Aston University

CS4700: Research Project

Artificial Intelligence based privacy preserving counselling system for students

Author: Daniel Yaroson
Student Number: 17001759
Supervisor: Nitin Naik
Submission Date: 04/07/2022
Master of Artificial intelligence with business strategy

Table of Contents

1 Abstract

There are many pressures that affect students during their time at university. These pressures can act as crouches enabling students to perform to their best ability and can have serious detrimental effects on their emotional and phycological wellbeing. To help university students with their issues, most universities have third parties such as supervisors or head of departments in which students can disclose their problems seeking a classification of their problem and eventual appointment of a professional to help them. However, these problems may contain sensitive information that student might not want to share to those who might not be able to help. To initially seek help students usually use emails or book appointments with third parties as a form of communication, but responses and booking times are rarely immediate. There is also the possibility of students having their queries wrongly classified. To try and tackle these issues this project proposes an artificial intelligence-based privacy preserving counselling system to classify student's queries. The cybersecurity and cryptography techniques federated learning and encryption, respectively, were leveraged to apply the privacy preserving feature of the system. The machine learning techniques neural networks, decision trees and the genetic algorithm were used to attempt to accurately classify student's queries. Finally, a business viewpoint was applied to assess the business value of the system to Aston university.

2 Definitions

To create a better understanding and improved context of the literature review and the thesis, technical terms used will be defined:

Artificial intelligence - The first original definition coined by John McCarthy who some recognise to be the father of artificial intelligence was stated to be ‘The science and engineering of making intelligent machines. Over the course of its inception artificial intelligence has adopted many definitions that slightly vary but ultimately all seek to attain one common goal, to mimic human intelligence and human-like thought processes such as learning, reasoning and self-correction (Kok et al., 2009).

Multi-class classification – A categorical problem from the machine learning subset which is a derivative of artificial intelligence. It seeks to assign labels to data when there are more than 2 classes that the data can fall under. For example, classifying a set of images of fruits which may be grapes, orangs, or pears.

Cyber security – A widely interdisciplinary, touching on main subjects such as engineering, technology, computer science, security and defence. However, to lesser degrees subjects such as law, healthcare, public administration, accounting, management, sociology, psychology, and education also have ties to the cyber security which has provided the subject with many definitions. Dan Craigen released a paper that acknowledges cyber securities multiple links and proposed a definition, ‘Cybersecurity is the organization and collection of resources, processes, and structures used to protect cyberspace and cyberspace-enabled systems from occurrences that misalign de jure from de facto property rights’ (Craigen et al, 2014).

Cryptography - A method used to secure information. It is the science of securing information in such a way that it can only be read or understood by the intended recipient. The most common method of cryptography is encryption. It involves transforming information into a form that is unreadable or unintelligible without a specific key or code.

Federated learning - An approach to cybersecurity that uses the results of previous machine learning runs to update a model and provide improved predictions on future data. Most notably, it can improve the performance of a model by combining the results of several learners into a single, improved model. It is the process of training a model and then distributing copies of the model to a pool of users to use them on their data, where each user can contribute only a small amount of learning. When users make contributions, they trust the model enough to use it on their own data, without sharing it with anyone else. The primary purpose is to provide privacy for a training set through distributed computation and to provide scalability for large training sets. Federated learning is most often used in conjunction with machine learning techniques such as randomized linear projection to achieve efficient distributed training.

Homomorphic encryption – A form of public-key encryption that allows an encryption function to process the same input data, and produce the same output data, whether the input data, the encryption function or the output is stored in a public or private manner. The term “homomorphism” refers to the fact that the same mathematical transformation can be applied to both plaintext and ciphertext, so long as they are both encrypted with the same key.

Differential privacy - The theory of preserving the privacy of personal information without jeopardizing the utility of the data for other purposes, such as research, analytics, and advertisement. It does so by requiring that the data provider not disclose information about the data subjects except under certain very limited conditions.

Multi-party computation - Refers to the use of a decentralized system to accomplish a joint task, where the performance of each member is not affected by what the others are doing, nor by whether they do their part. Multi-party computation differs from common multi-party computation (MPC) in that the cryptographic computations are not required to be performed simultaneously, but are instead performed in a sequence of steps. For example, in MPC, data from each party is first combined before being sent to all parties. In MPC, the parties do

3 Introduction

The university experience is encompassed by a series of unique and personalised challenges whereby students face different forms of academic and social development. Such environments can impose different pressures on its inhabitants; for some students these pressures can enable them to thrive and perform to a high standard. However, this is not the case for all students and the pressures of university can have detrimental effects on both mental and physical health affecting their general wellbeing, causing students to underperform. Extensive amounts of research have found that poor mental health is a rising issue amongst university students. The HARM network calculated that in 2021 more 185,000 students and staff members suffer from domestic issues every year. There has been an 8% increase of first-degree course students from 2019/20 to 2020/21 (Hesa.ac.uk, 2020). In 2021 it was reported that there has been a 450% increase in student mental health (UCAS, 2021). These statistics suggest that with an increase of students and a drastic increase in mental health and domestic issues, finding ways to aid students to seek help and categorize their problems is of the up most importance to better their university experience. Multiple studies have identified a trend of an increase in prevalence of mental health conditions such as anxiety, stress and depression amongst university student when compared to the general population (Mofatteh, 2021). There are various factors that contribute to the challenges students face during their university years, some of which include academic difficulties, domestic, social, financial and linguistic problems. In forming healthier educational and social environments for students, some of these pressures can be alleviated which in turn can provide students with greater chances of educational success. Alleviating such pressures equips students with greater chances of educational success; this has been proven by research which indicates that there is a reciprocal relationship between student health and educational achievement (El Ansari & Stock, 2010). Universities have a duty of care to students such that there should be implementations that aim to eliminate variables that hinder student health. Currently this is carried out by correct allocations of professionals and or counsellors that fit the classification of student's queries and worries.

Most universities have assigned professionals and departments that students are able to rely on to discuss their concerns and queries in hopes of improving their situations. However, a deficit can be identified in this process, such that at times students themselves are unable to classify their own queries and therefore find it difficult to ask for help. In addition to this, students may find difficulty in identifying the type of aid they require, or they are deterred by prolonged response times due to third party communications with supervisors or head of departments. Another disadvantage of this process is that students may not wish to disclose their queries to various personnel or anyone at all due to the fact that their queries may contain sensitive information. Furthermore, students may require the expertise of more than one counselling professional which can be difficult to identify. Upholding privacy standards in a counselling situation is of the up most importance to universities and students need to be reassured that their personal information is kept private and preserved.

Data privacy is a controversial, yet relevant topic in today's society. In most jurisdictions, privacy is considered a fundamental human right, and data protection laws exist to safeguard the general public's personal information (Data Protection Act 2018 CHAPTER 12, 2018). With the emergence of big data and large businesses utilising the power of data analytics (such as Facebook, Instagram, Netflix, Snapchat, Spotify, Twitter and more) the ethics of how data is used, collected, and shared has regularly been discussed. The definition of data privacy is the ability of a person to determine for themselves when, how, and to what extent personal information about them is shared with or communicated to others (Cloudflare, 2022). In regard to Aston university, they too must comply with data protection law and particularly the Data Protection, Privacy and Electronic Communications Regulations 2019 ("the UK GDPR"). This legislation provides issuers of data the right to request restrictions of the processing of their data. Failure to comply with legislation can result in financial penalties against the university. Currently when students seek out counselling help at the university, they often contact an intermediate party such as a personal tutor, receptionist, supervisor or head of department. Once the query is classified, the relevant party will relay the student's situation to a designated professional or counsellor who specialises in the aid that the student requires. This transmission of information whereby an intermediate party can view the potential sensitive information could make students reluctant to seek help as it entails them sharing sensitive information to those it may not concern or those who won't be able to directly help with their issue. Moreover, the inclusion of an intermediate party could potentially increase the time taken for a professional/counsellor to be allocated. When communicating via email, students rarely receive immediate responses from staff members and when booking appointments using Aston web appointment scheduling system, students must accommodate to staff members calendar slots. Whether it be communication via email or an appointment it is highly unlikely that student will receive immediate responses. There is also a potential for the intermediate party to wrongly classify the student's situation and guide the student to the wrong professional/counsellor. The process of classifying queries and adequate allocation of help for students currently has some drawbacks, however, there is the potential for this to be automated and add layers of privacy using technology.

To tackle the issues discussed previously, an Artificial intelligence (AI) based privacy preserving counselling system has been suggested in this paper. A novel app which students can use on their devices will be built to classify student's queries. This has been carried out through the use of machine learning and cyber security techniques. To uphold the confidentiality of university students, comply with data protection regulations and reduce the time for students to receive aid, the machine learning and cybersecurity technique federated learning will be used to initially leverage privacy. This will be done by localising client's information to students' devices and updating the weights of a machine learning model to a server where the model is hosted. Although this method brings about a degree of privacy it still has its drawbacks such as potential accessibility to hackers. Therefore, to ensure that privacy is as secure as possible, other cybersecurity technique will be researched to see if they can be integrated with federated learning, this will be further discussed in the literature review. The additional cybersecurity techniques that will be explored are homomorphic encryption (Yi, Paulet and Bertino, 2014), differential privacy (Dwork, 2021) and multi-party computation (Evans, Kolesnikov and Rosulek, 2018).

Regarding the issue of classification of student’s queries, a range of AI methods will be tested to ensure that accuracy is prioritised. As this is a multi-classification problem, the techniques that will be explored include decision trees, neural networks and the genetic algorithm. Research will be done on each method to see which one is more suited to sufficiently classify the inputted data. A user-friendly interface (app) will be made for student to access and input their data to be classified.

4 Literature Review

4.1 Introduction

The prevalent expansion of technology and data have meant that its application is seen in countless work sectors. Such that that there has been an increasing demand for the effective establishment of ethics, to allow data to be imbedded in social enterprises. Cybersecurity not only is a preventative measure used to reduce the risk of cyber-attacks on organisations, but jointly adopts and tackles aspects of the general ethics issues faced by the technology sector such as client data confidentiality. This literature review will dissect the cybersecurity space and will highlight the application of ethics within it. A significant focus that will be highlighted within the cybersecurity will be the application of field known as federated learning. The aim will be to detail its uses, advantages, and disadvantages and highlight its superior compatibility alongside other cybersecurity techniques. This project aims to solve the classification deficit associated with student queries during their university careers. This can be carried out using a variety of AI techniques which involve text classification. It is for this reason this review will also analyse the different artificial intelligence models available for text classification. The appointment of a text classification algorithm is dependent on the use case, as this project seeks to classify user queries, a focus on this aspect of algorithms will be established to see what has been successful in the field and what can potentially be done to improve these algorithms.

4.2 Background: Importance of data and the need for cybersecurity to protect client-privacy

There is an undeniable need for data in today's world, and it's becoming more relevant as society continues to adopt technology. The ability to gather, analyse, and use data to make informed decisions is a powerful tool that can improve lives, communities, and the world. In the case of businesses, data-driven organizations are 23 times more likely to acquire customers, 6 times more likely to preserve their customers and 19 times more likely to be profitable. Figures such as these suggest that data is a valuable resource that can be used as a source of profit. From 2002 to 2022 the volume of data/information created, captured, copied, and consumed worldwide has increased by 4750% to reach a current storage value of 97 zettabytes (Statista, 2020) (Berisha and Sc, 2021). The rate in which data is being generated is exponential, creating a need to harness robust information. Companies are spending over $10 billion on third-party audience data as of 2017 (Criteo, 2017), this figure suggest how big of an asset data is.

Data is a sensitive topic, it can reveal a great deal about an individual or an organisation. When working with data, it's important to keep in mind that the information being handled can be sensitive. The same rules that apply to paper documents also apply to digital information. Protecting client information is of the upmost importance, this rule also applies for the data itself which is being handled. It should be noted that by protecting customers personal information used for data

analysis, this also protects the organizations that handles the data. With the increased use of data and technology this also brings about an increased risk of cyber threats which amplifies the threat towards client data and organizations. A technological space that best demonstrates this risk to data is cloud technology which is one of the fastest rising sections of IT. Cloud technology has many benefits that justify its use, that of which include cost saving, efficiency, unlimited data access and the use as a service instead of a hardware, saving computer space and storage (Sether, 2016). However, the implications of cloud data lays within its beneficial characteristics, its storage-based services used over the internet make it vulnerable to data loss through theft, leakage, account or service hijacking and cloud downtime. In addition to this,
Customers will have minimal control over data, applications, and services. These disadvantages make the need for client privacy crucial. Significant data breaches that have occurred in the past decade include Yahoo with 3 billion records stolen in 2013, JP Morgan & Chase with 86 million records stolen in 2014, Uber with 57 million records stollen in 2017 and many others. Data and personal information are stolen regularly from data centres and organisations and are being sold to malicious users on the dark web (Stack, 2017). The price of data varies depending on the nature of data which is what drives the demand to illegally obtain data. This requires methods to protect against such threats. Common methods that have been tried to combat these breaches include firewalls and network security tools, However, these techniques are not always sufficient in protecting against internal threats within an organisation as attackers will have access to relevant system information. Thus, making it difficult to block the misuse of information (Morganclaypool.com, 2012).

In 2021 there were over 10 billion connected to the internet, this has rapidly increased with the use of the internet in vehicles and household appliances (Dataprot, 2022). However, in 2020 data breaches exposed 36 billion records in the first half of the year which is the highest ever recorded (Databasix UK Ltd, 2022). An increase in users and devices has led to the formation of more data, and as expressed previously methods such as firewalls and mundane network security can only help protect from threats to a certain degree. The fields cybersecurity and cryptography (conceived in the 1970's) are established attempts used to bolster the attempt to tackle cyber threats. Cybersecurity and cryptography now, have become critical components of organisations business strategies. Businesses can invest in the best equipment, hire the most skilled employees, and develop the most sophisticated processes, but if cybersecurity is not prioritized, businesses can face major data theft in today's digital world, cybersecurity has become a major priority for companies of all sizes and in all industries. This is due to the effect that a data breach or a cybersecurity incident can have, which can be devastating for businesses and their customers. In 2019 Forbes magazine released an article stating that a net loss of $2.1 trillion from the economy had been recorded due to the lack of cybersecurity (Apurva et al., 2017).

4.3 Background: Importance of artificial intelligence

AI is a derivative of computer science that harnesses the power of data to perform tasks that require human intelligence, such as visual perception, speech recognition, decision-making, translation, optimisation, and prediction. Machines that perform these tasks well can improve the efficiency of human activities and generate new ideas and products. The field of AI is particularly novel undergoing its conception in 1955, however, since then the field has undergone several waves of changes and has contributed to tremendous strides in technology, business, healthcare, and overall human lifestyle.

The increase of the usage of AI in recent years has allowed many businesses to flourish and rapidly increase their revenue. In 2015, 10% of business reported that they were either currently using artificial intelligence technology or had started developing plans to use the technology in the future (Evans and Heiman, 2022). This percentage increased to 37% in 2019, which means 1 in every 3 business either use artificial intelligence or have plans to use it (Evans and Heiman, 2022). It was then reported that the global market size of artificial intelligence in 2021 was $93.5 billion (Grandviewresearch.com, 2022), with expected growth to increase by 38.1% in 2030 (Gartner, 2019). Such large figures in the early commercial usage of artificial intelligence suggest the potential to generate profit and use cases in the field. Big named businesses that have adopted AI into their business plan include google, apple, tesla, Microsoft, amazon etc. For example, a popular google use case of AI is the algorithm embedded in the google search engine which predicts the next word that a user might type. In the case of Tesla, AI reinforcement learning is used to allow cars to self-drive.

Along with its business influence, AI also has government and societal benefits which have increased its importance and popularity. AI has the capability to aid in natural disaster prediction, development of hazard maps for the detection of real-life events, provision of situational awareness and decision support (K Monique et al, 2022). This has allowed analyst to adequately prepare and create extra time to allow civilians to escape danger zones, saving many lives. In Mexico’s Colima, data from River Core (Mendoza-Cano et al., 2021), sensors and weather stations were used to train AI machine learning models that were then used to detect flash floods in the Guadalajara metropolitan area. Detection of these floods allowed for residents to evacuate to safety before the flood took place, saving hundreds of lives. In China, AI has been adopted by the Chinese government to detect those who break the law through facial recognition. For instance, those who illegally cross the road In China can be easily identified and prosecuted by facial recognition in security cameras. This technology greatly reduces the number of crimes committed acting as a deterrent due to the simplicity of quick identification.

The medical sector has been one of the most promising fields for AI research. AI has already been used to improve medical care by diagnosing and treating patients, predicting disease, and providing tailored medical advice and has the potential to develops these areas further. In the past few years, the medical industry has seen several new medical apps and services powered by AI. For example, companies such as HealthTap offer online medical consultations with doctors, using machine learning algorithms to process patient information and provide tailored medical advice. The National Institute of Health has also developed an app called AiCure which monitors patient's medication usage via smartphone webcam access, lowering nonadherence rates (Labovitz et al., 2017). Research has been done by a study to see how AI can assist radiology (Mayo and Leung, 2018). The research detailed, that by labelling abnormal exams and identifying quick negatives with the aid of AI, computed topographies, X-rays and magnetic resonance images can quickly be assessed. This increases medical efficiency, saves time, and uses less available human resources.

Finally, a recent use case that justifies AI importance in the medical field and its contribution to wider society, is the application of the technology during the COVID-19 pandemic. In 2020 an AI based algorithm accurately detected 68% of COVID-19 positive cases in a dataset of 25 patients classified as negative instances by care professionals (Mei et al., 2020). This success saw the use of this technology in other countries such as Germany to aid the detection of the virus. Moreover, AI in the healthcare market saw a growth increase of 167.1% from 2019 to 2021 (Grandviewresearch.com, 2022). It is now estimated that a year-on-year growth between 34.9% to 48.0% in the next 5 years is expected. (Grandviewresearch.com, 2022).

It can now clearly be seen that the potential, popularity, and importance of AI is enormous, but it is to be approached with caution. The field of artificial intelligence is rapidly evolving, and it’s difficult to predict how organizations will use the technology. Moreover, as this technology utilises sensitive data, there is a need to question how this data is gathered, used, and protected.

4.4 Background: The need for student privacy concerning sensitive counselling data

In the modern world, it is becoming increasingly clear that privacy is an important topic. For this reason, many institutions have taken active steps to ensure the security and confidentiality of their data. A field in which the protection data is vital is education. The current state of the educational system requires a large amount of student data to be processed to provide the best possible services to students. Unfortunately, this often makes these organisations susceptible to data theft. In 2020 a survey conducted by the Information Commissioner’s Office found that 54% of UK universities reported data breaches (Technologyonecorp.co.uk, 2021). It has also been recorded that out of 86 Universities that responded to a Freedom of Information request from security firm Redscan, the majority recognised substantial flaws in their ability to protect against data breaches (The state of cyber security across UK universities An analysis of Freedom of Information requests, 2020).

Data privacy is an issue of increasing significance to businesses and organizations of all forms and volumes. Companies are increasingly aware of the risks associated with failing to protect sensitive data, and the potential reputational damage that can be caused by a security breach. The same is true for educational institutions. In an age of increasing paranoia about data privacy, universities need to take steps to ensure that student data is protected. The sensitivity of educational data and the data protection laws that universities must abide by, makes it especially important to ensure that it is properly safeguarded.

4.5 Main Body of the Literature Review

Although the field of artificial intelligence is relatively new, it has been extremely successful in recent years. These recent successes and application of artificial intelligence stems from the strength of its foundations and original curators. In 1955 John McCarthy coined the term artificial intelligence during the first academic conference on the field (Smith et al., 2006). This conference gathered some of the world's leading computer scientists, where McCarthy defined Artificial intelligence as: 'the science and engineering of creating highly intelligent computing devices' (Allganize, 2020). With a set definition and an established direction, this would provoke scientist to start spending time, money and resources on research in the field. Herbert Simon and Allen Newell developed the first artificial intelligence program. The program was called the Logic theorist and was designed to replicate human intellect to proof mathematical thermos in symbolic logic. This breakthrough managed to adhere to the definition of artificial intelligence, successfully reproducing functions of the human brain to complete a complex task. The results proved to be proficient, producing results that saw 38 out of 52 theorems proofed. The published paper had great significance at the time, as it would cause a spiralling affect that would see various artificial intelligence publication during the later stages of the 1950's and so on until present. The development of the first artificial intelligence program also led to the development of many computer languages such as FORTRAN, LISP and COBOL. As algorithms and programmes were being developed, researchers needed more efficient ways to transcribe their ideas on to computers. These computer languages would allow researchers to establish better communication between computers and users. With a clear definition of the field and new programming languages to instruct interfaces, the field would go on to produce revolutionary artificial intelligence technology.

As artificial intelligence is a general-purpose concept based on the ability of human intelligence, various tasks can be carried out utilising this field of study. Such tasks include natural language processing, reasoning, knowledge representation, planning, learning, optimisation, and perception. For this project the literature review will focus on natural language processing. Natural language processing is a theoretical range of computational techniques for analysing and representing text at different levels of linguistic analysis to achieve human like language processing for various tasks and applications (Liddy, 2001). The most common applications of natural language processing include information retrieval, information extraction, question-answering, summarization, machine translation and text classification etc (Liddy, 2001). A particular focus will be placed on the application of text classification within this field.

The most common techniques used for text classification are neural networks, Bayes classifier, association rule mining, genetic algorithm and decision trees (Kamruzzaman, 2010). These techniques have been studied and tested to be used to categorize text. However, depending on the use case different techniques yield better results. For example, neural networks, particularly recursive neural networks, have been ideal for text classification and natural language processing as they can learn tree like structures. As grammar often follows a tree-like structure, this allows the recursive neural networks to logically separate sentences and excel in natural language parsing. In addition, neural networks provide excellent results to use cases that express many features in the initial data sets as they can produce billions of neural pathways which lead to accurate results. However, in previous work neural network frameworks have learned based on single-task supervised objectives, which has made models suffer from insufficient training data (Liu et al., 2016). This has been seen in deep neural networks also as generally they require a large set of data and corpus due to the large number of parameters within the architecture of the model. To combat this issue researchers have utilised an unsupervised pre-training phase which has proved to increase the accuracy of various natural language processing tasks

(Collobert et al., 2011). An unsupervised pre-training phase can improve efficiency of performance however it cannot fully optimise the task. Motivated by Caruana (1997), the success of multi-task learning in the past, conducted research on the utilisation of multi-task learning to control supervised data from many related tasks (Liu et al., 2016). Multi-task learning uses the correlation between associated tasks to improve classification by learning tasks in parallel. The paper produced three recursive neural networks-based architectures that utilised multi-task learning and found that the models improved the performances of a group of related tasks by exploring common features. Compared to state of the art neural models such as NBO, MV-RNN, RNTN, DCNN, PV and Tree-LSTM, the multi-task recursive neural networks out performed on two out of four datasets.

The genetic algorithm is a machine learning technique that can be used to solve complex problems. The search algorithm was introduced by John Holland in the 1970’s (Alsaleh, Larabi-Marie-Sainte 2021). It is based on the principles of Charles Darwin’s evolution and genetics theory, where a population of possible solutions is iteratively improved over time, reminiscent to how favourable characteristics are passed on from parent to offspring in genetics (Whitley, 1994). Unlike other machine learning algorithms, genetic algorithms are black box: they do not have to predefine the correct answer; simply set up the criteria and the system finds the best solution. The genetic algorithm has been used to solve a wide range of problems, including machine learning, classification, search, and optimization. It can also be applied in a variety of ways depending on the type of problem being solved and can also be merged with other techniques such as neural networks. For example, a research paper was completed to optimise the weights of a convolutional neural network by the genetic algorithm (Ijjina and Chalavadi, 2016). By applying the genetic algorithm to the weights of the convolutional neural network, the overall accuracy of the neural network increased and reached an accuracy of 96.8% when tested on a YouTube video dataset. Another example of the genetic algorithm being used alongside a convolutional neural network is for crack detection in images (Ouellette et al., 2004). The study yielded an accuracy of 92.3% which when compared to backpropagation to renew the weights, was not statistically better, however, the study does mention implementation of the genetic algorithm was simpler.

Decision trees are a type of machine learning model that can be used to make predictions in a wide variety of contexts. They emerged as a way of representing the logic of human decisions and have since been used to solve a wide range of problems. They can be used to identify the best course of action in a given situation, or to predict the outcome of a course of action under a set of circumstances. They are also used to classify data, to make sense of complex collections of information. In 1963 an analytical algorithm called AID was developed which would later be known as the first decision tree regression algorithm (Ali, 1975). The focus of the research was to find relationships between age, race, education, occupation, and yearly income of the population. In 1972 the first classification decision tree was created to extend the AID algorithm to account for categorical outcomes (Ritschard, 2010). This was established by using a theta criterion which is a constraint that sorts sentences into grammatical and ungrammatical classes based on c-selection and s-selection. Up until 1986 all decision tree classification algorithms were binary trees, meaning they only had two answers. Researchers found that for problems that required multiple classes, the proposed algorithm was limited. John Ross Quinlan proposed a new model called Iterative Dichotomiser 3 that included trees with multiple branches, consequently multiple answers (Quinlan, 1986). With the addition of an impurity criterion called gain ratio, decision trees were able to match up to the capabilities of neural networks in terms of classification. Due to its tree like architecture, decision trees provide a more detailed view on how a system is likely to evolve and has an advantage over black box models such as neural networks and the genetic algorithm as it has greater comprehensibility because of their tree like structure (Kotsiantis, 2013). The visual representation of decision trees make interpretation and understanding of why a data point was classified into a certain category, easier compared to the genetic algorithm and neural networks. However, it has been noted that decision trees with multiple classifiers it can be more difficult for non-expert users to understand the underlying rational process leading to a decision or classification (Kotsiantis, 2011). Decision trees are also favourable over other machine learning methods as they learn at exceptional rates (Podgorelec, 2002). Unlike neural networks decision trees are not computing hundreds of thousands of calculations on weights and inputs, this vastly reduces the computational effort required from decision trees allowing for faster results. However, a fault of decision trees is that they have difficulty dealing with noise in training data (Podgorelec, 2002). Noise in data can confuse a decision tree and produce overfitting as the algorithm attempts to fit each data point, including noisy data. Pre and post pruning have been established to try and mitigate the noise during the tree induction phase however, noise in data is still an issue for decision trees.

As the world becomes more connected, cybersecurity has become an increasingly important field. Cybersecurity refers to the methods used to protect computer systems and data against unwanted interference, including the detection and prevention of unwanted intrusions and the detection of potential security breaches. It is a broad field that can be broken down into two main categories: reactive and proactive cybersecurity. Reactive cybersecurity involves the detection and prevention of unwanted intrusions after they have already occurred, while proactive cybersecurity involves the detection of potential security breaches before they have a chance to occur. Although research has been conducted to distinguish these categories and create combatants for problems, challenges from a decade ago have evolved alongside the growth of cyber security. Therefore, researchers in the field have had to constantly innovate new ideas in the defence against cyber threats. From the conception of the field in the early 1960’s many different forms of cybersecurity have taken form, however, there are a few that have constantly provided to aid against cyber-attacks and some new techniques that have potential to provide competent defence.

As described previously in the definitions section, federated learning is the process of training a machine learning model with data from multiple distributed sources, without the need for having to manage the data in one centralized location. The term federated learning was coined in 2016 by google (Mcmahan et al., 2016) and since then a plethora of adaptations have been published. The original goal of federated learning was to create a way to better the users experience, with language and image models, that use the user’s data to train the models, but preserve the data by keeping it locally on the user’s device. The results of the paper proved that such a technology was attainable and was able to produce high quality models on a variety of architectures: multi-layer perceptron, convolutional neural networks, two-layer character LSTM, and a large-scale word-level LSTM. As mentioned, federated learning was originally used on a linear model (perceptron) and neural networks, however, further research has reviled that federated learning is also compatible with tree models (Zhang et al., 2021). With the capability to allow various architectures of models to benefit from federated learning, this provides more ways to preserve data on devices, ultimately benefiting the user. The privacy aspect of federated learning has helped to give the cybersecurity technique notoriety. Allowing data to stay locally on user's device has dramatically improved privacy preservation, however, this has not eradicated the issue. It is still possible to access the local data, this can be a concern for customers and business.

There are two main threats that could compromise the privacy of a federated learning application: inference during the learning process and inference over the outputs (Truex et al., 2019). Both threats are susceptible to two types of attackers, insiders, who include participants of the federated learning such as data holders, and outsiders which include users of the final model and foreign entities. It is possible to extract data during the learning process from the parameters of the model that are used to train the model based on the user's data. Whilst knowing the possibility of data breaches, the paper suggests two main ways to combat this problem. Sharing the bare minimum of information required to update the generic model at the server and if a neural network is used, making it as complex as possible so that it's difficult to use the available gradients to extract information about the training samples. These solutions are possible and will produce some success but are not optimal.

Recent cybersecurity methods have been conducted and have provided additional solutions to the privacy preserving problem. A recent method used to prevent information leakage is the collaboration of differential privacy and federated learning (Wei et al., 2020). Differential privacy is a system that allows for data handlers to obtain useful information from data that contains sensitive information without divulging the personal identification about individuals by introducing an element of noise to the data (Laplace distribution). It allows data experts to execute all possible and useful statistical analysis without identifying any personal information. A study conducted by Wei et al (2020) found that by adding differential privacy to federated learning, noise is added to the parameters that are sent to the client making it difficult to access their data. It should be noted that differential privacy works best on large datasets as the applied noise will have less of an effect on the data. Utilising a small data set will significantly change the degree of the data, making the final model inaccurate. Whilst successfully being able to implement simulated differential privacy, the study was also able to come to three important conclusions. The first being trade-off between convergence and privacy protection is present as increasing convergence performance lowers protection levels. The second being, given a fixed privacy protection level increasing the number of client participants can increase convergence performance. The third being there is an ideal amount of maximum aggregation times in terms of convergence performance for a given protection level. These conclusions allow for further research to be done to optimise both convergence performance and privacy protection.

Another method that has been studied to increase privacy preservation is homomorphic encryption. As mentioned in the definitions section, homomorphic encryption is a cybersecurity and machine learning technique that allows computation on encrypted data without decrypting it first. Thus, leaving the resulting computation left in an encrypted form which, when decrypted, result in an identical output to that produced, had the operations been performed on the unencrypted data. As gradients and parameters have the potential to leak and expose personal information, homomorphic encryption can help protect against this by encrypting the model's calculations whilst preserving the training results of the model. Le Triou Phong et al (2018) released a paper that utilised additive homomorphic encryption to protect deep learning gradients from leaking within a cloud server (Phong et al., 2018). In addition to leveraging a cybersecurity preventative technique, the research was also able to maintain deep learning accuracy. In a paper released by Cheng et al (2019), entity alignment technology was used to obtain common data to build a decision tree model and used additive homomorphic encryption to protect the parameters of the model (Cheng et al., 2019). In a paper released by Liu et al (2016), a federated learning framework for transfer learning was proposed whilst also using additive homomorphic encryption to preserve privacy. These papers show the progression of Google's original federated learning idea, applying homomorphic encryption to preserve a variety of private data.

A separate cybersecurity method that has been used to ensure privacy is multi-party computation. Multi-party computation is a protocol that enables knowledge creation through a function whilst not disclosing the underlying data. Data contributed by the participating parties is kept private during the data contribution process. All parties only learn from the output. Multi-party computations are not utilized as much as other cybersecurity techniques to complement federated learning. Similar to homomorphic encryption many computation and communication resources are used (Li et al., 2021). It has been established that some ideas are too computationally costly or are costly in terms of communication that makes it less favourable (Bonawitz et al., 2017) (Phong et al., 2018). In a paper written by Runhua Xu (2019), the complication of communication costs was addressed by using a novel privacy preserving federated learning approach, employing a secure multi-party computation protocol based on functional encryption called HybridAlpha (Xu et al., 2019). By using differential privacy and a protocol from a multi-input functional encryption scheme, encryption speed compared to adapted multi-party computations was increased by 90%. Decryption speed increased by 70%. Overall encryption and decryption speeds were increased allowing for faster communication between server and client, thus reducing communication costs. Within research done in 2019, a federated learning model was combined with multi-party computation and differential privacy to preserve the integrity of the model and to protect against extraction attacks and collusion threats (Truex et al., 2019). Whilst keeping the data disclosed, multi-party computation also slightly reduces the noise provided by differential privacy, improving the accuracy that otherwise would be hindered. This is an example of two cybersecurity techniques used to make up for the trade-off in federated learning and for each other.

4.6 Conclusion

Data is an important resource that holds the promise of greater social and economic value in today's society and in the future. When used in collaboration with machine learning, with techniques such as neural networks and decision trees, the potential to extrapolate, predict, classify and optimise data can benefit various sectors such as business, medicine, education and governmental bodies. Regrading text classification, the most successful machine learning models are neural networks, Bayes classifier, association rule mining, genetic algorithm and decision trees. As the value of data rises, so does threats that can lead to sensitive data being stolen and misused. To protect data and preserve privacy of such data, it is necessary to use cybersecurity techniques to add layers of protection. Federated learning is a popular cybersecurity technique as it adds a layer of protection to client’s data by allowing data to stay locally on a device instead of to a centralised server. Moreover, it is compatible with machine learning techniques such as neural networks, decision trees and the genetic algorithm. Although federated learning adds a degree of protection against threats through decentralization, there is still a degree of centralization that can lead to breaches and allow data to be exposed. Recent studies have shown that to make up for federated learning downfalls, it is possible to combine multiple cybersecurity techniques such as differential privacy, homomorphic encryption and multi-party computation.

5 Methodology

5.1 Introduction

The aim of this project is to create an AI based privacy preserving counselling app for university students. While the app is developed, this project also seeks to confirm if a cybersecurity and cryptography methods can be leveraged to maximise privacy and protect against cyber threats, reduce the time it takes to classify students queries and to identify the optimal method for text classification. This entails producing a machine learning model that can adequately classify university students' queries, producing an app interface that is usable for students and using the python programming language to code cybersecurity and cryptography techniques. As the machine learning model is trying to achieve text classification, the data used to accomplish this is qualitative. The methodology of this project was based upon a federated learning client-server application (Gad A, 2020). In this project the human-app interface and the machine learning model were adapted to fit the context of student query classification.

5.2 Data curation

Before the machine learning model can be developed, firstly data must be created to train and test the machine learning models. The data used was created instead of collected. Ideally, to make the data reliable and accurate the preferred method to source the data would have been surveys and interviews filled out by university students. However, the decision to create the data by hand was made so the project would fit within the guidelines of the ethics related to the project. As the project revolves around university students queries and struggles during university, the questions asked on the surveys/interviews would have been sensitive and regarded as too personal. Questions asked would involve information regarding mental health, finances, and domestic violence scenarios etc, therefore the interviews/survey would have been considered outside the scope of the ethics boundaries. It is important to note that as the data was hand crafted and not sourced there is an element of bias that is attached to the data. This bias affects the reliability of the research as the data is being created with the intention to be categorised in to one of five categories. The data was initially created on a Microsoft excel file. Microsoft excel was chosen as there are a variety of easy methods to input excel files into python coding files. The data created had a total of 291 entries which comprised of two features, university student queries and nature of query. Minimal features were chosen, as too many irrelevant features in data can decrease the accuracy of the models. The university student queries were qualitative inputs using small to medium sentences, whilst the nature of queries was quantitative, either being a number between 0-5 which correlates to the classifications: academic, social, domestic, language and finance respectively. Small to medium size sentences were chosen because larger sizes would affect the byte size of the inputted data causing for much longer machine learning training and prediction times.

5.3 Data pre-processing

As the curated data initially is in text format, data pre-processing was required to convert the data into a format that can be inputted into the machine learning models. In addition, the data needs to be processed so that optimal accuracy and efficiency can be achieved. Firstly, the dataset was shuffled to help minimize the chances of bias within the dataset during training, keeping each data point independent and therefore representative. Once the dataset has been shuffled the stop words which are a set of commonly used words such as “the”, “is”, “and” etc, are removed to remove low-level information that does not help with classification. Punctuation is then removed from the text in the dataset. Tokenisation or one hot encoding is applied to break the text from the data set into small pieces. In the case of this project the small pieces are words, however, they can be broken into phrases or characters. The tokens are used to create a vocabulary which is the set of unique tokens in the corpus. Each token is essentially a number that now can be passed into the machine learning model. Padding is applied to each data point so that they are all the same length which is an array of 20 numbers. This allows the whole dataset to be trained equally. The data is then split into training and testing variables in the ratio of 80% and 20% respectively.

5.4 Machine learning models

One of the objectives of this project is to review and find the optimal classification technique to classify student queries. From research in the literature review conducted, three main classification techniques were established that are the most popular and provide the best results. These models were: neural networks, decision trees and the genetic algorithm. For this project two neural networks and a decision tree were created and trained on the data that was curated to see which one produced the best results in terms of correctly classifying student queries. The genetic algorithm was then added on the client file to find the optimum weights of the models created.

When producing the neural network model, the python library TensorFlow Keras was used. Initially the neural network architecture consisted of an embedding input layer with parameters that consist of input dimension of 598 to reflect the size of the vocabulary from the data set, output dimension of 32 and an input length of size 20. A LSTM layer with parameter that consist of an output dimension of 64 and a dropout value of 0.1. A dense layer with parameters which consist of an output dimension of 64 and an activation function of softmax. Finally, a dense output layer with output dimension of 5 and activation function of softmax. A LSTM layer was chosen as it is effective in memorizing important information from the data. If LSTM layers are used appropriately, the model will be able to find out the meaning in the inputted tokenised text and provide accurate classifications. Instead of using an optimiser as a stochastic gradient descent to update the neural network weights, the genetic algorithm was used. Once the model was created, it is then transformed in to a parameter to create an initial population of the genetic algorithm based on the Keras model, using the pygad.kerasga module and the KerasGA class.

For the second neural network the python library Pygad was used. The neural network architecture consisted of an input layer with parameter that consist of an input dimension of size 20. Two hidden layers with parameter that consist of neurons with values 20 and 14 respectively. Finally, an output layer with an output dimension of 5. The hidden layers activation function was set to relu and the output layer activation function was set to softmax.

To create the decision tree model, the python library scikit learn was used. Decision trees are much easier to implement than neural networks with only parameters needed to be established instead of creating an architecture. The parameters for the decision tree consist of a criterion attribute of gini, a maximum depth of 10 and a splitter strategy of best.

5.5 Federated learning

When creating the federated learning cybersecurity method, the python libraries socket, pickle and threading were used. On the server side of the application, a socket is made and is bound to an IP address and port number. The socket is then changed to a listening mode so that it can detect connections coming from the clients. Once the socket has detected a connection, it is then accepted and the server and client can communicate with each other. The python threading module is used to create a thread, allowing multiple clients to connect to the server. On the client end of the application a socket is made. The client connects to the server by bounding the same IP address and port number as the server. Once the connection has been established the Pygad instance of the genetic algorithm containing the machine learning model is sent to the client from the server in encrypted byte form. To send the instance of the genetic algorithm the cryptography technique encryption is used. The instance is encoded using the python library pickle on the server end and is decoded on the client end. Once the data has been sent to the client, the clients train the machine learning model locally using the testing data set. When fully trained the genetic algorithm finds the optimum weights of the model and the weights are then encoded and then sent back to the server. The new weights from the clients are then aggregated and averaged to produce new weights which are then updated to the model. A prediction error test is performed on the updated model. The model weights are continuously sent back to the client for training on till a prediction error of zero is achieved. Once the expected prediction error is achieved, the model weights are then sent back to the clients for the last time so that student query classification can be performed.

5.6 Human application interface

To achieve an AI based privacy preserving counselling model, a human application interface needs to be established so students can interact with the app to receive a classification for their query. In this project the app was created using an application development python framework called Kivy. Kivy was used instead of libraries such as Beeware, as it is a more matured framework in comparison, which has contributed to over 20 organizations, therefore the library can be trusted. Kivy is a cross platform library allowing it to be used on supported platforms such as windows, Android, Linux, OS X etc. This feature is important as when fully functional the counselling application will be accessible to all students regardless of their device.

The app interface mainly consists of buttons which trigger specified events. For the server the first button created is the create socket button which initiates the socket. Then the bind socket is created which binds the IP address and the port number to the socket. The listen accept button is created which listens and connects the clients to the socket allowing communication between the server and the client. Finally, the close socket button closes the connection between server and the clients.

The client interface consists of buttons and two text inputs for the IP address and port number, and for users query text input. Like the server, the create socket button initiates the socket. The IP address and port number is then inputted. A text box is then placed for the user to input their query. A button called submit query is then used to submit the inputted text. Next the connect button allows for the inputted IP address and port number to be used to connect to the server. A receive and train model button is then used to allow the client to accept the data from the server and to initiate the local training of the model. A query prediction button is used to convert the inputted text to a format this is readable to the machine learning model and to produce a classification of the text that is then displayed on the screen for the user. Finally, the close socket button closes the connection between client and the server.

6 Project management

This project was managed by utilising the waterfall software development plan. As applied to project management, waterfall project management is a process for the management of a project, project planning, and the control of scope. It is a process in which a project is conceived and proceeds through its various phases or stages of completion, under rigid time and cost constraints, with a very high degree of focus and precision. The process is also commonly known as the “waterfall model” of project management. Waterfall project management is commonly employed in the discovery or creation of new products, especially complex systems. The waterfall planning process was applied to this project by following the phases which it consist: requirements, system design, implementation, testing deployment and maintenance. The requirements stage consisted of gathering documentation and analysing techniques to be used in the project. This was conducted through the initial proposal of the project and the literature review. The system design phase was the planning stage of the project which identified the workflow. Originally the workflow of the project was human app interface development, federated algorithm development, AI machine learning model within federated learning algorithm development, AI cyber security technique testing and development, training and testing. However, this workflow was altered to data curation, data pre-processing, machine learning models, federated learning and human app interface. The workflow was changed to incorporate essential steps such as data curation and data pre-processing. The new workflow sliced the original workflow into more manageable steps such as separating the AI machine learning model within federated learning algorithm development stage. This allowed for more focus on the machine learning models instead of merging the stage with the federated cybersecurity stage. Moreover, the new workflow saw the removal of the testing stage to corelate with the waterfall plan. The implementation stage saw the development of the system design stages. Testing allowed for each element of the project to be analysed and examined to see if they met the requirements of the project. Finally, the maintenance phase which is an ongoing phase, looks at the upkeep and improvement of the product. In this project that entailed the improvement of neural network models and the attempted improvement of the genetic algorithm parameters. For future maintenance, the upkeep and development of federated learning and the human app interface should be looked at.

7 Design, development and implementation

7.1 Introduction

The general structure of the project was based around the requirements of federated learning. As federated learning seeks to send a model from a server to clients to be locally trained separately from the server, it was required to have two app interfaces, the server and the client. Therefore, the project was based around two main files of code. Regarding the machine learning models, the structural architecture where possible was kept relatively simple. The simplicity of the structures was due to the overall computational time of the project. As one of the essences of the project is to provide university students with faster classifications of their problems and therefore faster help, it was important to reduce the amount of time that the machine learning models took to train.

7.2 Server

The structural layout of the server code firstly includes the server app class which makes up the interface of the app. The server app class is comprised of functions that represent the buttons events in the app interface. The server app class also has a build function which provides the layout of the app by adding the widgets (buttons) to the box layout that the app is based on. After the sever app class, the next components of the server code are the data used to test the accuracy of the machine learning model, the machine learning model itself and the instance of the genetic algorithm. This is placed between the server app class and the socket threading class as the genetic algorithm instance needs to be established as a variable so it

```
def create_socket(self, *args):
        self.soc =
socket.socket(family=socket.AF_INET,
type=socket.SOCK_STREAM)
        self.label.text = "Socket Created"

        self.create_socket_btn.disabled = True
        self.bind_btn.disabled = False
        self.close_socket_btn.disabled = False
```

Figure 1: Example of server app class function Create socket button event function

```
except BaseException as e:
        print("Error Decoding the Client's Data: {msg}.\n".format(msg=e))
        self.kivy_app.label.text = "Error Decoding the Client's Data"


except BaseException as e:
        print("Error Sending Data to the Client: {msg}.\n".format(msg=e))
        self.kivy_app.label.text = "Error Sending Data to the Client:{msg}".format(msg=e)
```

Figure 2: Two examples of error messages within socket threading class

can be accessed by the socket threading class to be sent to the client. The socket threading class is the biggest block of code in the server file as it includes the data receiving function, model averaging function and data reply function. When implementing functions in the socket threading class, various error messages were coded within to indicate that a desired function has failed. This allows bugs and errors to be easily spotted and quickly rectified. When the client sends the weights of the trained machine learning model, once received by the model and aggregated with the original weights of the server, the machine learning model is then tested on the test dataset to find the accuracy of the model. If the accuracy is not 0, meaning that it failed to accurately predict all of the test data, then the weights of the model are sent back to the client to be re-trained. Once the machine learning model reaches an accuracy of 0, then the machine learning model can be sent back to the client to produce a prediction of the student's query classification. The last section of the server code is the listening thread class. The listening thread class allows the server to connect to multiple clients. This is done by wrapping the socket accept method in an infinite while loop. Once a connection has been accepted and the server and client have exchanged their data with each other, the connection stops, the nested while loop suspends, and the socket is ready to accept a new connection.

```
class ListenThread(threading.Thread):

    def __init__(self, kivy_app):
        threading.Thread.__init__(self)
        self.kivy_app = kivy_app

    def run(self):
        while True:
            try:
                connection, client_info = self.kivy_app.soc.accept()
                self.kivy_app.label.text = "New Connection from
{client_info}".format(client_info=client_info)
                socket_thread = SocketThread(connection=connection,
                                             client_info=client_info,
                                             kivy_app=self.kivy_app,
                                             buffer_size=4096,
                                             recv_timeout=60)
                socket_thread.start()
            except BaseException as e:
                self.kivy_app.soc.close()
                print(e)
                self.kivy_app.label.text = "Socket is No Longer Accepting
Connections"
                self.kivy_app.create_socket_btn.disabled = False
                self.kivy_app.close_socket_btn.disabled = True
                break
```

Figure 3: Listening thread class code

7.3 Client

The structural layout of the client code, similar to the server code, starts with the client app class which makes up the interface of the app. The class consists of functions for the buttons on the app and the build function. The predict function, within the class is where the text inputted by the client user is pre-processed and transformed into a numpy array to create a prediction. To produce a Pygad prediction, the genetic algorithm instance is required. As the function to receive data is in a separate class from the predict function, to access the instance, it needs to be declared as a global variable to be accessed. The next section of code within the client file is the fitness function and the call-back generation function. The fitness function calculates the classification accuracy of each solution in the population of the genetic algorithm. The call back generation function is applied to update the parameters of all networks used in the genetic algorithm population, using the updated population trained weights method. The next section of code consists of a function that prepares the genetic algorithm and the pre-processing of the training data. Finally, the last section of the class file contains the receive thread class. The receive thread class contains the receive function which accepts and decodes the data from the client and the run function which sends the data of the model's weights back to the server. The data sent between the server and the client is a python dictionary which contains the data and the status of the data. The status of the data describes whether the data has been trained adequately "done" or requires further training "echo".

```
def fitness_func(solution, sol_idx):
    global GANN_instance, data_inputs, data_outputs#, X_train

    predictions =
    pygad.nn.predict(last_layer=GANN_instance.population_networ
    ks[sol_idx], data_inputs=data_inputs)

    correct_predictions = numpy.where(predictions ==
    data_outputs)[0].size
    solution_fitness = (correct_predictions/data_outputs.size)*100

    return solution_fitness
```

Figure 4: Fitness function code

```
def callback_generation(ga_instance):
    global GANN_instance, last_fitness

    population_matrices =
    pygad.gann.population_as_matrices(population_networks=GAN
    N_instance.population_networks,

    population_vectors=ga_instance.population)
```

Figure 5: Call back generation code

7.4 Machine learning model

When implementing the neural network using the Keras library, initially the architecture of the model consisted of an embedding input layer, a LSTM layer, a dense layer and a dense output layer. However, applying the model to the genetic algorithm caused an error enabling the instance of the genetic algorithm to be sent to the server as the output shape was incompatible. To try and rectify this, the architecture of the neural network was changed to an embedding input layer, a flatten layer and a dense output layer. This was able to increase the training accuracy of the model from 0.7 to 0.9. However, it was not able to send the genetic algorithm to the client. This was the same case for the decision tree. The decision tree model was unable to utilise the genetic algorithm as the sci kit learn library and the Pygad library are incompatible.

8 Results, Analysis and Evaluation

The machine learning models used in this project were analysed to discover the accuracy of their classification. All machine learning models were trained and tested on the same dataset. To find the accuracy of the scikit learn decision tree, sci-kit learn offers novel precision metrics within their library. Using the testing dataset, the decision tree produced an accuracy value of 0.339 or 33.9%, this indicates that out of the test data only 33.9% of the data was accurately predicted. This value is extremely low and therefore would not provide accurate classification of student queries if implemented. The low accuracy could be attributed to a variety of factors such as: low number of training samples, lack of dataset features, overfitting of data and more. To increase the accuracy of the model in future work, it is suggested to vastly increase the number of data, so the model has more examples to work with and reduce overfitting. Tune the hyperparameters to find the optimum constraints needed to create the model. Finally, increasing the dataset with appropriate features such as university degree, to provide the model with better context to accurately give a prediction. As mentioned previously in the design, development and implementation section, the decision tree was unable to successfully merge with the genetic algorithm. The Pygad library, used to code the genetic algorithm is not compatible with the sci kit library used to create the decision tree, therefore the decision tree was not able to be used as a classification model in the project.

Table 1: Accuracy of machine learning models

Machine learning model	Testing accuracy
Scikit learn decision tree	0.339
Keras neural network	0.201
Pygad neural network	0.996

Using the Keras libraries categorical accuracy metrics, the accuracy of the Keras neural network was determined as 0.201 or 20.1%. This value is lower than the decision tree and once again proving that the efficiency of the neural network model is not good enough to classify student’s queries. The low level of accuracy could be attributed to, low number of training data, insufficient neural network architecture, lack of dataset features, overfitting of data and poor implementation of the genetic algorithm to find optimum weights of the neural network. The genetic algorithm is used to train the neural network model and obtain the best weights for the model. However, after 50 generations the fitness of the model does not improve past 0.62 as seen in figure 6. The generation parameter was then increased to 300 but the fitness result remained the same as seen figure 7. To achieve better accuracy and fitness of the neural network the following methods are suggested. Increase the number of data to train the model. Increase the dataset with appropriate features. Replace the genetic algorithm with backpropagation to train the neural network. When running the server code and the client code, the instance of the genetic algorithm was not able to be sent to the client from the server due to incompatibility within the neural network model. A value error occurred stating that the initial input layer of shape (none, 20) was called on an input with incompatible shape (none,).

The solution for this error was not able to be found, therefore the Keras neural network was not able to be used as a classification model in the project.

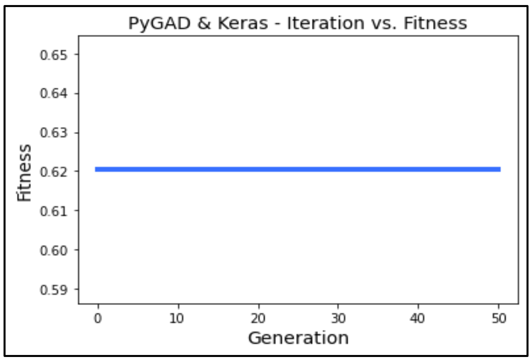

Figure 6: Fitness value of weights for 30 generations

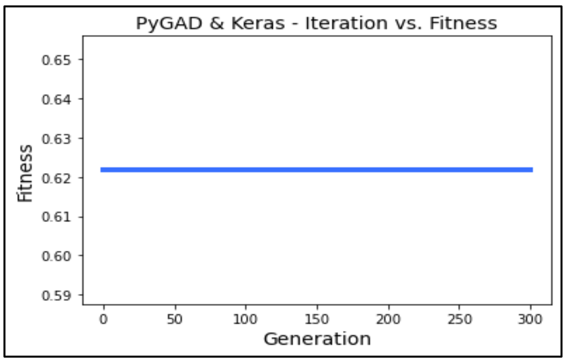

Figure 7: Fitness value of weights for 300 generations

The Pygad neural network produced the highest accuracy out of the machine learning models with a value of 99.6%. This accuracy level is highly favourable, making the Pygad neural network sufficient to act as a model to classify student queries. Although a high level of accuracy was achieved, like the Keras neural network, the fitness level of the model stayed stationary at a value of 0.62 over the generations. This suggests that the weights are not being trained by the genetic algorithm. Two conclusions could be made, either the optimum weights were found in the first generation, or the genetic algorithm is failing at some point. When running the server and client code the encrypted model is successfully sent between the server and the client. When the model has been trained locally on the client and sent back to the server, predictions are successfully made on sample data until the prediction error is zero. The server then replies to the client with the subject 'done' and the final instance of the model. Although the accuracy of the model is 99.6%, it takes a long period of time for the prediction error to take the value of zero. Moreover, at periods the connection between the server and the client times out before the value of zero can be achieved. This correlates with the second conclusion made about the genetic algorithm, suggesting that the genetic algorithm fails at some point as with each time the model is received by the server and the model, weights are aggregated with the original weights. The prediction accuracy should increase, however, as the genetic algorithm fitness does not increase per generation, this reduces the chance of the prediction error equating to zero. This increases the amount of time that a student must wait until receiving their prediction which goes against the goal of reducing classification time. Once the prediction error is zero the user can then receive a classification for their query. However, when testing the prediction on the client end using testing data, all predictions produce the value of zero, meaning the query is an academic classification. This fails to meet the project objective of producing accurate classification for student's queries. The client end was expected to accurately predict all test data as the model produced an accuracy value of 99.6%.

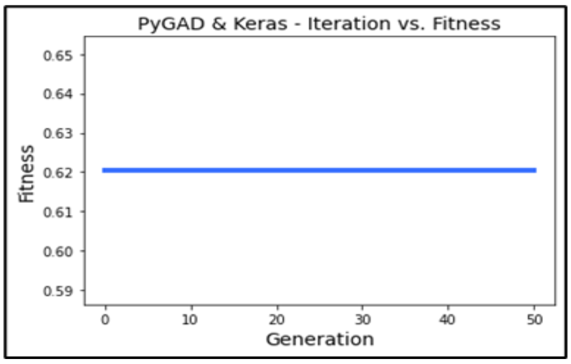

Figure 8: Listening thread class code

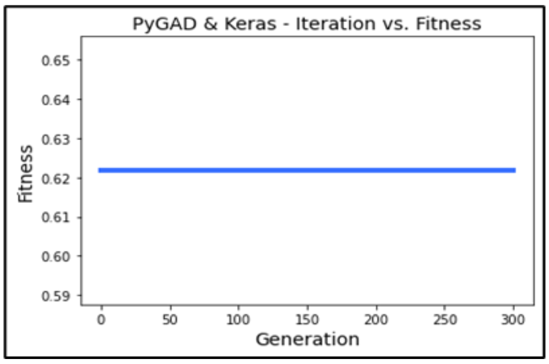

Figure 9: Listening thread class code

The implementation of the cybersecurity technique federated learning and the cryptography technique encryption within the project was successful. The Pygad neural network model weights were successfully encrypted on the server end and decoded on the client end. The model was then able to be trained locally on the clients end and then sent back to the server utilising encryption. The weights were then aggregated and updated to the model in the server. In addition, multiple clients were able to connect to the server to receive the weights of the model. This met the objective of applying a cybersecurity technique to leverage student privacy as student queries and training data remained local on the client's end, preserving their data.

9 Business Implementation

9.1 Business value of the end-product

The fully functional AI based privacy preserving counselling system provides the university with aid in providing a beneficial service for their beneficiaries. The universities beneficiaries are students who progress to be graduates and alumni. External organisations who employ students and engage with the university directly on research and educational programmes. Finally, the Birmingham and the West Midlands region, and wider society whom the organisation serve as a public institution (ASTON STRATEGY, 2018). By providing a novel app that seeks to streamline communication between students and the university, an improved university experience can be obtained by the students. This will allow more students at the university to navigate their university experience with efficient aid, allowing them to graduate and be employed by the external organisations, thus contributing to the wider society. Classifying student queries ultimately leads to a high-quality pool of student, serving as a pipeline of capable individuals to external organisations. This meets the universities business requirements of remaining focused on their beneficiaries and providing exceptional services (ASTON STRATEGY, 2018). One of the Aston universities requirements to be successful, is to operate professionally with agility and pace (ASTON STRATEGY, 2018). By utilising a functional privacy preserving counselling system, the university will be able to offer students with a faster paced form of aid. To provide students with the application created in this project, it is suggested that the service is adopted as an additional service on Aston's universities website and pre-existing mobile app 'myAston'. Adding this will provide extra traction to these platforms, allowing more students to engage and seek out the beneficial resources the university has to offer.

9.2 Potential implications

If the university were to adopt the service of the query classification app there are some implications that need to be addressed to obtain ideas of future development and to maintain the current climate of the institution. An implication that needs to be addressed is the potential displacement of current staff. As the query classification application seeks to essentially remove the intermediate party that asses a student's query before they are passed on to a staff member who can provide help, there may be staff members who will be made redundant as their role has been made obsolete. This can have a bad reflection on the University. To combat this, staff members could be offered training to maintain the software for staff and students. This would see to the employment of those who may be left without responsibilities. In addition, offering training would present staff members with newfound skills that will benefit the university and provide extra value to the employee.

Although the query classification application utilises the cybersecurity technique federated learning to provide a degree of privacy and protection against cyber threats, there is still the potential for attacks to occur. As described previously federated learning does not completely protect against threats and students may be liable to have their information stolen. This could leave the university to a worrisome dilemma having to face student, parents and government data regulations. A means of adding ease to the university and students is by adding additional protection to federated learning, this could be done by implementing, homomorphic encryption, differential privacy, or multi-party computation.

As the application in question will be a means of seeking counselling aid, the application needs to be accessible and easy to use so students can navigate the app and receive help without problems. This implies that the app requires constant monitoring and updates to constantly improve the user's experience. Students should be able to use the app during all twenty-four hours of the day indicating regular scheduled maintenance. This will require the university to either employ or train staff to satisfy the needs of the application.

As Aston university has a wide range of students and a large volume of international students, an implication that needs to be addressed is the accommodation for such individuals. The university has a responsibility to accommodate and cater to all its students and that applies with all services. To ensure that this is met, the app should consider providing a language setting, so students can change the language of the app to accommodate for themselves.

9.3 Strategy statement

To allow Aston university with the chance to offer students an efficient and streamlined form of communication to identify the issues they face without having to disclose sensitive and personal information to those it may not concern.

9.4 Three horizons analysis

1) The classification of queries for Aston university students
2) Expanding classification service to other universities in the United Kingdom
3) Expanding classification service globally to universities outside of the United Kingdom and other institutions such as secondary schools, sixth forms and organisations.

9.5 Ethics

Aston university has the obligation to abide by the Data Protection Act 2018 and the General Data Protection Regulation (EU) and the GDPR as retained in UK law by the Data Protection, Privacy and Electronic Communications (Amendments etc) (EU Exit) Regulations 2019 ("the UK GDPR"). This allows the university to process data about students such as name, address, date of birth and contact details. This also includes information about students' health and disability status, ethnicity, sexual orientation, gender reassignment or religion, if students have provided these (Aston, 2021). However, student have the right to restrict the access of such information and object to the processing of such information. Such rights require safeguarding techniques or methods in place to uphold students' rights liberties. The application adheres to student’s ethical right to restrict access to data with the use of federated learning. With the implementation of federated learning, information provided to produce a classification of student queries is kept local on student’s devices. therefore, is only ever seen by the student. Federated learning acts as a protective layer for student data, however, it does not provide complete protection against cyber theft. As there is still the potential for student data to be accessed through the reverse engineering of the sent weights to the client server, it needs to be stated that breaches could possibly occur. If such were to occur, this would put students’ data in danger therefore breaching the ethics and data regulation that must be adhered to. Intellectual property being accessed can also lead to destruction of university databases and requirements to notify and possibly compensate those affected.

10 Conclusion

In this project an AI based privacy preserving counselling system for students was made to speed up the process of student query classification and to create a streamline form of communication between Aston university and students. This system leveraged the cybersecurity technique federated learning and cryptography technique encryption, to conceal and preserve student data regarding topics such as academia, mental health, finance, domestic issues and language. To classify student queries the machine learning models decision trees, neural networks and the genetic algorithm were used and tested on curated data. From this project, neural networks were most successful in accurately predicting student queries achieving 99.6% accuracy on testing datasets. Once completed and executed the application system was unable to accurately classify student queries. However, the application of federated learning was successful in preserving data locally on client devices. To improve upon this project in future work backpropagation could be utilised as a replacement to the genetic algorithm to train the weights of the neural networks. To improve the data preservation aspect of the project, cybersecurity techniques homomorphic encryption and multi-party computation could be used in conjunction with federated learning to accommodate for its limitations. Finally, additional coding should be applied to produce more accurate final classifications of inputted student queries.

11 References

ASTON STRATEGY. (2018). [online] Available at: https://www.aston.ac.uk/sites/default/files/Aston%20University%20Strategy%202018%20%281%29.pdf.

Berisha, B. and Sc, B. (2021). *Big Data Analytics in Cloud Computing: An overview*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/348937287_Big_Data_Analytics_in_Cloud_Computing_An_overview/download.

Rich Caruana. (1997). Multitask learning. Machine learning, 28(1):41–75, 1997.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and ´ Pavel Kuksa. (2011). Natural language processing (almost) from scratch. The Journal of Machine Learning Research, 12:2493–2537, 2011.

Ali, M.A., Hickman, P.J. and Clementson, A.T. (1975). The Application of Automatic Interaction Detection (AID) in Operational Research. *Operational Research Quarterly (1970-1977)*, 26(2), p.243. doi:10.2307/3008458.

Allganize (2020). *How John McCarthy Shaped the Future of AI*. [online] Allganize. Available at: https://blog.allganize.ai/john-mccarthy/.

Apurva, A. et al. (2017). Redefining cyber security with big data analytics. *2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN)*. [online] doi:10.1109/ic3tsn.2017.8284476.

Aston (2021). *PRIVACY NOTICE MEMBERS OF COUNCIL Data Collection and Privacy Notice*. [online] Available at: https://www.aston.ac.uk/sites/default/files/privacy-notice-members-of-council-february-2021.pdf.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., Mcmahan, H., Patel, S., Ramage, D., Segal, A. and Seth, K. (2017). *Practical Secure Aggregation for Privacy-Preserving Machine Learning*. [online] Available at: https://eprint.iacr.org/2017/281.pdf.

Cheng, K., Fan, T., Jin, Y., Liu, Y., Chen, T., Papadopoulos, D. and Yang, Q. (2019). SecureBoost: A Lossless Federated Learning Framework. *arXiv.org*. [online] doi:10.48550/arXiv.1901.08755.

Cloudflare. (2022). *What is data privacy? | Privacy definition*. [online] Available at: https://www.cloudflare.com/en-gb/learning/privacy/what-is-data-privacy/.

Craigen, D., Diakun-Thibault, N. and Purse, R. (2014). *Technology Innovation Management Review Defining Cybersecurity*. [online] Available at: https://www.timreview.ca/sites/default/files/article_PDF/Craigen_et_al_TIMReview_October2014.pdf.

Criteo. (2017). *Companies Spend More Than $20b on Data Solutions Each Year | Criteo*. [online] Available at: https://www.criteo.com/blog/companies-spend-20b-data-solutions/.

Data Protection Act 2018 CHAPTER 12. (2018). [online] Available at: https://www.legislation.gov.uk/ukpga/2018/12/pdfs/ukpga_20180012_en.pdf.

Databasix UK Ltd. (2022). *20 Frightening Cyber Security Facts and Stats*. [online] Available at: https://www.dbxuk.com/statistics/cyber-security.

Dataprot. (2022). *Internet of Things statistics for 2022 - Taking Things Apart*. [online] Available at: https://dataprot.net/statistics/iot-statistics/.

Dwork, C. (2021). Differential Privacy: A Survey of Results. *Lecture Notes in Computer Science*, [online] pp.1–19. Available at: https://link.springer.com/chapter/10.1007/978-3-540-79228-4_1

Earnest Paul Ijjina and Krishna Mohan Chalavadi (2016). Human action recognition using genetic algorithms and convolutional neural networks. *Pattern Recognition*, [online] 59, pp.199–212. Available at: https://www.academia.edu/28687212/Human_action_recognition_using_genetic_algorithms_and_convolutional_neural_networks.

El Ansari, W., & Stock, C. (2010). Is the Health and Wellbeing of University Students Associated with their Academic Performance? Cross Sectional Findings from the United Kingdom. *International Journal of Environmental Research and Public Health*, *7*(2), 509–527. https://doi.org/10.3390/ijerph7020509

Evans, A. and Heimann, A. (2022). *Acrobat Accessibility Report*. [online] Available at: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attach ment_data/file/1045381/AI_Activity_in_UK_Businesses_Report__Capital_Economics_and_DCMS__January_2022__Web_accessible_.pdf.

Evans, D., Kolesnikov, V. and Rosulek, M. (2018). A Pragmatic Introduction to Secure Multi-Party Computation. *Foundations and Trends® in Privacy and Security*, 2(2-3), pp.70–246.

for, I. (2022). *Artificial Intelligence for Disaster Risk Reduction: Opportunities, challenges, and prospects*. [online] World Meteorological Organization. Available at: https://public.wmo.int/en/resources/bulletin/artificial-intelligence-disaster-risk-reduction-opportunities-challenges-and.

Gad, A. (2020). *Federated Learning Demo in Python (Part 1): Client-Server Application*. [online] Medium. Available at: https://medium.com/cometheartbeat/federated-learning-demo-in-python-part-1-client-server-application-cebdcfb96b9.

Gartner. (2019). *Gartner Survey Shows 37 Percent of Organizations Have Implemented AI in Some Form*. [online] Available at: https://www.gartner.com/en/newsroom/press-releases/2019-01-21-gartner-survey-shows-37-percent-of-organizations-have.

Grandviewresearch.com. (2022). *Artificial Intelligence In Healthcare Market Size Report, 2030*. [online] Available at: https://www.grandviewresearch.com/industry-analysis/artificial-intelligence-ai-healthcare-market# [Accessed 26 May 2022].

Grandviewresearch.com. (2022). *Artificial Intelligence Market Size Report, 2022-2030*. [online] Available at: https://www.grandviewresearch.com/industry-analysis/artificial-intelligence-ai-market.

Hesa.ac.uk. (2020). *Higher Education Student Statistics: UK, 2020/21 | HESA*. [online] Available at: https://www.hesa.ac.uk/news/25-01-2022/sb262-higher-education-student-statistics.

https://www.morganclaypool.com/doi/pdf/10.2200/S00431ED1V01Y201207DTM028.

Kamruzzaman, S., (2014). Text Classification using Artificial Intelligence. [online] Available at:https://www.researchgate.net/publication/46587273_Text_Classification_using_Artificial_Intelligence.

Kotsiantis, S.B. (2011). Decision trees: a recent overview. *Artificial Intelligence Review*, [online] 39(4), pp.261–283. doi:10.1007/s10462-011-9272-4.

Kotsiantis, S.B. (2013). Decision trees: a recent overview. Artif Intell Rev 39, 261–283 https://doi.org/10.1007/s10462-011-9272-4

Labovitz, D.L. et al. (2017). Using Artificial Intelligence to Reduce the Risk of Nonadherence in Patients on Anticoagulation Therapy. *Stroke*, [online] 48(5), pp.1416–1419. doi:10.1161/STROKEAHA.116.016281.

Li, Y., Zhou, Y., Jolfaei, A., Yu, D., Xu, G. and Zheng, X. (2021). Privacy-Preserving Federated Learning Framework Based on Chained Secure Multiparty Computing. *IEEE Internet of Things Journal*, [online] 8(8), pp.6178–6186. doi:10.1109/jiot.2020.3022911.

Liddy, E. (2001). *Natural Language Processing Natural Language Processing Natural Language Processing 1*. [online] Available at: https://surface.syr.edu/cgi/viewcontent.cgi?article=1043&context=istpub.

Liu, P., Qiu, X. and Huang, X. (n.d.). *Recurrent Neural Network for Text Classification with Multi-Task Learning*. [online] Available at: https://arxiv.org/pdf/1605.05101.pdf.

Mayo, R.C. and Leung, J. (2018). Artificial intelligence and deep learning – Radiology’s next frontier? *Clinical Imaging*, [online] 49, pp.87–88. doi:10.1016/j.clinimag.2017.11.007.

Mcmahan, H., Moore, E., Ramage, D., Hampson, S. and Agüera, B. (n.d.). *Communication-Efficient Learning of Deep Networks from Decentralized Data*. [online] Available at: https://arxiv.org/pdf/1602.05629.pdf.

Mei, X. et al. (2020). Artificial intelligence–enabled rapid diagnosis of patients with COVID-19. *Nature Medicine*, [online] 26(8), pp.1224–1228. doi:10.1038/s41591-020-0931-3.

Mendoza-Cano, O. et al(2021). Experiments of an IoT-based wireless sensor network for flood monitoring in Colima, Mexico. *Journal of Hydroinformatics*, [online] 23(3), pp.385–401. doi:10.2166/hydro.2021.126.

Mofatteh, M. (2021). Risk factors associated with stress, anxiety, and depression among university undergraduate students. *AIMS Public Health*, *8*(1), 36–65. https://doi.org/10.3934/publichealth.2021004

Morganclaypool.com. (2012). *Data Protection from Insider Threats | Synthesis Lectures on Data Management*. [online] Available at:

Ouellette, R., Browne, M. and Hirasawa, K. (2004). *Genetic algorithm optimization of a convolutional neural network for autonomous crack detection*. [online] undefined. Available at: https://www.semanticscholar.org/paper/Genetic-algorithm-optimization-of-a-convolutional-Ouellette-Browne/6f9f7ee9512b588f249ac398589effb42952bffb.

Phong, L.T., Aono, Y., Hayashi, T., Wang, L. and Moriai, S. (2018). Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. *IEEE Transactions on Information Forensics and Security*, [online] 13(5), pp.1333–1345. doi:10.1109/tifs.2017.2787987.

Quinlan, J.R. (1986). *Induction of Decision Trees*. [online] undefined. Available at: https://www.semanticscholar.org/paper/Induction-of-Decision-Trees-Quinlan/bcee7c85d237b79491a773ef51e746bbbcf48e35.

Ritschard, G. (2010). *CHAID and earlier supervised tree methods*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/254417591_CHAID_and_earlier_supervised_tree_methods.

Sether, A. (2016). *Cloud Computing Benefits*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/304380663_Cloud_Computing_Benefits/link/576d7a8508ae10de6395d28a/download.

Smith, C., McGuire, B., Huang, T. and Yang, G. (2006). *The History of Artificial Intelligence*. [online] Available at: https://courses.cs.washington.edu/courses/csep590/06au/projects/history-ai.pdf.

Stack, B. (2017). *Here's How Much Your Personal Information Is Selling for on the Dark Web*. [online] Experian.com. Available at: https://www.experian.com/blogs/ask-experian/heres-how-much-your-personal-information-is-selling-for-on-the-dark-web/.

Statista. (2020). *Total data volume worldwide 2010-2025 | Statista*. [online] Available at: https://www.statista.com/statistics/871513/worldwide-data-created/#:~:text=The%20total%20amount%20of%20data,replicated%20reached%20a%20new%20high.

Technologyonecorp.co.uk. (2021). *TechnologyOne upgrades security for all UK customers - TechnologyOne*. [online] Available at:

https://technologyonecorp.co.uk/resources/media-releases/TechnologyOne-upgrades-security-for-all-UK-customers#:~:text=According%20to%20research%2C%2054%25%20of,just%20behind%20healthcare%20(17%25).

The state of cyber security across UK universities An analysis of Freedom of Information requests. (2020). [online] Available at: https://www.redscan.com/media/The-state-of-cyber-security-across-UK-universities-Redscan-report.pdf.

Truex, S., Baracaldo, N., Anwar, A., Steinke, T., Ludwig, H., Zhang, R. and Zhou, Y. (2019). A Hybrid Approach to Privacy-Preserving Federated Learning. *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security - AISec'19*. doi:10.1145/3338501.3357370.

UCAS. (2021). *450% increase in student mental health declarations over last decade but progress still needed to address declarations stigma*. [online] Available at: https://www.ucas.com/corporate/news-and-key-documents/news/450-increase-student-mental-health-declarations-over-last-decade-progress-still-needed-address.

Wei K., Ding M., Farokhi F., Quek T. (2020) "Federated Learning With Differential Privacy: Algorithms and Performance Analysis," in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 3454-3469, , doi: 10.1109/TIFS.2020.2988575.

Whitley, D. (1994). A genetic algorithm tutorial. *Statistics and Computing*, [online] 4(2). doi:10.1007/bf00175354

Xu, R., Baracaldo, N., Zhou, Y., Anwar, A. and Ludwig, H. (2019). HybridAlpha. *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security - AISec'19*. doi:10.1145/3338501.3357371.

Yi, X., Paulet, R. and Bertino, E. (2014). Homomorphic Encryption. *Homomorphic Encryption and Applications*, [online] pp.27–46. Available at: https://link.springer.com/chapter/10.1007/978-3-319-12229-8_2

Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W. and Gao, Y. (2021). A survey on federated learning. *Knowledge-Based Systems*, [online] 216, p.106775. doi:10.1016/j.knosys.2021.106775.

12 Appendices

12.1 Pygad server code

```
import socket
import pickle
import threading
import time
import numpy

import pygad.nn
import pygad.gann
import pygad.kerasga

import kivy.app
import kivy.uix.button
import kivy.uix.label
import kivy.uix.textinput
import kivy.uix.boxlayout

class ServerApp(kivy.app.App):

    def __init__(self):
        super().__init__()

    def create_socket(self, *args):
        self.soc = socket.socket(family=socket.AF_INET,
type=socket.SOCK_STREAM)
        self.label.text = "Socket Created"

        self.create_socket_btn.disabled = True
        self.bind_btn.disabled = False
        self.close_socket_btn.disabled = False

    def bind_socket(self, *args):
        ipv4_address = self.server_ip.text
        port_number = self.server_port.text
        self.soc.bind((ipv4_address, int(port_number)))
        self.label.text = "Socket Bound to IPv4 & Port Number"

        self.bind_btn.disabled = True
        self.listen_btn.disabled = False

    def listen_accept(self, *args):
        self.soc.listen(1)
        self.label.text = "Socket is Listening for Connections"

        self.listen_btn.disabled = True
        self.listenThread = ListenThread(kivy_app=self)
        self.listenThread.start()

    def close_socket(self, *args):
        self.soc.close()
        self.label.text = "Socket Closed"

        self.create_socket_btn.disabled = False
        self.bind_btn.disabled = True
        self.listen_btn.disabled = True
        self.close_socket_btn.disabled = True

    def build(self):
        self.create_socket_btn = kivy.uix.button.Button(text="Create Socket",
disabled=False)
        self.create_socket_btn.bind(on_press=self.create_socket)

        self.server_ip = kivy.uix.textinput.TextInput(hint_text="IPv4 Address",
text="localhost")
        self.server_port = kivy.uix.textinput.TextInput(hint_text="Port Number",
text="10000")

        self.server_socket_box_layout =
kivy.uix.boxlayout.BoxLayout(orientation="horizontal")
        self.server_socket_box_layout.add_widget(self.server_ip)
        self.server_socket_box_layout.add_widget(self.server_port)

        self.bind_btn = kivy.uix.button.Button(text="Bind Socket", disabled=True)
        self.bind_btn.bind(on_press=self.bind_socket)

        self.listen_btn = kivy.uix.button.Button(text="Listen to Connections",
disabled=True)
        self.listen_btn.bind(on_press=self.listen_accept)

        self.close_socket_btn = kivy.uix.button.Button(text="Close Socket",
disabled=True)
        self.close_socket_btn.bind(on_press=self.close_socket)

        self.label = kivy.uix.label.Label(text="Socket Status")

        self.box_layout = kivy.uix.boxlayout.BoxLayout(orientation="vertical")

        self.box_layout.add_widget(self.create_socket_btn)
        self.box_layout.add_widget(self.server_socket_box_layout)
        self.box_layout.add_widget(self.bind_btn)
        self.box_layout.add_widget(self.listen_btn)
        self.box_layout.add_widget(self.close_socket_btn)
        self.box_layout.add_widget(self.label)
        return self.box_layout

model = None

data_inputs = numpy.array([[429, 590, 261, 330, 155,   0,   0,   0,   0,   0,   0,   0,   0,
0,   0,   0,   0,   0,
   0,   0],
                           [89, 159, 236, 236,  59, 393,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
0,   0,
   0,   0]])

data_outputs = numpy.array([4,
                            1])

num_classes = 5
num_inputs = data_inputs.shape[1]

num_solutions = 6
GANN_instance = pygad.gann.GANN(num_solutions=num_solutions,
                                num_neurons_input=num_inputs,
                                num_neurons_hidden_layers=[20, 14],
                                num_neurons_output=num_classes,
                                hidden_activations="relu",
                                output_activation="softmax")

class SocketThread(threading.Thread):

    def __init__(self, connection, client_info, kivy_app, buffer_size=4096,
recv_timeout=5):
        threading.Thread.__init__(self)
        self.connection = connection
        self.client_info = client_info
        self.buffer_size = buffer_size
        self.recv_timeout = recv_timeout
        self.kivy_app = kivy_app

    def recv(self):
        received_data = b""
        while True:
            try:

                data = self.connection.recv(self.buffer_size)
                received_data += data

                if data == b'':
                    received_data = b""

                    if (time.time() - self.recv_start_time) > self.recv_timeout:
                        return None, 0
                elif str(data)[-2] == '.':
                    print("All data ({data_len} bytes) Received from
{client_info}.".format(client_info=self.client_info, data_len=len(received_data)))
                    self.kivy_app.label.text = "All data ({data_len} bytes) Received from
{client_info}.".format(client_info=self.client_info, data_len=len(received_data))

                    if len(received_data) > 0:
                        try:
                            received_data = pickle.loads(received_data)
                            return received_data, 1

                        except BaseException as e:
                            print("Error Decoding the Client's Data: {msg}.\n".format(msg=e))
                            self.kivy_app.label.text = "Error Decoding the Client's Data"
                            return None, 0

                else:
                    self.recv_start_time = time.time()

            except BaseException as e:
                print("Error Receiving Data from the Client: {msg}.\n".format(msg=e))
                self.kivy_app.label.text = "Error Receiving Data from the Client"
                return None, 0

    def model_averaging(self, model, other_model):
        model_weights = pygad.nn.layers_weights(last_layer=model, initial=False)
        other_model_weights = pygad.nn.layers_weights(last_layer=other_model,
initial=False)

        new_weights = numpy.array(model_weights + other_model_weights)/2

        pygad.nn.update_layers_trained_weights(last_layer=model,
final_weights=new_weights)

    def reply(self, received_data):
        global GANN_instance, data_inputs, data_outputs, model
        if (type(received_data) is dict):
            if (("data" in received_data.keys()) and ("subject" in received_data.keys())):
                subject = received_data["subject"]
                print("Client's Message Subject is {subject}.".format(subject=subject))
                self.kivy_app.label.text = "Client's Message Subject is
{subject}".format(subject=subject)

                print("Replying to the Client.")
                self.kivy_app.label.text = "Replying to the Client"
                if subject == "echo":
                    if model is None:
                        data = {"subject": "model", "data": GANN_instance}
                    else:
```

```
                predictions = pygad.nn.predict(last_layer=model,
data_inputs=data_inputs)
                error = numpy.sum(numpy.abs(predictions - data_outputs))
                if error == 0:
                    data = {"subject": "done", "data": GANN_instance}
                else:
                    data = {"subject": "model", "data": GANN_instance}

            try:
                response = pickle.dumps(data)
            except BaseException as e:
                print("Error Encoding the Message: {msg}.\n".format(msg=e))
                self.kivy_app.label.text = "Error Encoding the Message"
        elif subject == "model":
            try:
                GANN_instance = received_data["data"]
                best_model_idx = received_data["best_solution_idx"]

                best_model = GANN_instance.population_networks[best_model_idx]
                if model is None:
                    model = best_model
                else:
                    predictions = pygad.nn.predict(last_layer=model,
data_inputs=data_inputs)

                    error = numpy.sum(numpy.abs(predictions - data_outputs))

                    if error == 0:
                        data = {"subject": "done", "data": GANN_instance}
                        response = pickle.dumps(data)
                        return

                    self.model_averaging(model, best_model)


                predictions = pygad.nn.predict(last_layer=model,
data_inputs=data_inputs)
                print("Model Predictions:
{predictions}".format(predictions=predictions))

                error = numpy.sum(numpy.abs(predictions - data_outputs))
                print("Prediction Error = {error}".format(error=error))
                self.kivy_app.label.text = "Prediction Error =
{error}".format(error=error)

                if error != 0:
                    data = {"subject": "model", "data": GANN_instance}
                    response = pickle.dumps(data)
                else:
                    data = {"subject": "done", "data": GANN_instance}
                    response = pickle.dumps(data)

                except BaseException as e:
                    print("Error Decoding the Client's Data: {msg}.\n".format(msg=e))
                    self.kivy_app.label.text = "Error Decoding the Client's Data"
            else:
                response = pickle.dumps("Response from the Server")

            try:
                self.connection.sendall(response)
            except BaseException as e:
                print("Error Sending Data to the Client: {msg}.\n".format(msg=e))
                self.kivy_app.label.text = "Error Sending Data to the Client:
{msg}".format(msg=e)

        else:
            print("The received dictionary from the client must have the 'subject' and
'data' keys available. The existing keys are
{d_keys}.".format(d_keys=received_data.keys()))
            self.kivy_app.label.text = "Error Parsing Received Dictionary"
    else:
        print("A dictionary is expected to be received from the client but {d_type}
received.".format(d_type=type(received_data)))
        self.kivy_app.label.text = "A dictionary is expected but {d_type}
received.".format(d_type=type(received_data))

  def run(self):
    print("Running a Thread for the Connection with
{client_info}.".format(client_info=self.client_info))
    self.kivy_app.label.text = "Running a Thread for the Connection with
{client_info}.".format(client_info=self.client_info)

    while True:
        self.recv_start_time = time.time()
        time_struct = time.gmtime()
        date_time = "Waiting to Receive Data Starting from {day}/{month}/{year}
{hour}:{minute}:{second} GMT".format(year=time_struct.tm_year,
month=time_struct.tm_mon, day=time_struct.tm_mday, hour=time_struct.tm_hour,
minute=time_struct.tm_min, second=time_struct.tm_sec)
        print(date_time)
        received_data, status = self.recv()
        if status == 0:
            self.connection.close()
            self.kivy_app.label.text = "Connection Closed with
{client_info}".format(client_info=self.client_info)
            print("Connection Closed with {client_info} either due to inactivity for
{recv_timeout} seconds or due to an error.".format(client_info=self.client_info,
recv_timeout=self.recv_timeout), end="\n\n")
            break
        self.reply(received_data)

class ListenThread(threading.Thread):

  def __init__(self, kivy_app):
    threading.Thread.__init__(self)
    self.kivy_app = kivy_app

  def run(self):
    while True:
        try:
            connection, client_info = self.kivy_app.soc.accept()
            self.kivy_app.label.text = "New Connection from
{client_info}".format(client_info=client_info)
            socket_thread = SocketThread(connection=connection,
                                         client_info=client_info,
                                         kivy_app=self.kivy_app,
                                         buffer_size=4096,
                                         recv_timeout=60)
            socket_thread.start()
        except BaseException as e:
            self.kivy_app.soc.close()
            print(e)
            self.kivy_app.label.text = "Socket is No Longer Accepting Connections"
            self.kivy_app.create_socket_btn.disabled = False
            self.kivy_app.close_socket_btn.disabled = True
            break

serverApp = ServerApp()
serverApp.title="Server App"
serverApp.run()
```

12.2 Pygad client code

```
import socket
import pickle
import numpy

import pygad
import pygad.nn
import pygad.gann

from tensorflow import keras
from keras.preprocessing.text import one_hot
from tensorflow.keras.preprocessing.sequence import pad_sequences
from sklearn.model_selection import train_test_split
from tensorflow import keras
import pandas as pd
import numpy as np

import kivy.app
import kivy.uix.button
import kivy.uix.label
import kivy.uix.boxlayout
import kivy.uix.textinput

import threading


class ClientApp(kivy.app.App):


    def __init__(self):
        super().__init__()


    def create_socket(self, *args):
        self.soc = socket.socket(family=socket.AF_INET,
type=socket.SOCK_STREAM)
        self.label.text = "Socket Created"

        self.create_socket_btn.disabled = True
        self.connect_btn.disabled = False
        self.close_socket_btn.disabled = False

    def connect(self, *args):
        try:
            self.soc.connect((self.server_ip.text, int(self.server_port.text)))
            self.label.text = "Successful Connection to the Server"

            self.connect_btn.disabled = True
            self.recv_train_model_btn.disabled = False
        except BaseException as e:
            self.label.text = "Error Connecting to the Server"
            print("Error Connecting to the Server: {msg}".format(msg=e))

            self.connect_btn.disabled = False
            self.recv_train_model_btn.disabled = True

    def text_submit(self, *args):

        print("Query: ", self.student_text_input.text)

        self.text_submit_btn.disabled = True
        self.connect_btn.disabled = False

    def recv_train_model(self, *args):
        global GANN_instance, best_sol_idx, status, ga_instance

        self.recv_train_model_btn.disabled = True
        self.predict_btn.disabled = False
        recvThread = RecvThread(kivy_app=self, buffer_size=4096, recv_timeout=10)
        recvThread.start()



    def predict(self, *args):
        global GANN_instance, status, subject, best_sol_idx

        self.predict_btn.disabled = True
        self.recv_train_model_btn.disabled = False

        query =[]

        query.append(self.student_text_input.text)

        df = pd.Series(query)

        import re
        import string

        def remove_punct(text):
            translator = str.maketrans("", "", string.punctuation)
            return text.translate(translator)

        df= df.map(remove_punct)

        import nltk
        nltk.download('stopwords')
        from nltk.corpus import stopwords
```

```
stop = set(stopwords.words("english"))

def remove_stopwords(text):
    filtered_words = [word.lower() for word in text.split() if word.lower() not in
stop]
    return " ".join(filtered_words)

df= df.map(remove_stopwords)

print(df)

from collections import Counter

def counter_word(text_col):
    count = Counter()
    for text in text_col.values:
        for word in text.split():
            count [word] += 1
    return count

counter = counter_word(df)

num_unique_words = len(counter)

vocab_size = 598

print(vocab_size)


encoded_messages = [one_hot(d, vocab_size) for d in df]

print(encoded_messages)

max_length = 20
padded_messages = pad_sequences(encoded_messages,
maxlen=max_length, padding='post', truncating='post')

X_train = []

X_train.append(padded_messages)

X_train = np.array(X_train)

ga_instance = prepare_GA(GANN_instance)

best_sol_idx = ga_instance.best_solution()[2]

predictions =
pygad.nn.predict(last_layer=GANN_instance.population_networks[best_sol_idx],
data_inputs= X_train)
```

```
        print("Model Predictions: {predictions}".format(predictions=predictions))
        self.label.text = "Model Predictions:
{predictions}".format(predictions=predictions)


    def close_socket(self, *args):
        self.soc.close()
        self.label.text = "Socket Closed"

        self.create_socket_btn.disabled = False
        self.connect_btn.disabled = True
        self.recv_train_model_btn.disabled = True
        self.predict_btn = True
        self.close_socket_btn.disabled = True

    def build(self):
        self.create_socket_btn = kivy.uix.button.Button(text="Create Socket")
        self.create_socket_btn.bind(on_press=self.create_socket)

        self.server_ip = kivy.uix.textinput.TextInput(hint_text="Server IPv4 Address",
text="localhost")
        self.server_port = kivy.uix.textinput.TextInput(hint_text="Server Port Number",
text="10000")

        self.server_info_boxlayout =
kivy.uix.boxlayout.BoxLayout(orientation="horizontal")
        self.server_info_boxlayout.add_widget(self.server_ip)
        self.server_info_boxlayout.add_widget(self.server_port)

        self.connect_btn = kivy.uix.button.Button(text="Connect to Server",
disabled=True)
        self.connect_btn.bind(on_press=self.connect)

        self.student_text_input = kivy.uix.textinput.TextInput(hint_text="Descrpition of
query", multiline = False)

        self.text_submit_btn = kivy.uix.button.Button(text="Submit query")
        self.text_submit_btn.bind(on_press=self.text_submit)

        self.recv_train_model_btn = kivy.uix.button.Button(text="Receive & Train
Model", disabled=True)
        self.recv_train_model_btn.bind(on_press=self.recv_train_model)

        self.close_socket_btn = kivy.uix.button.Button(text="Close Socket",
disabled=True)
        self.close_socket_btn.bind(on_press=self.close_socket)

        self.predict_btn= kivy.uix.button.Button(text="Query prediction", disabled=True)
        self.predict_btn.bind(on_press=self.predict)
        self.label = kivy.uix.label.Label(text="Socket Status")

        self.box_layout = kivy.uix.boxlayout.BoxLayout(orientation="vertical")
        self.box_layout.add_widget(self.create_socket_btn)
        self.box_layout.add_widget(self.server_info_boxlayout)
        self.box_layout.add_widget(self.student_text_input)
        self.box_layout.add_widget(self.text_submit_btn)
        self.box_layout.add_widget(self.connect_btn)
        self.box_layout.add_widget(self.recv_train_model_btn)
        self.box_layout.add_widget(self.predict_btn)
        self.box_layout.add_widget(self.close_socket_btn)
        self.box_layout.add_widget(self.label)

        return self.box_layout

def fitness_func(solution, sol_idx):
    global GANN_instance, data_inputs, data_outputs

    predictions = 
pygad.nn.predict(last_layer=GANN_instance.population_networks[sol_idx], 
data_inputs=data_inputs)

    correct_predictions = numpy.where(predictions == data_outputs)[0].size
    solution_fitness = (correct_predictions/data_outputs.size)*100

    return solution_fitness

def callback_generation(ga_instance):
    global GANN_instance, last_fitness

    population_matrices = 
pygad.gann.population_as_matrices(population_networks=GANN_instance.populati
on_networks,
                                                    population_vectors=ga_instance.population)


GANN_instance.update_population_trained_weights(population_trained_weights=po
pulation_matrices)

def prepare_GA(GANN_instance):
    population_vectors = 
pygad.gann.population_as_vectors(population_networks=GANN_instance.populatio
n_networks)

    initial_population = population_vectors.copy()

    num_parents_mating = 4
    num_generations = 500

    mutation_percent_genes = 5

    parent_selection_type = "sss"

    crossover_type = "single_point"

    mutation_type = "random"

    keep_parents = 1

    init_range_low = -2
    init_range_high = 5

    ga_instance = pygad.GA(num_generations=num_generations,
                           num_parents_mating=num_parents_mating,
                           initial_population=initial_population,
                           fitness_func=fitness_func,
                           mutation_percent_genes=mutation_percent_genes,
                           init_range_low=init_range_low,
                           init_range_high=init_range_high,
                           parent_selection_type=parent_selection_type,
                           crossover_type=crossover_type,
                           mutation_type=mutation_type,
                           keep_parents=keep_parents,
                           callback_generation=callback_generation)

    return ga_instance


df = pd.read_excel("sample data for disso with one feature.xlsx", names = ['text',
'category'])

df = df.sample(frac = 1)

import re
import string

def remove_punct(text):
    translator = str.maketrans("", "", string.punctuation)
    return text.translate(translator)

string.punctuation

df["text"] = df.text.map(remove_punct)

import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
```

```
stop = set(stopwords.words("english"))

def remove_stopwords(text):
    filtered_words = [word.lower() for word in text.split() if word.lower() not in stop]
    return " ".join(filtered_words)

df["text"] = df.text.map(remove_stopwords)

from collections import Counter

def counter_word(text_col):
    count = Counter()
    for text in text_col.values:
        for word in text.split():
            count [word] += 1
    return count

counter = counter_word(df.text)

counter.most_common(5)

num_unique_words = len(counter)

train_size = int(df.shape[0] * 0.8)

train_df = df[:train_size]
test_df = df[train_size:]

train_sentences = train_df.text.to_numpy()
train_labels = train_df.category.to_numpy()
test_sentences = test_df.text.to_numpy()
test_labels = test_df.category.to_numpy()

from tensorflow.keras.preprocessing.text import Tokenizer

tokenizer = Tokenizer(num_words=num_unique_words)
tokenizer.fit_on_texts(train_sentences)

word_index = tokenizer.word_index

train_sequences = tokenizer.texts_to_sequences(train_sentences)
test_sequences = tokenizer.texts_to_sequences(test_sentences)

from tensorflow.keras.preprocessing.sequence import pad_sequences

max_length = 20

train_padded = pad_sequences(train_sequences, maxlen=max_length,
padding="post", truncating="post")
```

```
test_padded = pad_sequences(test_sequences, maxlen=max_length,
padding="post", truncating="post")
train_padded.shape, test_padded.shape

reverse_word_index =dict([(idx, word) for (word, idx) in word_index.items()])

def decode(sequence):
    return " ".join([reverse_word_index.get(idx, "?") for idx in sequence])


data_inputs =np.array(train_padded)

data_outputs =np.array(train_labels)


class RecvThread(threading.Thread):

    def __init__(self, kivy_app, buffer_size, recv_timeout):
        threading.Thread.__init__(self)
        self.kivy_app = kivy_app
        self.buffer_size = buffer_size
        self.recv_timeout = recv_timeout

    def recv(self):
        received_data = b""
        while str(received_data)[-2] != '.':
            try:
                self.kivy_app.soc.settimeout(self.recv_timeout)
                received_data += self.kivy_app.soc.recv(self.buffer_size)
            except socket.timeout:
                print("A socket.timeout exception occurred because the server did not
send any data for {recv_timeout} seconds.".format(recv_timeout=self.recv_timeout))
                self.kivy_app.label.text = "{recv_timeout} Seconds of Inactivity.
socket.timeout Exception Occurred".format(recv_timeout=self.recv_timeout)
                return None, 0
            except BaseException as e:
                return None, 0
                print("Error While Receiving Data from the Server: {msg}.".format(msg=e))
                self.kivy_app.label.text = "Error While Receiving Data from the Server"

        try:
            received_data = pickle.loads(received_data)
        except BaseException as e:
            print("Error Decoding the Client's Data: {msg}.\n".format(msg=e))
            self.kivy_app.label.text = "Error Decoding the Client's Data"
            return None, 0

        return received_data, 1

    def run(self):
        global GANN_instance


        subject = "echo"
        GANN_instance = None
        best_sol_idx = -1

        while True:
            data = {"subject": subject, "data": GANN_instance, "best_solution_idx":
best_sol_idx}
            data_byte = pickle.dumps(data)

            self.kivy_app.label.text = "Sending a Message of Type {subject} to the
Server".format(subject=subject)
            try:
                self.kivy_app.soc.sendall(data_byte)
            except BaseException as e:
                self.kivy_app.label.text = "Error Connecting to the Server. The server
might has been closed."
                print("Error Connecting to the Server: {msg}".format(msg=e))
                break

            self.kivy_app.label.text = "Receiving Reply from the Server"
            received_data, status = self.recv()
            if status == 0:
                self.kivy_app.label.text = "Nothing Received from the Server"
                break
            else:
                self.kivy_app.label.text = "New Message from the Server"

            subject = received_data["subject"]
            if subject == "model":
                GANN_instance = received_data["data"]

            elif subject == "done":
                GANN_instance = received_data["data"]
                best_sol_idx = ga_instance.best_solution()[2]
                self.kivy_app.label.text = "Model is Trained"

                break
            else:
                self.kivy_app.label.text = "Unrecognized Message Type:
{subject}".format(subject=subject)
                break

            ga_instance = prepare_GA(GANN_instance)

            ga_instance.run()

            subject = "model"
            best_sol_idx = ga_instance.best_solution()[2]


clientApp = ClientApp()
clientApp.title = "Client App"
clientApp.run()
```

12.3 Keras server code

```
import numpy as np
import pandas as pd
import os
from tensorflow import keras
from keras.preprocessing.text import one_hot
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Embedding
from sklearn.model_selection import train_test_split
from tensorflow import keras
from tensorflow.keras import layers
import tensorflow.keras
import tensorflow as tf

import socket
import pickle
import threading
import time
import numpy

import tensorflow.keras
import pygad.kerasga


import kivy.app
import kivy.uix.button
import kivy.uix.label
import kivy.uix.textinput
import kivy.uix.boxlayout

class ServerApp(kivy.app.App):

    def __init__(self):
        super().__init__()

    def create_socket(self, *args):
        self.soc = socket.socket(family=socket.AF_INET,
type=socket.SOCK_STREAM)
        self.label.text = "Socket Created"

        self.create_socket_btn.disabled = True
        self.bind_btn.disabled = False
        self.close_socket_btn.disabled = False

    def bind_socket(self, *args):
        ipv4_address = self.server_ip.text
        port_number = self.server_port.text
        self.soc.bind((ipv4_address, int(port_number)))
        self.label.text = "Socket Bound to IPv4 & Port Number"

        self.bind_btn.disabled = True
        self.listen_btn.disabled = False

    def listen_accept(self, *args):
        self.soc.listen(1)
        self.label.text = "Socket is Listening for Connections"

        self.listen_btn.disabled = True

        self.listenThread = ListenThread(kivy_app=self)
        self.listenThread.start()

    def close_socket(self, *args):
        self.soc.close()
        self.label.text = "Socket Closed"

        self.create_socket_btn.disabled = False
        self.bind_btn.disabled = True
        self.listen_btn.disabled = True
        self.close_socket_btn.disabled = True

    def build(self):
        self.create_socket_btn = kivy.uix.button.Button(text="Create Socket",
disabled=False)
        self.create_socket_btn.bind(on_press=self.create_socket)

        self.server_ip = kivy.uix.textinput.TextInput(hint_text="IPv4 Address",
text="localhost")
        self.server_port = kivy.uix.textinput.TextInput(hint_text="Port Number",
text="10000")

        self.server_socket_box_layout =
kivy.uix.boxlayout.BoxLayout(orientation="horizontal")
        self.server_socket_box_layout.add_widget(self.server_ip)
        self.server_socket_box_layout.add_widget(self.server_port)

        self.bind_btn = kivy.uix.button.Button(text="Bind Socket", disabled=True)
        self.bind_btn.bind(on_press=self.bind_socket)

        self.listen_btn = kivy.uix.button.Button(text="Listen to Connections",
disabled=True)
        self.listen_btn.bind(on_press=self.listen_accept)

        self.close_socket_btn = kivy.uix.button.Button(text="Close Socket",
disabled=True)
        self.close_socket_btn.bind(on_press=self.close_socket)
        self.label = kivy.uix.label.Label(text="Socket Status")

        self.box_layout = kivy.uix.boxlayout.BoxLayout(orientation="vertical")

        self.box_layout.add_widget(self.create_socket_btn)
        self.box_layout.add_widget(self.server_socket_box_layout)
        self.box_layout.add_widget(self.bind_btn)
        self.box_layout.add_widget(self.listen_btn)
        self.box_layout.add_widget(self.close_socket_btn)
        self.box_layout.add_widget(self.label)

        return self.box_layout

model = None

data_inputs = numpy.array([[429, 590, 261, 330, 155,   0,   0,   0,   0,   0,   0,   0,   0,
0,   0,   0,   0,   0,
   0,   0],
                    [89, 159, 236, 236,  59, 393,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
0,   0,
   0,   0]])


data_outputs = numpy.array([4,
                    1])


df = pd.read_excel("sample data for disso with one feature.xlsx", names = ['text',
'category'])

df = df.sample(frac = 1)

import re
import string

def remove_punct(text):
   translator = str.maketrans("", "", string.punctuation)
   return text.translate(translator)


df["text"] = df.text.map(remove_punct)

import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords

stop = set(stopwords.words("english"))

def remove_stopwords(text):
    filtered_words = [word.lower() for word in text.split() if word.lower() not in stop]
    return " ".join(filtered_words)

df["text"] = df.text.map(remove_stopwords)

from collections import Counter

def counter_word(text_col):
    count = Counter()
    for text in text_col.values:
        for word in text.split():
            count [word] += 1
    return count

counter = counter_word(df.text)

num_unique_words = len(counter)
print(num_unique_words)

vocab_size = num_unique_words
encoded_messages = [one_hot(d, vocab_size) for d in df["text"]]

max_length = 20
padded_messages = pad_sequences(encoded_messages, maxlen=max_length,
padding='post', truncating='post')
padded_labels = df["category"]

X_train,X_test,y_train,y_test = train_test_split(padded_messages,
padded_labels,test_size=0.2)

X_train = np.array(X_train)
y_train = tensorflow.keras.utils.to_categorical(y_train)
y_train = np.array(y_train)

X_test = np.array(X_test)
y_test = tensorflow.keras.utils.to_categorical(y_test)
y_test = np.array(y_test)

X_test = X_test.reshape(-1,1)

embeded_vector_size = 40

inputs = keras.Input(shape=(20))
embedding_layer = layers.Embedding(num_unique_words, embeded_vector_size,
input_length=max_length, name="embedding")(inputs)
flatten_layer = layers.Flatten()(embedding_layer)
dense_layer = layers.Dense(5, activation='softmax')(flatten_layer)

model = keras.Model(inputs=inputs, outputs=dense_layer)
```

```
keras_ga = pygad.kerasga.KerasGA(model=model,
                                 num_solutions=10)

class SocketThread(threading.Thread):

    def __init__(self, connection, client_info, kivy_app, buffer_size=4096,
recv_timeout=60):
        threading.Thread.__init__(self)
        self.connection = connection
        self.client_info = client_info
        self.buffer_size = buffer_size
        self.recv_timeout = recv_timeout
        self.kivy_app = kivy_app

    def recv(self):
        all_data_received_flag = False
        received_data = b""
        while True:
            try:
                data = self.connection.recv(self.buffer_size)
                received_data += data

                try:
                    pickle.loads(received_data)
                    all_data_received_flag = True
                except BaseException:
                    pass

                if data == b'':
                    received_data = b""

                    if (time.time() - self.recv_start_time) > self.recv_timeout:
                        return None, 0

                elif all_data_received_flag:
                    print("All data ({data_len} bytes) Received from
{client_info}.".format(client_info=self.client_info, data_len=len(received_data)))
                    self.kivy_app.label.text = "All data ({data_len} bytes) Received from
{client_info}.".format(client_info=self.client_info, data_len=len(received_data))

                    if len(received_data) > 0:
                        try:
                            received_data = pickle.loads(received_data)
                            return received_data, 1

                        except BaseException as e:
                            print("Error Decoding the Client's Data: {msg}.\n".format(msg=e))
                            self.kivy_app.label.text = "Error Decoding the Client's Data"
                            return None, 0
            else:
                self.recv_start_time = time.time()

        except BaseException as e:
            print("Error Receiving Data from the Client: {msg}.\n".format(msg=e))
            self.kivy_app.label.text = "Error Receiving Data from the Client"
            return None, 0

    def model_averaging(self, model, best_model_weights_matrix):
        model_weights_vector =
pygad.kerasga.model_weights_as_vector(model=model)
        model_weights_matrix =
pygad.kerasga.model_weights_as_matrix(model=model,
                                                     weights_vector=model_weights_vector)

        new_weights = model_weights_matrix
        for idx, arr in enumerate(new_weights):
            new_weights[idx] = new_weights[idx] + best_model_weights_matrix[idx]
            new_weights[idx] = new_weights[idx] / 2

        model.set_weights(weights=new_weights)

    def reply(self, received_data):
        global keras_ga, data_inputs, data_outputs, model
        if (type(received_data) is dict):
            if (("data" in received_data.keys()) and ("subject" in received_data.keys())):
                subject = received_data["subject"]
                msg_model = received_data["data"]
                print("Client's Message Subject is {subject}.".format(subject=subject))
                self.kivy_app.label.text = "Client's Message Subject is
{subject}".format(subject=subject)

                print("Replying to the Client.")
                self.kivy_app.label.text = "Replying to the Client"
                if subject == "echo":
                    if msg_model is None:
                        data_dict = {"population_weights": keras_ga.population_weights,
                                     "model_json": model.to_json(),
                                     "num_solutions": keras_ga.num_solutions}
                        data = {"subject": "model", "data": data_dict}
                    else:
                        predictions = model.predict(X_test[0])
                        ba = tensorflow.keras.metrics.BinaryAccuracy()
                        ba.update_state(data_outputs, predictions)
                        accuracy = ba.result().numpy()

                        if accuracy == 1.0:
                            data = {"subject": "done", "data": None}
                        else:
```

```
                    data_dict = {"population_weights": keras_ga.population_weights,
                                 "model_json": model.to_json(),
                                 "num_solutions": keras_ga.num_solutions}
                    data = {"subject": "model", "data": data_dict}
                try:
                    response = pickle.dumps(data)
                except BaseException as e:
                    print("Error Encoding the Message: {msg}.\n".format(msg=e))
                    self.kivy_app.label.text = "Error Encoding the Message"
            elif subject == "model":
                try:
                    best_model_weights_vector = 
received_data["data"]["best_model_weights_vector"]

                    best_model_weights_matrix = 
pygad.kerasga.model_weights_as_matrix(model=model,

weights_vector=best_model_weights_vector)
                    if model is None:
                        print("Model is None")
                    else:
                        new_model = tensorflow.keras.models.clone_model(model)
                        new_model.set_weights(weights=best_model_weights_matrix)
                        predictions = model.predict(X_test[0])

                        ba = tensorflow.keras.metrics.BinaryAccuracy()
                        ba.update_state(data_outputs, predictions)
                        accuracy = ba.result().numpy()


                        if accuracy == 1.0:
                            data = {"subject": "done", "data": None}
                            response = pickle.dumps(data)
                            return

                        self.model_averaging(model, best_model_weights_matrix)


                    predictions = model.predict(X_tests[0])
                    print("Model Predictions: 
{predictions}".format(predictions=predictions))

                    ba = tensorflow.keras.metrics.BinaryAccuracy()
                    ba.update_state(data_outputs, predictions)
                    accuracy = ba.result().numpy()
                    print("Accuracy = {accuracy}\n".format(accuracy=accuracy))
                    self.kivy_app.label.text = "Accuracy = 
{accuracy}".format(accuracy=accuracy)

                    if accuracy != 1.0:
```

```
                    data_dict = {"population_weights": keras_ga.population_weights,
                                 "model_json": model.to_json(),
                                 "num_solutions": keras_ga.num_solutions}
                    data = {"subject": "model", "data": data_dict}
                    response = pickle.dumps(data)
                else:
                    data = {"subject": "done", "data": None}
                    response = pickle.dumps(data)

            except BaseException as e:
                print("reply(): Error Decoding the Client's Data: 
{msg}.\n".format(msg=e))
                self.kivy_app.label.text = "reply(): Error Decoding the Client's Data"
        else:
            response = pickle.dumps("Response from the Server")

        try:
            self.connection.sendall(response)
        except BaseException as e:
            print("Error Sending Data to the Client: {msg}.\n".format(msg=e))
            self.kivy_app.label.text = "Error Sending Data to the Client: 
{msg}".format(msg=e)

      else:
         print("The received dictionary from the client must have the 'subject' and 
'data' keys available. The existing keys are 
{d_keys}.".format(d_keys=received_data.keys()))
         self.kivy_app.label.text = "Error Parsing Received Dictionary"
    else:
      print("A dictionary is expected to be received from the client but {d_type} 
received.".format(d_type=type(received_data)))
      self.kivy_app.label.text = "A dictionary is expected but {d_type} 
received.".format(d_type=type(received_data))

  def run(self):
    print("Running a Thread for the Connection with 
{client_info}.".format(client_info=self.client_info))
    self.kivy_app.label.text = "Running a Thread for the Connection with 
{client_info}.".format(client_info=self.client_info)

    while True:
      self.recv_start_time = time.time()
      time_struct = time.gmtime()
      date_time = "Waiting to Receive Data Starting from {day}/{month}/{year} 
{hour}:{minute}:{second} GMT".format(year=time_struct.tm_year, 
month=time_struct.tm_mon, day=time_struct.tm_mday, hour=time_struct.tm_hour, 
minute=time_struct.tm_min, second=time_struct.tm_sec)
      print(date_time)
      received_data, status = self.recv()
      if status == 0:
```

```
                self.connection.close()
                self.kivy_app.label.text = "Connection Closed with
{client_info}".format(client_info=self.client_info)
                print("Connection Closed with {client_info} either due to inactivity for
{recv_timeout} seconds or due to an error.".format(client_info=self.client_info,
recv_timeout=self.recv_timeout), end="\n\n")
                break

            self.reply(received_data)

class ListenThread(threading.Thread):

    def __init__(self, kivy_app):
        threading.Thread.__init__(self)
        self.kivy_app = kivy_app

    def run(self):
        while True:
            try:
                connection, client_info = self.kivy_app.soc.accept()
                self.kivy_app.label.text = "New Connection from
{client_info}".format(client_info=client_info)
                socket_thread = SocketThread(connection=connection,
                                             client_info=client_info,
                                             kivy_app=self.kivy_app,
                                             buffer_size=4096,
                                             recv_timeout=60)
                socket_thread.start()
            except BaseException as e:
                self.kivy_app.soc.close()
                print("Error in the run() of the ListenThread class: {msg}.\n".format(msg=e))
                self.kivy_app.label.text = "Socket is No Longer Accepting Connections"
                self.kivy_app.create_socket_btn.disabled = False
                self.kivy_app.close_socket_btn.disabled = True
                break

serverApp = ServerApp()
serverApp.title="Server App"
serverApp.run()
```

12.4 Keras client code

```
import socket
import pickle
import numpy
import threading

import tensorflow.keras
import pygad.kerasga
import pygad
import pandas as pd
import numpy as np
import kivy.app
import kivy.uix.button
import kivy.uix.label
import kivy.uix.boxlayout
import kivy.uix.textinput

class ClientApp(kivy.app.App):

    def __init__(self):
        super().__init__()

    def create_socket(self, *args):
        self.soc = socket.socket(family=socket.AF_INET,
type=socket.SOCK_STREAM)
        self.label.text = "Socket Created"

        self.create_socket_btn.disabled = True
        self.connect_btn.disabled = False
        self.close_socket_btn.disabled = False

    def connect(self, *args):
        try:
            self.soc.connect((self.server_ip.text, int(self.server_port.text)))
            self.label.text = "Successful Connection to the Server"

            self.connect_btn.disabled = True
            self.recv_train_model_btn.disabled = False

        except BaseException as e:
            self.label.text = "Error Connecting to the Server"
            print("Error Connecting to the Server: {msg}".format(msg=e))

            self.connect_btn.disabled = False
            self.recv_train_model_btn.disabled = True


    def text_submit(self, *args):
        print("Query: ", self.student_text_input.text)


        self.text_submit_btn.disabled = True
        self.connect_btn.disabled = False


    def recv_train_model(self, *args):
        global GANN_instance, best_sol_idx, status

        self.recv_train_model_btn.disabled = True
        self.predict_btn.disabled = False
        recvThread = RecvThread(kivy_app=self, buffer_size=4096, recv_timeout=320)
        recvThread.start()


    def predict(self, *args):
        global model, best_sol_idx, status, keras_ga

        self.predict_btn.disabled = True
        self.recv_train_model_btn.disabled = False


        query =[]

        query.append(self.student_text_input.text)

        df = pd.Series(query)


        import re
        import string

        def remove_punct(text):
            translator = str.maketrans("", "", string.punctuation)
            return text.translate(translator)


        df= df.map(remove_punct)

        import nltk
        nltk.download('stopwords')
        from nltk.corpus import stopwords

        stop = set(stopwords.words("english"))

        def remove_stopwords(text):
            filtered_words = [word.lower() for word in text.split() if word.lower() not in
stop]
            return " ".join(filtered_words)
```

```
df= df.map(remove_stopwords)

print(df)

from collections import Counter

#count unique words
def counter_word(text_col):
    count = Counter()
    for text in text_col.values:
        for word in text.split():
            count [word] += 1
    return count

counter = counter_word(df)

num_unique_words = len(counter)
vocab_size = 598

print(vocab_size)

encoded_messages = [one_hot(d, vocab_size) for d in df]


print(encoded_messages)

max_length = 20
padded_messages = pad_sequences(encoded_messages,
maxlen=max_length, padding='post', truncating='post')



X_train = []

X_train.append(padded_messages)

X_train = np.array(X_train)

model = keras_ga.model

ga_instance = prepare_GA(GANN_instance)
best_sol_idx = ga_instance.best_solution()[2]


predictions = model.predict(X_train)
print("Model Predictions: {predictions}".format(predictions=predictions))
self.label.text = "Model Predictions:
{predictions}".format(predictions=predictions)
```

```
    def close_socket(self, *args):
        self.soc.close()
        self.label.text = "Socket Closed"

        self.create_socket_btn.disabled = False
        self.connect_btn.disabled = True
        self.recv_train_model_btn.disabled = True
        self.predict_btn = True
        self.close_socket_btn.disabled = True

    def build(self):
        self.create_socket_btn = kivy.uix.button.Button(text="Create Socket")
        self.create_socket_btn.bind(on_press=self.create_socket)

        self.server_ip = kivy.uix.textinput.TextInput(hint_text="Server IPv4 Address",
text="localhost")
        self.server_port = kivy.uix.textinput.TextInput(hint_text="Server Port Number",
text="10000")

        self.server_info_boxlayout =
kivy.uix.boxlayout.BoxLayout(orientation="horizontal")
        self.server_info_boxlayout.add_widget(self.server_ip)
        self.server_info_boxlayout.add_widget(self.server_port)

        self.connect_btn = kivy.uix.button.Button(text="Connect to Server",
disabled=True)
        self.connect_btn.bind(on_press=self.connect)

        self.student_text_input = kivy.uix.textinput.TextInput(hint_text="Descrpition of
query", multiline = False)

        self.text_submit_btn = kivy.uix.button.Button(text="Submit query")
        self.text_submit_btn.bind(on_press=self.text_submit)

        self.recv_train_model_btn = kivy.uix.button.Button(text="Receive & Train
Model", disabled=True)
        self.recv_train_model_btn.bind(on_press=self.recv_train_model)

        self.close_socket_btn = kivy.uix.button.Button(text="Close Socket",
disabled=True)
        self.close_socket_btn.bind(on_press=self.close_socket)

        self.predict_btn= kivy.uix.button.Button(text="Query prediction", disabled=True)
        self.predict_btn.bind(on_press=self.predict)



        self.label = kivy.uix.label.Label(text="Socket Status")
        self.box_layout = kivy.uix.boxlayout.BoxLayout(orientation="vertical")
        self.box_layout.add_widget(self.create_socket_btn)
        self.box_layout.add_widget(self.server_info_boxlayout)
        self.box_layout.add_widget(self.student_text_input)
        self.box_layout.add_widget(self.text_submit_btn)
        self.box_layout.add_widget(self.connect_btn)
        self.box_layout.add_widget(self.recv_train_model_btn)
        self.box_layout.add_widget(self.predict_btn)
        self.box_layout.add_widget(self.close_socket_btn)
        self.box_layout.add_widget(self.label)

        return self.box_layout

def fitness_func(solution, sol_idx):
    global keras_ga, data_inputs, data_outputs

    model = keras_ga.model

    model_weights_matrix = pygad.kerasga.model_weights_as_matrix(model=model,
                                                                 weights_vector=solution)
    model.set_weights(weights=model_weights_matrix)
    predictions = model.predict(data_inputs)
    bce = tensorflow.keras.losses.BinaryCrossentropy()
    solution_fitness = 1.0 / (bce(data_outputs, predictions).numpy() + 0.00000001)

    return solution_fitness


def callback_generation(ga_instance):
    global GANN_instance, last_fitness
    population_matrices = 
pygad.gann.population_as_matrices(population_networks=GANN_instance.populati
on_networks,
                                                            population_vectors=ga_instance.population)

GANN_instance.update_population_trained_weights(population_trained_weights=po
pulation_matrices)
    print("Generation = 
{generation}".format(generation=ga_instance.generations_completed))
    print("Fitness    = {fitness}".format(fitness=ga_instance.best_solution()[1]))
    print("Change     = {change}".format(change=ga_instance.best_solution()[1] - 
last_fitness))



def prepare_GA(server_data):
    global keras_ga

    population_weights = server_data["population_weights"]
    model_json = server_data["model_json"]
    num_solutions = server_data["num_solutions"]

    model = tensorflow.keras.models.model_from_json(model_json)
    keras_ga = pygad.kerasga.KerasGA(model=model,
                                num_solutions=num_solutions)

    keras_ga.population_weights = population_weights

    population_vectors = keras_ga.population_weights


    initial_population = population_vectors.copy()

    num_parents_mating = 4

    num_generations = 50
    mutation_percent_genes = 5
    ga_instance = pygad.GA(num_generations=num_generations,
                       num_parents_mating=num_parents_mating,
                       initial_population=initial_population,
                       fitness_func=fitness_func,
                       mutation_percent_genes=mutation_percent_genes)

    return ga_instance

data_inputs = numpy.array([[70, 466, 381, 314,  19, 220,   0,   0,   0,   0,   0,   0,   0,
0,   0,   0,   0,   0,
   0,   0],
                       [67, 218, 232,  95,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
0,
   0,   0]])

data_outputs = numpy.array([4,
                        0])

class RecvThread(threading.Thread):

    def __init__(self, kivy_app, buffer_size, recv_timeout):
        threading.Thread.__init__(self)
        self.kivy_app = kivy_app
        self.buffer_size = buffer_size
        self.recv_timeout = recv_timeout

    def recv(self):
        received_data = b""
        while True:
            try:
                self.kivy_app.soc.settimeout(self.recv_timeout)
                received_data += self.kivy_app.soc.recv(self.buffer_size)
                try:
                    pickle.loads(received_data)
                    self.kivy_app.label.text = "All data is received from the server."
                    print("All data is received from the server.")
                    break
                except BaseException:
                    pass

            except socket.timeout:
                print("A socket.timeout exception occurred because the server did not 
send any data for {recv_timeout} seconds.".format(recv_timeout=self.recv_timeout))
                self.kivy_app.label.text = "{recv_timeout} Seconds of Inactivity. 
socket.timeout Exception Occurred".format(recv_timeout=self.recv_timeout)
                return None, 0
            except BaseException as e:
                return None, 0
                print("Error While Receiving Data from the Server: {msg}.".format(msg=e))
                self.kivy_app.label.text = "Error While Receiving Data from the Server"

        try:
            received_data = pickle.loads(received_data)
        except BaseException as e:
            print("Error Decoding the Data: {msg}.\n".format(msg=e))
            self.kivy_app.label.text = "Error Decoding the Client's Data"
            return None, 0

        return received_data, 1

    def run(self):
        global server_data

        subject = "echo"
        server_data = None
        best_sol_idx = -1
        best_model_weights_vector = None

        while True:
            data_dict = {"best_model_weights_vector": best_model_weights_vector}
            data = {"subject": subject, "data": data_dict}

            data_byte = pickle.dumps(data)

            self.kivy_app.label.text = "Sending a Message of Type {subject} to the 
Server".format(subject=subject)
            try:
                self.kivy_app.soc.sendall(data_byte)
            except BaseException as e:
                self.kivy_app.label.text = "Error Connecting to the Server. The server 
might has been closed."
                print("Error Connecting to the Server: {msg}".format(msg=e))
                break

            self.kivy_app.label.text = "Receiving Reply from the Server"
            received_data, status = self.recv()
            if status == 0:
                self.kivy_app.label.text = "Nothing Received from the Server"
                break
            else:
                self.kivy_app.label.text = "New Message from the Server"

            subject = received_data["subject"]
            if subject == "model":
                server_data = received_data["data"]
            elif subject == "done":
                self.kivy_app.label.text = "Model is Trained"
                break
            else:
                self.kivy_app.label.text = "Unrecognized Message Type:
{subject}".format(subject=subject)
                break

            ga_instance = prepare_GA(server_data)


            ga_instance.run()

            subject = "model"
            best_sol_idx = ga_instance.best_solution()[2]
            best_model_weights_vector = ga_instance.population[best_sol_idx, :]

clientApp = ClientApp()
clientApp.title = "Client App"
clientApp.run()
```

12.5 Decision tree model

```
clf = DecisionTreeClassifier(criterion="gini", max_depth=10, splitter="best")

clf = clf.fit(X_train,y_train)

y_pred = clf.predict(X_test)
```

12.6 Sample data

Query text	Category
I have been experiencing difficulties with my course	0
I find my mathematics module extremely hard	0
My course has been giving me stress due to the difficulty	0
I have experienced serious challenges that have been affecting my performance in my modules	0
I think my course is too much for me to handle	0
The delivery of my course has been poor affecting my grades	0
I don’t think I have the ability to finish my course	0
I cannot cope with the workload	0
The course material puts too much pressure on me	0
I cant meet the expected deadlines	0
My course is affecting my mental health	1
The demand of my course is mentally strenuous	1
I think I’m suffering from depression	1
The pressures of university and covid-19 are taking a toll on my mental	1
I feel like I’m in a constant state of hopelessness	1
I’m not enjoying my course like I did at the start	1
I have been contemplating suicide	1
My course is affecting my sleep and appetite	1
My course makes me constantly anxious	1
My course is mentally draining	1
I am facing abuse at home	2
I can’t focus on my course because I’m experiencing domestic issues	2
My partner is distracting me from my work	2
I’m in an abusive relationship which is affecting my course	2
I’m in a fear-full relationship	2
I’m experiencing verbal abuse at home	2
I’m in a physical relationship which is affecting my studies	2
I’m not able to participate in lectures because of physical challenges at home	2
I’m being denied financial freedom from my partner	2
I have been sexually abused	2
I find it hard to understand my lecturer	3
The language barrier is affecting my degree	3

Making friends is hard because we don’t understand each other	3
I don’t understand the accent of my lectures which makes uni difficult	3
I need aid with language	3
The language barrier is very difficult	3
I can’t understand my lecturer	3
I can’t communicate with my lecturer	3
I cant understand the Birmingham accent	3
I’m having trouble talking to people	3
My financial difficulties are affecting my studies	4
I’m experiencing financial hardship	4
I cant handle my job and uni	4
Balancing university and my job is proving to be extremely difficult	4
My student loan is not enough for me to live through university	4
I was just let go from work and I cant pay my student loan	4
My family can no longer financially support me	4
My student loan has not been transferred to my bank account	4
I don’t have enough money to pay my rent	4
Corona virus has affected my income	4
My family is going through financial difficulties which is affecting me	4
I’m struggling to pay rent	4
I don’t have enough money to pay my tuition fees	4
Due to coronavirus, I lost my job	4
The loan I applied for is not adequate for my needs	4
My financial obligation to my family is hindering my university living	4
My main source of income has been terminated	4
My debts are increasing	4
I don’t have enough money to pay my next instalment of my student loan	4
I lost my job because of the pandemic and now I don’t have income to pay my rent or tuition fees	4
Communication is difficult because I am an international student	3
My speech impediment is a burden to me	3
I need an interpreter for my lectures	3
My English is not good	3
I’m struggling to learn English	3
Communication with my supervisor is difficult because I can’t understand them	3
My English is not good enough to have suitable conversations	3
I find it difficult to understand my lectures accent	3
The language barrier at the university is preventing me from making friends	3
My stammer makes it hard for me to participate during lectures.	3
My parents mistreat me while I do my university work	2
My partner abuses me to the point where I have visible marks	2
Domestic issues at home are affecting my studies	2
My actions are limited at home because of my partner	2
I’m struggling to commute to university because my finances are being controlled by my parents	2

I feel like I’m being emotionally manipulated at home	2
I experience constant shouting and yelling at home which affects my studying	2
I feel like my home is not a safe space for me	2
The dynamic in my house is not suitable for me to carry out my studies	2
I don’t feel comfortable in my current relationship which is affecting my grades	2
Personal family issues are affecting my mental health.	1
I’m Experiencing difficult issues at home which are affecting my mental.	1
My course is draining me.	1
I feel intimidated by my classmates which makes me feel inadequate and makes me mentally overwork myself.	1
My course is so demanding that from time to time it subdues me mentally.	1
Since moving into university I have been constantly lonely.	1
I fear that my course mates judge me because I don’t have the same background knowledge as them.	1
I fear that my mental challenges won’t allow me to complete my course.	1
I am new to the university and its very different and overwhelming from what I am used to.	1
As my course has progressed, I have been feeling numb and stressed.	1
Because of the pandemic I couldn’t access the library therefore limiting my ability for my exams and modules	0
My personal problems are affecting my grades	0
I am struggling with the course workload	0
I feel like I want to change courses	0
I don’t get along with my lectures which affects my ability to complete my work	0
Living far from the university mean I’m always late to my lectures	0
I’m having trouble sticking to my study schedule	0
The coursework deadline is not suitable for me	0
I won’t be able to submit my work in time for the allocated deadline	0
The nature of my course is overwhelming me	0
I've been having difficulties with my course.	0
I've faced significant challenges that have hampered my performance in my modules.	0
I don't think I'll be able to finish my course.	0
I am unable to keep up with the workload.	0
The deliverables of my module assignment are unattainable	0
I won't be able to meet my expected deadlines	0
The course material places an undue amount of pressure on me	0
My business strategy module is to challenging for me, I cannot manage the content.	0
My course is not rigorous enough	0
I feel like I am not being challenged enough by my course	0
My course does not seem to be challenging me enough.	0
I need extra help with my course	0

I have exhausted all my learning resources and still finding it difficult to understand my modules	0
I need academic help	0
I think I need a tutor to help me with my coding module	0
I think I need assistance with my modules	0
I’m struggling and need help with my course	0
I find my modules very hard and challenging	0
I don’t like my course because it is too academically challenging for me	0
I need additional help with my course	0
My mental health is suffering as a result of the pressures of university and covid-19.	1
My class is mentally taxing.	1
My course's requirements are mentally taxing.	1
I believe I am suffering from depression.	1
My course is interfering with my sleep and appetite.	1
I have no free time for myself which is overwhelming me	1
My life feels tedious and repetitive, and I feel like university is the cause	1
I feel alone and isolated at university	1
The university lifestyle is draining and is affecting my mental health.	1
The university lifestyle is exhausting and has a negative impact on my mental health.	1
I think I’m facing mental challenges	1
I need mental health help	1
I think I am experiencing mental health issues and need guidance	1
I think the mental challenges that come with university are affecting my performance	1
Because of the stress of my course, I have gained weight and struggle doing simple tasks	1
Gradually my mental health has been getting worse since I started this course	1
My personality has changed since I started my university course.	1
I think I need therapy to help with my mental issues	1
I think I’m facing depression	1
I’m in a state of mental unrest	1
I feel like I’m in a constant state of depression	1
My partner is denying me financial independence, but I’m too scared to confront them	2
I'm unable to concentrate on my studies because I'm dealing with domestic issues.	2
Due to my domestic challenges, I must seek refuge somewhere else	2
I don’t feel safe in my home	2
I feel like I’m being mistreated at home and I don’t know how to make it stop	2
I believe I am being mistreated at home, and I am unsure how to put a stop to it.	2
Because of physical difficulties at home, I am unable to participate in lectures.	2
I’m being raped at home	2

My parents are financially holding me hostage which is affecting my degree	2
My partner kicks me everyday	2
I get punched by my husband regularly which scares me and my children	2
Weapons are used against me at home	2
I receive threats from my parents which is affecting my health	2
I think I am a victim of domestic abuse	2
I think I need helps with my domestic situation	2
I need domestic help from the authorities	2
I live in a toxic situation which often reduces my focus from my studies.	2
I currently live in a violent household	2
I experience violence on a day to day from my family	2
I’m being harassed at home.	2
I’m being harassed via twitter.	2
I’m being harassed online.	2
My professors' accents are difficult for me to understand, making university difficult.	3
It's difficult to make friends because we don't understand each other.	3
My degree is being hampered by the language barrier.	3
The English language is too challenging for me to learn.	3
English is a difficult language for me to learn.	3
I don’t know anyone who speaks my native language	3
I’m afraid to speak because of my accent	3
Communication is extremely difficult within my classes	3
My grades are being affected because my lecturer doesn’t understand me	3
I can’t take part in group work because of the language barrier	3
Because of the language barrier, I am unable to participate fully in group projects.	3
I think I might need a translator for my lectures	3
I need someone to translate my lectures because I don’t understand a word being said	3
I need language help as I can only speak my native language	3
I can’t effectively take part in group work because my group can not understand me.	3
Struggling to learn English	3
I can’t understand English	3
Don’t understand English	3
I only speak Spanish	3
There is a big language barrier at the university that is preventing me from socialising.	3
My financial difficulties are interfering with my studies.	4
I am financially burdened	4
My student loan has not yet been deposited into my bank account.	4
Balancing university and my job is proving extremely difficult.	4
My income is not enough to support me through my studies	4
I have been laid off work due to the pandemic and have no means of financial support	4

I need a job to help fund my tuition	4
I can't pay my rent because I don't have enough money.	4
Since joining university my expenses have increased and my family are finding it hard to support me	4
I have lost contact with my financial sponsor	4
I need financial aid	4
I think I might have to leave university because I can’t financially cope with my debts	4
My student loan is insufficient to cover my living expenses during my time at university.	4
My financial difficulties are affecting my studies	4
My student loan has not yet been deposited into my bank account and I must pay for this term.	4
My family is no longer able to financially support me.	4
I don’t think I am financially literate enough to manage my living expenses	4
The pandemic has affected my income and has now made living at university challenging.	4
I don’t have the financial support to research placements away from where I live.	4
I desperately need financial advice.	4
I have difficulty grasping the course material	0
I find attending lectures difficult	0
I am struggling to go to my lectures	0
I think I have poor study habits which are affecting my learning	0
Because of the time zone difference I find it hard to attend lectures	0
My lectures are very late at night and since I am a international student this is affecting my education	0
My computer dosent have the requirments too accommodate for the softwares I need to use for my course	0
Due to my lack of experiece in mathematics I am struggling with my mathematics module	0
I want to withdraw from University	0
I don’t think University is for me after experiencing my course	0
The increase in difficulty from college to University is too much for me to handle	0
My lectures don’t provide additional help on my course	0
The lecturer for my product management moudule is very nonchalant and not very helpful	0
My lectures don’t pay me enough attention during class	0
Im not intrested in my course anymore	0
I am extremley confused in my marketing lectures	0
Im performing poorly in my course	0
My grades are not reflecting my effort	0
I have too many assignments, I don’t think I can complete them	0
I don’t think I am achieving the grades I should be and im not sure how to improve them.	0
I feel very anxtious which is causing me to procrastinate	1
Some of my modules are overwhelming me and making me feel anxtious	1

Since I have started University I feel like I have no control over my life.	1
I experience feelings of hopelessness and pessimism	1
I feel like University is causing me to lose intrest in my hobbies	1
I have noticed a drop in my energy since I started my course and my friends and family have been worrying about me	1
My friends and family are worried about my mental health	1
Im constantly sad due to my course	1
I think im experiecing mental health issues and I have no one to talk to abou it	1
I have a unwillingness to communicate with others	1
I find myslef regularly withdrawring myself from social situations such as group work.	1
I seem to be neglecting routine tasks, work and personal hygine and im not sure why.	1
I feel like I am a burden to others	1
For the past two month I have been experiencing mood swings which seem to be hindering my life	1
Ever since I started my course life has been feeling meaningless	1
I get teased in me classes because of my accent which is affecting my mental health	1
My relationships with my family and friends are collapsing and I think it sbecause of university but im not sure what to do	1
Im feeling homesick	1
My appetite has significantly decreased due to my mental concerns and im truggling to cope with it.	1
Im struggling with addiction and im not sure who to speak too about it	1
I have been receving threats at home	2
My partner is verbally abusive	2
My partner threatesn my life and I now feel like im not in a safe enviroment to carry out my studies	2
I am being mentaly manipulated at home	2
I think im being sexally assulted by my housemate	2
Im currently in a fearfull relationship at home and im ot sure who to talk too.	2
My partner held me down and shouted at me and when I struggled to get loose, he hit me.	2
My dad regularly insults and ridicules the women in our house	2
I cant make my way to university because my partner has taken my keys and phone	2
my parents have restricted my access to toiletries and and medication at home	2
I constanly get harrassed about imagined affairs by my partner	2
My girlfriend is manipulating me with lies and contradictions in our shared accomodation	2
My partner spits at me when I don’t agree with him and I feel unsafe	2
My bank accounts are in my parents name	2
My partner is forcing me to sign immigration documents	2
Where can I go to talk about domestic abuse issues?	2

My family are denying my right to work whilst I am in university	2
My partner forces me to have sex	2
My parents throw object at me at home and this is affecting my education.	2
My partner misuses my name for finacial reasons	2
I don’t understand the topic of most conversation in University because of the language barrier	3
All the English speakers on my course cant unerstand me.	3
I cant express my ideas and opinions because I can't speak English properly	3
I don’t think my English teacher is qualified to teach me however I am unsure as I am not an English speaker my self.	3
I live in a house with international students so I feel like im not able to practice my English thoroulgy	3
My lecturer dose not cater to the foregin speakers in out class	3
I only understand one of my lecturers in my course because he speaks my language	3
There are many grammatical errors in my essays because I don’t have a full grasp of the English language	3
I find it extremley hard to give oral reports in English	3
I can not communicate with anyone on campus due to the language barrier.	3
I feel like I am not learning anything from my course because of the language barrier.	3
The demands to study my course and learn English is too much.	3
I suffer with lack of self-confidence because I am struggling to learn English	3
I am struglling to learn English	3
I find speaking English fine however I struggle with writing in English	3
Formulating my opinions to critique readings is extremley challenging	3
I just moveed to the UK for University and I am struggling to undersatnd the different accents.	3
Im receiving a poor learning experience because I cant understand English	3
My English not very good	3
My English bad	3
I struggle with finacial discipline	4
Due to the recent increase in energy prices I am struggling to pay my hose bills	4
I don’t have enough money to feed myself	4
Im finding it hard to keep up with my tuition instalments	4
Im struggling to find a part-time job	4
I don’t have enough money to fund my placement	4
I cant pay for necessities	4
My sponsor can no longer finacially support me.	4

12.7 Instructions

To execute code written in this project it is instructed to paste the code into a python interpreted such as Jupyter notebook. Sample data should be pasted on to a Microsoft excel document and placed in the same folder as the server and client code files. Serve code should be run first followed by the client code. On the server, buttons should be executed in the order of; create socket, bind socket, listen to connections. Once server instructions have been executed client instructions should be executed as follows; create socket, type in text query on input widget, submit query, connect to server, receive and train model. When prediction error has reached a value of zero on the server end, the query prediction button should be executed on the client end to produce a classification of the inputted query. Once prediction has been received on the client end the close socket button on both the server and the client can be executed to terminate the app.