



**University of West Attica**  
**School of Engineering**  
**Department of Biomedical Engineering**  
**MSc program “Biomedical Engineering and Technology”**

# **A Speech-Based Approach for Depression Detection**

**BARDAKI AIKATERINI**

**bmet07**

**Supervisor**

**LUÍS COELHO**

**Athens 08 / 10 / 2024**

The Three-Member Examination Committee

Luis Coelho

Instituto Superior de  
Engenharia do Porto,  
Polytechnic of Porto,  
Portugal

Signature

Dimitrios Gklotsos

Department of  
Biomedical Engineering,  
University of West Attica,  
Greece

Signature

Spiridon Kostopoulos

Department of  
Biomedical Engineering,  
University of West Attica,  
Greece

Signature

## DECLARATION BY THE AUTHOR OF THE DIPLOMA THESIS


The signatory Aikaterini Bardaki of Nikolaos and Rozalia, with registration number bmet07, student of the MSc Program "Biomedical Engineering and Technology" of the University of West Attica, I declare responsibly that:

"I am the author of this Diploma Thesis and any help I had for its preparation is fully recognized and referenced. Also, any sources from which I have used data, ideas, or words, whether exact or paraphrased, are listed in their entirety, with full reference to the authors, the publisher, or the journal, including any sources that may have been used by the internet. I also certify that this work has been written exclusively by me and is a product of intellectual property of both myself and the University of West Attica.

Violation of my above academic responsibility is an essential reason for the revocation of my diploma ".

**Date: 08/ 10 / 2024**

**Signature**

A handwritten signature in black ink, appearing to read 'Aikaterini Bardaki', written over a horizontal line.

# CONTENTS

Abstract.....	6
Περίληψη.....	7
Acknowledgments.....	9
1 Introduction .....	10
1.1 Motivations.....	10
1.2 Objectives.....	10
1.3 Document Organization.....	11
2 Bibliographic Review of Depression.....	12
2.1 Symptoms .....	12
2.2 Screening.....	13
2.3 Treatment .....	13
2.4 Depression Categories .....	13
2.5 Clinical Evaluation Scales .....	14
2.6 Reverse Inference .....	14
2.7 Effects on speech .....	15
3 Bibliographic Review of Speech-Based Depression Classification .....	16
3.1 Machine Learning Strategies.....	16
3.2 Evaluation and Interpretability Methods.....	23
3.3 Bioacoustic and Linguistic Features .....	25
3.4 Related Work .....	27
4 Materials, Methods and Tools .....	29
4.1 Dataset.....	29
4.2 COVAREP.....	29
4.3 Methods and Tools .....	31
5 Development.....	32
6 Results.....	37
6.1 Performance Evaluation.....	37
6.2 Evaluation Scenarios and Interpretability Metrics.....	43
7 Discussion.....	50
8 Conclusion and Future Work .....	52
Appendix 1: SVM Script.....	57
Appendix 2: NN Script.....	66



## Abstract

Depression is a widespread mental health disorder with severe impacts on quality of life. Current screening relies mainly on psychiatrist evaluations, lacking technological support. Advances in AI now enable rapid pattern recognition, allowing researchers to leverage speech analysis for depression detection. As a result, the present thesis aims to a) compare different machine learning models in terms of their classification accuracy in separating depressed and non-depressed people on voice data obtained from the DAIC-WOZ dataset, b) detect significantly important features that contribute to speech patterns related to depression, and c) fine-tune the optimal machine learning model found in stage a) ensuring accurate and robust results for real-world data.

The used dataset is part of the Distress Analysis Interview Corpus (DAIC) from the University of Southern California, which aids in diagnosing depression. It consists of 189 English interview sessions with pre-extracted voice features, processed with the Cooperative Voice Analysis Repository (COVAREP) toolbox. In the present thesis, these features were processed via Python programming with the Anaconda Distribution package.

The dataset was originally divided into training, validation, and testing sets, but a new split of 80% training and 20% testing was chosen. The 74 pre-extracted COVAREP features represent time-series data, which created a large matrix that posed computational challenges. Two approaches were employed to address these: (1) selecting core features (7000 middle rows of each session) and (2) aggregating time series data into four statistical features per feature, reducing it to one row per participant. Both approaches included preprocessing steps to handle missing or infinite values and standardization. In the first approach, two dataset versions were tested: one non-balanced and one balanced through row deletion (392000 rows for each class). Models evaluated included Neural Networks, Convolutional Neural Networks, Long Short-Term Memory, AdaBoost, Multilayer Perceptron, and Decision Tree. In the second approach, the data balanced with the SMOTE technique was used to evaluate the Support Vector Machine (SVM) algorithm on the aggregated features.

The SVM demonstrated the best performance across scenarios, achieving 81% accuracy, 79% precision, 90% recall, 74% F1-score, and 72% specificity. Interpretability tools (LIME, SHAP, PDP) identified three key features contributing to the model's predictions.

**Keywords:** *depression, voice detection, machine learning*

## Περίληψη

Η κατάθλιψη είναι μια ευρέως διαδεδομένη διαταραχή της ψυχικής υγείας με σοβαρές επιπτώσεις στην ποιότητα ζωής. Ο σημερινός έλεγχος βασίζεται κυρίως σε αξιολογήσεις ψυχιάτρων, χωρίς κάποια τεχνολογική υποστήριξη. Οι εξελίξεις στην τεχνητή νοημοσύνη επιτρέπουν πλέον την ταχεία αναγνώριση μοτίβων, επιτρέποντας στους ερευνητές να αξιοποιήσουν την ανάλυση ομιλίας για την ανίχνευση της κατάθλιψης. Ως εκ τούτου, η παρούσα διπλωματική εργασία αποσκοπεί α) στη σύγκριση διαφορετικών μοντέλων μηχανικής μάθησης όσον αφορά την ακρίβεια ταξινόμησής τους στο διαχωρισμό καταθλιπτικών και μη καταθλιπτικών ατόμων σε δεδομένα φωνής που ελήφθησαν από τη βάση δεδομένων DAIC-WOZ, β) στον εντοπισμό στατιστικά σημαντικών χαρακτηριστικών που συμβάλλουν σε μοτίβα ομιλίας που σχετίζονται με την κατάθλιψη και γ) στην προσαρμογή του βέλτιστου μοντέλου μηχανικής μάθησης που βρέθηκε στο στάδιο α), εξασφαλίζοντας ακριβή και αξιόπιστα αποτελέσματα για δεδομένα πραγματικού κόσμου.

Το σύνολο δεδομένων που χρησιμοποιήθηκε είναι μέρος της βάσης δεδομένων Distress Analysis Interview Corpus (DAIC) του Πανεπιστημίου της Νότιας Καλιφόρνιας, το οποίο βοηθά στη διάγνωση της κατάθλιψης. Αποτελείται από 189 συνεδρίες στην αγγλική γλώσσα με έτοιμα εξαχθέντα χαρακτηριστικά φωνής μέσω του Cooperative Voice Analysis Repository (COVAREP). Στην παρούσα εργασία η επεξεργασία τους έγινε μέσω προγραμματισμού Python με το Anaconda Distribution.

Τα δεδομένα ήταν ήδη χωρισμένα σε σύνολα εκπαίδευσης (training), επικύρωσης (validation) και δοκιμής (testing), αλλά επιλέχθηκε ένας νέος διαχωρισμός 80% εκπαίδευση και 20% δοκιμή. Τα 74 ήδη εξαχθέντα χαρακτηριστικά COVAREP αποτελούν δεδομένα χρονοσειρών, τα οποία δημιούργησαν έναν μεγάλο πίνακα που παρουσίασε υπολογιστικούς περιορισμούς. Για την αντιμετώπισή τους, χρησιμοποιήθηκαν δύο προσεγγίσεις: (1) η επιλογή των βασικών χαρακτηριστικών (7000 ενδιάμεσες γραμμές από κάθε συνεδρία) και (2) η συγκέντρωση των δεδομένων χρονοσειρών σε τέσσερα στατιστικά χαρακτηριστικά ανά χαρακτηριστικό, μειώνοντάς τα αποτελέσματα του πίνακα σε μία γραμμή ανά συμμετέχοντα. Και οι δύο προσεγγίσεις περιλάμβαναν βήματα προ-επεξεργασίας για την διαχείριση των μηδενικών ή άπειρων τιμών και την κανονικοποίηση. Στην πρώτη προσέγγιση, δοκιμάστηκαν δύο εκδοχές: μία με μη-ισορροπημένα δεδομένα και μία με ισορροπημένα μέσω της διαγραφής γραμμών για την εξισορρόπηση τους (392000 γραμμές κάθε κλάση). Τα μοντέλα που αξιολογήθηκαν περιλαμβάνουν τα Νευρωνικά Δίκτυα, Συνελικτικά Νευρωνικά Δίκτυα, Long Short-Term Memory, AdaBoost, Multilayer Perceptron, και τα Δέντρα Αποφάσεων. Στη δεύτερη προσέγγιση, τα δεδομένα που εξισορροπήθηκαν με την τεχνική SMOTE χρησιμοποιήθηκαν για την δοκιμή και την αξιολόγηση του αλγορίθμου Support Vector Machine (SVM).

Το SVM επέφερε την καλύτερη απόδοση από όλα τα σενάρια, επιτυγχάνοντας 81% ακρίβεια (accuracy), 79% ευστοχία (precision), 90% recall (ευαισθησία), 74% F1-score και 72% εξειδίκευση (specificity). Μέσω εργαλείων που υποστηρίζουν την ερμηνεία των αποτελεσμάτων (LIME, SHAP, PDP) εντοπίστηκαν τρία στατιστικώς σημαντικά χαρακτηριστικά.

*Λέξεις κλειδιά: κατάθλιψη, φωνητική ανίχνευση, μηχανική μάθηση*



## Acknowledgments

I would like to express my deepest gratitude to my supervisor Professor Luis Coelho for his invaluable support and guidance throughout this work, his expertise and mentorship have been essential to my academic path. I also appreciate the resources and organization of the University of West Attica for providing a supportive and resourceful environment, as well as to my professors and colleagues for their continuous encouragement, innovative insights, and for equipping me with the knowledge and skills necessary for the next steps in my career. Finally, I am thankful to my family and friends for their support and belief in me throughout my academic journey, their encouragement has been a constant source of strength and inspiration.

# 1 Introduction

## 1.1 Motivations

Statistical metrics regarding depression show a robust increase in cases over the recent years due to various social, financial, and possibly political issues, and even more after the pandemic of COVID-19. Unlike other health conditions, such as those related to cardiovascular, musculoskeletal, or digestive systems, etc., in mental health disorders, the signs are most of the time very complex and difficult to be stabilized. Depending on the severity the appropriate treatment needed is different and there are many cases that can even lead to suicide.

On the other hand, due to technological advances and AI systems, computational power increased while computational time decreased significantly, even for complex algorithms. Moreover, the amount of available data online has opened the path for scientists to research the topic of depression and detect patterns via machine learning. The scientific community has insisted on researching the most efficient and accurate ways of screening depression based on a variety of different metrics, such as imaging, biochemical markers, text, voice, and facial expression.

Finally, the growing need for telehealth solutions, even more in underserved and remote areas, and the above-mentioned gap in the detection of depression combined with the advance of machine learning became motivating factors to conduct this present thesis.

## 1.2 Objectives

The general vision of this research is to develop an automated early-diagnosis depression tool for psychiatrists that can support medical diagnosis, helping to face the rise of depression by using speech analysis. What is also important for this system, is to be a non-invasive and cost-effective alternative that complies with ethical regulations, data privacy standards, and confidentiality respecting patient autonomy.

The main objectives of this current thesis are:

- a) Compare different machine learning models in terms of their classification accuracy in separating depressed and non-depressed people on voice data obtained from the DAIC-WOZ dataset
- b) Detect significantly important features that contribute to speech patterns related to depression
- c) Fine-tune the optimal machine learning model found in stage a) ensuring accurate and robust results for real-world data

### 1.3 Document Organization

This thesis is structured in seven chapters, wherein another perspective of the current topic contributes to a better understanding of the final results. In more detail, the present document is organized as follows:

Chapter 1. Introduction: The first chapter describes the motivations, objectives as well as the organization of this present thesis.

Chapter 2. Bibliographic Review of Depression: In this chapter, depression is analyzed in its different aspects, covering demographic data, common symptoms, ways of screening and treatment as well as the obstacles deriving from the nature of the disease, and finally the affection that it has on speech.

Chapter 3. Bibliographic Review of Speech-Based Depression Classification: In this part of the thesis, machine learning methodologies regarding depression recognition are researched and all the theoretical background of the methods used in the present research is provided. In addition, bioacoustic and linguistic features that address this situation are examined. Finally, a table summarizes the related work and mentions the most important information, including the dataset/cohort, the model, and the performance of each research.

Chapter 4. Materials, Methods and Tools: Chapter 4 analyses the Dataset and the repository COVAREP, used in the project development, to provide a better understanding of the sample of this thesis. Furthermore, the computational background that supported the implementation of the methods is discussed.

Chapter 5. Development: This chapter describes the path taken to support the objective of this thesis, including the proposed methodologies, the obstacles encountered along the way, and the solutions that led to the final results.

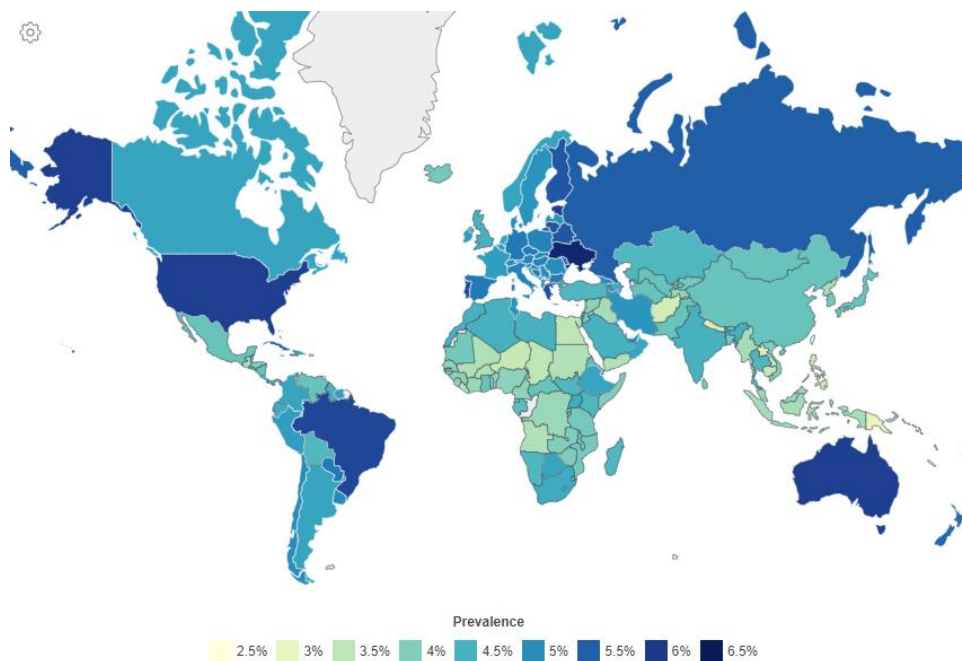
Chapter 6. Results: In the sixth chapter, the results of the methodologies' performance metrics are reported in detail in accordance with the relative explanations. In addition, the interpretability tools provide plots to evaluate the highest impact and significantly important features.

Chapter 7. Discussion: In this chapter, the results of the development of the machine learning methodologies and the research findings are discussed and evaluated.

Chapter 8. Conclusion and Future Work: This final part of the thesis summarizes the key findings of the research, and proposes potential directions for further development in the field.

## 2 Bibliographic Review of Depression

Psychological diseases especially depression are topics of high interest nowadays due to the increased rates of cases over the years in the global population. More specifically, the COVID-19 pandemic has boosted the global rates by 25% according to the World Health Organization. It is also estimated by the Global Health Data Exchange that 251-310 million people worldwide suffer from depression which means around 3.4% of the global population. Additionally, studies have shown that 1 to 6 people have experienced depression at least one time in their lives. More specific rates regarding the prevalence of depression globally are shown in the following image (Figure 2.1). It is important to mention that this increase is also caused by the fact that more and more people are becoming more sensitive to examining and treating their mental health. Nonetheless, in many countries, especially in the lower or mid economies, people do not have the proper means to diagnose it or major issues like hunger, poverty or war put their psychological condition as a lower priority [1] [2]. But how does the depression diagnosis take place?



**Figure 2.1:** Global prevalence of depression [2]

### 2.1 Symptoms

Depression symptoms vary from changes in appetite, loss of energy and interest in activities, difficulties in sleep and concentration to a persistent feeling of sadness, negativity, lack of confidence, and suicide or self-harm thoughts. Other symptoms also affect a person's movements speech and expressions and can be used as methods of diagnosis. Nevertheless, these symptoms can be connected with additional pathologies such as hormone disorders caused by other diseases like thyroid problems, lack of vitamins, or other biochemist

unbalances or can be misinterpreted with other mental disorders, Central nervous system disorders [3]. These indications are also affected by the unique characteristics of each person's personality and external environment.

## 2.2 Screening

The screening of mental disorders is mainly based on psychiatrists' diagnoses. Individuals are conducting clinical interviews with mental health care professionals whose diagnosis is connected with established diagnostic criteria based on discussion, metrics, and questionnaires. In addition, studies have shown, only in major depressive disorders, strong indexes for structural differentiations in individuals' brains due to severe depression episodes but, to the best of our knowledge, no structural alteration shown by Magnetic Resonance Imaging (MRI) can be used as a metric due to the biological complexity of this specific disorder and have no clinical utility [3].

## 2.3 Treatment

Once individuals are diagnosed with depression, their treatment can be carried out in different ways. One of the most common paths is the anti-depressant medication that stabilizes the brain's chemistry and varies among individuals. Another method, most of the time combined with the previous one, is psychotherapy. Brain electrical stimulation known as electroconvulsive therapy (ECT) is a way to treat very severe cases of depression. Undoubtedly, the management is different depending on the category [1].

## 2.4 Depression Categories

There are different categories depending on the duration as well as the severity of the symptoms.

- Major depression is when the symptoms are present for more than two weeks and influence the functionality of daily life.
- Seasonal affective disorder is when the symptoms are present depending on the season, usually during winter or autumn, and calm down after the passing of the season.
- Perinatal or postpartum depression is known in women during or after pregnancy accordingly.
- Persistent depressive disorder stands for cases with less intense symptoms but longer duration than two years also known as dysthymia.
- Depression with psychotic symptoms is a more severe case in which the individual faces visual or audio delusions.
- Manic depression is a bipolar disorder in which one comes across both depressive and manic symptoms [4].

## 2.5 Clinical Evaluation Scales

The need to categorize the severity of depression into a quantitative index led to the scale's creation. The two types of extracting the information are mentioned, the observer-rated scale and the self-rated scale.

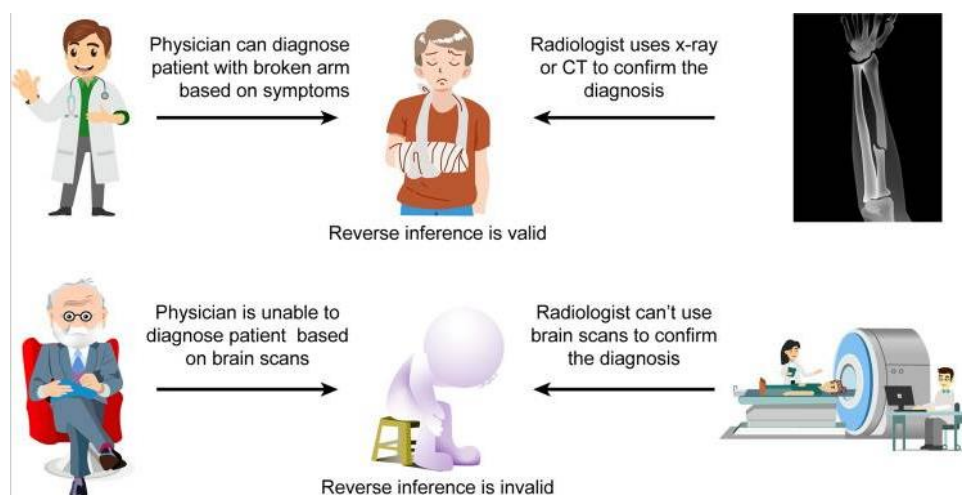
Hamilton scale (HAMD-17) is the most commonly used scale for a clinician rating, lasts 20-30 minutes, and includes questions for depression symptoms, anxiety, and side effects from drug treatment. The person conducting the interview needs to be well-trained and play an important role in the reliability of the results [5].

On the other hand, the Patient Health Questionnaire (PHQ-9) scale interprets another approach using the contribution of the person assessing themselves to detect depression. It has been proven to be a reliable and quick method [5]. Yet, these two scales used for clinical diagnosis of depression still have the objectivity that lies in the individual's recalling to answer.

Other scales, such as Beck Depression Inventory, DASS-21, or CES-D are also used for depression evaluation but their use is less prevalent.

## 2.6 Reverse Inference

Reverse inference is common in clinical diagnosis of most pathologies but it mostly fails when it comes to psychiatry (Figure 2.2). Providing an example of a patient with a broken hand who visits an orthopaedist, the clinician can come to a conclusion about the patient's issue based on symptoms and by confirming using an x-ray. On the other hand, in the case of an individual suffering from depression and addressing a doctor, the psychiatrist is unable to conclude validly neither based only on their symptom recall nor via MRI evidence [3].



**Figure 2.2:** The role of reverse inference in psychiatry [3]

The lack of depression screening technologies has pushed scientists' interest to orient their research in various parameters –more than a decade now– such as EEG signals, facial

expressions, eye gaze and pose of the head, and word and voice identification parameters to support clinical assessment. The latter has proved notable results in clean lab-based datasets and being at the same a non-invasive technique promises an easy and efficient tool for detecting depression indicators. Certainly, real-world applications have brought limitations that open new ways of categorizing features which are going to be discussed in more detail in the following chapters [6].

## 2.7 Effects on speech

The rise of speech analysis as a biomarker for supporting clinical depression detection seems to promise various benefits, such as improvement in the objectivity of conclusions, remote ability in individual monitoring, as well as cost and time reduction. Speech analysis can provide both linguistic and paralinguistic information. Most studies have shown that paralinguistic clues are inherently harder to compare than linguistic ones have more clear differences and provide more functional diagnostic tools. Nevertheless, automated speech-based techniques can gather information from both sides and enhance them with other metrics such as video and come up with robust and valid conclusions analyzed by machine and deep learning models [7].

The biological process of speech production is mainly characterized by three basic stages. It starts with the cognitive stage of creating phonetic and prosodic information in the brain. This process includes the activation of certain brain pathways responsible for supporting both phonological, syntactic processing and semantic processing. The next stage is the physiological stage in which around 100 muscles are temporally coordinated to move the articulators by receiving the proper signals from the motor nuclei to finally generate the proper speech sounds. The tongue, lips, lower jaw, and velum are articulators whose role is to be reshaped and create resonances. The air comes from the lungs and through the glottis and vibrates the vocal fold (glottal flow pulses). The third stage is the vocal result emitted from nasal and oral cavities [7].

One of the most widely known indicators of depression is the psychomotor retardation which can influence thoughts, physical movements, eye movements, facial expressions, and also speech. Regarding speech, studies have shown many significant changes such as slowing down the pitch rate, monotony, larger pauses, reduced tone, and changes in articulation. Various complex neurophysiological changes occur when an individual suffers from depression, mainly connected with the reduction of dopamine neurotransmission to the basal ganglia and striatum that control vocal movements. A high number of research works have proved that this reduction can be linked with anhedonic behavior and also influence speech coordination [8].

In addition, several studies related to other neuropsychiatric disorders like Parkinson's disease, schizophrenia, and Huntington's disease have shown psychomotor changes. In all of them, the basal ganglia take the major role in coordinating the motor process under the support of the neurotransmitters [9].

### 3 Bibliographic Review of Speech-Based Depression Classification

Taking a look at the past, in 1986 one of the first research projects on emotion detection using speech was conducted by Bezooijen and Tolkmitt using statistical tools. [10] Then, in 1999 one of the first systems that connects emotion with speech using a specific model was proposed by Moriyama [11]. Later a rise of studies due to computers' evolution started using models like Gaussian Mixtures Models (GMM), Random Forest, Support Vector Machine, and Hidden Markov Models (HMM) to achieve the best classification. Nowadays, these methods are called traditional machine learning methods, and the selected features define the accuracy of the results. These techniques are well-performed for a few amount of data. Deep learning is the latest technique and it is possible to be implemented due to higher computing speed. It handles a large amount of data and it is a sub-category of Neural Networks (NN) [12].

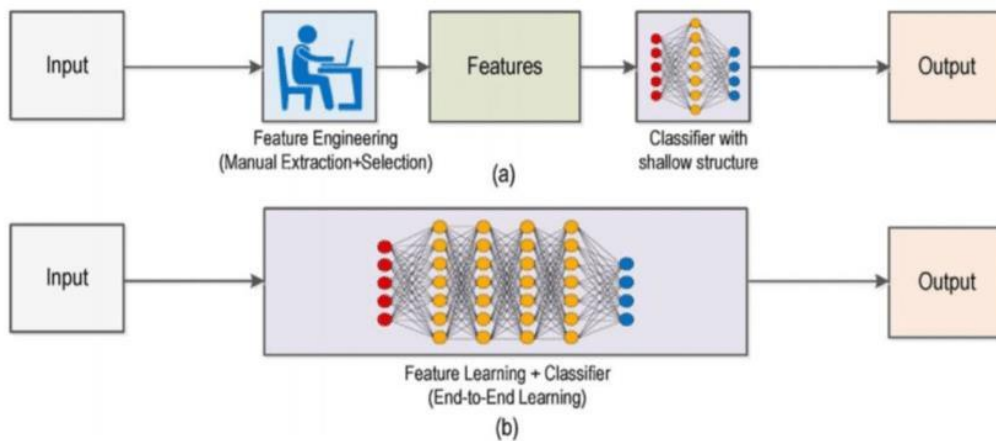
The challenging need for an objective diagnostic tool for depression has led the research community to examine various possible methodologies of automatic detection and speech has been proven to be a powerful tool and mostly more effective than visual and textual data. However, they are often combined with facial and textual to make an enhanced multimodal approach [13].

#### 3.1 Machine Learning Strategies

According to the present bibliography, we can divide the current approaches of the depression detection techniques into two categories, hand-crafted feature-based (traditional machine learning) and deep learning-based. Deep neural networks (DNN) in literature have achieved impressive results and outperformed the hand-crafted feature-based approach in the detection of neuropsychological disorders [14]. More specifically in the case of depression, Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) were both proven to be effective. The former technique used in non-sequential visual data has shown impressive results yet with no temporal ability for sequential data while the latter can provide temporal information in sequential data, although it is not yet clear which one performs better [13].

More specifically, the deep learning techniques can also be divided into 2 different subcategories (figure 3.1.). The first one is using feature engineering meaning a manual extraction of features and the other is called end-to-end. The basic difficulty that machine learning techniques face is the choice of suitable features. It is a fact that depending on the dataset different features have shown better performance [13]. The end-to-end approach aims to overcome this obstacle by reducing complexity when automatically learning from the input of the raw data to extract features without the need for human intervention, and it has proved to produce impressive results [15] [16]. A schematic illustration is shown in Fig. 3.1. In the study of Srimadhur N.S, Lalitha S [17], spectrogram-based CNN and proposed end-to-end CNN are compared using AVEC 2016 DAIC-WOZ dataset for validation. The proposed method indicates better performance than the spectrogram-based.





**Figure 3.1:** Machine learning pipelines for feature-based approach (a) and end-to-end approach (b) [18].

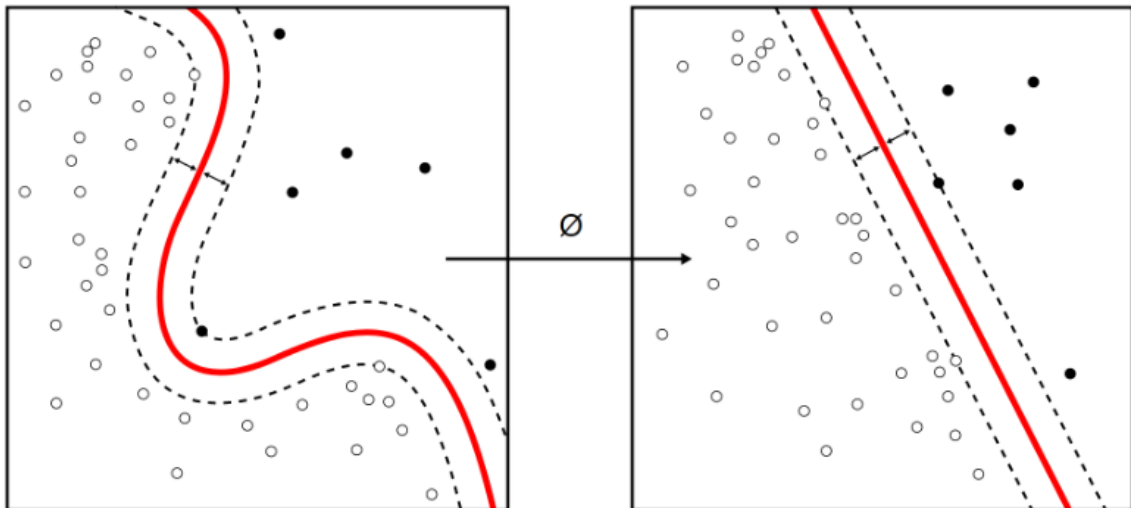
In the present research, several methodologies including traditional machine learning and deep learning, were used comparatively to detect the most accurate performance on the analysis of already extracted features. These were the following:

### Support Vector Machine

Support Vector Machine (SVM) is a supervised model that works well for both classification and regression applications. The fundamental concept that supports (SVM) is the hyperplane. Hyperplanes work as boundaries for dividing the data into classes and according to the dimensionality they are presented as a line (two-dimensional space) or a plane (for more than two dimensions). In addition, the decision function determines in which side of the hyperplane the new data points are placed. Their classification is conducted according to the optimal boundary for maximum margin between classes. [19]

In more detail, the margin separates the nearest data points and the maximum-margin hyperplane achieves the best prediction. Soft margin is also important because it allows the method to give some extra space to deal with errors. Additionally, the kernel function is a mathematical tool used to increase the existing dimensionality of the data so they become linearly separable as it appears schematically in Figure 3.2. The choice of kernel is critical for each dataset and can be either linear, or polynomial, sigmoid, Gaussian, etc [20] [21].

Overall, the SVM's ability to maximize the margin and utilize kernel functions makes it a robust tool for both classification and regression tasks, including applications in voice detection and recognition.



**Figure 3.2:** The kernel function [21]

### AdaBoost

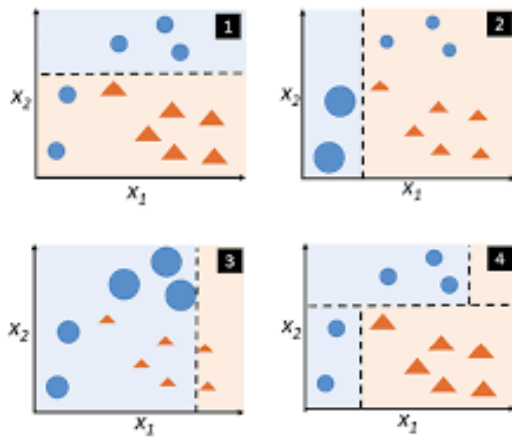
Adapting Boosting or shortly AdaBoost algorithm is an ensemble approach of Freund and Schapire who based their idea on combining a set of weak classifiers to form a single stronger classifier.

More specifically, if we regard an input pattern  $x_i$  for each  $k_j$  classifier to form a decision function  $C(x_i)$  corresponding to the weights  $a_1, a_2, \dots, a_l$ , assuming the training vectors are separated into two classes  $k_j(x_i) \in \{-1, +1\}$ : [27] [22]

$$C(x_i) = a_1 k_1(x_i) + a_2 k_2(x_i) + \dots + a_l k_l(x_i) \quad (1)$$

The weak learners are adjusted according to an error function after each each iteration. This adjustment involves decreasing the weights of the correctly classified samples and increasing the weights of the misclassified ones, thereby iteratively enhancing the classification performance.

In the example of Fig. 3.3, blue triangles and orange squares represent features whose weight is according to the size. In the first diagram, all features carry equal weight until correctly classified features are down-weighted and incorrectly classified feature are increasing their weight respectively, as shown in the second diagram. In the third diagram, the same process is repeated. Finally, on the last diagram, a strong classifier is produced [23].



**Figure 3.3:** A schematic of the AdaBoost method [23]

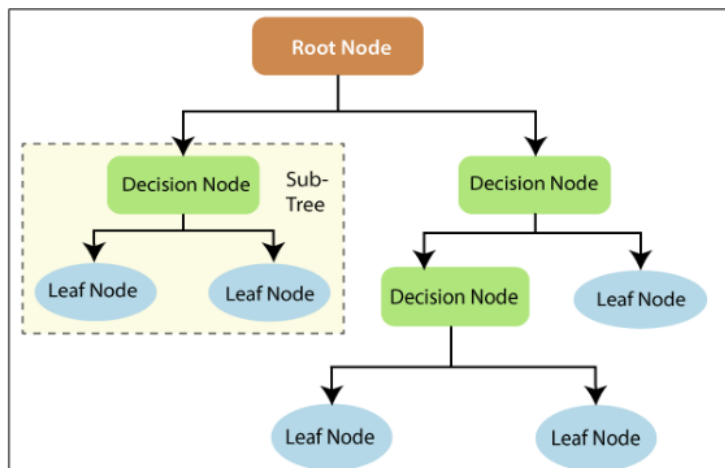
In conclusion, the AdaBoost algorithm's iterative reweighting mechanism and combination of weak classifiers make it a powerful tool for improving classification accuracy.

### Decision Trees

The Decision Tree is one of the most popular supervised algorithms for either classification or regression tasks.

Practically, nodes and branches compose a tree and each internal node represents a test on an attribute, and the branch the result of the test, while each leaf node corresponds to a class label (classification) or a continuous value (regression) as shown in Figure 3.4. [24].

Firstly, the best attribute is defined which divides the dataset into classes. Until no longer splitting is possible, the data are divided into subsets recursively.



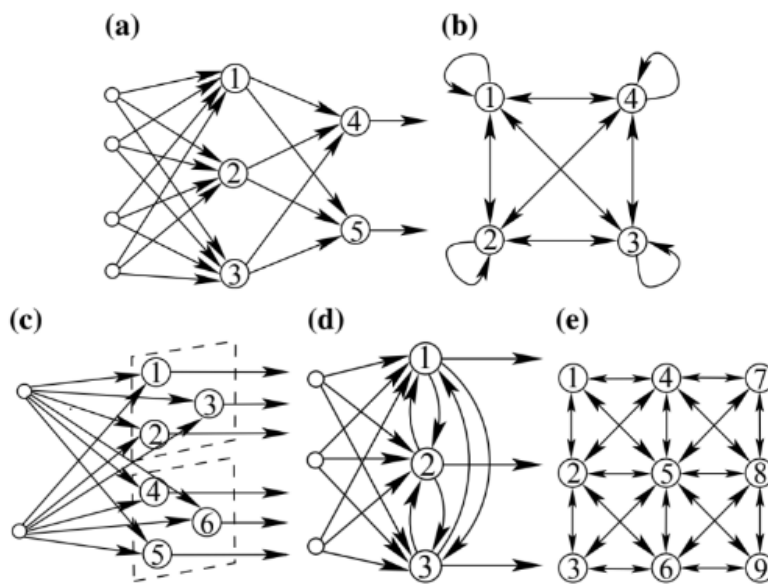
**Figure 3.4:** Decision Tree [24]

This algorithm is known for its' simplicity and it can be also used both for numerical and categorical data. This fact leads to the conclusion that the algorithm can be versatile for many different types of datasets [24] [25].

### Neural Networks and Deep Learning

Neural Networks (NN) or Deep Learning are frameworks within machine learning, inspired by the human brain function and they are basically applied in supervised machine learning. They manage to learn from a set of parameterized differentiable functions with many connected layers so that they can organize past data and generate predictions [26].

NNs consist of interconnected nodes, named neurons which are arranged in layers so node  $i$  is connected to a node  $j$  with connection weights  $w_{ij}$  represented in a weight matrix  $W$ . Some of the most widely used forms of NNs as shown in Figure 3.5., are the fully connected layered feedforward networks (a), recurrent (b), lattice (c), fully connected with lateral connections (d), and cellular (e) as well as convolutional NNs [27].



**Figure 3.5:** Neural Network representations [27]

The general rule says, that the first stage of the process in NNs is learning or in other words optimization or training, and the second stage is generalization or recalling that the model is evaluated is unseen data. During the first stage, it is important to define the hyperparameters of the number of epochs and batches. In more detail, batches are small subsets that the training dataset is divided into in order to facilitate the training process. On the other hand, the epoch's number indicates the number of times that the dataset will be submitted to training [28].

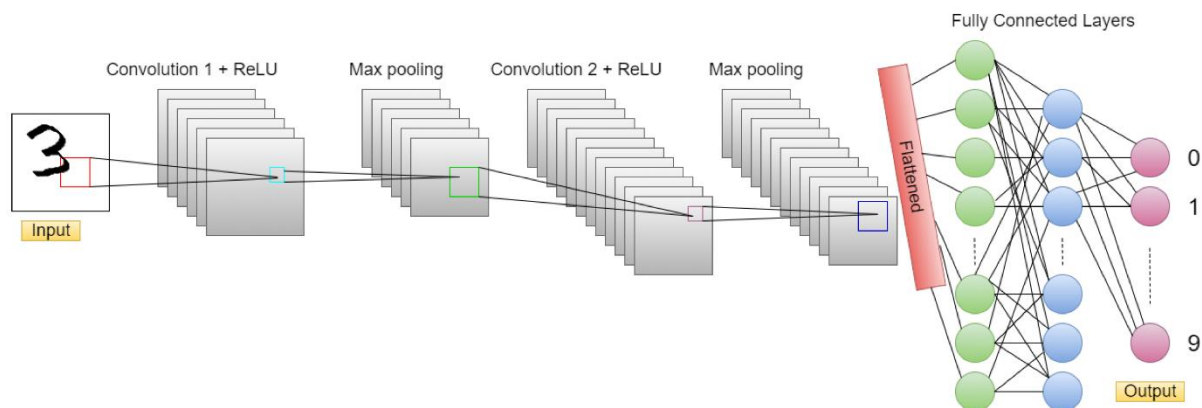
In addition, there are some key parametrization factors during the training that are important to be tuned properly in order to achieve the desired model's performance. One important factor is the activation function, whose role is to control the node's output and train them to capture intricate patterns and this brought innovation in NNs. Currently, one of the most commonly used functions that address successfully to complex patterns, is the rectified linear unit (ReLU) and its variants [29]. Furthermore, another important factor is the optimization algorithm, and the most commonly used in NNs is «adam». Its' role is to enhance the training process by which adjusting the network's weights during training to minimize prediction error [30].

The latest approaches, according to the bibliography regarding voice recognition tasks, either use as an input hand-crafted extracted features to feed the (DNN) or just raw audio signal due to their ability to process sequential and high-dimensional data [14].

- Convolutional Neural Networks (CNNs)

Convolutional Neural Networks or CNN is one of the most famous algorithms related to NNs. It is a structured architecture with multiple layers supported by certain functions to extract features from the input.

Considering the general structure of CNNs, the first layer receives the raw data values and the convolutional layers follow to operate the extraction. After each convolutional layer, there is one activation layer with the function ReLU to support the effective learning process. The following layers are the pooling (max pooling, or average pooling layers) that reduce the dimensions of the feature matrices. Finally, the network has created a set of vectors of the most relevant information, highly discriminative and at the same time with lower dimensionality (Figure 3.6.) [29] [31].



**Figure 3.6:** CNN representation [31]

CNNs outperform compared to other NN approaches due to various advantages. Starting with the fact that the weight-sharing feature reduces the parameters, boosts the generalization, and limits the overfitting. In addition, feature extraction layers and classification layers are trained simultaneously, so they are directly optimized. Finally, CNNs are well-designed to handle high-dimensional data thanks to the pooling layers [32].

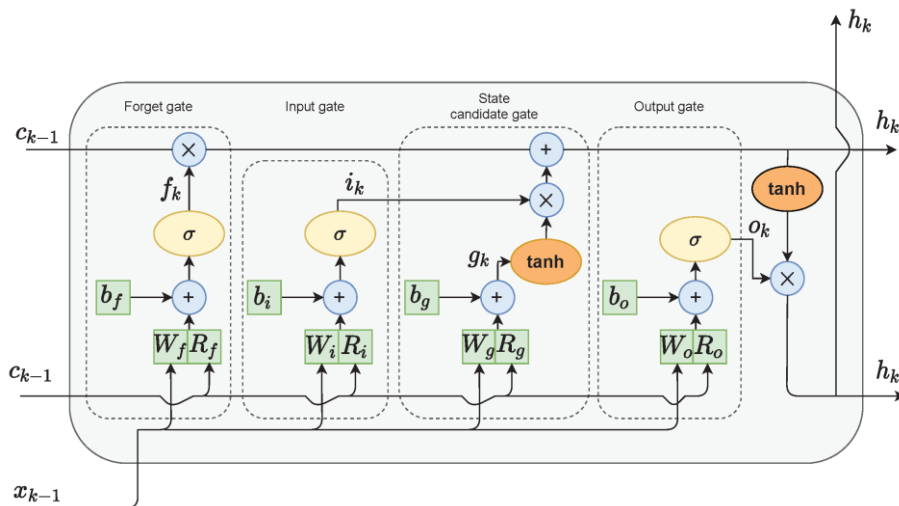
CNNs find a lot of applications in medical imaging (grid-like data). However, they are vulnerable in sequential data due to their inability to measure the dependencies of distant elements as well as their position or order.

- Long Short-Term Memory (LSTM)

Long Short-Term Memory Neural Networks (LSTM) falls into the category of a recurrent network which differentiates with its ability to handle long sequences. This fact is achieved By the use of gates and regulating the information's flow [33].

In more detail, there are three categories of gates, the forget gate, the input gate, and the output gate. The first one is responsible for controlling the discarded information from the

cell state, the second one handles the input that is stored in the cell state, and finally, the exit information from the cell state is decided by the output gate. The cell state distinguishes the important information while in classical recurrent networks, the hidden state is the memory and at the same time the output of the network [34]. Therefore, the LSTM cell (Figure 3.7.) is an extension of a recurrent cell enhanced with robustness and versatility [33].



**Figure 3.7:** LSTM cell structure representation [34]

Additionally, compared to typical recurrent networks, LSTM networks incorporate two activation functions to enhance performance, the tanh function, and the sigmoid activation function additionally which filter unnecessary information [34]. The functions are shown below:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (2)$$

In LSTM networks:

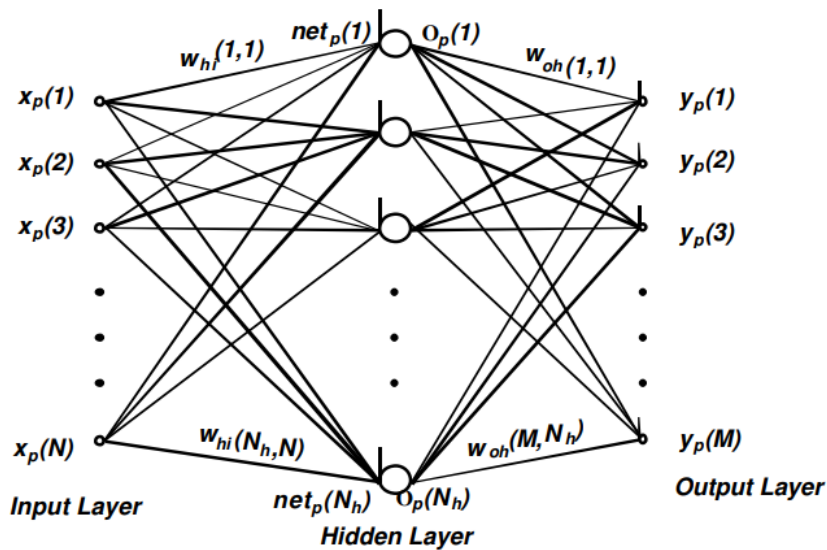
$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (3)$$

These attributes make LSTMs effective tools for long-term dependency tasks in sequential data.

- **Multilayer Perceptron**

Multilayer Perceptron is a type of feedforward NNs, widely used for both classification and regression applications. They are composed of layers, and every layer consists of connected nodes. The input and the output nodes have linear activation functions and the hidden nodes non-linear.

In Figure 3.8. an example of a three-layer perceptron is represented, where all of the inputs are also connected directly to all of the outputs. The hidden unit and output nodes have thresholds associated with them in addition to the weights. So, the signal is fed into a node and in any connected layer and it has the original input multiplied by a weight, adding a threshold, and then passed through an activation function [35].



**Figure 3.8:** Three-layer - Multilayer Perceptron [35]

### 3.2 Evaluation and Interpretability Methods

There are a lot of critical tools that assist the interpretation of models' results by providing insights and explaining the way as well as the reasons models make certain predictions. In this specific research, the following tools were used for evaluation and interpretability:

#### Evaluation Metrics:

- a. Confusion Matrix: A matrix that interprets the model's performance by comparing the actual with the predicted values

In more detail, the general structure of the confusion matrix is shown in Figure 3.9. in which true positive (TP), true negative (TN), false positive (FP) and false negative (FN) cases are represented:

Confusion Matrix		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

➤ TP = Outcome is correctly identified as positive.  
 ➤ TN = Outcome is correctly identified as negative.  
 ➤ FP = Outcome is incorrectly identified as positive.  
 ➤ FN = Outcome is incorrectly identified as negative

**Figure 3.9:** General Structure of Confusion Matrix [19]

- b. Accuracy: A vital metric that measures the rate of correct predictions over the dataset and based on the Confusion Matrix:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

- c. Recall: Counts if the models capture the actual positive cases and based on the Confusion Matrix:

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

- d. Precision: Measures how well the model detects the actual positives and based on the Confusion Matrix [19]:

$$Precision = \frac{TP}{TP+FP} \quad (6)$$

- e. F1-SCORE: The harmonic mean of precision and recall and based on the Confusion Matrix [36]:

$$F1\ Score = \frac{2TP}{FN+FP+2TP} \quad (7)$$

- f. Specificity: Counts the actual negative correctly identified and based on the Confusion Matrix [37]:

$$Specificity = \frac{TN}{TN+FP} \quad (8)$$



- g. Training time: Another important performance indicator that shows the time (in seconds) the model needs to train on a dataset [38].

#### Interpretability Methods:

- a. LIME (Local Interpretable Model-agnostic Explanations) method is a model interpretability tool used in machine learning that shows multiple local estimations based on particular predictions by using the input and applying an additive feature attribution technique. For this current research, binary vectors were exported indicating feature presence or absence [39].
- b. SHAP (SHapley Additive exPlanation) Values is another interpreting method which is based on cooperative game theory. It explains how each feature's contribution affect the output in the end [39]. In more detail, the output of a function  $f$  is explained by SHAP values, which are the sum of the impacts  $\phi_i$  of each feature given to a conditional expectation. Crucially, the introduction order of features affects non-linear functions. SHAP values are the average of all possible orderings [40].
- c. The Partial Dependence Plot (PDP) interprets the small influence that one or two features have on the expected outcome. If there is a complex relationship between a feature and a target, a PDP plot can demonstrate it [41].
- d. Permutation importance is a technique that determines the significance of individual features in machine learning models. As part of the process, each feature is permuted to disrupt the relationship between the feature and the target variable, and metrics as such as accuracy or mean squared error show the change in the model's performance. This approach helps in identifying which features are critical for the experiment leading to a more unbiased estimation of feature importance. However, it can be computationally intensive as it requires multiple evaluations of the model, and in cases of high correlation of features, bias may occur [42].

### 3.3 Bioacoustic and Linguistic Features

It is important to consider during the feature extraction that the segments of the time in different window lengths can also affect the quality of the results. The bioacoustic features are mostly categorized into the following groups, but depending on the study they have been differently categorized [7] [14].

- The source features represent the voice of quality characteristics, like the jitter which is the frequency variation/pitch, and the shimmer, meaning the amplitude variation/loudness. A typical range for adults is 0,5-1% for the former and 0.05 db to 0.22 db for the latter. Glottal Closure Instant (GCI) marks the glottal closure phase and helps determine the previous parameters [7].

- The spectral features refer to articulatory prosodic and phonetic characteristics related to motor control in speech production in frequency and spectral analysis [7] [14].
- The “cepstral” features (the word comes from “Spectrum”), like the Mel Frequency Cepstral Coefficients (MFCCs) are the most common features, describing the variation of low frequencies of the signal. They achieve high performance in the mel-frequency domain to mimic the human auditory system [14].
- The prosodic features describe the variation in pitch, loudness stress, and rhythm related to differences between individuals’ speaking ways. They are long-term features. For example, in women, F0 is usually around 200-220 Hz compared to men’s value which is 100-120 Hz, due to thinner and shorter vocal folds. Sound pressure level (SPL) and zero-crossing rate (ZCR) represent the time-domain prosodic features that indicate the acoustic wavelength and the number of times the signal is zero [7].
- The formant features detect information regarding the coordination of articulators (tongue, lips, lower jaw, and velum). Values F1 and F2 characterize the vowel quality and F3 F4, and F5 the colour of one’s voice [7].
- Deep audio features are considered the features extracted from the raw signal or extracted from it and fed into the Deep Neural Network (DNN) [14].

Many tools support feature extraction nowadays, like the OpenSmile toolkit (<https://www.audeering.com/research/opensmile/>), Python libraries such as Librosa (<https://librosa.org/doc/main/tutorial.html>), and COVAREP toolkit [43].

Apart from the bioacoustic characteristics, studies have also shown remarkable results regarding linguistic features. A well-known tool that measures the frequency of certain word categories is Linguistic Inquiry and Word Count (LIWC) [44]. According to the bibliography, Rude et al 2004, depression has been significantly associated with the following word categories:

- First-person singular pronouns

There are many different reactions and ways of responding when someone faces a difficulty, and one of them is being trapped into him/her self-regulatory cycle. (Pyszczynski & Greenberg, 1987). During this situation, the person strongly focuses on himself/herself. Many studies have shown there is a correlation between the increased usage of words in the first-person singular and this depressed reaction [29] [45] [46]. However, in this field, there is still room for further research.

- Depression and emotional words

Aaron Beck’s theory mentions that depressed people use more negative emotional words than non-depressed people [46] and more researchers have also concluded to significant results that enhance this statement. On the other hand, regarding to the positive emotion words, fewer of them are used by a depressed group compared to a non-depressed. Although both sides include literature that supports the fact that, this area needs more research to clarify [45].

### 3.4 Related Work

After thorough research regarding the different approaches to voice evaluation of depressive presence, absence, severity, or traits, the relevant articles summarize the most widely used techniques below (Table 3.1). The divided input dataset and the performance metrics are also noted for each method.

**TABLE 3.1: SPEECH DATASETS FOR THE STUDY OF DEPRESSION**

Ref.	Input	Model	Performance
1. Detection of Major Depressive Disorder Based on a Combination of Voice Features: An Exploratory Approach [47]	Subjects from five institutions  102 patients (“depressed group (HDRS ≥ 8)”) and 129 non-depressed	Logistic regression with regularization	~90% sensitivity, specificity, and accuracy in the training set ~80% sensitivity, specificity, and accuracy in the test set.
2. Detection of major depressive disorder using vocal acoustic analysis and machine learning—an exploratory study [48]	22 patients with previous diagnosis of major depressive disorder and 11 non-depressed participants	Random forest with 100 trees	accuracy (87.5575% ± 1.9490), kappa index 0.7508 ± 0.0319 and specificity 0.8354 ± 0.0254
3. Estimating Depressive Symptom Class from Voice [49]	Recruited subjects from depressed patients at the National Defense Medical College Hospital in Japan 110 participants	Decision Tree- divided into two groups with distinct symptom profiles	Accuracy 79% sensitivity 83% and specificity 76%
4. An End-to-End model for Detection and Assessment of Depression Levels using Speech [17]	AVEC 2016 DAIC-WOZ Dataset: 146 non-depressed and 43 depressed participants	<ul style="list-style-type: none"> <li>• Spectrogram based CNN</li> <li>• End-to-End CNN</li> </ul>	<ul style="list-style-type: none"> <li>• Overall accuracy 59.2%</li> <li>• Overall accuracy 61.32%</li> </ul>
5. Adieu Features? End-to-End Speech Emotion Recognition Using a Deep Convolutional Recurrent Network [16]	RECOLA database – 45 participants	End-to-End Deep Convolutional Recurrent Network	Arousal prediction: pc = 0.686 Valence Prediction: pc = 0.261
6. Depression Detection in Speech Using Transformer and Parallel Convolutional Neural Networks [50]	DAIC-WOZ Dataset: 189 participants Training & development: Depressed: 30+ 12 Non-depressed: 77+ 23 Test Set: 47 participants MODMA dataset: Depressed: 23 Non-depressed: 29	Transformer and Parallel Convolutional Neural Networks (CNNs)	DAIC-WOZ dataset: TCC-softmax variant F1-score of 93.6%  MODMA dataset: TCC-softmax variant F1-score of 96.7%
7. Depression Speech Recognition With a Three-Dimensional Convolutional Network [12]	DAIC-WOZ English dataset Training Set: 107 participants (30 depressed, 77 non-depressed) Validation Set: 35 participants (12 depressed, 23 non-depressed) Test Set: 47 participants	Three-Dimensional Convolutional filter bank with Highway Networks and Bidirectional GRU with an Attention mechanism (3D-CBHGA).	Accuracy: 83.1% F1 Score: 0.812

8. End-to-End Multimodal Clinical Depression Recognition using Deep Neural Networks: A comparative Analysis [13]	DAICWOZ dataset (80% training, 10% validation, 10% test split)	Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM)	LSTM-based Audio Features: Accuracy = 66.25% CNN-based Audio Features: Accuracy = 65.60%
9. MFCC-based Recurrent Neural Network for Automatic Clinical Depression Recognition and Assessment from Speech [14]	DAIC-WOZ dataset 56 depressed 133 non-depressed participants	MFCC-based Recurrent Neural Network	overall accuracy: 76.27%

## 4 Materials, Methods and Tools

### 4.1 Dataset

For the development of the current thesis, the clinical interview database DAIC-WOZ was used as a source to feed and train the algorithm. This database, as a part of a larger corpus, the Distress Analysis Interview Corpus (DAIC), that supports the diagnosis of depression, anxiety, and post-traumatic disorders, contains audio and video information driven by a virtual interviewer, called Ellie, controlled by a human in another room. All of the participants have signed an agreement of consent that allowed their data to be shared for research purposes. In this dataset, 189 interview sessions are included. For each session, the dataset also includes the already extracted features supported by the Cooperative Voice Analysis Repository, COVAREP toolbox (v. 1.3.2)[43] [51] [52].

### 4.2 COVAREP

More specifically, collaborative free repository, or COVAREP, is widely used for speech processing and provides methods and algorithms. It also supports a platform where researchers can share and exchange knowledge with the common aim of encouraging speech-processing research.

The main methodologies used in COVAREP are the following and are presented in (Figure 4.1). Starting with pitch tracking via the Summation of the Residual Harmonics (SRH), this method supports the estimation of F0 with robust results regarding the additive noise, a major challenge that has been a topic of research for many scientists in order to measure the rate of vocal fold vibrations.

Furthermore, speech polarity detection is another vital method in COVAREP based on the skewness of the LP residual signal, which dramatically impacts the performance of various analysis and synthesis techniques. In addition, the GCI detection algorithm called SEDREAMS (Speech Event Detection using the Residual Excitation and a Mean-based Signal) is a major methodology used in COVAREP analyzing the precise timings of significant excitation in the vocal folds, crucial for many pitch-synchronous analysis procedures.

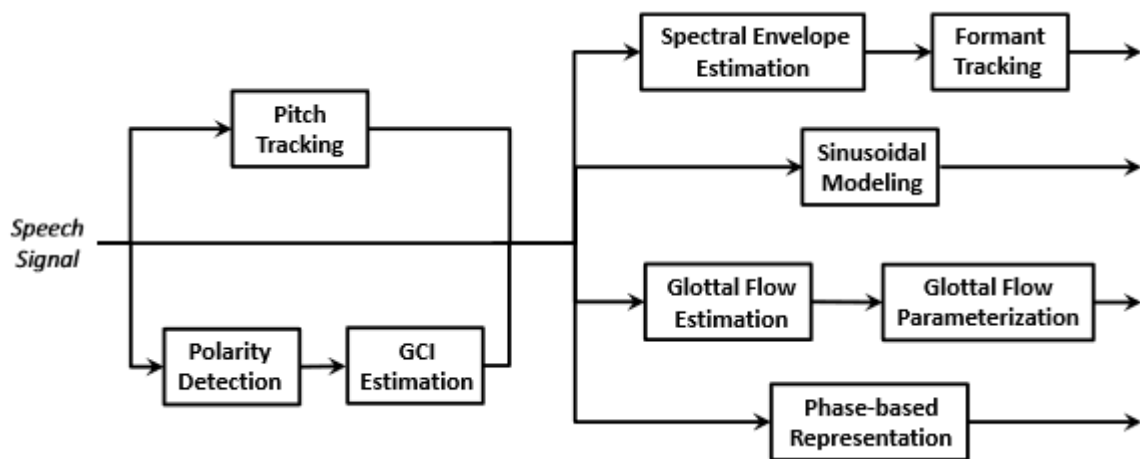
Moreover, the regular vibrations of the vocal cords lead to the harmonic structure of the frequency spectrum of the speech. Using the discrete Fourier transform (DFT), peaks are observed at frequencies that are multiples of the fundamental frequency (F0). These peaks are crucial because they contain the most important spectral information for perceiving voiced speech. COVAREP uses different models to represent it, such as the Sinusoidal Model (SM), Harmonic Model (HM), and Adaptive Harmonic Models (aHM).

Spectral Envelope Estimation is also undertaken in COVAREP, measuring the response of the vocal tract filter, by using a set of methods such as True Envelope (TE), Discrete All-Pole (DAP) Model, Temporally Weighted LP Methods, Advanced Formant Tracker, (SM) and (HM) models.

In addition, Glottal Flow (GF) Estimation is a challenging and complex process of separating the vocal tract and glottal flow components. COVAREP uses Iterative Adaptive Inverse Filtering (IAIF) and Complex Cepstrum-based Decomposition (CCD), though other methodologies need to be compared to lead to robust results. Parameterizing GF is used to support overcoming the changes in phonation types, with parameters such as NAQ (Normalized Amplitude Quotient), QOQ (Quasi-Open Quotient), H1-H2, HRF (Harmonic Richness Factor), PSP (Parabolic Spectral Parameter), MDQ (Maxima Dispersion Quotient) and Peak Slope and Rd Shape Parameter.

Finally, phase processing is another challenge for researchers. Although handling this kind of information is complex, it plays a significant role in accurately modeling speech. In COVAREP, Relative Phase Shift (RPS) and Phase Distortion (PD) are the used features.

In the current dataset, for each different candidate’s interview session, a final matrix of COVAREP features is composed. This matrix, in each column, contains different values representing the relative extracted features regarding pitch, formants, glottal parameters, spectral characteristics, and phase information, as detailed previously. Each row represents a temporal snapshot of the speech signal. Every snapshot results from the method of Consistent interval sampling in which overlapping frames are applied. This is a fundamental procedure that ensures the accurate capture and representation of temporal characteristics of speech. More specifically, the speech signal is divided into fixed-duration frames of 25ms with a slight overlap of 10ms [43]. In this present research, the already extracted COVAREP features were used for further analysis.



**Figure 4.1:** Implemented methods in COVAREP [43]

### 4.3 Methods and Tools

The present analysis was developed using the open-source package Anaconda Distribution, and compiled into Python programming language. For the implementation of the methods described in Chapter 3, several Python packages were used. In Table 4.1. each used library and its role are described:

**TABLE 4.1: PYTHON LIBRARIES**

<b>Libraries:</b>	<b>Role:</b>
Os	Managing the CSV files
Pandas	<ul style="list-style-type: none"> <li>Managing the files &amp; input data</li> <li>Extracting of statistical features of mean, std, min, and max using the function <code>extract_features()</code></li> </ul>
Numpy	Handling infinite data, feature & permutation importance
Matplotlib	Confusion matrices, feature & permutation importance plots, SHAP, LIME, and PDP visualizations
Shap	SHAP plot
lime.lime_tabular	LIME plot
Seaborn	Confusion matrices
Time	Measuring training time
imblearn	SMOTE balancing
sklearn	Handling NaN values, Standardization, Data splitting, Machine learning models
*tensorflow.keras.models	NN models

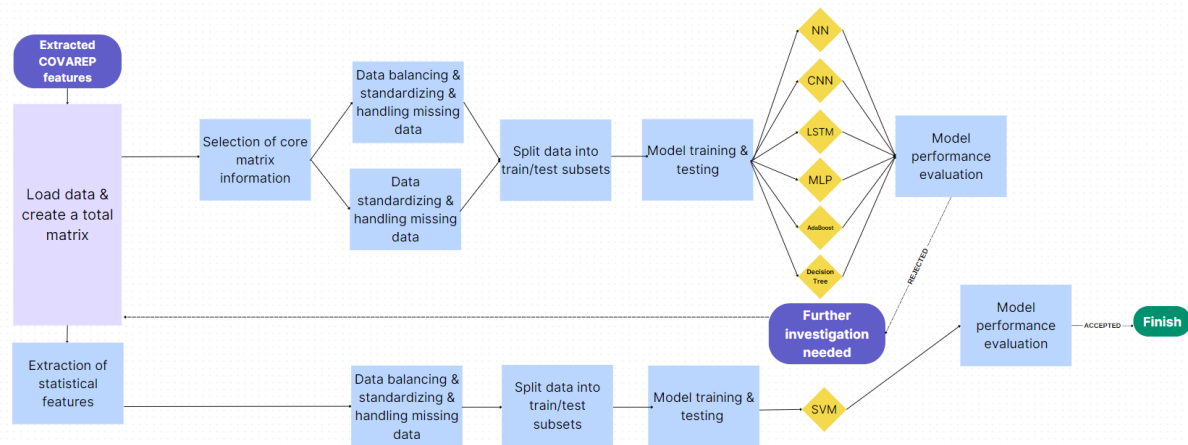
Based on the previous table, the sklearn library, also called «scikit-learn», is an important library used in several tasks. More specifically, the function «SimpleImputer()» assisted the handling of NaN values, the «StandardScaler()» function also assisted the standardization of data values, the «train\_test\_split()» function the data splitting in 80 train/20 test division, as well as the following machine learning models:

- Decision Tree model: using «DecisionTreeClassifier()» function
- MLP model: using «MLPClassifier()» function
- AdaBoost model: using «AdaBoostClassifier()» function
- SVM model: using «SVC()» function

In addition, tensorflow.keras.models library supported the implementation of deep learning models of Simple NN, CNN, and LSTM.

## 5 Development

To support the goal of this thesis and compare different machine learning models in terms of their classification accuracy in separating depressed and non-depressed people in the already extracted voice features taken from the DAIC-WOZ dataset, all the methods and tools mentioned in Chapter 4 were used. In more detail, Figure 5.1. shows a schematic representation of the methodology pipeline.



**Figure 5.1:** Methodology pipeline [Original image]

More specifically, the research pipeline, according to the previous figure (fig. 5.1.), is described below:

**Data loading:** Gathering all the extracted COVAREP features of each separate session and creating a large matrix format. This matrix was difficult to process due to its large size which caused computational restrictions.

### Methodology 1:

To overcome the computational obstacle, a method of core information selection was first attempted. This means, that for each COVAREP file setting the matrix, the first 2000 rows were skipped, the next 7000 rows were only selected, and the rest were discarded. So, the amount of data for each class is:

Number of rows in the class of non-depressed: 931000

Number of rows in the class of depressed: 392000

**Data Preprocessing:** Handling missing/infinite data and standardization of data values

**Data Splitting:** Splitting the total data into 80% train and 20% test subset:

Training set: 1058400 rows

Test set: 264600 rows

**Model Performing:** Six different models were performed separately; these were Simple NN, CNN, LSTM, Decision Tree, MLP, and AdaBoost with the following parameterization:



➤ Simple NN:

First dense layer:

Number of units: 64 units/neurons: support capturing complex patterns

Activation Function: «relu»: support capturing complex patterns

Input Shape: 1 indicates the dimensionality of the input data

Output layer:

Number of units: 1 for binary classification

Activation function: «sigmoid» for binary classification

Optimizer: «Adam», with a learning rate: 0.001

Number of epochs: 10: pass the training dataset 10 times

Batch Size: 64: process 64 samples at a time before updating the weights

➤ CNN:

Conv1D Layer:

Filters: 64 filters used : the layer will learn 64 different features

Kernel: 3 specifies the width of the convolutional filter

Activation Function: «relu»: support capturing complex patterns

Input Shape: 1 indicates the dimensionality of the input data

MaxPooling1D Layer:

Pool Size: 2 reduces computation and control overfitting

Flatten Layer:

This layer reshapes the 2 dimension data from the convolutional and pooling layers into a 1dimension/vector

Dense Layers:

1<sup>st</sup> dense layer: 64 units with «relu» activation: learn patterns and features from the flattened data

Output Dense layer: 1 unit with a «sigmoid» activation function: binary classification

➤ LSTM:

LSTM layer:

Number of units: 64 units/neurons: learning temporal dependencies in sequential data

Input Shape: 1 indicates the dimensionality of the input data

Activation functions: its default activation functions internally: tanh for the cell state and sigmoid for the gating mechanisms (input, output, and forget gates): Activations help in retaining or discarding information through time steps

### Output Dense Layer:

Number of units: 1 unit for binary classification.

Activation Function: «sigmoid» for binary classification

Optimizer: «Adam», with a learning rate: 0.001

Number of epochs: 3: pass the training dataset 3 times

Batch Size: 64: process 64 samples at a time before updating the weights

- Decision Tree: all the default parameters
- MLP:

Hidden Layer Sizes: 64: specifies a single hidden layer with 64 neurons (model learns intermediate features that help map the input to the output)

Activation Function: «relu» used for the neurons in the hidden layer

Solver: «adam» model optimization

Max Iterations: maximum number of epochs or iterations the algorithm can perform

Random state: 42: fixing the randomness to a particular seed value

- AdaBoost:

Number of estimators/weak learners: 100 trained to correct the errors of the previous learners

Algorithm: «SAMME»: specifies the boosting algorithm used

Random state: 42: fixing the randomness to a particular seed value

**Model Performance Evaluation:** In this phase, confusion matrices and accuracy were visualized to estimate the models' performance. In addition, the class distribution was plotted and showed that the imbalance between the classes of non-depressed and depressed is remarkable, so the model was found biased. Further investigation was necessary.

### Methodology 2:

In this attempt, in the **data preprocessing** step, handling missing/infinite data and standardization were conducted as methodology 1. In addition, data balancing was applied by deleting rows from the outperformed class to make both classes equal, so the number for each class was:

Number of rows in the class of non-depressed before balancing: 931000

Number of rows in the class of depressed before balancing: 392000

Number of rows in the class of non-depressed after balancing: 392000

Number of rows in the class of depressed after balancing: 392000

All the following steps of **data splitting** (80% training set - 20% testing set), **model performing and parameterization** (NN, CNN, LSTM, AdaBoost, MLP, and Decision Tree) and **model performance evaluation** were the same as the methodology 1. Thus:

Training set: 627200 rows

Test set: 156800 rows

Consequently, adjusting the data balancing reduced the models' accuracy results, so further research was needed.

### Methodology 3:

In this methodology, another approach was tested, compared to methodologies 1 and 2, in order to handle the computational restrictions and improve the model's performance. The new approach was the extraction of statistical features of mean, std, min, and max that aggregated the time series data to one row per participant. Then, instead of row deletion conducted in methodology 2, data balancing was achieved with the SMOTE technique, Thus:

Number of rows before balancing: 189

Number of rows after balancing: 266

The rest of the **data preprocessing** (handling missing/infinite data and standardization) and **data splitting** (80% training set - 20% testing set) phases that followed were the same as methodologies 1&2. Thus:

Training set: 212 rows

Test set: 54 rows

**Model Performing:** In this methodology, the SVM model was performed with the following key parameters:

- Kernel: linear: separation in the original feature space
- Probability: true: calibration to the decision function to produce probabilities output
- Verbose: true: print out detailed messages about its training process
- Random state: 42: fixing the randomness to a particular seed value

### **Model Performance Evaluation and Interpretability Metrics:**

The confusion matrix and accuracy were visualized to check the model's performance which proved satisfying results compared to the previous methodologies and related work. More metrics regarding the model's performance were then calculated. These were precision, recall specificity, F1-score, and training time. Finally, the tools of SHAP, LIME, PDP, feature, and permutation importance were used to extract useful information regarding the models' predictions.

The relevant scripts of SVM (methodology 3) and NN (methodology 2) are provided in the appendix section.

Furthermore, giving some more technical details regarding the data development, the data transformations of methodology 3 are schematically represented in Figure 5.2. More specifically:

- The script reads the COVAREP data from 189 total sessions of interviews. The given DAIC-WOZ dataset had already split the labels of the sessions for training, testing, and validation but it was decided to be used in a different division. So, the script reads all the COVAREP CSV files from a specific directory by reading each participant's number from three different CSV files and creates a total DataFrame of features and a separate Panda Series for the labels. The total DataFrame has a very large number of rows ( $n \times 10^x$ ) and 74 columns while the labels is a Panda Series of 189 numbers.
- Extracts four statistical features for each participant, and aggregates the time series data to one row per participant, creating a new smaller DataFrame. The new DataFrame has 189 rows and 296 columns.
- Adds the labels and creates a total DataFrame The label for each participant being depressed is «1» and for non-depressed participants is «0». The result is that the DataFrame is consistef of 189 rows and 297 columns.
- After the SMOTE balancing, the data has been converted to a numpy array with final dimensions of 266 rows and 297 columns.

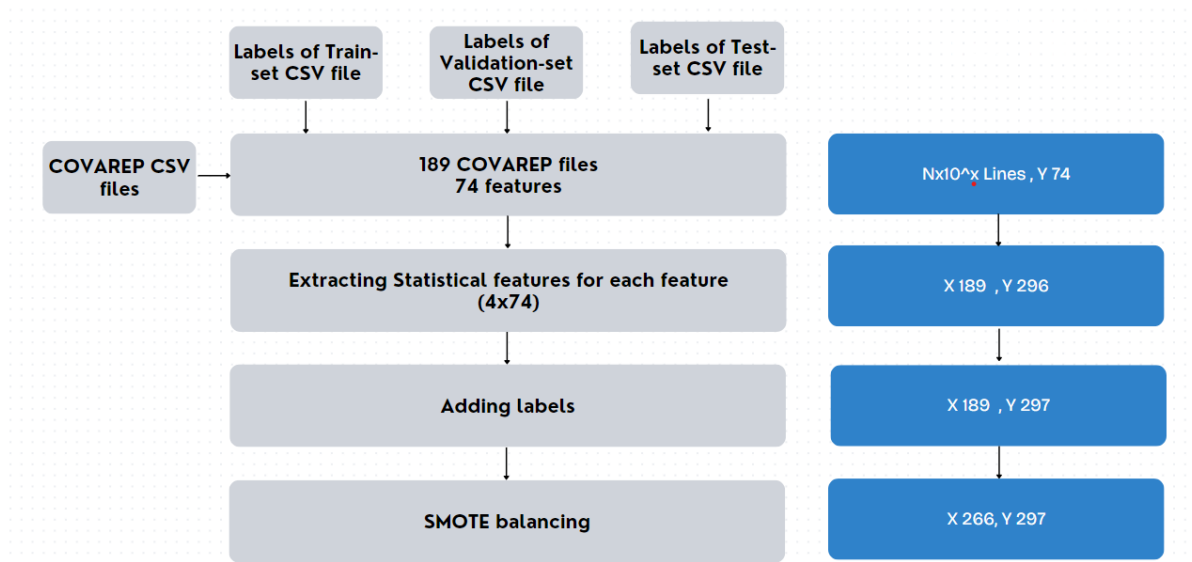


Figure 5.2: Data transformation [Original image]

## 6 Results

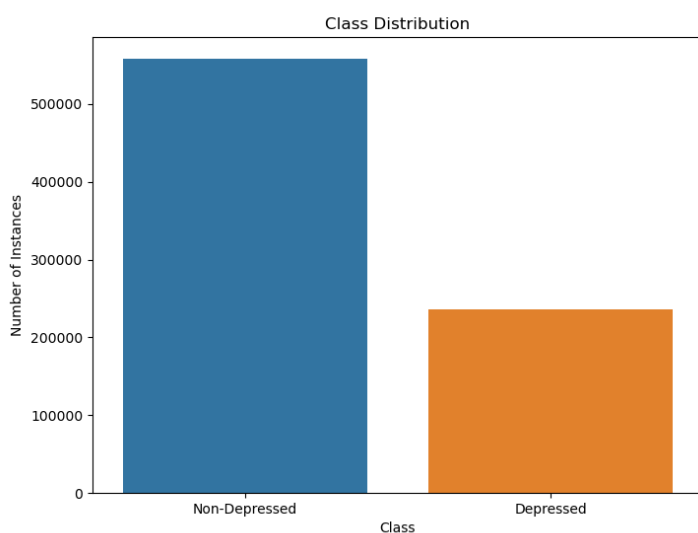
### 6.1 Performance Evaluation

This chapter details the results of the metrics that supported the performance evaluation of the algorithms developed in the three methodologies mentioned in Chapter 5. Algorithms' results of methodologies 1&2, of both balanced and unbalanced conditions, are represented in the following table (Table 6.1).

**TABLE 6.1:** MACHINE LEARNING ALGORITHMS RESULTS FOR BOTH BALANCED AND UNBALANCED DATASET

Algorithm	Accuracy for non-balanced dataset	Accuracy for balanced dataset
Simple NN (10 epochs)	71.82%	63.25%
CNN	72.34%	51.00%
LSTM	71.00%	58.96%
Multilayer Perceptron	72.63%	64.05%
AdaBoost	70.44%	58.96%
Decision Tree	73.51%	69.27%

The class distribution chart in Figure 6.1. shows a remarkable unbalance between classes of non-depressed (0) and depressed (1) before data balancing. In addition, the results of the algorithms show that the unbalanced condition of the data strongly affected the models' performance.

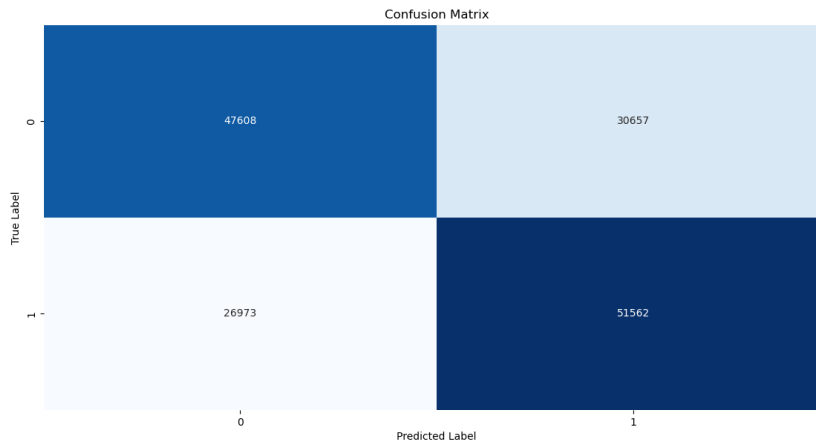


**Figure 6.1:** Class distribution before data balancing

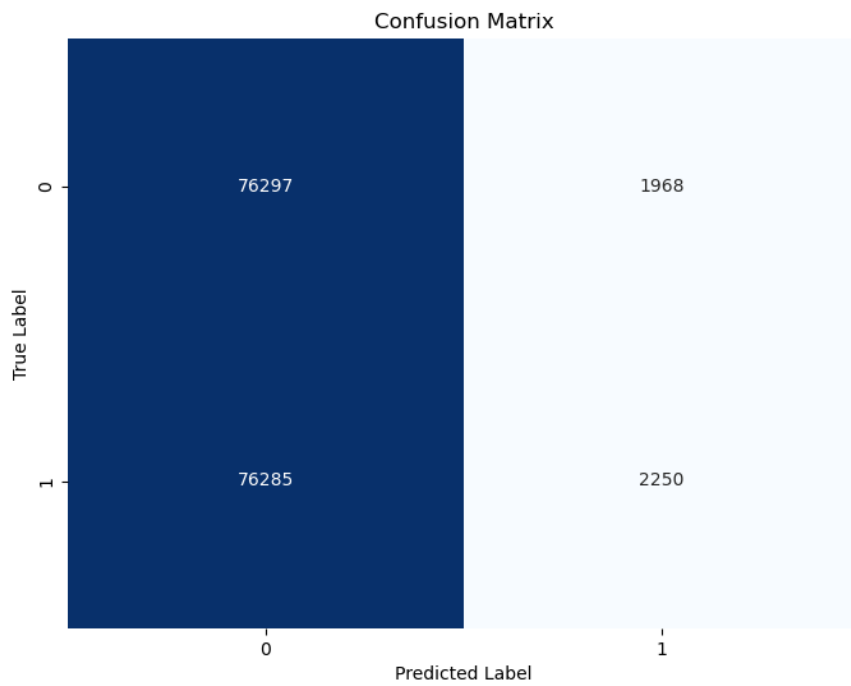
Furthermore, confusion matrices (as shown in figures 6.2, 6.3, 6.4., 6.5, 6.6., 6.7.) were displayed to examine the algorithms' performance in methodology 2 (the confusion matrices of methodology 1 are not included, as the model was biased and therefore their information

is not reliable). In the following confusion matrices, it is important to note that the large values are due to the time-series data.

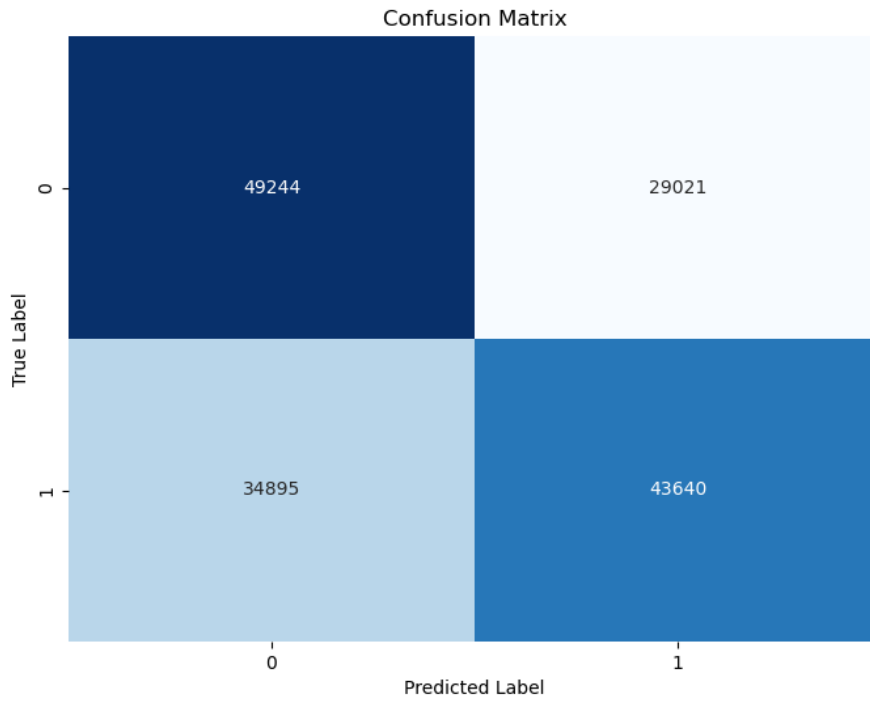
Confusion matrices:



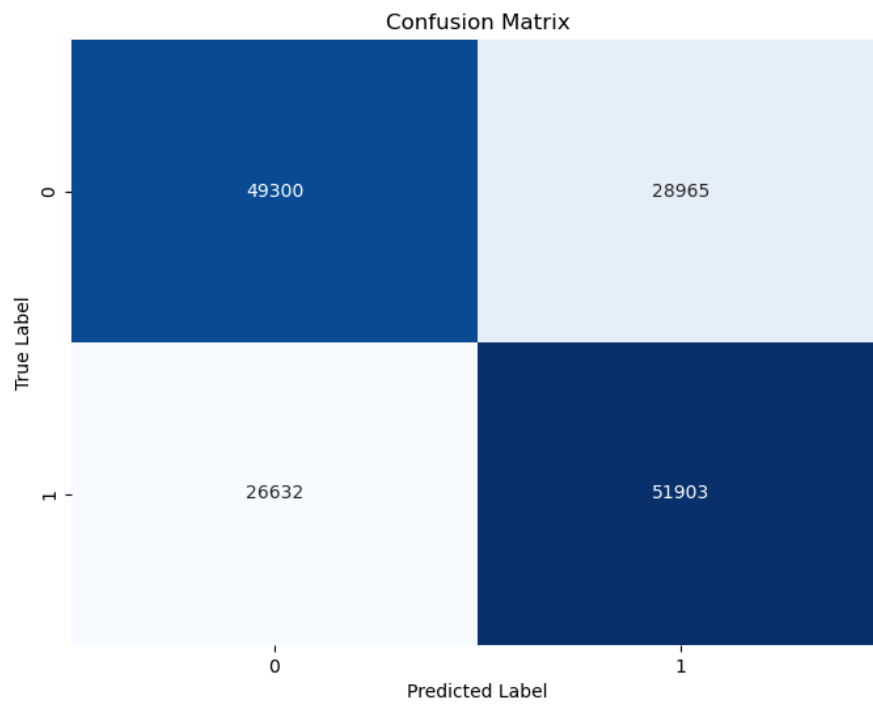
**Figure 6.2:** Confusion matrix of Simple NN algorithm



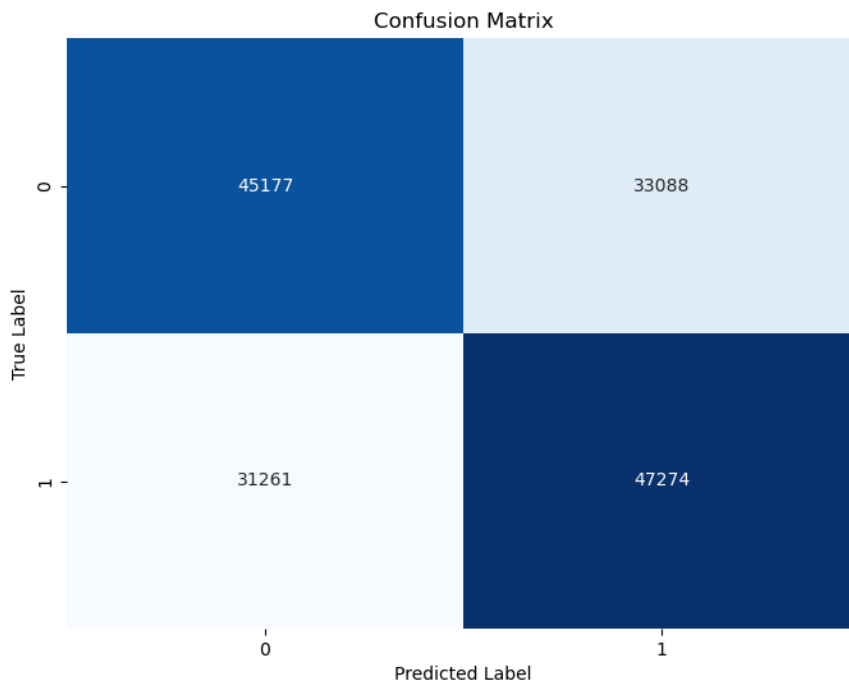
**Figure 6.3:** Confusion matrix of CNN algorithm



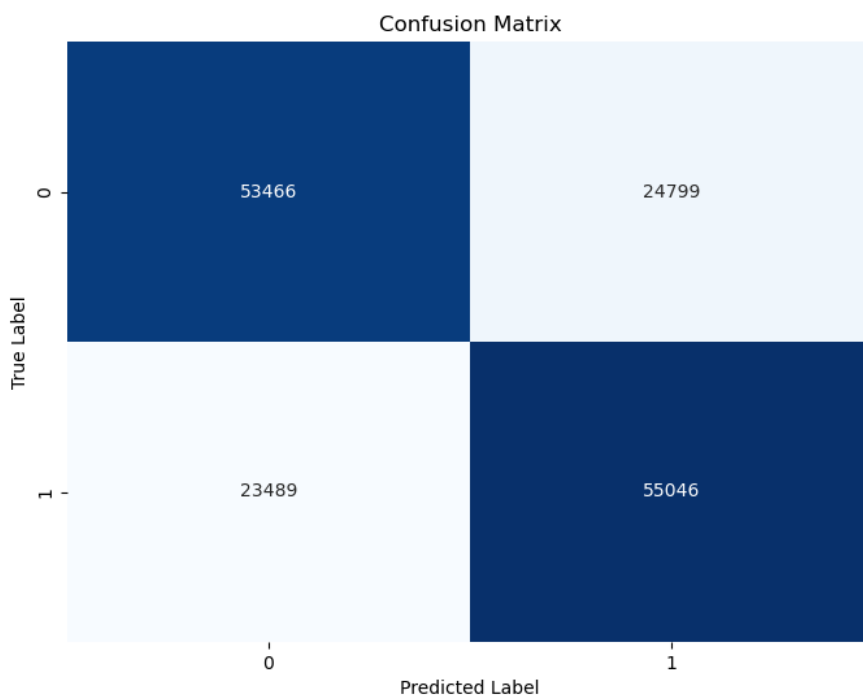
**Figure 6.4:** Confusion matrix of LSTM algorithm



**Figure 6.5:** Confusion matrix of MLP algorithm



**Figure 6.6:** Confusion matrix of AdaBoost algorithm



**Figure 6.7:** Confusion matrix of Decision Tree algorithm

Finally, the lower accuracy results combined with the unsatisfactory results of the confusion matrices consider the models' performance poor in methodology 2 and lead to further investigation. Methodology 3 proved to be the most robust scenario compared to the previous methods in terms of accuracy, confusion matrices, and balanced dataset. The



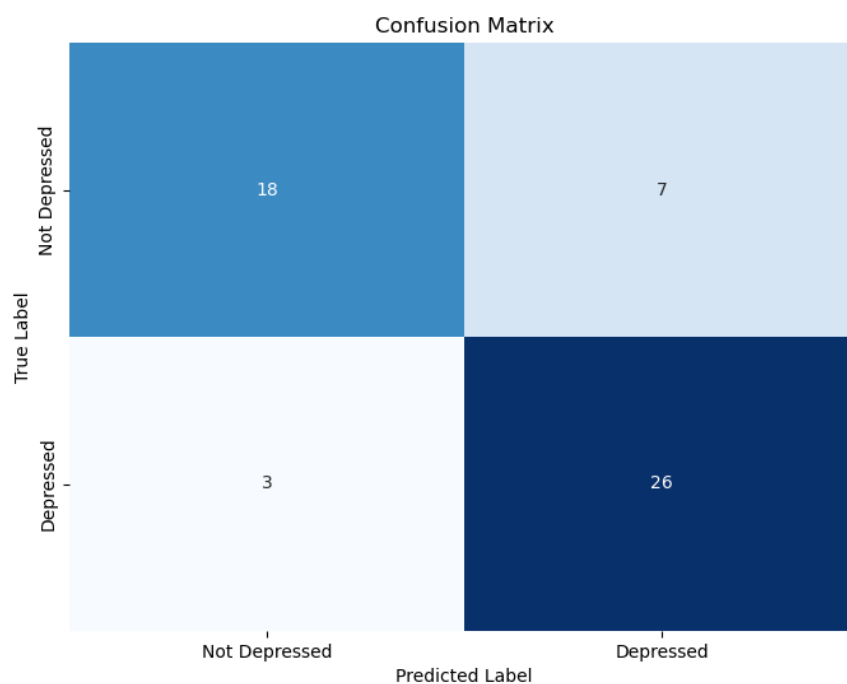
methodology's 3 metrics of performance evaluation are shown in Table 6.2. and Figure 6.8.. As methodology 3 was considered superior to the previous ones, more metrics were calculated in order to evaluate and compare it with the related work in literature. These were precision, recall, f1-score, and specificity which are represented in Table 6.2..

**TABLE 6.2: MODEL PERFORMANCE**

<b>Metrics</b>	<b>Values</b>
Accuracy	0.81
Precision	0.79
Recall	0.90
F1-Score	0.84
Specificity	0.72

According to the previous table, the score of 0.81 accuracy represents a solid performance highlighting an effective model. Moreover, the precision metric checks if the model considers many false positive instances as "depressed". A score of 0.79 is a satisfying result avoiding many false predictions. In addition, the recall metric counts if the model captures the actual depressed cases, and a score of 0.9 means a very good identifier. Regarding F1-Score which balances precision and recall, the model achieved 0.84 which is a strong result that identifies positive cases and avoids false positive ones. Finally, specificity is another important metric that captures the actual negatively depressed cases. The score of 72% is lower compared to recall, so overall it is easily concluded that the model is good in identifying depressed cases and less effective in identifying true negatives, compared also to related work in the literature.

Furthermore, the Confusion Matrix Interpretation summarises the table's results in a four-box figure (Fig. 6.8.) showing the results calculated from the test set (54 cases in total). From the total cases, 26 were predicted depressed correctly (TP), 7 were predicted depressed incorrectly (FP), 18 correctly predicted negative (TN) and 3 false non-depressed (FN). In the present confusion matrix, lower values than the previous methodologies' confusion matrices are noticed due to the aggregation of the time series data to one row per participant.



**Figure 6.8:** Confusion Matrix

Finally, another metric that is important to take into account when choosing a model for real-world data, is computational efficiency. For this aim, the training time was measured at 0.0899 seconds, an excellent score that allows real-world applications.

## 6.2 Evaluation Scenarios and Interpretability Metrics

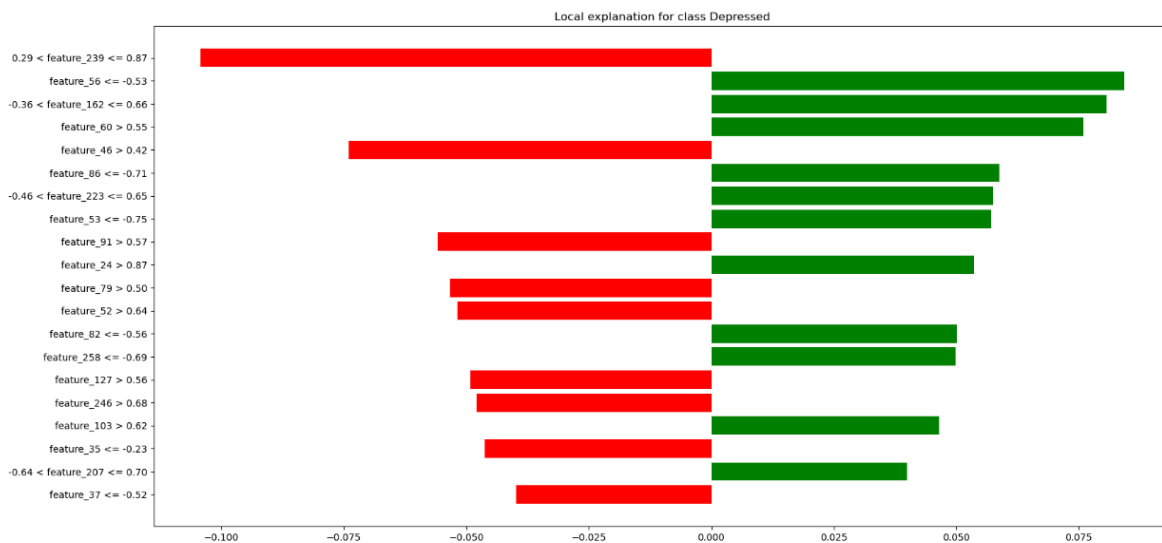
According to the metrics discussed in Chapter 6.1., methodology 3 proved to be the most robust alternative of all the examined scenarios. To evaluate the models' predictions of methodology 3, the following interpretability tools were used:

### LIME Plot

The LIME interpretability method provides a detailed explanation of why the model classifies a specific instance as depressed or non-depressed, giving a local approximation. In more detail, it shows which features contributed the most to the final decision as well as their specific value ranges [39].

Based on Fig. 6.9. the top 20 are represented and ranked in depending on the size of the bar, showing the larger bar the stronger the contribution. In addition, the green colour represents the positive contribution to the «depressed» result while the red colour the negative contribution to the «depressed». On the left side, the feature label is represented as well as the interval in which the feature value lies relative to the decision boundary within the model. The rest 276 features were chosen not to be presented on the graph as they offer less contribution to the final result and they would make the picture more illegible.

According to the LIME graph, in the present model, feature 239 with the interval  $0.29 < \text{feature\_239} \leq 0.87$  has a large negative contribution (red bar), suggesting that within this interval, it strongly influences the model towards predicting that the individual is not depressed. Less influence but still toward predicting an individual is not depressed, feature 46 has when its value is more than 0.42, feature 91 when it is higher than 0.57, feature 79 when its value is higher than 0.5, feature 52 when it is higher than 0.64, feature 127 when it is higher than 0.56, feature 246 when it is higher than 0.68, feature 35 when it is less or equal than -0.23 and feature 37 when it is less or equal than -0.52. Conversely, feature\_162 within the interval  $-0.36 < \text{feature\_162} \leq 0.66$  has a strong positive contribution (green bar), indicating that this feature is pushing the model towards classifying the individual as depressed as well as feature 56 when it is less or equal to -0.53, feature 60 when it is higher than 0.55, feature 86 when its value is less or equal than -0.71, feature 223 within the interval  $-0.46 < \text{feature\_223} \leq 0.65$ , feature 53 when it is less or equal to -0.75, feature 24 when it is higher than 0.87, features 82 and 258 when they are less or equal than -0.56 and -0.69 accordingly, feature 103 when it is higher than 0.62 and finally feature 37 when its value is less or equal to -0.52.



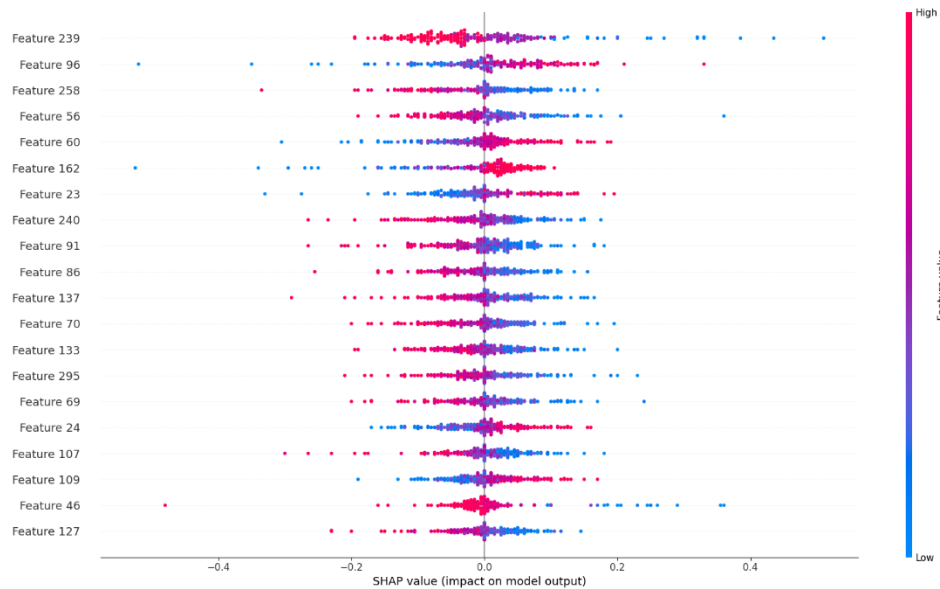
**Figure 6.9:** LIME plot

### SHAP Plot

The following graph that provides valuable information and supports the detection of the machine learning model's results is the SHAP plot as shown in fig. 6.10., which shows the contribution of each feature to the model's predictions both locally and globally [39]. Features are sorted on the y-axis based on their overall significance, which is determined by the mean absolute SHAP value, so the top 20 features are presented. In more detail, positive SHAP values indicate that the feature increases the model's predicted output, so making the model more likely to predict "Depressed" while negative SHAP values do the opposite, indicating that the feature decreases the model's predicted output and making the model less likely to predict "Depressed". In addition, the dots' colour indicates if the actual feature number is high or low with red and blue colours accordingly.

According to the present SHAP graph, it is noticed that feature 239 has a strong impact on the model's prediction as its high values correspond to negative SHAP values which means that the high values are pushing the model towards reducing the likelihood of the "depressed" classification. On the contrary, feature 96 has more blue dots on the left side which means that lower feature values push slightly the model towards non-depressed but the impact is small. Regarding features 258 and 56, the concentration of red dots on the left and blue dots on the right suggests that lower values may push the model towards predicting "depressed," while higher values may increase the likelihood of predicting "no depressed." In feature 60, red points are predominantly on the right part of the graph, indicating that higher values increase the likelihood of a "Depressed" prediction as well as in feature 162 in which the pattern is very clear. In feature 23, it is noticed that lower values push the model to reduce the likelihood of depression while higher ones have the opposite impact as the same happens with features 24 and 109 in slightly less impact. The opposite happens in features 240, 91, 86,137,70,133,295,69, 107, and 127 that have almost the same pattern with higher values reducing the possibility of depression and lower values increasing it. Finally, in feature 46

higher values have less impact as according to the graph, they approach zero SHAP values while lower values push the model towards "depressed" prediction. The last features of SHAP plot are less significant than features higher up on the list which doesn't mean they are unimportant, but they have a lighter influence compared to the top-ranked ones.



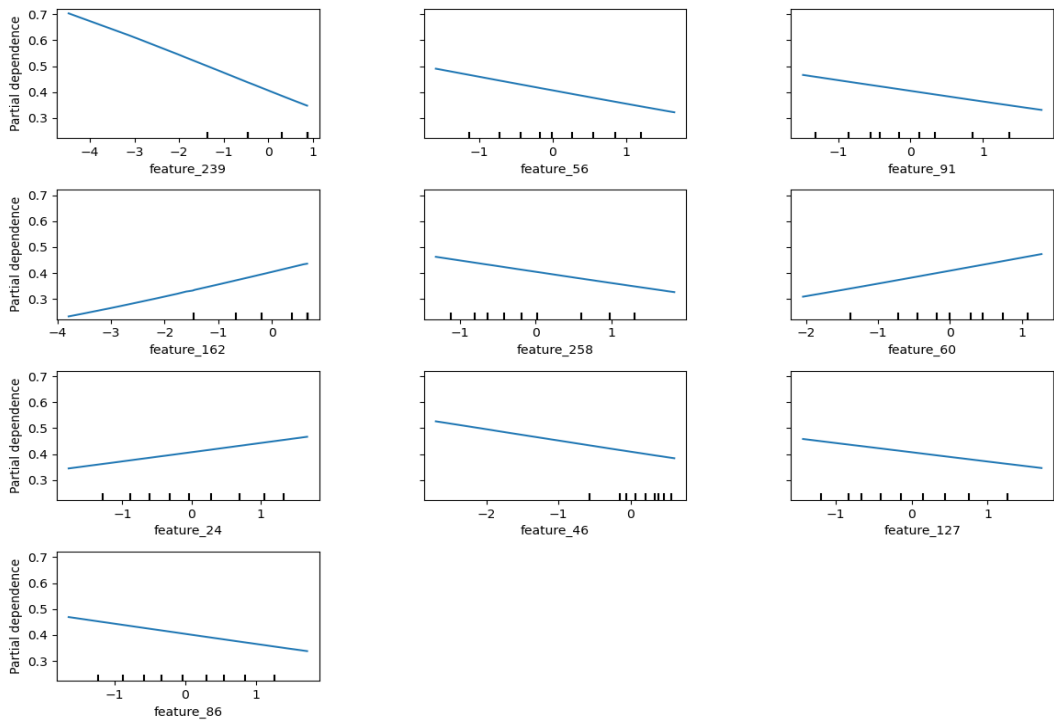
**Figure 6.10:** SHAP plot

### PDP Plot

Each subplot of PDP plots represents a different feature and its impact on the prediction probability, revealing features' trends and how feature values push the model towards or away from predicting the positive class (depressed) [41]. Figure 6.11. illustrates 10 manually selected features that are present in both the LIME and SHAP plots.

In more detail, according to the following plots, features 239, 56, 91, 258, 46, 127, and 86 show downward trends, indicating that as these feature values increase, the likelihood of predicting the "depressed" class decreases, suggesting a negative relationship between the features and the likelihood of predicting "depressed." On the other hand, features 162, 60, and 24 have upward trends meaning that higher values of these features increase the likelihood of predicting the "depressed" class.

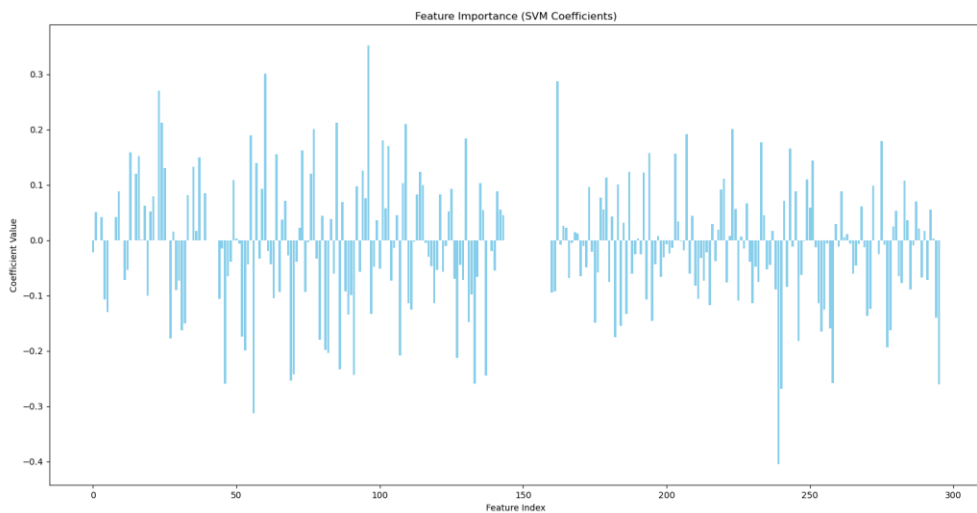
In addition, differences in the gradient reveal useful information too. A steep gradient, whether upward or downward, indicates that small changes in the feature values lead to significant changes in the predicted outcome.



**Figure 6.11:** PDP plots

### Feature and Permutation Importance

Figure 6.12. aims to represent the importance of the features, estimated during the training process. Based on this graph, the features with positive coefficients estimated that they contribute positively or negatively to predicting the "depressed" class are shown in Table 6.3..

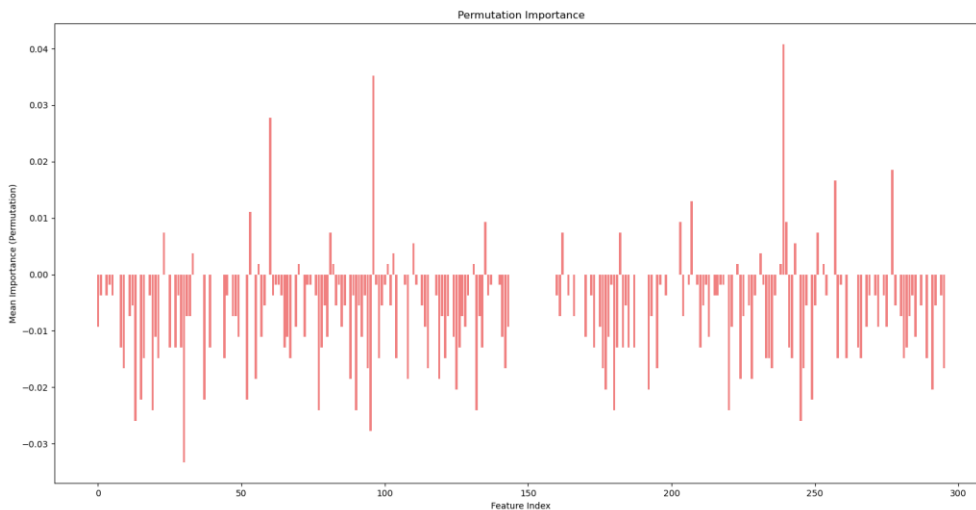


**Figure 6.12:** Feature importance graph

**TABLE 6.3: FEATURES IMPORTANCE**

Features with positive feature importance:
1 3 8 9 13 15 16 17 18 20 21 23 24 25 28 33 35 36 37 39 49 50 55 57 59 60 64 66 67 72 73 76 77 80 83 85 87 92 94 95 96 99 101 102 103 106 108 109 113 114 115 121 124 125 130 135 136 141 142 143 162 164 165 168 169 173 177 178 179 181 183 185 187 190 192 194 197 203 204 207 209 216 218 219 220 222 223 224 226 228 233 234 237 241 243 245 249 250 251 259 261 262 263 268 272 275 279 280 283 284 287 288 290 292 293
Features with negative feature importance:
0 4 5 11 12 19 27 29 30 31 32 44 45 46 47 48 51 52 53 54 56 58 61 62 63 65 68 69 70 71 74 75 78 79 81 82 84 86 88 89 90 91 93 97 98 100 104 105 107 110 111 112 116 117 118 119 120 122 123 126 127 128 129 131 132 133 134 137 138 139 140 160 161 163 166 167 170 171 172 174 175 176 180 182 184 186 188 189 191 193 195 196 198 199 200 201 202 205 206 208 210 211 212 213 214 215 217 221 225 227 229 230 231 232 235 236 238 239 240 242 244 246 247 248 252 253 254 255 256 257 258 260 264 265 266 267 269 270 271 273 274 276 277 278 281 282 285 286 289 291 294 295
Positive permutation importance features
23 33 53 56 60 70 81 82 96 101 103 110 131 135 162 182 203 207 223 231 238 239 240 243 251 253 257 277

In addition, figure 6.13. follows to detect which of the features were actually important and effective after the training process. Table 6.3. also represents the features with positive mean importance.



**Figure 6.13:** Permutation importance graph

Combined Results

Feature 239 has the steepest gradient of a downward trend, indicating that as its value increases, the predicted probability of a "depressed" class decreases according to the PDP plot. In addition, the LIME plot shows the larger red bar from all the features within the range

" $0.29 < \text{feature\_239} \leq 0.87$ ." This indicates that, for this particular instance, feature 239 significantly decreases the likelihood of the "depressed" prediction, pushing it towards "non depressed." According to the SHAP plot, feature 239 is the most significant of all the features. The concentration of red dots on the left side of the plot suggests that when the feature value is high, it significantly contributes to decreasing the likelihood of predicting the "depressed" class. Finally, the feature and permutation importance metrics show that feature 239 has been estimated to have a negative coefficient on its importance regarding the prediction of the "depressed" class as well as it shows a positive coefficient in the permutation importance suggesting that this feature is truly effective in practice.

Regarding feature 56, globally as shown in the PDP, higher values tend to reduce the likelihood of a "depressed" prediction. However, according to the LIME plot, for a specific instance, the feature can significantly increase the likelihood of predicting "depressed" when it is within the range of " $\text{feature\_56} \leq -0.53$ ". The SHAP plot confirms that the impact of this feature varies across the dataset, with both high and low values influencing the model's predictions by decreasing and increasing accordingly the probability of the "depressed" class. Finally, the feature and permutation importance metrics indicate that this feature has been estimated to have a negative coefficient on its importance regarding the prediction of the "depressed" class and indeed it shows a positive coefficient in the permutation importance suggesting that this feature is effective in practise.

In addition, feature 91 has a consistent negative impact on the "depressed" prediction, both globally (as seen in the PDP and SHAP plots) and locally (as seen in the LIME plot). Its trend in the PDP is downward, in the LIME plot the bar is red, and the distribution of SHAP values, indicated by red dots on the left side, shows that it pushes the prediction towards "non depressed", characterizing it as a protective feature, reducing the likelihood of the model predicting the "depressed" class for higher values.

Feature 162 has consistently a positive impact on the "depressed" class, both globally and locally. The PDP plot shows an upward trend, the LIME plot a large green bar and the SHAP plot indicates that when values are high it significantly contributes to increasing the likelihood of predicting "depressed" based on the concentration of red dots on the right side of the plot. This feature can be characterized as a predictor for a "depressed" class when the values are high.

Feature 258 has a negative impact on the "depressed" prediction, as shown in the PDP plot. However, the LIME plot shows that in a specific instance, lower values than  $-0.69$  significantly increase the likelihood of predicting "depressed", according to the green bar. The SHAP plot confirms this dual influence, showing that this feature can both decrease and increase the probability of a "depressed" class prediction, depending on its value, and for lower values it is more likely to push the model towards predicting "depressed" while for higher values to "non depressed."

Regarding feature 60, it is noticed to have a positive impact on the "depressed" prediction based on the PDP plot which represents an upward trend. The LIME plot confirms that showing a green bar for higher values than  $0.55$  the likelihood of predicting "depressed" significantly increases. The SHAP plot supports this statement too, showing that higher



feature values tend to increase the SHAP values so that the possibility of "depressed" prediction. The feature and permutation importance metrics indicate that it has been estimated to have a positive coefficient on its importance regarding the prediction of the "depressed" prediction and indeed it shows a positive coefficient in the permutation importance. This concludes with the fact that feature 60 is actually effective.

Feature 46 has a downward trend according to the PDP plot, a red bar according to the LIME plot for values higher than 0.42, and finally, the SHAP plot agrees with the statement that higher values tend to decrease the likelihood of predicting the "depressed" class.

The same pattern with the previous feature has feature 127, showing a downward trend according to the PDP plot, and a red bar according to the LIME plot for values higher than 0.56. The SHAP plot shows red dots on the left side proving that for higher values there is less possibility of a "depressed" prediction while for lower values the prediction is pushed towards "depressed". This feature is the last one on the SHAP plot which means that it is the least significant compared to the other mentioned features.

Feature 86 is downward according to the PDP plot. For values less than -0.71, according to the LIME plot, a green bar indicates an increasing tend of the likelihood of a "depressed" prediction. Finally, the SHAP plot confirms by showing blue dots on the right part and red dots on the left part, the increased possibility of the "depressed" predicted statement in lower values and "non depressed" in higher values.

Finally, feature 24 is upward based on the PDP plot, and for values higher than 0.87 the green bar of the LIME plot indicates the prediction is pushing significantly towards "depressed". The SHAP plot also shows that the higher the feature values the higher the probability of a "depressed" prediction.

## 7 Discussion

After thorough research of several methodologies regarding voice detection of depression to support the aim of the present thesis, the SVM algorithm used in the already extracted COVAREP features of the DAIC-WOZ dataset balanced with the SMOTE technique proved the best performance compared to all examined methods.

Considering the results of the proposed algorithm, three features provided significantly important predictions regarding the depressive class, which are the following:

- The presence of feature 239 within the interval of, " $0.29 < \text{feature\_239} \leq 0.87$ " indicates that the likelihood of the "depressed" prediction is significantly decreased, pushing the model's prediction towards the "non-depressed" class. This feature could be considered a protecting factor for this specific interval. Furthermore, feature 239 corresponds to the max value of the 60th initial feature in the original 74 feature set, which means that if the max value of feature 60 is within the interval of, " $0.29 < \text{max\_of\_feature\_60} \leq 0.87$ ", there is a strong indication that the person does not have depression.
- Feature 56, for values less than or equal to -0.53, significantly increases the probability of the category "depressed", according to the LIME chart. Furthermore, according to feature and permutation importance metrics, this feature has also proved a negative coefficient regarding the "depressed" class. These statements conclude with the fact that the higher the values of feature 56 the less evidence of depression, and the lower the values the greater the likelihood of a "depressed" class, so it could be considered as a risk factor for values less or equal to -0.53 and as a protecting factor for higher values. In addition, this feature is the mean value of the 14th initial feature in the original 74 feature set, which means that if the mean value of feature 14 is less or equal to -0.53, there is a strong indication that the person might have depression otherwise if its mean value is higher there is a strong indication that this person does not.
- Feature 60, for values higher than 0.55, significantly increases the likelihood of predicting "depressed" and all the metrics taken into account in this present research confirm this statement, so this feature could be considered as a risk factor. Finally, feature 60 corresponds to the mean value of the 15th initial feature in the initial feature set, which means that if the mean value of feature 15 exceeds 0.55, there is a strong indication that the person has depression.

Regarding the performance of the proposed model, due to its high recall score, it has the advantage of catching most of the true "depressed" cases. It also shows a good balance between precision and recall (as shown by the F1-score) pointing to another strength in distinguishing the depressed from non-depressed cases. However, the specificity score indicates an increased number of false positives and could be a useful point for further improvement.

The present thesis results cannot be directly compared to the related work provided in Chapter 3 because in the research papers, different methods of balancing and feature extraction have been used, directly from the raw wave, already extracted COVAREP features were not used, and in most of them multimodal approaches have been adopted.

Nevertheless, cross-referencing is useful to be conducted. The present thesis shows higher scores in terms of accuracy compared to the spectrogram-based CNN (59.2%) and End-to-End CNN (61.32%) of the «An End-to-End model for Detection and Assessment of Depression Levels using Speech» [17], the same happens with the Convolutional Neural Networks (CNNs) (65.60%) and Long Short-Term Memory (LSTM) (66.25%) results of the «End-to-End Multimodal Clinical Depression Recognition using Deep Neural Networks: A comparative Analysis» [13] and MFCC-based Recurrent Neural Network (76.27%) of «MFCC-based Recurrent Neural Network for Automatic Clinical Depression Recognition and Assessment from Speech» [14], but they also differentiate in the other metrics. F1 scores of Transformer and Parallel Convolutional Neural Networks (CNNs) (93.6%) of «Depression Detection in Speech Using Transformer and Parallel Convolutional Neural Networks» [50] have outperformed the present thesis, as well as in «Depression Speech Recognition With a Three-Dimensional Convolutional Network» [12] with higher accuracy( 83.1%) and slightly lower F1 Score (0.812).

## 8 Conclusion and Future Work

The present thesis researched the topic of the speech-based detection of depression. Starting with a general overview of this psychological condition, someone can easily come to the conclusion that symptoms can vary from mild or not even visible to very severe, leading even to suicide. It affects a great number of people, in a percentage around 3.4% of the global population while 1 to 6 people have experienced depression at least one time in their lives, based on official published scientific data. The proper treatment relies on detecting the severity of each unique case and can be either a pharmaceutical medication, psychotherapy, or a combination of both.

Since there is a significant gap in screening technologies for depression, scientists have been driven to research various parameters, and voice has proven to be a very promising indicator. According to the bibliography, many bioacoustics and linguistic patterns connected with speech and voice have been associated with neuropsychological diseases and more specifically with depression.

On the other hand, the rise of applied sciences and artificial intelligence have opened the path for research through machine learning methodologies that can be combined in complex ways and currently in a short time with powerful computing power. Some examples of promising algorithms used in the present thesis, are Support Vector Machine, AdaBoost, Decision Trees, and Neural Networks.

In this current work, the DAIC-WOZ dataset, with 189 English interview sessions included, was used as a source of research, and more specifically the already extracted COVAREP features. 74 features in each session were processed via Python programming with the Anaconda package. The Support Vector Machine algorithm with the SMOTE balancing technique proved the best performance compared to all the examined methods with 81% accuracy, 74% F1-score, and 72% specificity. LIME, SHAP, PDP, feature and permutation importance, and confusion matrices were used to support the interpretation of the model's predictions.

The results of the present study highlight a significant effect of three specific characteristics on the prediction classes. More specifically based on the machine learning outcome, it was noticed that when the max value of feature 60 is within the interval of " $0.29 < \text{max\_of\_feature\_60} \leq 0.87$ ", there is a strong indication that the person does not have depression. Furthermore, when the mean value of feature 14 is " $\text{mean\_of\_feature\_14} \leq -0.53$ ", there is a strong indication that the person suffers from depression otherwise as the mean value increases the probability of depression decreases. Finally, when the mean value of feature 15 is " $\text{mean\_of\_feature\_15} > 0.55$ ", there is a strong indication that the person has depression. These results indicate that feature 60 can be considered a protecting factor, feature 15 a risk factor, and feature 14 can be both depending on its value.

Eventually, the model performs well with a significant focus on identifying "depressed" instances according to the high accuracy and F1-score. The slightly weak specificity score gives room for further development with appropriate parameter tuning to minimize the false positive results and identify more accurately the "non-depressed" cases, avoiding false predictions and reducing unnecessary stress for patients. This adjustment could enhance this

machine learning model to become a low-cost and non-invasive additional tool for physicians to improve their diagnostic process by complying with all requirements regarding data privacy. Finally, this algorithm was trained including voice patterns in the English language. Many of the acoustic features analyzed may be applicable across languages, yet, further improvement with cross-language validation would ensure that the findings are also valid in other linguistic contexts.

## References

- [1] <https://worldpopulationreview.com/country-rankings/depression-rates-by-country>, (n.d.).
- [2] [https://www.who.int/news/item/02-03-2022-covid-19-pandemic-triggers-25-increase-in-prevalence-of-anxiety-and-depression-worldwide#:~:text=Wake%2Dup%20call%20to%20all,mental%20health%20services%20and%20support&text=In%20the%20first%20year%20of,Health%20Organization%20\(WHO\)%20today,](https://www.who.int/news/item/02-03-2022-covid-19-pandemic-triggers-25-increase-in-prevalence-of-anxiety-and-depression-worldwide#:~:text=Wake%2Dup%20call%20to%20all,mental%20health%20services%20and%20support&text=In%20the%20first%20year%20of,Health%20Organization%20(WHO)%20today,) (n.d.).
- [3] C. Zhuo, G. Li, X. Lin, D. Jiang, Y. Xu, H. Tian, W. Wang, X. Song, The rise and fall of MRI studies in major depressive disorder, *Transl. Psychiatry* 9 (2019) 335. <https://doi.org/10.1038/s41398-019-0680-6>.
- [4] <https://www.nimh.nih.gov/health/topics/depression>, (n.d.).
- [5] <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8599822/>, (n.d.).
- [6] Investigation of Speech Landmark Patterns for Depression Detection, (n.d.).
- [7] S.A. Almaghrabi, S.R. Clark, M. Baumert, Bio-acoustic features of depression: A review, *Biomed. Signal Process. Control* 85 (2023) 105020. <https://doi.org/10.1016/j.bspc.2023.105020>.
- [8] J.S. Buyukdura, S.M. McClintock, P.E. Croarkin, Psychomotor retardation in depression: Biological underpinnings, measurement, and treatment, *Prog. Neuropsychopharmacol. Biol. Psychiatry* 35 (2011) 395–409. <https://doi.org/10.1016/j.pnpbp.2010.10.019>.
- [9] [psychomotor-function-in-affective-disorders-an-overview-of-new-m-1981.pdf](#), (n.d.).
- [10] D. Ververidis, C. Kotropoulos, Emotional speech recognition: Resources, features, and methods, *Speech Commun.* 48 (2006) 1162–1181. <https://doi.org/10.1016/j.specom.2006.04.003>.
- [11] T. Moriyama, S. Ozawa, Emotion recognition and synthesis system on speech, in: *Proc. IEEE Int. Conf. Multimed. Comput. Syst.*, IEEE Comput. Soc, Florence, Italy, 1999: pp. 840–844. <https://doi.org/10.1109/MMCS.1999.779310>.
- [12] H. Wang, Y. Liu, X. Zhen, X. Tu, Depression Speech Recognition With a Three-Dimensional Convolutional Network, *Front. Hum. Neurosci.* 15 (2021) 713823. <https://doi.org/10.3389/fnhum.2021.713823>.
- [13] M. Muzammel, H. Salam, A. Othmani, End-to-end multimodal clinical depression recognition using deep neural networks: A comparative analysis, *Comput. Methods Programs Biomed.* 211 (2021) 106433. <https://doi.org/10.1016/j.cmpb.2021.106433>.
- [14] E. Rejaibi, A. Komaty, F. Meriaudeau, S. Agrebi, A. Othmani, MFCC-based Recurrent Neural Network for automatic clinical depression recognition and assessment from speech, *Biomed. Signal Process. Control* 71 (2022) 103107. <https://doi.org/10.1016/j.bspc.2021.103107>.
- [15] S. Wang, G. Li, Overview of end-to-end speech recognition, *J. Phys. Conf. Ser.* 1187 (2019) 052068. <https://doi.org/10.1088/1742-6596/1187/5/052068>.
- [16] G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M.A. Nicolaou, B. Schuller, S. Zafeiriou, Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network, in: *2016 IEEE Int. Conf. Acoust. Speech Signal Process. ICASSP, IEEE, Shanghai, 2016*: pp. 5200–5204. <https://doi.org/10.1109/ICASSP.2016.7472669>.
- [17] N.S. Srimadhur, S. Lalitha, An End-to-End Model for Detection and Assessment of Depression Levels using Speech, *Procedia Comput. Sci.* 171 (2020) 12–21. <https://doi.org/10.1016/j.procs.2020.04.003>.
- [18] A. Real, F. Dorado, J. Durán, Energy Demand Forecasting Using Deep Learning: Application to the French Grid, 2020. <https://doi.org/10.20944/preprints202003.0158.v1>.
- [19] Awad, M., Khanna, R. (2015). Support Vector Machines for Classification. In: *Efficient Learning Machines*. Apress, Berkeley, CA. [https://doi.org/10.1007/978-1-4302-5990-9\\_3](https://doi.org/10.1007/978-1-4302-5990-9_3), (n.d.).
- [20] W.S. Noble, What is a support vector machine?, *Nat. Biotechnol.* 24 (2006) 1565–1567. <https://doi.org/10.1038/nbt1206-1565>.

- [21] HandWiki, "Support Vector Machine," Encyclopedia, Available: <https://encyclopedia.pub/entry/29353>. Accessed: Jun. 11, 2024., (n.d.). HandWiki, "Support Vector Machine," Encyclopedia, Available: <https://encyclopedia.pub/entry/29353>. Accessed: Jun. 11, 2024.
- [22] R. Wang, AdaBoost for Feature Selection, Classification and Its Relation with SVM, A Review, *Phys. Procedia* 25 (2012) 800–807. <https://doi.org/10.1016/j.phpro.2012.03.160>.
- [23] J. Tang, A. Henderson, P. Gardner, Exploring AdaBoost and Random Forests machine learning approaches for infrared pathology on unbalanced data sets, *The Analyst* 146 (2021) 5880–5891. <https://doi.org/10.1039/D0AN02155E>.
- [24] B. Charbuty, A. Abdulazeez, Classification Based on Decision Tree Algorithm for Machine Learning, *J. Appl. Sci. Technol. Trends* 2 (2021) 20–28. <https://doi.org/10.38094/jastt20165>.
- [25] H. Blockeel, L. Devos, B. Frénay, G. Nanfack, S. Nijssen, Decision trees: from efficient prediction to responsible AI, *Front. Artif. Intell.* 6 (2023) 1124553. <https://doi.org/10.3389/frai.2023.1124553>.
- [26] Y. Goldberg, *Neural Network Methods for Natural Language Processing*, (n.d.).
- [27] K.-L. Du, M. N. S. Swamy, *Neural networks and statistical learning*, Springer, London, United Kingdom, 2019, n.d. [https://www.google.gr/books/edition/Neural\\_Networks\\_and\\_Statistical\\_Learning/IUmvDwAAQBAJ?hl=en&gbpv=1&dq=neural%20networks%20theory%20scholarly%20articles&pg=PA11&printsec=frontcover](https://www.google.gr/books/edition/Neural_Networks_and_Statistical_Learning/IUmvDwAAQBAJ?hl=en&gbpv=1&dq=neural%20networks%20theory%20scholarly%20articles&pg=PA11&printsec=frontcover).
- [28] J. Brownlee, What is the Difference Between a Batch and an Epoch in a Neural Network?, (n.d.).
- [29] L. Pinto-Coelho, How Artificial Intelligence Is Shaping Medical Imaging Technology: A Survey of Innovations and Applications, *Bioengineering* 10 (2023) 1435. <https://doi.org/10.3390/bioengineering10121435>.
- [30] D.P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, (2017). <http://arxiv.org/abs/1412.6980> (accessed October 12, 2024).
- [31] A. Al Bataineh, D. Kaur, M. Al-khassaweneh, E. Al-sharoha, Automated CNN Architectural Design: A Simple and Efficient Methodology for Computer Vision Tasks, *Mathematics* 11 (2023) 1141. <https://doi.org/10.3390/math11051141>.
- [32] L. Alzubaidi, J. Zhang, A.J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M.A. Fadhel, M. Al-Amidie, L. Farhan, Review of deep learning: concepts, CNN architectures, challenges, applications, future directions, *J. Big Data* 8 (2021) 53. <https://doi.org/10.1186/s40537-021-00444-8>.
- [33] R.C. Staudemeyer, E.R. Morris, Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks, (2019). <http://arxiv.org/abs/1909.09586> (accessed June 19, 2024).
- [34] K. Zarzycki, M. Ławryńczuk, LSTM and GRU Neural Networks as Models of Dynamical Processes Used in Predictive Control: A Comparison of Models Developed for Two Chemical Reactors, *Sensors* 21 (2021) 5625. <https://doi.org/10.3390/s21165625>.
- [35] *Recent\_Developments\_in\_Multilayer\_Percep.pdf*, (n.d.).
- [36] D.J. Hand, P. Christen, N. Kirielle, F\*: an interpretable transformation of the F-measure, *Mach. Learn.* 110 (2021) 451–456. <https://doi.org/10.1007/s10994-021-05964-1>.
- [37] M. Sokolova, N. Japkowicz, S. Szpakowicz, Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation, in: A. Sattar, B. Kang (Eds.), *AI 2006 Adv. Artif. Intell.*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006: pp. 1015–1021. [https://doi.org/10.1007/11941439\\_114](https://doi.org/10.1007/11941439_114).
- [38] R.M. AlZoman, M.J.F. Alenazi, A Comparative Study of Traffic Classification Techniques for Smart City Networks, *Sensors* 21 (2021) 4677. <https://doi.org/10.3390/s21144677>.
- [39] S. Lundberg, S.-I. Lee, A Unified Approach to Interpreting Model Predictions, (2017). <http://arxiv.org/abs/1705.07874> (accessed July 1, 2024).

- [40] S.M. Lundberg, G.G. Erion, S.-I. Lee, Consistent Individualized Feature Attribution for Tree Ensembles, (2019). <http://arxiv.org/abs/1802.03888> (accessed July 31, 2024).
- [41] Sinha, R. K. (2024). Book review: Christoph Molnar. 2020., *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*, Metamorph. 231 92-93 (n.d.). <https://doi.org/10.1177/09726225241252009>.
- [42] A. Altmann, L. Toloşi, O. Sander, T. Lengauer, Permutation importance: a corrected feature importance measure, *Bioinformatics* 26 (2010) 1340–1347. <https://doi.org/10.1093/bioinformatics/btq134>.
- [43] G. Degottex, J. Kane, T. Drugman, T. Raitio, S. Scherer, COVAREP &#x2014; A collaborative voice analysis repository for speech technologies, in: 2014 IEEE Int. Conf. Acoust. Speech Signal Process. ICASSP, IEEE, Florence, Italy, 2014: pp. 960–964. <https://doi.org/10.1109/ICASSP.2014.6853739>.
- [44] J.W. Pennebaker, C.K. Chung, M. Ireland, A. Gonzales, R.J. Booth, The University of Texas at Austin and The University of Auckland, New Zealand, (n.d.).
- [45] K.B. Tølbøll, Linguistic features in depression: a meta-analysis, (2019).
- [46] Language Use of Depressed and Depression-Vulnerable College Students, DOI 10108002699930441000030 (n.d.). [https://www.researchgate.net/publication/254221761\\_Language\\_Use\\_of\\_Depressed\\_and\\_Depression-Vulnerable\\_College\\_Students](https://www.researchgate.net/publication/254221761_Language_Use_of_Depressed_and_Depression-Vulnerable_College_Students).
- [47] M. Higuchi, M. Nakamura, S. Shinohara, Y. Omiya, T. Takano, D. Mizuguchi, N. Sonota, H. Toda, T. Saito, M. So, E. Takayama, H. Terashi, S. Mitsuyoshi, S. Tokuno, Detection of Major Depressive Disorder Based on a Combination of Voice Features: An Exploratory Approach, *Int. J. Environ. Res. Public. Health* 19 (2022) 11397. <https://doi.org/10.3390/ijerph191811397>.
- [48] C.W. Espinola, J.C. Gomes, J.M.S. Pereira, W.P. Dos Santos, Detection of major depressive disorder using vocal acoustic analysis and machine learning—an exploratory study, *Res. Biomed. Eng.* 37 (2021) 53–64. <https://doi.org/10.1007/s42600-020-00100-9>.
- [49] T. Takano, D. Mizuguchi, Y. Omiya, M. Higuchi, M. Nakamura, S. Shinohara, S. Mitsuyoshi, T. Saito, A. Yoshino, H. Toda, S. Tokuno, Estimating Depressive Symptom Class from Voice, *Int. J. Environ. Res. Public. Health* 20 (2023) 3965. <https://doi.org/10.3390/ijerph20053965>.
- [50] F. Yin, J. Du, X. Xu, L. Zhao, Depression Detection in Speech Using Transformer and Parallel Convolutional Neural Networks, *Electronics* 12 (2023) 328. <https://doi.org/10.3390/electronics12020328>.
- [51] The Distress Analysis Interview Corpus of human and computer interviews., (n.d.). Gratch J, Artstein R, Lucas GM, Stratou G, Scherer S, Nazarian A, Wood R, Boberg J, DeVault D, Marsella S, Traum DR. The Distress Analysis Interview Corpus of human and computer interviews. InLREC 2014 May (pp. 3123-3128).
- [52] SimSensei kiosk: a virtual human interviewer for healthcare decision support, (n.d.). DeVault, D., Artstein, R., Benn, G., Dey, T., Fast, E., Gainer, A., Georgila, K., Gratch, J., Hartholt, A., Lhommet, M., Lucas, G., Marsella, S., Morbini, F., Nazarian, A., Scherer, S., Stratou, G., Suri, A., Traum, D., Wood, R., Xu, Y., Rizzo, A., and Morency, L.-P. (2014). “SimSensei kiosk: A virtual human interviewer for healthcare decision support”. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS’14)*, Paris.



## Appendix 1: SVM Script

Appendix 1 includes the code used to implement the SVM algorithm described in Chapters 4 and 5 as outlined in methodology 3.

```
import os

import pandas as pd

import numpy as np

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split

from sklearn.svm import SVC

from sklearn.feature_selection import RFE

from sklearn.metrics import accuracy_score, classification_report

from sklearn.metrics import precision_score, recall_score, f1_score, confusion_matrix

import matplotlib.pyplot as plt

import seaborn as sns

from imblearn.over_sampling import SMOTE

from sklearn.impute import SimpleImputer

from sklearn.inspection import permutation_importance, PartialDependenceDisplay

import shap

from sklearn import svm

from lime.lime_tabular import LimeTabularExplainer

from sklearn.tree import DecisionTreeClassifier

from sklearn.tree import plot_tree

import time

# Function to read a CSV file and return a DataFrame

def read_csv_to_dataframe(csv_path):

    try:

        dataframe = pd.read_csv(csv_path, header=None)

        return dataframe
```

```

except FileNotFoundError:
    print(f"File not found: {csv_path}")
    return None

except pd.errors.EmptyDataError:
    print(f"The CSV file {csv_path} is empty.")
    return None

except pd.errors.ParserError:
    print(f"Error parsing the CSV file {csv_path}.")
    return None

```

# Function to extract statistical features from COVAREP matrices

```

def extract_features(df):
    # Extracting statistical features
    features = []
    for column in df.columns:
        features.append(df[column].mean())
        features.append(df[column].std())
        features.append(df[column].min())
        features.append(df[column].max())
    return features

```

# Function to read all CSV files in a directory and concatenate them

```

def read_csv_files_in_directory(directory, numbers_df, label_column):
    all_dataframes = []
    for number in numbers_df[label_column]:
        csv_file_path = os.path.join(directory, f"{number}_COVAREP.csv")
        try:
            df = pd.read_csv(csv_file_path, header=None)
            # Extract features from the COVAREP matrix

```

```

features = extract_features(df)

# Add label to the features

label = numbers_df[numbers_df['Participant_ID'] == number][label_column].values

if len(label) > 0:

    features.append(label[0])

else:

    raise KeyError(f"Label for participant {number} not found")

all_dataframes.append(features)

print(f"Loaded and processed CSV file for number {number}")

except FileNotFoundError:

    print(f"CSV file for number {number} not found at {csv_file_path}")

except KeyError as e:

    print(e)

    print(f"Skipping participant {number} due to missing label.")

return pd.DataFrame(all_dataframes)

```

```

def train_and_evaluate_svm(X_train, y_train, X_test, y_test):

    # Create an SVM classifier

    svm_classifier = SVC(kernel='linear', probability=True, verbose=True)

    # Train the SVM classifier

    svm_classifier.fit(X_train, y_train)

    # Evaluate the SVM classifier

    y_pred = svm_classifier.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)

    return accuracy, y_pred, svm_classifier

```

```

# Directory containing CSV files

csv_directory = r"D:\COVAREP"

```

```

# Load the CSV files that contain the numbers
numbers_df = pd.read_csv(r"D:\train_split_Depression_AVEC2017.csv")
numbers_df2 = pd.read_csv(r"D:\full_test_split.csv")
numbers_df3 = pd.read_csv(r"D:\dev_split_Depression_AVEC2017.csv")

# Read all CSV files in the directory and apply label from numbers_df
print("Reading CSV files for training data...")
train_df = read_csv_files_in_directory(csv_directory, numbers_df, 'PHQ8_Binary')
print("Reading CSV files for development data...")
dev_df = read_csv_files_in_directory(csv_directory, numbers_df3, 'PHQ8_Binary')
print("Reading CSV files for test data...")
test_df = read_csv_files_in_directory(csv_directory, numbers_df2, 'PHQ_Binary')

# Combine all dataframes vertically
print("Combining all dataframes vertically...")
combined_df = pd.concat([train_df, dev_df, test_df], axis=0, ignore_index=True)

# Check sizes after concatenation
print(f"Size of combined_df: {combined_df.shape}")

# Ensure the last column is the label column
features = combined_df.iloc[:, :-1]
labels = combined_df.iloc[:, -1]

# Replace infinite values with NaN
features.replace([np.inf, -np.inf], np.nan, inplace=True)

# Check for NaN values and handle them using SimpleImputer
imputer = SimpleImputer(strategy='mean')

```

```

features = imputer.fit_transform(features)

# Balance the dataset using SMOTE
print("Balancing the dataset using SMOTE...")
smote = SMOTE(random_state=42)
X_balanced, y_balanced = smote.fit_resample(features, labels)

# Check sizes after balancing
print(f"Size of X_balanced: {X_balanced.shape}, y_balanced: {y_balanced.shape}")

# Standardize features using StandardScaler
scaler = StandardScaler()
X_balanced = scaler.fit_transform(X_balanced)

# Split data into train and test sets
print("Splitting data into train and test sets...")
X_train, X_test, y_train, y_test = train_test_split(X_balanced, y_balanced, test_size=0.2,
random_state=42)
print("Data split completed.")
print(f"Training set size: {X_train.shape}, Test set size: {X_test.shape}")
###
# Measure Training Time
start_train_time = time.time()

# Train and evaluate the SVM classifier using the full dataset
print("Training and evaluating the SVM classifier using the full dataset...")
accuracy_full, y_pred_full, svm_classifier = train_and_evaluate_svm(X_train, y_train, X_test,
y_test)

end_train_time = time.time()

```

```

training_time = end_train_time - start_train_time
print(f"Training Time: {training_time:.4f} seconds")

# Calculate the Confusion Matrix
cm = confusion_matrix(y_test, y_pred_full)
tn, fp, fn, tp = cm.ravel()

# Plot and save the Confusion Matrix as an image
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False,
            xticklabels=['Not Depressed', 'Depressed'],
            yticklabels=['Not Depressed', 'Depressed'])
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('Confusion Matrix')
plt.savefig('confusion_matrix.png')
plt.show()

# Calculate Precision, Recall, F1-Score, and Specificity
precision = precision_score(y_test, y_pred_full)
recall = recall_score(y_test, y_pred_full)
f1 = f1_score(y_test, y_pred_full)
specificity = tn / (tn + fp)

# Interpretation of the Results
print("\nInterpretation:")

print(f"Accuracy: {accuracy_full:.2f} - The proportion of correctly predicted instances out of
all instances.")

```

```

print(f"Precision: {precision:.2f} - The proportion of positive predictions that are actually correct.")
print(f"Recall: {recall:.2f} - The proportion of actual positives that are correctly predicted.")
print(f"F1-Score: {f1:.2f} - The harmonic mean of precision and recall, balancing the two.")
print(f"Specificity: {specificity:.2f} - The proportion of actual negatives that are correctly identified.")
print(f"\nConfusion Matrix Interpretation:")
print(f"True Positives (TP): {tp} - Correctly predicted 'Depressed' instances.")
print(f"False Positives (FP): {fp} - Incorrectly predicted 'Depressed' instances (should be 'Not Depressed').")
print(f"True Negatives (TN): {tn} - Correctly predicted 'Not Depressed' instances.")
print(f"False Negatives (FN): {fn} - Incorrectly predicted 'Not Depressed' instances (should be 'Depressed').")

# Explain the model's predictions using SHAP
explainer = shap.Explainer(svm_classifier.predict, X_train, max_evals=2 * X_train.shape[1] + 1)
shap_values = explainer(X_train)

# Plot feature importance
shap.summary_plot(shap_values, X_train)

# LIME Integration
# Explain a single prediction using LIME
explainer = LimeTabularExplainer(X_train, feature_names=[f"feature_{i}" for i in range(X_train.shape[1])], class_names=['Not Depressed', 'Depressed'], verbose=True, mode='classification')

# Choose an instance to explain
i = 0 # index of the instance to explain
exp = explainer.explain_instance(X_train[i], svm_classifier.predict_proba, num_features=20)

```

```

# Display the explanation using matplotlib
exp.as_pyplot_figure()
plt.show()

# Partial Dependence Plot (PDP)
feature_names = [f"feature_{i}" for i in range(X_train.shape[1])] # Replace with actual feature
names if available
features_to_plot = [239, 56, 91, 162, 258, 60, 24, 46, 127, 86] # Adjust based on feature
importance or specific features of interest

fig, ax = plt.subplots(figsize=(14, 16))
pdp_disp = PartialDependenceDisplay.from_estimator(
    svm_classifier, X_train, features_to_plot, feature_names=feature_names, ax=ax
)
plt.subplots_adjust(hspace=0.4, wspace=0.4) # Adjust hspace and wspace to reduce label
overlap

# 1.a. Get the feature importance (coefficients from SVM)
feature_importance = svm_classifier.coef_.flatten() # Flatten if needed
print(f"Feature importance (coefficients): {feature_importance}")

# 1.b. Plot Feature Importance
plt.figure(figsize=(10, 6))
plt.bar(np.arange(len(feature_importance)), feature_importance, color='skyblue')
plt.xlabel('Feature Index')
plt.ylabel('Coefficient Value')
plt.title('Feature Importance (SVM Coefficients)')
plt.show()

# 1.c. Identify and print features with positive and negative coefficients
positive_feature_indices = np.where(feature_importance > 0)[0]

```



```

negative_feature_indices = np.where(feature_importance < 0)[0]
print(f"Features with positive feature importance: {positive_feature_indices}")
print(f"Features with negative feature importance: {negative_feature_indices}")

# 2.a. Get the permutation importance
result = permutation_importance(svm_classifier, X_test, y_test, n_repeats=10,
random_state=42)
permutation_importance_mean = result.importances_mean
print(f"Permutation importance: {permutation_importance_mean}")

# 2.b. Plot Permutation Importance
plt.figure(figsize=(10, 6))
plt.bar(np.arange(len(permutation_importance_mean)), permutation_importance_mean,
color='lightcoral')
plt.xlabel('Feature Index')
plt.ylabel('Mean Importance (Permutation)')
plt.title('Permutation Importance')
plt.show()
positive_importance_indices = np.where(permutation_importance_mean > 0)[0]
print(f"Features with positive permutation importance: {positive_importance_indices}")

```

## Appendix 2: NN Script

Appendix 2 includes the code used to implement the NN algorithm described in Chapters 4 and 5 as outlined in methodologies 1 and 2.

```
import os

import pandas as pd

import random

import numpy as np

import tensorflow as tf

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense

from tensorflow.keras.optimizers import Adam

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score

from sklearn.metrics import confusion_matrix

# Function to read a CSV file and return a DataFrame

def read_csv_to_dataframe(csv_path):

    try:

        dataframe = pd.read_csv(csv_path, header=None)

        return dataframe

    except FileNotFoundError:

        print(f"File not found: {csv_path}")

        return None

    except pd.errors.EmptyDataError:

        print(f"The CSV file {csv_path} is empty.")

        return None

    except pd.errors.ParserError:

        print(f"Error parsing the CSV file {csv_path}.)")
```

```

return None

# Function to read all CSV files in a directory and concatenate them
def read_csv_files_in_directory(directory, numbers_df):
    all_dataframes = []
    for number in numbers_df['Participant_ID']:

        csv_file_path = os.path.join(directory, f"{number}_COVAREP.csv")
        try:

            # Extracting the filename without extension
            filename = os.path.splitext(os.path.basename(csv_file_path))[0]

            # Extracting the number part from the filename
            file_number = int(filename.split('_')[0])

            # Check if the file number is less than 400
            if file_number < 493:

                df = pd.read_csv(csv_file_path, header=None, skiprows=2000, nrows=7000) # Read with
header=None

                # Now you have loaded the CSV file corresponding to the current number

                # Check the value of 'PHQ8_Binary' column

                df.reset_index(drop=True, inplace=True)
                phq8_binary = numbers_df[numbers_df['Participant_ID'] == number]['PHQ8_Binary'].values[0]

                # Create a new column filled with '0' or '1' based on the value of 'PHQ8_Binary'
                df['New_Column'] = phq8_binary

                # Now you have added the new column to the dataframe
                print(f"Loaded CSV file for number {number}")

                print(df.head()) # For demonstration, printing the first few rows

                all_dataframes.append(df)
        except FileNotFoundError:

            print(f"CSV file for number {number} not found at {csv_file_path}")

```

```
return all_dataframes
```

```
# Function to read all CSV files in a directory and concatenate them (for numbers_df2)
```

```
def read_csv_files_in_directory_numbers_df2(directory, numbers_df):
```

```
    all_dataframes = []
```

```
    for idx, row in numbers_df.iterrows():
```

```
        csv_file_path = os.path.join(directory, f"{row['Participant_ID']}_COVAREP.csv")
```

```
        try:
```

```
            # Extracting the filename without extension
```

```
            filename = os.path.splitext(os.path.basename(csv_file_path))[0]
```

```
            # Extracting the number part from the filename
```

```
            file_number = int(filename.split('_')[0])
```

```
            # Check if the file number is less than 400
```

```
            if file_number < 493:
```

```
                df = pd.read_csv(csv_file_path, header=None, skiprows=2000, nrows=7000) # Read with  
header=None
```

```
                # Now you have loaded the CSV file corresponding to the current number
```

```
                # Check the value of 'PHQ8_Binary' column
```

```
                # Check if any row is empty, if so, ignore this dataframe
```

```
                if df.empty:
```

```
                    continue
```

```
                # Now you have loaded the CSV file corresponding to the current participant
```

```
                # Check the value of 'PHQ_Binary' column
```

```
                phq_binary = row['PHQ_Binary']
```

```
                # Create a new column filled with '0' or '1' based on the value of 'PHQ_Binary'
```

```
                df['New_Column'] = phq_binary
```

```
                # Now you have added the new column to the dataframe
```

```

    print(f"Loaded CSV file for number {row['Participant_ID']}")
    print(df.head()) # For demonstration, printing the first few rows
    all_dataframes.append(df)
except FileNotFoundError:
    print(f"CSV file for number {row['Participant_ID']} not found at {csv_file_path}")
return all_dataframes

# Directory containing CSV files
csv_directory = r"D:\COVAREP"

# Load the CSV files that contain the numbers
numbers_df = pd.read_csv(r"D:\train_split_Depression_AVEC2017.csv")
numbers_df2 = pd.read_csv(r"D:\full_test_split.csv")
numbers_df3 = pd.read_csv(r"D:\dev_split_Depression_AVEC2017.csv")

# Read all CSV files in the directory and apply label from numbers_df
all_dataframes = read_csv_files_in_directory(csv_directory, numbers_df)
all_dataframes3 = read_csv_files_in_directory(csv_directory, numbers_df3)

# Read CSV files for numbers_df2 and apply label
all_dataframes2 = read_csv_files_in_directory_numbers_df2(csv_directory, numbers_df2)

# Combine all dataframes into one list
#all_dataframes.extend(all_dataframes3)
#all_dataframes3.extend(all_dataframes2)

combined_df = pd.concat([pd.concat(all_dataframes), pd.concat(all_dataframes2),
pd.concat(all_dataframes3)], axis=0)

print(combined_df)

```

```

# Display sizes of the original DataFrames
# for i, df in enumerate(combined_df, start=1):
#     print(f"Size of DataFrame {i}:", df.shape)

# Concatenate all DataFrames vertically
#final_df = pd.concat(all_dataframes, ignore_index=True)

# Display size of the final concatenated DataFrame
print("Size of Concatenated DataFrame:", combined_df.shape)

# Stack the DataFrame into a Series
#stacked_series = combined_df.stack()

# Convert the stacked Series to numeric type, coercing errors to NaN
#numeric_series = pd.to_numeric(stacked_series, errors='coerce')

# Check if there are any non-numeric values (NaN) in the Series
#if numeric_series.isna().any():
#     print("There are non-numeric values in the dataframe.")
#else:
#     print("All values in the dataframe are numeric.")

# Display the final concatenated DataFrame
#print("Final Concatenated DataFrame:")
#print(final_df)

# Count the number of zeros and ones in the last column
num_zeros = (combined_df.iloc[:, -1] == 0).sum()
num_ones = (combined_df.iloc[:, -1] == 1).sum()

print("Number of zeros before deletion:", (combined_df.iloc[:, -1] == 0).sum())
print("Number of ones before deletion:", (combined_df.iloc[:, -1] == 1).sum())

```

```

# Create a boolean mask for rows where the last column is equal to 0
zero_mask = combined_df.iloc[:, -1] == 0

# Sample the rows with label 0 to match the number of rows with label 1
to_keep = combined_df[zero_mask].sample(n=num_ones, random_state=42)

# Concatenate the sampled rows with the rows with label 1
balanced_df = pd.concat([to_keep, combined_df[~zero_mask]])

# Confirm that the number of zeros and ones is now equal
print("Number of zeros after deletion:", (balanced_df.iloc[:, -1] == 0).sum())
print("Number of ones after deletion:", (balanced_df.iloc[:, -1] == 1).sum())

# Split data into train and test sets
x_train, x_test, y_train, y_test = train_test_split(balanced_df.iloc[:, :-2], balanced_df.iloc[:, -1],
test_size=0.2, random_state=42)

# Standardize features using StandardScaler
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)

# Define a simple neural network model
model = Sequential([
    Dense(64, activation='relu', input_shape=(x_train.shape[1],)),
    Dense(1, activation='sigmoid')
])

```

```

# Compile the model

model.compile(optimizer=Adam(learning_rate=0.001), loss='binary_crossentropy',
metrics=['accuracy'])

# Train the model on the training set with validation data

history = model.fit(x_train, y_train, epochs=10, batch_size=64) # Increase epochs if necessary

# Evaluate the model on the test set

test_loss, test_accuracy = model.evaluate(x_test, y_test)

print(f'Test Accuracy: {test_accuracy * 100:.2f}%')

# Predictions on the test set

test_probabilities = model.predict(x_test)

test_predictions = (test_probabilities > 0.5).astype(int)

# Create a confusion matrix

cm = confusion_matrix(y_test, test_predictions)

# Plot confusion matrix

plt.figure(figsize=(8, 6))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False)

plt.title('Confusion Matrix')

plt.xlabel('Predicted Label')

plt.ylabel('True Label')

plt.show()

```