good results in sentiment classification and have their advantages and disadvantages, but these models are not the best fit for this project. The reason is that in the paper (Antypas, et al., 2021), BERT as proven to perform better than classical machine learning models, and we are using the same dataset to train our BERT model. Another reason is that in textual data, words are not independent of each other. Classical machine learning models fail to consider the complex semantic relationships between words when being trained.

Also another big issue with classical machine learning models is that they are not pretrained language models, and their weight must be trained and set from scratch. They need to be created from scratch, so they are not as efficient as the Deep learning models such as BERT, which are pretrained on a large corpus of data and only need fine tuning according to the task at hand. So, the Models such as BERT works better and have the ability to work on different bodies of text which maybe a little outside the domain they are being trained on. AS the are neural network-based models, they will perform better the more they are fine-tuned. We will discuss BERT in the 3.4.1.2.3 section.

This project aims to create a task-independent model that can be used for any covid-19 user reviews, surveys, and even other texts to classify sentiment. But classical machine learning models are task-dependent and can predict accurately only the type of data on which it was trained. Hence, classical machine learning models are unsuitable for creating a task-independent model.

Now let's discuss some models that were considered in this project. We will start with classical machine learning mode, then move to deep learning models, and finally conclude with which model will be the best for this project. We will discuss why classical machine learning is not suitable for this project. Then finally, we will conclude which deep learning model will be the best fit for our project.

3.4.1.2 Deep Learning Models

In this section, we will discuss the Deep Learning models (LeCun, et al., 2015) Considered for this project, namely RNN, LSTM and BERT.

3.4.1.2.1 Recurrent Neural Network (RNN)

Recently, neural networks have shown ground-breaking results in a variety of data classification applications. RNNs are a generalised form of machine learning classification. Due to their structure, RNNs can learn from data, but they heavily rely on the results of past data, such as those from the stock market and language processing. Figure 4 shows how a typical RNN is visualized (LeCun, et al., 2015). RNNs are built using a hidden state (s_t), which serves as that of the memory for neural networks, and an input state (x_t) at time t. The probability distribution is established using a softmax function, and an output layer then generates the output at time t (LeCun, et al., 2015). The calculation uses the same weights (U, W, and V) for each layer. The formulas below can be used to calculate the hidden state and output layer, where f can be a nonlinear equation like Tanh, Sigmoid, or ReLU in (Buduma, 2015) (Ho, et al., 2017).

$$s_t = f(Ux_t + Ws_{t-1}) \tag{3.6}$$

$$o_t = softmax(Vs_t) \tag{3.7}$$

After computing the projected output, RNN backpropagates to change the weights of each layer in order to determine the smallest lost value (Rumelhart, et al., 1986).

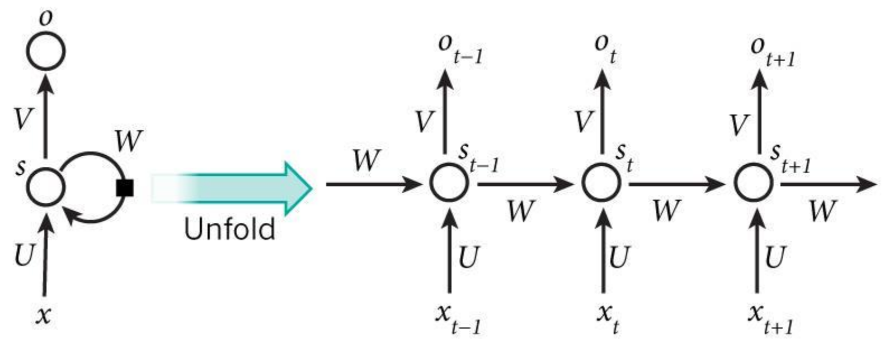


Figure 4: RNN Structure and its layers (LeCun, et al., 2015)

The backpropagation approach, which calculates the partial derivative of the error function with respect to the weight, introduces the vanishing gradients problem (Buduma, 2015), in which the gradient value goes exponentially decreasing with time. The vanishing gradient problem is encountered during the computation of the partial derivative of the error function with respect to the weight by the backpropagation method (Buduma, 2015). Because of this, RNN is ineffective at controlling long-term memory. This problem is solved by LSTM, which will be covered in the following.

3.4.1.2.2 LSTM

LSTM replaced RNN as it resolved its vanishing gradient problem. Input, output, forget, and cell gates are the four gates that make up the LSTM layer (Donahue, et al., 2017). The arithmetic equation below is used to calculate each gate.

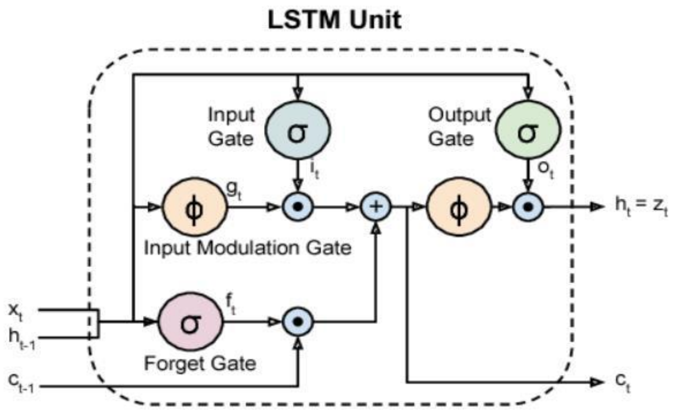


Figure 5: Structure of LSTM Unit (Donahue, et al., 2017)

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{3.8}$$

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right) \tag{3.9}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{3.10}$$

$$g_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{3.11}$$

$$c_t = f_t{\circ}c_{t-1} + i_t{\circ}g_t \tag{3.12}$$

$$h_t = o_t{\circ}tanh(c_t) \tag{3.13}$$

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{3.14}$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.15}$$

Here ∘ indicates the product of each element. Also, W indicates the weight matrices, and b is the parameter representing the bias vector.

The diagram and equations work as follows:

- First, the forget gate's responsibility is to choose which information should be retained and which needs to be overlooked. It will check h_{t-1} (previous hidden state) and x_t (which is the current input state) and then σ (sigmoid function) to give an output of 1 to keep and an output of 0 to forget.
- Then upon computing the sigmoid function of the previous hidden state, the current input gate i_t determines the significance of the current input vector.
- After that, the Sigmoid function results are used by the output gate o_t to decide which part of the LSTM cell's output should be presented in the next timestep.
- In the fourth step, when determining how much to write to the cell, the input modulation gate uses an activation function called tanh.
- Then c_t which is the cell state, is computed using all the 3 gates.

- The o_t and c_t are used to determine h_t in the last step (Oak, et al., 2016).

Although LSTM allows handling a long data sequence depending on previous sequences, such as the text, it has some drawbacks. The first reason is that LSTM is slow since the number of parameters is large in LSTM networks. LSTM takes data in sequence, and it takes a lot of time for the neural network to learn. Also, it is not the best model to capture the true meaning of words.

One of the most significant drawbacks of RNN and LSTM is that it's impossible to do transfer Learning in them. Transfer Learning is a machine learning technique in which a model is trained on one task and is re-purposed on a second related task. This is where the Pretrained-language model, such as BERT, beats models such as RNN, LSTM, etc. and will be discussed next.

3.4.1.2.3 BERT

BERT (Bidirectional Encoder Representations from Transformers), a pre-trained language model, attempts to understand ambiguous semantic relations in the text by extrapolating the context from the surrounding text. This model merely needs to be fine-tuned on new datasets after being pretrained on Wikipedia Text in order to be applied to a specific task. Figure 6 shows the structure of pretraining and fine-tuning BERT. It is a Transformer-based model developed by (Devlin, et al., 2019)**.** By concurrently conditioning the left and right context of words in all layers, BERT is intended to pre-train bidirectional representations from an unlabeled text (Devlin, et al., 2019).

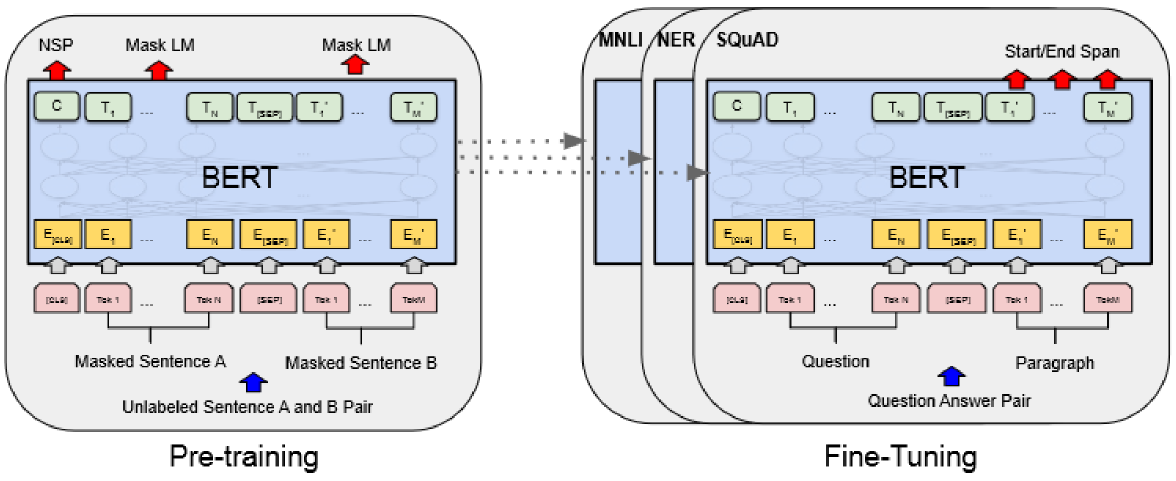


Figure 6: Pre-training and fine-tuning BERT (Devlin, et al., 2019)

As a result, the pre-trained BERT can be finetuned with just one additional output layer to create a model for various tasks such as question answering, sentiment classification, fill-in-the-blank, etc., without substantial task-specific architectural changes. The input embeddings of BERT were explained in section 3.3.4. In this section, we will give an overview of BERT's architecture and the intuition behind using BERT.

To Understand BERT, we need to understand Transformers as BERT is derived from it.

3.4.1.2.3.1 Transformer

The transformer is a model of an attention mechanism created for linguistic conversion (Vaswani, et al., 2017). To enhance the encoder-decoder model's performance, an attention mechanism was included. By emphasising some input data and downplaying others, the attention mechanism seeks to emulate the cognitive attention of the human brain. (Bahdanau, et al., 2015) came up with the idea of considering not only the context vector but also the relative importance that should also be given to each one of them. So, whenever a proposed model generates a sentence, it is searched for positions in the encoder hidden states where the most relevant information is available and is called "attention" (Anon., 2019).

Earlier encoder-decoder models used stacks of RNN/LSTM units (Bahdanau, et al., 2015). But RNN/ LSTM and issues mentioned in previous sections do not work well with long sentences. To solve these problems, the researcher created a technique in Paper (Cho, et al., 2014), where the authors presented a novel RNN-based Encoder-decoder. Although it performed great, it was demonstrated that the encoder-decoder network degrades rapidly as the input sentence length increases. Although LSTM is supposed to capture long-range sentences, it is seen to become forgetful in specific cases. In the paper (Vaswani, et al., 2017), the author proposed a novel encoder-decoder model, which performed better than previously existing encoder models and is called Transformers.

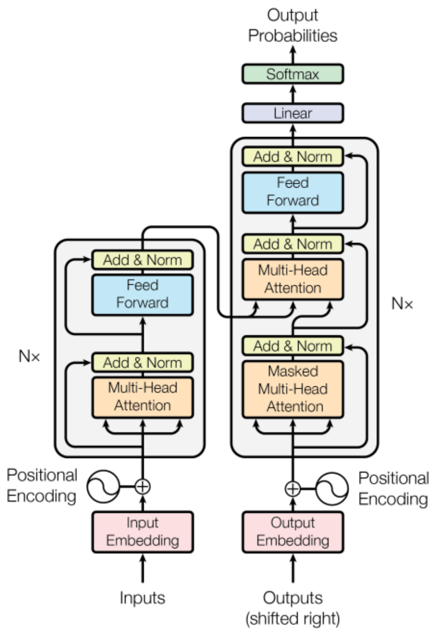


Figure 7: Architecture of Transformer (Vaswani, et al., 2017)

The main goal of the Transformer is to translate the language. As shown in figure 7, the transformer is divided into two layers. The left block is the encoder, and the right one is the decoder. The encoder used the text input to understand the semantic and contextual relations, and the role of the Decoder was to translate. But for sentiment classification, we don't require translational. Hence, BERT only uses the encoders' stacks for sentiment classification. Although Transformer has been explained in detail in the paper (Vaswani, et al., 2017), a Summary of how the encoder works is given below:

- Input Embedding: It takes text inputs in the form of vectors using word-piece embeddings (Wu, et al., 2016).
- Positional Encoding: Transformers contain no recurrence and no convolution. So, to inject some information about the position of the tokens in sequence.
- Multi-Head Attention Layer: This layer creates attention vectors that help determine which part of the model's input text should focus on.
- Feed-Forward Network: It transforms the attention vector to a suitable form for the next encoder or decoder block unit.

3.4.1.2.3.2 BERT Intuition

The Transformers encoder reads the input, and the decoder produces the prediction for the task. The transformer aimed to translate one language to another language. The encoder used the text input to understand the semantic and contextual relations, and the role of the Decoder was to translate. But since BERT's goal is to produce a language model, only the encoder part is necessary.

Unlike directional models such as RNN and LSTM, the transformer encoder reads the entire sequence of words simultaneously and looks at both the right and left sides of a word. That is why it is considered bidirectional. This characteristic allows the model to learn the context of words based on all surroundings, i.e., both left and right (Horev, 2018)**.** The Bert can be converted into a classifier by adding a classifier layer on top of the encoder output.

The process of pre-training the model on data and then fine-tuning it to a specific task is given in detail in (Devlin, et al., 2019).

Bert is pre-trained on two different NLP tasks that employ bidirectional capability: Masked Language Modelling and Next Sentence Prediction.

Masked Language Modelling (MLM): This training method aims to mask certain words in the text and have the BERT model predict the masked words based on the surrounding context. The masked words are substituted by the [MASK] token, and the BERT model recognizes it and predicts the word it should be replaced with. This training method helps the BERT to understand the bidirectional relationships in sentences.

Next Sentence Prediction: This training method is aimed at teaching the BERT model to predict whether the two sentences have any sequential and meaningful relationship or not. After MLM, the Next sentence prediction is the next unsupervised task BERT is Trained on. BERT is pre-trained on a binarized version of the next sentence prediction derived from a monolingual corpus (Devlin, et al., 2019).

3.4.1.2.3.3 Pretraining and Finetuning BERT

BERT needs to be trained beforehand and then fine-tuned on a task-specific dataset before being used for any purpose, such as sentiment analysis. Both time and resources are used in this process. For instance, (Devlin, et al., 2019) employ 16 TPU chips in the Pre-training BERT model. Because of this, there are websites where users can import models that have already been trained and fine-tuned on certain data directly into projects, saving both time and resources.

The advantage of BERT over other neural networks and classical machine learning models is that it can understand the contextual relationship in words in more detail. So, if the same word is used in a different context BERT model will realize it.

3.4.2 Topic Extraction

Topic analysis or extraction in NLP is a technique that helps extract meaning from documents by finding common or recurrent topics. Many methods have been built to extract topics. Statistical and clustering methods can help remove topics such as LDA, K means clustering and HDBSCAN. Let's discuss them and conclude which will be advantageous for this project.

3.4.2.1 LDA

LDA is a statistical model which can be used for topic modelling. The mathematical explanation of how it extracts topics, and its functioning is explained in (Jelodar, et al., 2019). The assumptions that LDA considers are:

- Each document is just a bag of words, and the order or grammatical role is not considered in the model.
- High Frequency words such as ‘is’, ‘are’, ‘a’, etc. don’t carry information and can be eliminated without losing any information.
- We know the number of topics beforehand.

These assumptions make this model bad as no contextual relationships are considered while extracting topics. Also, if the data is new and large, it’s challenging to find out how many topics there will be. So it becomes time-consuming to find the correct number of topics If the data is large. LDA also suffers from more general problems, such as a lack of ground truth to evaluate their models. We can use only the experts to create topic vocabulary and have annotators apply the vocabulary to the text. So LDA is an ideal method for this project due to time constraints. We need to investigate unsupervised models such as K means clustering and HDBSCAN.

3.4.2.2 K-means clustering

K-means is a popular centroid-based flat clustering algorithm that MacQueen introduced in 1967 (Na, et al., 2010.). It is a simple algorithm widely used for clustering large sets of data. To understand how it works, let us suppose that a dataset has n data points x_1 x_2, … x_n such that each data is in a real d-dimensional space, **R**^{d}**,** the issue of trying to the clustering with the minimum variance of the data set into k clusters is that of finding k points (m_j, where j = 1,2, 3,….k) in the **R**^{d} such that

$$\frac{1}{n}\sum_{i=1}^{n}\left[\min_{j} d^2(x_i, m_j)\right] \tag{3.16}$$

will be minimized, where $d\ \ (x_i, m_j)$ represents the Euclidean distance between x_i and m_j points. $\{m_j\}$ are the cluster centroids (Fahim, et al., 2006). The issue in equation 3.16 is to find k cluster centroids in a way that the MSE (Mean Squared Error) or the averaged squared Euclidean distance between a data point and the nearest cluster centroid is minimized.

The K-means algorithm is a straightforward method to execute. It can be thought of as a procedure of gradient descent. And the steps are as follows. It begins at initiating cluster centroids and iteratively updates these centroids to decrease the objective function in equation 3.16. K-means always converge to a local minimum, and a local minimum found depends on the cluster centroid initiated at the start**.** Figure 8 shows the k-means- algorithm.

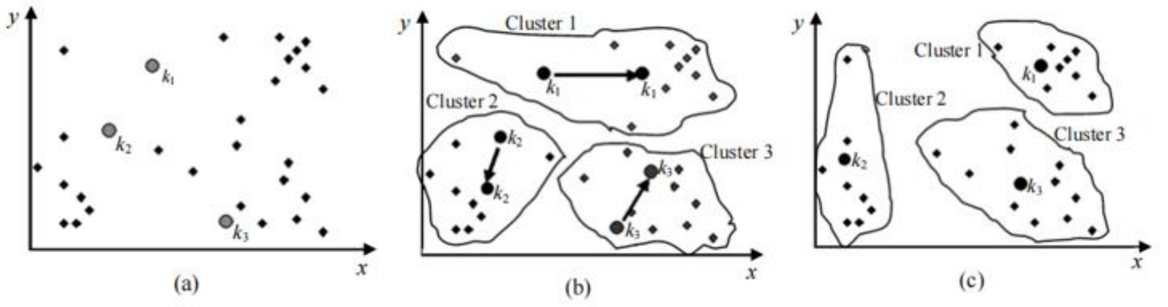


Figure 8: (a) initial centroids of a dataset; (b) Position of Centroids are recalculated; (c) Final position of the centroid (Fahim, et al., 2006)

K-means clustering algorithm is good when the data size is small, is lower dimensional, and we have a good understanding of the dataset. It is because k means cluster all the data points and ignores that some data points are noise and should not be included. This drawback can be resolved by density-based algorithms such as HDBSCAN, which will be discussed next.

3.4.2.3 HDBSCAN

HDBSCAN is a density-based hierarchal clustering algorithm that performs very well even on high-dimensional data with noise. It was developed by (Campello, et al., 2013), and as mentioned in (McInnes, et al., n.d.), it extends the DBSCAN model by changing it to a hierarchical clustering algorithm and using a technique to fetch a flat clustering based on clusters' stability. To understand the process of HDBSCAN, we can simplify the steps of density-based clustering into three stages (Berba, 2020):

1. Estimating the densities in the data: A common way is using 'core distance' to evaluate the density around specific points. Core distance is a data point's density from the K-th nearest neighbour. Points in the high-density regions will have a small core distance, as shown in figure 9.

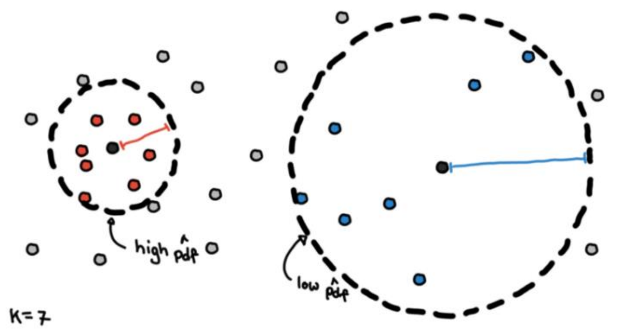


Figure 9: Core distance when k=7 (Berba, 2020)

2. Find regions of high densities: There are two methods for finding areas of high density. We can set a global threshold if we have a similar density in different clusters. By getting the points above this threshold and grouping points together, we get clusters. For data with carrying density regions, HDBSCAN creates a hierarchical structure to understand which varying densities (peaks of denser regions) need to be merged and what their order should be. It also asks which cluster should be kept together or split into subclusters (Berba, 2020). The images below (figures 10 and 11) give an excellent example of merging or splitting data of varying densities

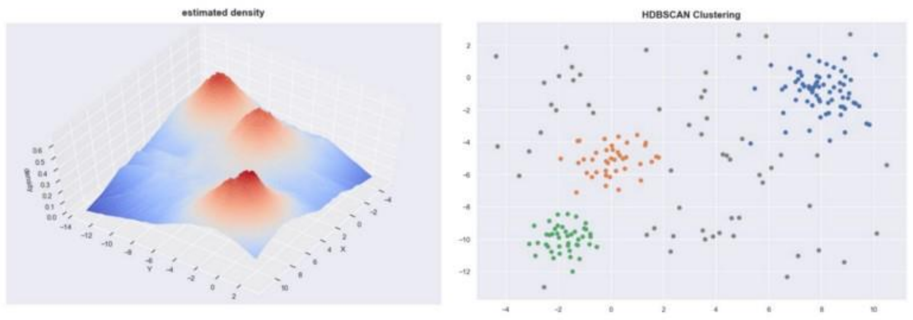


Figure 10: HDBSCAN clustering for 3 cluster dataset (Berba, 2020)

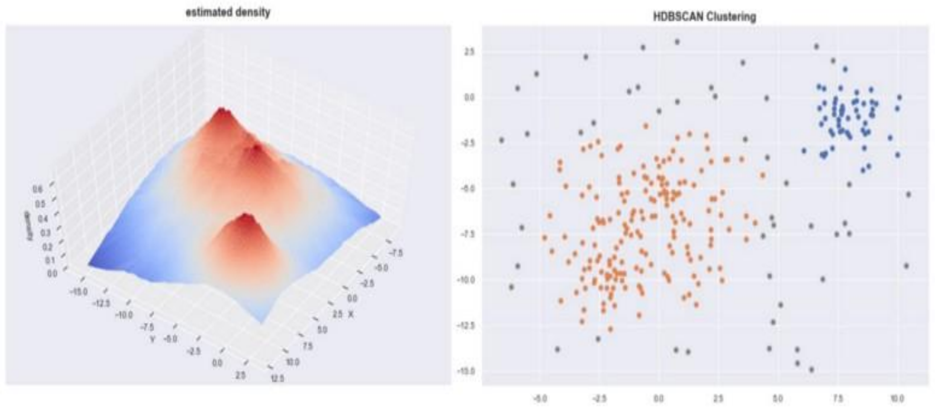


Figure 11: HDBSCAN clustering for a 2-cluster Dataset. Although there are three peaks, HDBSCAN merges two peaks by understanding the hierarchical structure (Berba, 2020)

3. Combine the points in these high-density regions: Finally, we can group the data points in the high-density regions and create clusters while avoiding the noisy points.

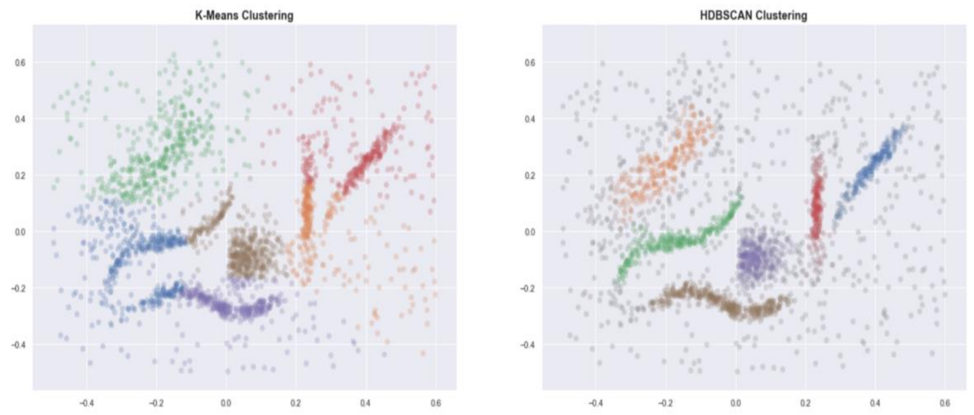


Figure 12: Comparing the final clusters made by K-means and HDBSCAN (Berba, 2020)

As shown in figure 12**,** HDBSCAN works better in clustering the relevant data points and avoids noise. The k-means clustering algorithm clusters all the data points without considering the density of similar data points, which is why HDBSCAN is better.

3.5 Optimizer

3.5.1 ADAM

Adam is a technique for stochastic objective function optimization using first-order gradients that is based on adaptive estimations of lower-order moments. (Kingma & Ba, 2014). The expected value of a random variable to the power of n is known as the n-th moment. Adam is a method for computing adaptive learning rates that determines individual learning rates for different parameters. For deep neural networks, this optimizer has taken the role of stochastic gradient descent since it updates the weights of the neural networks much more quickly and effectively. In most cases, its hyperparameters don't need much fine adjustment. The Adam optimizer is a default optimizer in the Pre-trained language model BERT and is used in neural networks to update the weights of neural networks. The pseudo-code of the Adam Optimizer is as follows (Kingma & Ba, 2014):

```
**Require:** $\alpha$: Stepsize
**Require:** $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
**Require:** $f(\theta)$: Stochastic objective function with parameters $\theta$
**Require:** $\theta_0$: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  **while** $\theta_t$ not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep $t$)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t/(1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t/(1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t/(\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  **end while**
  **return** $\theta_t$ (Resulting parameters)
```

Figure 13: Pseudo Code of Adam optimizer (Kingma & Ba, 2014)

3.6 Performance Evaluation

When we train the machine learning model, it is essential to check its performance and evaluate it. For instance, after training the sentiment classifier model in this project, we need to check its performance. To achieve this, we will list some popular evaluation techniques and discuss which one will be the most useful for us in this project.

3.6.1 Confusion Matrix

The confusion matrix counts the percentage of the data points that were truly predicted as negative or positive, which are referred to as TP (True Positive) and TN (True Negative). Also, it is used to find the percentage of falsely predicted data points as positive or negative, and they are called FP (False Positive) and FN (False Negative) (Fawcett, 2006). The Table below in figure 14 shows the structure of a confusion matrix.

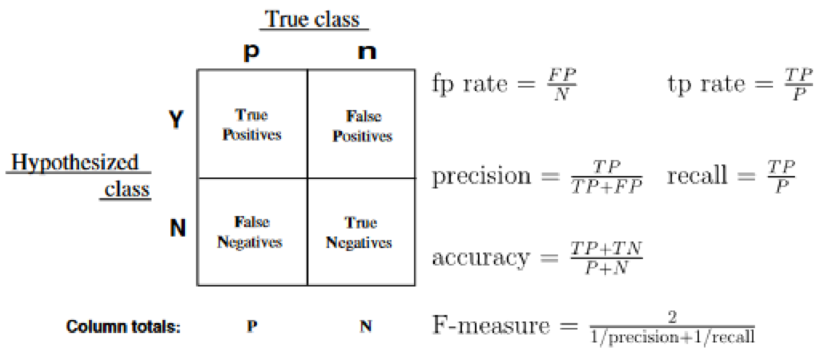


Figure 14: Confustion Matrix (Fawcett, 2006)

Using the Confusion Matrix, we can calculate other essential performance measures which are discussed below.

3.6.2 Accuracy

From the confusion matrix, we can derive the accuracy, and it can be computed as follows:

$$\frac{TP+TN}{TP+TN+FP+FN} \tag{3.17}$$

Accuracy is an excellent measure to calculate if the dataset is balanced. If the Dataset is imbalanced, we need to calculate precision, recall, and F1 Score to understand the performance of the trained machine learning model.

3.6.3 Precision and Recall

This measure is used to compute the ratio of True Positives to the total positively predicted values (POWERS, 2011).

$$\frac{TP}{TP+FP} \tag{3.18}$$

The recall is used to compute the ratio of True Positives to the values that were actually positive (POWERS, 2011).

$$\frac{TP}{TP+FN} \tag{3.19}$$

3.6.4 F1 Score

To calculate how the classification model, such as the Sentiment Analysis model, has performed, we use precision and recall to compute the value of the F1 Score (He & Garcia, 2009.), and the equation is as follows:

$$\frac{2(Precision \times Recall)}{Precision + Recall} \tag{3.20}$$

3.7 Overfitting

When a machine learning model is trained on a dataset to predict all the data points of the training set with reasonable accuracy but fails to achieve good performance on an unseen new dataset, we call the machine learning model overfit (Dietterich, 1995). It is because the model cannot generalize (ability to react to new data). Then there is a need to apply specific steps depending on the machine learning model to reduce overfitting. One standard method is regularization (Tian & Zhang, 2022).

Summarization

So, in this chapter, we introduced various methods, concepts, and algorithms, that we explored while building this project. This section listed some Data Pre-processing techniques required to build NLP models, some machine learning techniques used for sentiment classification, and discussed the widely used methods for topic extraction. We also discussed reasons for not using some models while discussing the advantages of certain algorithms, which helped us choose the right tools for this project. In the end, we discussed some performance evaluation metrics and the role of overfitting while building the machine learning models for NLP (Natural Language Processing) tasks.

In the next section, we will use the knowledge of this section to choose the right tools and explain the steps of building this project.

CHAPTER 4: Implementation and Results

Introduction

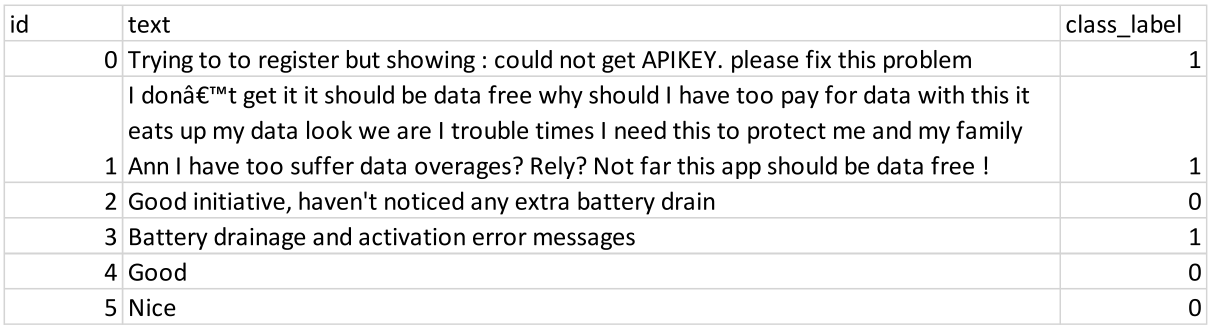
In this section, we will explain the process of building the project. The flow chart in figure 1 already gives an excellent overview of the process of building this project. This section will begin by describing the datasets used to develop our Topic-Wise Temporal Sentiment Classification model for analyzing NHS COVID-19 App Reviews. Then the tools used to create the project are introduced. Finally, the process and steps of building the project will be explained thoroughly.

4.1 Dataset Description

For this project, we have used two datasets. The first dataset will be referred to as Dataset 1, and the second will be referred to as Dataset 2.

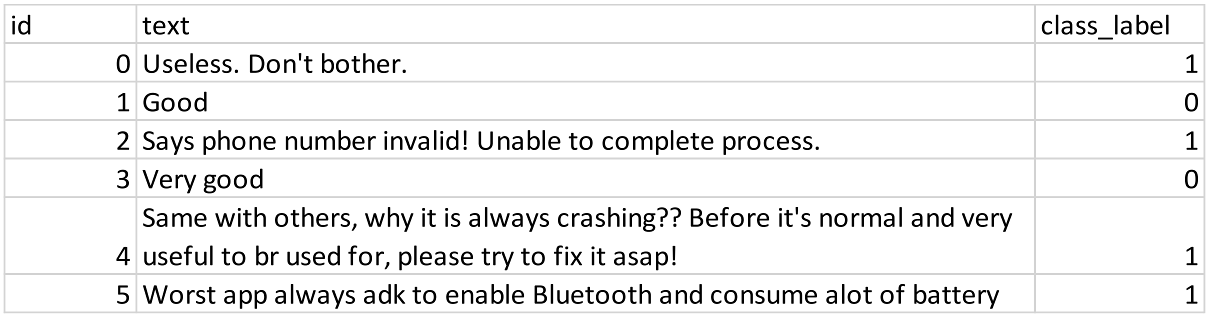
4.1.1 Dataset 1 – A Benchmark Dataset for Sentiment Analysis of Users' Reviews on COVID-19 Contact Tracing Applications

Dataset 1 consists of user reviews of COVID-19 Contact Tracing applications from 46, and this data set also contains manually annotated labels mentioning the sentiment (0 means positive and 1 means negative) of these reviews. This dataset has been created by the paper's authors (Antypas, et al., 2021)**,** and the dataset is provided in (Ahmad, et al., 2021)**.** Dataset 1 is composed of two sub-datasets, one for training and the other one for testing. For easy understanding, we will refer to the Training dataset of Dataset 1 as Dataset1_Train and the test data of Dataset 1 as Dataset1_Test. The images below (Figure 15 and 16) shows some reviews of the Dataset1_Train and as Dataset1_Test along with the label.



id	text	class_label
0	Trying to to register but showing : could not get APIKEY. please fix this problem	1
1	I donâ€™t get it it should be data free why should I have too pay for data with this it eats up my data look we are I trouble times I need this to protect me and my family Ann I have too suffer data overages? Rely? Not far this app should be data free !	1
2	Good initiative, haven't noticed any extra battery drain	0
3	Battery drainage and activation error messages	1
4	Good	0
5	Nice	0

Figure 15: User Reviews of Dataset1_Train and the class labels



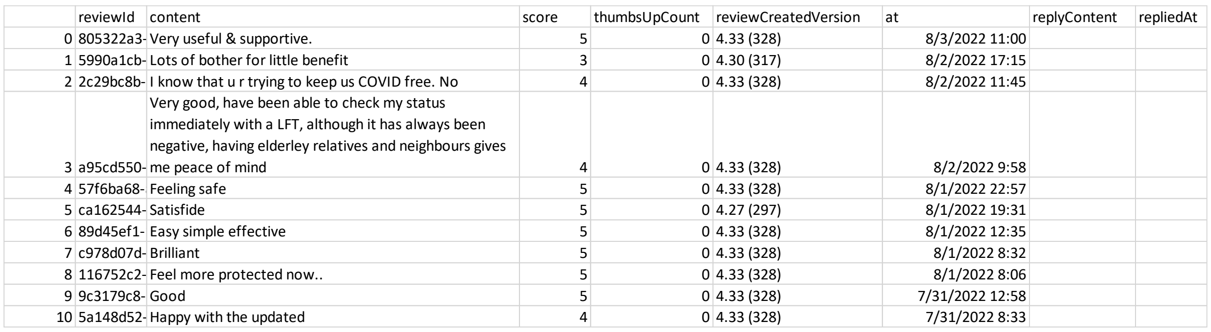
id	text	class_label
0	Useless. Don't bother.	1
1	Good	0
2	Says phone number invalid! Unable to complete process.	1
3	Very good	0
4	Same with others, why it is always crashing?? Before it's normal and very useful to br used for, please try to fix it asap!	1
5	Worst app always adk to enable Bluetooth and consume alot of battery	1

Figure 16 User Reviews of Dataset1_Test and the class labels

Figure 15 shows Dataset1_Train, and Figure 16 shows Dataset1_Test. In both the Train and Test Datasets, the column 'text' contains the user reviews, and the column 'class_label' contains the sentiment score where 0 means positive and 1 means negative. Dataset1_Train has 22286 user reviews, and Dataset1_Test has 10978 user reviews. Out of 22286 reviews in Dataset1_Train, 11906 reviews have class label 1 (Negative Sentiment), and 10380 reviews have class label 0 (positive sentiment). For Dataset1_Test, out of 10978 user reviews, 5770 user reviews have a class label 1 (negative sentiment), and 5207 user reviews have a class label 0 (positive sentiment). It means that the dataset created by the authors of (Antypas, et al., 2021) is balanced.

4.1.2 Dataset 2 – Dataset of User Reviews of NHS COVID-19 App from Google Play Store

Google Play is an online store that comes preinstalled with Android devices that support Google Play. People can find different applications in the Google Play store, including games, movies, fitness, etc. The NHS COVID-19 App, a contact tracing app for detecting and warning people who have come in touch with coronavirus-positive patients, is available on Google Play Store. To create Dataset 2, we have scraped the user reviews for the NHS COVID-19 app from the google Play Store. We scraped other metadata about the reviews, such as ID, Username, User Image, Date and time of Review, Rating (Score), which users give to the application (out of 5), etc. Figure 17 shows the first 11 rows of the dataset. We have removed the Username column to make it anonymous. The column 'context' contains the user reviews, the column 'score' has the score given by the users to the application, and the 'at' column shows the data of when the review was posted. The 'score' column contains the user rating, and its values can be 1, 2, 3, 4 or 5, where 5 is the highest rating and 1 is the lowest.



	reviewId	content	score	thumbsUpCount	reviewCreatedVersion	at	replyContent	repliedAt
0	805322a3-	Very useful & supportive.	5	0	4.33 (328)	8/3/2022 11:00		
1	5990a1cb-	Lots of bother for little benefit	3	0	4.30 (317)	8/2/2022 17:15		
2	2c29bc8b-	I know that u r trying to keep us COVID free. No	4	0	4.33 (328)	8/2/2022 11:45		
3	a95cd550-	Very good, have been able to check my status immediately with a LFT, although it has always been negative, having elderley relatives and neighbours gives me peace of mind	4	0	4.33 (328)	8/2/2022 9:58		
4	57f6ba68-	Feeling safe	5	0	4.33 (328)	8/1/2022 22:57		
5	ca162544-	Satisfide	5	0	4.27 (297)	8/1/2022 19:31		
6	89d45ef1-	Easy simple effective	5	0	4.33 (328)	8/1/2022 12:35		
7	c978d07d-	Brilliant	5	0	4.33 (328)	8/1/2022 8:32		
8	116752c2-	Feel more protected now..	5	0	4.33 (328)	8/1/2022 8:06		
9	9c3179c8-	Good	5	0	4.33 (328)	7/31/2022 12:58		
10	5a148d52-	Happy with the updated	4	0	4.33 (328)	7/31/2022 8:33		

Figure 17: Scraped User reviews of NHS COVID-19 app from Google Play store

Dataset 2 is not processed or cleaned, and we will need to go through some cleaning steps so that it can be used to build our project.

4.2 Model Selection for Topic-wise Temporal Sentiment Classification

In the Background section, we discussed various models for sentiment classification and Topic Extraction. We discussed their advantages and disadvantages, which will help us select the best models for building this project.

For Sentiment Classification, this project aims at creating a model which is not just task-specific for classifying user reviews of COVID-19 contact tracing applications but also be applied to similar tasks. This can only be achieved by the Transfer Learning model, which leverages the power of being pretrained on huge data sets and needs fine tuning to new data. Also, Transfer Learning models perform better on unseen data even if the context might differ from the data points on which the model was trained. Hence it also becomes essential for the models to understand the context in which the words are used in a textual document. Classical machine learning models fail at that, and even many neural networks cannot achieve it, as discussed in the Background Section. But BERT is a pre-trained language model which performs well for this task. Also in (Ahmad, et al., 2021), BERT has proven to perform better on Dataset 1 than the classical machine learning models such as SVM and Multinomial Naïve Bayes. Hence, we will choose the BERT-based model for building our Sentiment Classifier.

For Topic extraction, models such as K-means and LDA ignore the noise in the data and cluster everything included. This is because LDA is just a statistical method, and K-means is a centroid-based clustering algorithm. To exclude Noisy data points that are not like most data points, density-based clustering algorithms such as HDBSCAN are preferred. Hence for topic extraction, this project will use HDBSCAN.

4.3 Tools

This project was built using Python 3.9.12, which is a programming language. The code was written in Jupyter Note, and various Python Libraries were used to construct this project. Some of the most important ones are as follows:

- NumPy – It is a python package for scientific computing. It provides a multidimensional array object and various derived objects like matrices, masked arrays, and different mathematical functionalities (Oliphant, n.d.).
- Pandas – It is an open-source library providing various data structures and analysis tools which are easy to use for Python Programming (McKinney, n.d.).
- TensorFlow – It is a module developed and maintained by Google, which can be used within Python to do a lot of different scientific computing, machine learning, and artificial intelligence implementations. It also contains a lot of pre-trained machine learning models such as BERT, which can be used for various Natural Processing Language tasks such as sentiment classification. These models are pretrained on large datasets and can be finetuned on new datasets according to specific use cases. This is known as Transfer Learning (Google, 2022).
- BERTopic – It is a topic modeling technique that leverages Transformers and c-TF-IDF to create denser clusters of data points which results in easily interpretable topics while keeping the essential words that result in the creation of clusters in topic description (Grootendorst, 2022).
- Matplotlib – It is a comprehensive and robust library for visualizations in Python (The Matplotlib development team, n.d.).
- HDBSCAN – It is a library that lets us use unsupervised learning to create a density-based hierarchical cluster of datasets using the algorithm of HDBSCAN.
- Re – Regular expression is a sequence of characters that helps us create a search pattern. Re is a library of python which works with Regular expression (Python Software Foundation, n.d.).
- Seaborn – It is a Python data Visualisation library based on Matplotlib. It provides a higher-level interface for creating beautiful statistical Visualizations. For instance, we have used this library to create a confusion matrix in this project (Waskom, n.d.).

4.4 Building the Sentiment Classification model

In this section, we will mention the steps that were taken to build the sentiment classifier model.

4.4.1 Data Pre-Processing

For training and testing the sentiment classifier model, Dataset 1 (4.1.1) is divided into two sub-datasets, Dataset1_Train and Dataset1_Test, each for training and testing tasks.

These datasets have no null values and require no Data-preprocessing. The reason for not requiring data Preprocessing is that Pretrained language models such as BERT use all the information in a sentence, including punctuation and the stop-words (Vaswani, et al., 2017).

4.4.2 Training and Testing the Sentiment Classification model

The following steps were taken to build the sentiment classification model:

- Import Dataset1_Train to the Jupyter Notebook and create a data frame using Pandas.
- Then we split the data frame into a training and validation set in the ratio of 4:1. Also, in the training set. User reviews (‘text’ column) and labels (‘class_label’ column) in both the training set and validation set will be stored in different variables. So, there will be four variables in total. The first variable will store user reviews of the training set; the second variable will store labels of the training set; the third variable will store user reviews of the validation set; and the fourth variable will store labels of the validation set.
- Then we import DistilBertTokenizerFast (Delangue & Chaumond, n.d.) (identical to BERT Tokenizer), the type of BERT tokenizer used to create embedding to convert the user reviews into texts.

- Then we use distil-base-uncased (Bahdanau, et al., 2015) (a distilled version of the BERT base model) to train our model and will feed the user reviews and labels of training set into it. Hyperparameters were changed ten times till we got the ideal results. After tuning the BERT model, the model is saved to the local system so that it can be loaded anytime.
- To check if the model is good, a validation set is used, which is unseen data for the trained model. A confusion matrix (using the seaborn library) is created to check the model's performance. The confusion matrix can be seen in figure 18 (a)**.** The model's performance is good as we have achieved high values for the True Positive and True Negative. Using equation 3.17, the model's accuracy in percentage comes out to be 97.31 % which is excellent.
- Then we import Dataset1_Test, perform sentiment classification using the trained BERT model, and check its performance using a confusion matrix. The confusion matrix is shown in figure 18 (b), and by using equation 3.17, the accuracy comes out to be 94.06 %.

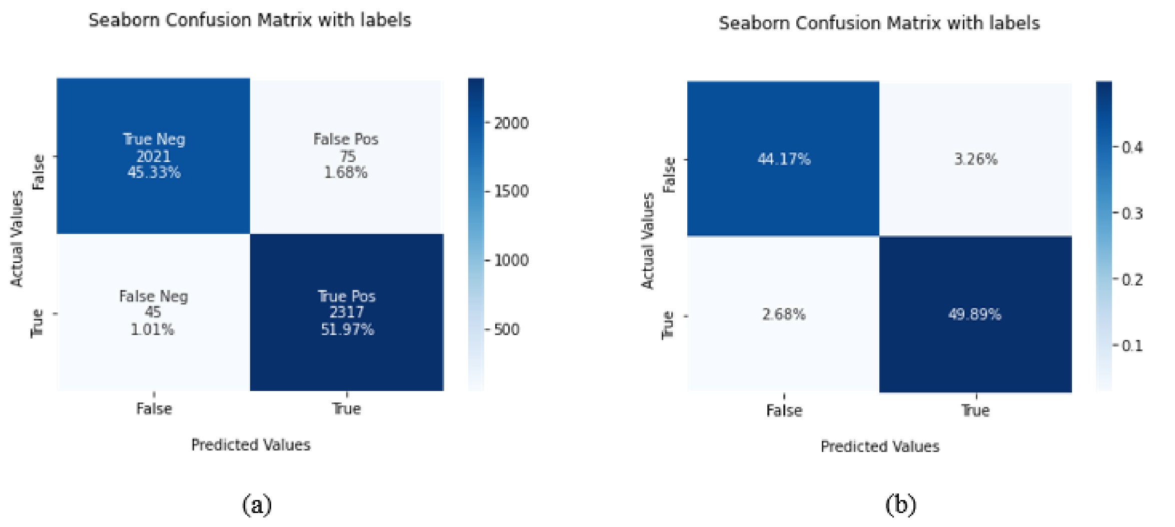


Figure 18: (a) Confusion Matrix of sentiment classifier for validation dataset; (b) Confusion Matrix of sentiment classifer on Dataset1_Test

The Code for this sentiment classification task is available on Gitlab, and the link to the code is available in the Appendix.

4.4.3 Hyperparameters for BERT Sentiment Classifier

To train the BERT sentiment classifier, some hyper-parameters (Delangue & Chaumond, n.d.) needed to be set. In total, ten combinations of values for hyper-parameters were selected, and the best hyper-parameters were selected based on the accuracy the trained model showed in the confusion matrix of the validation dataset. These hyper-parameters are:

- num_train_epochs (Epochs): This parameter decides how many times the model should run on the whole dataset. If this value is substantial, it can result in overfitting. But this value is also constrained by the processing power of the Computer. (Devlin, et al., 2019) had done only three epochs for their fine-tuning studies. For our project, the model was overfitting at four epochs. 2 epochs gave the best results.
- per_device_training_batch_size (Batch Size): This parameter decides the size per GPU/TPU core/CPU for training. So these values depend on the machine (computer) on which this model is being trained. If this value is high and the computer has sufficient GPU/TPU core/CPU, the shorter the period it will take to train the model. The optimal value for this system depends on the machine. (Peltarion, n.d.) recommends keeping the product of batch size and max sequence length lower than 3000. So if the sequence length is 512, batch size should be kept at 8 or lower than 8. The sentiment classifier performed well when the batch size was kept as 2.

- learning_rate (Learning Rate): This is the learning rate for the AdamW optimizer, which implements the Adams algorithm for optimizing the weights of BERT. The learning rate controls the size of the update steps along the gradient per epoch. The author of the BERT paper (Devlin, et al., 2019) recommends using learning rates of 5e-5, e-5, and 2e-5. We used the value of 5e-5 for this project as it gave good results. If this value is not selected correctly, the Sentiment classifier model can become forgetful and give wrong predictions.
- warmup_steps: This value is used to decide the number of steps to use for a linear warmup from 0 to the learning rate. It is the number of warmup steps for the learning rate scheduler. The author (Devlin, et al., 2019) used 10,000 learning rate warm steps to pre-train the BERT as the data was very large and the model was being trained from scratch. But were are using a pre-trained model and just fine-tuning it. So Large Wam up steps is not required. We choose this value as 500 after repeatedly training the model and checking the accuracy of predictions on the validation dataset. It means that for the first 500 steps, the model’s learning rate is kept low, and after 500 steps, the regularly scheduled learning rate is used.
- weight_decay: It is a way of applying regularization to neural networks to avoid overfitting. This value should be kept modestly, or else the model may become underfit and unable to predict the training data accurately. The optimal value for this project was 0.01. Even the author of the BERT (Devlin, et al., 2019) used the value of 0.01 for weight decay for pre-training the BERT. Weight decay is applied to all the layers except all bias and LayerNorm weights in AdamW Optimiser.

4.5 Topic Extraction and Topic-wise Temporal Sentiment Classification model

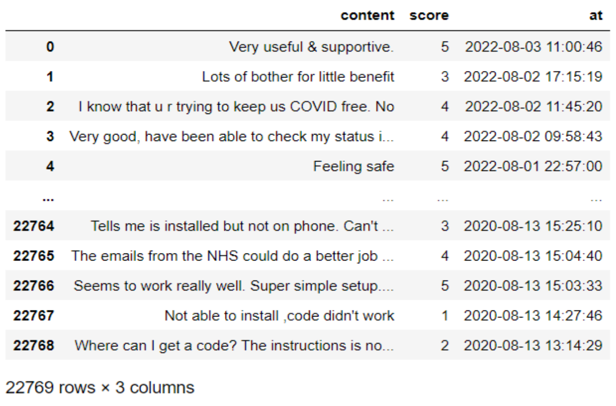
We will begin by explaining the topic extraction process and then define steps for temporal sentiment classification of user reviews per topic.

4.5.1 Data Scraping and Pre-processing

For Topic Extraction and Topic-wise Sentiment Analysis, we will use Dataset 2. This dataset is created by first scraping from the User review section of the NHS COVID-19 App on the Google Play store. The dates of the user reviews ranged from August 2020 to August 2022. To scrape, we used the google-play-scraper library of python.

After scraping, the dataset is saved and needs to go through preprocessing. The steps involved in preprocessing are as follows:

- We first remove columns named ‘reviewid’, ‘thumbsUpCount’, ‘reviewCreatedVersion’, ‘replyContent’, ‘replyAt’, as shown in Figure 17. So, we are left with the columns ‘content’, ‘score’, ‘at’, which contain the user reviews, ‘score’ (rating ranges from 0 to 5, 5 being the highest) of the app by the user, and the time at which the review was posted respectively.
- Then we remove emojis from the text. This was done to achieve better accuracy when we performed sentiment classification using the model trained in section 4.4, as the model was trained on text without emojis.
- After that, the rows with null/empty values are removed.



	content	score	at
0	Very useful & supportive.	5	2022-08-03 11:00:46
1	Lots of bother for little benefit	3	2022-08-02 17:15:19
2	I know that u r trying to keep us COVID free. No	4	2022-08-02 11:45:20
3	Very good, have been able to check my status i...	4	2022-08-02 09:58:43
4	Feeling safe	5	2022-08-01 22:57:00
...	...	...	...
22764	Tells me is installed but not on phone. Can't ...	3	2020-08-13 15:25:10
22765	The emails from the NHS could do a better job ...	4	2020-08-13 15:04:40
22766	Seems to work really well. Super simple setup....	5	2020-08-13 15:03:33
22767	Not able to install ,code didn't work	1	2020-08-13 14:27:46
22768	Where can I get a code? The instructions is no...	2	2020-08-13 13:14:29

22769 rows × 3 columns

Figure 19: Dataset 2 is Pre-processed to achieve this dataset

The cleaned and pre-processed dataset is shown in figure 19. The user reviews are in the 'content' column, user rating (values can be 1, 2, 3, 4 and 5) is in the 'score' column, and the 'at' column shows the dates.

4.5.2 Topic Extraction process

After the preprocessing of Dataset 2, we moved on to the next step to extract topics from it. The steps are as follows:

- We have used BERTopic, a python Library Developed in paper (Grootendorst, 2022) on the preprocessed**.** The flow diagram of BERTopic is shown in figure 20. BERTopic allows us to create topics for text documents in 6 steps (Grootendorst, n.d.) which are as follows
  1. Convert Documents (for instance, a user review in our project is a document) to vector representation using pre-trained language models. In our project, we have used the default language model that comes with this library, BERTopic ("all-MiniLM-L6-v2"), as it works great for documents in English.
  2. As the vector embedding has high dimensionality, and the same data can be represented at lower dimensionality while keeping the maximum data variance, BERTopic applies dimensionality reduction to reduce the dimensionality of documents so that machine learning algorithms have an easy time finding clusters. We have used the Default UMAP (McInnes, et al., 2018) reduction method that comes with the BERTopic library.
  3. A clustering algorithm such as HDBSCAN, K-means etc. is used to cluster reduced dimensionality vectors to find semantically similar documents. HDBSCAN has been used for this project, and its benefit over K-means is explained in the Background Section.
  4. Each topic is tokenized into a bag-of-words representation, allowing BERTopic to process data without affecting the input embedding. While tuning the model for better topics, this project selected Count Vectorizer to prevent stop words from appearing in the topic representation.
  5. Words interesting to each topic are calculated with a class-based TF-IDF process known as c-TF-TDF (Grootendorst, 2022). The formula for calculating c-TF-IDF is

     $$W_{x.c} = \|tf_{x,c}\| \times log\left(1 + \frac{A}{f_x}\right) \tag{3.21}$$

     Where $tf_{x,c}$ = frequency of x in class c, f_x = frequency of word x in all classes and A = average number of words per class
  6. MMR (Maximal Marginal Relevance) can be used to make similar and repeating words more varied. For instance, if an extracted topic from a set of documents has the topmost important words as 'bicycle', 'bicycles' and 'cycling', the MMR algorithm will convert these words representing the topic to 'bicycle, 'pedals' and 'chains' It is an optional step and not mandatory. This project does not use this step as we want to stick to the words used by the users in their review of the NHS COVID-19 application.

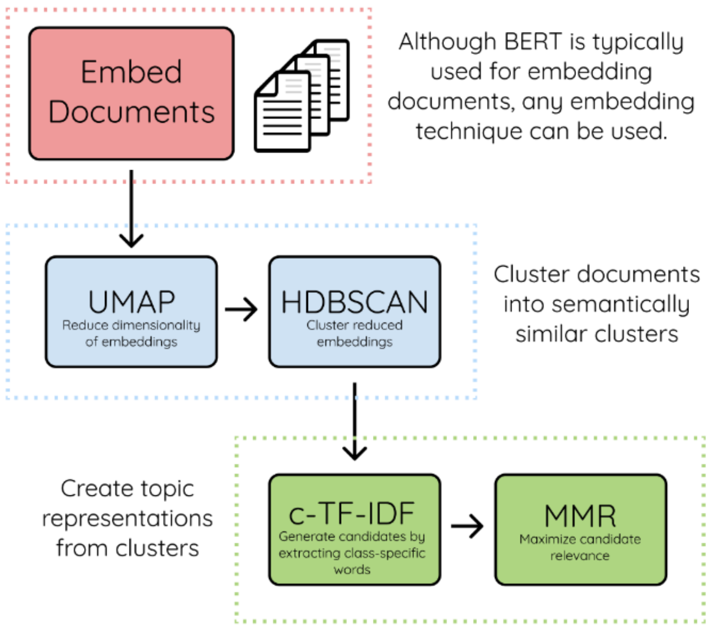
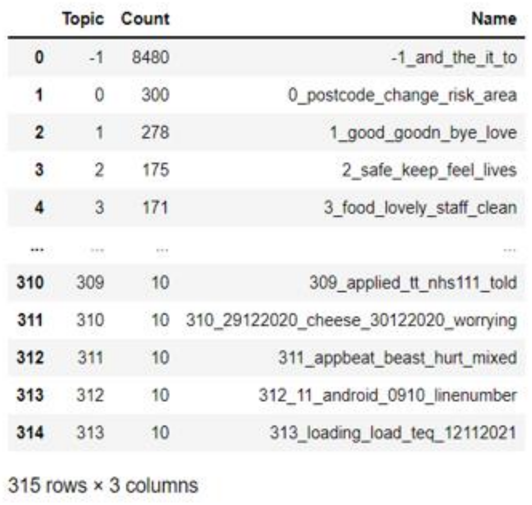

Embed Documents
Although BERT is typically used for embedding documents, any embedding technique can be used.
UMAP
Reduce dimensionality of embeddings
HDBSCAN
Cluster reduced embeddings
Cluster documents into semantically similar clusters
Create topic representations from clusters
c-TF-IDF
Generate candidates by extracting class-specific words
MMR
Maximize candidate relevance

Figure 20: Flow diagram of BERTopic *(Jawa & Anand, 2022)*

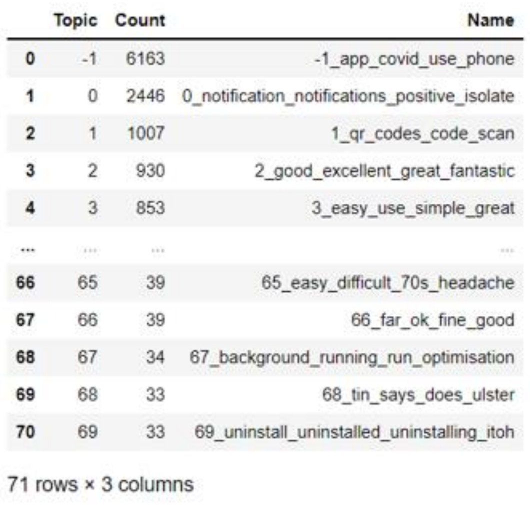
- First, we use the default setting of BERTopic to extract topics and then tune the parameters of BERTopic to get better results. In the default setting, we got about 314 topics. Topics in the default setting are shown in figure 21 (a). -1 topic refers to the user reviews which are outliers and do not fit in any topic. But many of those topics are similar, as noticed in the heat map in figure 22. So, we need to reduce the number of topics so that we can focus on topics that differ from each other.



	Topic	Count	Name
0	-1	8480	-1_and_the_it_to
1	0	300	0_postcode_change_risk_area
2	1	278	1_good_goodn_bye_love
3	2	175	2_safe_keep_feel_lives
4	3	171	3_food_lovely_staff_clean
...	...	...	...
310	309	10	309_applied_tt_nhs111_told
311	310	10	310_29122020_cheese_30122020_worrying
312	311	10	311_appbeat_beast_hurt_mixed
313	312	10	312_11_android_0910_linenumber
314	313	10	313_loading_load_teq_12112021

315 rows × 3 columns

(a)



	Topic	Count	Name
0	-1	6163	-1_app_covid_use_phone
1	0	2446	0_notification_notifications_positive_isolate
2	1	1007	1_qr_codes_code_scan
3	2	930	2_good_excellent_great_fantastic
4	3	853	3_easy_use_simple_great
...	...	...	...
66	65	39	65_easy_difficult_70s_headache
67	66	39	66_far_ok_fine_good
68	67	34	67_background_running_run_optimisation
69	68	33	68_tin_says_does_ulster
70	69	33	69_uninstall_uninstalled_uninstalling_itoh

71 rows × 3 columns

(b)

Figure 21: (a) BERTopic results for default hyper-parameter values; (b) BERTopic results after tuning the hyper-parameters

- We tuned the parameters of the BERTopic, such as n-gram_range, min_topic_size (minimum number of topics documents that must be present in each topic), and nr_topics. After several trial and error attempts and changing the parameters 25 times, optimal hyperparameters were achieved that gave desirable topics.

We also added stop words to the CountVectorizer to prevent stop words from showing up in words representing the topics.

- The model managed to extract 70 topics (numbered from 0 to 69), and after comparing the heatmap in figure 22 and figure 23, we noticed that we had reduced the number of similar topics. And the topics are shown in figure 21 (b)**.** Also figure 24 shows the top 10 topics and the top 5 words/terms used to find that topic.

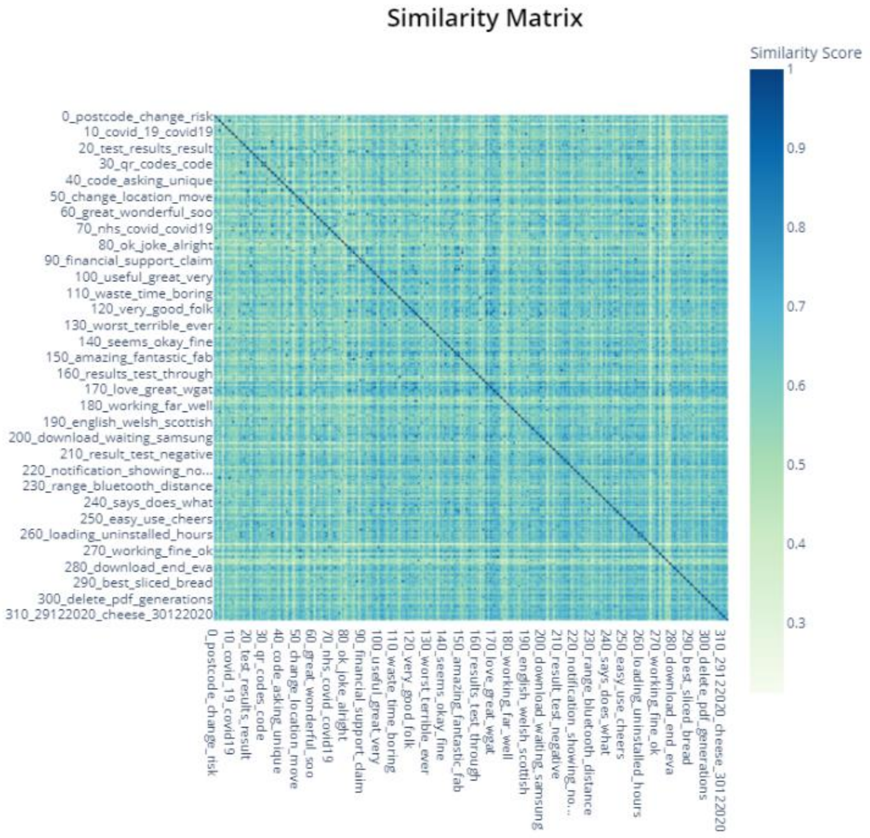


Figure 22: Heat-map showing similarity in topics for BERTopic trained using default hyper-parameter values

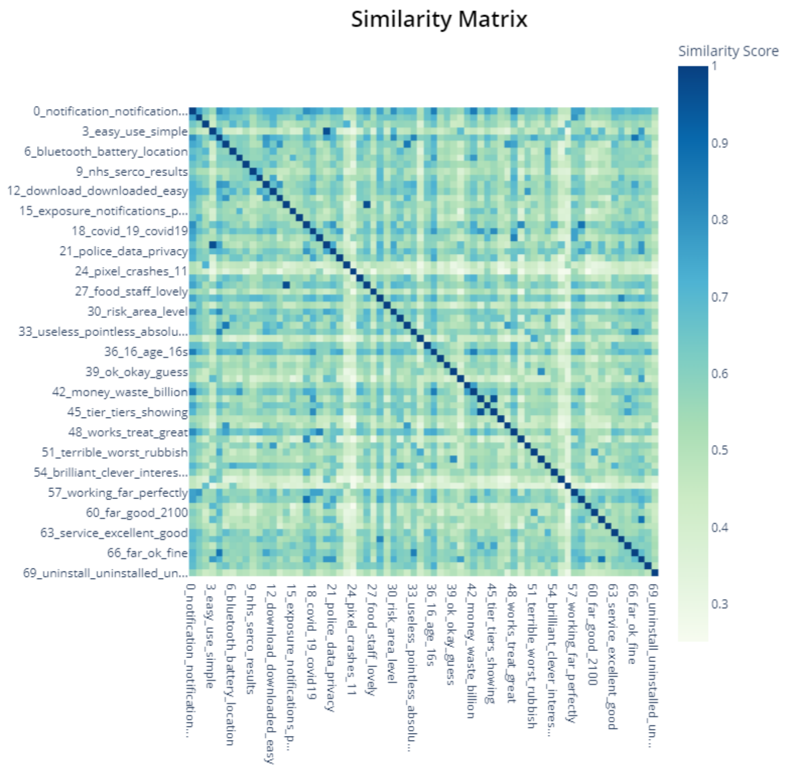


Figure 23: Heat-map showing similarity in topics for BERTopic trained using tuned hyper-parameter values

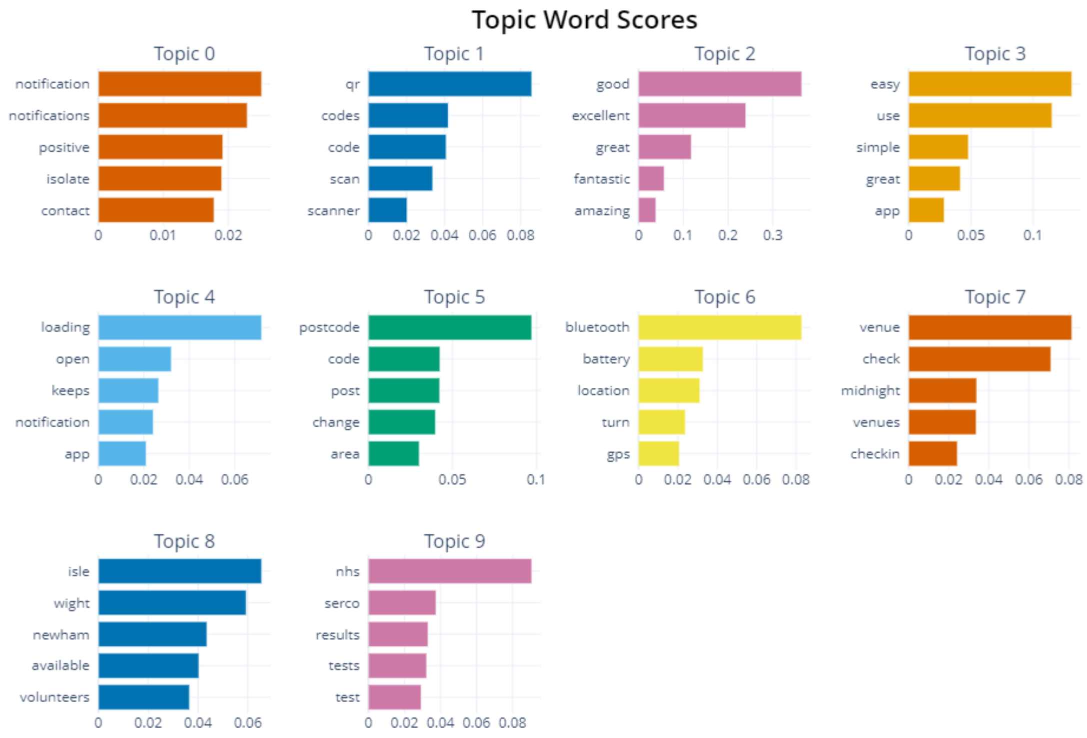


Figure 24: Top 10 Topics for BERTopic after tuning the hyper-parameters

4.5.2.1 Hyperparameters for Topic Extraction Model

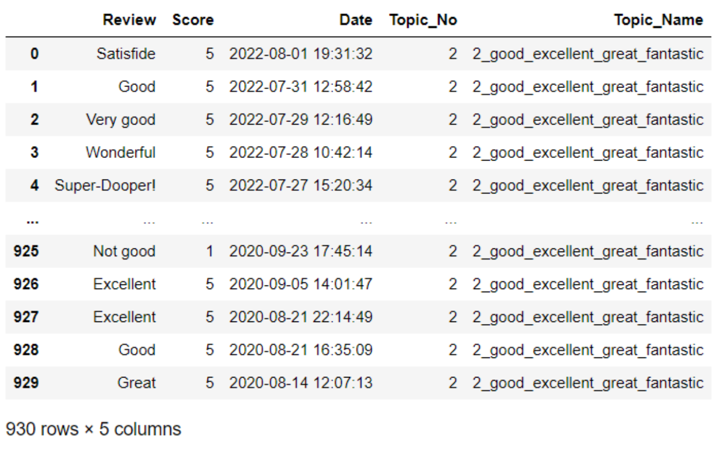
Let's discuss some hyper-parameter needed to be tuned to get good results from the BERTopic:

- n-gram_range: This is a range of values to decide the sequence of consecutive words in a text/document. The creator of BERTopic (Grootendorst, 2022) recommends the value to range from (1,3), meaning the words in a text will be converted to unigram, bigram and trigram as well.
- min_topic_size: This parameter decides the minimum number of documents in each topic. Increasing this value will reduce the number of clusters. After several hits and trials, this value was found to give desirable results at the value equal to 30.
- nr_topics: This parameter reduces the maximum number of clusters possible. This reduction is slow as each reduction (-1) activates a c-TF-IDF computation. This value we have set to 'auto' to allow HDBSCAN to reduce topics automatically.

4.5.3 Topic-wise temporal sentiment Analysis

We have seen the steps to train the sentiment classifier model and topic extraction till now. Now to perform Topic-wise temporal sentiment classification, the steps are as follows:

- For each topic, create a separate data frame containing the user reviews ('content' column) belonging to that topic, the date at which the review was posted ('at' column), and the rating ('score' column) given by the user to the NHS COVID-19 App. For instance, for topic 2 in our project, we have the following data frame (figure 25).

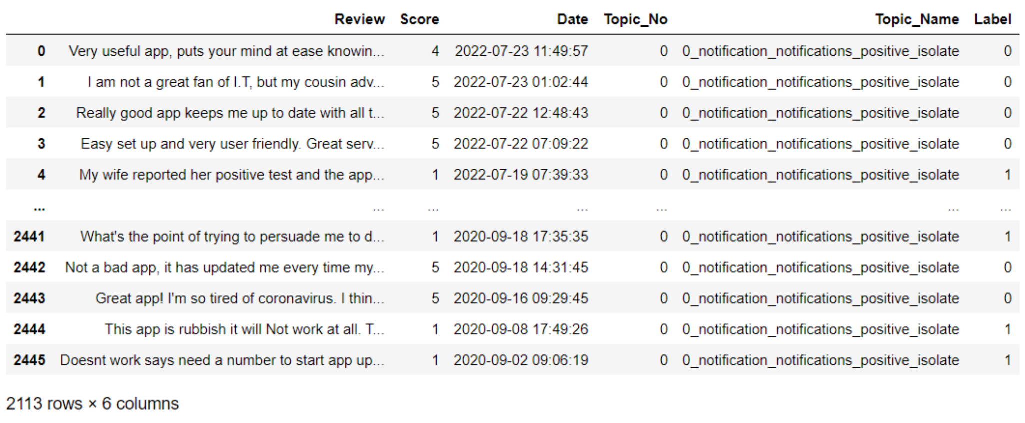


	Review	Score	Date	Topic_No	Topic_Name
0	Satisfide	5	2022-08-01 19:31:32	2	2_good_excellent_great_fantastic
1	Good	5	2022-07-31 12:58:42	2	2_good_excellent_great_fantastic
2	Very good	5	2022-07-29 12:16:49	2	2_good_excellent_great_fantastic
3	Wonderful	5	2022-07-28 10:42:14	2	2_good_excellent_great_fantastic
4	Super-Dooper!	5	2022-07-27 15:20:34	2	2_good_excellent_great_fantastic
...	...	...	...	...	...
925	Not good	1	2020-09-23 17:45:14	2	2_good_excellent_great_fantastic
926	Excellent	5	2020-09-05 14:01:47	2	2_good_excellent_great_fantastic
927	Excellent	5	2020-08-21 22:14:49	2	2_good_excellent_great_fantastic
928	Good	5	2020-08-21 16:35:09	2	2_good_excellent_great_fantastic
929	Great	5	2020-08-14 12:07:13	2	2_good_excellent_great_fantastic

930 rows × 5 columns

Figure 25: One of the data frames representing a topic along with the user reviews associated with this topic

- Remove those user reviews from each data frame for which the 'score' column has a value of 3. So only those user reviews are kept with a score of 1, 2, 4 and 5. This removes the possibility of neutral reviews, as we focus only on positive and negative reviews.
- Using the 'score' column, we will create a sentiment label for each user review on each data frame. If the 'score' column value is equal to 1 or 2, we keep the label as 0 (positive sentiment) for the user review, and if the 'score' column value is 4 or 5 for a review, we keep the value of the label as 1 (negative sentiment). For instance, figure 26 below shows the data frame of topic number 0, and the 'label' column is added to it using the 'score' column.



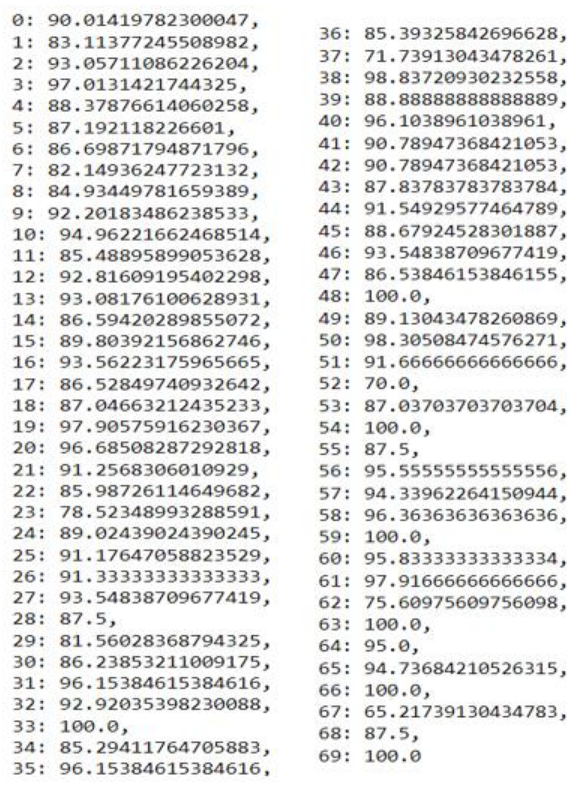
	Review	Score	Date	Topic_No	Topic_Name	Label
0	Very useful app, puts your mind at ease knowin...	4	2022-07-23 11:49:57	0	0_notification_notifications_positive_isolate	0
1	I am not a great fan of I.T, but my cousin adv...	5	2022-07-23 01:02:44	0	0_notification_notifications_positive_isolate	0
2	Really good app keeps me up to date with all t...	5	2022-07-22 12:48:43	0	0_notification_notifications_positive_isolate	0
3	Easy set up and very user friendly. Great serv...	5	2022-07-22 07:09:22	0	0_notification_notifications_positive_isolate	0
4	My wife reported her positive test and the app...	1	2022-07-19 07:39:33	0	0_notification_notifications_positive_isolate	1
...	...	...	...	...	...	...
2441	What's the point of trying to persuade me to d...	1	2020-09-18 17:35:35	0	0_notification_notifications_positive_isolate	1
2442	Not a bad app, it has updated me every time my...	5	2020-09-18 14:31:45	0	0_notification_notifications_positive_isolate	0
2443	Great app! I'm so tired of coronavirus. I thin...	5	2020-09-16 09:29:45	0	0_notification_notifications_positive_isolate	0
2444	This app is rubbish it will Not work at all. T...	1	2020-09-08 17:49:26	0	0_notification_notifications_positive_isolate	1
2445	Doesnt work says need a number to start app up...	1	2020-09-02 09:06:19	0	0_notification_notifications_positive_isolate	1

2113 rows × 6 columns

Figure 26: Label created using the Score Column for topic 0

- For each data frame, we will apply the trained sentiment classifier model to predict the sentiment of the user reviews as positive or negative.

- We check the accuracy of the predicted values of the sentiment model by comparing to this label for each data frame. For each of the 70 topics (numbered from 0 to 69), the accuracy (%) of the sentiment classification model on the user reviews per topic is as follows:



```
0: 90.01419782300047,
1: 83.11377245508982,
2: 93.05711086226204,
3: 97.0131421744325,
4: 88.37876614060258,
5: 87.192118226601,
6: 86.69871794871796,
7: 82.14936247723132,
8: 84.93449781659389,
9: 92.20183486238533,
10: 94.96221662468514,
11: 85.48895899053628,
12: 92.81609195402298,
13: 93.08176100628931,
14: 86.59420289855072,
15: 89.80392156862746,
16: 93.56223175965665,
17: 86.52849740932642,
18: 87.04663212435233,
19: 97.90575916230367,
20: 96.68508287292818,
21: 91.2568306010929,
22: 85.98726114649682,
23: 78.52348993288591,
24: 89.02439024390245,
25: 91.17647058823529,
26: 91.33333333333333,
27: 93.54838709677419,
28: 87.5,
29: 81.56028368794325,
30: 86.23853211009175,
31: 96.15384615384616,
32: 92.92035398230088,
33: 100.0,
34: 85.29411764705883,
35: 96.15384615384616,
```

```
36: 85.39325842696628,
37: 71.73913043478261,
38: 98.83720930232558,
39: 88.88888888888889,
40: 96.1038961038961,
41: 90.78947368421053,
42: 90.78947368421053,
43: 87.83783783783784,
44: 91.54929577464789,
45: 88.67924528301887,
46: 93.54838709677419,
47: 86.53846153846155,
48: 100.0,
49: 89.13043478260869,
50: 98.30508474576271,
51: 91.66666666666666,
52: 70.0,
53: 87.03703703703704,
54: 100.0,
55: 87.5,
56: 95.55555555555556,
57: 94.33962264150944,
58: 96.36363636363636,
59: 100.0,
60: 95.83333333333334,
61: 97.91666666666666,
62: 75.60975609756098,
63: 100.0,
64: 95.0,
65: 94.73684210526315,
66: 100.0,
67: 65.21739130434783,
68: 87.5,
69: 100.0
```

Figure 27: Accuracy of sentiment classifier of user reviews belonging to a specific topic

- Finally, for each data frame, we will create two trendlines. The X-axis will represent the months, and the Y-axis will represent the sum of positive sentiments for each month (positive sentiment trend) or the negative sentiment for each month (negative sentiment trend). So, in total, we get 70 plots, and the plots of all the 70 topics are shown in figure 28, figure 29, figure 30 and figure 31. In these plots, the red line shows the trend for negative sentiment, and the black line indicates the trend for positive sentiment

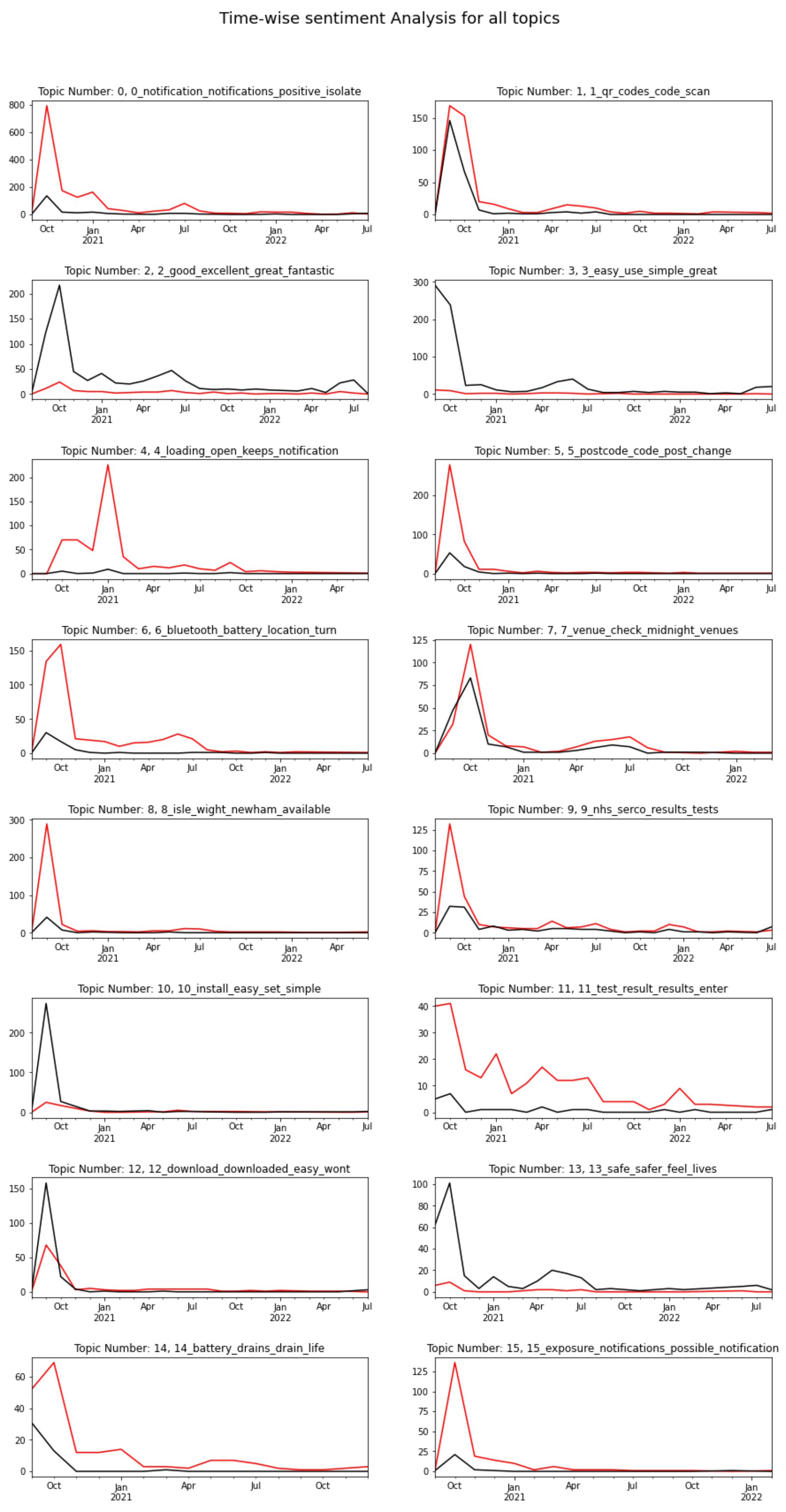


Figure 28: Topic-wise Sentiment trends for topics 0 to 15. The Red line represents negative sentiments, and the Black line Represents positive sentiments.

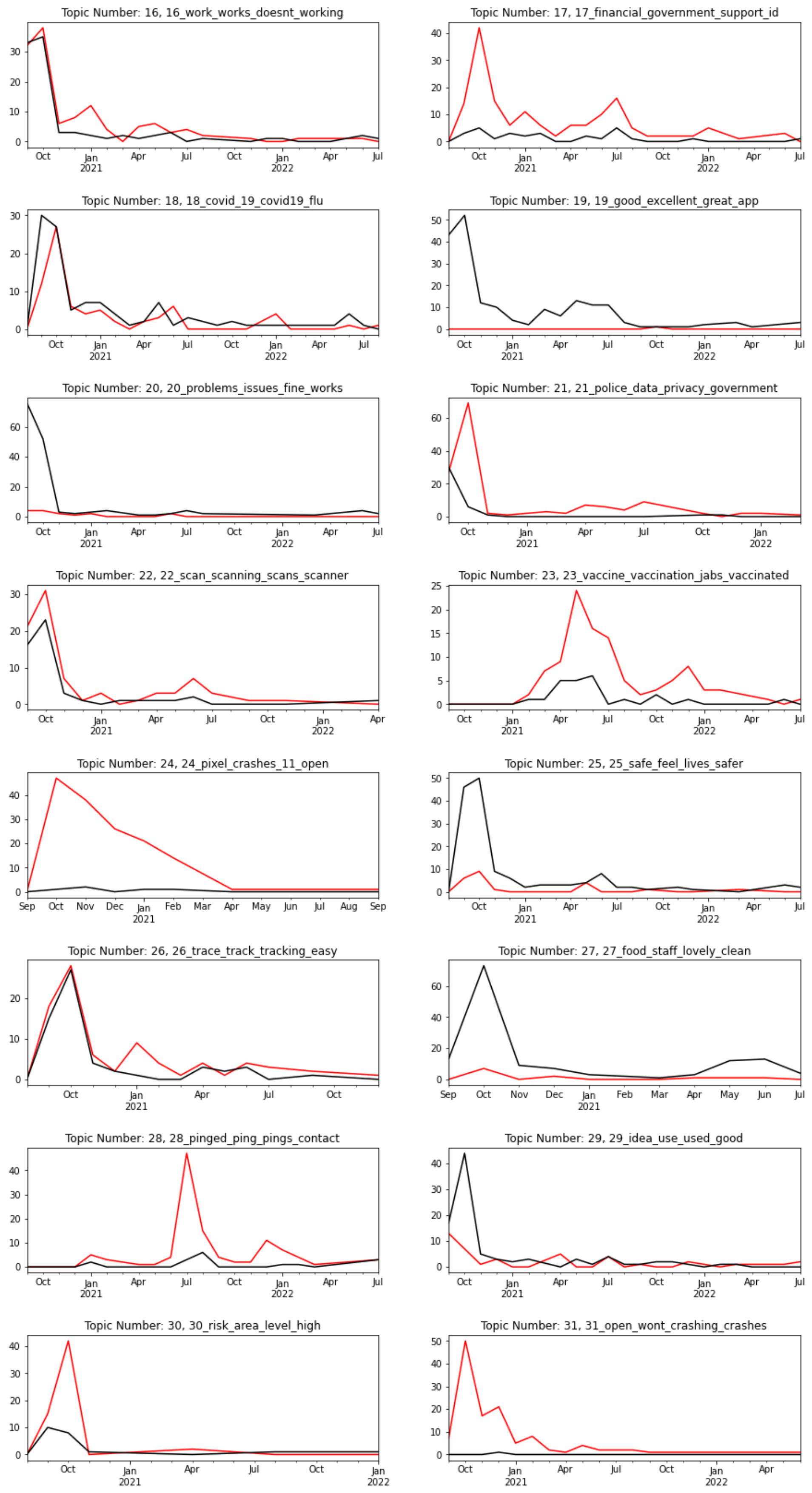


Figure 29: Topic-wise Sentiment trends for topics 16 to 31. The Red line represents negative sentiments, and the Black line Represents positive sentiments.

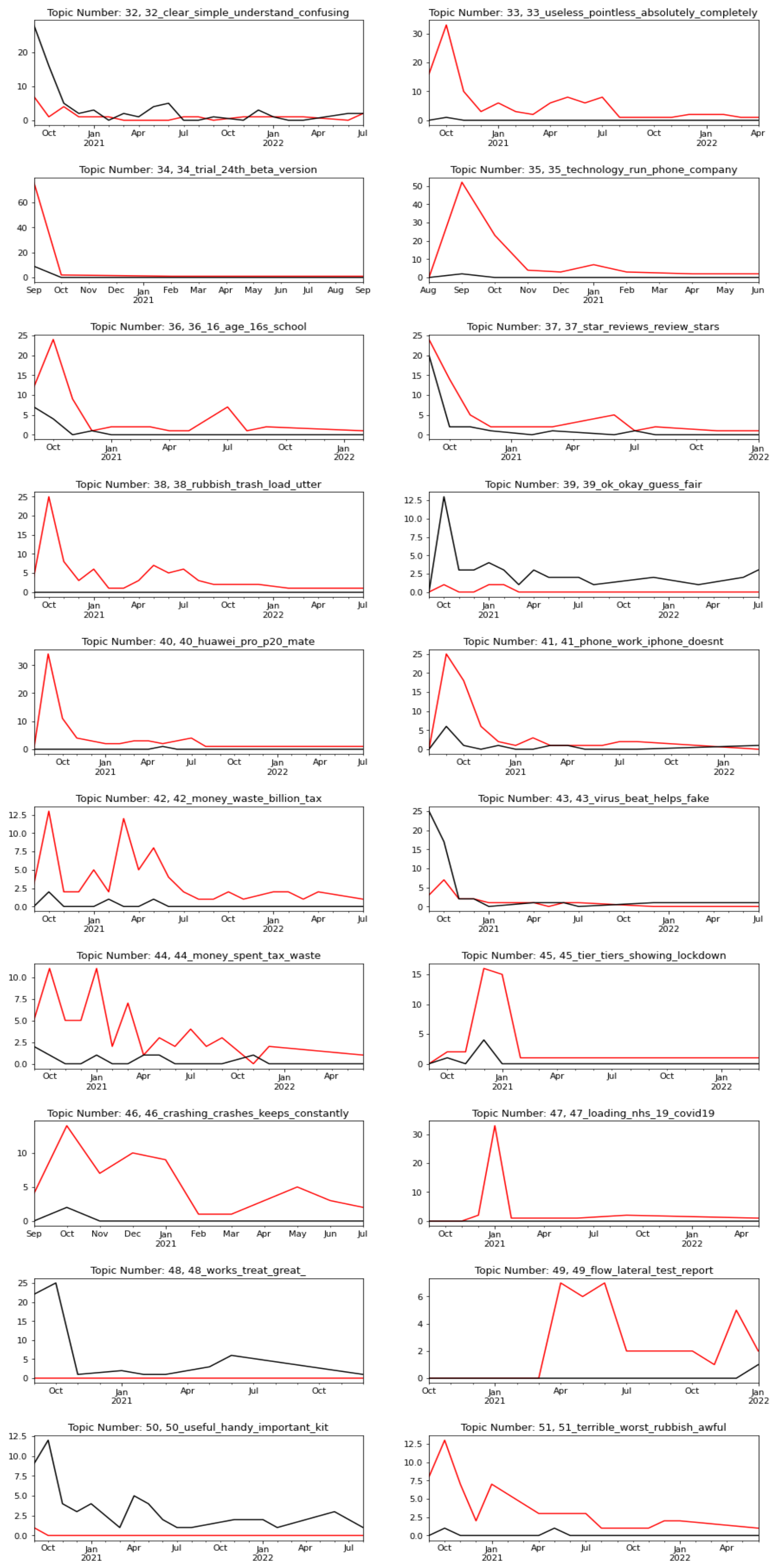


Figure 30: Topic-wise Sentiment trends for topics 32 to 51. The Red line represents negative sentiments, and the Black line Represents positive sentiments.

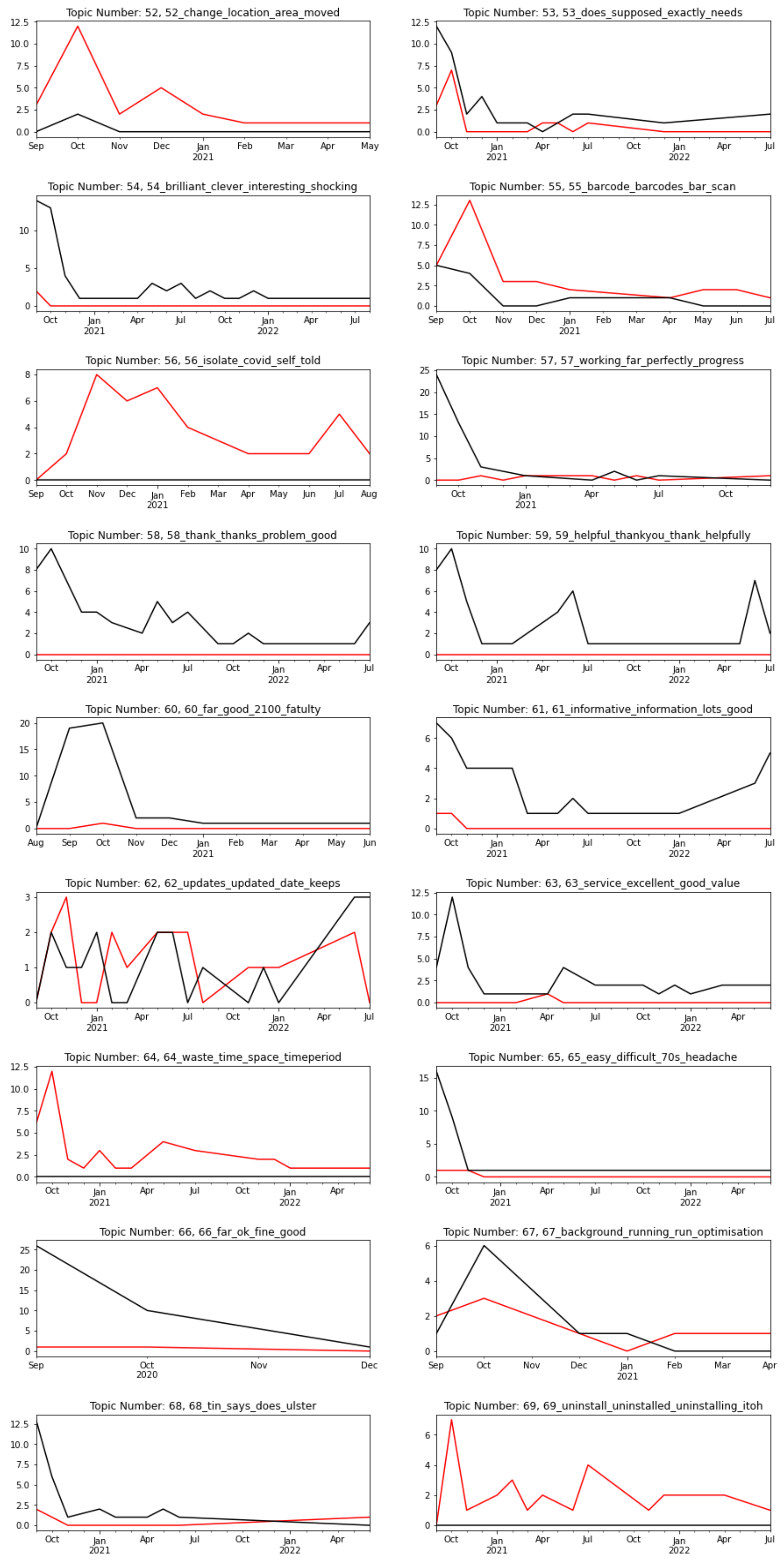


Figure 31: Topic-wise Sentiment trends for topics 52 to 69. The Red line represents negative sentiments, and the Black line Represents positive sentiments

CHAPTER 5: DISCUSSION

In chapter 4, the steps of building the project have been explained in detail, along with the results this project accomplished. In this section, we will analyze and explore these results and achievements. We will also discuss some common doubts that might arise in the minds of the readers of this paper, such as how a model trained on a different dataset performs well on another dataset. Then finally, at the end of this section, the limitations and deficiencies, and discuss what more could have been done if more time was available.

In section 4.4.2, we saw that the sentiment classifier model had achieved an accuracy of 97.31 % on the validation dataset and 94.06 % on the test dataset. These accuracies are excellent, and as the models are performing well on both validation and test datasets, the model is not overfitting. Also, one might wonder why the F1 score, precision, and recall are not calculated for this sentiment classifier model. The reason is that the F1 score is needed only when the dataset the machine learning model is being trained on is biased toward a specific label (0-Positive and 1-Negative in this project). Dataset1_Train (explained in section 4.1.1) has 11906 reviews with class label 1 (Negative Sentiment) and 10380 with class label 0 (Positive Sentiment), which means the data is balanced and unbiased.

In section 4.5, we showed the process of topic extraction on Dataset 2 (explained in section 4.1.2), which contains the user reviews of the NHSCOVID-19 app and managed to get 70 topics. We also saw the process of segregating the user reviews into different data frames (each data frame is a unique topic), as mentioned in section 4.5.3. Then, for user reviews in each data frame (representing the topic), we use the user rating to create a label (0 for good and 1 for negative) and check the accuracy of the sentiment model, and get the accuracies as shown in figure 27. We achieved good accuracies for almost all the data frames (where each topic represents one topic).

But how can a model be trained on a different dataset (Dataset 1 explained in section 4.1.1) containing user reviews of COVID-19 contact tracing apps from 46 countries and predict well on another dataset (Dataset 2-scraped user reviews of NHS COVID-19 app from Google Play store explained in section 4.1.2)? There are two reasons for it which are as follows:

- First, the model used to create a sentiment classifier model uses BERT, a pre-trained language model. It is pre-trained on a huge corpus of data, and we used the pre-trained model to fine-tune on Dataset 1, which, although not precisely the same as Dataset 2, is talking about similar issues and has a similar context. BERT Models can understand the semantic relationships unlike classical machine learning models and hence can be used for predictions on data slightly different than the ones they are trained on. Therefore BERT sentiment classifier models can predict the sentiment of new data, which might be somewhat different from the ones it is trained on with high accuracy.
- The second reason is that when the labels were created for the NHS COVID-19 app using user rating ('score' column in figure 26) for each topic, we removed the user reviews with a score rating of 3 as most of the reviews were neutral. As our sentiment classifier is trained on only positive and negative reviews and to achieve better results, neutral reviews were removed from the user reviews of the NHSCOVID-19 App while predicting sentiment.

As shown in figure 27, there are some topics for which the sentiment classifier did not achieve high accuracy. These are more important than the ones where the accuracy is good because when the accuracy is good, the label created using the user rating ('score' column) for the NHS COVID-19 App is sufficient on its own to predict the sentiment. But when the accuracy is low, there could be two reasons for it. They are:

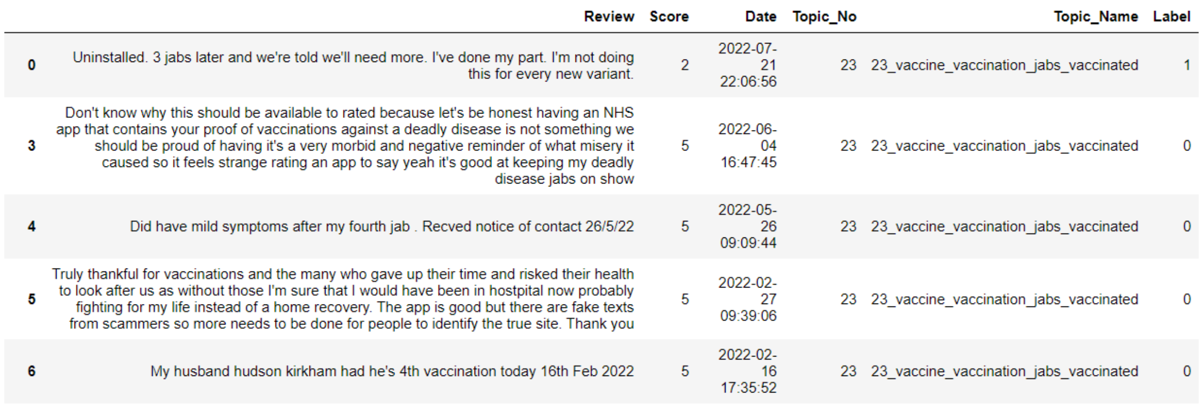
- The BERT sentiment classifier model was making incorrect predictions. But as our model was trained and tested with flying colours, this is highly unlikely, although more analysis will be needed to check this.
- When used to create a label, the user rating does not tell the complete story. It means that although the sentiment of the user review was negative and predicted correctly by the Sentiment classifier model, the rating given by that user contradicts the sentiment of the user review. In figure 32, if we read the text in the second row with a user rating ('score' column) of 5, we can easily see that the review's sentiment is negative and contradicts the user rating. Also in figure 23, The plot for topic 23 shows high rate of negative sentiment. So if the sentiment classifier is predicting most of the reviews correctly, it means user reviews sentiment are negative but the user rating is high. Hence to understand the true meaning of the review given by the user, just a user rating is not sufficient.

In this project, in the end, graphs were plotted to show trends of positive and negative sentiment for each topic, and the results are shown in figures 28, 29, 30, and 31, where the red line is for negative sentiment and the black line is for positive sentiment. These four figures show all 70 plots along with the topic number and name. Although self-explanatory, these plots might raise questions such as why most plots show a decreasing trend over time. There are two prime reasons for it. First is that most people downloaded the application at the beginning of the Coronavirus pandemic and the rate of people posting reviews decreased with time. The second reason will be people stopped having issues with the application as the NHS COVID-19 Developers constantly updated the application.

It is interesting to note that in figure 31, for the plot with Topic number 62, which is represented by the words ‘updates’, ‘updated’, ‘date’, and ‘keeps’, the trendlines for negative and positive sentiment for topic 62 keep fluctuating with time and do not have a decreasing trend. It means that the application had regular updates. It also means some people were happy with the updates, and some were not. Each plot has a story behind it and needs to be explored further, but due to time constraints on this project, we only managed to create plots and perform fundamental analysis.

One of the essential points to be noted while exploring these plots is that some topics are not showing a decreasing trend even after knowing that the number of downloads will decrease with time. Also, some topics had peaks, not at the beginning but later, such as Topic Number 49, 61, 56, 28, and 59. Even in the topics showing a decreasing trend in sentiments, there are plots with a sudden decrease in positive or negative trends, such as Topic Number 5,10 and 12, and some plots where the plots decrease slowly for either positive or negative sentiment, such as 44, 58, and 33, and 46.

We must focus on topics for which the sentiment classifier achieved low accuracy. For instance, in figure 27, we can see the sentiment classifier achieved an accuracy of 78 % for topic number 23. When we look at its plot in figure 29, we see that the negative sentiments are more significant than the negative sentiment. But we assume that the sentiment classifier has accurately predicted the sentiments of most user reviews. Then, in that case, it means the user rating contradicts the sentiment of the user review, which can be confirmed from the image below.



	Review	Score	Date	Topic_No	Topic_Name	Label
0	Uninstalled. 3 jabs later and we're told we'll need more. I've done my part. I'm not doing this for every new variant.	2	2022-07-21 22:06:56	23	23_vaccine_vaccination_jabs_vaccinated	1
3	Don't know why this should be available to rated because let's be honest having an NHS app that contains your proof of vaccinations against a deadly disease is not something we should be proud of having it's a very morbid and negative reminder of what misery it caused so it feels strange rating an app to say yeah it's good at keeping my deadly disease jabs on show	5	2022-06-04 16:47:45	23	23_vaccine_vaccination_jabs_vaccinated	0
4	Did have mild symptoms after my fourth jab . Recved notice of contact 26/5/22	5	2022-05-26 09:09:44	23	23_vaccine_vaccination_jabs_vaccinated	0
5	Truly thankful for vaccinations and the many who gave up their time and risked their health to look after us as without those I'm sure that I would have been in hostpital now probably fighting for my life instead of a home recovery. The app is good but there are fake texts from scammers so more needs to be done for people to identify the true site. Thank you	5	2022-02-27 09:39:06	23	23_vaccine_vaccination_jabs_vaccinated	0
6	My husband hudson kirkham had he's 4th vaccination today 16th Feb 2022	5	2022-02-16 17:35:52	23	23_vaccine_vaccination_jabs_vaccinated	0

Figure 32: Dataframe of user reviews for topic 23 with user rating (score), the date on which review was posted (Date), Topic Number, Topic Name and Label

It can be seen in the 32 that although class labels are user rating of the review is 5 in the second row, the sentiment of the text is negative for topic number 23, which is the reason for low accuracy. It means if we have achieved a lower accuracy in the sentiment classifier when the user rating is used to create positive/negative class labels, the user rating must contradict the sentiment in the user review and hence cannot tell the complete story or meaning of the user review.

5.1 Future Work

Although this project has accomplished its goal of creating a pipeline for Topic wise Temporal Sentiment Analysis, a few things could not be completed. In our project, we have removed the neutral reviews from the user reviews of the NHS COVID-19 App. For this project, neutral reviews had a user rating of 3. In the future, if possible, this project can try to include neutral reviews. Also, this project managed to create plots for topic-wise temporal sentiment analysis, which are self-explanatory, needs a detailed study needs to be done to reveal more insights into the topics.

CHAPTER 6: CONCLUSION

This project aimed to analyze the topics and topic-wise sentiment trends of the user reviews of the NHS COVID-19 application on the Google Play store. The dates of the user reviews ranged from August 2020 to August 2022. To achieve this, we trained a sentiment classifier model to predict whether the sentiment is positive or negative. Then we used a Topic Extraction method which involved grouping user reviews within topics. Then finally, for user reviews under each topic, we created two trendlines (one for positive sentiment and the other for negative sentiment) where one trendline shows the sum of positive sentiments per month and the other trendline shows the sum of negative sentiments per month. Finally, we discussed the doubts that arose after analysing the topic-wise sentiment trends and also mentioned the limitations of the project.

This project successfully combined the concepts of Topic Extraction and Sentiment Analysis to make the Topic-wise Temporal Sentiment Classification model and visualise the results. This project also showed that connecting both fields give a much deeper understanding of the text than just performing topic extraction or sentiment analysis.

REFERENCES

Agarwal, A., Baechle, C., Behara, R. & Zhu, X., 2018. A natural language processing framework for assessing hospital readmissions for patients with COPD. IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS, 22(2), pp. 588-596.

Ahmad, K. et al., 2021. A Benchmark Dataset for Sentiment Analysis of Users' Reviews on COVID-19 Contact Tracing Applications, s.l.: Harvard Dataverse.

Ahmad, K. et al., 2021. Sentiment Analysis of Users' Reviews on COVID-19 Contact Tracing Apps with a Benchmark Dataset. arXiv.org, 1 March.

Ahmed, N. et al., 2020. A survey of COVID-19 contact tracing apps. IEEE Access, Volume 8, p. 134577–134601.

Anon., 2019. A Comprehensive Guide to Attention Mechanism in Deep Learning for Everyone. [Online] Available at: https://www.analyticsvidhya.com/blog/2019/11/comprehensive-guide-attention-mechanism-deep-learning/

Antypas, D., Rogers, D., Preece, A. & Camacho-Collados, J., 2021. COVID-19 and Misinformation: A Large-Scale Lexical Analysis on Twitter. s.l., Association for Computational Linguistics, pp. 119-126.

Apple, 2008. App Store. [Online], Available at: https://www.apple.com/uk/app-store/

Babu, Y. P. & Eswari, R., 2020. CIA_NITT at WNUT-2020 Task 2: Classification of COVID-19 Tweets Using Pre-trained Language Models. s.l., Association for Computational Linguistics, Online, p. 471–474.

Bahdanau, D., Cho, K. & Bengio, Y., 2015. Neural Machine Translation by Jointly Learning to Align and Translate. San Diego, CA, USA, open access.

Bang, Y. et al., 2021. Model Generalization on COVID-19 Fake News Detection. arxiv.

Bengio, Y. et al., 2021. Inherent privacy limitations of decentralized contact tracing apps. Journal of the American Medical Informatics Association, 28(1), p. 193–195.

Berba, P., 2020. A gentle introduction to HDBSCAN and density-based clustering. [Online] Available at: https://pberba.github.io/stats/2020/07/08/intro-hdbscan/

Bianchi, F., Terragni, S. & Hovy, D., n.d. Pre-training is a Hot Topic: Contextualized Document Embeddings Improve Topic Coherence. Online, Association for Computational Linguistics, p. 759–766.

Buduma, N., 2015. Fundamentals of Deep Learning: Designing Next-Generation Artificial Intelligence Algorithms: Designing Next-Generation Machine Intelligence Algorithms. s.l.:O'Reilly Media.

Campello, R., Moulavi, D. & Sander, J., 2013. Density-Based Clustering Based on Hierarchical Density Estimates. In: Advances in Knowledge Discovery and Data Mining. PAKDD 2013. Lecture Notes in Computer Science(). s.l.:Springer, Berlin, Heidelberg.

Cer, D. et al., 2018. Universal Sentence Encoder. arXiv.

Chen, L. et al., 2020. In the Eyes of the Beholder: Analyzing Social Media Use of Neutral and Controversial Terms for COVID-19. arXiv.

Cho, H., Ippolito, D. & Yu, Y. W., 2020. Contact tracing mobile apps for COVID-19: Privacy considerations and related tradeoffs. arxiv.prg.

Cho, K. et al., 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. Doha, Qatar, Association for Computational Linguistics, p. 1724–1734.

Conneau, A. et al., 2020. Unsupervised Cross-lingual Representation Learning at Scale. s.l., Association for Computational Linguistics., p. 8440–8451.

Dangeti, P., 2017. Statistics for Machine Learning [WWW Document]. s.l.:Packt.

Dave, K. S. & V. V., 2010. Pattern based keyword extraction for contextual advertising. New York, New York, USA, ACM Press, p. 1885–1888.

Delangue, C. & Chaumond, J., n.d. DistilBERT. [Online], Available at: https://huggingface.co/docs/transformers/model_doc/distilbert

Delangue, C. & Chaumond, J., n.d. Transformers Documentation - Trainer. [Online], Available at: https://huggingface.co/docs/transformers/main/en/main_classes/trainer#transformers.TrainingArguments.per_device_eval_batch_size

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K., 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Minneapolis, Minnesota, Association for Computational Linguistics, p. 4171–4186.

Dietterich, T. G., 1995. Overfitting and undercomputing in machine learning. ACM Computing Surveys, 27(3), p. 326–32.

Donahue, J. et al., 2017. Long-Term Recurrent Convolutional Networks for Visual Recognition and Description. s.l., IEEE Xplore, pp. 677-691.

Fahim, A., Salem, . A., Torkey, F. & RAMADAN, M., 2006. An efficient enhanced k-means clustering algorithm. Journal of Zhejiang University Science A, Volume 7, p. 1626–1633.

Fawcett, T., 2006. An introduction to ROC analysis. Pattern Recognition Letters, 27(8), pp. 861-874.

Garousi, V., Cutting, D. & Felderer, M., 2020. Mining user reviews of COVID contact-tracing apps:An. arxiv e-print.

Gharavi, E., Nazemi, N. & Dadgostari, F., 2020. Early Outbreak Detection for Proactive Crisis Management Using Twitter Data: COVID-19 a Case Study in the US. arxiv.org.

Giachanou, A. & Crestani, F., 2016. Like It or Not: A Survey of Twitter Sentiment Analysis Methods. ACM Computing Surveys, 49(2), pp. Article number 28, pp. 1-41.

Google, 2012. Google Play Store. [Online], Available at: https://play.google.com/store

Google, 2022. Transfer learning and fine-tuning Tutorial. [Online], Available at: https://www.tensorflow.org/tutorials/images/transfer_learning

Grootendorst, M., 2022. BERTopic Website. [Online], Available at: https://maartengr.github.io/BERTopic/index.html

Grootendorst, M., 2022. BERTopic: Neural topic modeling with a class-based TF-IDF procedure. arXiv.

Grootendorst, M., 2022. reference for c-TF-IDF process. [Online], Available at: https://maartengr.github.io/BERTopic/api/ctfidf.html

Grootendorst, M., n.d. BERTopic Algorithm. [Online], Available at: https://maartengr.github.io/BERTopic/algorithm/algorithm.html

Han, Y. et al., 2009. The Improved Logistic Regression Models for Spam Filtering. s.l., IEEE, pp. 314-317.

Hearst, M. A. et al., 1998. Support vector machines. IEEE Intelligent Systems and their applications, 13(4), p. 18–28.

He, H. & Garcia, E. A., 2009.. Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), p. 1263–1284.

Ho, C. C., Baharim, K. N., Abdulsalam, A. & Alias, M. S. B., 2017. Deep Neural Networks for Text: A Review. Langkawi, Malaysia, s.n.

Horev, R., 2018. BERT Explained: State of the art language model for NLP. [Online], Available at: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270#:~:text=How%20BERT%20works,a%20prediction%20for%20the%20task.

Hutto, C. & Gilbert, E., 2014. VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text. Proceedings of the International AAAI Conference on Web and Social Media, 8(1), p. 216–225.

Jawa, V. & Anand, M., 2022. Accelerating Topic modeling with RAPIDS and BERT models. [Online], Available at: https://medium.com/rapids-ai/accelerating-topic-modeling-with-rapids-and-bert-models-be9909eeed2

Jelodar, H. et al., 2019. Latent Dirichlet allocation (LDA) and topic modeling: models, applications, a survey. Multimed Tools Appl, Volume 78, p. 15169–15211.

Jurafsky, D. & Martin, J. H., 2014. SPEECH AND LANGUAGE PROCESSING. s.l.:Pearson.

Kalyan, K. S. & Sangeetha, S., 2020. SECNLP: A survey of embeddings in clinical natural language processing. Journal of Biomedical Informatics, p. 101.

Kalyan, K. S. & Sangeetha, S., 2021. BertMCN: Mapping colloquial phrases to standard medical concepts using BERT and highway network. Artificial Intelligence In Medicine, p. 112.

Kingma, D. . P. & Ba, J. L., 2014. Adam: A Method for Stochastic Optimization. arXiv.

Lalmuanawma, S., Hussain, J. & Chhakchhuak, L., 2020. Applications of machine learning and artificial intelligence for. Chaos, Solitons & Fractals, Volume 129, p. 110059.

Lash, R. R. et al., 2020. COVID-19 Contact Tracing in Two Counties — North Carolina, June–July 2020. s.l., PubMed Central, pp. 1360-1363.

Latif, S. et al., 2020. Leveraging Data Science to Combat COVID-19:. IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE, Volume Vol. 1, No. 1, pp. 85-103.

LeCun, Y., Bengio, Y. & Hinton, G., 2015. Deep learning. Nature, Volume 521, p. 436–444.

Li, J. & Guo, X., 2020. COVID-19 contact-tracing apps: A survey on the global deployment and challenges. arxiv.org.

Liu, Y. et al., 2019. RoBERTa: A robustly optimized bert pretraining approach. arXiv.

López, F. R., Jiménez-Salazar, H. & Pinto , D., 2007. A Competitive Term Selection Method for Information Retrieval. In: Computational Linguistics and Intelligent Text Processing. CICLing 2007. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer.

Mbunge, E., 2020. Integrating emerging technologies into COVID-19 contact tracing: Opportunities, challenges and pitfalls.. Diabetes & Metabolic Syndrome: Clinical Research & Reviews, 16(6), p. 1631–1636.

McInnes, L., Healy, J. & Astels , S., n.d. How HDBSCAN Works. [Online], Available at: https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html#:~:text=HDBSCAN%20is%20a%20clustering%20algorithm,in%20the%20stability%20of%20clusters.

McInnes, L., Healy, J. & Melville, J., 2018. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv.

McKinney, W., n.d. pandas documentation. [Online], Available at: https://pandas.pydata.org/docs/

Medhat, W., Hassan, A. & Korashy, H., 2014. Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal, 5(4), pp. 1093-1113.

Mikolov, T. et al., 2013. Distributed Representations of Words and Phrases and their Compositionality. Lake Tahoe, Nevada, Curran Associates Inc., p. 3111–3119.

Mohri, M., Rostamizadeh, A. & Talwalkar, . A., 2018. Foundations of Machine Learning, second edition, Adaptive Computation and Machine Learning series. s.l.:MIT Press.

Nabity-Grover, T., Cheung, C. M. & Thatcher, J. B., 2020. Inside out and outside in: How the COVID-19 pandemic affects self-disclosure on social media. International Journal of Information Management, 55(102188).

Na, S., Xumin, L. & yong, G., 2010. Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm. s.l., s.n., p. 63–67.

Nguyen, D. Q., Billingsley, R., Du, L. & Johnson, M., 2015. Improving Topic Models with Latent Feature Word Representations. Transactions of the Association for Computational Linguistics 2015, Volume 3, p. 299–313.

O'Callaghan, M. E. et al., 2020. A national survey of attitudes to COVID-19 digital contact tracing in the Republic of Ireland. Irish Journal of Medical Science, 190(3), pp. 863-887.

Oak, M. et al., 2016. Generating clinically relevant texts: A case study on life-changing events. San Diego, California, Association for Computational Linguistics, p. 85–94.

Oliphant, T., n.d. NumPy documentation. [Online], Available at: https://numpy.org/doc/stable/index.html

Oyebode, O., Alqahtani, F. & Orji, R., 2020. Using Machine Learning and Thematic Analysis Methods to Evaluate Mental Health Apps Based on User Reviews. IEEE Access, Volume 8, pp. 2169-3536.

Oyebode, O. et al., 2022. COVID-19 Pandemic: Identifying Key Issues Using Social Media and Natural Language Processing. Journal of Healthcare Informatics Research, Volume 6, p. 174–207.

P., V., 2021. Word2Vec Explained. [Online], Available at: https://towardsdatascience.com/word2vec-explained-49c52b4ccb71

Paltoglou, G. & Thelwall, M., 2010. A Study of Information Retrieval Weighting Schemes for Sentiment Analysis. Uppsala, Sweden, Association for Computational Linguistics, p. 1386–1395.

Park, A. & Conway, M., 2017. Tracking Health Related Discussions on Reddit for Public Health Applications. Published online, Pubmed Central, p. 1362–137.

Peltarion, n.d. Peltarion Knowledge Center. [Online], Available at: https://peltarion.com/knowledge-center/modeling-view/build-an-ai-model/blocks/multilingual-bert

Porter, M., 1980. An algorithm for suffix stripping. s.l., s.n., pp. 130-137.

POWERS, D., 2011. EVALUATION: FROM PRECISION, RECALL AND F-MEASURE TO ROC, INFORMEDNESS, MARKEDNESS & CORRELATION. Journal of Machine Learning Technologies, 2(1), pp. 37-63.

Preece, A. et al., 2018. Sentinel: A Codesigned Platform for Semantic Enrichment of Social Media Streams. IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, Volume VOL. 5, NO. 1, pp. 118-131.

Python Software Foundation, n.d. re — Regular expression operations. [Online], Available at: https://docs.python.org/3/library/re.html

Qayyum, A. et al., 2021. Collaborative Federated Learning For Healthcare: Multi-Modal COVID-19 Diagnosis at the Edge. arxiv.

Reichert, L., Brack, S. & Scheuermann, B., 2020. Privacy-Preserving Contact Tracing of COVID-19 Patients. Cryptology ePrint Archive, p. Paper 2020/375.

Rekanar, K. et al., 2020. Sentiment Analysis of User Feedback on the HSE Contact Tracing App. Research Square.

Rogers, S. & Girolami, M., 2016. A First Course in Machine Learning. s.l.:CRC Press.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J., 1986. Learning representations by back-propagating errors. Nature, Volume 323, p. 533–536.

Salton, G. M., Wong, A. & Yang, C., 1975. A vector space model for automatic indexing. Communications of the ACM, 18(11), p. 613–620.

Sanders, A. C. et al., 2021. Unmasking the conversation on masks: Natural language processing for topical sentiment analysis of COVID-19 Twitter discourse. s.l., PubMed Central, pp. 555-564.

Sang, E. F. T. K. & Meulder, . F. D., 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. Morristown, NJ, USA, Association for Computational Linguistics (ACL), pp. 142-147.

Sanh, V., Debut, L., Chaumond, J. & Wolf, T., 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv.

Sia, S., Dalmia, A. & Mielke, S. J., 2020. Tired of Topic Models? Clusters of Pretrained Word Embeddings Make for Fast and Good Topics too!. Online, Association for Computational Linguistics, p. 1728–1736.

Singh, H., 2021. How to select Best Split in Decision trees using Gini Impurity. [Online] Available at: https://www.analyticsvidhya.com/blog/2021/03/how-to-select-best-split-in-decision-trees-gini-impurity/

Terragni, S. et al., 2021. OCTIS: Comparing and Optimizing Topic models is Simple!. Online, s.n., p. 263–270.

The Matplotlib development team, n.d. Matplotlib: Visualization with Python. [Online] Available at: https://matplotlib.org/

Tian, Y. & Zhang, Y., 2022. A comprehensive survey on regularization strategies in machine learning. Information Fusion, Volume 80, pp. 146-166.

Trieschnigg, D., Kraaij, W. & Jong, F. d., 2007. The influence of basic tokenization on biomedical document retrieval. Online, Presented at the Reliability Engineering & System Safety - RELIAB ENG SYST SAFETY, pp. 803-804.

Vaswani, A. et al., 2017. Attention Is All You Need. Long Beach, CA, USA, s.n.

Waskom, M., n.d. seaborn: statistical data visualization. [Online], Available at: https://seaborn.pydata.org/

Wiederhold, B. K., 2020. Social Media Use During Social Distancing. Cyberpsychology, Behavior, and Social Networking, 23(5), pp. 275-276.

Wikipedia, n.d. Word2Vec. [Online], Available at: https://en.wikipedia.org/wiki/Word2vec

Wu, Y. et al., 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv preprint.

Xue, J. et al., 2020. Public discourse and sentiment during the COVID 19 pandemic: Using Latent Dirichlet Allocation for topic modeling on Twitter. PLoS ONE 15(9): e0239441.

Appendix

The Git Repository link provided by the University of Birmingham:

https://git-teaching.cs.bham.ac.uk/mod-msc-proj-2021/axp545

Code execution

In the git repository (link provided above), there are four folders and ten files. This project was done on Jupyter Notebook. There are three Jupyter files named ScrapingData.ipynb, SentimentclassificationOfUserReviews.ipynb, and TopicWiseTemporalSentimentAnalysis.ipynb. Also, two CSV files named t2_data_train.tsv and t2_data_test.tsv contain the dataset of user reviews of contact tracing apps from 46 countries. The folder named 'trainedmodel' will save and store the trained sentiment classifier model and is necessary. The folder named '.ipynb_checkpoints' can be deleted as it saves the checkpoint of the Jupyter notebook when it is manually saved. The folder named 'results' is the output directory where the prediction and checkpoints of the sentiment classifier model are stored in binary format. The TensorBoard uses it to visualise various parameters related to the model, such as loss per epoch, model graph etc. But this folder is not directly related to the project and is considered metadata. The folder named 'logs' store the summary files whenever the TensorFlow training model is run. Logs folders should be kept, but the summary files are not used in this project. Covid19_GooglePlayData.csv, covid19_googleplaydata_with_topics1.csv and covid19_googleplaydata_with_topics2.csv are the output files that we will get after we run the Jupyter Files.

The steps to run the code are as follows:

- First, run the ScrapingData.ipynb. This file will scrape the NHS COVID-19 app user reviews on the google play store and save it in a CSV file named Covid19_GooglePlayData.csv. Libraries needed to run this code are google_play_scraper, Pandas and NumPy.
- Then run the SentimentclassificationOfUserReviews.ipynb file.This file is being used to train and test the sentiment classifier model. We need t2_data_train.tsv and t2_data_test.tsv files as these files contain the datasets on which the sentiment classifier model is trained and tested. The Trained model will be saved in the 'trainedmodel' folder. Libraries required to run this Jupyter file are transformers, TensorFlow, pandas, NumPy, matplotlib, and sklearn.
- Finally, we need to run the TopicWiseTemporalSentimentAnalysis.ipynb Jupyter file. We will also need Covid19_GooglePlayData.csv, which contains the dataset for which we perform Topic-wise Temporal Sentiment Analysis. The sentiment classifier model saved in the 'trainedmodel' will be imported into this Jupyter File. After running this code, two CSV files, covid19_googleplaydata_with_topics1.csv and covid19_googleplaydata_with_topics2.csv, will be created. covid19_googleplaydata_with_topics1.csv contained the user reviews and topics when the Topic Extraction model (BERTopic) was used with its default hyper-parameters. covid19_googleplaydata_with_topics2.csv includes the user reviews and the topics extracted after tuning the hyper-parameters of BERTopic. Libraries needed to run TopicWiseTemporalSentimentAnalysis.ipynb are regular expressions, pandas, NumPy, BERTopic, hdbscan, matplotlib, sklearn, transformers and TensorFlow.