EUROPEAN UNIVERSITY OF LEFKE

FACULTY OF ENGINEERING

Graduation Project II

Sentiment Analysis and Text Classification using Mlpack C++

JOHN KELECHUKWU OBI

20140237

Sentiment analysis of textual data used to assess products in discussion forums and e-commerce settings is the main objective of this web-based platform project. It makes an effort to perform a thorough text classification. It will be used to help arrange text based on sentimental significance in order to quickly support the clients who are providing feedback. Finding the text's emotions, whether favourable or negative, is the goal.

Supervisor

Asst. Prof. Dr. Cem Kalyoncu

Publish Date

21/12/2023

Table of Contents

Table of figures

1.Introduction

1.1 Problem definition

Text as we know convey a lot of emotion or sentiments and when coupled with a system that receives thousands of texts in a small time span it becomes difficult to address the customers or users feedback. The aim is to address the challenge of analysing the sentiment in text data. Sentiment analysis involves determining the emotional tone expressed in a piece of text, classifying it as positive or negative.

This is crucial for understanding user and public opinions, customer feedback and social media sentiment. Where the problem lies is in accurately categorizing diverse textual inputs which may contain subtle nuances and context-dependent sentiments. The challenge is to develop a robust system that can with less issues discern sentiments effectively and provide valuable insights into the emotional tone of the text. Large text volumes are time consuming to analysis and very easily prone to bias.

The complexity of this problem arises from the inherent subjectivity and variability of human language. Different people may express the same sentiment in different ways and a single individual may express their sentiments differently at different times. [1]

The project's objective is to create an accurate text classification system with a user interface that can easily integrate with third party platforms.

The project follows four stages; data collection, cleaning and preprocessing where feature engineering techniques are applied; building and comparing machine learning models using ML methods supported by Mlpack.

The hosting phase involves serving the model, over HTTPS to allow access while the web server phase involves communication, with the model server and serving users.

Example-Problems:

- **User-unfriendliness;** From the standpoint of the end user, interacting directly with most machine learning models can be somewhat complicated; this project streamlines the interaction process. I'm referring to the hugging face model platform, where using a model might be really difficult to begin with.

- **Integration;** Most machine learning model are also quite hard to integrate into an already existing application and those that can may involve adding and going through twisted procedures just to have a real-time text classification.

- **Huge data**; Having a database of stored key words that includes negative terms was the conventional or outdated method of completing this assignment. This becomes more expensive if there is a lot of data in the database and you have to go through lengthy rows of texts word by word to analyse a text.

1.2 Goals

- **Real time usage:**

  The ability to perform or provide real time sentiment analysis and text classification. Crucial for social media apps and business apps where users feedback is valuable as quick as possible.

- **Cost reduction**;

  Just taking the traditional way into consideration, the act of storing bad words and iterating through the rows word for word to compare is too time and resource consuming and not cost efficient. This project goal is to minimize such cost.

- **User friendly and Integration friendly**;

  Utilize dynamic HTML content to create an expressive and intuitive frontend experience that is easy for developers and end users to utilize.

- **Accurate and automated sentiment analysis**

  To be able to develop a system that have high accuracy in sentiment analysis, ensuring reliable text classification of positive and negative sentiments.

- **Api endpoints**

  To create a robust backend to handle Api endpoints for data collection and model interaction and database updates. Enables seamless communication between the frontend and the ml models.

- **Database integration, data storage and retrieval**

Integrate a database, in the system we use PostgreSQL to store and manage textual data and model predictions and additional information for historical tracking and retrieval.

- **Containerization and consistent development environment**
  To be able to employ docker for testing, development and deployment ensuring consistency and reproductivity across different environments.

2. Literature Survey

Eden Ai

It is a platform that offers unified Api from multiple Ai providers and provides an interface for businesses and developers who want to incorporate Ai into their systems. [2]

My project's dev-friendly model access is similar to Eden AI's. The distinction is that my project solely focuses on review sentiment solutions, which allows my system to meet the demands and specifications of sentiment analysis tasks.

For instantaneous analysis While it's unclear if Eden AI enables real-time analysis, my project offers real-time analysis.

In terms of user-friendliness, Eden AI currently only offers an API for developers or other non-end users, whereas my project uses an interactive web interface to serve both immediate users and end users.

Eden ai provides the opportunity to build custom ai powered chatbot, customize the model’s features, add custom data collection whilst mine

only provide sentiment text classification. [2] Majority of the models used by Eden ai is mainly powered by python whilst mine uses C++.

Levity Ai

This platform very similar to mine and also focuses more on automated workflows using AI whilst mine do not. We share similarities in integration and usage of AI capabilities. One noticeable difference is Levity AI has a sign in policy while mine don't.

With further in depth digging and research about levity AI and its tech stack tools, a difference of toolsets is also among what differentiate my project with theirs. They utilize a lot of open-source models and fine tune it whilst mine is built from scratch. [3]

Hugging face

Hugging face is an open-source project that builds and deploys pre-trained models. They enable the download of already pre-trained models mine does not. Has a platform where researchers, students and developers share their work. They provide wider range of tools whilst mine is more lightweight and focus on a tiny little sector of what hugging face serves. The models served by hugging face are very complex to use and also to integrate you will need a deeper understanding of NLP concepts and python programming and bash skills before you can integrate it into your system.

The goal of hugging face is also very different from mine and the previous compared platform. They are on a mission to democratize good machine learning, one commit at a time. [4]

Lexalytics

Lexalytics is a platform that provides advanced text analysis and natural language processing service. They have been developing Natural language processing service for over 19 years with wide range of capabilities including sentiment analysis, entity detection, categorisation and more. They provide flexible deployment use cases, from public cloud infrastructure to on-prem. They support over 29 languages and provide industry-specific configurations. Their services also include access to pre-built models, custom machine learning model deployment, salience; a complete NLP platform for data scientists and architects, semantria; a restful API for integration into their cloud-base enterprise data analytics and spotlight; a tooling for storing, analysing and managing unstructured text data. [5]

Qualtrics

Qualtrics is a management platform that uses AI to transform data insights into personalized experience and impact. They offer 3 suites; Customer experience, employee experience and strategy & research.

This company main focus is to extract the insights or sentiments from customers feedback and use it to drive their decisions on how to market better to their customers. [6]

3. Background Information

3.1 Required software

- **Typescript**

  A strongly typed programming language for JavaScript that helps set rules in order to reduce bugs and runtime errors that stems from JavaScript dynamic nature. [7]

- **C++**

  It is a well-known robust, powerful and efficient language. İt was picked to be able to handle the real time analysis of the system with low latency and also was picked due to the needs by my supervisor.

- **Mlpack Library**

  Due to the use of C++ programming language, arises the need of a machine learning library in C++ and Mlpack is a lightweight, fast, header-only library. Once you have developed a machine learning workflow, deployment is straightforward. Mlpack directly depends on only three libraries, Armadillo, Ensmallen and cereal. [8]
  It's quite mature and has support for use in production settings.

- **Uwebsockets**

  a C++ library that is lightweight and used for building real-time web servers with high performance and scalable solution. Its best features is been fast, easy to integrate into existing projects and memory efficient. [9] This is the recommended C++ server library to host the Mlpack

model but this project will only rely on the HTTP part of the library instead on the WebSocket.

- **Nlohmann json library**

  Json library that supports manipulation of Json data in C++ because most of the data collected will be in Json file format therefore this library is needed to handle them.

- **ExpressJS:**

  A JavaScript library for building backend servers and Api routes and capable of running in almost all JavaScript runtimes be it bun, node, deno, etc.

- **PostgreSQL C++ library**

  To connect your C++ application with PostgreSQL database, you would need a C++ library. This library provides an API to interact with PostgreSQL.

- **EJS engine**

  this is a JavaScript templating engine, runs with any JavaScript runtime that is use to generate dynamic html content for the web.

- **PostgreSQL**

  A robust modern database for the system. This database is picked due to the cloud hosting platform's easy Postgres integration.

3.2 Other software

- **Adobe Illustrator**

  Adobe Illustrator is a vector graphics editor and design program developed and marketed by Adobe Inc. [10]

  Use to create mock-up of the web interface design and draw the use case diagrams and workflows of the project.

- **Visual Studio Code**

  A code editor that supports multiple languages and have remote ssh capabilities to build and develop inside dockerized containers in isolated environments.

- **Matplotlibcpp**

  A C++ header wrapper for a python graphing software library to help visualize the dataset and performance stats during model training. Matplotlibcpp wraps around matplotlib which is a library for create interactive, static or animated visuals of datasets. [11]

- **Docker**

  A containerisation platform that is used to set up the development environments for building the projects, where I use visual studio code to remote access into an isolated docker container, package and deploy the software. This is one of the best tools I love to use. It helps you test and try different OS images and find the optimal image that just does what you want and no more.

- **Cmake**

  A building system that is used to build the various C/C++ libraries for both the development environments and production environments.

  It is the de-facto standard for building C++ code, it is comprehensive, powerful solution for managing the build process. [12]

- **Git**

  A version control system to track the project's progress and development's history. This also helps me roll back to previous working state of the project when errors appear.

4. Modules

4.1 Training Module

This module is used by the developer and is responsible for collecting and preprocessing the review data, training the text classification model using Mlpack library.

4.1.1 Data Collection and preprocessing

This submodule collects data for training from various source but for this project it's from a specific source. Cleans, breaks the test down into individual tokens/text by a tokenizer type and vectorize the tokens into numerical vectors.

4.1.2 Model Training algorithm selection

Responsible for picking out the machine learning methods that is best suitable for the prepared features with its targets. Trains and optimizes the hyperparameters of the model.

4.1.3 Model prediction and evaluation

Responsible for transforming new textual data into the appropriate format, feeding newly pre-processed data into the trained model and evaluates the model's performance. Comparison of different model's techniques are also carried out here. Saving of the best model is the final step of the module.

4.2 Server Module

This module is in charge of loading the saved models gotten from the Training module's final step using Mlpack supported deserialization and serialization methods and serving the trained model for the web through defined routes and handles HTTP requests and respond with the classification results.

It utilizes Uwebsockets to create a responsive and efficient web server, listens for incoming requests and in turn is all containerized in a docker container to ensure consistent server environment.

4.3 Web server module

This module is responsible for providing the user-friendly interface that interacts with the server that hosts the machine learning model.

4.3.1 Frontend/View module

A module responsible for the presentation logic, how the data is presented to the end user and handles the end user's inputs and alters the view accordingly to user inputs. Tooling used in here is the ejs engine. For dynamic content generation.

4.3.2 Backend module

This module handles the backend logic of providing dynamic content generation, handling API request, interacting with databases.

This uses express to define the Api endpoints and also to communicate with the database to store or retrieve data and also generate dynamic html.

4.4 Database module

This module purpose is very straight forward. It stores and manages all forms of data based on the system requirement for the system use. Also helps to persists the data and provides it when readily needed.

5. Methodology

Introduction

The creation of a sentiment prediction web application stands as a testament to the power of interdisciplinary collaboration, merging the precision of machine learning with the artistry of design and the robustness of software engineering. The journey begins with the acquisition datasets which was pre-processed, which is the cornerstone of any machine learning endeavours. This dataset undergoes a rigorous process of cleaning and preprocessing to ensure that the data fed into the model is free of inconsistencies or contains minimal impurities and is representative of the real-world scenario it aims to model. Feature extraction is then meticulously carried out, identifying the most relevant data points that will feed into the predictive model. The techniques used to pick what text feature extraction is also utilized here.

The selection of the machine learning algorithm is a critical decision that influences the effectiveness of the application. Algorithms are evaluated based on their performance, ease of interpretation, and computational efficiency. Once chosen, the model is trained iteratively, with each cycle refining its ability to discern patterns and make accurate predictions. Hyperparameters are fine-tuned to optimize the model's performance, a task that requires both technical expertise and a deep understanding of the underlying data.

The backend of the web application is engineered to support the computational demands of the machine learning model, ensuring swift and reliable predictions. It acts as the backbone of the application, managing data flow and user interactions with efficiency and security. The frontend, meanwhile, is designed with the user in mind, providing an intuitive interface that simplifies complex processes into user-friendly interactions. It is the conduit through which users engage with the application, inputting their data and receiving predictions.

The integration of the machine learning model into the web application is a delicate process that requires careful planning and execution. The model must be seamlessly embedded within the application's architecture, allowing for real-time predictions that are both accurate and rapidly delivered. This integration is pivotal to the user experience, as it bridges the gap between the sophisticated workings of the machine learning model and the straightforward simplicity of the user interface.

Finally, the deployment of the web application marks the culmination of this intricate journey. It is a stage that not only involves the technical aspects of hosting and maintaining the application but also encompasses considerations of scalability, accessibility, and user support. The application is monitored and updated regularly, ensuring that it continues to function at its best and adapts to the evolving landscape of technology and healthcare.

In essence, the development of a diabetes prediction web application is a multifaceted endeavour that spans various domains of expertise. It is a blend of science and technology, creativity and analysis, all converging to create a tool that has the potential to impact healthcare outcomes significantly. Through this comprehensive process, developers and healthcare professionals alike can contribute to the early detection and management of diabetes, ultimately enhancing the quality of life for those affected by this condition.

Preprocessing

The dataset utilized to train the machine learning model came from Snap's web data of Amazon reviews. The data span a period of 18 years up to March 2013. Reviews include product and user information, ratings and a plaintext review. [13]

A row of the dataset contains the review's text, score, summary, user id, profile name, time and helpfulness also the product's product id, price, title. The review text is the actually text we need to clean and used for prediction whilst the review score is the criteria we will use to spilt the sentiment of the review text. The threshold for it was 3, above 3 is positive and below 3 is negative sentiment.

This project has a class for preprocessing that handles all preprocessing tasks.

Steps performed during preprocessing;

Json Parsing

This preprocessing task was only needed due to how the dataset were stored, so to be able to effectively extract needed information and discard unnecessary data, the entire dataset needed to be parsed and the utilization of the library called nholmann Json helped a lot

Stop words removal

In terms of text preprocessing there are words that has no real sentimental meaning and therefore not necessarily needed as that will screw the metrics of validity of the predictions. Therefore, the removal of stop words tends to be one of the first early steps to preprocessing text for any Natural language processing task. The common examples of stop words are I, a, am, the, it etc.

The approach I used to handle this was to have a method function for the preprocessing class. A text file that contains known list of stop words. The method functions load the text file for stop words and convert it into lowercase and also the text to preprocess is also transformed into lowercase too. The text is then checked if it contains a

word in the stop words vector, if one is found, it is replaced with whitespace. Thus, removing the word from the text vector.

Stemming

Stemming is another text preprocessing task whose job is to make sure all words with the same meaning are represented by a single word. The importance of this arises from words which have different forms due to when they were used and which direction they meant but they all have the same sentiment weight. A good example of the is excellent, it also has the form excellence. They both have the same weight to sentiments but they are different in spellings and can cause the prediction model to classify it as two different entities, therefore stemming ensures they both have the same weight and same spelling thus viewed as one entity. Stemming isn't all that easy but there are stemming algorithms that we can utilize like the porter stemming algorithm.

The porter stemming algorithm also known as porter stemmer is a process for removing the commoner morphological and inflexional endings from words in English. It normalises information crucial for information retrieval systems. [14]
One of the caveats for porter stemmer is that is does not leave a real word after its operations this is dues to its purpose to bring variant divergent forms of a word together and not to map words to a parent word or paradigm form.

The approach used to handle the stemming of the words was done in the preprocessor stem member functions.

Arbitrary character removal

This I found is very much needed as there are some characters that have absolutely zero use for sentiment analysis characters like the number systems, foreign symbols etc. Alphanumeric words are an example of these types of words and after performing the operation to remove arbitrary characters from the corpus of text and noticed the best way to actually have almost a high percent of the corpus cleaner is to first

remove the stop words, then remove the arbitrary characters then remove stop words again to help get those words that were kind of messed up by the operation of removing the characters. An example of this is the words **"123a456"** and **"9shan't"**, if stop word operation is first ran through this, they both won't be removed, but when the arbitrary character removal operation is run through it becomes like this **"a"** and **"shan't"**. These can easily be picked and removed by the stop word removal operation.

Threshold set

This operation purpose is very simple and easy to understand. It divides the entire corpus of text into the two sentimental boundaries of positive and negative. The default threshold used is 3 out of the range of 0 to 5. This operation can be found in the preprocessor class method function.

Save Equally

This simply finds with corpus with the smallest number of dataset/text and set all to be of that length. The importance of this operation is to enable that all classes for classification have the exact same number of occurrences in order to prevent biases when been utilized by certain model during prediction.

Set Limit

This operation purpose is to set the exact number for the corpus sentimental boundaries. It checks the length of one of the equally divided corpuses to see if the new limit set exceeds the length. If false it cuts the length of all the corpuses to be that length but if true (i.e. the new specified length is larger than the length of the corpuses) then nothing happens.
The importance of the method function is to limit the total number of dataset points in order to use it in a limited memory/small memory use case.

Feature Extraction

This is crucial as machine or computers only understand numbers and not text, so we need a way to represent the text into a format the computer would understand and would have meaning.

Feature extraction is the process of extracting information from text to help analyse the text data for many applications namely data mining, clinical record etc. [15]

Speeches and text are the most common form of unstructured data. NLP helps to obtain information and analyse this data to conduct tasks such as span filtering, on-the-time language or text translation, sentiment analysis. It does this by classifying the text into group based of its context and assigns predefined categories or tags. [15]

Machine learning models will require numerical inputs that have or contain a meaning, so we convert the text into numerical vectors. There are several techniques but two were mainly used in this project:

Bag of Words:

This is the simplest technique to understand and implement and has seen great success in problems. It is a representation of text that describes the occurrences of words within a document and involves two points. A vocabulary of known words and the measure of the presence of known words. It is coined “bag” due to the discard of information about the order of words in the document and only mainly concerned with whether the word occur in the document not where in the document. [16]

It gives the impression that documents are similar if similar words appear and from that may use it for something relating to the meaning of the document. [16]

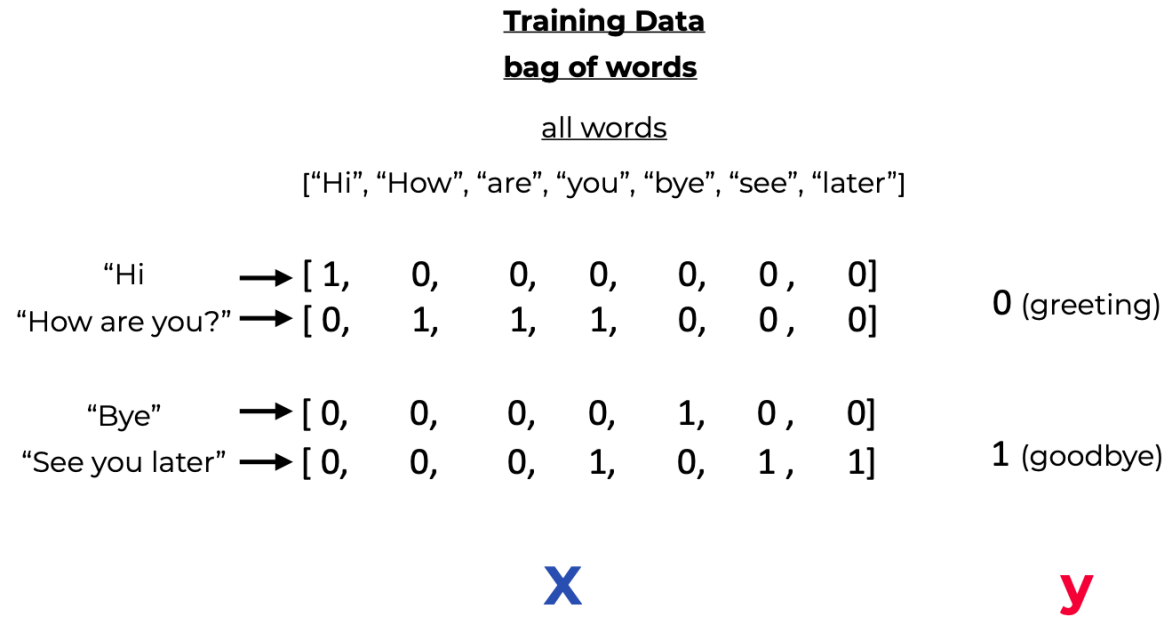


Figure 1Bag of Word [27]

Word scoring:

The appearances of words in the documents need to be score by:

Counts; the total number of times each word resurfaces in the document.

Frequency; the word frequency that each word resurfaces in a document out of the rest. [16]

TF-IDF:

This is a technique to help punish words that always frequently appear almost everywhere which may be problematic for classification. The highly frequent words start dominating and screwing the accuracy if measures aren't taken to give them low precedence or priority. [16]

TF-IDF is abbreviation for Term Frequency – Inverse Document Frequency. Term frequency is for the frequency of the word in the current document. While the Inverse Document Frequency is for how rare the word across in the documents. [16]

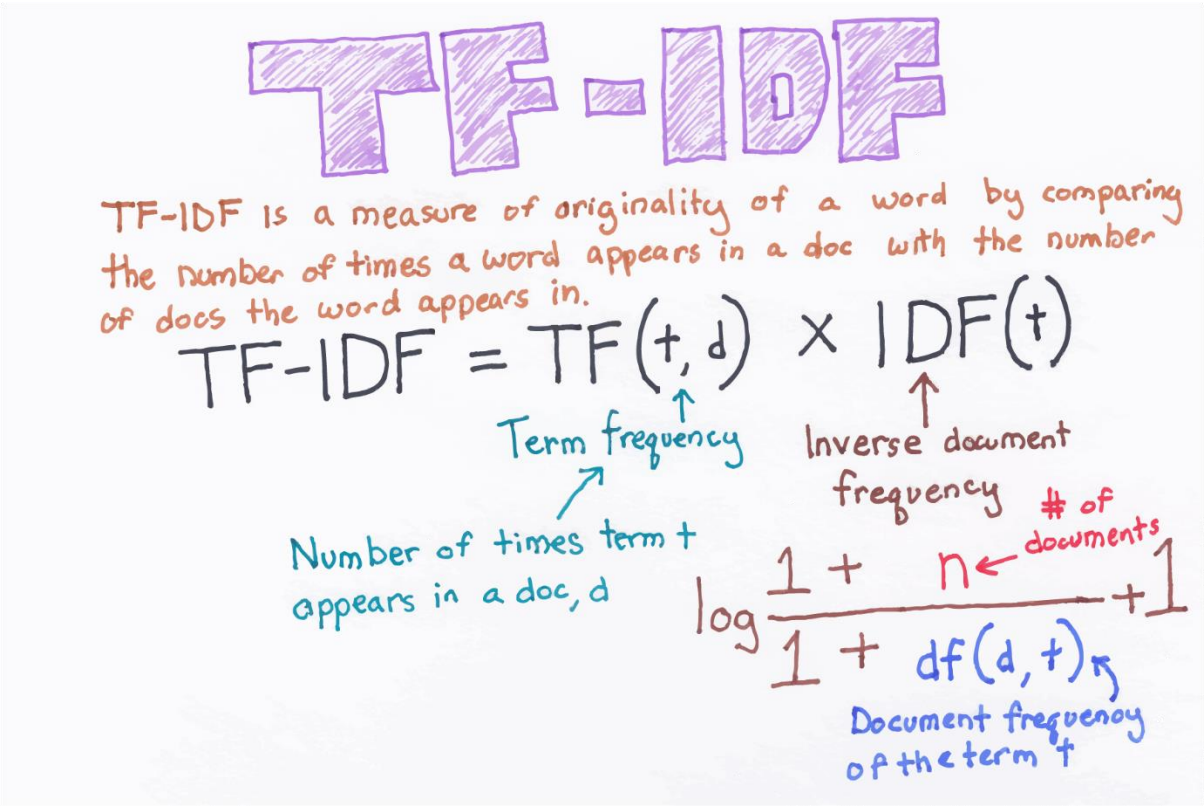


Figure 2 TF-IDF [17]

An example of this may be some strong words like genocide, war, death etc. These words are usually rare in normal conversation or rarely gotten from data sources that deal with generalize data, but if they appear in the document. They hold and carry a very lot of weight and bend all metric and computations to their favour.

Modelling

Here is usually the place where repetitions and having fundamental knowledge on why it works and what to modify if needs be require advance skill in machine learning algorithms. These are some of the models used for this project:

Logistic Regression;

This is a statistical model that estimates the log-odds of an event as a linear combination of one or more independent variables. These independent

variables that it uses for this project is individual unique words. Logistic regression is a supervised machine learning technique used for binary classification and it just tends to favour use as we are only looking for positive and negative sentiments which can be gracefully mapped as binary 1's and 0's.

Its technique is the usage of the logistic function's ability to model the probability of binary outcomes. It is called regression as it is an extension of linear regression mainly used in classifying and some difference between linear regression and logistic regression is that logistic regression fit an S shaped logistic function which is also known as sigmoid function instead of fitting a regression line.

There are three types of logistic regression based on categories;

Binomial:
As the name implies, there can only be two possible types of the dependent classes and are majorly used in binary classification.

Multinomial:
There can be more than 2 possible types of the dependent classes that are unordered.

Ordinal:
Same as multinomial but the dependent classes are ordered.

Nearest mean classifier;

The Nearest mean classifier which is also known as Nearest prototype classifier or Nearest centroid classifier is a simple machine learning classification algorithm.

This type of algorithm uses the entire training dataset to predict the outcome from new data. Most models update weights to match and learn to be able to predict. The model is first fitted with the train data and uses the mean vectors of the classes to classify the new data points.

The mean of all the vectors in each class that belongs to a feature are calculated and stored. To make predictions, the distance to each mean of the classes is calculated and the class which holds the smallest or shortest distance is selected as the predicted class.

K-nearest neighbour;

KNN is a supervised algorithm that can be used for classification and regression and one of the simplest yet powerful machine learning algorithm but for my project it was the one that performed the worst.

KNN is similar to Nearest mean classifier but the only difference is that NMC fits the training data and stores the mean of the data whilst KNN stores the entirety of the training data.

Prediction process is also similar to NMC where distance equations are used to calculate the distance to all training samples. Then the K(number) of training samples with the smallest distance to the test sample are taken.

The next step depends on the engineer method of picking the final result, most use majority vote, most use the top most sample etc.

Feed forward artificial neural network;

This is a type of artificial neural network where data flows only in one direction (forward) from the input layer through the hidden layers and then to the output layer. It lacks loops and cycles in the network which is the primarily key definition for recurrent neural networks.

A simplified description of FFN looks like this;

Input node/layer;

This node receives the data for the network and the amount of data in this node is equivalent to the number of features in the input dataset.

Hidden node/layer;

These are the nodes that are sandwiched between the output and input layer. These nodes or should I say layers acts as the layer that learns, adjust weights and biases and can have any number of hidden nodes. The data that passes through this layer also passes through activation functions.

Output node/layer;

This is the final point in the network and the number of data points in this layer is equivalent to the expected number of points needed.

Bias and Weights;

Each and every connection between the nodes has each dependent weight and bias which will be modified when training to minimize error and steer the network to the expected results in the process reducing errors and improving accuracy. Learning rate can also be included in this section too, but its best that the learning rate is constant for all node and layers in the network.

Training;

The network uses backpropagation which entails the feeding the data that was forwarded from the network to generate prediction and going backward in the reverse direction in order to adjust the weights. Before the network moves backward in order to modify the weights and biases the forward prediction is tested with the already provided labels and the error is taken with it backwards.

Linear Regression (shocking, I know);

This is a statistical model used to predict a dependent variable. At first, I was very sceptical of using this model as I did not expect the dataset to be almost possible to be cut by a straight line but was shocked that at the initial build it performed better than Linear SVM and Naïve bayes but below Random Forest, FFN, Logistic regression and NMC. The base expectation for this model was low but it performed above its expectations.

This model uses a straight line to fit between its dependent and independent variables. It assumes the data is linear.

Linear-SVM;

Linear Support Vector Machine (Linear SVM) is a type of Support Vector Machine (SVM) that uses a linear function to separate the data points of

different classes. It's a supervised machine learning algorithm used for outlier detection, classification and regression. [18]

Here's how it works:

1. Model: The Linear SVM algorithm finds a hyperplane in an N-dimensional space (N is the number of features) that distinctly classifies the data points.

2. Training: During the training phase, the Linear SVM algorithm tries to find the optimal hyperplane that separates the data points into their respective classes. The hyperplane is chosen such that the margin between the closest points of different classes is maximized.

3. Prediction: When given a new, unseen instance, the Linear SVM algorithm determines which side of the decision boundary the new instance falls on. This determines the class of the new instance.

In a 2-dimensional space, the hyperplane is a line, and in a 3-dimensional space, it's a plane. [19] For more than three dimensions, it's hard to visualize but the concept remains the same.

Linear SVMs are very suitable when the data can be precisely linearly separated[2]. However, they may not perform well if the actual relationship between the variables is not linear.

Naïve bayes;

Naive Bayes is a probabilistic machine learning algorithm that's typically used for classification tasks. [20] It's based on the Bayes' Theorem and makes a strong (or 'naive') assumption that all features in the dataset are independent of each other. [21]

Here's how it works:

1. Training: During the training phase, the Naive Bayes algorithm calculates the probabilities of each attribute (feature) of the data given each class label. This is done by counting the frequency of each attribute-value pair for each class.

2. Prediction: When given a new, unseen instance, the Naive Bayes algorithm calculates the probability of each class given the instance's attributes. It does this by applying Bayes' Theorem and then assumes that all attributes are

independent. The class with the highest probability is then assigned to the instance.

Random Forest;

Random forest is a powerful, versatile and in my words one of the slowest models built in this project and it's used for classification tasks.

Decision tree is a type of machine learning algorithm used to classify data. The most low-level explanation of it looks like a tree of fi-else statements that creates a path to a decision or outcome. It is a supervised algorithm that combines and grows multiple decision trees that are uncorrelated or low to no correlation between the trees, merge them together to produce a forest for a more accurate prediction. The algorithm chose its result based on majority votes. The trees inside the random forest algorithm are usually trained using the **"bagging"** method. [22]

The "bagging" method is a type of ensemble machine learning algorithm called Bootstrap Aggregation. An ensemble method combines predictions from multiple machine learning algorithms together to make more accurate predictions than an individual model. Random Forest is also an ensemble method.

Bootstrap randomly performs row sampling and feature sampling from the dataset to form sample datasets for every model.

Aggregation reduces these sample datasets into summary statistics based on the observation and combines them. Bootstrap Aggregation can be used to reduce the variance of high variance algorithms such as decision trees.

Variance is an error resulting from sensitivity to small fluctuations in the dataset used for training. High variance will cause an algorithm to model irrelevant data, or noise, in the dataset instead of the intended outputs, called signal. This problem is called overfitting. An overfitted model will perform well in training, but won't be able to distinguish the noise from the signal in an actual test.

Bagging is the application of the bootstrap method to a high variance machine learning algorithm. [22]

The advantages of the model in this project were that it was the best and easier to set up and train and was among the ones with the highest accuracy with little hyper parameter tunning

Its only disadvantage is that this model was by far the slowest of all the models trained for this project. Also, the eats a whole lot of memory due to it using many decision trees.

Evaluation

This is the next important step in the entire machine learning process which involves validating the model's performance to determine how much the model has learned from the train data and how well it we perform on unseen data.

There are several aspects of evaluating the models used;

Holdout method;

This is the approach of splitting the entire dataset into train and test set. The model is feed and fit with the train dataset and evaluated on the test dataset. [23]

Metrics;

These are quantitative assessments used to track the performance of the model. This is because developing models is a multi-section process and checks on how well the model predicts should be done and these are the metrics used for the model evaluation;

Accuracy;

This is a metric that measures the overall correctness of a model's predictions. It represents the ration of the correctly predicted samples to the total number of samples in the dataset. It is expressed mathematically as

$$Accuracy = \frac{\text{Nuber of correct predictions}}{\text{Total number of predictions}}$$

Precision;

This metric is concerned at the rate of having true positives in respects to other aspects of positives and can be mathematically be represented as

$$Precision = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall;

This metric is for the true positives rate sensitivity from the model's prediction and can be mathematically be described as

$$Recall = \frac{\text{True Positive}}{\text{True Positives} + \text{False Postives}}$$

F1 Score

This is a metric evaluation technique that utilizes the precision and recall of the model's prediction. The value of F1 score lies between 0 and 1 with 1 being the best score. It is the harmonic mean of precision and recall and can be expressed mathematically as

$$F1\ Score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Support

This metric just all about the number of the actual appearance of the class in the dataset. This metrics gives a view on which class the model is leaning on in its prediction process and is very helpful in hyper tunning the model bias towards a particular class.

Mean Square error;

This metric is widely popular for regression tasks as it measures the average difference squared between the predicted value and the actual value. If one is familiar with Euclidean distance the mean square error look like it but without the square root. It can be mathematically represented as

$$MSE = \frac{1}{N} \times \sum_{i}^{N} (\hat{}_{i}Y - Y_i)^2$$

Where $\hat{}_{i}Y$ is the predicted value by the model and Y_i is the true value of the sample

Comparison with other models

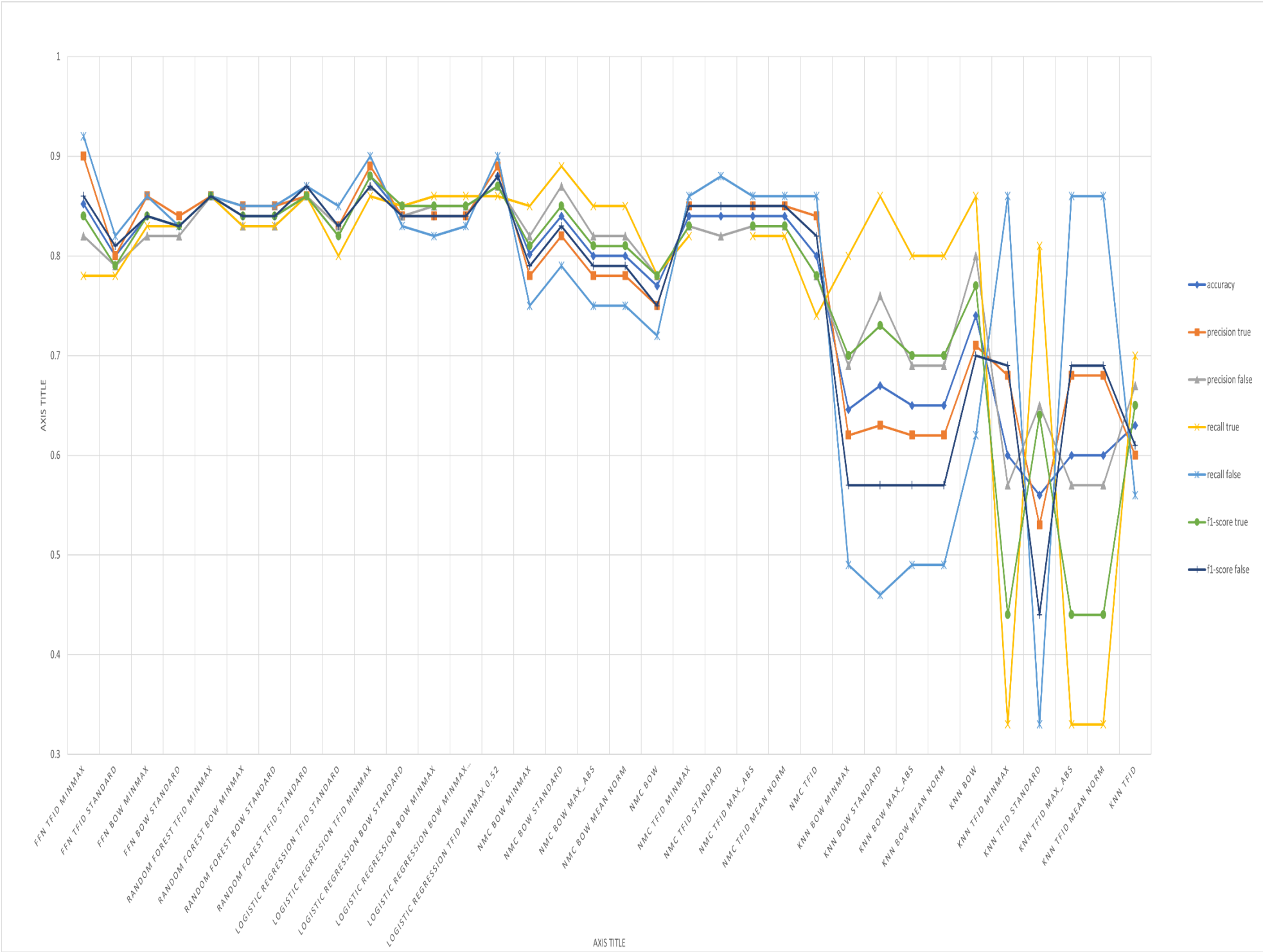
This is usually the last step in the evaluation process of building models where multiple models are built and test and the decision to pick which model performs the best towards a given set of criteria is done here.
Sometimes the criteria may be the speed that makes the model to be chosen. Sometimes it may be how accurate it is with a stab to its speed. Sometimes it's having both good speed and good accuracy. Sometimes it's the size in disk or memory is consumes. There are so many issues that affect the type of model chosen and it's all up to the situation presented to the developer.

Experiment, Result and Discussions
During the train stage of the experimentations of the models these are the result I was able to acquire from the different models and different scalar methods plus string encoding policies. Below are the table results

	accuracy	**precision true**	**precision false**	**recall true**	**recall false**	**f1-score true**	**f1-score false**
FFN TFID MINMAX	0.85214	0.9	0.82	0.78	0.92	0.84	0.86
FFN TFID STANDARD	0.8	0.8	0.79	0.78	0.82	0.79	0.81
FFN BOW MINMAX	0.84	0.86	0.82	0.83	0.86	0.84	0.84
FFN BOW STANDARD	0.83	0.84	0.82	0.83	0.83	0.83	0.83
RANDOM FOREST TFID MINMAX	0.86	0.86	0.86	0.86	0.86	0.86	0.86
RANDOM FOREST BOW MINMAX	0.84	0.85	0.83	0.83	0.85	0.84	0.84
RANDOM FOREST BOW STANDARD	0.84	0.85	0.83	0.83	0.85	0.84	0.84
RANDOM FOREST TFID STANDARD	0.86	0.86	0.86	0.86	0.87	0.86	0.87

LOGISTIC REGRESSION TFID STANDARD	0.82	0.83	0.83	0.8	0.85	0.82	0.83
LOGISTIC REGRESSION TFID MINMAX	0.88	0.89	0.87	0.86	0.9	0.88	0.87
LOGISTIC REGRESSION BOW STANDARD	0.84	0.84	0.84	0.85	0.83	0.85	0.84
LOGISTIC REGRESSION BOW MINMAX	0.84	0.84	0.85	0.86	0.82	0.85	0.84
LOGISTIC REGRESSION BOW MINMAX 0.52	0.84	0.84	0.85	0.86	0.83	0.85	0.84
LOGISTIC REGRESSION TFID MINMAX 0.52	0.88	0.89	0.87	0.86	0.9	0.87	0.88
NMC BOW MINMAX	0.801556	0.78	0.82	0.85	0.75	0.81	0.79
NMC BOW STANDARD	0.84	0.82	0.87	0.89	0.79	0.85	0.83
NMC BOW MAX_ABS	0.8	0.78	0.82	0.85	0.75	0.81	0.79
NMC BOW MEAN NORM	0.8	0.78	0.82	0.85	0.75	0.81	0.79
NMC BOW	0.77	0.75	0.78	0.781	0.72	0.78	0.75
NMC TFID MINMAX	0.84	0.85	0.83	0.82	0.86	0.83	0.85
NMC TFID STANDARD	0.84		0.82		0.88		0.85
NMC TFID MAX_ABS	0.84	0.85	0.83	0.82	0.86	0.83	0.85
NMC TFID MEAN NORM	0.84	0.85	0.83	0.82	0.86	0.83	0.85
NMC TFID	0.8	0.84	0.78	0.74	0.86	0.78	0.82
KNN BOW MINMAX	0.645914	0.62	0.69	0.8	0.49	0.7	0.57
KNN BOW STANDARD	0.67	0.63	0.76	0.86	0.46	0.73	0.57
KNN BOW MAX_ABS	0.65	0.62	0.69	0.8	0.49	0.7	0.57
KNN BOW MEAN NORM	0.65	0.62	0.69	0.8	0.49	0.7	0.57
KNN BOW	0.74	0.71	0.8	0.86	0.62	0.77	0.7
KNN TFID MINMAX	0.6	0.68	0.57	0.33	0.86	0.44	0.69
KNN TFID STANDARD	0.56	0.53	0.65	0.81	0.33	0.64	0.44
KNN TFID MAX_ABS	0.6	0.68	0.57	0.33	0.86	0.44	0.69
KNN TFID MEAN NORM	0.6	0.68	0.57	0.33	0.86	0.44	0.69
KNN TFID	0.63	0.6	0.67	0.7	0.56	0.65	0.61

Table 1 Models results


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
AXIS TITLE
AXIS TITLE
FFN TFID MINMAX
FFN TFID STANDARD
FFN BOW MINMAX
FFN BOW STANDARD
RANDOM FOREST TFID MINMAX
RANDOM FOREST BOW MINMAX
RANDOM FOREST BOW STANDARD
RANDOM FOREST TFID STANDARD
LOGISTIC REGRESSION TFID STANDARD
LOGISTIC REGRESSION TFID MINMAX
LOGISTIC REGRESSION BOW STANDARD
LOGISTIC REGRESSION BOW MINMAX
LOGISTIC REGRESSION BOW MINMAX...
LOGISTIC REGRESSION TFID MINMAX 0.52
NMC BOW MINMAX
NMC BOW STANDARD
NMC BOW MAX_ABS
NMC BOW MEAN NORM
NMC BOW
NMC TFID MINMAX
NMC TFID STANDARD
NMC TFID MAX_ABS
NMC TFID MEAN NORM
NMC TFID
KNN BOW MINMAX
KNN BOW STANDARD
KNN BOW MAX_ABS
KNN BOW MEAN NORM
KNN BOW
KNN TFID MINMAX
KNN TFID STANDARD
KNN TFID MAX_ABS
KNN TFID MEAN NORM
KNN TFID
accuracy
precision true
precision false
recall true
recall false
f1-score true
f1-score false

These were the few models tested on the train dataset and looking at it raw in this form may not be the best way to get insights on which models perform better therefore a chart depicting them will be better.

From the graph above ***FFN TFID MINMAX*** is the best performing model I have trained for this project. But even if the FFN model was the best one, the logistic regression model ***TFID MINMAX 0.52*** was second place in terms of how high is evaluations are and some might say it's the better one as all the evaluation metrics are closer to each other than the FFN model. The caveat of the FFN model is that its binary size is too large compared to what the logistic regression model leaves behind and due to the low memory of the cloud provider the logistic model will be picked.

Below are individual graphs for individual models built;

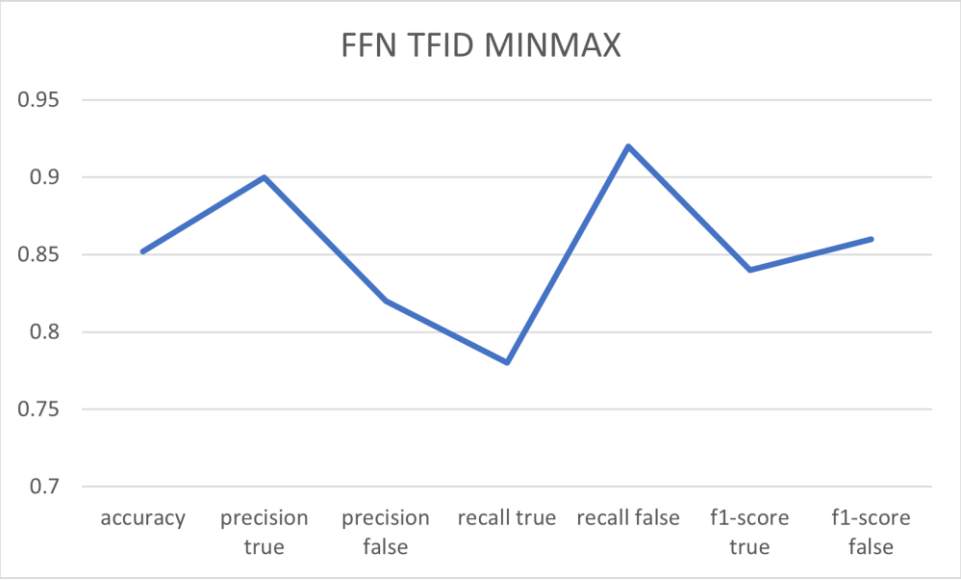


Figure 3FFN evaluation metrics

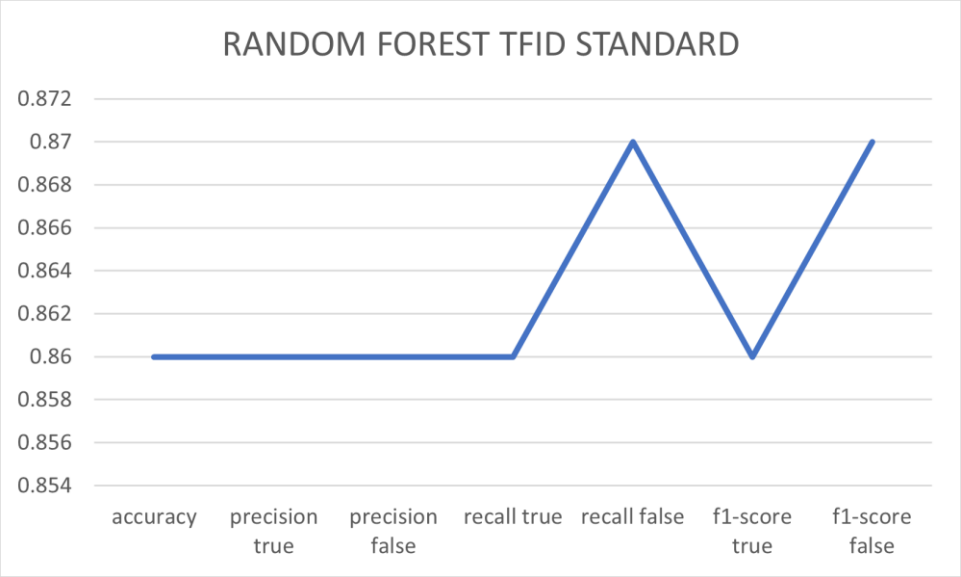


Figure 4Random Forest evaluation metrics

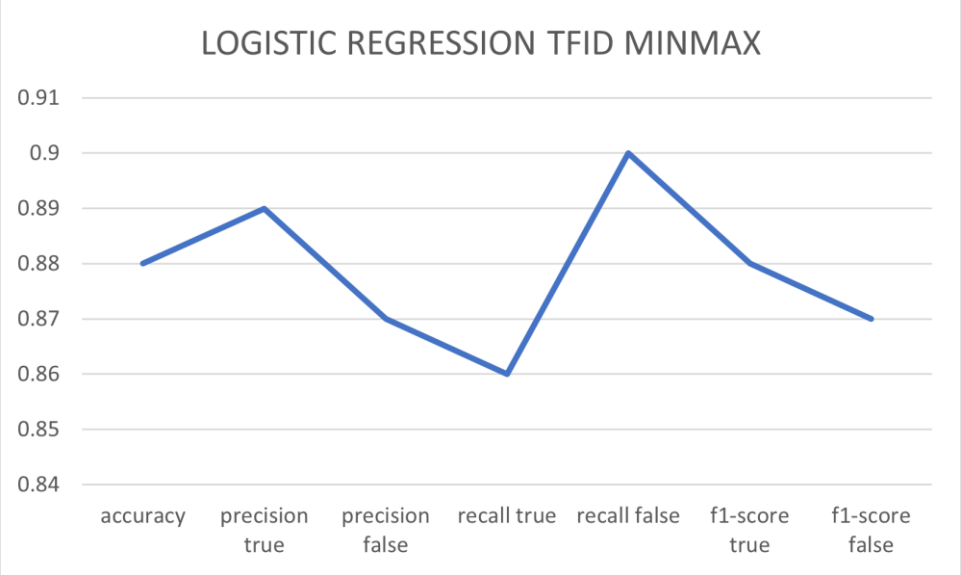


Figure 5Logistic regression evaluation metrics

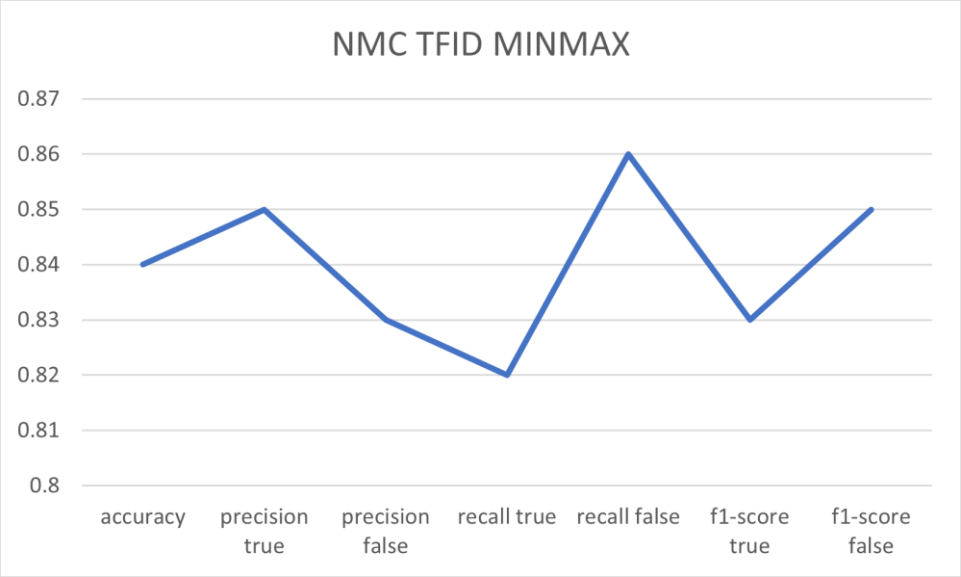


Figure 6Nearest Mean Classifier evaluation metrics

There is virtually no need to bring in the report from KNN model as it was very underperforming. From the graphs there is something quite noticeable with the graphs. Model FFN, NMC and Logistic regression all have similar evaluation graph shape.

The models NMC and KNN was rebuilt by me from scratch so therefore one of them needs to be shipped for end-users to use and the choice of which gets picked is very clear. The KNN model performed poorly while NMC performed above the 80% threshold.

The keyword MINMAX is an acronym for the Min-Max Normalization.

Min-Max Normalization which is a technique used to transform numerical data into a specific range, where a vector point is subtracted from the minimum vector point in the full vector space and divided by the subtraction of the maximum vector point in the space from the minimum one.

$$(\ MinMax\ Norm\ ) \quad X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Standard Scaler is a technique that standardizes features in a dataset to have zero mean and unit variance and it's calculated with the vector being subtracted by the mean and divided by the standard deviation.

$$(Standard\ Scaler\ )\quad Z = \frac{X - \mu}{\delta}$$

Model server development

The process of building the server that will hosts the machine learning model was quite interesting as the models were built with C++ programming language so therefore having a server that serves in model in the same language is the best optimal solution.

The server library used was Uwebsockets which is a wrapper to usockets library. The usockets library is a very low-level library for non-blocking optimized networking. Due to it hosting the machine learning models which can grow tremendously in size and consume disk space, usockets light weight and no dependencies feature make it's a very brilliant TCP library to use whilst Uwebsockets which builds on usockets provides some higher-level implementation of handling network sockets and abstracting a lot.

The first step was to find a version of usockets and Uwebsockets to use. The finalized settled versions were v0.8.7 for usockets and v20.56.0 for Uwebsockets. A custom Cmake file had to be included into the usockets directory for the root Cmake file to know how to build and include it into the final executable.

Uwebsocket provides two type of server app; SSL and no-SSL app. The importance of SSL helps ensure that whatever you server send through the internet is encrypted and hard to read from outsiders. Another easier way of understanding the need of SSL is to prevent web browsers from warning your end-users that the content is unsafe. No SSL app is just like the SSL, has all functionalities but no secure data communication and web browser will go on haywire mode, warning end users of potential risks of communicating with the No SSL server.

The app provided by Uwebsockets contains member methods for GET, POST, PUT etc. The model server is only concerned with GET and PUSH requests. In order to also ensure that most systems won't communicate directly to the

model server I added some security checks. It is a fairly simple one but it will only be known if the person works at the development of the web server that communicates with the model server or the third party has access to the source code of the model server.

The guard checks for certain keywords value and content in the requests and if the request lacks the hard coded required expected value the request is denied without the data flow reaching the models. I personally think this will also help in reducing computational costs as the server will halt the data flow to the models which may carry out much high computational power than the rest of the instructions in the executable.

If the request is a genuine one, the data instruction flows to the model which performs the necessary tasks and returns a Json string that tells the results of the request. The requests will usually enter into a preprocessing and feature extraction state before being feed into the model for results.

Previously the roadmap for processing the request before feeding it into the model was like this;

i. Request comes
ii. Request checks processes
iii. Request is json parsed
iv. Request value/data is all made lowercase
v. Stop words are removed
vi. Stem the requests
vii. Arbitrary characters are removed
viii. Stem the requests again
ix. Unseen tokens by the encoder are removed
x. Seen tokens are vectorize based of the encoding policy the model uses
xi. Vectorize data is then sent into the model
xii. Matrix of the prediction is returned
xiii. String/Json value interpretation of the Matrix
xiv. Response is sent in String/Json format

But there were so many unnecessary operations that were done, so I decided to settle on this roadmap;

i. Request comes
ii. Request checks processes
iii. Request is json parsed
iv. Request value/data is all made lowercase
v. Arbitrary characters are removed
vi. Stem the requests
vii. Seen tokens are vectorize based of the encoding policy the model uses
viii. Vectorize data is then sent into the model
ix. Matrix of the prediction is returned
x. Response is sent in String/Json value interpretation of the Matrix

Uwebsockets also helps for 404 requests, where none defined URL path are handled. During reading from requests with Uwebsockets, a handle for on-Aborted must always be called. In the documentation, uwebsocket reads data from the body of the requests using the on-Data member method on the request object and in case of an error in reading the data the on aborted function is called to help handle the server error.

The other features of the library are very much familiar to express framework in the JavaScript ecosystem. The uwebsocket app similar to express app has functions for all the request methods that take in lambda functions that handles the response and requests. It has an on-Listen function that sets the app to run at a specified port and finally the function that runs it all.

Web server development

The tooling's used to build the web server that interacts with the model were;

i. Prisma
ii. Express.JS
iii. PostgreSQL
iv. Ejs – Embedded JavaScript templates
v. TypeScript
vi. Tailwind

Prisma;
It is a next-generation ORM (object relational model) that consists of these tools;

Prisma Client:

This tool helps provide auto generated and type-safe query builder for Node.js and typescript.

Prisma Migrate:

This tool is a declarative data modelling and migration system. It helps create the data structure for you if the database doesn't have any or help migrate to a new database model structure if there are any specified changes in the schema's Prisma file.

Prisma Studio;

This tool is essentially a GUI to view and edit data in your database. It may be a very fit alternative to PGadmin or DBeaver that are graphical user interface programs to view and edit data.

Prisma client is the most used in the project source code and can run in any typescript or Node.js backend application that also includes microservices and serverless applications.

The Prisma studio might have helped but I used DBeaver to interact with the data in the database and was able to extract the beautiful schema diagram that shows the relationship of the data collection to each other.

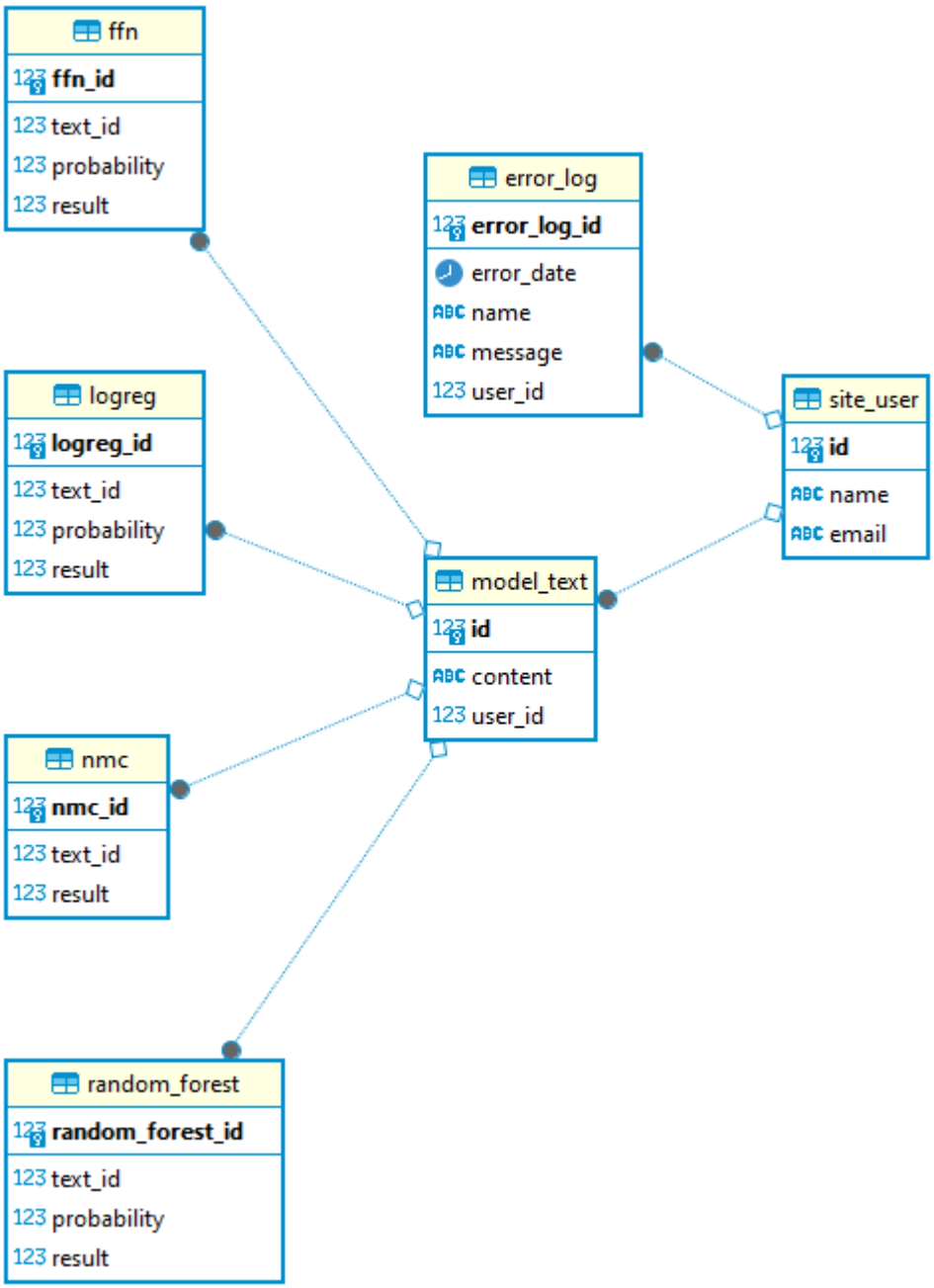


Figure 7 Database Schema

Express;

Express is a fast and unopinionated, minimalist web framework for Node.js with TJ Holowayvhuk as the original author and Douglas Christopher Wilson.

Ejs;

This a essentially a JavaScript runtime, whose sole purpose is to execute JavaScript. It is used widely with express.js to render html document that are

then served by express. There have been countless reports for security issues with ejs, but to clarify ejs is just a JavaScript runtime and if you render vulnerable code snippets, that's what you're going to get and you are responsible for the results.

A key importance is the developer should never allow the end-user access to the Ejs render method since this behaviour of using ejs is an un-secure high-risk way.

Typescript;
A language for application-scale JavaScript and supports tools for JavaScript browser applications in large scale for any OS.

Tailwind;

This is a more fancy, short and fast way of implementing already defined CSS styles for the web server. It is advantageous due to the fact that it negates hours of learning so many CSS styles and concepts.

Web Server modules

Middleware;
The current action performed by the middleware of the web server is to save errors and its messages to the database. This is useful for future references to be able to pinpoint the exact place where the issues arise.

Controllers;
This module action is to help with the necessary control of important utils by the webserver. The module contains the database controller which is responsible for the provision of information for the index page and also saving new text entries into the database. The module also contains the model action that is responsible for reading the model's server URL for the application.

Api;
This module is responsible for structuring and providing the interface or protocol on how to both send and receive data to and fro.

Public/Static;

This module or should I say section of the application contains public files that the client will need to better serve and handle some of their action on the web interface.

Views;

This module is in charge of providing the skeletal structure of how the website will look. All files located here are in ejs format that are rendered are runtime and served by express to the client.

Main application entry module;

This was in charge of handling everything in the web server. It shuts down if no model server URL or port is specified. Its handles cors requests and runs the express application. It utilizes ejs as the rendering engine. It allows express to be able to handle Json requests and response.

Deployment

At first this project was initially documented to be launched with Fl0.com, since they were the 2nd best product of the year 2023. Recently they decided to restrict access to their resource and can only be used through an invite. This sudden change was unexpected but fear not as a perfect and much better clone of them was available render.com. The provide everything fl0.com offers and more. Therefore, the deployment of the project is been done with Render.

The deployment of the model server was done with docker. The initial plan was to save the already built model binaries and training matrixes in Azure

blob storage, build the docker image from scratch using the Docker file in the root directory and during the build stage use wget or curl to download the model binaries and training matrixes from Azure. The approach chosen was to build the image locally with the model binaries and training matrix and then upload the built docker image to docker's hub repository.

The second approach was indeed the fastest and shortest route and so was used. The requirement to deploy it on render.com was to give it the built docker image repository Uri and that's all it took.

Render.com provides an optional to set up a PostgreSQL database and once created returns back with the database credentials which the owner can use to login remotely into the database.

For the web server, it was deployed in render.com as a web service using the node runtime engine. The necessary environment variables were given and the git branch for deploying for the web server was selected. That branch is solely meant for redeploying and is only touched whenever the web server needs an update.

COURSE KNOWLEDGE USED IN THIS PROJECT

COMP117 COMPUTING FOUNDATIONS

This course acts as the introduction to computers and sets the foundational knowledge on how computers work and perform their tasks.

COMP119 INTRODUCTION TO PROFESSION

This course helps address the area of engineering I would be primarily be working on and helps guide the path for which area of expertise my project will be based on.

COMP124 COMPUTER PROGRAMMING

The most important introductory course that introduces the world of programming and the sheer range of what is possible and can be achieved

using programming languages. The majority of the programming skills were first built while taking this course.

MATH109 LINEAR ALGEBRA

The area of the project where this course was of great help were in areas where I need to understand the structure of the matrixes that contains the vectors of vectorize text and how to preform mathematical operations on them.

COMP217 DATA STRUCTURES

This knowledge from data structures came in handy during the model server phase where once the model is been loaded into memory instead of copying the entire model, the reference/pointer to the model is passed and the other operations and use the model by using the pointer that points to the address in memory where the model lives. This was really useful as the RAM used to deploy the model server was small.

COMP218 OBJECT ORIENTED PROGRAMMING I

The area of the project where the knowledge gotten from this course were of help was mainly during the preprocessing phase where a preprocessor class was made to handle all related methods for preprocessing and helps keeps track of the state or type of data file that is been currently pre-processed or cleaned.

MATH226 PROBABILITY & STATISTICS METHODS

Probability and statistics are foundational to machine learning and artificial neural networks for Uncertainty quantification, data distribution understanding and model evaluation as evaluation metrics uses statistical measures.

COMP337 DATABASE MANAGEMENT SYSTEMS

This course resources were utilized in structuring how the data given by the models and how it relates to the text is stored in system. The errors the system

encounters and at which request and how they relate to the error message that pops up.

COMP368 SOFTWARE ENGINEERING

The field of study that involves the disciplined and systematic approach towards developing software with the goal of producing and shipping out maintainable and reliable software. [24]

The aspects of designing where proper planning of the software solution and knowing the required requirements for the project.

The developing and compiling the code and the software tests for errors and bugs. The maintainability of the software was taken into account where I tried to implement error catchers that will be stored in the database for improving and fix bugs and address security vulnerabilities.

COMP464 INTERNET PROGRAMMING

The project needs to have an outlet for end-users to interact with and therefore the creation of the web server where end-user can easily interact with the model. This course teaches a lot on HTML, CSS, JavaScript and those where all used in one way or the other in this project. The html was not necessary used directly but was used as EJS. The CSS was also not used directly but was used in a more advance and convenient form called Tailwind. JavaScript was used with its superset Typescript. The typescript just ensures the JavaScript code always does what it is intended to do without doing some arbitrary operation.

COMP448 ARTIFICAL NEURAL NETWORKS

This is essentially almost the entire scope of this project, utilizing machine learning algorithms and functions to perform classification. There were a lot of models been built and trained and hyperparameter tunings. Comprehensive knowledge of the model and its structure is needed to properly pick which model will be able to perform better and optimally in terms of memory and speed.

ENG434 ENGINERRING ETHICS

Engineering ethics is crucial in Machine learning and Artificial neural networks for ensuring trustworthy Ai development and deployment, whilst promotes fairness and bias, privacy, transparency and safety.

6. Risk Analysis

- **Data quality and model accuracy**

  There is a huge risk on how good the data been feed into the model during training will be, as a model can always be as good as the data been feed to it.

  Mitigation: Regularly update and retrain the model with diverse datasets from different product reviews too.

- **Data security and privacy**

  The project handles user reviews, therefore there is a risk of sensitive information to be exposed. Measures of anonymizing data are be taken into consideration.

  Mitigation: Removal of personal identifiable information from the data before processing. The storage of data securely to prevent unauthorized access to the data. For this project the dataset source will not live in with the system. The dataset will be stored in a completely separate system that has not connection to the model system.

- **Scalability**

This risk may only be gotten from big applications or systems that handle enormous amount of data per second. This risk was also taken into account with the use of C++ which is a very fast and robust language and the design of the system with scalability in mind

Mitigation: conducting stress testing and identify performances bottlenecks.

- **Dependencies mayhem**

  This is really a huge problem even when setting up the development environment. That is why I used docker dev containers to build and find the optimal environment that can run it efficiently

  Mitigation: Documenting the libraries and their dependency versions, building a docker image to hold a snapshot of the dev environment and production environment to run and build the system. Regularly check for updates and dependencies changes.

- **Deployment**

  There are risks during the deployment stage of the model which may require additional resources before it will be ready to serve. Failure to meet up with these requirements can lead to downtime, poor service. This risk can be mitigated with the use of docker for deployment.

  Mitigation: implement redundant server configurations, regularly server checkup and have a backup plan available.

7. Ethics

- **Bias**

  The model may be subject to unintentional biases residing in the data used to train it. Procedures of monitoring biases from the data source are also taken into consideration by regularly addressing and evaluating biases found in training data. [25]

- **Privacy**

  The system will have access to organizations customers or users reviews data. This can raise privacy concerns. The system already has data anonymization of the textual data. [26]

- **Transparency and Consent**

  The users of the system will be informed in full details how the data collected from them either using data source gathering or during predictions are been used and stored.

- **Accountability**

  It should be clear who is responsible for the model's actions and outcomes. The task of correcting any harm caused by the model and the continuous update and monitoring of the model to always stay ethical.

- **Safety**

  İn terms of safety, the system will try it best to ensure it adopts and uses efficient and less power consuming cloud providers and software.

8. Conclusion

Due to its use of C++, this system exhibits expertise in memory management while working with large datasets, data pretreatment, natural language processing, docker usage, and web API development that set it apart from other projects. The goal of this project is to lower the cost of text classification in AI for developers and end users.

8.1 Benefits

Benefits to users

- In order to free up supplier time for other responsibilities, the technology will enable the automatic classification of review data.
- **Real-time feedback:** The system will be able to help business users to identify issues with their products from customers reviews more quickly and react swiftly.
- **Understanding user opinions:** Provide good in-depth knowledge of customer sentiment to help businesses improve their products and services, automating many labours intensive process of analysing text and saving time and resources.

Benefits to me

- *Skill improvements*: The project helps me further develop my skill in text vectorization and classification in a different language, system and development tools.
- *Portfolio improvement*: Can help boost my portfolio and showcase my ability in handling NLP tasks.
- *Problem solving skills improvement:* it introduces me to more complex libraries and compilation techniques, had to solve tasks such as

compiling right and managing sizes, something that is abstracted in other modern languages.

Reason for choosing this project

I chose this project due to my previous knowledge of using Sklearn library in python to build a sentiment model with 82% accuracy result. Remaking this in C++ which doesn't have extensive support for NLP and few libraries for it was also a motivation as I want to broaden my knowledge in using languages and tools sets with virtually almost zero tutorials unlike with python or JavaScript where there are so many numerous tutorials for it.

8.2 Future Works

- İmprove on the number of data set used to train it
- Enable the feature of using it to analyse in real-time video chat
- Introduce more classification features that can be used to monitor brand trends in user chat forums, market research and competitive research
- Content and topic categorization, spam detection and malicious content detection.
- Be an addon to a speech to text system where it will analyse the text transformed by the speech to text system.

9. References

[1] B. Liu, "The Problem of Sentiment Analysis," Cambridge University Press eBooks, june 2015. [Online]. Available: https://www.cambridge.org/core/books/abs/sentiment-analysis/problem-of-sentiment-analysis/A0AFE2C49D72C5914C34DAC763BCD931. [Accessed 17 Dec 2023].

[2] Eden AI, "Eden AI Workflows," Edenai.co, 2023. [Online]. Available: “Eden AI Workflows,” Edenaihttps://www.edenai.co/workflows?referral=sentiment-analysis-vs-custom-text. [Accessed 17 Dec 2023].

[3] Levity, "Levity," Github.com, 04 Jul 2023. [Online]. Available: https://github.com/orgs/levityai/repositories?type=all. [Accessed 17 Dec 2023].

[4] Huggingface, "Huggingface," Huggingface.co, 11 Sep 2023. [Online]. Available: “huggingface (Hugging Face),” Huggingface.co, Sep. 11, 2023. https://huggingface.co/huggingface (accessed Dec. 22, 2023).. [Accessed 17 Dec 2023].

[5] Lexalytics, «Home - Lexalytics,» Lexalytics, 10 Mar 2022. [Çevrimiçi]. Available: https://www.lexalytics.com. [Erişildi: May 10 2024].

[6] Qualtrics, «Qualtrics XM - Experience Management Software,» Qualtrics, 13 Oct 2015. [Çevrimiçi]. Available: https://www.qualtrics.com. [Erişildi: 11 May 2024].

[7] typescriptlang, "JavaScript With Syntax For Types," typescriptlang, 2020. [Online]. Available: https://www.typescriptlang.org/. [Accessed 17 Dec 2023].

[8] R. R. C. e. al., "mlpack 4: a fast, header-only C++ machine learning library," *Journal of Open Source Software*, vol. vol 8 no. 82, no. R. R. Curtin et al., “mlpack 4: a fast, header-only C++ machine learning library,” Journdoi: https://doi.org/10.21105/joss.05026. url: https://arxiv.org/pdf/2302.00820.pdf, p. p. 5026, 2023.

[9] G. Ben-Evgi, "Exploring the Power of uWebSockets: A High-Performance WebSocket Library," Medium, 27 Jun 2023. [Online]. Available: https://ga1.medium.com/exploring-the-power-of-uwebsockets-a-high-performance-websocket-library-83aedc26ae1. [Accessed 17 Dec 2023].

[10] Wikipedia Contributors, "Adobe Illustrator," Wikipedia, 15 Dec 2023. [Online]. Available: https://en.wikipedia.org/wiki/Adobe_Illustrator. [Accessed 17 Dec 2023].

[11] Matplotlib, "Matplotlib — Visualization with Python," Matplotlib, 2023. [Online]. Available: https://matplotlib.org/. [Accessed 17 Dec 2023].

[1 Cmake.org, "CMake - Upgrade Your Software Build System," Cmake.org, 2023.
2] [Online]. Available: https://cmake.org/. [Accessed 17 Dec 2023].

[1 J. Leskovec, «SNAP: Web data: Amazon reviews,» Stanford.edu, 2024. [Çevrimiçi].
3] Available: https://snap.stanford.edu/data/web-Amazon-links.html. [Erişildi: 1 5 2024].

[1 «Porter Stemming Algorithm,» Tartarus.org, 2018. [Çevrimiçi]. Available:
4] https://tartarus.org/martin/PorterStemmer/. [Erişildi: 1 5 2024].

[1 D. A. A. A. a. A. F. A. Saif Safaa Shaker, «Feature Extraction based Text Classification:
5] A review,» ResearchGate, 26 May 2022. [Çevrimiçi]. Available:
https://www.researchgate.net/publication/361226607_Feature_Extraction_based_Text_Classification_A_review. [Erişildi: 4 May 2024].

[1 J. Brownlee, «A Gentle Introduction to the Bag-of-Words Model,»
6] machinelearningmastery.com, 08 Oct 2017. [Çevrimiçi]. Available:
https://machinelearningmastery.com/gentle-introduction-bag-words-model/. [Erişildi: 04 May 2024].

[1 S. Chouksey, «Demonstrating Calculation of TF-IDF From Sklearn,» medium, 21 April
7] 2020. [Çevrimiçi]. Available: https://medium.com/analytics-vidhya/demonstrating-calculation-of-tf-idf-from-sklearn-4f9526e7e78b. [Erişildi: 08 May 2024].

[1 M. McGregor, «SVM Machine Learning Tutorial – What is the Support Vector Machine
8] Algorithm, Explained with Code Examples,» freeCodeCamp.org, July 2020.
[Çevrimiçi]. Available: https://www.freecodecamp.org/news/svm-machine-learning-tutorial-what-is-the-support-vector-machine-algorithm-explained-with-code-examples/. [Erişildi: 09 May 2024].

[1 GeeksforGeeks, «Support Vector Machine (SVM) Algorithm,» GeeksforGeeks, 20 Jan
9] 2021. [Çevrimiçi]. Available: https://www.geeksforgeeks.org/support-vector-machine-algorithm/. [Erişildi: 08 May 2024].

[2 Z. Zhang, «Naive Bayes Explained - Towards Data Science,» Medium, 14 Aug 2019.
0] [Çevrimiçi]. Available: https://towardsdatascience.com/naive-bayes-explained-9d2b96f4a9c0. [Erişildi: 07 May 2024].

[2 «How Naive Bayes Algorithm Works? (with example and full code) | ML+,» Machine
1] Learning Plus, 04 Nov 2018. [Çevrimiçi]. Available:
https://www.machinelearningplus.com/predictive-modeling/how-naive-bayes-algorithm-works-with-example-and-full-code/. [Erişildi: 10 May 2024].

[2 R. MELTZER, «What is Random Forest? [Beginner’s Guide + Examples],»
2] CareerFoundry, 31 Aug 2023. [Çevrimiçi]. Available:
https://careerfoundry.com/en/blog/data-analytics/what-is-random-forest/. [Erişildi: 08 May 2024].

[23] GeeksforGeeks, «Machine Learning Model Evaluation,» GeeksforGeeks, 07 Jan 2023. [Çevrimiçi]. Available: https://www.geeksforgeeks.org/machine-learning-model-evaluation/. [Erişildi: 08 May 2024].

[24] GeeksforGeeks, «Introduction to Software Engineering Software Engineering,» GeeksforGeeks, 12 Oct 2018. [Çevrimiçi]. Available: https://www.geeksforgeeks.org/software-engineering-introduction-to-software-engineering/. [Erişildi: 08 May 2024].

[25] S. M. Mohammad, "Ethics Sheet for Automatic Emotion Recognition and Sentiment Analysis," in *Computational Linguistics*, vol. vol 48, doi.org, 2022, p. pp. 239–278.

[26] S. M. Mohammad, "Ethics Sheet for Automatic Emotion Recognition and Sentiment Analysis," arXiv.org, 2021. [Online]. Available: https://arxiv.org/abs/2109.08256. [Accessed 17 Dec 2023].

[27] Python-engineer.com, «chatbot-pytorch,» Python-engineer.com, 14 June 2020. [Çevrimiçi]. Available: https://www.python-engineer.com/img/2020-06-14-chatbot-pytorch/bag.png. [Erişildi: 08 May 2024].