Master of Technology

Pattern Recognition System Project

Covid-19 Vaccine Adverse Effect Analyser

Team members:

Antonia Devina	A0127686R
Benjamin Quek Xiang Yi	A0229973M
Chwa Choon Xiang	A0229962R
Gerard Ong Zi Quan	A0229967H

Table of Contents

List of Figures

List of Tables

1. EXECUTIVE SUMMARY

Covid-19 pandemic, caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), has emerged as one of the deadliest pandemics in history, with over 5 million deaths confirmed from December 2019 to November 2021. In order to reduce the strain on healthcare systems, medical organizations around the world have turned to vaccination as the solution to build immunity against the highly infectious virus and to reduce the severity of the symptoms experienced by patients. However, due to the accelerated rate of vaccine development and administration, there have been concerns raised on the adverse effects or reactogenicity of these vaccines. Patients who experience post-vaccination symptoms may report to the Vaccine Adverse Event Reporting System (VAERS). Medical staff would be required to further process these reports such as labelling the medical symptoms from unstructured text and providing follow up decision for the patients, which has led to significant workload given the high rates of vaccination. To address the issues related to vaccine adverse events, our team has developed an intelligent system to provide predictive analytics and automation of decision-making processes & recommendations.

The system platform is designed to provide interactive predictions and recommendations to patients, while medical staff are given administrative permission to tune predictive models to meet changing business needs. It is built using Django-reactJS framework, with incorporation of various machine learning modules in the back-end processing. Datasets from VAERS database and Covid-19 Open Research Database (CORD-19) were pre-processed to extract relevant features for subsequent training of machine learning modules. These modules include multi-label classification of unstructured text to medical symptoms using Support Vector Machine (SVM), Deep Transfer Learning (DTL) and word vectorization techniques. Using the classified symptoms and additional patient information from the VAERS database, prediction of hospitalization outcome was performed and compared using ensemble methods (XGBoost, LightGBM, RandomForest) and deep feed-forward neural networks, with optimization of hyperparameters and network architecture. To improve the model performance, hybrid models using different decision rulesets were also explored and the evaluation scores of the models are reported. In addition, the classified symptoms were also used for literature mining of relevant CORD-19 medical articles, through an indexing and retrieval system, based on semantic & statistical search, question & answer model and sentiment analysis. The recommended medical articles are provided to the user to address vaccination concerns using scientific evidence, along with the decision for further hospitalization.

Our Covid-19 vaccine adverse event analyser system provides a full stack solution for predictive analytics using data-driven machine learning approach. Implementation of our project was demonstrated in a case study provided, highlighting the user experience from both the patient perspective and administrative functions of medical staff. Finally, the challenges and limitations of the project were discussed and future improvements are proposed.

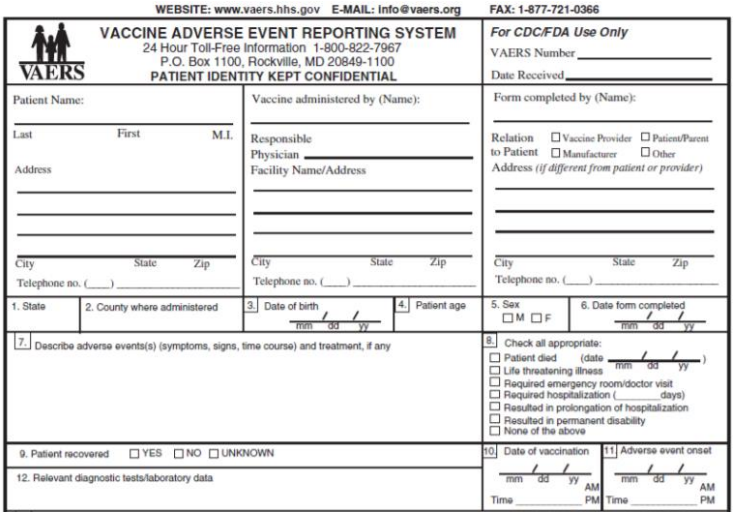
2. PROBLEM STATEMENT & OBJECTIVES

2.1. Covid-19 Vaccine: Background, Issues & Machine Learning

The Covid-19 pandemic, caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), has resulted 252 million reported cases and 5.09 million deaths globally, from December 2019 to November 2021 [1]. The virus is known to be transmitted by droplets in the air and common symptoms include dry cough, fever and loss of taste and smell. Emergency medical attention is often required when these symptoms evolve to shortness of breath, state of confusion and persistent pain in the chest [2].

Due to the highly infectious nature of the virus, the pandemic has put a strain on healthcare systems and various health organizations around the world have turned to vaccine development as the solution to manage the fatality rate. Several types of vaccines have been developed such as mRNA vaccines (Pfizer-BioNTech, Moderna), adenovector vaccine (Johnson & Johnson) and inactivated virus technology (Sinovac). While the Covid-19 vaccines have been reported to show high efficacy rates of up to 95% and reduction of severe symptoms and fatality rates, there has also been several issues regarding the use and adoption of vaccines to manage the pandemic. Adverse events have been reported against Covid-19 vaccines: pain at injection site, fever, allergic reactions, dizziness, chest pain, most of which experienced within 1-2 days of dosing and a median duration of 1 day [3]. However, there has been more scrutiny towards severe adverse effects such rare blood clotting disorder thrombosis, inflammatory heart conditions myocarditis and pericarditis, leading to further hospitalization and reported cases of death [4, 5, 6, 7].The risk of severe consequences arising from vaccine dosage could also be amplified by age factor or pre-existing conditions [8, 9]. Although medical studies and clinical trials have not concluded the presence of correlation between severe adverse events and vaccine dosage, these risks have generated public hesitancy towards vaccination and has impacted government policies on economic recovery and travel restrictions [10, 11].

Furthermore, medical organizations also face constraints to treat vaccine adverse effects on top of the overstrained resources for Covid-19 treatments. In the Vaccine Adverse Events Reporting System (VAERS), users fill up the reported symptoms using unstructured text, which will be further converted to MedDRA (Medical Dictionary for Regulatory Activities) terminologies by medical professionals [12]. Using supplementary information such as medical profiling and pre-existing conditions, the risk of severe adverse effects leading to hospitalization admission will be ascertained as well, on the context of available hospital facilities and treatment staff. As vaccination rates increase, these activities put additional workload on the medical staff to deal with the high rate of adverse event reporting.



WEBSITE: www.vaers.hhs.gov E-MAIL: info@vaers.org FAX: 1-877-721-0366

VACCINE ADVERSE EVENT REPORTING SYSTEM
24 Hour Toll-Free Information 1-800-822-7967
P.O. Box 1100, Rockville, MD 20849-1100
PATIENT IDENTITY KEPT CONFIDENTIAL

VAERS

For CDC/FDA Use Only
VAERS Number
Date Received

Patient Name: Last First M.I.
Address
City State Zip
Telephone no. (___)

Vaccine administered by (Name):
Responsible Physician
Facility Name/Address
City State Zip
Telephone no. (___)

Form completed by (Name):
Relation to Patient ☐ Vaccine Provider ☐ Patient/Parent ☐ Manufacturer ☐ Other
Address (if different from patient or provider)
City State Zip
Telephone no. (___)

1. State | 2. County where administered | 3. Date of birth mm dd yy | 4. Patient age | 5. Sex ☐ M ☐ F | 6. Date form completed mm dd yy

7. Describe adverse events(s) (symptoms, signs, time course) and treatment, if any

8. Check all appropriate:
☐ Patient died (date mm dd yy)
☐ Life threatening illness
☐ Required emergency room/doctor visit
☐ Required hospitalization (_____ days)
☐ Resulted in prolongation of hospitalization
☐ Resulted in permanent disability
☐ None of the above

9. Patient recovered ☐ YES ☐ NO ☐ UNKNOWN

10. Date of vaccination mm dd yy Time _____ AM PM

11. Adverse event onset mm dd yy Time _____ AM PM

12. Relevant diagnostic tests/laboratory data

Figure 1: Example for vaccine adverse effect reporting form.

In the field of vaccine study and development, machine learning (ML) has been used as a tool for designing of novel molecules and predictive responses to the coronavirus [13, 14]. For example, research has been conducted using feed-forward neural networks to analyse viral peptide present on molecules and its effect on natural immunity [15]. Ong et al. utilized XGBoost-based ML and reverse vaccinology to derive proteins which are promising vaccine candidates to SARS-CoV-2 [16]. For prediction of vaccine immunogenicity and reactogenicity, machine learning methods can be designed based on variety of input information to the desired outcome, as illustrated in Figure 2 [17, 18]. Machine learning outcomes could also have applications in addressing vaccine hesitancy through the use of data-driven analytics [19].

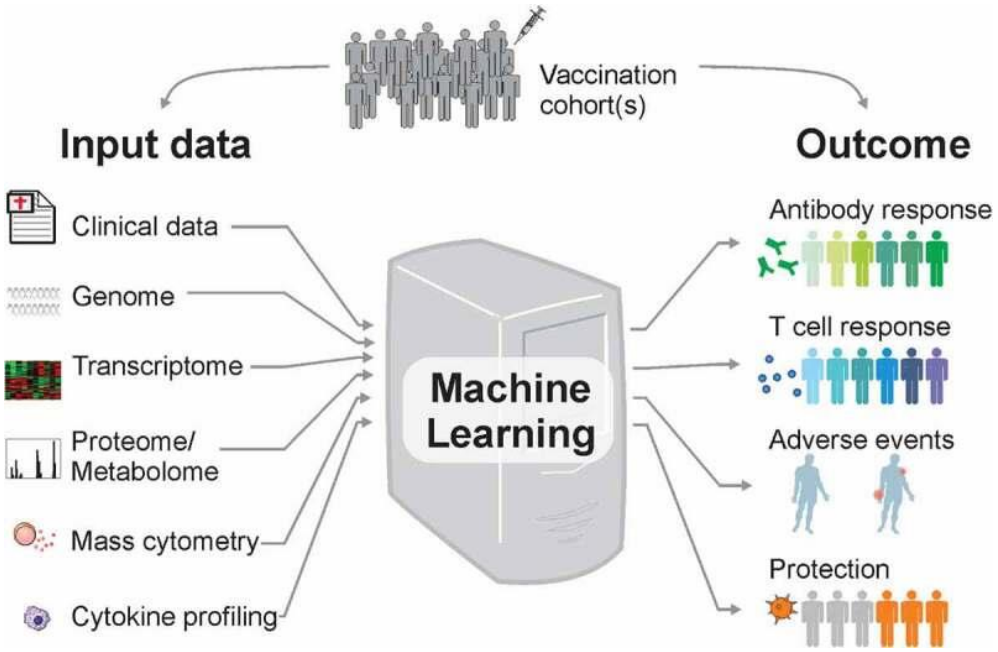


Figure 2: Machine learning applications for vaccination immunogenicity and reactogenicity [17].

In this project, machine learning will be utilised to analyse the above-mentioned issues regarding Covid-19 vaccine adverse effect and to develop a prototype system. The goal of the system is to reduce workload of medical staff through smart predictive analytics and to provide evidence based clinical studies to address vaccination concerns for the public users. Various AI techniques will be explored to tackle problems of unstructured text classification for medical symptoms, prediction of hospitalisation from adverse effects, and recommendation of clinical results from literature mining. Comparison of different machine learning models will be made to select a suitable model for accurate prediction of vaccine adverse effect outcomes, and ensemble/hybrid methods will also be evaluated to improve the accuracy of predictions. The objective of the project is to develop data-driven predictions to manage the use and adoption of Covid-19 vaccines more effectively, to combat the pandemic.

2.2. Project Scope and Objectives

This project aims to develop a smart predictive application with key features to address the issues related to Covid-19 vaccine adverse effects as outlined in the above section:

- Unstructured text classification for medical symptoms, to reduce workload in manual conversion by medical staff.
- Prediction of hospitalisation from adverse effects, to streamline the admission of patients requiring medical treatment.
- Recommendation of clinical outcomes from literature mining, to provide scientific evidence to address public concerns towards vaccination.

In order to develop a system for Covid-19 vaccine adverse effect analysis, the following components are to be included:

- Supervised learning of classification models using machine learning techniques
- Deep transfer learning of pretrained models with fine-tuning for specific tasks
- Design of multi-model decision making and workflow as hybrid/ensemble system
- Development of interactive frontend UI

The system developed by our group will be used as a platform to interact with users to obtain reporting variables for vaccine adverse events. Various machine learning modules are incorporated into the system architecture. The unstructured text collected will be processed into medical symptoms through an automated multi-label classification model. Along with the other structured profiling information, the aggregated data will be channelled to a predictive model for hospital admission decision. Users will also be recommended relevant medical articles pertaining to their adverse conditions, to inform on scientific outcomes of vaccination. From the administrative view, medical organisations will be able to view patient profile and customise prediction parameters to meet business needs such as availability of medical facilities. This system serves as an automated decision-making interface between users and medical organizations.

3. SOLUTION

3.1. System overview

Our team has developed a solution for analysis of Covid-19 vaccine adverse effects using machine learning. In essence, it is a full stack application system based on Django python web framework, comprising of several machine learning modules in the back-end processing. Users will interact with our UI by providing vaccine adverse event information, and they will receive automated results of hospitalisation prediction and recommended articles to their symptoms. Figure 3 below illustrates the overview of our system architecture and the integration of various machine learning modules for different tasks. In depth details of the specific components (ML modules & system design) are described in the following sections below.

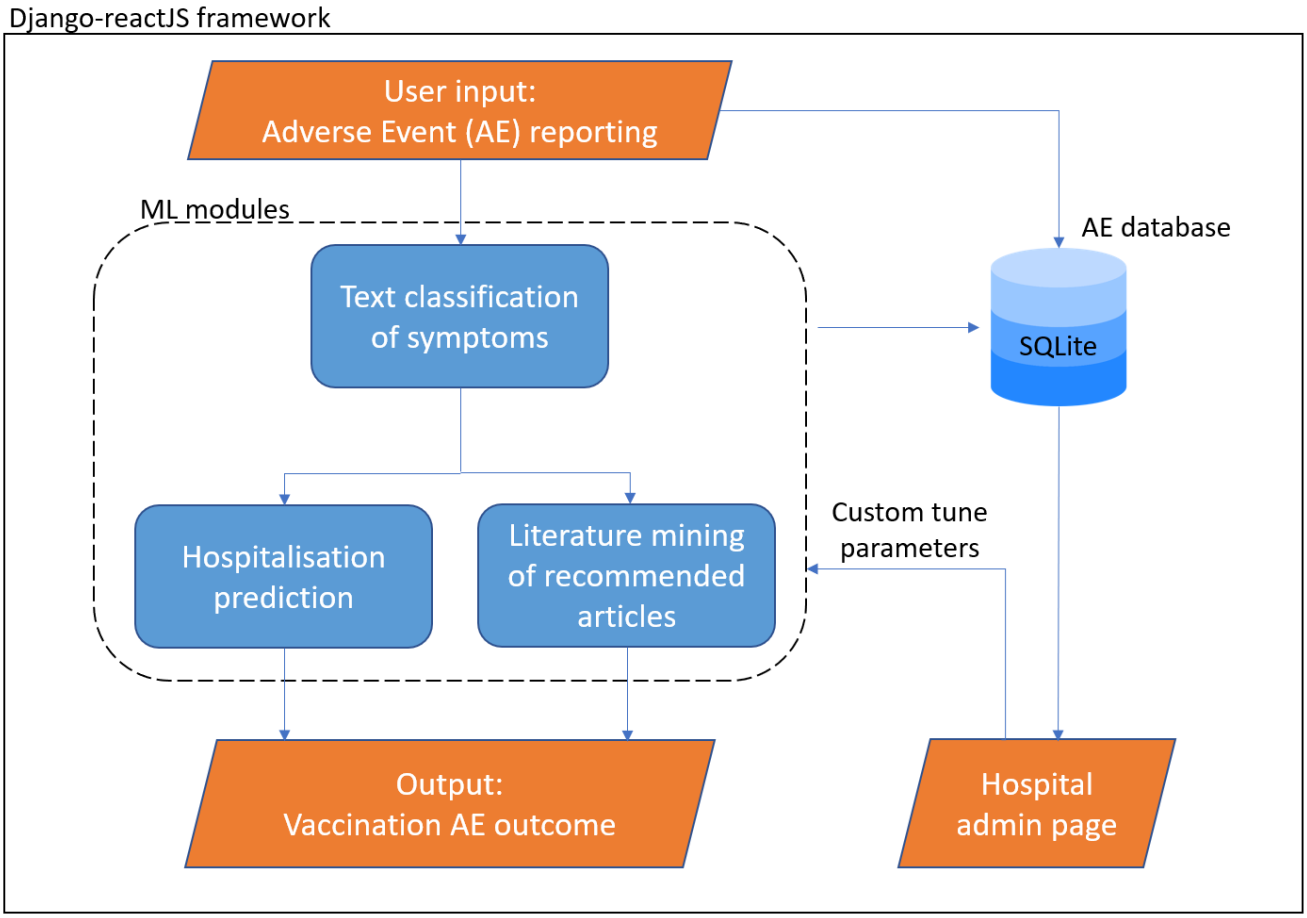


Figure 3: Overview of system application for Covid-19 vaccine adverse event analyser.

The detailed system architecture and UI design will be described in session 3.3 System design.

3.2. Machine Learning Modules: Models & Performances

3.2.1. *Datasets*

In total, there are 2 datasets used for this project. For adverse effect prediction, the models are build based on widely available vaccination data maintained by the Vaccine Adverse Effect Reporting System (VAERS, http://vaers.hhs.gov) [20]. VAERS recorded every vaccination related side effect report in the US starting from 1990, the data are then categorized into 3 tables and updated at the end of the month. The range of data used in this project are from January 2020 – July 2021 data VAERS data [21].

The files downloaded are in the form of CSV categorized by year and the sub-table. Individual files are saved locally in directory encoded as **DIR**. Data from different year are merge by vertical concatenation yielding 3 main dataset to work with, the *vaers* [VAERS data], vax [VAERS vaccination] and symptoms [VAERS symptoms] dataset.

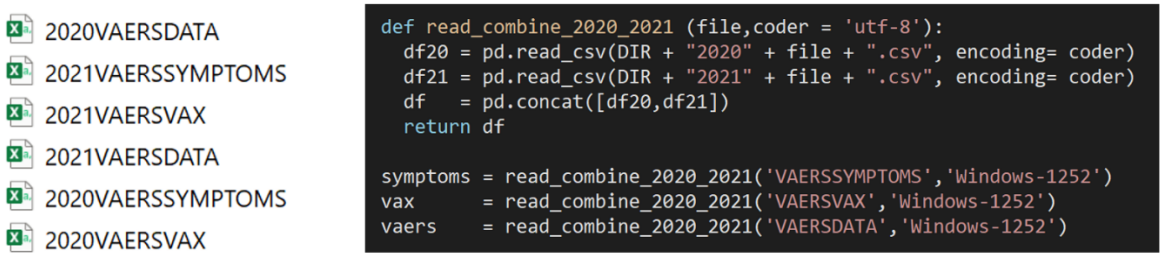


Figure 4: VAERS raw data and combination.

The 3 tables in the VAERS can also be represented in term of their features in the figure below. The VAERS data are mainly containing personal details, VAERS Vaccine dataset contains the vaccination related data and lastly the VAERS Symptoms containing the Symptoms based on MedDRA terms. The three datasets are easily merge through a common column of *VAERS_ID* or running number used to identify the individual. The details of each dataset are summarized in the figure below.

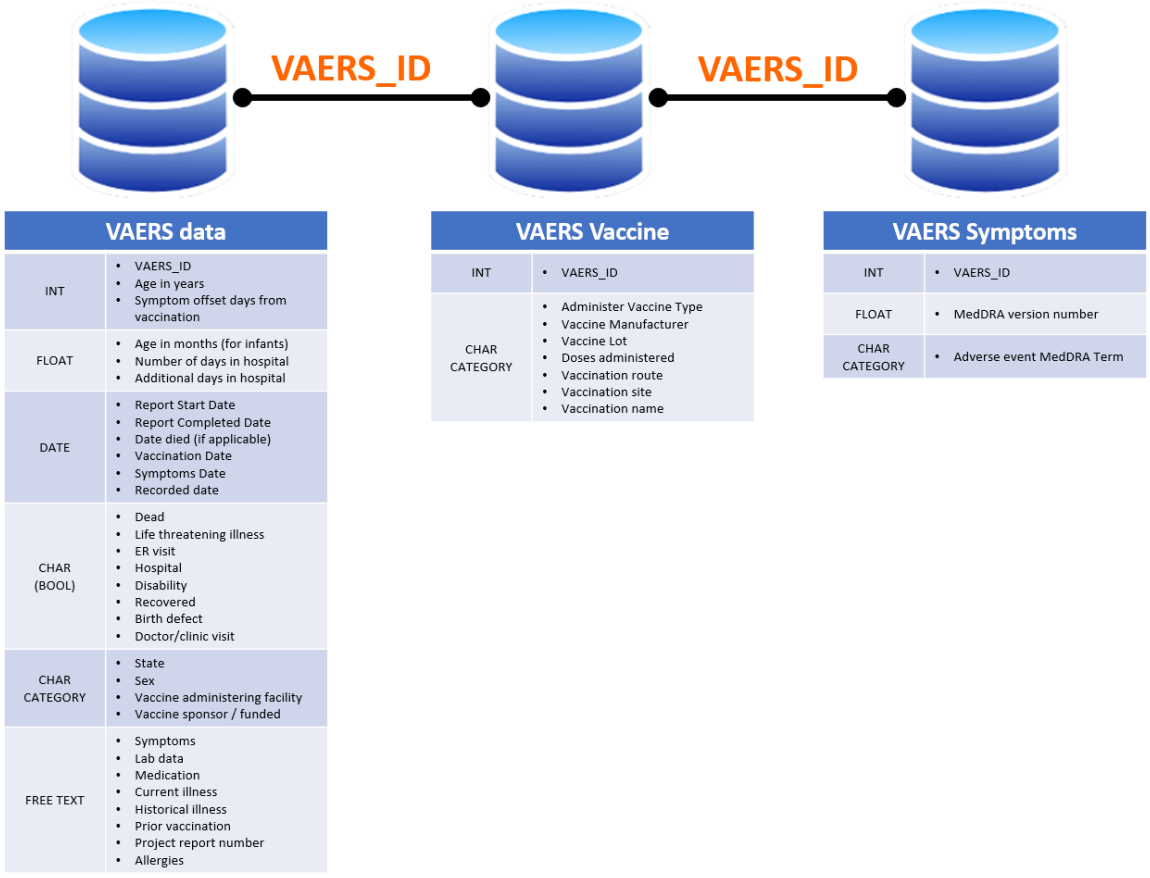


Figure 5: VAERS dataset and the information in each table.

The second dataset was used for literature mining of recommended articles. Covid-19 Open Research Dataset (CORD-19) was created by researchers at Allen Institute for AI (AI2), as a compilation of research progress made in the field of Covid-19 [22]. It contains a collection of publications retrieved from PubMed Central (PMC), bioRxiv, medRxiv and World Health Organization (WHO) database. Using the key words Covid-19 and Coronavirus research, there are over 280,000 related articles published on this topic. With the large volume of research made towards Covid-19, it is difficult for any individual or research group to digest all the critical knowledge that has been published, hence CORD-19 was designed as a database for literature mining of required information. CORD-19 is updated weekly to the present date at time of writing (Nov 2021), and is available at https://www.semanticscholar.org/cord19 [23].

The dataset primarily consists of a metacsv file and JSON files of the article PDF. The contents of the metacsv file include: Title, Author, Journal, Abstract, and path link to the JSON PDF. It is mainly used for quick filtering of desired articles, as the CSV file is small and does not contain the full text. For the JSON PDF, it contains additional information such as paragraph text, citation indexes and bibliography, as shown in Figure ***7***Figure ***6*** below.

	cord_uid	source_x	title	license	abstract	publish_time	authors	journal	pdf_json_files
0	ug7v899j	PMC	Clinical features of culture-proven Mycoplasma...	no-cc	OBJECTIVE: This retrospective chart review des...	2001-07-04	Madani, Tariq A; Al-Ghamdi, Aisha A	BMC Infect Dis	document_parses/pdf_json/d1aafb70c066a2068b027...
1	02tnwd4m	PMC	Nitric oxide: a pro-inflammatory mediator in l...	no-cc	Inflammatory diseases of the respiratory tract...	2000-08-15	Vliet, Albert van der; Eiserich, Jason P; Cros...	Respir Res	document_parses/pdf_json/6b0567729c2143a66d737...
2	ejv2xln0	PMC	Surfactant protein-D and pulmonary host defense	no-cc	Surfactant protein-D (SP-D) participates in th...	2000-08-25	Crouch, Erika C	Respir Res	document_parses/pdf_json/06ced00a5fc04215949aa...
3	2b73a28n	PMC	Role of endothelin-1 in lung disease	no-cc	Endothelin-1 (ET-1) is a 21 amino acid peptide...	2001-02-22	Fagan, Karen A; McMurtry, Ivan F; Rodman, David M	Respir Res	document_parses/pdf_json/348055649b6b8cf2b9a37...

Figure 6: Data from CORD-19 metacsv file.

```
"body_text": [

    {
        "text": <str>,
        "cite_spans": [],
        "ref_spans": [],
        "eq_spans": [],
        "section": "Introduction"
    },
    ...
    {
        ...,
        "section": "Conclusion"
    }
],
"bib_entries": {
    "BIBREF0": {
        "ref_id": <str>,
        "title": <str>,
        "authors": <list of dict>

        "year": <int>,
        "venue": <str>,
        "volume": <str>,
        "issn": <str>,
        "pages": <str>,
        "other_ids": {
            "DOI": [
                <str>
            ]
        }
    },
    "BIBREF1": {},
    ...
    "BIBREF25": {}
},
"ref_entries":
    "FIGREF0": {
        "text": <str>,
        "type": "figure"
    },
```

Figure 7: Section of JSON schema of full text PDF.

3.2.2. *Text classification for adverse effect symptoms*

3.2.2.1. Data Pre-processing

In the VAERS symptom dataset, each unique VAERS_ID is recorded to have up to 5 symptoms, each listed in a separate column (SYMPTOM1-SYMPTOM5). First, we extracted the symptoms for each unique VAERS_ID from the columns, and concatenated them into a single new column for easier processing.

VAERS_ID	SYMPTOM	SYMPTOM	SYMPTOM	SYMPTOM	SYMPTOM	SYMPTOM	SYMPTOM	SYMPTOM	SYMPTOM	SYMPTOMVERSION5
855017	Arthralgia	22.1	Chills	22.1	Injection s	22.1	Pyrexia	22.1		
855018	Chills	22.1	Fatigue	22.1	Hypertensi	22.1	Hypoaesth	22.1	Injected lir	22.1
855018	Muscular v	22.1	Pain in ext	22.1	Pyrexia	22.1	Tremor	22.1	Vertigo	22.1
855019	Pain	22.1	Pruritus	22.1	Rash	22.1				
855020	Chills	22.1	Influenza l	22.1	Myalgia	22.1	Pain in ext	22.1	Pyrexia	22.1
855021	Chills	22.1	Dizziness	22.1	Nausea	22.1	Palpitation	22.1		

VAERS_ID	SYMPTOMS
896636	arthralgia,confusional state,fatigue,feeling abnormal,head discomfort,memory impairment,pain in extremity,peripheral swelling,physiotherapy,pyrexia
902418	hypoaesthesia,injection site hypoaesthesia
902440	headache
902446	erythema,feeling hot,flushing
902464	dizziness,electrocardiogram normal,hyperhidrosis,laboratory test normal,presyncope
902465	dysgeusia,oral pruritus,paraesthesia,paraesthesia oral,parosmia,sensory disturbance,tremor
902468	chest discomfort,chills,defaecation urgency,diarrhoea,dizziness,dyspnoea,feeling abnormal,flushing,presyncope

Figure 8: Concatenated symptoms from VAERS symptom dataset.

Thereafter we performed our first down-sampling of the symptoms data to only include symptoms with count of more than 100. However, even after this first round of down-sampling we found that in total we still had over 14,000 unique symptoms and was impractical for our model to be able to predict whether a person has 5 out of this total number of symptoms. We decided we must perform further data down-sampling to make model training and inference more feasible in real-world situations.

As the final goal of the system is to predict the probability of someone who received a vaccine needing to be hospitalized, we decided that we should first split the dataset into VAERS_ID who were hospitalized and those who were not, then do filtering on each dataset to down-sample the number of symptoms for the text classification model to predict. For each of the hospitalized and non-hospitalized datasets, we performed a count on each of the symptoms and plotted the counts against the index representing the individual symptoms.

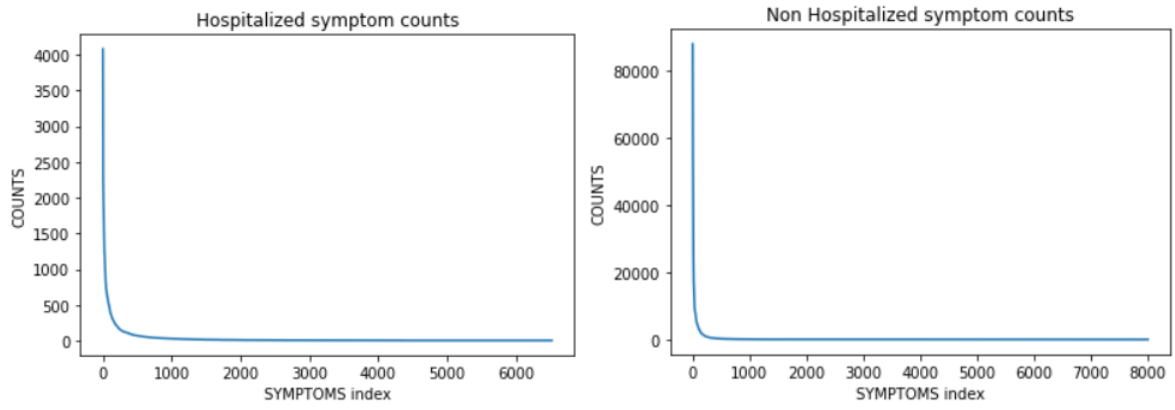


Figure 9 Plot of symptom counts against symptom index.

From here we applied the elbow method to find the optimal cut-off point where the number of counts for the symptoms start to plateau. The symptoms before the plateau are chosen for the model training and inference as they represent the most common symptoms found in the dataset for both hospitalized and non-hospitalized patients separately.

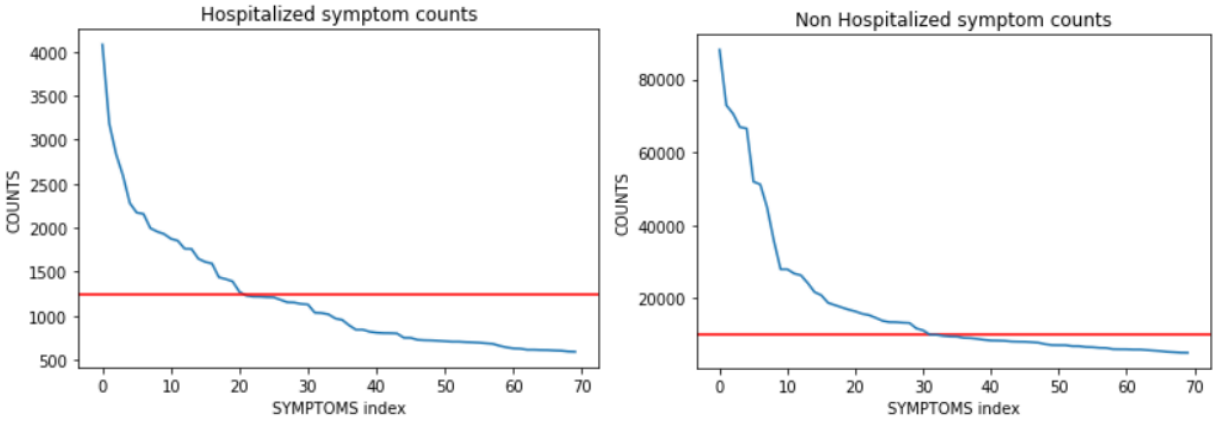


Figure 10: Plot of symptom counts against symptom index with cut-off line.

For the hospitalized subset, 21 symptoms were chosen and for the non-hospitalized subset, 30 symptoms were chosen. After combining the symptoms and removing duplicates, we have the final 41 symptoms for model training and inference.

```
Index(['SYMPTOMSdyspnoea', 'SYMPTOMSpyrexia', 'SYMPTOMSchest pain',
       'SYMPTOMSheadache', 'SYMPTOMSfatigue', 'SYMPTOMScovid-19',
       'SYMPTOMSblood test', 'SYMPTOMSnausea', 'SYMPTOMSasthenia',
       'SYMPTOMScomputerised tomogram', 'SYMPTOMSpain', 'SYMPTOMSdizziness',
       'SYMPTOMSsars-cov-2 test positive', 'SYMPTOMSvomiting',
       'SYMPTOMSchills', 'SYMPTOMScerebrovascular accident',
       'SYMPTOMSpulmonary embolism', 'SYMPTOMSelectrocardiogram',
       'SYMPTOMSechocardiogram', 'SYMPTOMSpain in extremity',
       'SYMPTOMStroponin increased', 'SYMPTOMSinjection site pain',
       'SYMPTOMSarthralgia', 'SYMPTOMSmyalgia',
       'SYMPTOMSinjection site erythema', 'SYMPTOMSrash', 'SYMPTOMSpruritus',
       'SYMPTOMSinjection site swelling', 'SYMPTOMSerythema',
       'SYMPTOMSinjection site pruritus', 'SYMPTOMShyperhidrosis',
       'SYMPTOMSparaesthesia', 'SYMPTOMSurticaria', 'SYMPTOMSdiarrhoea',
       'SYMPTOMShypoaesthesia', 'SYMPTOMSinjection site warmth',
       'SYMPTOMSmalaise', 'SYMPTOMSlymphadenopathy',
       'SYMPTOMSfeeling abnormal', 'SYMPTOMSperipheral swelling',
       'SYMPTOMSsyncope'],
      dtype='object')
```

Figure 11: List of 21 down sampled symptoms.

We then performed one-hot encoding to convert the symptoms for both the non-hospitalized and hospitalized subsets into a format that can be accepted by downstream models.

	VAERS_ID	SYMPTOMS	SYMPTOMSdyspnoea	SYMPTOMSpyrexia	SYMPTOMSchest pain	SYMPTOMSheadache	SYMPTOMSfatigue
2	855172	asthenia,back pain,chest x-ray,computerised to...	0	0	0	0	0
3	855178	abdominal pain,appendicectomy,henoch-schonlein...	0	0	0	0	0
9	855341	chest x-ray,computerised tomogram head,dysphag...	0	0	0	0	0
10	855349	guillain-barre syndrome,muscular weakness,pain	0	0	0	0	0
11	855373	blood test,electrocardiogram,malaise,syncope	0	0	0	0	0
...	...	...	...	...	...	...	...
28062	1514787	alcoholism,dysstasia,eating disorder,fall,fore...	0	0	0	0	0
28064	1514802	deep vein thrombosis,dyspnoea,fatigue,flank pa...	1	1	0	0	1
28065	1514807	bedridden,condition aggravated,gait disturbanc...	0	0	0	0	0
28066	1514835	blood culture,blood gases,blood lactic acid,co...	1	0	0	0	0
28067	1514845	asthenia,confusional state,diarrhoea,dizziness...	0	0	0	0	1

Figure 12: VAERS symptom one-hot vector.

Here we made sure that all 41 symptoms were represented in both the non-hospitalized and hospitalized subset one-hot vectors. After removing rows with all zeroes, we concatenated both vectors and ensured that there are no duplicated VAERS_ID. Here we have obtained our 41 symptom, one-hot encoded dataset containing 384,014 entries.

```
Data columns (total 43 columns):
 #   Column                              Non-Null Count   Dtype
---  ------                              --------------   -----
 0   VAERS_ID                            384014 non-null  int64
 1   SYMPTOMS                            384014 non-null  object
 2   SYMPTOMSdyspnoea                    384014 non-null  int64
 3   SYMPTOMSpyrexia                     384014 non-null  int64
 4   SYMPTOMSchest pain                  384014 non-null  int64
 5   SYMPTOMSheadache                    384014 non-null  int64
 6   SYMPTOMSfatigue                     384014 non-null  int64
 7   SYMPTOMScovid-19                    384014 non-null  int64
 8   SYMPTOMSblood test                  384014 non-null  int64
 9   SYMPTOMSnausea                      384014 non-null  int64
 10  SYMPTOMSasthenia                    384014 non-null  int64
 11  SYMPTOMScomputerised tomogram       384014 non-null  int64
 12  SYMPTOMSpain                        384014 non-null  int64
 13  SYMPTOMSdizziness                   384014 non-null  int64
 14  SYMPTOMSsars-cov-2 test positive    384014 non-null  int64
 15  SYMPTOMSvomiting                    384014 non-null  int64
 16  SYMPTOMSchills                      384014 non-null  int64
 17  SYMPTOMScerebrovascular accident    384014 non-null  int64
 18  SYMPTOMSpulmonary embolism          384014 non-null  int64
 19  SYMPTOMSelectrocardiogram           384014 non-null  int64
 20  SYMPTOMSechocardiogram              384014 non-null  int64
 21  SYMPTOMSpain in extremity           384014 non-null  int64
 22  SYMPTOMStroponin increased          384014 non-null  int64
 23  SYMPTOMSinjection site pain         384014 non-null  int64
 24  SYMPTOMSarthralgia                  384014 non-null  int64
 25  SYMPTOMSmyalgia                     384014 non-null  int64
 26  SYMPTOMSinjection site erythema     384014 non-null  int64
 27  SYMPTOMSrash                        384014 non-null  int64
 28  SYMPTOMSpruritus                    384014 non-null  int64
 29  SYMPTOMSinjection site swelling     384014 non-null  int64
 30  SYMPTOMSerythema                    384014 non-null  int64
 31  SYMPTOMSinjection site pruritus     384014 non-null  int64
 32  SYMPTOMShyperhidrosis               384014 non-null  int64
 33  SYMPTOMSparaesthesia                384014 non-null  int64
 34  SYMPTOMSurticaria                   384014 non-null  int64
 35  SYMPTOMSdiarrhoea                   384014 non-null  int64
 36  SYMPTOMShypoaesthesia               384014 non-null  int64
 37  SYMPTOMSinjection site warmth       384014 non-null  int64
 38  SYMPTOMSmalaise                     384014 non-null  int64
 39  SYMPTOMSlymphadenopathy             384014 non-null  int64
 40  SYMPTOMSfeeling abnormal            384014 non-null  int64
 41  SYMPTOMSperipheral swelling         384014 non-null  int64
 42  SYMPTOMSsyncope                     384014 non-null  int64
dtypes: int64(42), object(1)
memory usage: 128.9+ MB
```

Figure 13: VAERS symptom one-hot vector info.

3.2.2.2. Term frequency–inverse document frequency and Support Vector Machine Text Classification

The term frequency- inverse document frequency is broadly applied in real world to extract the feature importance of a word appeared in the document. It is mainly calculated from the two different metrics such as term frequency and inverse document frequency. The term frequency counts the appearance of the words in a document while the inverse document frequency is calculated from the division between the total number of documents by the number of documents that contain the word.

The multiplication between term frequency and inverse document frequency will form a score for the vocabulary as shown as below:

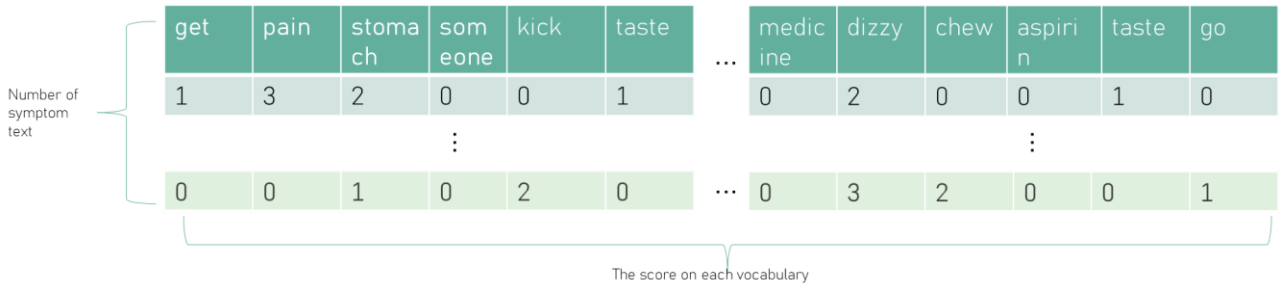


Figure 14: Visualisation on TF-IDF feature extraction on the symptom text.

The characteristic of the TF-IDF feature extraction on the keyword extraction will be beneficial to the machine learning model to know which vocabulary provided a significant meaning in a text document. A vocabulary with the high score should have the most relevant to that document and could be considered as the important feature for that document. For example, a text to describe the patient has been suffering from “headache” and the keyword “headache” should be recognised by the machine learning model for extracting the symptoms from text classification.

Even though the TF-IDF feature extraction method can provide the keywords appeared in the document, the processed data will be highly sparse due to large amount of uncommon vocabulary detected among all the text extracted from the patients. A suitable machine learning model which can deal with high sparse data for the text classification is critical. Support Vector machines (SVM) is the most powerful model to perform more accurate text classification.

Support vector machines (SVM) were the state-of-the-art ML model before the advent of Deep Learning due to its high accuracy. SVM is a supervised algorithm that is commonly used in classification and, at its most basic level, is based on the idea of searching for the optimal hyperplane that best separates a dataset into two classes for a binary classification task [24].

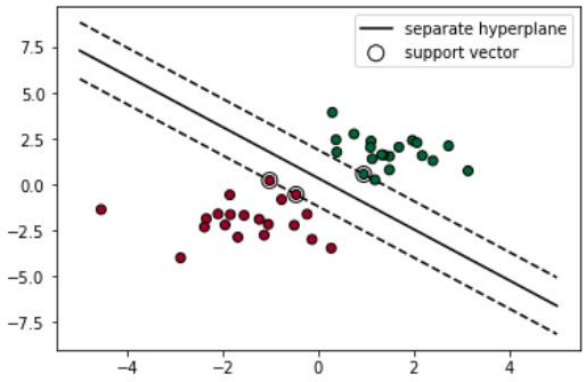


Figure 15: Example of Linearly separable SVM [24].

As seen in the image above, the hyperplane separates data from the 2 classes and the data points nearest to the hyperplane are labelled as support vectors. Support vectors are by definition the points closest to the optimal hyperplanes, but first how is the optimal hyperplane obtained?

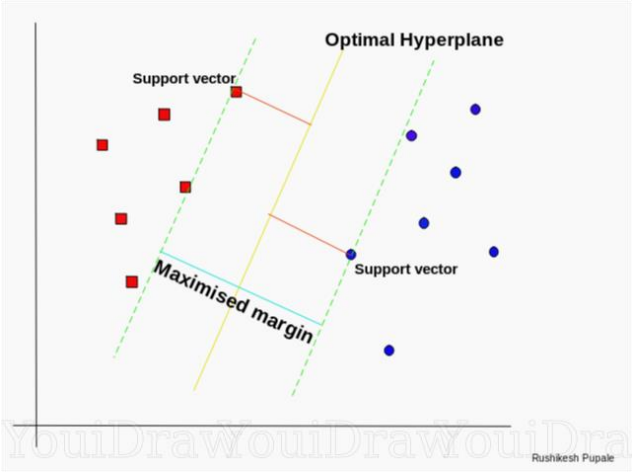


Figure 16: Illustration of Support vectors and margins [25].

As seen in Figure 16, the concept of margins is introduced. The goal of the SVM algorithm is to maximize the margin and this is done by calculating the distance between the candidate hyperplanes and the support vectors.

The above has described how SVM handles linearly separable data, for non-linearly separable data there are 2 methods by which SVM can handle it: Use of soft margins (soft SVM) or using Kernel Trick.

Using soft margins extend the SVM within the linear framework to classify non-linearly separable datasets. Applying soft margin allow the linear SVM to tolerate some datapoints getting misclassified and balances finding an optimal hyperplane that maximizes the margin and minimizes error [25].

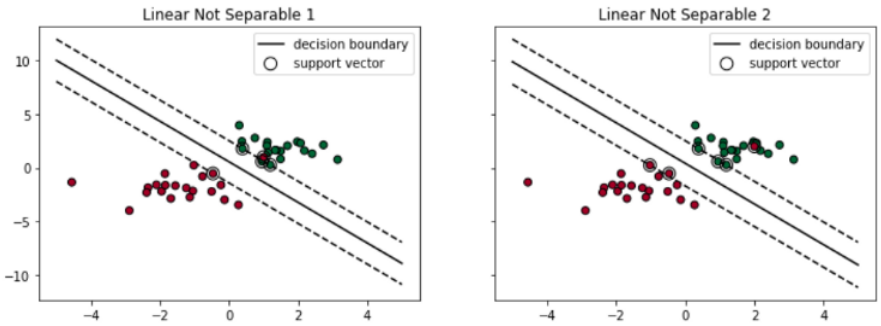


Figure 17: Example of Soft margin SVM [24].

For data that is highly non-linear, using the Kernel Trick might be more suitable for SVM. The Kernel Trick is basically applying a mapping of the current space of data to a higher dimension where a linear hyperplane to separate the data can be found.

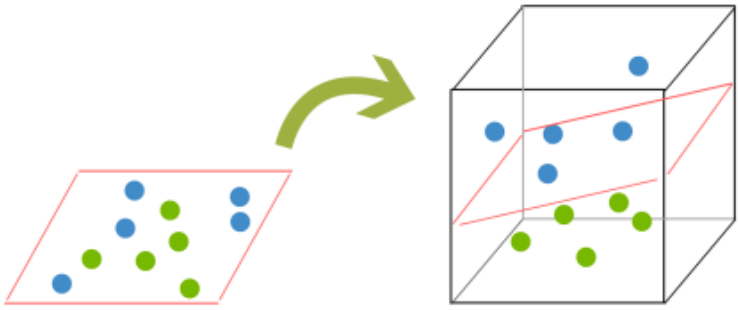


Figure 18: Kernel trick transformation to higher dimension [26].

As seen in Figure 18, data which is not linearly separable in 2 dimensions can be linearly separable in 3 dimensions through mapping the data to a higher dimension.

As there are 41 target classes for the text classification to be learned in the model, the SVM model is trained under one vs all classifiers. Each class will be trained with one binary classifier independently. The knowledge will the end be extracted and learnt by the SVM model to perform text classification. A radial basis function for kernel type and regularization parameter C which is set to 1 are the parameters to be used in the SVM Model [26].

The flowchart of the feature extraction and model definition are shown as below:

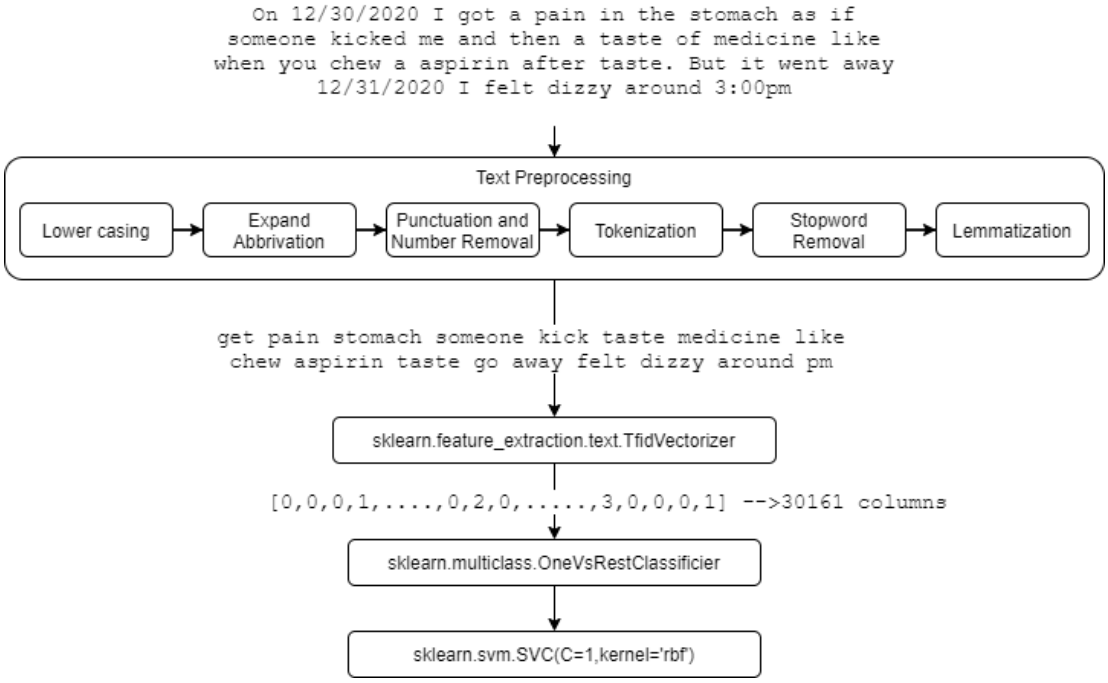


Figure 19: flowchart for the TF-IDF SVM Model.

Knowing that there are 384,014 entities in the dataset provided in 3.2.2.1 and the dataset is highly imbalance, the training of the data with 300k entities will be time consuming and will cause overfitting with the most frequent class. The AUC score of the least frequent class will be relatively low as comparing with other class and

affect the performance of the model. Thus, a down sampling method as shown in Figure ***20*** is applied to extract all the less frequent class.

```
sorted_columns = list(data_complete.iloc[:,2:].sum().sort_values().index)
dataframe_full = data_complete.copy()
dataframe_downsampling = pd.DataFrame(columns = data_complete.columns)
for column in sorted_columns[:4]:
  dataframe_concated = dataframe_full.loc[dataframe_full[column]==1].copy()
  dataframe_downsampling = pd.concat([dataframe_downsampling,dataframe_concated]).copy()
  dataframe_full = dataframe_full.loc[dataframe_full[column]==0].copy()
```

Figure 20: Formula to perform down sampling on the dataset.

The algorithm of the formula is to extract all the rows of the first four less frequent class and form another dataset called dataframe_downsampling. After the extraction, the leftover dataset will be performed down-sampling again and concatenate back to the dataset dataframe_downsampling. The final dataset after a few iterations of down sampling will be shown as below:

Figure 21: Frequency of the appearance of every target before and after down sampling.

SVM will then be fit with the down sampled dataset and the performance will be observed in 3.2.2.6.

3.2.2.3. Word2Vec feature extraction with Deep learning text classification

The TF-IDF feature extraction method only emphasize on the frequency of the words appeared in the document. Due to the characteristics, the meaning of a word based on the surrounding context words may not be taken care by the previous model. We will introduce the Word2Vec feature extraction with the Tensorflow Convolutional Neural Network for the text classification.

Word2Vec feature extraction is well-known to be used to learn word embedding from the large dataset which is successfully applied on a variety of natural language processing tasks.

The Word2Vec algorithms include continuous skip-gram model which predict words within a certain range before and after the current word in the same sentences and continuous bag-of-words model which predict the middle word based on the surrounding context words but the position of the bags of words are not considered [27].

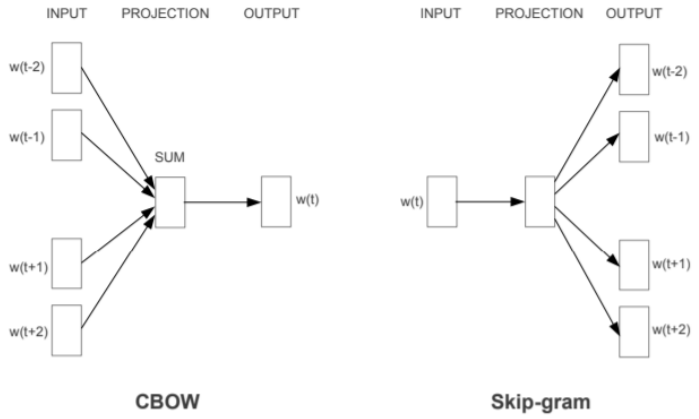


Figure 22: CBOW model which predicts the current word based on the context and the Skip-gram predicts surrounding words given the current word [27].

With the help of the algorithms, we can extract the word vector of all the vocabulary appeared in the dataset. All the vocabulary will form a vector matrix which is recognised as the weight matrix to be trained in the word embedding layer of the Deep Learning model.

Before we define the Word2Vec model to fit with raw text, there are various parameters to be taken note as below:

- size: Dimensionality of the word vectors which will define the number of outputs of the Embedding layer of Deep Learning Model
- window: Maximum distance between the current and predicted word within a sentence
- workers: Use these many worker threads to train the model. The parameter will decide how long the Word2Vec Model to learn all the text.
- min_count: Ignores all words with total frequency lower than this.

After the training of all the text data with the Word2Vec Model with parameters size = 200, window = 5, workers = 8 and min_count = 1, the level of semantic similarity can be extracted as illustrated below:

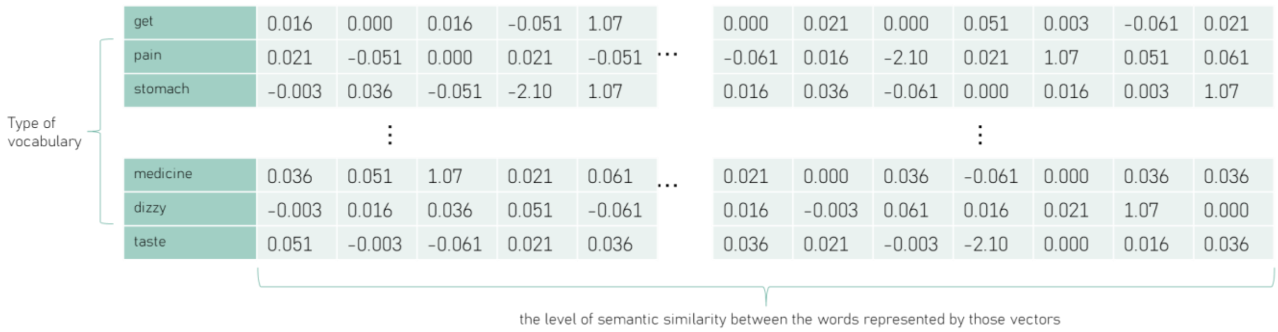


Figure 23: Vector matrix to represent the semantic similarity between the words.

All the detected vocabulary will form a matrix which consist of 200 unit of vector value and apply as a pre-trained word embedding layer prepared on a large dataset. As described in the flow chart shown in Figure ***24***, an input layer with 2845 output dimension is defined to connect with an embedding layer with a vocabulary size of 72639 and 2845 as the input dimension and length and 200 as the output dimension. The weight matrix has 14,527,800 parameters which has been trained in Word2Vec Model to show that the Word2Vec method has massively decreased the time consumed in training the parameters of an embedding layers.

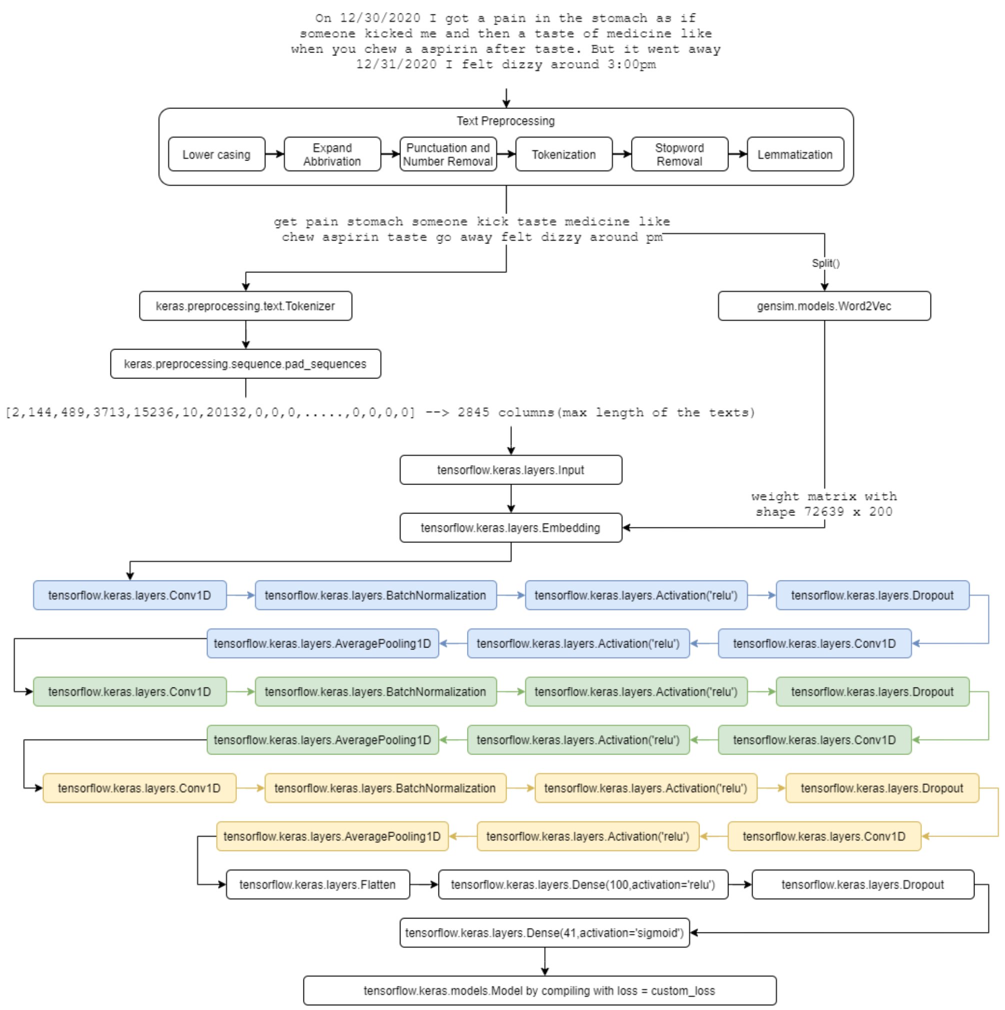


Figure 24: Flowchart of the Word2Vec Deep Learning model.

The embedding layers will then be connected sequentially with three set of the Convolutional-Batch Normalization Layer and Convolutional-Average Pooling Layer before flattening and condensing into 41 classes. Each set of the layers is set to be in the same channel and perform feature extraction by using convolutional

and average pooling method to decrease the dimension of the input. While there is a cross between each set of the layers, the channel size is set to be increased by 100. The method for feature extraction is observed to be helpful in the text classification to prevent overfitting, meanwhile, it also helps to decrease the number of the parameters to be trained. After three set of the layer combination, the parameters are observed to be reduced massively as the high reduction of the dimension of the last layer before flattening. Refer to the following information as stated:

Model: "model"

input_1 (InputLayer)	[(None,2845)]	0
embedding (Embedding)	(None,2845,200)	14527800
conv1d (Conv1D)	(None,1423,200)	320200
batch_normalization (BatchNormalization)	(None,1423,200)	800
activation (Activation)	(None,1423,200)	0
dropout (Dropout)	(None,1423,200)	0
conv1d_1 (Conv1D)	(None,712,200)	320200
activation_1 (Activation)	(None,712,200)	0
average_pooling1d (AveragePo	(None,356,200)	0
conv1d_2 (Conv1D)	(None,178,300)	480300
batch_normalization_1 (BatchNormalization)	(None,178,300)	1200
activation_2 (Activation)	(None,178,300)	0
dropout_1 (Dropout)	(None,178,300)	0
conv1d_3 (Conv1D)	(None,89,300)	720300
activation_3 (Activation)	(None,89,300)	0
average_pooling1d_1 (Average	(None,45,300)	0
conv1d_4 (Conv1D)	(None,45,400)	480400
batch_normalization_2 ((BatchNormalization)	(None,45,400)	1600
activation_4 (Activation)	(None,45,400)	0
dropout_2 (Dropout)	(None,45,400)	0
conv1d_5 (Conv1D)	(None,45,400)	640400
activation_5 (Activation)	(None,45,400)	0
average_pooling1d_2 (Average	(None,23,400)	0
flatten (Flatten)	(None,9200)	0
dense (Dense)	(None,100)	920100
dropout_3 (Dropout)	(None,100)	0
dense_1 (Dense)	(None,41)	4141

Total params: 18,417,441

Trainable params: 3,887,841

Non-trainable params: 14,529,600

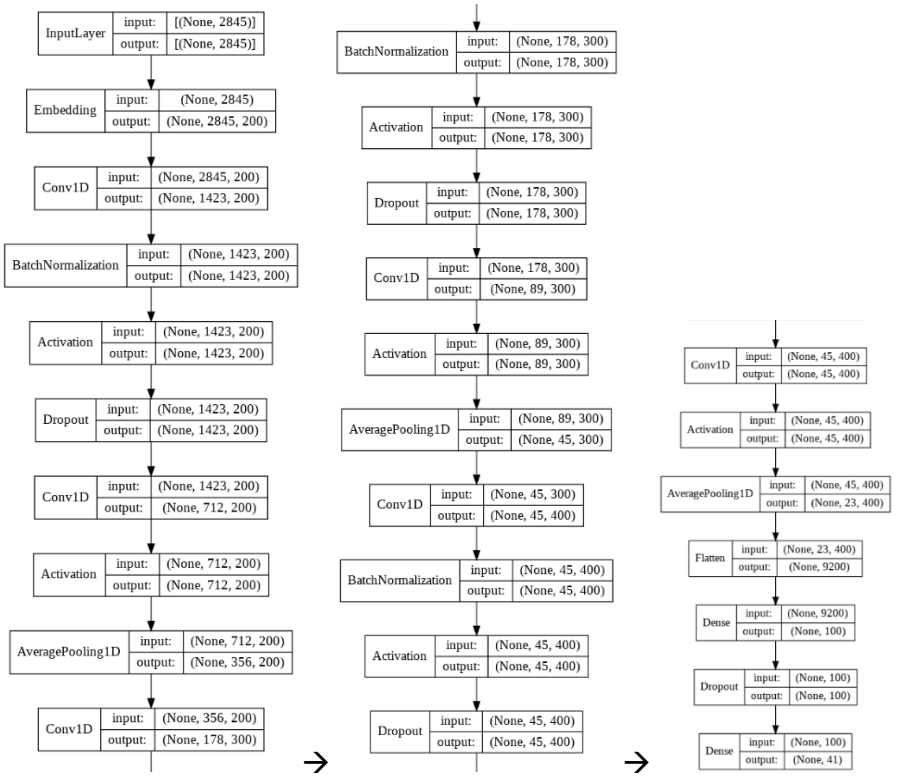


Figure 25: Optimisation of best Neural Network Architecture.

As mentioned in TF-IDF Model, the training data is highly imbalance, and it may affect the performance of the model. Unlikely the normal TF-IDF Model, down-sampling method will highly impact the performance of the Deep Learning model due to insufficient data and possibility to discard important dataset. Therefore, a custom loss function to include the class weight of the dataset is applied during compilation of the model.

```
[ ] from tensorflow.keras import backend as K
    from tensorflow.keras.losses import Loss

    def calculating_class_weights(y_true):
        from sklearn.utils.class_weight import compute_class_weight
        number_dim = np.shape(y_true)[1]
        weights = np.empty([number_dim, 2])
        for i in range(number_dim):
            weights[i] = compute_class_weight('balanced', [0.,1.], y_true[:, i])
        return weights

    class custom_loss(Loss):
      weights = {}
      def __init__(self,weights):
        super().__init__()
        self.weights = weights

      def call(self, y_true, y_logit):
          '''
          Multi-label cross-entropy
          * Required "Wp", "Wn" as positive & negative class-weights
          y_true: true value
          y_logit: predicted value
          '''
          bce = tf.keras.losses.BinaryCrossentropy()
          loss = K.mean((self.weights[:,0]**(1-tf.cast(y_true, tf.float32)))*(self.weights[:,1]**(tf.cast(y_true, tf.float32)))*bce(y_true, y_logit), axis=-1)
          return loss

    class_weights_train = calculating_class_weights(y_train)
    class_weights_test = calculating_class_weights(y_test)
```

Each of the weight of the classes has been taken into consideration as the factor to control the overall loss function. Higher penalties are applied to tune the parameters while any true positive of the class is not correctly predicted by the model. The effect is observed to highly as all the classes have been taken into consideration without missing any important dataset.

3.2.2.4. BERT for symptom classification

Transfer learning in deep learning tasks, especially in computer vision has been repeatedly proven to be a valuable means to rapidly deploy a model for specialized tasks. It is a technique where a deep learning model trained on a large dataset is used to perform another task on a different dataset, and one of the recently popular models in the field of NLP transfer learning application is BERT. BERT applies the bidirectional training of Transformers to language modelling.

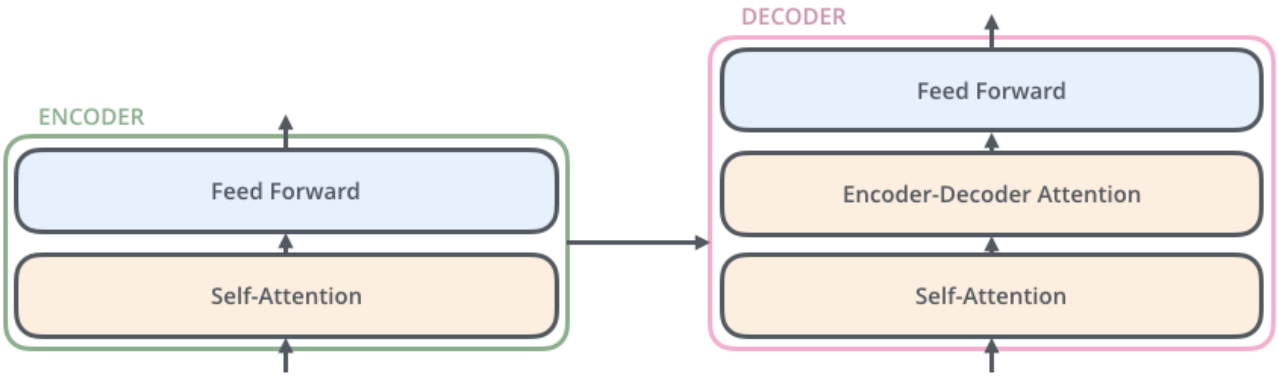


Figure 26: Transformer framework [28].

The above figure provides a high-level overview of the Transformer framework. BERT utilizes Transformer, an attention mechanism that learns the contextual relations between words in a text regardless of their respective positions. A basic Transformer consists of an encoder to read input text and decoder to perform prediction. BERT is essentially a trained Transformer encoder stack, enabling the embedding of input sentences. Of course, there have already been other word-embedding methods such as Word2Vec and Glove, but what makes Transformers and BERT different is the inclusion of Self-Attention. When the model processes each word in each position in the input text, Self-Attention enables it to look at other positions in the sequence for clues to create a better encoding [28].

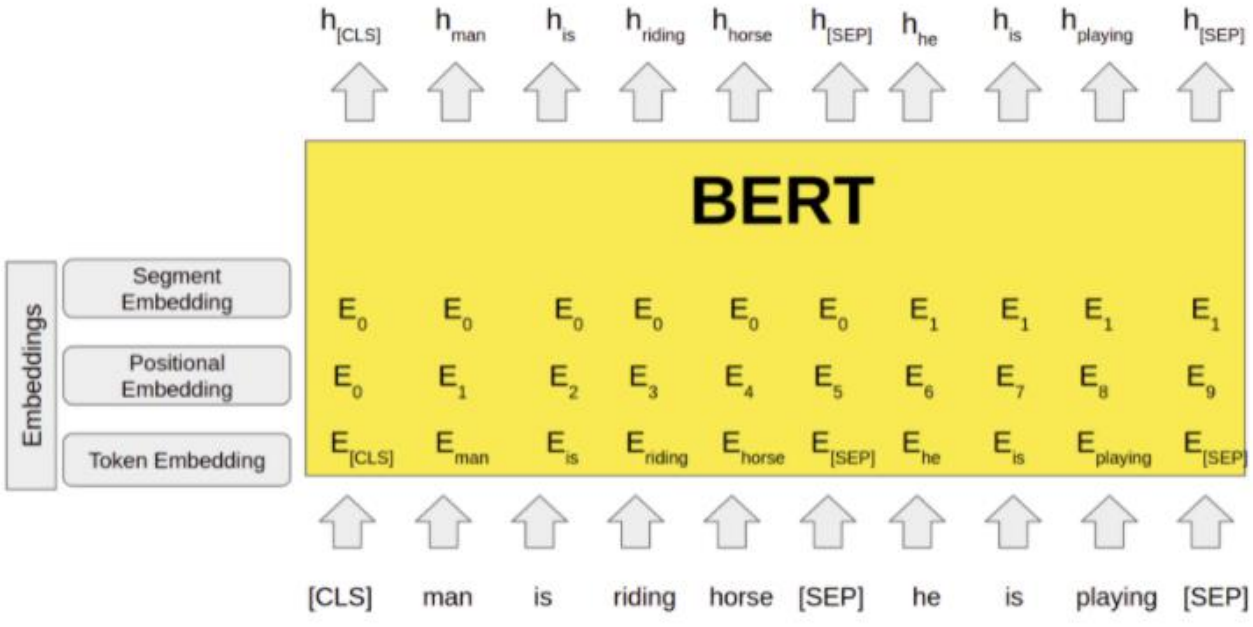


Figure 27: BERT high-level framework [29].

Figure 27 provides a high-level look at the BERT framework and its three embeddings. In the Token Embedding, BERT uses [CLS] as a special token to be inserted at the beginning of the first sentence and [SEP] to be inserted at the end of the sentence. For the Positional Embedding, BERT stores information about the position of tokens in the sequence. Finally, Segment Embeddings acts as a marker indicating the encoder to distinguish between sentences in sentence pairs. The embeddings are then summed and fed into deep bidirectional layers to get the hidden state vector which is used as an input for downstream NLP tasks like in our case, text classification of symptoms [29].

For our implementation of the BERT model for text classification for adverse effect symptoms, we use the Huggingface Transformers library and Keras TensorFlow. The base model we will use is the BERT-base-uncased which we will fine tune with our symptoms dataset as described in section 3.2.2.1.

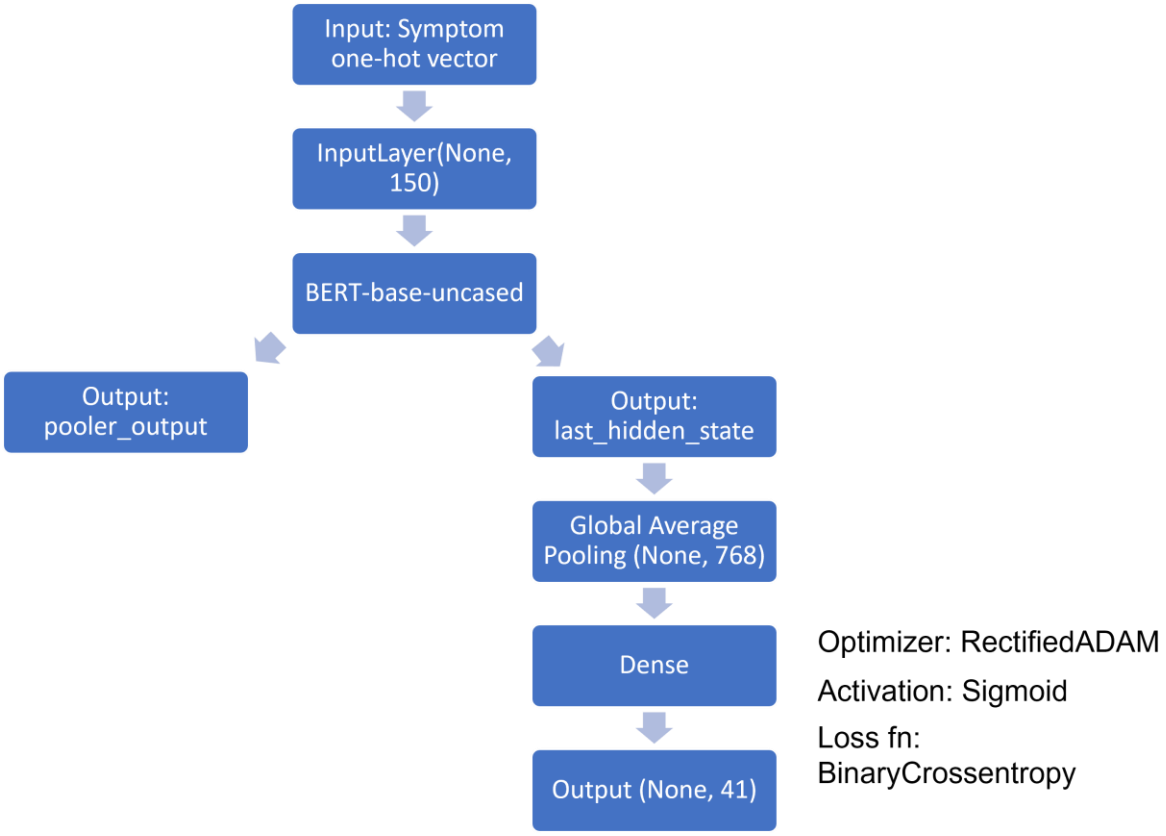


Figure 28: BERT text classification for adverse effect symptoms.

Figure 28 shows the overview for the text classification for adverse effect symptoms architecture using BERT model. For the input layer, 150 was derived from the max sequence length for tokenized symptoms free-text.

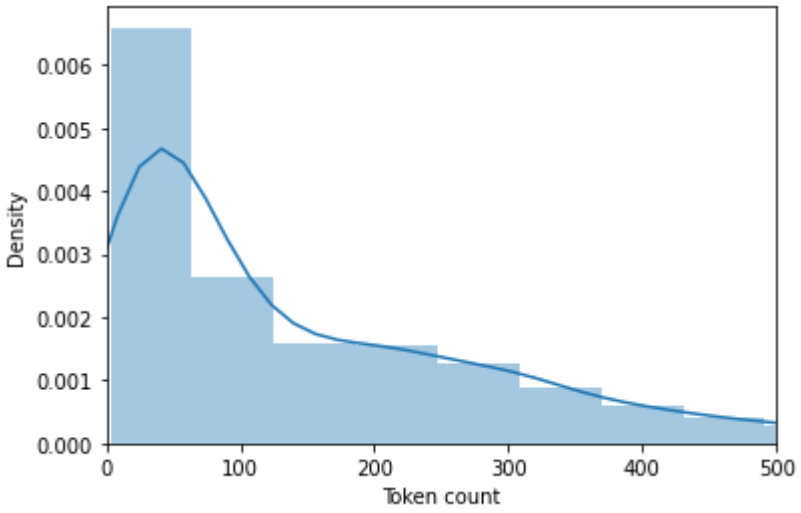


Figure 29: Distribution of length of tokenized symptoms free-text.

Running the tokenizer from the BERT-base-uncased base model on the training dataset and plotting the counts as seen in Figure 29, the point where the density starts to taper off is when token count = 150. Thus, max sequence length is set to 150 in the input layer.

After running BERT Huggingface returns two outputs: pooler_output and last_hidden_state. Pooler output is not used as it has been said in the Huggingface Transformer documentation to not usually be a good summary of the semantic content of the input. The documentation instead recommended to pool the sequence of hidden-states for the whole input. That is why we used a Global Average Pooling on the last_hidden_state output which pools the 768-dimensional embeddings for each token in the given sentence.

Finally, the pooled embeddings are used as an input to a dense layer for classification with an output of dimension 41 corresponding to the 41 down-sampled symptoms. Activation function used is sigmoid instead of softmax as out use case is multilabel and not multiclass classification. One key difference is that probabilities produced by the sigmoid function are independent of each other and are not constrained to sum to one as compared to in softmax where the outputs are all interrelated and the probabilities will always sum to one. For multilabel problems, we are not looking for the best one answer and instead accept multiple answers. Thus, sigmoid is more appropriate by design as it does take each symptom as independent and does not alter their probabilities. Binary cross-entropy is used as the loss function as each of the output classes (symptoms) is modelled as a single Bernoulli trial. Rectified ADAM is used as it rectifies the generalization and variance issues found in other adaptive learning rate optimizers by applying a warmup with a low initial learning rate and turns off momentum term for first few sets of input training batches.

3.2.2.5. Hybrid model

Even though we have determined the imbalance issue on the dataset and applied some method such as data down sampling and class weighted loss function, the model tends to predict fewer positive classes due to the large weightage of the negative classes in the dataset. To handle the multi-label classification, one of the ideal resolutions is to train each class by feeding with a balance dataset. However, the solution is not practical to be applied in the project because of the time constraints and increasing complexity on the system such as more model to be loaded in the system.

As we observed the imbalance issue found on the dataset, another resolution is to apply a hybrid method to combine all the output of three different model such as TF-IDF SVM Model, Word2Vec Deep Learning Model and Bert Model. Due to the rarity of the positive classes to be predicted by each model, any symptom predicted by the model will be considered as the confirmed symptom extracted through the text classification. The method to concatenate all the symptom is recognised as a OR rule based system as shown as below:

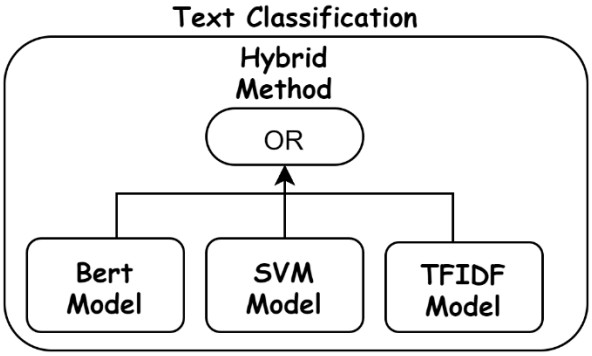


Figure 30: Text classification hybrid method.

3.2.2.6. Results & Discussion

For the TF-IDF SVM Model, with a train/test split of 90/10, we can obtain a micro-average precision recall AUC score of 0.84.

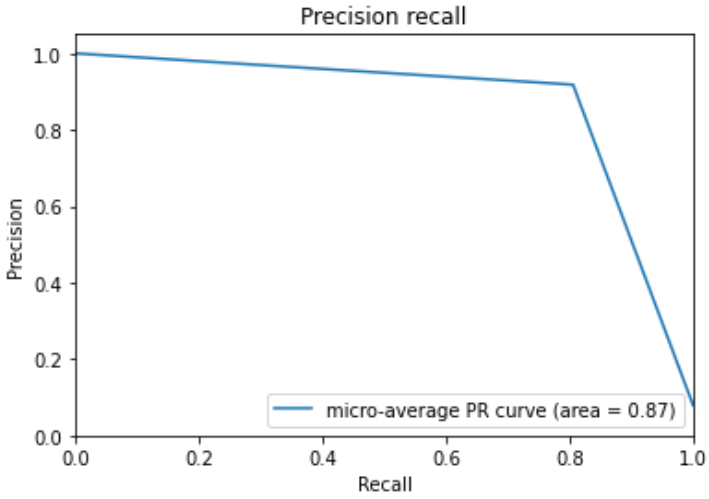


Figure 31: TF-IDF SVM model precision recall curve.

For the Word2Vec Deep Learning Model, with a train/test split of 67/33 and running for 15 epochs and 128 batch size, we can obtain a micro-average precision recall AUC score of 0.92.

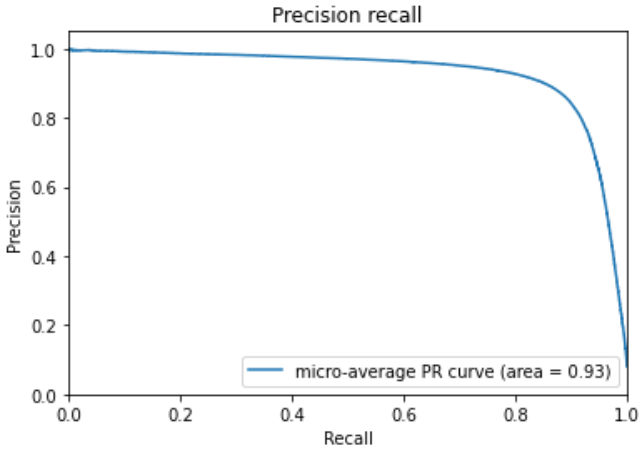


Figure 32: Word2Vec DL model precision recall curve.

From the observation of the experiment, the learning of the Word2Vec Deep Learning model is observed to be overfitted after 15 epochs and large batch size is helpful for speeding up the parallelism of the GPU so that the training can be performance faster and less overfit.

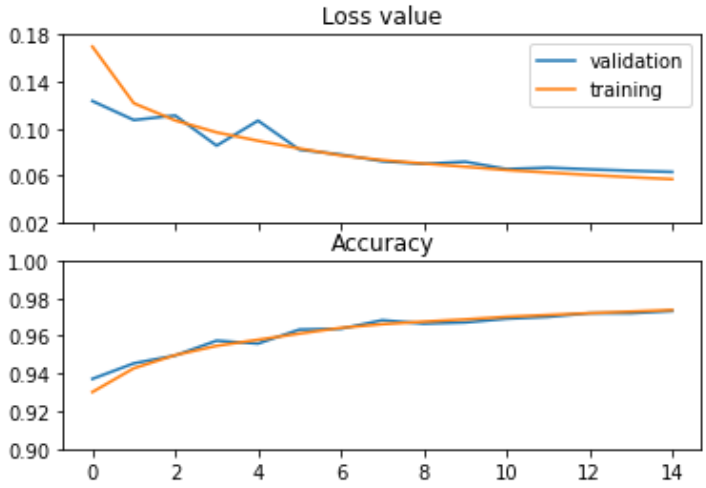


Figure 33: Training performance on each epoch.

For the BERT model, with a train/test/validate split of 80/10/10 and running for 2 epochs we obtained a micro-average precision recall AUC score of 0.93.

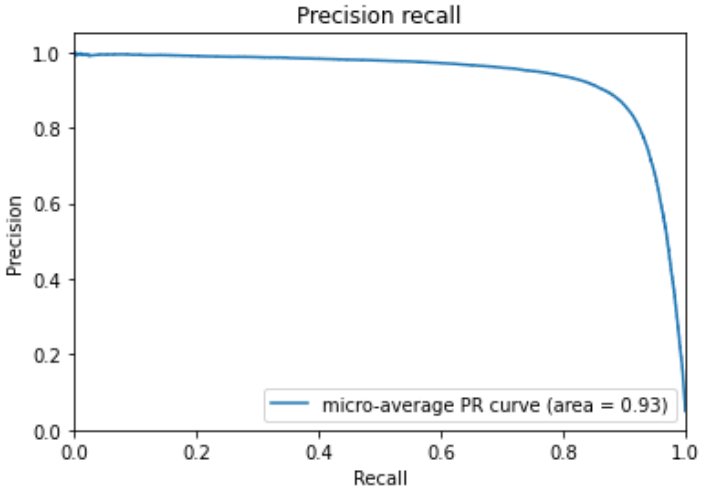


Figure 34: BERT model precision recall curve.

The reason why we only ran the model for 2 epochs is because BERT is very deep with 109,482,240 trainable parameters in the base model, and we did not want it to overfit the data.

After the combination of the three model, we obtained a micro average precision recall of AUC score of 0.89 as shown as below:

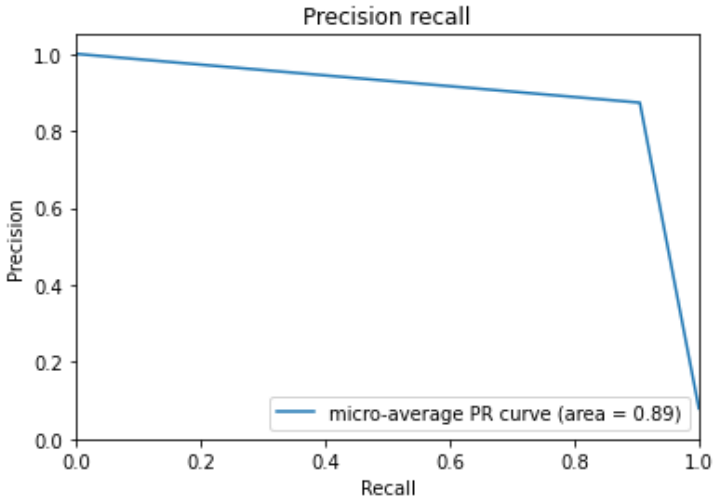


Figure 35: Hybrid model precision recall curve.

The area of the PR curve of the hybrid model is slightly lower than the Bert and Word2Vec DL Model because of the rounding of the prediction rate. We use micro-averaging for precision and recall as all the individual True Positives, True Negatives, False Positives and False Negatives for each class are summed up and their average is taken. The reason we used the AUC for the Precision Recall (PR) curve is because our symptom dataset even after down-sampling, still has symptoms where each patient has more true negative classes than positive. This is because as mentioned there are a total of over 14,000 unique symptoms in the data and we are training the model to predict the most populous 41. We needed a metric that will not be inflated due to the number of true negatives detected to have a good measure of model performance. We use the PR curve as Precision is not affected by many negative samples as it measures the number of true positives out of the samples predicted as positives (TP+FP).

Add to the fact that our task is to classify symptoms from free text entered by laymen to be used in downstream hospitalization prediction, we are more concerned with positive predictive value (Precision). We want to ensure that only people with actual symptoms have them correctly identified and if they are severe, have our system suggest to them to visit the hospital. Conversely, those who have symptoms correctly identified as not serious will have a lower priority to be referred to hospital to relieve the stress on the healthcare system. As seen in both cases, there is importance in correctly identifying symptoms for those who actually have it.

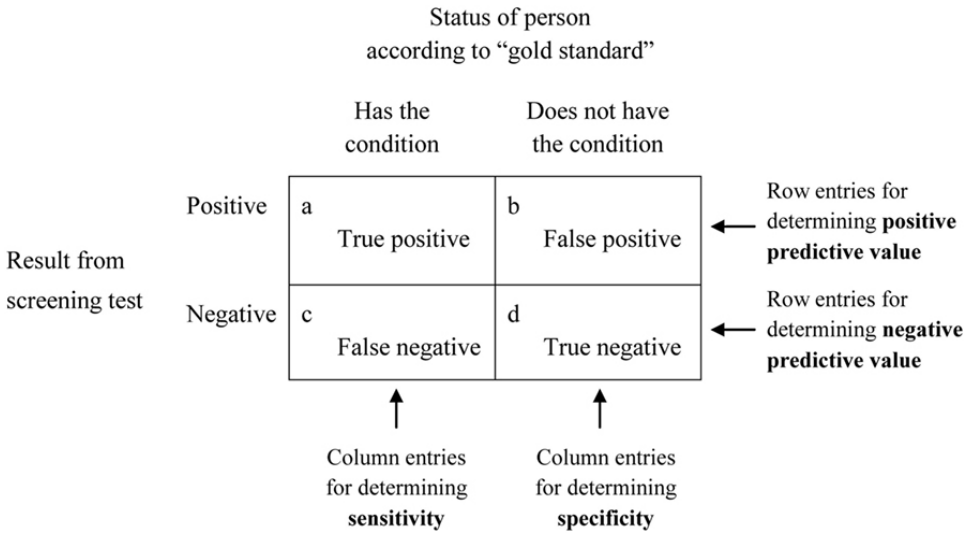


Figure 36: Positive predictive value in healthcare use-cases.

3.2.3. *Hospitalisation prediction and analytics*

This section uses VAERS dataset to predict hospitalization and death rate that happened post COVID 19 vaccinations, which allow the user of the app to have some confidence if they should check out their symptoms with doctors.

3.2.3.1. Data Pre-processing

The database structure is general and designed to be used for various vaccination. As this report focuses on COVID-19 adverse effect, many of the generic column can be dropped to clean the input dataset. The features that will be dropped with its reason is summarised in Table ***1*** with the trimmed features represented in Figure ***37*** below.

Table 1: Filtered Features from VAERS dataset.

Dataset	Data Type	Features	Reason why features ware dropped
VAERS data	Float	- Age in month	- COVID-19 vaccination is not available for infant
	Date	- Report Start Date - Report Completed Date - Date died (if applicable) - Vaccination Date - Symptoms Date - Recorded date	- The recorded date / vaccination date / symptoms date are for recording purposes. - Important features such as – *the number of days after vaccination when the symptoms starts* has already been extracted as feature.
	CHAR Category	- State - Vaccine administering facility - Vaccine sponsor / funded	- Vaccine administering facility / sponsor / state create sparse data with high memory while containing little information.
	FREE TEXT	- Lab data - Prior vaccination - Project report number - Medication - Current illness - Historical illness - Allergies	- >99% of Lab data is null for covid related vaccination - Prior vaccinations / Project report number are in free text and is removed for project simplicity - Although medication / current / historical illness / allergies might play a role in the hospitalization prediction, the features are dropped for simplification, to deal with for future text mining exercise.
VAERS vaccine	CHAR Category	- Vaccine Lot - Doses administered - Vaccination route - Vaccination site - Vaccination name - Administer Vaccine Type	- Vaccine lot contains thousands different category and is removed to prevent overfitting. - Doses administered / vaccination site / administer vaccine types are common for all COVID19 vaccines. - Vaccination name and dose administered are common with vaccine manufacturer, which is the feature kept.

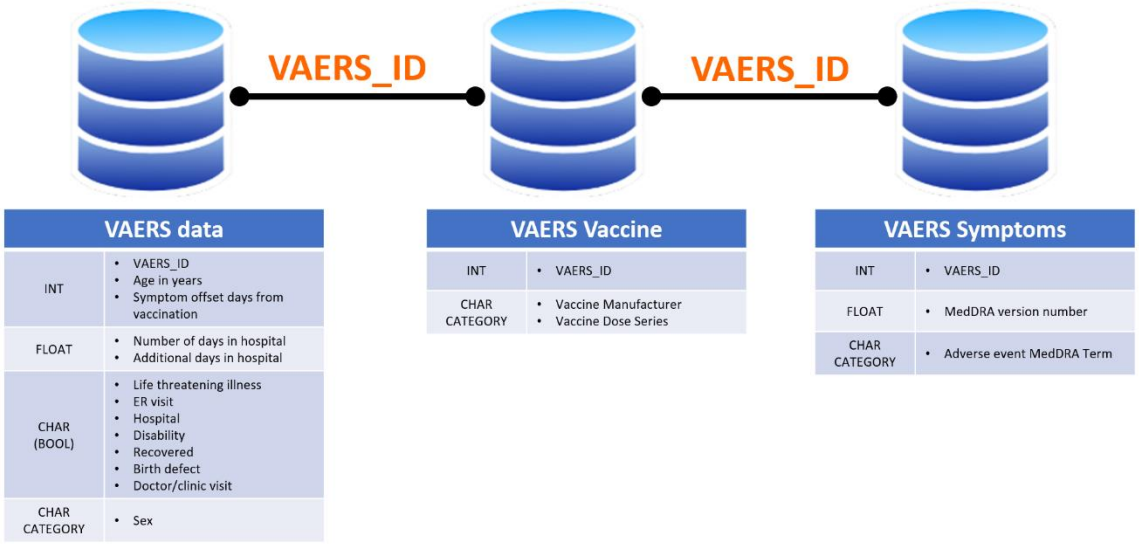


Figure 37: Post feature filtering on VAERS dataset.

In previous section, *3.2.2* Text classification for adverse effect symptoms , VAERS symptom dataset has been pre-processed and the chosen 41 symptoms features are one hot encoded. This section will be building on top of the previously used dataset combined with VAERS data and VAERS vaccine from Figure ***37***.

Firstly, duplicated rows and VAERS_ID were checked to ensure no data repetition and Null values were filled through the use imputer package. Data explorations are done on categorical data, one example is for *vaccine dose series* there are small number of individuals who get more than 3 doses in order to simplify the data and reduce the number of dimensions after one hot encoded, *vaccine dose series* for dose more than 3 doses are combined.

Secondly, hospitalization and death related parameters such as *number of days in hospital* and *additional days stay in hospital* are also dropped to ensure that the model are learning based on realistic and readily available parameter from the patient. The VAERS_ID column is also dropped as it only ser verse as identifiers and does not add any information about the individuals nor the vaccinations.

Next, Boolean features converted into a 1 and 0 instead of 'Y' and blank and symptoms columns are converted to one hot encoded symptom. This results in 2 integer features, 4 Boolean features 3 categorical features and 40 symptoms in Boolean form. In totals results in 60 features to be used in ML and NN to predict the hospitalization and death associated with COVID-19 Vaccines.

Through data exploration, it was found that there are only 6.25% hospitalized and 1.39% death record over a total training set of 458,222 data. This imbalance class and cause machine learning model skewed to predicting negative as compared to positive prediction.

3.2.3.2. Machine Learning Model

In this section 4 machine learning models will be explored to predict hospitalization based on the pre-selected 60 features. The four models trained are XGBoost, LightGBM, Random Forest, and Logistic Regression Classifier.

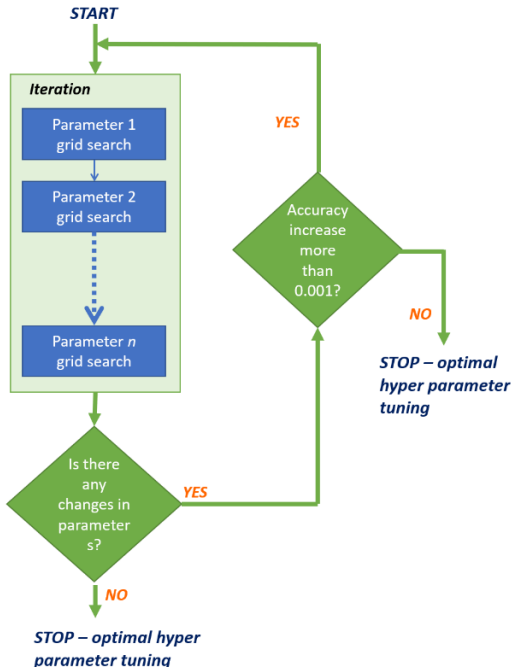


Figure 38: Machine Learning hyper parameter tuning.

Each machine learning's hyperparameter will be explored and tuned based on grid search and cross validation model. The overview hyperparameter tuning is summarized on the figure on the left. Whereby in every iteration, all variables are checked through cross validation and optimal values are search in ones/pairs. The iteration will only stop, when previous iterations do not have any other changes on the parameters or the accuracy has increased less than 0.001. This will prevent local optimization and enable higher probability to find the global optimal point. While also reducing the amount of resources used when there are no significant improvement in the model. XGBoost model will be used as a showcase while other models optimization details can be found in the attachment.

Initial values are chosen for the chosen hyperparameter. Parameters in blue are parameters to be kept constant, while the parameters in red are the parameters to train and optimize.

```
learning_rate =0.1, n_estimators=140, max_depth=5,
min_child_weight=1, gamma=0, subsample=0.8, colsample_bytree=0.8,
reg_alpha=0.00001, objective= 'binary:logistic', nthread=4,
scale_pos_weight=1, seed=21
```

After the values has been initialized, two parameters were chosen to be optimized though 5 cross validation and grid search. The chain of hyper parameter tuning python code are shown below. The first iteration and first grid search result optimal max_depth of and min_child_weight of 9 and 5 respectively. As both values are at the end of the range given, grid search is re-applied with higher graduality and more localized range which result in max_depth of 11 and min_child_weight of 6.

```
#%% 1st iteration
param_to_opt_1 = {
 'max_depth':range(3,10,2),
 'min_child_weight':range(1,6,2)
}
grid_search_1 = GridSearchCV(estimator = XGBClassifier(learning_rate =0.1, n_estimators=140, max_depth=5,
 min_child_weight=1, gamma=0, subsample=0.8, colsample_bytree=0.8,
 objective= 'binary:logistic', nthread=4, scale_pos_weight=1, seed=21),
 param_grid = param_to_opt_1, scoring='roc_auc',n_jobs=4,iid=False, cv=5)
grid_search_1.fit(X_train,y_train)
grid_search_1.best_params_, grid_search_1.best_score_
```

```
#%% finding max_depth and min_child_weight in higher granuality
param_to_opt_2 = {
 'max_depth':[8,9,10,11,12],
 'min_child_weight':[4,5,6,7,8]
}
grid_search_2 = GridSearchCV(estimator = XGBClassifier( learning_rate =0.1, n_estimators=140, max_depth=9,
 min_child_weight=5, gamma=0, subsample=0.8, colsample_bytree=0.8,
 objective= 'binary:logistic', nthread=4, scale_pos_weight=1, seed=21),
 param_grid = param_to_opt_2, scoring='roc_auc',n_jobs=4,iid=False, cv=5)
grid_search_2.fit(X_train,y_train)
grid_search_2.best_params_, grid_search_2.best_score_
```

Figure 39: Hyper parameter tuning. Top finding crude value of the hyper parameter. Bottom - smaller increments to find the best parameter.

The parameters in the model are the replaced with the newly found optimized max_depth and min_child_weight and the next parameter are searched. The optimal gamma value is found to be 0.4.

```
#%% searching for optimum Gemma
param_to_opt_3 = {
 'gamma':[i/10.0 for i in range(0,5)]
}
grid_search_3 = GridSearchCV(estimator = XGBClassifier( learning_rate =0.1, n_estimators=140, max_depth=11,
 min_child_weight=6, gamma=0, subsample=0.8, colsample_bytree=0.8,
 objective= 'binary:logistic', nthread=4, scale_pos_weight=1,seed=21),
 param_grid = param_to_opt_3, scoring='roc_auc',n_jobs=4,iid=False, cv=5)
grid_search_3.fit(X_train,y_train)
grid_search_3.best_params_, grid_search_3.best_score_
```

Figure 40: Repeating the optimization procedure on other parameters.

The same method is then applied to other hyperparameter to complete the first iteration of hyperparameter tuning optimization.

The first iteration of hyperparameter is summarized in the table below, with green cells showing the changes in the parameters. Since there are parameter still changed, the model must go through another iteration of the hyperparameter tuning, which is the same iterative process. The change in hyper parameters tuning is summarized in the second table below, with overall increase of accuracy from 0.9024 to 0.9280.

Table 2: First iteration of hyper parameter tuning in XGBoost.

Parameters	Initialized	Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7
max_depth	5	9	11	11	11	11	11	11
min_child_weight	5	5	6	6	6	6	6	6
gamma	0	0	0	0.4	0.4	0.4	0.4	0.4
subsample	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
colsample_bytree	0.8	0.8	0.8	0.8	0.5	0.4	0.4	0.4
reg_alpha	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001
reg_lambda	0	0	0	0	0	0	0.01	0.01
Learning rate	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Table 3: XGBoost hyper parameter change with iterations with each accuracy in each iteration.

Parameters	Initialized	Iteration 1	Iteration 2	Iteration 3
max_depth	11	12	15	15
min_child_weight	6	7	7	7
gamma	0.4	0	0	0
subsample	0.8	0.8	0.9	0.9
colsample_bytree	0.4	0.5	0.5	0.5
reg_alpha	0.00001	0.01	0.01	0.01
reg_lambda	0.01	0	0	0
Learning rate	0.1	0.03	0.05	0.05
Accuracry	0.9024	0.9188	0.9280	0.9280

Models and parameters		Initialized	Final
Xgboost	max_depth	11	15
	min_child_weight	6	7
	gamma	0.4	0
	subsample	0.8	0.9
	colsample_bytree	0.4	0.5
	reg_alpha	0.00001	0.01
	reg_lambda	0.01	0
	Learning rate	0.1	0.05
	Accuracry	0.90240	0.92804
RFC	n_estimators	100	500
	max_features	'auto'	log2'
	max_depth	8	30
	criterion	'gini'	'entropy'
	min_samples_split	2	20
	class_weight	{0:1,1:1}	{0:1,1:1}
	random_state	21	21
	Accuracry	0.90491	0.92731
Lightgbm	max_depth	10	17
	feature_fraction	0.8	0.7
	num_leaves	20	40
	boosting	'dart'	'dart'
	subsample	0.75	0.75
	learning_rate	0.1	0.1
	lambda_l1	0.01	0.01
	Accuracry	0.92653	0.92723
Logistic	C	0.1	1
	intercept_scaling	1	1
	fit_intercept	TRUE	FALSE
	penalty	l2	l2
	tol	0.0001	0.0001
	Accuracry	0.90424	0.90425

Figure 42: ML hyperparameter initial and finalized after tuning

Similar operations are performed on LightGBM, Random Forest, and Logistic with each individual model starting accuracy and final accuracy summarized in the table below. XGBoost and Random Forest Classifier are benefited the most with slightly above 0.02 increase in model accuracy and LightGBM and Logistic Classifier having minimum increment in accuracy after hyper parameter tuning. The model limitation will be evaluated in upcoming section.

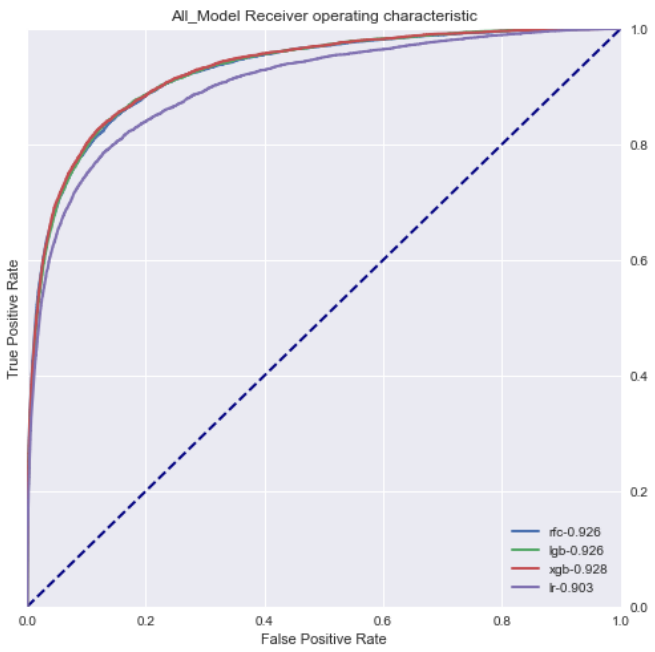


Figure 41: ROC curve for optimized ML model

For better prediction, 3 types of assembled model were tried out, ***majority voting*** with 2 positive predictions considered to be positive, ***AND logic*** prediction where all models must predict positive to output a positive prediction, and ***OR logic*** predictive positive to give an overall positive prediction. The models are evaluated and presented in section 3.2.3.4 results & discussion.

3.2.3.3. Neural Net Model

Deep Neural net model was explored, the *number of layers*, the *number of nodes* and *batch normalization* and *dropout* parameters, *learning rate* and *optimizer* are optimized through trial and error based on the movement in accuracy, f1-score and model loss. 2 NN architecture will be tested and compare, the first is conventional NN architecture 1 whereby the 59 features input are used as an input into the Multi-Layer Perceptron (MLP); and the second NN integrate the probability predicted from the ML train in the previous section to make the final predictions.

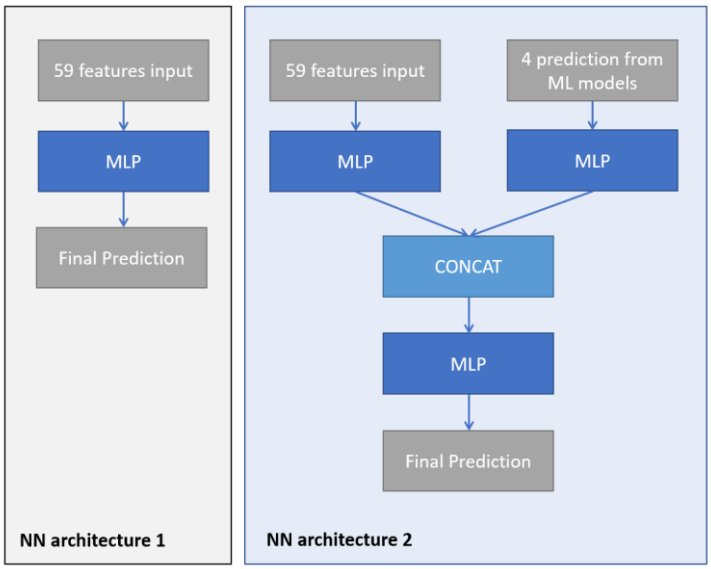


Figure 43: Two NN architecture evaluated.

Activation function - ReLU activation function is used for all the activation layers beside the last output layer which uses sigmoid function.

Number of layers – The number of dense layers in each MLP were varied, from 0 (direct prediction / combination) to up 10 dense layers.

Number of Nodes – The number of nodes in each MLP were also varied between 3 to 350. As the number is too many, large increment was used first (e.g. 100) followed by lower and lower granularity. One outline that was found useful were, n+1 layer will have less than or equal to nodes as n layer.

Learning Rate – Adam optimizer used with different learning rate / constant and exponential decay.

Batch normalization / Dropout / Regularization – were used to reduced overtraining and stricter implementation are used in layers closer to the output layer.

Activation function location – the activation function can be in the dense layer (before batch normalization) or post batch normalization. Both methods are explored, shown in the diagram below.

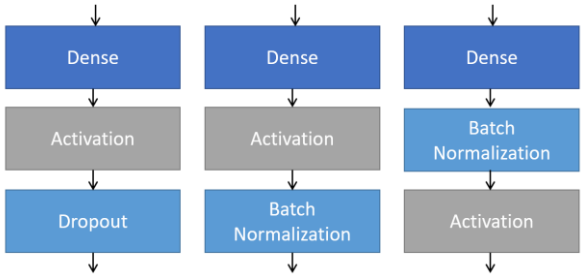


Figure 44: Activation, Batch Normalization and Dropout order evaluated.

The development of the models is found in the *Appendix A:* , with the best performing NN shown below. The maximum accuracy and f1-score on NN architecture 1 is 95.2043 and 0.4998, respectively while for NN architecture 2, the maximum score obtain are 95.4792% accuracy and 0.5544 f1-score.

NN architecture 2, or the assemble model, has slightly higher prediction score. Suspect due to the NN able to distinguish the limitation of each ML.

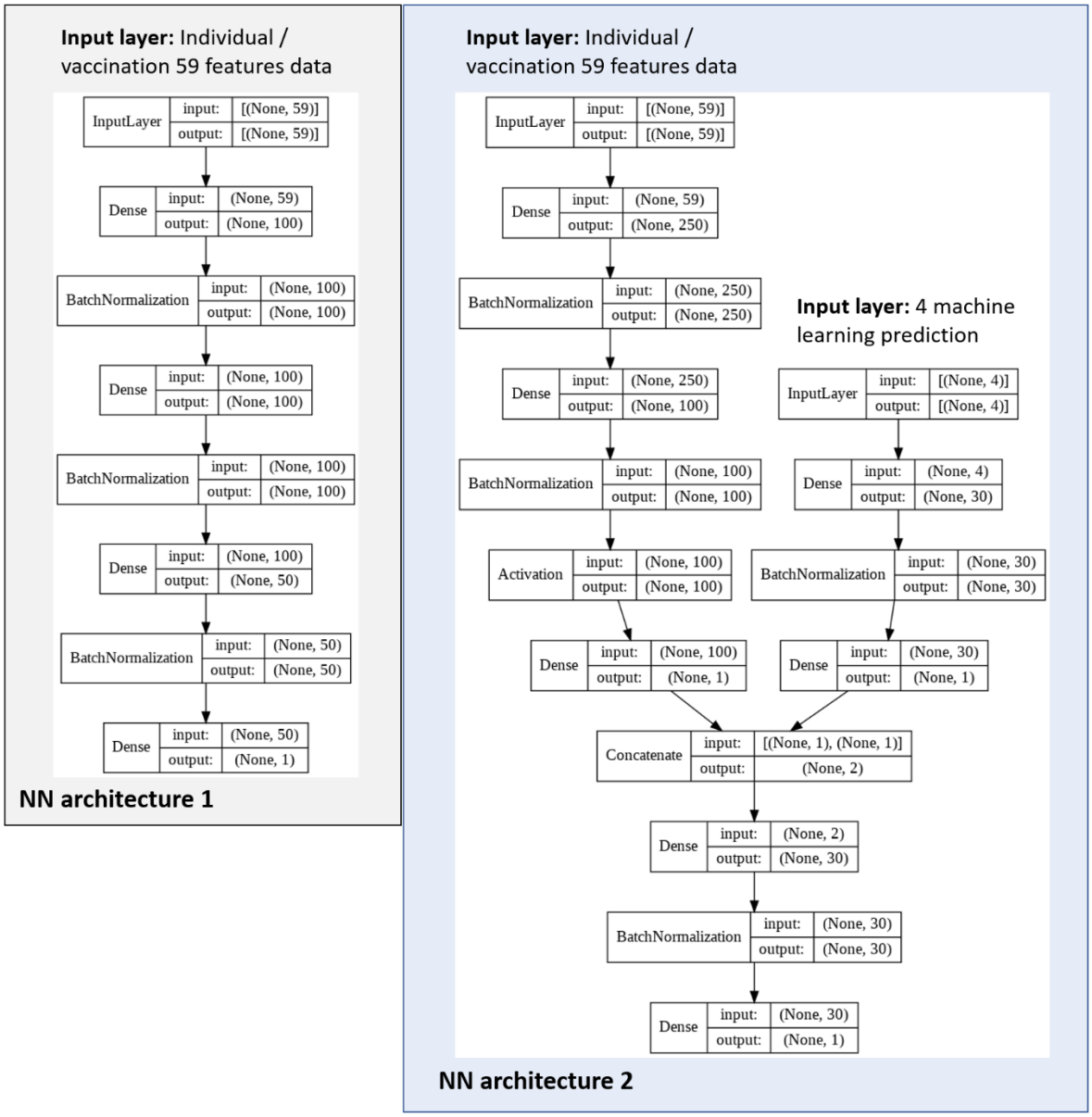


Figure 45: Best NN models for architecture 1 and 2.

3.2.3.4. Hospitalization Prediction Results & Discussion

The model train in ML and NN section are summarized in the table below. Number mark in green and blue are the highest and second highest value, respectively.

Table 4: Summary of all the trained and optimized model.

Models	Average Training Time per model	Train				Test			
		Accuracy	recall	precision	f1-score	Accuracy	recall	precision	f1-score
Xgboost	1.5 min	**98.7795**	**0.8383**	**0.9614**	**0.8956**	94.8147	0.4708	0.6125	0.5324
RFC	5 min	97.9742	0.7088	0.9554	0.8138	95.3516	0.4680	0.7188	0.5467
Lightgbm	1 min	98.5255	0.8100	0.9462	0.8728	94.9227	**0.4732**	0.6258	0.5389
Logistic	0.2 min	98.256	0.7847	0.9248	0.849	94.8245	0.4126	0.6341	0.4999
Majority Voting	-	**98.5869**	0.8208	0.9459	0.8789	95.3833	0.4690	0.6955	**0.5602**
AND logic	-	98.375	0.7667	**0.9662**	0.855	**95.4815**	0.3884	**0.7807**	0.5188
OR logic	-	98.5351	**0.8554**	0.9049	**0.8795**	87.4810	**0.8258**	0.3118	0.4527
NN	30 min	95.2861	0.3963	0.7243	0.5123	95.2043	0.3822	0.7221	0.4998
NN + ML	30 min	96.5475	0.5518	0.8408	0.6663	**95.4792**	0.4485	**0.7257**	**0.5544**

Observation and findings:

- ML models generally overfit on the test dataset although **cross validation** has been implementing during hyperparameter tuning and **regularization** / **learning rate** is part of tuned parameter.
- The drop in accuracy from ML model can be understand through the drop in ~50% of the recall value and ~30% of precision value when comparing the train and test dataset.
- **Majority voting** have the best overall f1-score (dynamic mean of recall and precision) which is an important factor to consider as the data of interest are imbalanced.
- **AND logic voting** has the highest accuracy and precision. This can be understood as AND logic require all ML model to predict positive result to output positive output, which will result in predicting more true cases from all predicted case (high precision) and lower number of predicted case and false negative cases (low recall). The minimum precision value that AND logic voting will have is equivalent to the highest precision of the individual ML model evaluated.
- **OR Logic voting**, or the opposite of AND logic have a very high recall values and low precision as it predicts more cases as a positive case due to more lenient constraint.
- **NN** – the best NN observed not to be overfit from the difference between train and test models,
- **NN + ML** – although does not have the best individual performance matrix (accuracy, recall, precision, f1-score) however it is second in accuracy, precision, and f1-score.

NN + ML score are chosen with a balance of the scoring matrix to predict individua hospitalization rate. With estimated prediction time of less than 2s per cases.

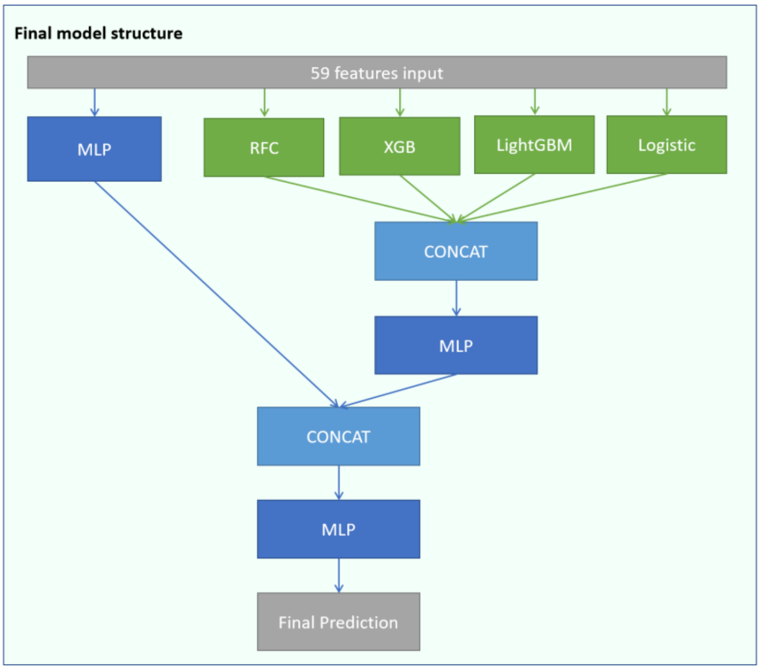


Figure 46: Overall model structure in predicting the hospitalization occurrence for individuals.

Albeit the optimization that has been done to tune the ML and NN models, the accuracy continues to hover around 0.93-0.96 with recall and precision value < 0.9 due to 2 main reasons. Upon further investigation the models are unable to increase in accuracy as the data are **not separable** as data with the same exact input might have different output. Below are some examples that were found. On the left of the table are the vaers_ID (individual identifier) and hospitalization. The middle are the input data into ML and NN, whereby there are 2 standardized numerical input and 57 Boolean input. In the table blue represent 0 and orange represent 1 for illustrative purposes. As can be seen in the table the input can be exactly the same for all the Bolen inputs and only slight differences in age / number of days since vaccination with different hospitalize prediction output. In this case, highlight suspect that the two classes are mixed and not separable. More data are needed to increase the separation between the classes and without its further improvement might be impossible as shown in Figure ***48***: Illustration of classes separability .Figure ***48***: Illustration of classes separability .

VAERS_ID	HOSPITAL	AGE_YRS	NUMDAYS	SEX__F	SEX__M	SEX__U	BIRTH_DEFECT__0	BIRTH_DEFECT__1	VAX_MANU__JANSSEN	VAX_MANU__MODERNA	VAX_MANU__PFIZER\BIONTECH	VAX_MANU__UNKNOWN MANUFA	VAX_DOSE_SERIES__1	VAX_DOSE_SERIES__2	VAX_DOSE_SERIES__3	VAX_DOSE_SERIES__3+	VAX_DOSE_SERIES__UNK	SYMPTOMSheadache	SYMPTOMSpyrexia	SYMPTOMSfatigue	SYMPTOMSchills	SYMPTOMSpain	SYMPTOMSdizziness	SYMPTOMSnausea	SYMPTOMSpain in extremity	SYMPTOMSinjection site pain	SYMPTOMSarthralgia	SYMPTOMSmyalgia	SYMPTOMSrash
1407670	1	2.333947	0.979292	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1376938	0	-1.961383	-0.620389	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1417262	1	2.446982	2.934457	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1010831	0	-0.322375	0.979292	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1204764	1	1.090562	2.045745	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1103205	0	0.525387	-0.442647	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1296437	1	1.37315	2.934457	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1291965	0	-0.944068	2.934457	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1317532	1	0.525387	2.934457	1	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1207224	0	1.82529	0.09058	1	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
923402	0	-0.548445	-0.620389	1	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0
1402656	1	-0.491928	2.756715	1	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0
1378459	1	-1.904865	-0.087162	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1400147	0	0.412352	-0.620389	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1194793	1	1.938325	0.801549	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1421559	0	1.938325	1.157034	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1279592	1	1.881807	-0.442647	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
936738	0	2.27743	-0.087162	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1255716	1	-1.170138	2.40123	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1320242	0	-4.02E-16	1.157034	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1018647	1	-1.33969	-0.442647	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1324493	0	-0.491928	2.934457	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

VAERS_ID	SYMPTOMSdyspnoea	SYMPTOMSinjection site erythema	SYMPTOMSpruritus	SYMPTOMSvomiting	SYMPTOMSinjection site swelling	SYMPTOMSasthenia	SYMPTOMSinjection site pruritus	SYMPTOMShyperhidrosis	SYMPTOMSparaesthesia	SYMPTOMSerythema	SYMPTOMSdiarrhoea	SYMPTOMSurticaria	SYMPTOMShypoaesthesia	SYMPTOMSlymphadenopathy	SYMPTOMSfeeling abnormal	SYMPTOMSmalaise	SYMPTOMSchest pain	SYMPTOMSinjection site warmth	SYMPTOMSsyncope	SYMPTOMSperipheral swelling	SYMPTOMScovid-19	SYMPTOMSblood test	SYMPTOMSsars-cov-2 test positive	SYMPTOMSelectrocardiogram	SYMPTOMScomputerised tomogran	SYMPTOMSechocardiogram	SYMPTOMSpulmonary embolism	SYMPTOMScerebrovascular acciden	SYMPTOMStroponin increased	DISABLE	RECOVD	OFC_VISIT	GROUP
1407670	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
1376938	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
1417262	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	2
1010831	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	
1204764	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
1103205	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1296437	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1291965	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1317532	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	4
1207224	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
923402	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
1402656	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1378459	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
1400147	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1194793	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	7
1421559	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	
1279592	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	8
936738	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
1255716	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	9
1320242	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
1018647	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	10
1324493	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

Figure 47: Table representing same input different prediction targeted output

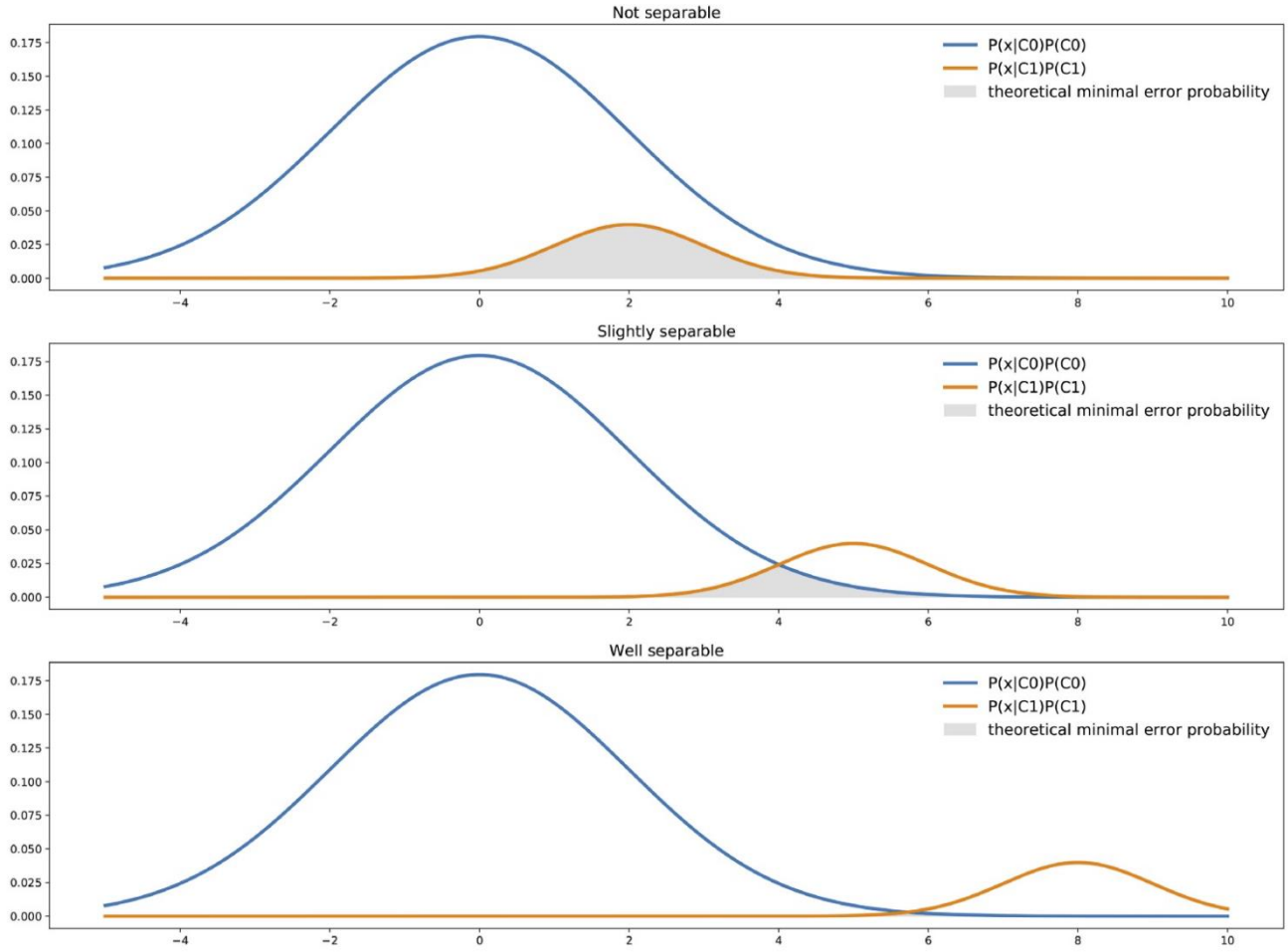


Figure 48: Illustration of classes separability [30].

Secondly the data are limited to VAERS dataset whereby individual purposely reports their vaccine adverse effect. This means, the possible data from people who are vaccinated and choose not to report their symptoms are not recorded and available, similar to people who are vaccinated but do not experience any symptoms. This skew the data to those who have symptoms and those who are more likely to report their symptoms.

3.2.4. *Covid vaccine literature mining and recommendation*

3.2.4.1. Data Pre-processing

Within the CORD-19 dataset, there are entries with missing values and duplicated articles. Pre-processing was done by first selecting the range of published date from 01/01/21 to 01/11/21, followed by dropping the data with missing values and duplicated entries. A total of 83,287 articles remains in the dataset. To further reduce the amount of processing data to a manageable size, the dataset was further narrowed to the topic of "covid-19 vaccine adverse effect". This was conducted by tokenizing the article abstract, followed by filtering of keywords: Covid-19, Vaccine, Adverse. After filtering, 296 articles formed the subset of relevant articles to the project.

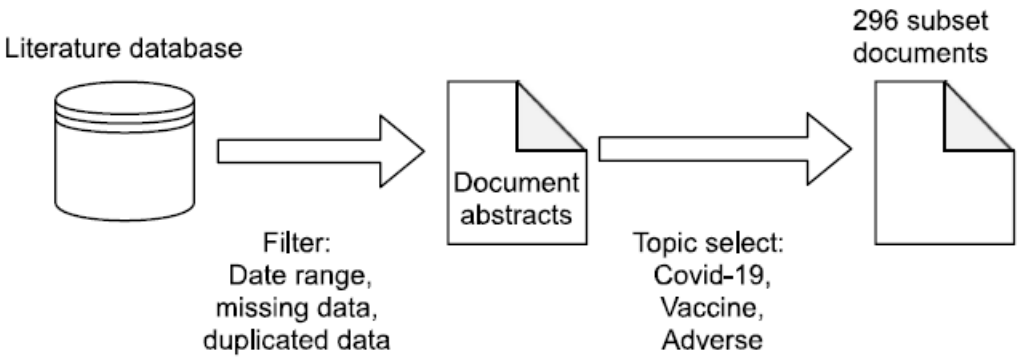


Figure 49: Pre-processing of Cord-19 dataset.

3.2.4.2. Literature mining methodology & output

The overall approach used for literature mining can be split into multiple tasks: paragraph identification, question & answer retrieval and sentiment analysis. This approach is inspired from related work by Salesforce researchers Esteva et al., where they developed the CO-Search methodology to achieve state of the art results in information retrieval [31]. In this work, due to limitations in computational capabilities, pretrained weights were utilised and fine-tuning was conducted to apply for our intended tasks.

Figure ***50*** below shows the workflow of literature mining. It contains 2 segments which are indexing and retrieval. In the indexing segment, the CORD-19 subset documents are mapped to a feature space using 2 methods: sentence level embedding using Siamese-BERT (SBERT) and word-based frequency statistics using TF-IDF. For the retrieval segment, the query is obtained from the symptoms generated from text classification module in section 3.2.2. It is further converted into a statement that forms a standard query about the relationship of the symptom to vaccine adverse events. In order to identify the relevant paragraphs to the query, the query was mapped to the similar feature space and cosine similarity was used as the metric to measure the similarity of the query to the embedded paragraphs in vector space. Scoring function was used to combine SBERT and TF-IDF scores, and the 5 paragraphs with the highest scores were selected for Q&A retrieval to the input query. Sentiment analysis was then used to determine if there are any alerts of negative outcome that should be provided to the user.

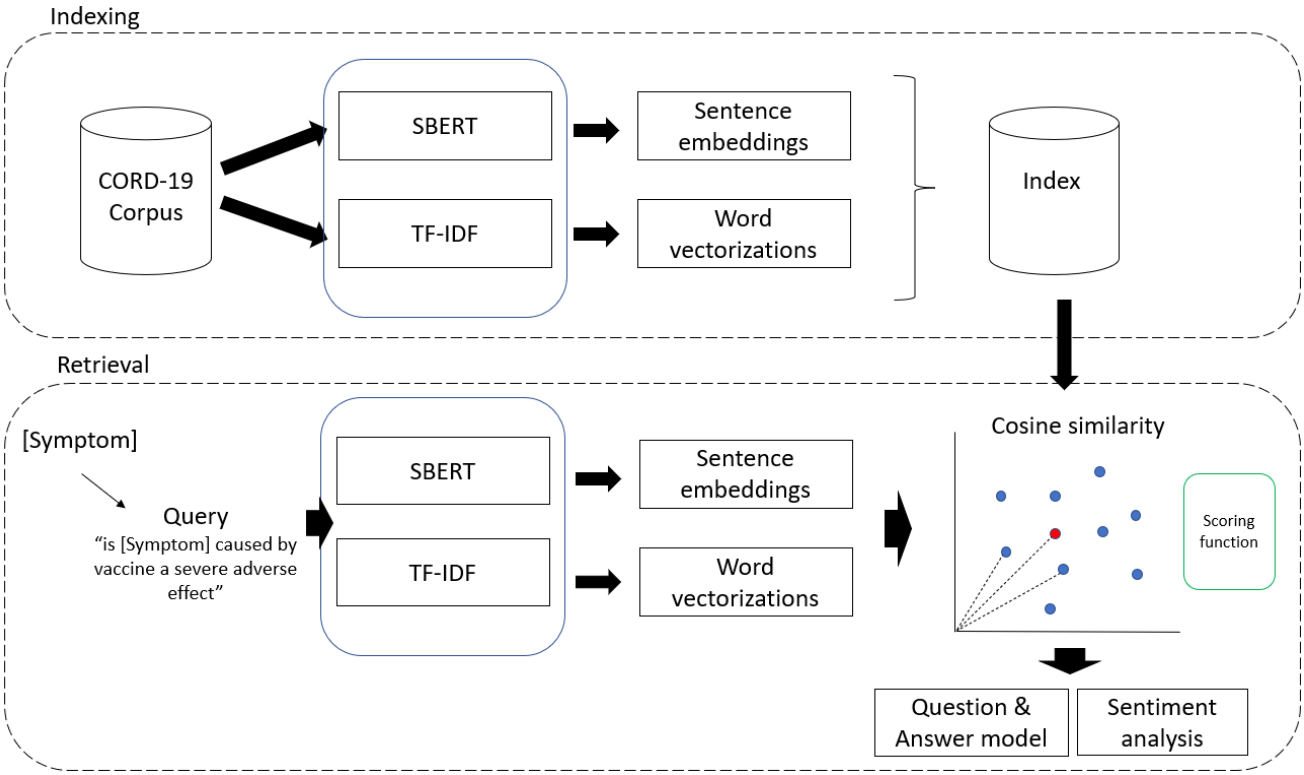


Figure 50: Workflow for literature mining of Covid Vaccine Adverse Effect.

SBERT is the sentence level embedding methodology proposed by Reimers & Gurevych [32]. It consists of 2 separate BERT models which takes in 2 sentences and maps them on an embedding space. The cosine similarity score is returned for 2 input sentences within the embedded space, representing the semantic relationship of entire sentences. To embed the queries using SBERT, SentenceTransformer package was utilized and 'msmarco-dilstilbert-base-v4' model was used as the embedder. 'msmarco-dilstilbert-base-v4' model was pre-trained on a large scale corpus (8.8 million passages) based on Bing search engine, and used for semantic search of phrases or questions to find relevant passages. In this case, where the query is much shorter than the retrieved paragraph, it is also termed as asymmetric semantic search [33]. Hence, the model suits the asymmetrical nature of the Covid vaccine symptom query and literature-obtained paragraphs.

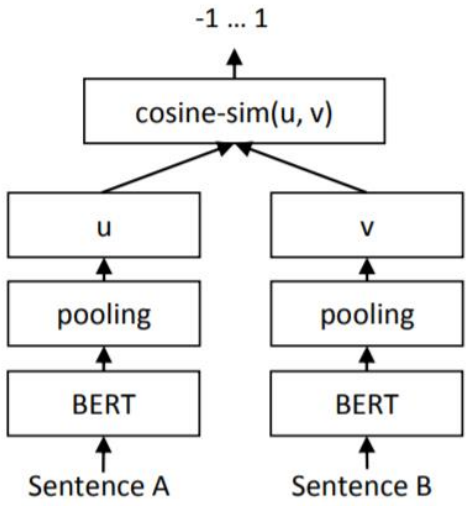


Figure 51: SBERT architecture for cosine similarity calculation [32].

For TF-IDF, each paragraph was tokenized and vectorization was performed using Sklearn.feature_extraction package to derive a vocabulary list. This list was matched to the vectorized query to obtain the cosine similarity score. Using the cosine similarity scores from both SBERT and TF-IDF, a linear scoring function was used,

$$Score = k(Cos\_sim_{SBERT}) + (1-k)(Cos\_sim_{TF-IDF})$$

where k is the ratio coefficient between the two components. The value of k distributes the weightage between semantic and statistical search, as illustrated in Table 5 below. When the value of k is large, the returned paragraph addresses the query on symptom-vaccine relationship but the correct symptom may not be captured. On the other hand, when k is small, the symptom is present but it does not directly discuss the outcome related to vaccine adverse events. For the final model, a value of 0.7 was selected to have good observed outcome, similar to [31].

Table 5: Example of returned text for different score coefficient k values.

Symptom: Fever	k = 0.9	k= 0.1	k = 0.7
Sample text	'In the population analysed, only one serious adverse event (0.05%) was reported in an approximately 30-year-old female subject, who was immediately hospitalised after the second dose of vaccine due to symptoms similar to anaphylaxis (skin rash and glottal oedema). This serious adverse reaction was completely resolved within 24 hours.'	'In general, the fever of tumor patients can be divided into tumor fever, drug fever, and infection. Tumor fever refers to the response of the immune system to tumor necrosis and stress factors produced by tumors which is generally ≤38.5 • C with no inducement and lasts for a long time. Experimental treatment with antibiotics is ineffective (Pasikhova et al., 2017) . Drug fever means that fever occurs during medication and stops after drug withdrawal. The median time of fever caused by antineoplastic drugs is about 0.5 days after administration (Patel and Gallagher, 2010) . Fever caused by infection usually has a higher temperature and can be reflected by blood routine tests.'	'Overall, the most frequent adverse effects reported after the administration of the three vaccines considered in this review consisted of local reactions at the injection site (sore arm and erythema) followed by non-specific systemic effects (myalgia, chills, fatigue, headache, and fever). These adverse reactions are considered as the result of the immune system's reaction that occurs soon after vaccination (also referred to as reactogenicity) and resolves shortly [35] . Given the scale-up of mass vaccination campaigns across the world, severe adverse events may occur and will likely generate concerns among patients and requests for evaluation.'

Selection of the 5 most relevant paragraphs was based on highest combined score ratings, and used for question-and-answer retrieval. BioBERT, a pre-trained language representation model for biomedical text mining, was used as the base model for subsequent fine-tuning. As BioBERT was pre-trained on biomedical corpora in PubMed, it has shown better domain-specific performance in understanding biomedical texts [34]. Fine-tuning of BioBERT was conducted using SQuAD2.0 (Stanford Question Answering Dataset Version 2), which contains over 150,000 questions to retrieve segment of answers from passages or determine as unanswerable [35]. To prevent overtraining, 3 epoch was used to fine-tune the model and the evaluation score is reported in Table 6 below. During fine-tuning, training hyperparameters were: learning rate=2e-5, batch size=16, Adam

optimizer and linear learning rate scheduler was used. The final model, "biobert-v1.1-pubmed-finetuned-squad", can be found on https://huggingface.co/gerardozq/biobert_v1.1_pubmed-finetuned-squad, saved at the second epoch.

Table 6: Training and validation loss of BioBERT fine-tuned SQuaD2.0.

Epoch	Training loss	Validation loss
1	1.0213	0.9448
2	0.9478	0.9625
3	0.9189	1.0306

Next, sentiment analysis was performed on the retrieved answers from the query, to determine the risk alert of the reported adverse symptoms. VaderSentiment package was used to distinguish between negative, neutral and positive sentiments, which is based on VADER (Valence Aware Dictionary and sEntiment Reasoner), a lexicon and rule-based sentiment analysis tool [36]. For example, Figure ***52***a shows that if the adverse outcomes are not severe, it will be classified as neutral, while Figure ***52***b reflects the scenario when a negative outcome is detected. As a rule, if the negativity score is above 0.5, an alert will be reflected on the UI to inform the user on potential risks.

a) Neutral sentiment

```
 4: {'text': 'This is among the first series to report multiple cases of myocarditis in adults following vaccination against SA
RS-CoV-2. All four patients were young, between 20 and 30. All presented with chest pain two to five days after their second va
ccine dose. All had significantly elevated troponin-I levels. Though one had a viral prodrome, all had negative serologies. Non
e reported prior COVID-19 infection. None had stigmata of autoimmune disease, and the one who underwent a rheumatologic workup
while hospitalized had unremarkable autoimmune serologies. Reassuringly, the two patients who have returned for follow up in th
e weeks following discharge had normalized CRP values and denied symptom recurrence.',
  'title': 'Myocarditis following mRNA vaccination against SARS-CoV-2, a case series',
  'authors': 'King, William W.; Petersen, Matthew R.; Matar, Ralph M.; Budweg, Jeffery B.; Cuervo Pardo, Lyda; Petersen, John
W.',
  'publish_time': '08/09/2021',
  'journal': 'Am Heart J Plus',
  'sentiment': {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}},
```

b) Negative sentiment

```
 3: {'text': 'A 66-year-old male with past medical history of hypertension, hyperlipidemia, and surgically resected renal cell
carcinoma (RCC) presented to the emergency department with a chief complaint of right flank pain and right pleuritic chest pai
n. He was in his usual state of health until he received his second dose of the Moderna SARS-CoV-2 vaccine dose 10 days prior.
24 h after the immunization he experienced fevers, chills, and arthralgias that transitioned to progressive right-sided flank p
ain and pleuritic chest pain. There was no history of recent immobility or surgery, and he was exercising daily before presenti
ng. He denied any personal or family history of VTE, PE, or known hypercoagulable state.',
  'title': 'A Case of Acute Pulmonary Embolus after mRNA SARS-CoV-2 Immunization',
  'authors': 'Wiest, Nathaniel E.; Johns, Gretchen S.; Edwards, Eric',
  'publish_time': '08/14/2021',
  'journal': 'Vaccines (Basel)',
  'sentiment': {'neg': 0.524, 'neu': 0.476, 'pos': 0.0, 'compound': -0.5106}},
```

Figure 52: Examples of a) neutral and b) negative sentiments for symptom "chest pain".

Finally, the combined output of recommended articles to symptoms are provided to display on the UI, which contains the relevant paragraph, document details and sentiment alert (Figure 53). Through these articles, the user will have access to factual information based on scientific studies to improve their understanding and confidence towards vaccination outcomes.

UI output:

{1: {'text': 'Severe allergic reactions due to anaphylaxis represent a very rare side effect associated with most vaccines, including mRNA vaccines. For this reason, it has been recommended that subjects with a previous history of severe allergic reactions to any vaccine ingredient should not receive it [36] . This safety concern is currently followed-up via routine pharmacovigilance.',
'title': 'Safety of COVID-19 vaccines administered in the EU: Should we be concerned?',
'authors': 'Hernández, Antonio F.; Calina, Daniela; Poulas, Konstantinos; Docea, Anca Oana; Tsatsakis, Aristidis M.',
'publish_time': Timestamp('2021-04-20 00:00:00'),
'journal': 'Toxicol Rep',
'sentiment': {'neg': 0.545, 'neu': 0.455, 'pos': 0.0, 'compound': -0.5859}},

Recommended articles:

1. 'Severe allergic reactions due to anaphylaxis represent a very rare side effect associated with most vaccines, including mRNA vaccines. For this reason, it has been recommended that subjects with a previous history of severe allergic reactions to any vaccine ingredient should not receive it [36] . This safety concern is currently followed-up via routine pharmacovigilance.'

   Title: Safety of COVID-19 vaccines administered in the EU: Should we be concerned?
   Authors: Hernández, Antonio F.; Calina, Daniela; Poulas, Konstantinos; Docea, Anca Oana; Tsatsakis, Aristidis M.
   Journal: Toxicol Rep
   Publish date: 2021-04-20
   Alert:

if neg>0.5, orange
Else: grey

Figure *53*: Example of UI display output with relevant document details from literature mining module.

3.3. System design

The Covid-19 Vaccine Adverse Event Reporting Analyzer System is a full-stack single page application solution which provide the patient an interface to report any unwell condition and behaviour after vaccination and facilitate the medical staff to understand the symptoms suffered from the patient. The full system architecture shown in Figure ***54*** mainly consist of two parts: client-side application and server-side application. By using Django a high-level Python Web framework, the front-end and backend application are fitted into a gapless application to perform better.

The client-side application is the web template which is based on the reactJS framework and generated by using webpack. The routing of the pages has been configured into the web template and this web template will then be served by the Django server under the main endpoints ('/').

Server-side applications will also be served with the Django server under the same endpoint with different directory. All the events to query or store data into the database will call the different web services to trigger different web event algorithm in the Django Framework. The serialized model also facilitates the data extraction from the database to efficiently query massive data from database.

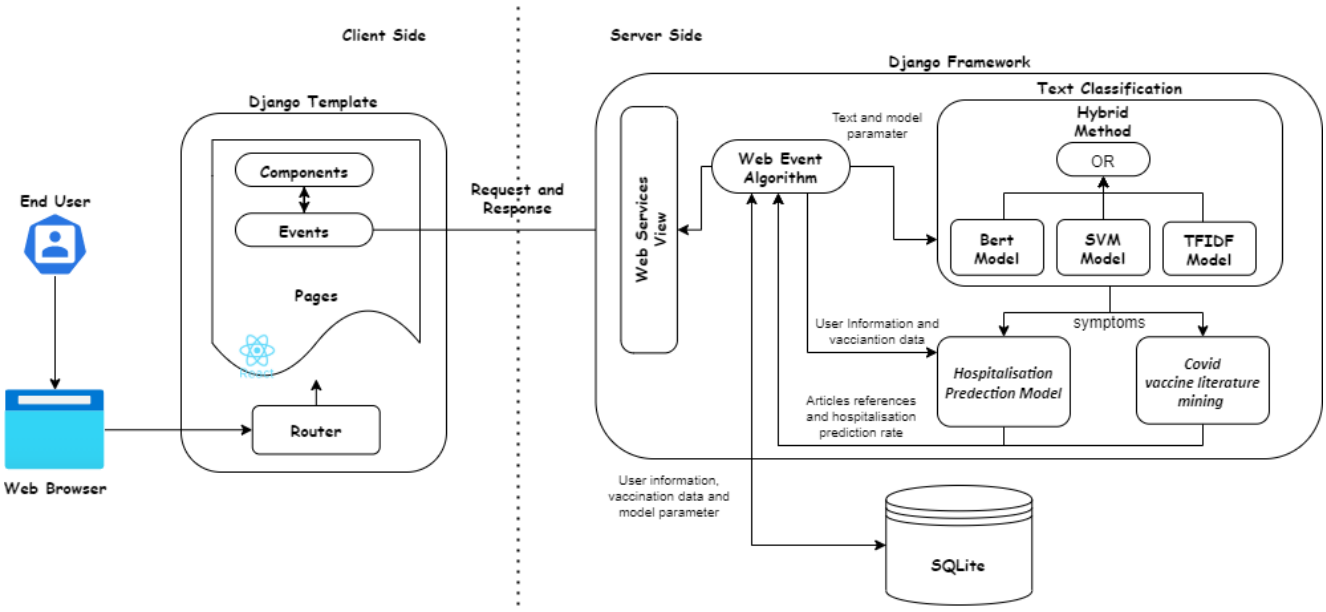


Figure 54: System Architecture of the Reporting Analyzer System.

Data collection and alignment are always most prioritised concern to be resolved. The models are pretrained with the training dataset, however, the current user data is required to be stored to aid the data transaction between the user interface components.

Django framework provided a favourable method to create and define the SQL database. Data type and structure are defined clearly in the Django Model and the relation of the primary keys and foreign keys between the tables gave us data integrity on the low level to prevent creating a record which does not fulfil the relation. The application data are divided into 5 SQL tables with each table representing a clear data boundary to reduce the storage size. Data in the *MODEL_THRESHOLD* SQL table was defined according to the number of types of the models used in the system, while there remaining SQL tables are filled from user interaction. The databases representation detail shown in Table ***7*** and their connection best described in the database diagram below (Figure 55).

Table 7: Database table and event representation.

SQL Table	Event (row) representation
MODEL_THRESHOLD	A threshold of different model
USER_INFORMATION	A unique user id and personal data
USER_SYMPTOMS	User symptoms extracted from different models.
USER_TEXTSYMPTOMS	A raw text and preprocessing text.
USER_VACCINATION	A unique user vaccination data.

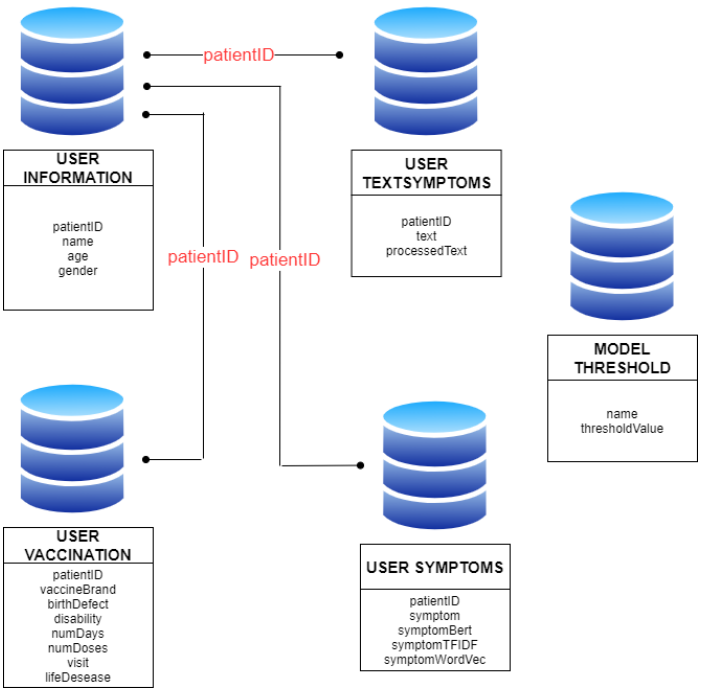


Figure 55: Database graph showing their features and connections.

The ultimate purpose of front-end is to create a smooth user experience while interacting with the knowledge model and application logic (back-end). The system uses Django and react to deliver a web-app user interface that is accessible through any browsers and devices and ease the integration with back-end as Django uses python as the primary programming language. As user interface is not the focus of the project, this section will emphasis on implementation overview and overall design of the front-end web app.

3.3.1. *Implementation overview*

User-Application interaction can be simplified, as shown earlier in Figure ***54***. In description, all the users can interact with the UI in web browser in their devices. By using the HTTP protocol, the web browser will fetch the web resources, such as HTML documents, pictures, JSON data etc. Some of the webpage event services are defined in the client side to send separate GET/POST http request to the Django Framework, which will eventually pass the data to back-end application logic to handle the request. The response is then complied, and web browser will receive the response from the server to render the components or views.

3.3.2. *Webpage connections design*

The web-application contains 6 web pages. The interconnection/linkage between the pages are captured in the graph representation below (Figure 56).

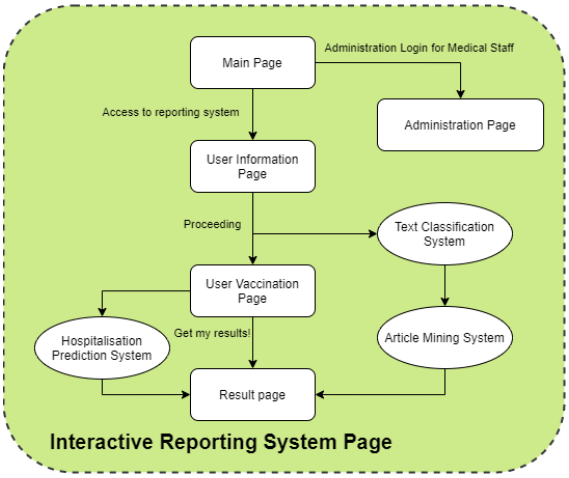


Figure 56: Webpage Linkage Graph.

Each page is designed to deliver the user a better informative visualization. The widgets in the pages are understandable from the user's perspective. There are still many improvements and adaptability on the user interface and the functionalities of the system. However, the system is aimed to deliver the conceptual idea in the simplified webpage to provide user a reporting system to describe the experience after the vaccination and advise them the next step to be taken to prevent any irreversible consequences. The details of the pages are written as below:

- **Main Page**: An entrance for the reporting analyzer system. User can proceed to report a vaccine adverse event or review the patient data and model parameter if it is requested by medical staff.

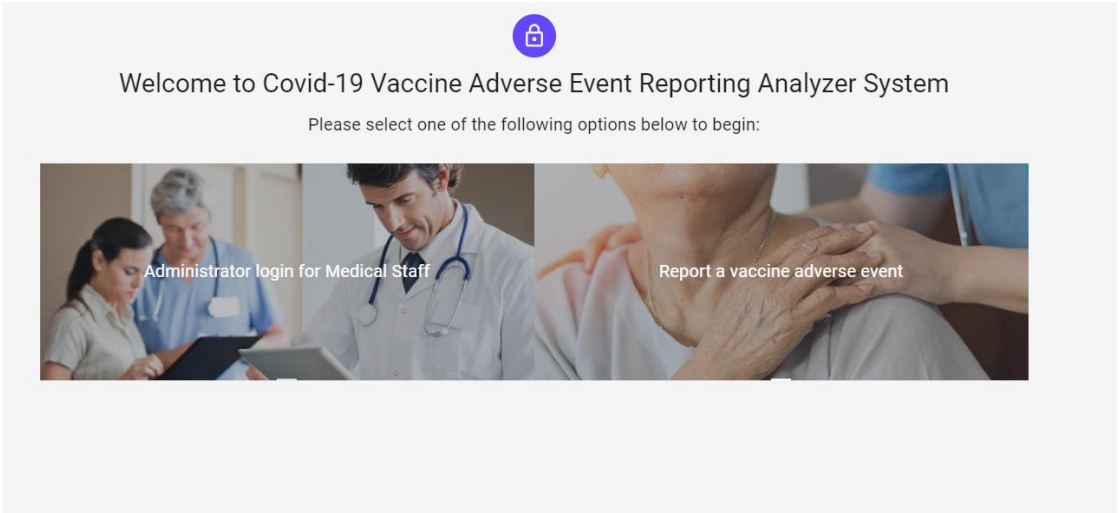


Figure 57: Entrance Page for the Reporting Analyzer System.

- **User Information Page**: A page for registering the user information with the available patient ID and record the description of the symptoms suffered from the patient after vaccination. The registration will be blocked if the user key in a patient ID which have been recorded in the system. The patient's name will be shown accordingly to the user.

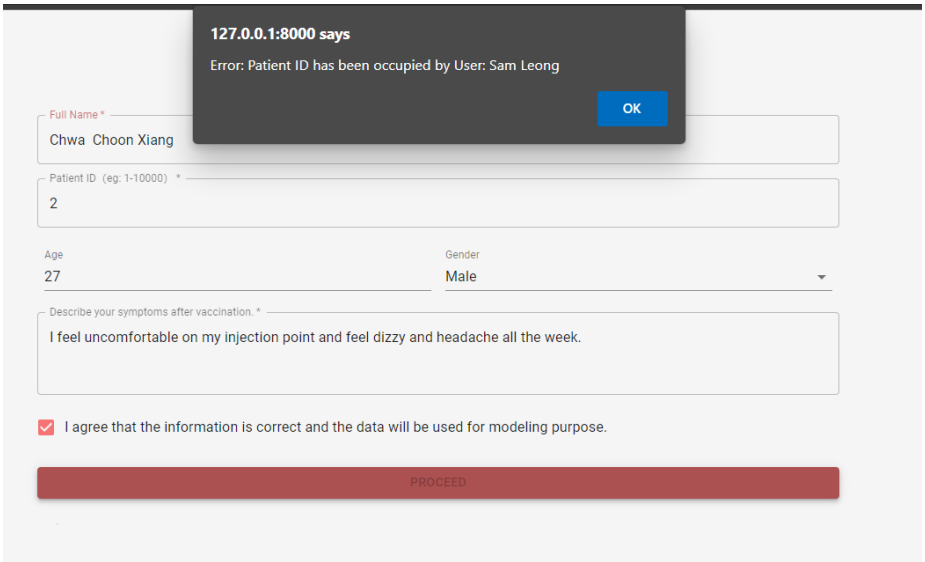


Figure 58: Error prompted if the patient ID has been occupied.

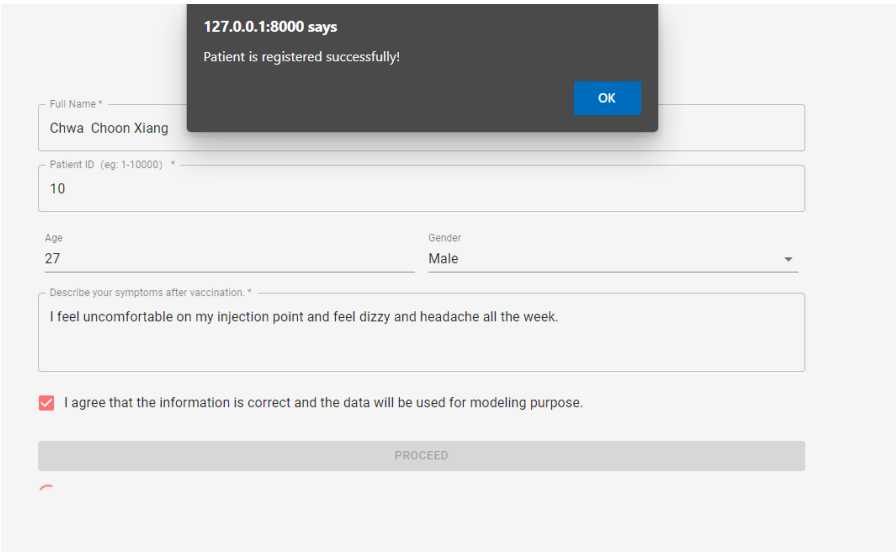


Figure 59: Successful registration of the user data.

- **User Vaccination page:** After the submission of the user information and successful registration of the patient ID with the recorded symptoms, this page will be directed to allow user for input more information related the vaccination such as number of doses taken and number of days after vaccination etc. All the information is compulsory as it will affect the accuracy of the hospitalisation prediction which will be executed sequentially.

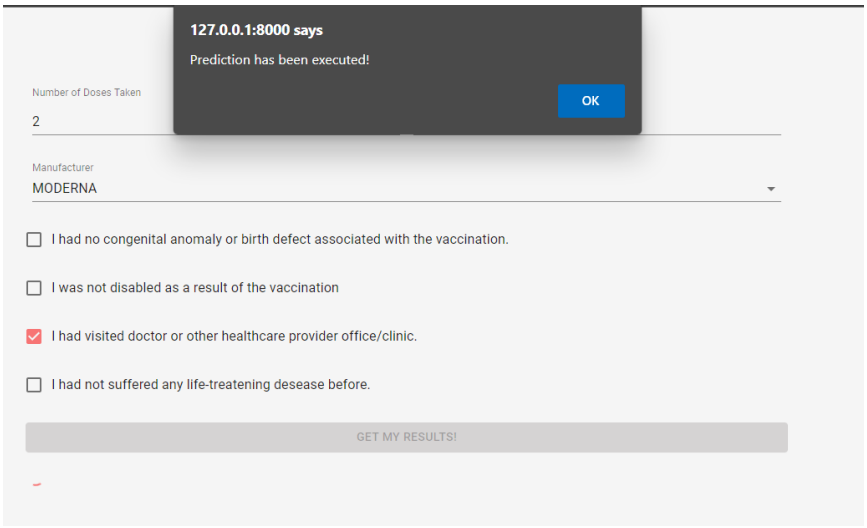


Figure 60: User vaccination page to record compulsory information from the patient.

- **Result page**: A result of the hospitalization rate will be reflected to show the adverse event evaluation outcome. As the threshold of the hospitalization prediction was maintained in the backend, a green slider will indicate the patient who is in normal condition and does not have high risk after vaccination. On the other hand, the patient will be warned with the alert message together with an orange slider. Meanwhile, the Covid-19 vaccine literature which was related to the symptoms suffered by the patient will be populated in the page to show the patient with the strong evidence about the adverse effect of the vaccination.

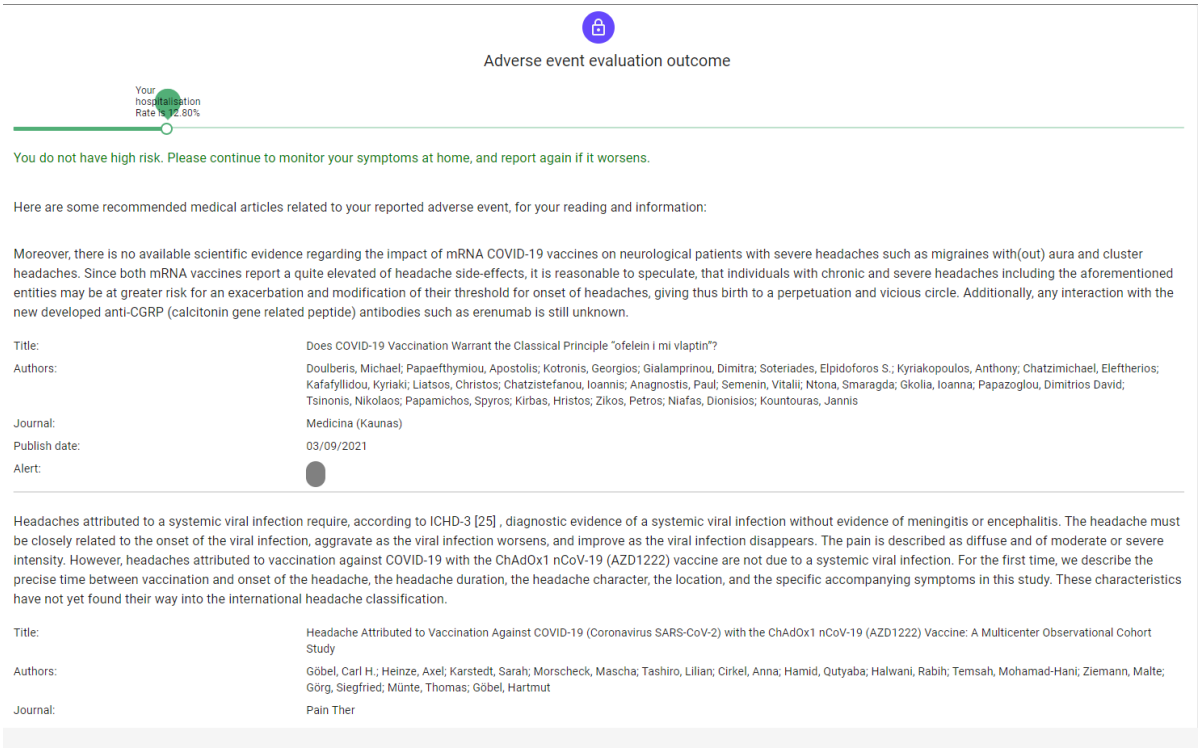


Figure 61: Adverse event evaluation out come with the low hospitalisation prediction outcome.

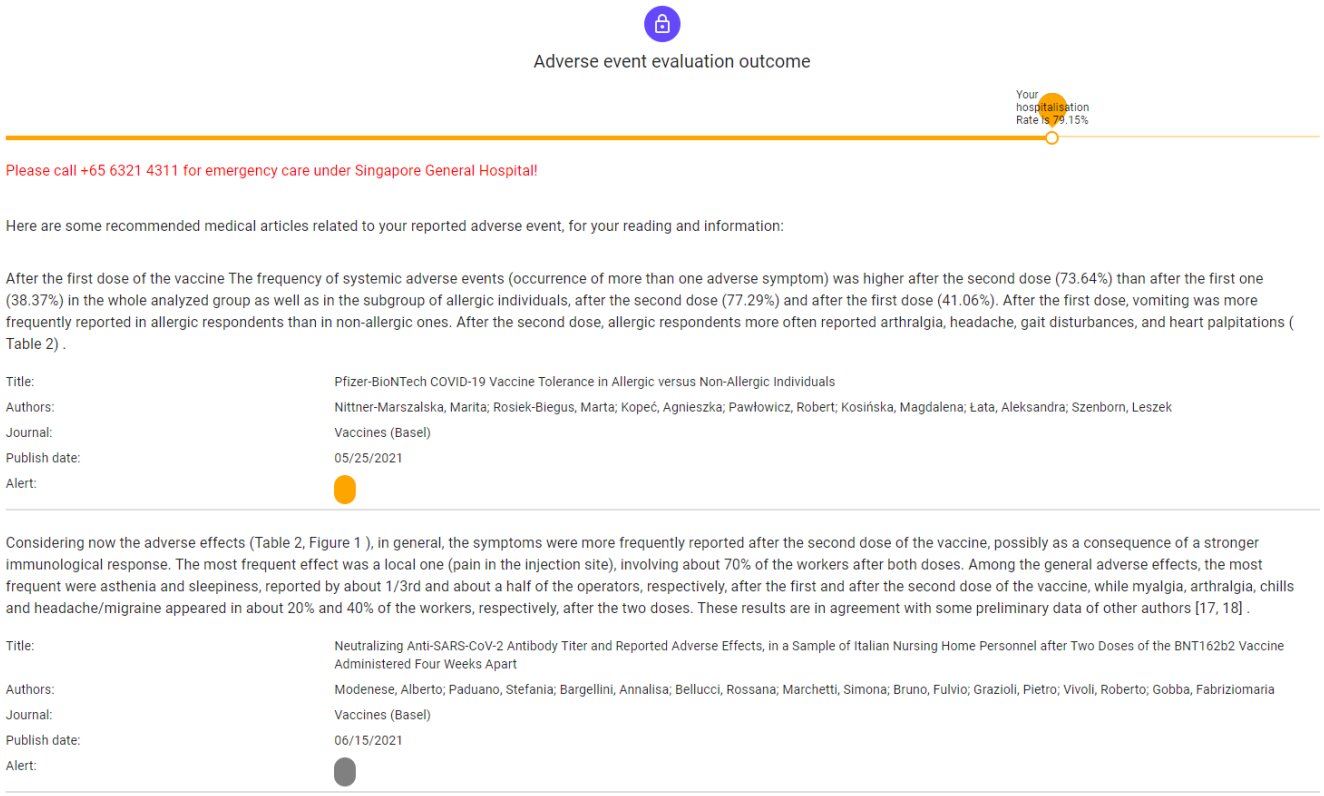


Figure 62: Adverse event evaluation out come with the low hospitalisation prediction outcome.

3.3.3. *Installation Guide*

This section act as an installation guide for the system in Windows environment to ensure minimum installation problem by users. Pip install is used instead of conda install due to some packages require the instalment through pip. Although there is other way to install all the dependencies, we high encourage users to follow the guide below.

1. Download / clone this file into your directory.
2. Navigate into the downloaded/cloned directory. By default, the directory can be found in C:/<username>/Documents/GitHub/PRSPM
3. Run the following command line by line

pip install -r pip_requirements.txt

4. Navigate into *MyWebsite\PRSPM* folder
5. Run the line below to start the program running locally.

python manage.py runserver

6. Go to localhost:8000 from web browser, link: http://127.0.0.1:8000/

4. PROJECT IMPLEMENTATION: A CASE STUDY

Case study:

User/Patient

Lim Chin Shan is a 55-year-old male who has been recently notified to take his Covid vaccination at his nearest community centre. He has registered for Pfizer-BioNTech vaccine 2nd dose, and proceeded to receive his vaccination, although he has heard about negative adverse effects from his friends. After receiving his vaccination, he did not experience any adverse symptoms immediately, and went home afterwards. 6 hours later, he started to feel an increase in body warmth at the injection site and was overall feverish. Worried about his condition, he wanted to seek advice from his nearest hospital and was informed that they have implemented a new adverse event analyser application to streamline the reporting process. He proceeded to use the application to analyse his condition.

Profile of Lim Chin Shan:

- 55 years old, Male
- Vaccine: Pfizer-BioNTech, 2nd dose
- Days after vaccination: <1 day
- Description of adverse event: experienced fever and felt increase in warmth at injection site
- Others: He has not had any life threatening conditions before, nor any major illnesses.

Upon entering the vaccine adverse event analyser application, he registers using the following information above:

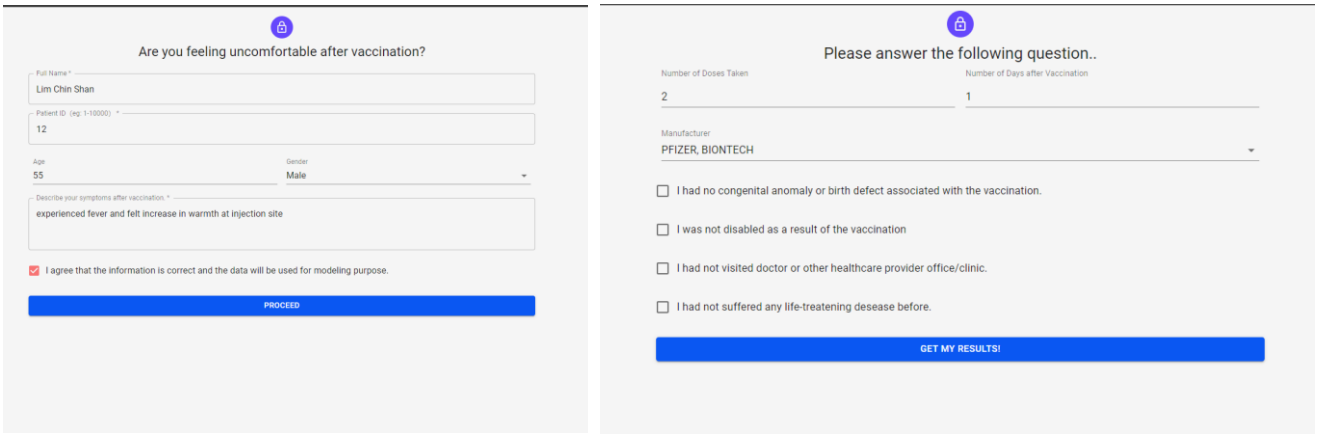


Figure 63: Registration of user and reporting of vaccine adverse event.

After completing the registration and input of reporting data, he is redirected to the hospitalisation prediction page. As seen in Figure 64 below, he was informed that his risk rate was 14.3% and was a low chance of developing into severe condition. He was further advised to rest at home and continue to monitor his situation.

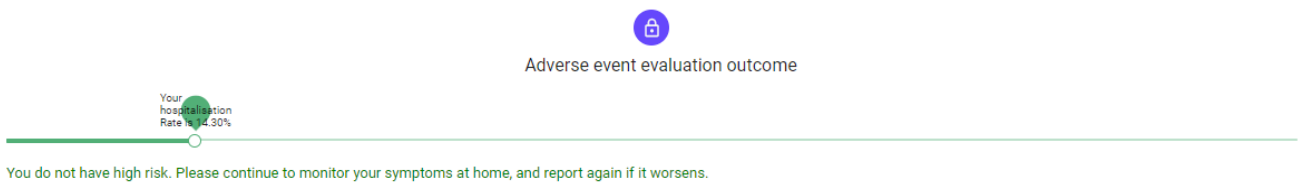


Figure 64: Risk prediction outcome and advise by system.

Still worried by the comments made by his friends on severe adverse events after vaccination, Chin Shan wanted to find out more information by reading the recommended articles provided in the results page (shown in Figure ***65***). For the first article, it was mentioned that fever and warmth at injection site were “minor side effects” and “do not pose a risk for future allergic reactions to the vaccine”. The second article also mentions that the vaccine was “safe and well-tolerated” and “No serious adverse reactions related to the vaccine occurred”. Chin Shan also used the information details of the articles to read the full medical article, and he was reassured that there was low likelihood of severe adverse events for his scenario.

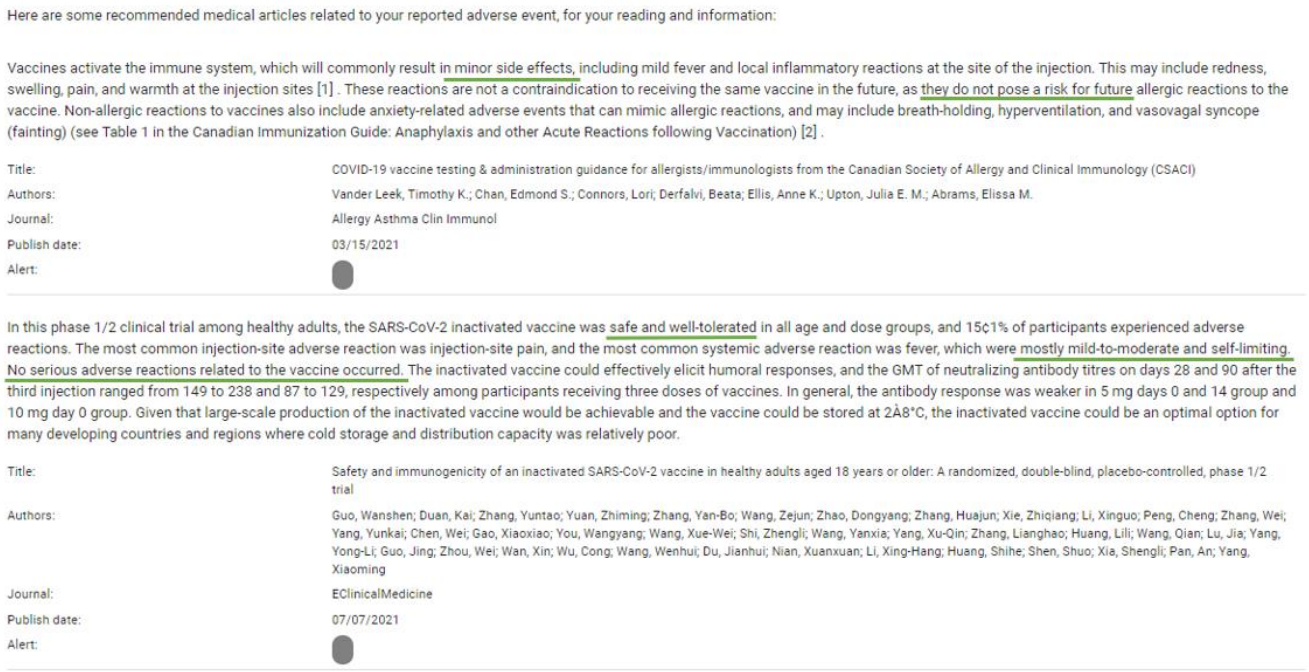


Figure 65: Recommended articles to reported symptoms.

Medical staff

Due to the high volume of Covid-19 cases recently, the hospital has been facing shortage of medical facilities and overloaded medical staff. They are unable to cope with the number of patients and would have to prioritise those with higher risk of severe conditions. They have made the decision to reduce the number of hospital admission from adverse vaccination, in order to treat Covid-19 cases. To do so, they utilised the application by entering the administration page and toggled the threshold for prediction of hospitalisation.

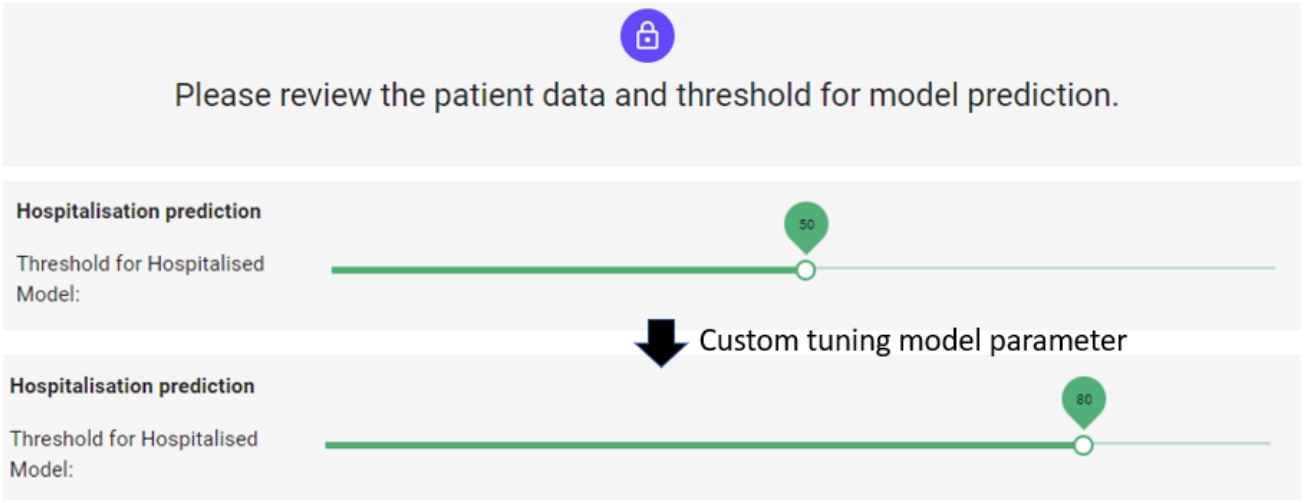


Figure 66: Custom tuning of model parameters for business requirements.

For a user that utilises the application, in the default scenario he would be advised to be admitted to hospital as there are available treatment facilities (Figure 67 top). With the custom tuned model parameters, he is instead advised to rest at home and monitor his condition (Figure 67 bottom). If his condition worsens, he would have a higher prediction score, which will surpass the new threshold and be advised for medical treatment at the hospital. On the other hand, if his condition becomes better, the hospital would have managed to allocate resources more effectively to other patients in need.

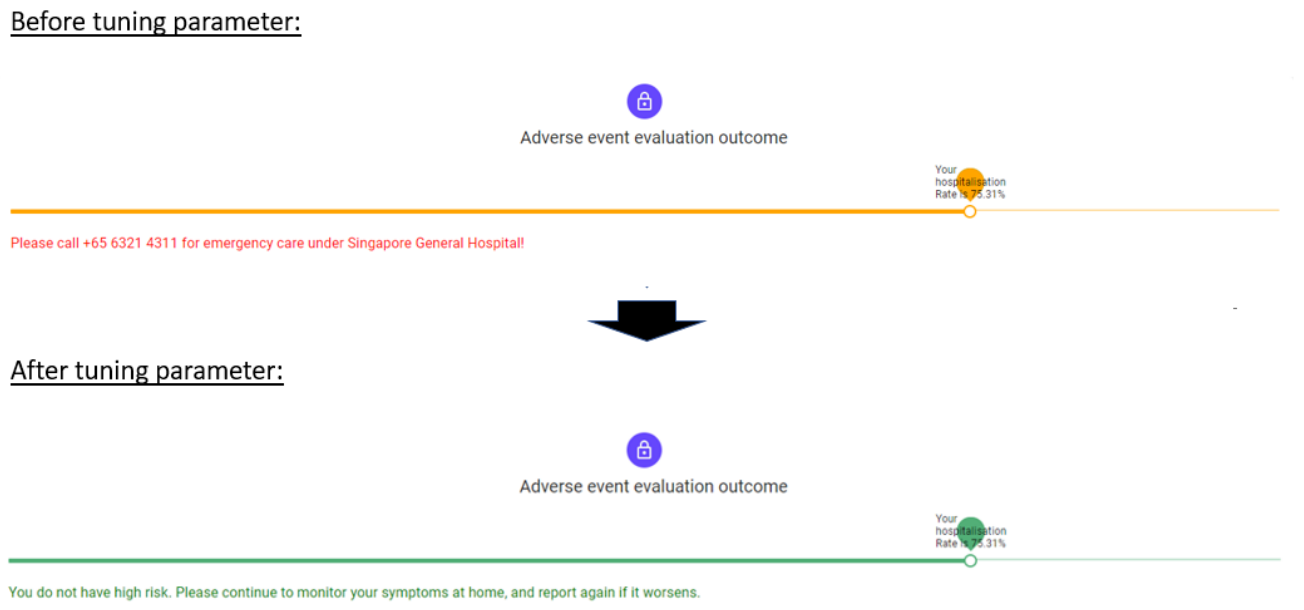


Figure 67: Model prediction before tuning (top) and after tuning (bottom) parameter.

These examples showcase the use case scenarios and potential benefits of the application to both users and medical staffs. It helps to automate the decision-making process and also provides flexibility to meet changing business needs.

5. CHALLENGES & FUTURE IMPROVEMENTS

Through the development of our solution, we encountered some challenges & limitations and would like to discuss possible future improvements:

- The training of the text classification is normally intensive and become time consuming when tuning the hyperparameters of the model. Due to limited resources and cost, we have spent more than hundred hours to search and tune for the best parameters to be used in the model. Multi-label classification could be simplified by training each class separately and the combination of the different model with different symptom classification should performance better ideally. However, the method is not practical enough to complete the task of the text classification in single environment and increase the complexity while integrating the backend system. The only improvement is to have better planning on the resources and machine to train with various models. To-Do tasks should be recorded as always so that we can learn from a failure to improve better in the next fine-tuning of the parameter of the model.
- Sparsity of symptom data is a common issue faced by the models for text classification as there are too many symptom types listed in the VAERS SYMPTOMS dataset, even though we have performed rounds of down-sampling we still have the issue of class imbalance where there are more true negatives. A possible future improvement is to perform clustering to identify potential groups of symptoms due to underlying relationships in the data and cross-refer them with a medical expert to ensure that the groupings make sense in reality. Doing so can reduce the dimensionality of the symptom data and improve model training and inference results.
- In Hospitalization prediction, some data are not separable as same input will have different predicted output. There could be other factor that were not capture in the input which also affect whether an individual will be hospitalized. Example of other factors that might affect hospitalization rate are medication history. In future work, liaising with government and health care provider to provide medical history for the individuals might provide better separation between the hospitalized and non-hospitalization prediction.
- To achieve good outcomes from literature mining and question-answering models, training of deep networks on very large datasets (e.g. Wikipedia, Bing, PubMed) are required. Furthermore, training on domain-specific datasets will improve prediction accuracy, as it encompasses specific terminologies and contextual meaning. However, due to limited computational resources, we would have to conduct fine-tuning on training corpus available online. For future developments, availability of large-size domain-specific datasets and computational resources to train on them, would benefit model predictions.

6. CONCLUSION

In this project, we have developed an intelligent system for predictive analytics of vaccine adverse event through machine learning techniques. It encompasses an exploration of different models, such as SVM, MLP, DL, transfer learning, vectorisation, QA search and retrieval. These models have shown to be effective on a variety of tasks within the system: multi-label text classification, prediction of hospitalization rate, and literature mining of article corpus. As a whole, the various machine learning modules are integrated as an end-to-end solution where patient data is extracted and analysed at each component. The output of the system provides users with indication on the severity of the vaccine adverse event and further actions to be taken. Automation of the process helps to reduce workload of medical staff to manage the large volume of data and decision-making. To meet the volatile and changing business requirements of hospitals, such as availability of medical resources and facilities, medical staff are also provided with the ability to custom tune prediction parameters accordingly. Lastly, to reduce misinformation and hesitancy towards vaccination, relevant medical articles are provided to users for their further understanding of vaccination.

Our team would like the Covid-19 Vaccine Adverse Effect Analyser System to serve as a demonstrator of machine learning application to improve efficiency of real-world processes and to tackle pressing issues in society. For future development and implementation to potential customers, further collection and integration of additional data could be conducted to improve the model prediction on new vaccines or boosters, which could be introduced for emerging variants of coronavirus.

7. BIBLIOGRAPHY

[1] Johns Hopkins University of Medicine, "Johns Hopkins Coronavrus Resource Center," [Online]. Available: https://coronavirus.jhu.edu/. [Accessed 14 November 2021].

[2] Centers for Disease Control and Prevention, "Covid-19 Symptoms," [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html. [Accessed 14 November 2021].

[3] Centers for Disease Control and Prevention, "Pfizer-BioNTech COVID-19 Vaccine Reactions & Adverse Events," [Online]. Available: https://www.cdc.gov/vaccines/covid-19/info-by-product/pfizer/reactogenicity.html. [Accessed 14 November 2021].

[4] J. Hippisley-Cox, M. Patone, X. W. Mei, D. Saatci, S. Dixon, K. Khunti, F. Zaccardi, P. Watkinson, M. Shankar-Hari, J. Doidge, D. A. Harrison, S. J. Griffin and A. Sheikh, "Risk of thrombocytopenia and thromboembolism after covid-19 vaccination and SARS-CoV-2 positive testing: self-controlled case series study," *BMJ,* vol. 374, 2021.

[5] M. Franchini, G. M. Liumbruno and M. Pezzo, "COVID-19 vaccine-associated immune thrombosis and thrombocytopenia (VITT): Diagnostic and therapeutic recommendations for a new syndrome," *European Journl of Haematology,* vol. 107, no. 2, pp. 173-180, 2021.

[6] S. Choi, S. Lee, J.-W. Seo, M.-J. Kim, Y. H. Jeon, J. H. Park, J. K. Lee and N. S. Yeo, "Myocarditis-induced Sudden Death after BNT162b2 mRNA COVID-19 Vaccination in Korea: Case Report Focusing on Histopathological Findings," *Journal of Korean Medical Science,* vol. 36, no. 40, p. 286, 2021.

[7] B. Bokurtz, I. Kamat and P. J. Hotez, "Myocarditis With COVID-19 mRNA Vaccines," *Circulation,* vol. 144, no. 6, 2021.

[8] S. Dutta, R. J. Kaur, J. Charan, P. Bhardwaj, P. Sharma, S. Ambwani, M. Haque, A. Tandon, J. P. Abhayanand, S. Sukhija, S. V. Suman and S. Misra, "Serious adverse events reported from the COVID-19 vaccines: A descriptive study based on WHO database," *medRxiv,* 2021.

[9] A. P. Desai, A. P. Desai and G. J. Loomis, "Relationship between pre-existing allergies and anaphylactic reactions post mRNA COVID-19 vaccine administration," *Vaccine,* vol. 39, no. 32, pp. 4407-4409, 2021.

[10] R. J. Kaur, S. Dutta, P. Bhardwaj, J. Charan, S. Dhingra, P. Mitra, K. Singh, D. Yadav, P. Sharma and S. Misra, "Adverse Events Reported From COVID-19 Vaccine Trials: A Systematic Review," *Indian Journal of Clinical Biochemistry,* vol. 36, no. 4, pp. 427-439, 2021.

[11] N. Barda, N. Dagan, Y. Ben-Shlomo, E. Kepten, J. Waxman, R. Ohana, M. A. Hernan, M. Lipsitch, I. Kohane, D. Netzer, B. Y. Reis and R. D. Balicer, "Safety of the BNT162b2 mRNA Covid-19 Vaccine in a Nationwide Setting," *New England Journal of Medicine,* vol. 385, pp. 1078-1090, 2021.

[12] T. Botsis, M. D. Nguyen, E. J. Woo, M. Markatou and R. Ball, "Text mining for the Vaccine Adverse Event Reporting System: medical text classification using informative feature selection," *Journal of the American Medical Informatics Association,* vol. 18, no. 5, pp. 631-638, 2011.

[13] G. Arora, J. Joshi, R. S. Mandal, N. Shrivastava, R. Virmani and T. Sethi, "Artificial Intelligence in Surveillance, Diagnosis, Drug Discovery and Vaccine Development against COVID-19," *Pathogens,* vol. 10, p. 1048, 2021.

[14] H. Lv, L. Shi, J. W. Berkenpas, F.-Y. Dao, H. Zulfiqar, H. Ding, Y. Zhang, L. Yang and R. Cao, "Application of artificial intelligence and machine learning for COVID-19 drug discovery and vaccine design," *Briefings in Bioinformatics,* vol. 22, no. 6, 2021.

[15] M. Prachar, S. Justesen, D. B. Steen-Jensen, S. Thorgrimsen, E. Jurgons, O. Winther and F. O. Bagger, "Identification and validation of 174 COVID-19 vaccine candidate epitopes reveals low performance of common epitope prediction tools," *Scientific Reports,* vol. 10, p. 20465, 2020.

[16] E. Ong, M. U. Wong, A. Huffman and Y. He, "COVID-19 Coronavirus Vaccine Design Using Reverse Vaccinology and Machine Learning," *Frontiers in Immunology,* vol. 11, p. 1581, 2020.

[17] P. Gonzalez-Dias, E. K. Lee, S. Sorgi, D. S. de Lima, A. H. Urbanski, E. L. Silveira and H. I. Nakaya, "Methods for predicting vaccine immunogenicity and reactogenicity," *Human Vaccines & Immunotherapeutics,* vol. 16, no. 2, pp. 269-276, 2020.

[18] M. M. Ahamad, S. Aktar, M. J. Uddin, M. Rahsed-Al-Mahfuz, A. Azad, S. Uddin, S. A. Alyami, I. H. Sarker, P. Lio, J. M. Quinn and M. A. Moni, "Adverse effects of COVID-19 vaccination: machine learning and statistical approach to identify and classify incidences of morbidity and post-vaccination reactogenicity," *medRxiv,* 2021.

[19] Y. Kim, J.-H. Jang, N. Park, N.-Y. Jeong, E. Lim, S. Kim, N.-K. Choi and D. Yoon, "Machine Learning Approach for Active Vaccine Safety Monitoring," *Journal of Korean Medical Science,* vol. 36, no. 31, p. 198, 2021.

[20] "Vaccine Adverse Event Reporting System," [Online]. Available: https://vaers.hhs.gov/. [Accessed 14 November 2021].

[21] "VAERS Data Sets," [Online]. Available: https://vaers.hhs.gov/data/datasets.html. [Accessed 14 November 2021].

[22] L. L. Wang, K. Lo, Y. Chandrasekhar, R. Reas, J. Yang, D. Eide, K. Funk, R. Kinney, Z. Liu, W. Merrill, P. Mooney, D. Murdick, D. RIshi, J. Sheehan, Z. Shen, B. Stilson, A. D. Wade, K. Wang, C. Wilhelm, B. XIe, D. Raymond, D. S. Weld, O. Etzioni and S. Kohlmeier, "CORD-19: The Covid-19 Open Research Dataset," *ArXiv,* 2020.

[23] "CORD-19," [Online]. Available: https://www.semanticscholar.org/cord19. [Accessed 14 November 2021].

[24] L. Chen, "Support Vecotr Machine - Simply Explained," [Online]. Available: https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496. [Accessed 14 November 2021].

[25] R. Pupale, "Support Vector Machines (SVM) - An Overview," [Online]. Available: https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989. [Accessed 14 November 2021].

[26] N. B. Aylien, "Support Vector Machines: A Simple Explanation," [Online]. Available: https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html. [Accessed 14 November 2021].

[27] T. Mikolov, K. Chen, G. Corrado and J. Dean, “Efficient Estimation of Word Representations in Vecotr Space,” *arXiv,* 2013.

[28] J. Alammar, “The Illustrated Transformer,” [Online]. Available: https://jalammar.github.io/illustrated-transformer/. [Accessed 14 November 2021].

[29] R. Kumar, “All You Need to know about BERT,” 27 May 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/05/all-you-need-to-know-about-bert/. [Accessed 14 November 2021].

[30] B. Rocca, “Handling imbalanced datasets in machine learning,” 28 January 2019. [Online]. Available: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28. [Accessed 14 November 2021].

[31] A. Esteva, A. Kale, R. Paulus, K. Hashimoto, W. Yin, D. Redev and R. Socher, “CO-Search: COVID-19 Information Retrieval with Semantic Search, Question Answering, and Abstractive Summarization,” *arXiv,* 2020.

[32] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,” *arXiv,* 2019.

[33] N. Reimers and I. Gurevych, “The Curse of Dense Low-Dimensional Information Retrieval for Large Index Sizes,” *arXiv,* 2021.

[34] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So and J. Kang, “BioBERT: a pre-trained biomedical language representation model for biomedical text mining,” *arXiv,* 2019.

[35] P. Rajpurkar, R. Jia and P. Liang, “Know What You Don't Know: Unanswerable Questions for SQuAD,” *arXiv,* 2018.

[36] C. Hutto and E. GIlbert, “VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text,” *Eight International AAAI Conference on Weblogs and Social Media,* vol. 8, no. 1, 2014.

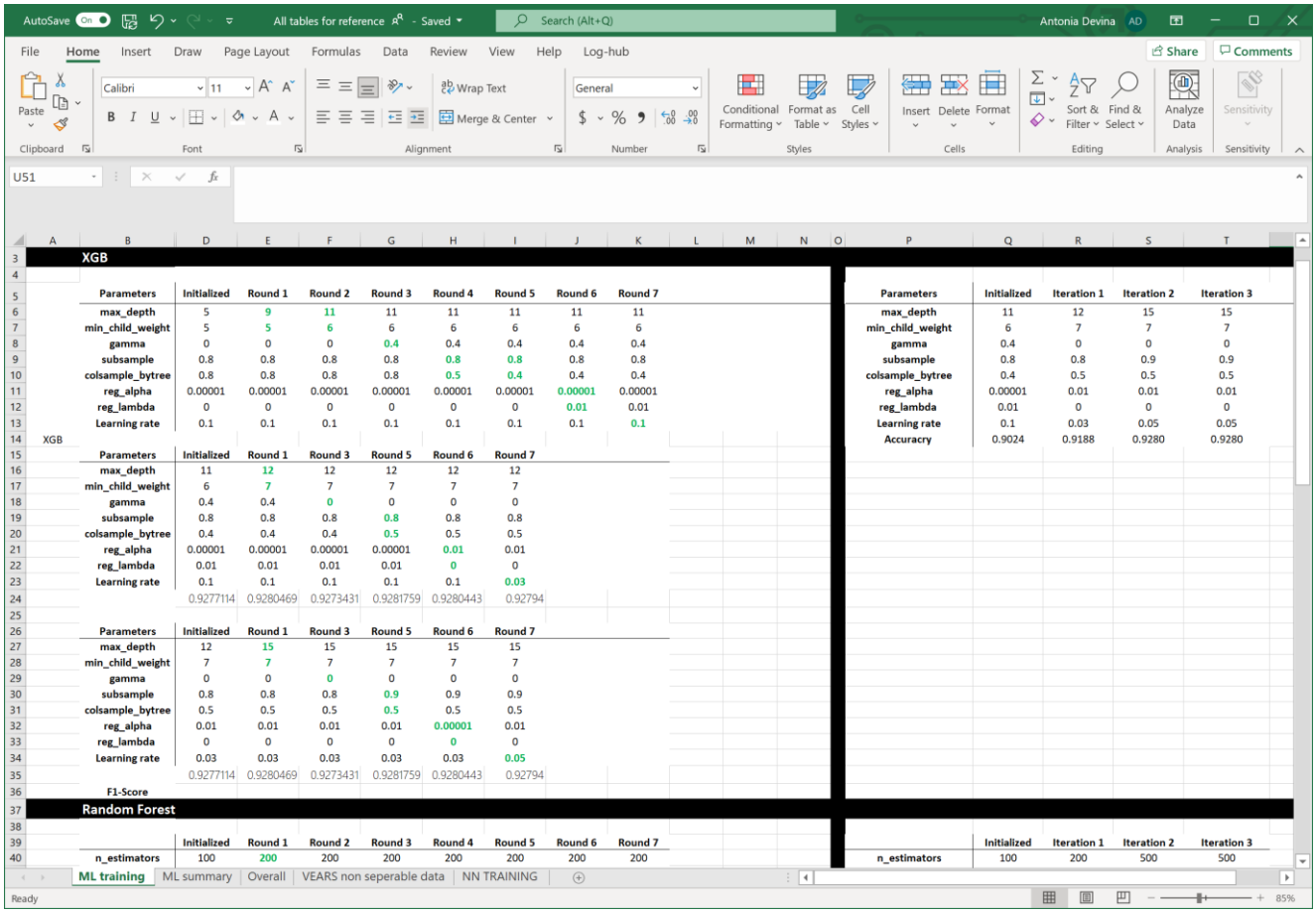
[37] Pharmaceutical Technology, “Covid-19 vaccines: serious adverse events explained,” [Online]. Available: https://www.pharmaceutical-technology.com/features/covid-19-vaccines-adverse-events-explained/. [Accessed 14 November 2021].

8. APPENDIX

8.1. Appendix A: ML and NN Attachments

All the training datasets and parameter optimization can be found under the ***Code\HospitalisationPrediction\ All tables for reference.xlsx***.

The first tab contains all the iteration run for ML hyper parameter tuning and overall summary for each ML model and iteration.



XGB

Parameters	Initialized	Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7
max_depth	5	9	11	11	11	11	11	11
min_child_weight	5	5	6	6	6	6	6	6
gamma	0	0	0	0.4	0.4	0.4	0.4	0.4
subsample	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
colsample_bytree	0.8	0.8	0.8	0.8	0.5	0.4	0.4	0.4
reg_alpha	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001
reg_lambda	0	0	0	0	0	0	0.01	0.01
Learning rate	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Parameters	Initialized	Iteration 1	Iteration 2	Iteration 3
max_depth	11	12	15	15
min_child_weight	6	7	7	7
gamma	0.4	0	0	0
subsample	0.8	0.8	0.9	0.9
colsample_bytree	0.4	0.5	0.5	0.5
reg_alpha	0.00001	0.01	0.01	0.01
reg_lambda	0.01	0	0	0
Learning rate	0.1	0.03	0.05	0.05
Accuracy	0.9024	0.9188	0.9280	0.9280

XGB

Parameters	Initialized	Round 1	Round 3	Round 5	Round 6	Round 7
max_depth	11	12	12	12	12	12
min_child_weight	6	7	7	7	7	7
gamma	0.4	0.4	0	0	0	0
subsample	0.8	0.8	0.8	0.8	0.8	0.8
colsample_bytree	0.4	0.4	0.4	0.5	0.5	0.5
reg_alpha	0.00001	0.00001	0.00001	0.00001	0.01	0.01
reg_lambda	0.01	0.01	0.01	0.01	0	0
Learning rate	0.1	0.1	0.1	0.1	0.1	0.03
	0.9277114	0.9280469	0.9273431	0.9281759	0.9280443	0.92794

Parameters	Initialized	Round 1	Round 3	Round 5	Round 6	Round 7
max_depth	12	15	15	15	15	15
min_child_weight	7	7	7	7	7	7
gamma	0	0	0	0	0	0
subsample	0.8	0.8	0.8	0.9	0.9	0.9
colsample_bytree	0.5	0.5	0.5	0.5	0.5	0.5
reg_alpha	0.01	0.01	0.01	0.01	0.00001	0.01
reg_lambda	0	0	0	0	0	0
Learning rate	0.03	0.03	0.03	0.03	0.03	0.05
	0.9277114	0.9280469	0.9273431	0.9281759	0.9280443	0.92794

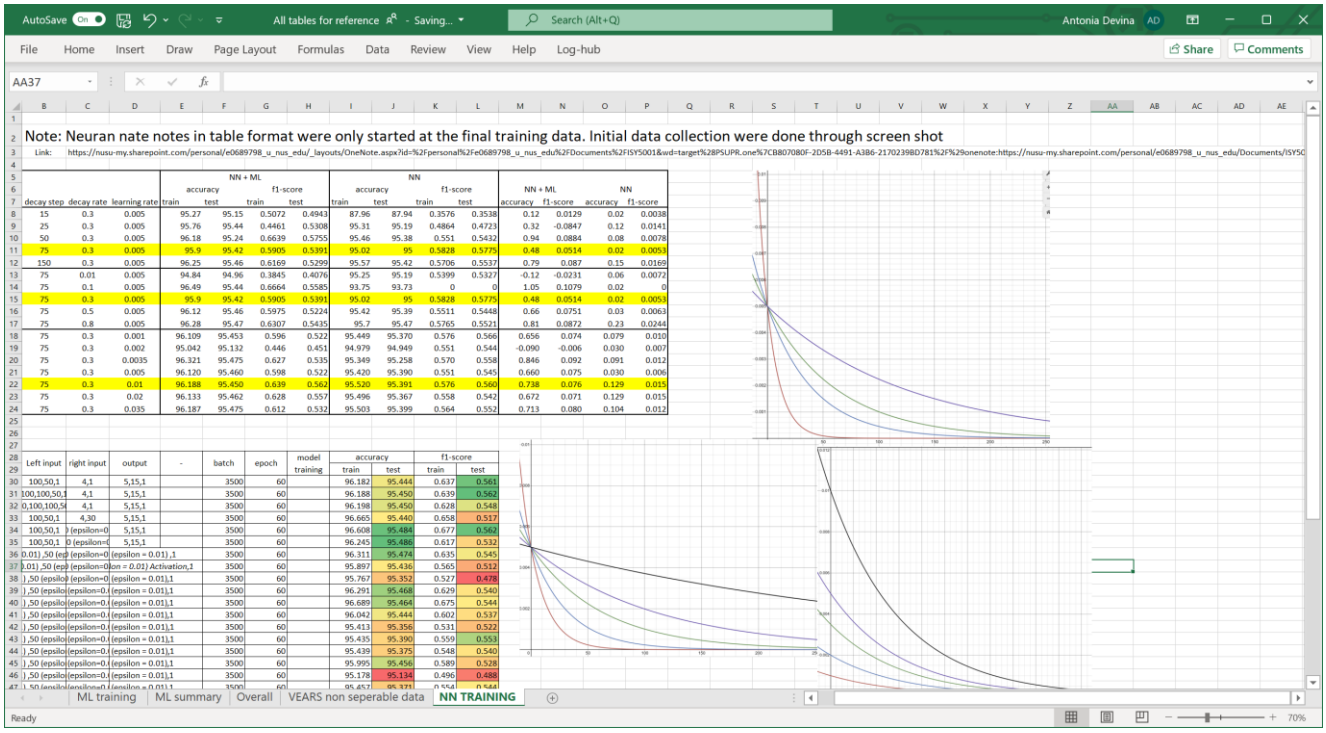
F1-Score

Random Forest

	Initialized	Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7
n_estimators	100	200	200	200	200	200	200	200

	Initialized	Iteration 1	Iteration 2	Iteration 3
n_estimators	100	200	500	500

The last sheet contains the NN training summary. However, it only contains the latest NN optimization as the initial optimization were recorded through screen shot. The initial data can be found in the link provided in the excel sheet.



Note: Neuran nate notes in table format were only started at the final training data. Initial data collection were done through screen shot

Link: https://nusu-my.sharepoint.com/personal/e0689798_u_nus_edu/_layouts/OneNote.aspx?id=%2Fpersonal%2Fe0689798_u_nus_edu%2FDocuments%2FISY5001&wd=target%28PSUPR.one%7CB807080F-2D5B-4491-A3B6-2170239BD781%2F%29onenote:https://nusu-my.sharepoint.com/personal/e0689798_u_nus_edu/Documents/ISY5

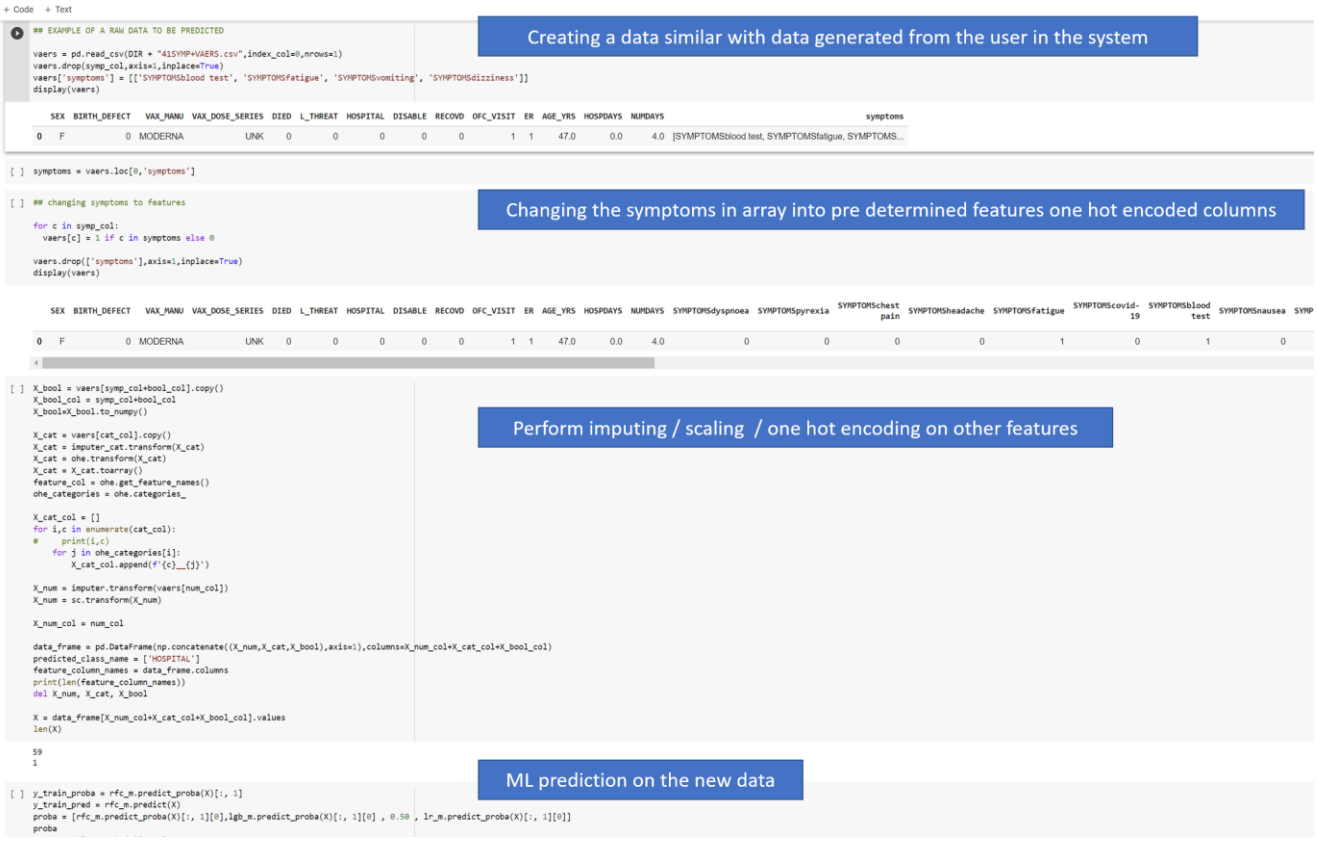
			NN + ML				NN				NN + ML		NN	
			accuracy		f1-score		accuracy		f1-score					
decay step	decay rate	learning rate	train	test	train	test	train	test	train	test	accuracy	f1-score	accuracy	f1-score
15	0.3	0.005	95.27	95.15	0.5072	0.4943	87.96	87.94	0.3576	0.3538	0.12	0.0129	0.02	0.0038
25	0.3	0.005	95.76	95.44	0.4461	0.5308	95.31	95.19	0.4864	0.4723	0.32	-0.0847	0.12	0.0141
50	0.3	0.005	96.18	95.24	0.6639	0.5755	95.46	95.38	0.551	0.5432	0.94	0.0884	0.08	0.0078
75	0.3	0.005	95.9	95.42	0.5905	0.5391	95.02	95	0.5828	0.5775	0.48	0.0514	0.02	0.0053
150	0.3	0.005	96.25	95.46	0.6169	0.5299	95.57	95.42	0.5706	0.5537	0.79	0.087	0.15	0.0169
75	0.01	0.005	94.84	94.96	0.3845	0.4076	95.25	95.19	0.5399	0.5327	-0.12	-0.0231	0.06	0.0072
75	0.1	0.005	96.49	95.44	0.6664	0.5585	93.75	93.73	0	0	1.05	0.1079	0.02	0
75	0.3	0.005	95.9	95.42	0.5905	0.5391	95.02	95	0.5828	0.5775	0.48	0.0514	0.02	0.0053
75	0.5	0.005	96.12	95.46	0.5975	0.5224	95.42	95.39	0.5511	0.5448	0.66	0.0751	0.03	0.0063
75	0.8	0.005	96.28	95.47	0.6307	0.5435	95.7	95.47	0.5765	0.5521	0.81	0.0872	0.23	0.0244
75	0.3	0.001	96.109	95.453	0.596	0.522	95.449	95.370	0.576	0.566	0.656	0.074	0.079	0.010
75	0.3	0.002	95.042	95.132	0.446	0.451	94.979	94.949	0.551	0.544	-0.090	-0.006	0.030	0.007
75	0.3	0.0035	96.321	95.475	0.627	0.535	95.349	95.258	0.570	0.558	0.846	0.092	0.091	0.012
75	0.3	0.005	96.120	95.460	0.598	0.522	95.420	95.390	0.551	0.545	0.660	0.075	0.030	0.006
75	0.3	0.01	96.188	95.450	0.639	0.562	95.520	95.391	0.576	0.560	0.738	0.076	0.129	0.015
75	0.3	0.02	96.133	95.462	0.628	0.557	95.496	95.367	0.558	0.542	0.672	0.071	0.129	0.015
75	0.3	0.035	96.187	95.475	0.612	0.532	95.503	95.399	0.564	0.552	0.713	0.080	0.104	0.012

Left input	right input	output	-	batch	epoch	model training	accuracy		f1-score	
							train	test	train	test
100,50,1	4,1	5,15,1		3500	60		96.182	95.444	0.637	0.561
100,100,50,1	4,1	5,15,1		3500	60		96.188	95.450	0.639	0.562
0,100,100,50	4,1	5,15,1		3500	60		96.198	95.450	0.628	0.548
100,50,1	4,30	5,15,1		3500	60		96.665	95.440	0.658	0.517
100,50,1	) (epsilon=0	5,15,1		3500	60		96.608	95.484	0.677	0.562
100,50,1	0 (epsilon=0	5,15,1		3500	60		96.245	95.486	0.617	0.532
0.01) ,50 (ep	(epsilon=0	(epsilon = 0.01) ,1		3500	60		96.311	95.474	0.635	0.545
0.01) ,50 (ep	(epsilon=0	on = 0.01) Activation,1		3500	60		95.897	95.436	0.565	0.512
),50 (epsilo	(epsilon=0	(epsilon = 0.01),1		3500	60		95.767	95.352	0.527	0.478
),50 (epsilo	(epsilon=0.(	(epsilon = 0.01),1		3500	60		96.291	95.468	0.629	0.540
),50 (epsilo	(epsilon=0.(	(epsilon = 0.01),1		3500	60		96.689	95.464	0.675	0.544
),50 (epsilo	(epsilon=0.(	(epsilon = 0.01),1		3500	60		96.042	95.444	0.602	0.537
),50 (epsilo	(epsilon=0.(	(epsilon = 0.01),1		3500	60		95.413	95.356	0.531	0.522
),50 (epsilo	(epsilon=0.(	(epsilon = 0.01),1		3500	60		95.435	95.390	0.559	0.553
),50 (epsilo	(epsilon=0.(	(epsilon = 0.01),1		3500	60		95.439	95.375	0.548	0.540
),50 (epsilo	(epsilon=0.(	(epsilon = 0.01),1		3500	60		95.995	95.456	0.589	0.528
),50 (epsilo	(epsilon=0.(	(epsilon = 0.01),1		3500	60		95.178	95.134	0.496	0.488

ML training | ML summary | Overall | VEARS non seperable data | NN TRAINING

Ready

Example code for how the system run can be found in this directory: ***Code\HospitalisationPrediction\ HOSP_EXAMPLE.ipny***



+ Code + Text

Creating the model skeleton

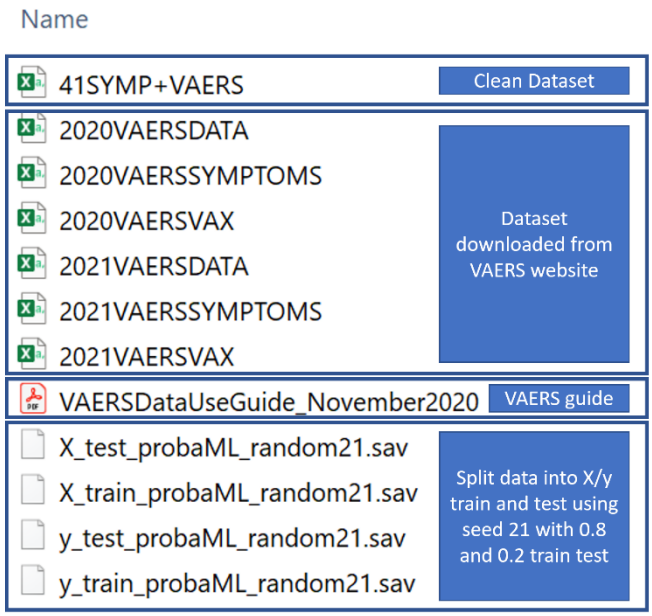
```
def createModel():

    Lin      = Input(shape=(59), name = 'leftIn')
    Lx       = Dense(250,activation='relu')(Lin)
    Lx       = BatchNormalization(momentum=0.99, epsilon=0.001, center=True, scale=True)(Lx)
    Lx       = Dense(100,kernel_regularizer=regularizers.l2(0.01))(Lx)
    Lx       = BatchNormalization(momentum=0.99, epsilon=0.001, center=True, scale=True)(Lx)
    Lx       = Activation('relu')(Lx)
    Lx       = Dense(1,activation='relu',kernel_regularizer=regularizers.l2(0.15))(Lx)

    Rin      = Input(shape=(4), name = 'RightIn')
    Rx       = Dense(30,activation='relu',kernel_regularizer=regularizers.l2(0.1))(Rin)
    Rx       = BatchNormalization(momentum=0.99, epsilon=0.01, center=True, scale=True)(Rx)
    Rx       = Dense(1,activation='relu',kernel_regularizer=regularizers.l2(0.1))(Rx)

    x       = concatenate([Lx,Rx], axis=-1)
    x       = Dense(30,activation='relu',kernel_regularizer=regularizers.l2(0.25))(x)
    x       = BatchNormalization(momentum=0.99, epsilon=0.01, center=True, scale=True)(x)
    x       = Dense(1,activation='sigmoid',kernel_regularizer=regularizers.l2(0.5))(x)

    model     = Model(inputs=[Lin,Rin],outputs=x)
    model.compile(loss=tf.keras.losses.BinaryCrossentropy(),
                  optimizer=optmz,
                  metrics=[tf.keras.metrics.BinaryAccuracy(),tf.keras.metrics.BinaryCrossentropy(),
                           tf.keras.metrics.FalsePositives(),tf.keras.metrics.FalseNegatives()])

    return model

                                                                    # Step 4
modelGo       = createModel() # This is meant for training

folderpath      = DIR
filepath        = folderpath + modelname + ".hdf5"

modelGo.load_weights(filepath)
modelGo.compile(optimizer=optmz,
          loss=tf.keras.losses.BinaryCrossentropy(),
          metrics=tf.keras.metrics.BinaryCrossentropy())


predicts_train    = modelGo.predict([np.array(X).reshape(1,-1),np.array(proba).reshape(1,-1)])
labelname   = ['Not Hosp','Hospitalized']

predout_train = predicts_train.copy()
print(predicts_train)
predout_train[predicts_train <= 0.5] = 0
predout_train[predicts_train > 0.5] = 1

print(predout_train)
```

Predicting hospitalization occurrence on new input

```
[[0.12233526]]
[[0.]]
```

8.1.1.1. VAERS Raw Data

Raw data training set can be found in this directory. Clean dataset is saved to minimized the time taken for data loading and eliminate the data cleaning processing and time needed. Directory extension for the raw data used is: **\Code\HospitalisationPrediction\Training Dataset**



8.1.1.2. Pre-processed models

When making a prediction, the pretrained model that is saved through pickle is loaded and the new data hospitalization occurrence are predicted. The pre-learn models can be found in the directory below: **\Code\HospitalisationPrediction\PreLearn Models**