BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER SYSTEMS
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

ANALYZING A PERSON'S HANDWRITING FOR RECOGNIZING HIS/HER EMOTIONAL STATE

ANALÝZA RUKOPISU ČLOVĚKA PRO ROZPOZNÁNÍ JEHO/JEJÍHO EMOČNÍHO STAVU

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR **Bc. ALEŠ CHUDÁREK**
AUTOR PRÁCE

SUPERVISOR **doc. AAMIR SAEED MALIK, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2024

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Master's Thesis Assignment

153394

Institut:	Department of Computer Systems (DCSY)
Student:	**Chudárek Aleš, Bc.**
Programme:	Information Technology and Artificial Intelligence
Specialization:	Machine Learning
Title:	**Analyzing a person's handwriting for recognizing his/her emotional state**
Category:	Biocomputing
Academic year:	2023/24

Assignment:

1. Study and learn about the various emotions and moods and how they affect the various features of the handwriting.
2. Get acquainted with image & video processing methods as well as machine learning techniques and their application to the recognition of emotions and moods.
3. Find out the challenges for emotion and mood interpretation from handwriting as well as the limitations of the existing methods.
4. Design an algorithm for interpretation of emotion and mood from a person's handwriting.
5. Implement the designed algorithm.
6. Create a set of benchmark tasks to evaluate the quality of emotion and mood recognition from a person's handwriting as well as the corresponding computational performance and memory usage.
7. Conduct critical analysis and discuss the achieved results and their contribution.

Literature:
Note: Dataset is available for this project.

- According to supervisor's advice.

Requirements for the semestral defence:

- Items 1 to 4 of the assignment.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor:	**Malik Aamir Saeed, doc., Ph.D.**
Head of Department:	Sekanina Lukáš, prof. Ing., Ph.D.
Beginning of work:	1.11.2023
Submission deadline:	17.5.2024
Approval date:	30.10.2023

Abstract

Emotion recognition from handwriting is a challenging and interdisciplinary task that can provide insights into the psychological and emotional aspects of the writer. In this study, we developed and evaluated a machine learning model that can predict the emotional state of a writer from their handwriting samples. We utilized the EMOTHAW dataset, which consists of handwriting and drawing samples from subjects whose emotional states are measured by the DASS test, which gives a score for depression, anxiety, and stress and the CIU Handwritten database for verification and experimentation. We extracted a large number of features that are inspired by the standard graphology work, as well as features that are specific to online data. We used ANOVA to select statistically significant features and normalized the data using Z-Score, MinMax, IQR or Log. We reduced the dimensionality of the features using Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). We employed a meta approach Ensemble learning that seeks to reduce the errors of a single model by exploiting the diversity and complementarity of multiple models. The structure of our classifier is dependent on multiple arguments resulting in over 300,000 different configurations. We optimized arguments using argument freezing. We found the best classifiers for binary and trinary classification for each emotion, resulting in six optimal models. We evaluated our models using different metrics, such as accuracy, precision, recall, and F1-score. Our models reached adequate results in all metrics. In addition to finding the classifiers, this thesis explored the importance of each extracted feature, providing a sorted list of the most significant features used for emotion recognition from handwriting. We also enhanced the EMOTHAW database by identifying tasks that are more indicative of specific emotions, thereby reducing the need for a full task battery for emotional analysis.

Abstrakt

Rozpoznávání emocí z rukopisu je náročný a interdisciplinární úkol, který může poskytnout vhled do psychologického a emočního stavu pisatele. V této diplomové práci byl vyvinut a vyhodnocen model strojového učení schopný predikovat emoční stav pisatele na základě vzorků jeho rukopisu. Byl využit dataset EMOTHAW, který obsahuje vzorky rukopisu a kreseb od subjektů, jejichž emoční stavy byly změřeny pomocí testu DASS, který hodnotí úroveň deprese, úzkosti a stresu, a CIU Handwritten databázi pro ověření a experimentování. Bylo extrahováno množství příznaků inspirovaných standardní grafologií, stejně jako příznaky specifické pro online data. Pomocí ANOVA byly vybrány statisticky významné příznaky, které byly normalizovány pomocí Z-Score, MinMax, IQR nebo logaritmické transformace. Dimenzionalita příznaků byla snížena pomocí analýzy hlavních komponent (PCA) a lineární diskriminační analýzy (LDA). Pro klasifikaci byl použit meta-přístup Ensemble learning, který se snaží snížit chyby jednoho jednoduchého modelu využitím rozmanitosti a doplňkovosti více modelů. Struktura klasifikátoru závisí na mnoha argumentech, což vede k více než 300 000 různým konfiguracím. Optimální argumenty a tudíž optimální struktura byla hledána pomocí zamrazování argumentů. Byly identifikovány nejlepší klasifikátory pro binární a trinární klasifikaci každé emoce, což vedlo k šesti optimálním modelům. Tyto modely byly hodnoceny pomocí různých metrik, jako jsou accuracy, precision, recall a F1 Skóre, a dosáhly adekvátních výsledků ve všech metrikách. Kromě nalezení klasifikátorů tato práce zkoumala význam každého extrahovaného příznaku, čímž byl vytvořen seznam nejvýznamnějších příznaků použitých pro rozpoznávání emocí z rukopisu. Dále tato práce rozšiřuje databázi EMOTHAW identifikací úkolů, které jsou více indikativní pro specifické emoce, čímž se snižuje potřeba kompletní baterie úkolů pro emoční analýzu.

Keywords

Emotion recognition, Graphology, Depression, Anxiety, Stress, Handwriting analysis, Machine learning, Classification, Dataset, Feature, Preprocessing, Feature extraction, Feature selection, ANOVA, Normalization, Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Random forest, Support Vector Machine (SVM), K-Nearest Neighbors algorithm (KNN), Artificial neural network (ANN), Decision tree (DT), Esemble learning, Meta-model, Arguments, Confusion matrix

Klíčová slova

Rozpoznávání emocí, Grafologie, Deprese, Úzkost, Stres, Analýza rukopisu, Strojové učení, Klasifikace, Dataset, Příznak, Předzpracování, Extrakce příznaků, Výběr příznaků, ANOVA, Normalizace, Analýza hlavních komponent (PCA), Lineární diskriminační analýza (LDA), Random forest, Support Vector Machine (SVM), algoritmus K-nejbližších sousedů (KNN), Umělá neuronová síť (ANN), Rozhodovací strom (DT), Esemble learning, Meta-model, Argumenty, Matice záměn

Reference

CHUDÁREK, Aleš. *Analyzing a person's handwriting for recognizing his/her emotional state*. Brno, 2024. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor doc. Aamir Saeed Malik, Ph.D.

Rozšířený abstrakt

V rámci této diplomové práce se zaměřujeme na rozpoznávání emocí z rukopisu, což je interdisciplinární oblast na pomezí psychologie, grafologie a strojového učení (ML). Emoce mají klíčový význam v našem každodenním životě a jejich analýza může přinést přínos v mnoha aplikacích, od diagnostiky duševního zdraví až po zlepšení uživatelské interakce s počítačovými systémy. Hlavním cílem práce je prozkoumat a rozšířit možnosti ML v analýze rukopisu a vyvinout model schopný s vysokou přesností identifikovat specifické emoce, jako jsou deprese, úzkost a stres.

K dosažení tohoto cíle jsme využili dvou datasetů, konkrétně EMOTHAW a CIU Handwritten databáze. Všech sedm úkolů z databáze EMOTHAW bylo použito jako náš hlavní trénovací a validační zdroj dat, zatímco CIU Handwritten databáze sloužila především na validaci a další experimentování. Pro extrakci příznaků byly použity speciálně vyvinuté knihovny v Pythonu, díky nimž jsme mohli analyzovat rukopis z různých úhlů pohledu a získat tak přes 500 příznaků, které jsme následně podrobili důkladné statistické analýze. Tyto příznaky obsahovaly jak klasické grafologické příznaky jako je zkosení, šířka, smyčky, mezery, ale také příznaky extrahovatelné pouze za pomocí speciálního tabletu. Mezi tyto příznaky patří například tlak, sklon pera, rychlost, atd.

Jelikož tyto příznaky popisují velmi rozdílné faktory, mají tak i velmi rozdílné hodnoty. Aby nedošlo k dominanci příznaků s vysokými hodnotami, je zapotřebí hodnoty příznaků normalizovat. Normalizace dat byla provedena s využitím několika metod, aby bylo možné data správně zpracovat a připravit pro další fáze analýzy. Implementovaná je normalizace Z-Score, MinMax, IQR (Mezi kvartilové rozpětí) a Log. Jelikož výběr normalizační metody neměl velký význam na klasifikaci, bylo využito především normalizace MinMax.

Výběr použitých příznaků byl klíčovým krokem našeho výzkumu. Ve většině pročtené literatury se výběrem příznaků nikdo příliš nezabývá a příznaky se tak vyberou podle grafologických standardů. Množství takových příznaků není mnohdy vysoké a nemusí tedy předávat dostatečné množství informace. V naší práci jsme přistoupili k postupu, kdy extrahujeme mnohonásobně větší množství příznaků a automatizujeme proces výběru těch signifikantních. Pomocí analýzy rozptylu (ANOVA) jsme identifikovali příznaky s největším potenciálem pro rozpoznávání emocí. Jednotlivé příznaky byly ohodnoceny a seřazeny podle jejich p hodnoty. ANOVA označuje příznak za statisticky signifikantní pokud tato p-hodnota klesne pod 5 %. Díky tomuto kroku můžou jednotlivé klasifikátory vyžadovat i proměnlivý počet příznaků. Pro zamezení nedostatečného počtu příznaků jsme zavedli argument *minimum_features* který zajistí, aby bylo vybráno alespoň minimum nejsignifikantnějších příznaků. Tento krok byl velmi důležitý, jak jsme zjistili v sekci experimentování.

Pro snížení dimensionality dat jsme podle nastudované literatury použili transformační matice Analýzy hlavních komponent (PCA) a Lineární diskriminační analýzy (LDA). Implementovali jsme čtyři možnosti pro tento krok, a to využití pouze PCA transformační matice, využití pouze LDA transformační matice, využití obou PCA i LDA transformačních matic v tomto pořadí a nebo kompletní vynechání tohoto kroku.

Jádrem našeho přístupu bylo využití Ensemble learning, což je meta přístup pro klasifikaci za pomocí strojového učení. Ensemble learning umožnilo kombinovat síly jednotlivých jednoduchých klasifikátorů a dosáhnout tak vyšší přesnosti a robustnosti. Díky tomu jsme mohli prozkoumat různé struktury našeho modelu a optimalizovat je pro konkrétní úkoly rozpoznávání emocí. Ensemble learníng nabízí 3 metody, a to bagging, boosting a stacking. Bagging (bootstrap aggregating) je metoda která trénuje ten samý model vícekrát na jiném subsetu trénovacích dat. Tyto subsety jsou vytvořeny postupným náhodným výběrem s nahrazením. Výsledná predikce tohoto modelu je pak získána hlasováním většiny. Boosting

je metoda, která připojuje sekvenčně další model, zaměřující se na chybové případy jeho předchůdce. Počáteční model je trénován a ohodnocen na trénovací sadě, kde všechna data mají stejnou váhu při hodnocení. Následně je seznam vah změněn podle predikcí tohoto klasifikátoru. Trénovací dataset s novými vahami je pak vstupem do následujícího modelu. Tento proces je opakován podle inicializovaného počtu vnitřních modelů. Stacking (stacked generalization) využívá na rozdíl od předchozích metod více různých klasifikačních modelů. Každý z těchto vnitřních modelů je natrénován na subsetu trénovacích dat a jejich výsledné predikce jsou pak vstupem do meta modelu. Tento meta-model je pak trénován na rozeznávání kombinací predikcí vnitřních modelů. Tento přístup se snaží kombinovat silné stránky různých modelů.

Pro široký výběr možností jsme pro ensemble klasifikátory vybrali 25 modelů strojového učení, které mohou hrát roli jak vnitřních modelů, tak roli meta-modelu.

Díky těmto krokům máme velmi dynamický klasifikační model. Jeho konfiguraci popisujeme pomocí několika argumentů, jejichž kombinace pak tvoří různé klasifikátory. Na výběr jsou 3 emoce (deprese, úzkost, stres), 7 různých úkolů, 4 varianty redukce dimensionality, 3 metody Ensemble (bagging, boosting, stacking), 25 možných vnitřních/meta modelů, 25 možných počtů vnitřních modelů a výběr binárního, či trinárního klasifikátoru. Přes 300 000 kombinací.

Abychom nemuseli zkoumat všechny kombinace, využili jsme pro nalezení vhodných argumentů techniku zmražování. Postupně jsme zamrazili všechny argumenty až na jeden na stejné hodnotě a pouze ten jediný jsme měnili a sledovali, jak kvalitní klasifikátory generujeme, jakmile najdeme ideální hodnotu argumentu, zamrazíme ho a iterujeme přes ostatní argumenty.

Výsledky naší práce jsou povzbudivé. Podrobná analýza a optimalizace vnitřní konfigurace vedly k vývoji šesti klasifikátorů, které dosahují vysoké přesnosti a dalších metrik jako je precision, recall a F1 skóre. Našli jsme tři binární a tři trinární klasifikátory pro klasifikaci jednotlivých emocí.

Kromě nalezených klasifikátorů jsme přišli na několikerá zjištění, jako je například to, že k finální klasifikaci celého datasetu stačí jen některé ze sedmi úkolů, jelikož některé úkoly mají větší význam při klasifikaci určité emoce. Dále jme sestrojili seřazený seznam příznaků, které nejvíce napomáhají klasifikaci emocí z rukopisu. Tento seznam byl vytvořen na základě frekvence, s jakou ANOVA jednotlivé příznaky vybrala do klasifikátoru.

Závěrem lze říci, že naše práce přináší nové poznatky do oblasti rozpoznávání emocí z rukopisu a otevírá dveře pro další výzkum. Představuje komplexní metodologii, která může být aplikována v různých oblastech a nabízí solidní základ pro budoucí studie zaměřené na další rozvoj této fascinující disciplíny.

Analyzing a person's handwriting for recognizing his/her emotional state

Declaration

I hereby declare that this Master's thesis was prepared as an original work by the author under the supervision of doc. Aamir Saeed Malik Ph.D.. The supplementary information was provided by Asst. Prof. Dr. Yasemin Bay and doc. Ing. Jiří Mekyska, Ph.D. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Aleš Chudárek
May 16, 2024

Acknowledgements

I extend my appreciation to doc. Aamir Saeed Malik, Ph.D., for his guidance and the connections he facilitated, particularly the introduction to Asst. Prof. Dr. Yasemin Bay and doc. Ing. Jiří Mekyska, Ph.D. His advice and the materials provided were stepping stones in the completion of my research.

Special thanks are due to doc. Ing. Jiří Mekyska, Ph.D., whose invaluable assistance with the handwriting libraries and clarifications on complex topics were instrumental to my work. His expertise and willingness to share knowledge have left a lasting impact on my thesis.

I am also grateful to Asst. Prof. Dr. Yasemin Bay for granting access to her comprehensive database and for her insights into emotion recognition from handwriting, which have enriched my understanding and broadened the scope of my research.

Above all, I owe a debt of gratitude to my friends and family, whose unwavering support was my stronghold during the challenging process of writing this thesis. In particular, Bc. Andrea Chimenti's encouragement and moral support were the bedrock of my perseverance. His guidance, despite facing his own academic challenges, was a source of inspiration and strength. His companionship and motivation were pivotal in maintaining my mental well-being and focus.

Contents

1 Introduction **6**

2 Literature review **8**
2.1 Emotions 8
2.1.1 Handwriting analysis 8
2.2 Databases 9
2.2.1 Methods for Obtaining Handwriting Data 9
2.2.2 Handwriting Data Types 10
2.2.3 Available Datasets 10
2.2.4 Conclusion on Datasets 11
2.3 Data Preprocessing and Feature Extraction 12
2.3.1 Offline data features 12
2.3.2 Online data features 12
2.4 Classification methods 13
2.5 Related work 13
2.6 Chapter summary 16

3 Proposed Methodology **18**
3.1 Proposed Databases 18
3.2 Proposed Data Preprocessing, Feature Extraction and Feature Selection . . 19
3.2.1 Dataset modifications 20
3.2.2 Library for extracting features 20
3.2.3 Normalization of Features 24
3.2.4 Feature Selection via ANOVA 25
3.2.5 Dimensionality Reduction Techniques 25
3.3 Proposed Classification Method 25
3.4 Model Training and Evaluation 28
3.5 Quality metrics 29
3.6 Chapter summary 30

4 Implementation **32**
4.1 Dataset Preprocessing 32
4.2 Feature Extraction 32
4.3 Feature Normalization 33
4.4 Feature Selection 33
4.5 Dimensionality Reduction 34
4.6 Ensemble Classification 35
4.6.1 Ensemble Method Selection 35

4.6.2 Meta-Classifier and Base models . . . 35
4.6.3 Normalization and Feature Selection . . . 35
4.6.4 Data Splitting and Transformation . . . 36
4.6.5 Ensemble Training and Evaluation . . . 36
4.6.6 Result Documentation and Classifier Deployment . . . 36
4.7 Deployment . . . 36
4.7.1 Model and Data Association . . . 36
4.7.2 Feature Extraction . . . 36
4.7.3 Normalization . . . 37
4.7.4 Data Transformation and Classification . . . 37
4.7.5 Performance Metrics and Manual Experimentation . . . 37

5 Experiments **38**
5.1 Argument Search . . . 38
5.1.1 Adaptation to failed experiments . . . 39
5.1.2 Binary classifier . . . 40
5.1.3 Trinary Classifier . . . 45
5.2 Feature Importance Analysis . . . 49
5.3 Comparative Analysis of Induced and Measured Stress . . . 51
5.4 Failed Experiments . . . 52

6 Results **54**
6.1 Optimal Arguments for Classifier Search . . . 54
6.1.1 Binary Classifier Arguments . . . 54
6.1.2 Trinary Classifier Arguments . . . 55
6.2 Performance of Optimal Classifiers . . . 55
6.2.1 Classifier Performance Metrics . . . 55
6.3 Feature Importance in Emotion Classification . . . 56
6.3.1 Key Features for Emotion Classification . . . 56
6.4 Conclusion on the Comparative Analysis of Stress . . . 57
6.4.1 Implications of the Results . . . 57
6.4.2 Future Directions . . . 57
6.4.3 Potential for Merging Stress Conditions . . . 57
6.5 Task Effectiveness in Emotion Classification . . . 58
6.5.1 Binary Classification Task Effectiveness . . . 58
6.5.2 Trinary Classification Task Effectiveness . . . 58

7 Conclusion **59**

Bibliography **61**

A Detailed Account of Unsuccessful Experiments **64**
A.1 Argument Search Using Cross-Validation . . . 64
A.1.1 Binary classifier . . . 64
A.1.2 Trinary Classifier . . . 69
A.2 Concluding Remarks on Preliminary Experiments . . . 73

B Feature Importance for Classification **74**
B.1 Features by Average ANOVA Position . . . 74

B.2 Features by Selection Frequency 75
B.3 Merged Base Features by Selection Frequency 76

List of Figures

3.1 The scatter of the emotion levels of subjects in the EMOTHAW database. . 19
3.2 The example of the SVC data structure ready for feature extraction. 20
3.3 Depiction of Azimuth and Tilt of the pen. 23
3.4 Depiction of Writing width, Writing height, Intra-stroke intersections and Inter-stroke intersections. 23
3.5 Diagram of Bagging ensemble model [8]. 26
3.6 Diagram of Boosting ensemble model [8]. 27
3.7 Diagram of Stacking ensemble model [8]. 28
3.8 The proposed pipline for training a classification model for one emotion. . . 31

5.1 The graph showcases the comparative accuracy of classifiers utilizing a different minimum number of features used for feature selection step. We can see that 30 features is our optimal choice for further argument search. . . . 40
5.2 The graph compares the accuracy of binary classification models under different preprocessing methods, highlighting the superior performance achieved through PCA and the PCA-LDA pipeline. 41
5.3 This graph presents the accuracy of classifiers for each emotion, showcasing the distinct influence of task-specific input data on model performance. . . 42
5.4 Accuracy trends of binary classifiers with varying inner model counts using bagging, highlighting 10 as the optimal number. 43
5.5 Accuracy trends of binary classifiers with varying inner model counts using boosting, highlighting 50 as the optimal number. 43
5.6 Accuracy trends of binary classifiers with varying inner model counts using stacking, highlighting 5 as the optimal number. 44
5.7 The graph compares the accuracy of binary classifiers using different ensemble techniques, showing negligible differences in their performance. 44
5.8 The bar graph illustrates the performance of each meta/base model for classifying emotions, with no single model demonstrating clear superiority. . . . 45
5.9 The graph compares the accuracy of trinary classification models under different preprocessing methods, highlighting the superior performance achieved through PCA and the PCA-LDA pipeline. 46
5.10 This graph presents the accuracy of classifiers for each emotion, showcasing the distinct influence of task-specific input data on model performance in trinary classification. 46
5.11 Accuracy trends of trinary classifiers with varying inner model counts using bagging, highlighting the optimal lower numbers. 47
5.12 Accuracy trends of trinary classifiers with varying inner model counts using boosting, indicating 50 as the optimal number. 48

5.13 Accuracy trends of trinary classifiers with varying inner model counts using stacking, showing a peak at 15 models. 48
5.14 The graph compares the accuracy of trinary classifiers using different ensemble techniques, indicating a broader range of results for stacking and a slight advantage for bagging over boosting. 49
5.15 The bar graph illustrates the performance of each meta/base model for classifying emotions in trinary classification, with some models showing notably poor performance. 49

A.1 The graph showcases the comparative accuracy of binary classification models utilizing a stacking ensemble approach. The models, which include either five or fifteen inner models, are evaluated with a meta-model consisting of KNN, Logistic Regression, or Random Forest. The accuracy is measured across different preprocessing techniques, highlighting the effectiveness of LDA in improving model performance. 65
A.2 The graph showcases the comparative accuracy of binary classification models employing different ensemble techniques. Stacking, bagging, and boosting were analyzed under uniform conditions. The models, which include either five or fifteen inner models, are evaluated with a meta-model consisting of KNN, Logistic Regression, or Random Forest. Stacking showing a slight edge in performance. 66
A.3 The graph presents the accuracy achieved by classifiers for each emotion, highlighting the varying impact of task-specific input data on model performance. 67
A.4 The graph illustrates the relationship between the number of inner models and the accuracy of the classifiers. It highlights the minimal impact that increasing the number of inner models has on the overall accuracy. 68
A.5 The graph displays the performance of classifiers across different meta-models, highlighting the negligible differences in accuracy, suggesting the need for further exploration of all meta-models. 69
A.6 The graph illustrates the accuracy improvements achieved through LDA preprocessing in trinary classification. 70
A.7 The graph compares the performance of different ensemble techniques on trinary data, indicating stacking as the most consistent performer. 70
A.8 The graph reveals task-specific input data's impact on trinary classification accuracy, showing distinct preferences for different emotions. 71
A.9 The graph depicts the subtle effect of varying inner model quantities on the accuracy of trinary classifiers. 72
A.10 This graph compares the accuracy of trinary classifiers across various meta-models, highlighting the overall minimal impact on performance. 72

Chapter 1

Introduction

Emotions are complex and dynamic phenomena that affect human behavior, cognition, and communication. Emotions can be expressed and perceived through various modalities, such as facial expressions, vocal tones, body gestures, and or written texts. Among these modalities, handwriting is a unique and rich source of information that can reveal the emotional state of the writer. Handwriting is influenced by various factors, such as the writer's personality, mood, intention, and context. Therefore, analyzing handwriting can provide insights into the psychological and emotional aspects of the writer.

Emotion recognition from handwriting is a challenging and interdisciplinary task that requires the collaboration of different fields, such as psychology, graphology, computer vision, and machine learning. Emotion recognition from handwriting can have various applications, such as personality assessment, mental health diagnosis, forensic analysis, and human-computer interaction. However, emotion recognition from handwriting is also a relatively new and under-explored domain, which poses many difficulties and limitations, such as the lack of standardized and reliable datasets, the diversity and subjectivity of emotions and handwriting styles, and the complexity and variability of handwriting features and classifiers.

The main goal of this study is to develop and evaluate a machine learning model that can predict the emotional state of a writer from their handwriting. The emotions that we are trying to recognize are depression, anxiety, and stress, which are common and important mental health issues that affect many people. We used the EMOTHAW dataset, which is one of the most comprehensive and reliable datasets for emotion recognition from handwriting, as well as the CIU Handwritten database for verification of our results and other experiments.

In this thesis, we employed fundamental machine learning techniques due to their simplicity, speed, and interpretability. These methods are particularly well-suited for scenarios with limited data and computational resources, unlike deep learning models which require extensive data and processing power. Additionally, the transparency of traditional machine learning algorithms facilitates a clearer understanding of model decisions. We implemented Ensemble learning meta approach to reduce the errors of a single model. We focused on the preprocessing and feature extraction steps, which are essential for transforming the raw data into meaningful and relevant representations that can capture the emotional information. We used various features that are inspired by the standard graphology work, such as slant, baseline, size, pen-pressure, spacing, margins, strokes, loops, etc. We also used some features that are specific to online data, such as speed, acceleration, jerk, curvature, etc. We used statistical tests to select statistically significant features. For dimentionality

reduction we further converted the data using Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA).

Our model's performance is assessed using a suite of metrics—accuracy, precision, sensitivity, specificity, and F1-score—to provide a comprehensive evaluation of its strengths and limitations. By addressing the research gap in metric diversity and feature selection, and employing an ensemble learning strategy, this thesis contributes a novel perspective to the field of emotion recognition from handwriting.

The structure of this thesis unfolds as follows: a literature review that contextualizes our work within the field, a detailed methodology proposal, the implementation of this methodology, a series of experiments to test our hypotheses, and the presentation of our results. We conclude with a discussion on the implications of our findings and suggestions for future research directions.

Our contributions are manifold. We have developed six classifiers with robust confusion matrices for binary and trinary classification of depression, anxiety, and stress. We have enhanced the EMOTHAW database by identifying tasks that are more indicative of specific emotions, thereby reducing the need for a full task battery for emotional analysis. Furthermore, we have ranked a comprehensive list of 525 potential features by their significance in emotion recognition, paving the way for future explorations in this domain.

In summary, this thesis not only advances the understanding of emotion recognition from handwriting but also offers practical tools and methodologies that can be applied in various psychological and computational contexts.

Chapter 2

Literature review

2.1 Emotions

Emotions are reactions that human beings experience in response to events or situations. According to the Oxford Learner's Dictionaries, emotion is: „A strong feeling such as love, fear or anger; the part of a person's character that consists of feelings“ [25]. Emotions have a strong influence on our daily lives and choices. However, emotions are not simple or static phenomena. They are shaped by culture, language, and context. They also vary across time and space. Therefore, understanding emotions and how they are recognized is a challenging and fascinating task for researchers from different disciplines [28].

The history of emotions dates back to ancient times, when philosophers such as Plato, Aristotle, and Seneca tried to define and classify emotions, and to understand their causes and effects. They also proposed different methods for regulating and expressing emotions, such as reason, rhetoric, and ethics. In the Middle Ages and the Renaissance, emotions were often associated with the passions of the soul, and were influenced by religious and moral doctrines [27]. In the Enlightenment and the Romantic era, emotions were seen as natural and individual expressions of the self, and were valued as sources of creativity and inspiration. In the modern and contemporary period, emotions became the subject of scientific inquiry, and were studied by psychologists, sociologists, anthropologists, and neuroscientists, among others. They also became the object of artistic and literary representation, and of political and social manipulation.

The recognition of emotions is the process of identifying and interpreting the emotional states of oneself and others. It is a crucial skill for human communication and social interaction. However, it is not always easy or accurate, as emotions can be subtle, complex, or ambiguous. Different methods and techniques have been developed and used for emotion recognition, such as facial expressions, body language, voice, text, and physiological signals. Each of these methods has its own advantages and limitations, depending on the context and the purpose of the recognition [10, 32, 31, 15]. In this thesis, we will focus on one specific method: emotion recognition from handwriting. We will review the existing literature on this topic, and propose a novel approach based on artificial intelligence and machine learning.

2.1.1 Handwriting analysis

Handwriting analysis, also known as graphology, is a scientific way of determining, assessing, and comprehending personality traits and emotional states based on the strokes and

patterns revealed by handwriting. Graphologists claim that handwriting can reveal various aspects of one's psychological and emotional condition, such as mood, stress, anxiety, depression, honesty, or intelligence [22]. Handwriting analysis is used for various purposes, such as personality assessment, career guidance, forensic examination, and health diagnosis [1]. However, handwriting analysis is not a standardized or widely accepted method, and its validity and reliability are often questioned by critics and researchers. Some of the challenges and limitations of handwriting analysis include the lack of empirical evidence, the influence of external factors, the subjectivity of interpretation, and the ethical and legal issues [12].

2.2 Databases

One of the main challenges in the field of emotion recognition from handwriting is the acquisition of data that reflects the emotional states of the writers. Emotions are complex and dynamic phenomena that are not easy to produce or control on demand. Therefore, different methods have been proposed to elicit, measure or detect emotions in handwriting experiments.

2.2.1 Methods for Obtaining Handwriting Data

In order to perform emotion recognition from handwriting, it is essential to have a reliable way of obtaining the ground truth labels for the handwriting samples. The literature review reveals that there are different methods that have been used for this purpose. These methods can be broadly divided into two main categories, depending on how the emotions are elicited and measured in the participants [21, 3].

Inducing emotions

One of the methods is to induce a certain emotion in a subject and measure it's effectiveness [3]. The most common approach is to use emotion-oriented media, such as videos, music, images, etc., to induce the desired emotion on the subject before or during the handwriting task. This method requires careful selection of the media stimuli that can effectively influence the majority of the subjects in the intended way. The media stimuli can vary in type, intensity, duration, and timing, depending on the research design and the target emotion. The handwriting task can also vary in complexity, content, and modality, such as drawing shapes, copying words, or writing sentences. The handwriting task is usually performed several times, with different media stimuli and different emotions, to capture the changes in the handwriting features that are related to the emotional state of the writer. Some datasets also incorporate a survey for subjective emotion response score, which is a self-report measure of the emotion experienced by the subject after the handwriting task. This measure helps to verify if the media stimuli had the intended effect on the subject and to label the handwriting samples according to the corresponding emotion.

Measuring emotional state

A different method to collect data for emotion recognition from handwriting is to assess the current emotional state of a large number of subjects without manipulating their emotions [21]. Usually, some kind of psychological test is used to evaluate the emotional state or the intensity of each emotion. For example, the Depression Anxiety Stress Scales test (DASS

test) is a self-report measure of depression, anxiety, and stress, which can be taken and evaluated without a professional psychologist present [21].

With this method, the subjects are not exposed to any emotion-oriented media, but it is assumed that the dataset covers a range of emotional states. This method has the drawback of having skewed emotions in the dataset, because some emotions are more frequent or more easily reported than others. The advantage of this method is that it can capture the complexity and diversity of emotions, as one subject can have multiple emotions at the same time. The final evaluation of one's emotional state is a set of scores of all assessed emotions.

2.2.2 Handwriting Data Types

The next crucial step is to collect the data that reflects subjects handwriting or drawing performance. There are two main data types when it comes to handwriting; offline and online data.

Offline data

One of the methods of collecting the written data is offline data collection. This method involves performing the handwriting tasks on normal paper, and then converting the paper documents into digital images. This can be done by scanning or photographing the paper documents. The digital images are then processed as a series of pixels. The only other data point that is available is the evaluation of the emotional state of the writer. This method is simpler and more accessible than online data collection, but it does not capture a lot of crucial information that is related to the handwriting process, such as the duration of the task, the sequence of strokes, the pen pressure, and so on. The benefit of this method is that it can enable a classifier to predict emotions from any texts, even historical ones, without requiring any special equipment [4, 18].

Online data

Online data collection is another method of obtaining the written data. This method involves using a special electronic tablet and pen, that record the written data in real time. The data consists of a stream of information points, such as the pen position, orientation, pressure, and so on. The captured data is determined by the used tablet. According to Doctor Bay's article [3], the writing experience of the subject influences the validity of the data. Therefore, most newer research papers use tablets that can be covered with a normal sheet of paper and pens that can write on the paper as well as collect all the data. This way, the subject can have a natural human experience while writing [21, 29].

2.2.3 Available Datasets

Emotion recognition from handwriting is a complex task that requires suitable datasets to perform. However, there are not many publicly available datasets for this purpose, due to the difficulty of obtaining them. Table 2.1 shows some of the datasets that are available for this study. The most common choice is EMOTHAW [21], which consists of 129 subjects (71 female, 58 male) who performed writing and drawing tasks. Their emotional state was measured 2.2.1 by the DASS test, which gave a score for depression, anxiety, and stress. Another dataset is the CIU Handwritten database, which was used in Doctor Bay's article

[3]. This dataset differs from EMOTHAW in that it uses emotion induction in subjects 2.2.1 by showing them happy and sad media, and simulating stress by imposing a time limit for a task. The last dataset presented is the Personality prediction dataset [11], which is a simple offline dataset of images of student's handwriting, evaluated by a Big five personality test.

Table 2.1: Datasets for emotion recognition from handwriting

Name	Category	Emotions	No. subjects
EMOTHAW	online	depression, anxiety, stress	129
The CIU Handwritten DB	offline/online	happy, sad, stress	134
Personality prediction DB	offline	Big Five personality traits	±110

2.2.4 Conclusion on Datasets

In this section, we reviewed the available datasets for emotion recognition from handwriting. We found that EMOTHAW is the most comprehensive and reliable dataset, as it contains both writing and drawing samples from a large number of subjects, and measures their emotional state using a validated psychological test. Based on the analysis of the datasets, we decided to use basic machine learning methods instead of deep learning methods for our study of emotion recognition from handwriting. We will discuss the classification method in detail later in the thesis 3.3. We justified this decision by considering the following factors:

- Data availability - Deep learning methods require a large amount of labeled data to train and test the models, while basic machine learning methods can work with smaller datasets. The datasets that we reviewed are not sufficient to support deep learning methods, as they have limited samples, labels, and features. Even with data augmentation techniques, such as rotation and scaling, we would not be able to generate enough data for deep learning methods to perform well.

- Data quality - Basic machine learning methods are more robust and can handle noise and outliers in the data, while deep learning methods are sensitive to them. The datasets that we reviewed are not very consistent, as they have variations in handwriting styles, emotions, and tasks. Moreover, some of the labels are subjective and unreliable, such as the personality test scores, which may not reflect the true emotions of the subjects.

- Computational complexity - Deep learning methods require a lot of computational resources and time to train and test the models, while basic machine learning methods are simpler and faster. The computational resources that we have for our study are limited. Therefore, we opted for basic machine learning methods that can run on our personal computers and laptops.

- Interpretability - Basic machine learning models, especially those based on algorithms like decision trees or linear regression, offer clear insights into how input variables are associated with the output. This transparency allows researchers and practitioners to understand and trust the decisions made by the model. In contrast, deep learning models, often described as "black boxes," provide limited insight into their decision-making processes, making them less desirable in fields where understanding the rationale behind predictions is crucial. Given the importance of interpretability in

emotion recognition from handwriting, where deciphering subtle nuances is essential, basic machine learning methods present a more suitable choice for our research.

2.3 Data Preprocessing and Feature Extraction

Data preprocessing is a critical phase in data analysis and machine learning, designed to transform raw data into structured and interpretable information suitable for further processing. This stage encompasses a variety of tasks, such as dimensionality reduction, which simplifies the data by focusing on the most relevant features. Noise removal and outlier detection are employed to enhance data quality by eliminating irrelevant or erroneous data points. Error correction and handling of missing values are also integral to ensure the completeness and accuracy of the dataset [6].

Normalization is a particularly important preprocessing step, especially in the context of handwriting analysis, where it ensures that no single feature disproportionately influences the outcome due to its scale. This process adjusts the features to a common scale, allowing for a balanced evaluation of all attributes. Error correction and handling of missing values are equally crucial, as they address inaccuracies and gaps that could otherwise lead to biased or incorrect model predictions [6].

Once the features are preprocessed, dimensionality reduction techniques such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) can be applied. PCA reduces the feature space by identifying the principal components that capture the most variance, simplifying the data while retaining essential information. LDA, in contrast, aims to maximize class separability, which can be particularly beneficial for distinguishing between different emotional states in handwriting [24, 2].

2.3.1 Offline data features

As discussed in Section 2.2.2, offline data refers to the data that is obtained after the handwriting task is completed, such as the scanned image of the paper. This resembles the traditional graphologist way of analyzing handwriting. Therefore, the extracted features are inspired by the standard graphology work. In [5] paper, Bhattacharya explains how certain features indicate different emotions. These features include Slant, Baseline, Size, Pen-pressure, Spacing, Margins, Strokes, Loops, 't'-bar, 'i'-dots, etc. These features can be computed using image processing techniques, such as edge detection, segmentation, thinning, etc. However in our study, we mainly focus on online data features, which can provide more information than offline data features.

2.3.2 Online data features

As mentioned in Section 2.2.2, online data can provide much more information than offline data, as it captures the dynamic aspects of handwriting. Therefore, it is very important to simplify this data and extract relevant features. Because of the real-time capture of the data, we can also reconstruct the written image, and extract similar features as for offline data, such as Slant, Baseline, Size, etc. Moreover, we can extract additional features that are specific to online data, such as Speed, Acceleration, Jerk, Curvature, etc. These features can be computed using mathematical and statistical methods, such as differentiation, integration, smoothing, etc. For example, the dataset EMOTHAW from article [21] captures for each time stamp an xy-coordinate of the pen tip, pen status (if it's touching the paper

or not), the pressure and tilt and azimuth of the pen. These data points can be used to compute various features that can be used for emotion recognition.

2.4 Classification methods

The final step in emotion recognition from handwriting is the classifier, which is a machine learning model that can predict the emotional state of the writer based on the extracted features from the handwriting samples. However, there is no single optimal classifier for handwriting samples, as different classifiers may have different strengths and weaknesses depending on the data and the task. In my research, We reviewed many papers that attempted to classify emotions from handwriting using simple and generic classification methods, which are widely used in machine learning. The most common methods were **Support Vector Machine (SVM)** [24, 13], **K-Nearest Neighbors algorithm (KNN)** [3, 13], **Repeated Incremental Pruning to Produce Error Reduction (JRIP)** [3, 13], **Random Forest** [3, 2] and more.

2.5 Related work

In this section we will summarize some previous works on similar topic that served as a study material and a reference for comparison with my findings.

Recognition of Emotional State Based On Handwriting Analysis and Psychological Assessment

In this article [18], Dr. Kedar et al. present one of the first attempts to recognize emotions from handwriting using a machine learning approach. The authors use a Convolutional Neural Network (CNN), which is a type of deep learning model that can learn from images, to assign emotions to handwriting samples. The dataset they use is similar to the EMOTHAW database [21], which consists of handwriting and drawing samples from subjects whose emotional states are measured by the DASS test. Detecting negative emotion such as depression, anxiety and stress and their combinations. The dataset contains 1600 samples from different participants, who were asked to write on the A4 paper. The authors evaluated their model with accuracy, precision and recall, which are metrics that measure the quality of the predictions. They report the average accuracy of **91.25%**, which is a high score for this task. They also show the precision-recall values for each emotion in Table 2.2. This paper is one of the few that depicts these metrics, which are very important for evaluating the performance of the model.

Table 2.2: Precision-Recall results from [18] by Dr. Kedar et al. for anxiety-stress, depression, depression-anxiety, depression-anxiety-stress, moderate anxiety, normal and severe anxiety classification.

Class Name	Precision	Recall	F1-score
AS	1.00	0.90	0.95
D	0.83	0.83	0.83
DA	1.00	0.92	0.96
DAS	0.77	1.00	0.87
MA	0.92	0.85	0.88
N	0.92	1.00	0.96
SA	1.00	0.91	0.95

Emotional State Prediction From Online Handwriting and Signature Biometrics

In this article [3], Doctor Bay et al. introduce a novel database 2.2.3 of offline and online handwriting and signature biometrics, which contains emotional status labels (happy, sad, and stress) and demographic information (age, gender, handedness, education level, and nationality) of the writers. The database comprises 134 participants with 804 handwriting and 8040 signature biometric samples. The article also describes experiments on predicting the emotional state of the writers from their biometrics, using different thresholds. The article reports high accuracy for stress detection from handwriting and happiness detection from signature. The article argues that handwriting and signature biometrics provide more information than identity recognition and verification, and can be used for personal characteristics estimation. The article aims to demonstrate the potential of handwriting and signature biometrics for various applications. The emotion prediction model employs 11 features that are commonly used in signature and handwriting processing as hown in Table 2.3. For classification, three classification methods are used and compared (K-Nearest Neighbor (KNN), JRIP and Random Forest). Several experiments are conducted to analyze and demonstrate the emotion prediction accuracy. The resulting accuracy for stress detection from handwriting is **82.52%** (KNN), **89.32%** (Jrip) and **92.23%** (Random Forest). No further metrics than accuracy were provided.

Table 2.3: Features extracted by Doctor Bay in [3].

F1	Average pen velocity in x
F2	Average pen velocity in y
F3	Maximum pen velocity in x - Average pen velocity in x
F4	Maximum pen velocity in y - Average pen velocity in y
F5	Maximum pen velocity in x - Minimum pen velocity in y
F6	Average pen acceleration in x
F7	Average pen acceleration in y
F8	Azimuth
F9	Altitude
F10	Pressure
F11	The number of times pen passes though the midline of the signature/handwriting

Mood State Detection in Handwritten Tasks Using PCA–mFCBF and Automated Machine Learning

In this article [24] Dr. Juan Arturo Nolazco-Flores et al. present a novel method to detect the mood state of a user from their handwriting or drawing. The method uses various features extracted from the sensor data, such as temporal, kinematic, statistical, spectral and cepstral features. The method selects the best features using Principal Component Analysis (PCA) and modified Fast Correlation–Based Filtering (mFCBF), which reduce the dimensionality and redundancy of the features. The method uses automated machine learning to train and test different classifiers, both plain and ensembled, on the EMOTHAW database. The method achieves **100%** accuracy for detecting two mood severities (normal and abnormal), and high accuracy for detecting three mood states.

Emotion Detection from Handwritten Text using Agglomerative Clustering

This paper [4] presented a novel method for emotion detection from offline handwritten text images. The method employs agglomerative hierarchical clustering, a type of unsupervised learning, to assign each image pixel to one of the five predefined emotions: anger, sadness, depression, happiness, and excitement. The clustering is based on the distance between the pixel and the cluster centroid, which is determined by a threshold value. The paper claims that the method achieves an accuracy above **75%** for each emotion, without requiring any annotated data.

Unveiling Emotions through Handwriting: A Data Analysis Approach

This paper [2] applies feature engineering, correlation analysis, and dimensionality reduction techniques, such as Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA), to extract and select the most relevant handwriting features from EMOTHAW writing samples. We can see extracted features in Table 2.4. The paper employs a Random Forest model to predict emotional states from handwriting features, and evaluates the model using accuracy and confusion matrix. The paper attains promising accuracy in binary emotion detection. The accuracies indicate that PCA yields the best result for depression and anxiety (**76.53%, 96.59%**), while LDA produces the best result for depression (**84.9%**). The confusion matrix wasn't included.

Table 2.4: Features extracted by Azmi in [2].

F1	Time spent performing the task while in the air
F2	Total time spent performing task on paper
F3	Duration of the entire task
F4	The count of strokes used to complete the task on the paper

Identifying Dominant Emotional State Using Handwriting and Drawing Samples by Fusing Features

This paper [29] proposes a novel technique to identify the dominant emotional state (depression, anxiety, or stress) of a person from their handwriting and drawing samples. The authors combine temporal, spectral, and Mel Frequency Cepstral Coefficient (MFCC) methods to extract features from the signals, as well as spatial features from the velocities of

the pen movements. A BiLSTM network is employed to classify the features into the emotional states. The technique is evaluated on the EMOTHAW dataset [21] and compared with several baseline approaches, such as SVM, KNN, and CNN. The results show that the technique outperforms the baseline approaches on all emotional states and tasks. The fusion of temporal, spectral, and cepstral features improves the accuracy significantly. The recorded accuracy can be seen in Table 2.5.

Table 2.5: Accuracy results from [29] by Atta Ur Rahman & Zahid Halim

	Drawing	Writing	Both
Depression	83.28	89.21	87.11
Anxiety	76.12	74.54	80.03
Stress	75.39	75.17	74.38

2.6 Chapter summary

One of the main challenges in emotion recognition is to evaluate the performance of different models and methods. As shown in Table 2.6, the state-of-the-art models can achieve high accuracy in classifying emotions from various sources of data. However, accuracy alone is not a sufficient metric to measure the effectiveness of emotion classifiers, especially when dealing with imbalanced or noisy data. Therefore, in this paper, we will also consider other evaluation metrics, such as precision, sensitivity, specificity, F1-score and more. These metrics can provide more insights into the strengths and weaknesses of different approaches. Precision measures the proportion of correctly predicted positive instances among all predicted positive instances, sensitivity measures the proportion of correctly predicted positive instances among all actual positive instances, specificity measures the proportion of correctly predicted negative instances among all actual negative instances, and F1-score is the harmonic mean of precision and sensitivity. These metrics can help us to assess how well a model can identify the relevant emotions and avoid false positives or false negatives. In this paper, we will ensure that all of these metrics are high and reflect the good quality of our model.

Another challenge in emotion recognition is to select and process the appropriate data for training and testing the models. In the literature, we can observe that the most common dataset used for emotion recognition is EMOTHAW or its variants. However, relying on a single dataset poses some risks, such as overfitting, bias and generalization issues. Therefore, we need to be careful about the validity and reliability of the annotations, the representativeness and diversity of the samples, and the comparability and reproducibility of the results.

Due to the limitations of the datasets and their sizes, we decided not to use deep learning methods for emotion recognition. Deep learning methods, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have shown impressive results in various domains, such as computer vision, natural language processing and speech recognition. However, these methods require large amounts of data, high-quality data and high computational resources to achieve their full potential. In contrast, our datasets are relatively small, noisy and heterogeneous, which makes them unsuitable for deep learning methods. Therefore, we opted for a basic machine learning method, which is simpler, faster and more transparent.

Based on the related works, we identified that the most important step in emotion recognition is preprocessing, feature extraction and feature selection. Preprocessing and feature extraction are essential for transforming the raw data into meaningful and relevant representations that can capture the emotional information. Using a higher number of features, along with algorithms that orthogonalize the data and reduce dimensionality, can lead to better results than using a powerful classifier with few feature inputs. Therefore, in this paper, we will focus on developing and evaluating different preprocessing and feature extraction techniques.

Table 2.6: Comparative Analysis of Related Studies. Note: An 'x' denotes that the result was either not mentioned or not calculated in the respective study.

Title	DB	Algorithm	Feature extraction	Acc	Prec	Sens	Spec	F1
Recognition of Emotional State Based On Handwriting Analysis and Psychological Assessment 2.5	personal	Convolutional Neural Network (CNN)	Convolutional feature maps	91.25%	91.52%	91.87%	x	91.34%
Emotional State Prediction From Online Handwriting and Signature Biometrics 2.5	The CIU Handwritten DB	KNN, JRIP, Random Forest	11 features 2.3	92.23%	x	x	x	x
Mood State Detection in Handwritten Tasks Using PCA–mFCBF and Automated Machine Learning 2.5	EMOTHAW	AutoML	30 features [24] (PCA, mFCBF)	100%	x	x	x	x
Emotion Detection from Handwritten Text using Agglomerative Clustering 2.5	personal	Agglomerative Hierarchical Clustering	Binary image 512x512	75%	x	x	x	x
Unveiling Emotions through Handwriting: A Data Analysis Approach 2.5	EMOTHAW	Random forest	4 features 2.4	86.01%	x	x	x	x
Identifying Dominant Emotional State Using Handwriting and Drawing Samples by Fusing Features 2.5	EMOTHAW	BiLSTM	MFCC	81.54%	x	x	x	x

Chapter 3

Proposed Methodology

In this chapter, we will present the general approach that we will follow to implement an automated system for emotion recognition from handwriting using machine learning. We will describe the main steps and components of our system, such as data analysis and preprocessing, feature extraction and selection, machine learning models and evaluation metrics.

3.1 Proposed Databases

The availability of comprehensive datasets for automated emotion detection from handwriting is limited, as discussed in Section 2.2.4. Therefore, we selected the EMOTHAW database [21], which is widely used in this field, to train our model. This database contains samples of 7 handwriting tasks performed by 129 subjects, as shown in Table 3.1.

The distribution of emotion levels in the EMOTHAW dataset can be seen in Figure 3.1. The database measured the emotional state of the subjects using the DASS test, as explained in Section 2.2.1.

To verify the validity and interchangeability of this emotion measuring technique with the emotion inducing technique, we will also apply our model to the CIU Handwritten database, which evaluates the written samples based on happy, sad, and stress levels.

The CIU Handwritten database consisted of 134 subjects who completed 4 handwriting tasks, as described in Section 3.2. In addition, the subjects wrote 15 signatures after each task. The subjects rated their emotional state on a scale from 1 (lowest) to 10 (highest) after each task. We concentrated on task 4, where the subjects were exposed to a stressful situation. We aimed to investigate whether our model, which was trained to detect depression, anxiety, and **stress**, could also identify **stress** in these samples.

Table 3.1: Tasks performed by subjects in the EMOTHAW dataset.

T1	Copy of a two-pentagon drawing
T2	Copy of a house drawing
T3	Writing of four Italian words in capital letters (*BIODEGRADABILE*, *FLIPSTRIM*, *SMINUZZAVANO* (to crumble), *CHIUNQUE* (anyone))
T4	Loops with left hand
T5	Loops with right hand
T6	Clock drawing test
T7	Writing of the following phonetically complete Italian sentence in cursive letters: *I pazzi chiedono fiori viola, acqua da bere, tempo per sognare* (Crazy people are seeking for purple flowers, drinking water and dreaming time)

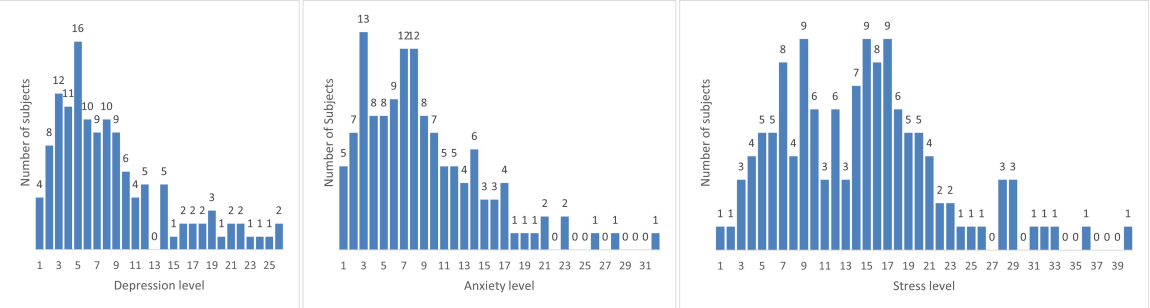

Figure 3.1: The scatter of the emotion levels of subjects in the EMOTHAW database.

Table 3.2: Tasks performed by subjects in the CIU handwritten dataset.

T1	Copy a predefined text in a neutral emotion state: *The communication method: subroutine call or method invocation will not exit until the next invoked computation has been terminated. Asynchronous message passing, by contrast, can result in a response arriving a significant time after the request message has been sent through the net.*
T2	Watch media conveying a positive/happy message and write some text with your own cues.
T3	Watch media conveying a negative/sad message and write some text with your own cues.
T4	Copy a predefined text under 10*s* time limit to convey stressful environment: *All questions asked by five watch experts amazed the judge*

3.2 Proposed Data Preprocessing, Feature Extraction and Feature Selection

As mentioned in Section 2.3, preprocessing and feature extraction are a way how to turn raw data into a workable dataset. Because of the EMOTHAW and the CIU handwritten databases captured their data with a similar INTUOS WACOM tablet, the data has the same content and similar structure. In both cases we have to work with the online data type 2.2.2.

3.2.1 Dataset modifications

To use the same algorithm for extracting features, we want our datasets to have the same structure. As we will discuss later in Section 3.2.2, we want our data to have the following structure. Each task is captured in a separate SVC file, where the first line represents the number of captured datapoints and the rest of the file are lines representing the states of the tablet in each capturing time. The tablet state is represented as 7 values, which are the x coordinate of the pen tip, the y coordinate of the pen tip, the time stamp, the pen status (1 - touching / 0 - not touching the tablet), azimuth, altitude and pressure. The capture should start with the first contact of the pen on the paper. An example of desired data structure can be seen in Figure 3.2. We will develop a converter script capable of converting the CIU handwritten database into this file structure. This simplifies the next step of extracting features from dataset.

```
14058
7290 23750 528 1 1540 500 166
7290 23764 535 1 1540 500 235
7290 23776 543 1 1540 500 297
7290 23786 550 1 1540 500 341
7290 23793 558 1 1540 500 363
```

Figure 3.2: The example of the SVC data structure ready for feature extraction.

3.2.2 Library for extracting features

In the domain of feature extraction, an open-source Python library was provided by doc. Ing. Jiří Mekyska, Ph.D. [14]. This library, installable via PyPi, offers a user-friendly and contemporary approach for extracting a diverse array of handwriting features. It primarily focuses on kinematic, dynamic, spatial, and temporal analyses of online handwriting and drawing. The foundation of this library is the Handwriting Sample package [23], which facilitates straightforward class-based manipulation of online handwriting data. This library was designed to process the EMOTHAW database, which means there is no need to modify that database. To ensure comprehensive information retrieval from our dataset, we advocate for the utilization of this library to extract the maximal range of possible features. The libraries have been slightly modified; therefore, the revised versions are included in the submission folder until the original BDALab libraries are updated.

Acquirable features

The library in question possesses the capability to extract an extensive suite of features, subsequently presenting them through various statistical measures such as the mean, median, maximum, minimum, interquartile range, and the slope of the linear regression, among others. This process culminates in a total of 525 distinct feature values. Our objective is to harness the full potential of this library by extracting each feature. Subsequently, we will apply feature selection methodologies to discern which features are most informative for the purpose of classification. An exhaustive list of features extractable from our databases is delineated below.

1. Kinematic features
    - Velocity - The speed of the writing movement.

- Acceleration - The rate of change of velocity of the writing movement.
- Jerk - The rate of change of acceleration of the writing movement.

2. Dynamic features

   - Azimuth - The angle of the pen relative to the horizontal axis of the paper (see Figure 3.3).
   - Tilt - The angle of the pen relative to the paper (see Figure 3.3).
   - Pressure - The amount of force exerted by the writing instrument on the paper.

3. Spatial features

   - Stroke length - The distance covered by pen on the paper in one stroke.
   - Stroke height - The vertical distance between the highest and lowest points of one stroke.
   - Stroke width - The horizontal distance of one stroke from left to right.
   - Writing length - The total distance covered by the pen on the paper in the whole writing sample.
   - Writing height - The vertical distance between the highest and lowest points of the whole writing sample (see Figure 3.4).
   - Writing width - The horizontal distance of the whole writing sample from left to right (see Figure 3.4).
   - Number of intra-stroke intersections - The number of times a stroke crosses itself within the same stroke. For example, the letter “e” has one intra-stroke intersection (see Figure 3.4).
   - Relative number of intra-stroke intersections - The number of intra-stroke intersections divided by the time duration of the stroke.
   - Total number of intra-stroke intersections - The sum of intra-stroke intersections for all strokes in the writing sample.
   - Relative total number of intra-stroke intersections - Total number of intra-stroke intersections divided by the duration of the whole writing sample.
   - Number of inter-stroke intersections - The number of times a stroke crosses another stroke in the writing sample (see Figure 3.4).
   - Relative number of inter-stroke intersections - The number of inter-stroke intersections divided by the duration of the whole writing sample.
   - Vertical peaks indices - The indices of the points where the pen reaches the maximum height in a loop.
   - Vertical valleys indices - The indices of the points where the pen reaches the minimum height in a loop.
   - Vertical peaks values - The height values of the points where the pen reaches the maximum height in a loop.
   - Vertical valleys values - The height value of the points where the pen reaches the minimum height in a loop.
   - Vertical peaks velocity - The values of the velocity at the vertical peaks in loops.

- Vertical valleys velocity - The values of the velocity at the vertical valleys in loops.
- Vertical peaks distance - The distance covered by the pen from the previous vertical peak to the current one.
- Vertical valleys distance - The distance covered by the pen from the previous vertical valley to the current one.
- Vertical peaks duration - The time elapsed from the previous vertical peak to the current one.
- Vertical valleys duration - The time elapsed from the previous vertical valley to the current one.

4. Temporal features

   - Stroke duration - An array of durations of each on-surface or in-air stroke.
   - Ratio of stroke durations (on-surface/in-air strokes) - An array of ratios of each on-surface stroke duration divided by the upcoming in-air stroke duration.
   - Writing duration - The total time spent on-surface or in-air of the whole writing sample.
   - Writing duration overall - The time elapsed from the start to the end of the whole writing sample.
   - Ratio of writing durations (on-surface/in-air writing) - Writing duration on-surface divided by writing duration in-air.
   - Number of interruptions - The number of times the pen changed form on-surface to in-air and vice versa in the whole writing sample.
   - Number of interruptions relative - The number of interruptions divided by the writing duration overall.

5. Composite features

   - Writing tempo - The ratio of the writing length to the writing duration.
   - Writing stops - An array of durations of in-air strokes.
   - Number of changes in x profile - The number of times the sign of the first derivative of the x coordinate changes in the writing sample.
   - Number of changes in y profile - The number of times the sign of the first derivative of the y coordinate changes in the writing sample.
   - Number of changes in azimuth - The number of times the sign of the first derivative of the azimuth angle changes in the writing sample.
   - Number of changes in tilt - The number of times the sign of the first derivative of the tilt angle changes in the writing sample.
   - Number of changes in pressure - The number of times the sign of the first derivative of the pressure changes in the writing sample.
   - Number of changes in velocity profile - The number of times the sign of the first derivative of the velocity changes in the writing sample.
   - Relative number of changes in x profile - The number of changes in x profile divided by the writing duration.

- Relative number of changes in y profile - The number of changes in y profile divided by the writing duration.
- Relative number of changes in azimuth - The number of changes in azimuth divided by the writing duration.
- Relative number of changes in tilt - The number of changes in tilt divided by the writing duration.
- Relative number of changes in pressure - The number of changes in pressure divided by the writing duration.
- Relative number of changes in velocity profile - The number of changes in velocity profile divided by the writing duration.

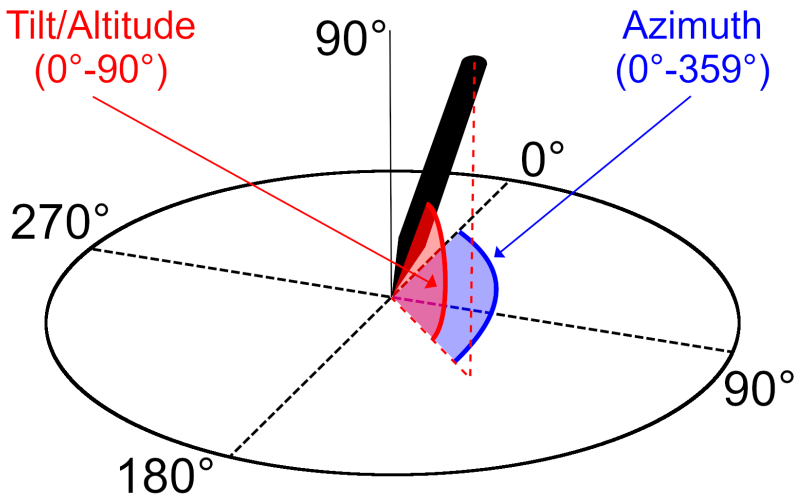

Figure 3.3: Depiction of Azimuth and Tilt of the pen.

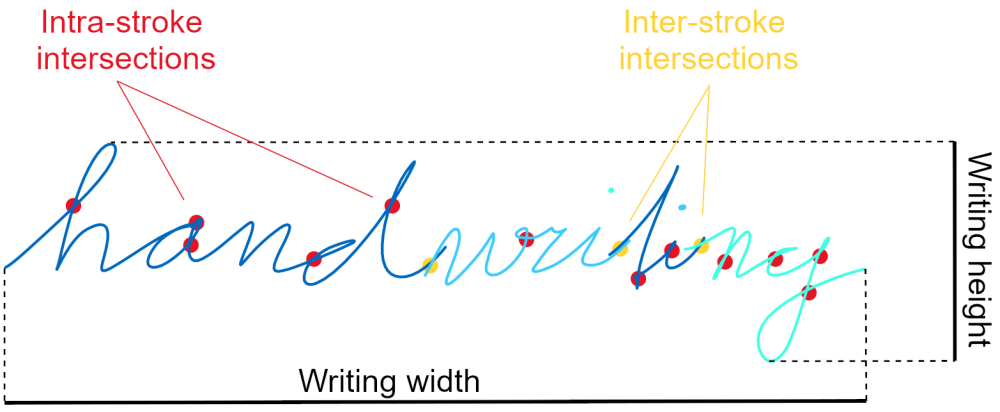

Figure 3.4: Depiction of Writing width, Writing height, Intra-stroke intersections and Inter-stroke intersections.

3.2.3 Normalization of Features

Normalization is an indispensable process that follows the extensive feature extraction detailed in Section 3.2.2. It addresses the challenge of feature heterogeneity, where each feature possesses its unique scale and measurement units. Without normalization, classification algorithms could become biased towards features with larger numeric ranges. To counter this, we implement a normalization procedure that aligns all feature values to a uniform scale, typically between 0 and 1, ensuring equitable contribution to the classification results.

For a range of options we implement four principal normalization techniques, each with its mathematical formulation and purpose:

Z-score Normalization

The Z-score normalization centers the features around the mean with a unit standard deviation, enabling comparison across different scales. It is mathematically represented as:

$$x_{\text{z-score}} = \frac{x - \mu}{\sigma} \tag{3.1}$$

where x is the original feature value, μ is the mean, and σ is the standard deviation of the features within the normalization group.

Min-Max Normalization

Min-Max normalization rescales the feature values to a fixed range of [0, 1], maintaining the original distribution of the data. The formula is given by:

$$x_{\text{minmax}} = \frac{x - \min}{\max - \min} \tag{3.2}$$

where *min* and *max* are the minimum and maximum values of the features, respectively.

Interquartile Range (IQR) Normalization

The IQR normalization is robust to outliers, focusing on the central 50% of the data. It is expressed as:

$$x_{\text{iqr}} = \frac{x - Q1}{Q3 - Q1} \tag{3.3}$$

where $Q1$ and $Q3$ represent the first and third quartiles, respectively.

Logarithmic Normalization

Logarithmic normalization is particularly effective for data with heavy-tailed distributions. The computation is as follows:

$$x_{\log} = \log(x - \min + 1) \tag{3.4}$$

This ensures positivity of the input x by adjusting for the minimum value and avoiding the logarithm of zero.

3.2.4 Feature Selection via ANOVA

Feature selection is pivotal in enhancing model interpretability and reducing overfitting. To identify the most informative features, we will employ the Analysis of Variance (ANOVA) statistical method. ANOVA is instrumental in determining the features that exhibit the most significant mean differences across various groups, which is indicative of their discriminative power. By focusing on features with the highest F-values, we can isolate those that contribute most substantially to class separation.

3.2.5 Dimensionality Reduction Techniques

Dimensionality reduction is a fundamental step in the preprocessing of high-dimensional data, serving to simplify the dataset while retaining its most informative aspects. By reducing the number of features, we can alleviate issues related to computational complexity, storage requirements, and potential overfitting. Two prominent techniques for dimensionality reduction are Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), each with its unique approach to preserving essential data characteristics [17, 2]. Both PCA and LDA will be implemented to assess their individual and combined effectiveness in reducing dimensions while preserving the most salient features for classification.

PCA

Principal Component Analysis (PCA) is an unsupervised method renowned for its effectiveness in capturing the maximum variance within a dataset. By transforming the original features into a new orthogonal basis, PCA identifies principal components that are linear combinations of the initial features. The first principal component accounts for the largest variance, followed by subsequent components explaining progressively smaller portions of the variance. This technique enables the reduction of data dimensionality by retaining only those principal components that contribute significantly to the total variance. Beyond dimensionality reduction, PCA is also utilized for purposes such as data visualization, noise reduction, and feature extraction.

LDA

Linear Discriminant Analysis (LDA) is a supervised dimensionality reduction technique that excels in classification tasks. It operates under the assumption that each class's data is distributed according to a multivariate Gaussian model with a shared covariance matrix but distinct mean vectors. LDA aims to discover linear discriminants that optimize class separability by maximizing between-class variance and minimizing within-class variance. The transformation of original features into linear discriminants, which are adept at class differentiation, allows for a reduction in data dimensionality. LDA retains only those discriminants with the highest discriminatory power and is also applied in data visualization, feature extraction, and regularization.

3.3 Proposed Classification Method

After selecting significant features and applying PCA or LDA for dimensionality reduction, we need to choose a suitable classification method for our problem. There are many possible models to choose from, such as Artificial Neural Networks, Support Vector Machines (SVM),

Decision Trees, K-Nearest Neighbor (KNN), etc. However, it is difficult to know which model will perform the best on our data, as each model has its own strengths and limitations. Moreover, using a single model may not be sufficient to capture the diversity and complexity of the data. Therefore, we decided to use an ensemble learning approach, which combines the predictions of multiple models to obtain a better predictive performance than any of the individual models alone.

Ensemble learning is a general meta approach to machine learning that seeks to reduce the errors of a single model by exploiting the diversity and complementarity of multiple models. There are three main types of ensemble learning methods: bagging, boosting, and stacking. Each of these methods has a different way of generating and combining the models. We will briefly describe each of these methods[30, 26, 8].

Bagging

Bagging, which stands for bootstrap aggregating, is an ensemble learning method that aims to reduce the variance of a single model by averaging the predictions of multiple models trained on different subsets of the data. The subsets are obtained by sampling the data with replacement, which is called bootstrapping. Each model is trained independently and in parallel on a different subset. The final prediction is obtained by taking the majority vote for classification of the individual predictions. The process of bagging ensemble can be seen in Figure 3.5.

Bagging is useful for reducing the overfitting of models that have high variance, such as decision trees. By averaging the predictions of multiple models, bagging reduces the effect of noise and outliers in the data. However, bagging may not be very effective for reducing the bias of models that have low variance, such as linear models. Moreover, bagging does not take into account the performance or the diversity of the individual models, which may lead to suboptimal results.

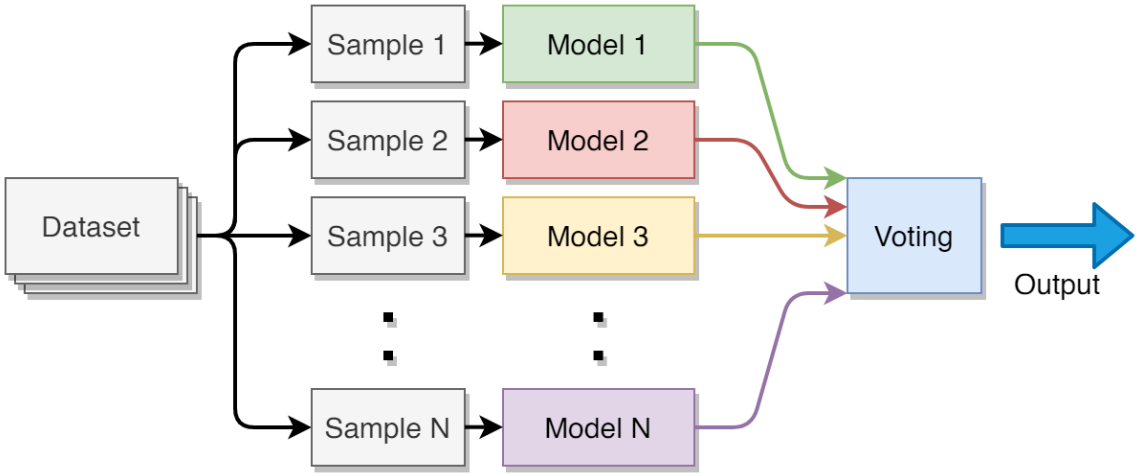

Figure 3.5: Diagram of Bagging ensemble model [8].

Boosting

Boosting is an ensemble learning method that aims to reduce the bias of a single model by sequentially adding models that correct the errors of the previous models. The models are

trained on weighted versions of the data, where the weights are updated based on the errors of the previous models. The final prediction is obtained by taking a weighted average of the individual predictions, where the weights reflect the performance of the models. The process of boosting ensemble can be seen in Figure 3.6.

Boosting is useful for improving the accuracy of models that have high bias, such as weak learners. A weak learner is a model that performs slightly better than random guessing. By sequentially adding models that focus on the hard-to-classify instances, boosting increases the complexity and the expressiveness of the ensemble. However, boosting may also increase the variance of the ensemble, as it is more prone to overfitting the data. Moreover, boosting requires careful tuning of the learning rate and the number of models, which may affect the performance of the ensemble.

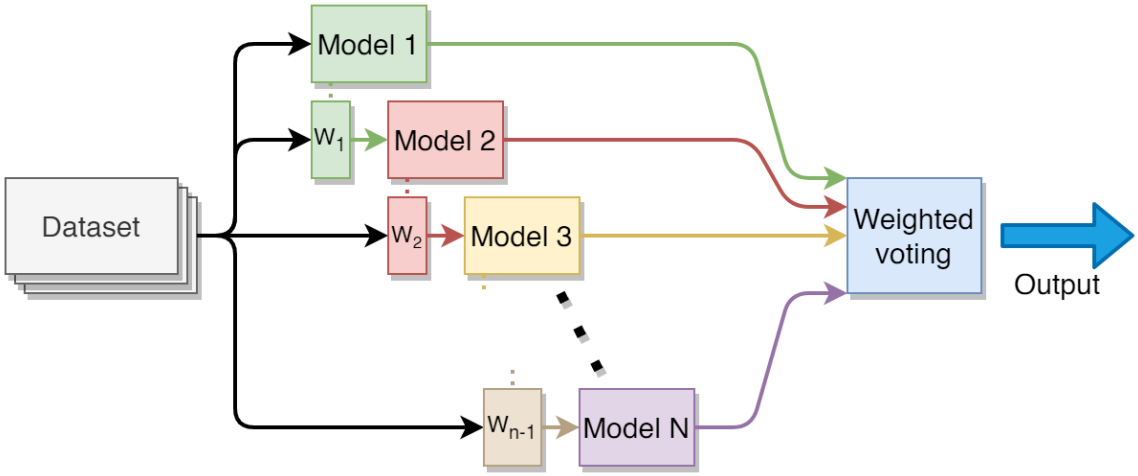

Figure 3.6: Diagram of Boosting ensemble model [8].

Stacking

Stacking, which stands for stacked generalization, is an ensemble learning method that aims to combine the predictions of multiple models by learning a meta-model that maps the individual predictions to the final prediction. The models are trained on different subsets of the data, which are obtained by splitting the data into multiple folds. The predictions of the models on the validation folds are used as the input for the meta-model, which is trained on the target variable. The final prediction is obtained by applying the meta-model on the predictions of the models on the test fold. The process of stacking ensemble can be seen in Figure 3.7.

Stacking is useful for combining the predictions of models that have different strengths and weaknesses, such as heterogeneous models. A heterogeneous model is a model that uses a different learning algorithm or a different representation of the data. By learning a meta-model that optimizes the combination of the individual predictions, stacking can achieve better performance than any of the individual models alone. However, stacking may also introduce more complexity and computational cost, as it requires training multiple models and a meta-model. Moreover, stacking may suffer from overfitting the meta-model, especially if the number of models is large.

In the realm of ensemble stacking, a diverse range of models are employed as base learners. These models, each with their unique strengths and capabilities, contribute to the

robustness and predictive power of the ensemble. The following models are among the most commonly used in stacking ensembles due to their proven effectiveness across a variety of tasks and datasets [9]:

- **Artificial neural network (ANN)**: A nonlinear model that can learn complex patterns and features from the data using multiple layers of neurons.
- **Support vector machine (SVM)**: A linear model that can learn a maximum margin hyperplane that separates the data into different classes.
- **Decision tree (DT)**: A non-parametric model that can learn a hierarchical structure of rules that split the data based on the values of the features.
- **K-nearest neighbor (KNN)**: A lazy model that can classify a new instance based on the majority vote of its k closest neighbors in the data.

These models have been selected for their complementary advantages and potential to enhance ensemble accuracy and robustness. For instance, while the ANN is adept at learning complex nonlinear relationships, it may be prone to overfitting and require extensive training time. Conversely, the SVM excels at identifying linear decision boundaries but may struggle with nonlinearities and memory consumption. The DT offers simple, interpretable rules but can be subject to high variance and instability. The KNN is effective at capturing local patterns but may be adversely affected by noise and outliers.

In addition to these models, our methodology will incorporate a variety of other models to create a versatile and comprehensive ensemble. This approach allows us to leverage the distinct advantages of a broader set of models, enhancing the ensemble's overall performance. By evaluating a wide spectrum of models, we can select the most effective ones after training, ensuring that our ensemble is not only robust but also highly adaptable to various data characteristics.

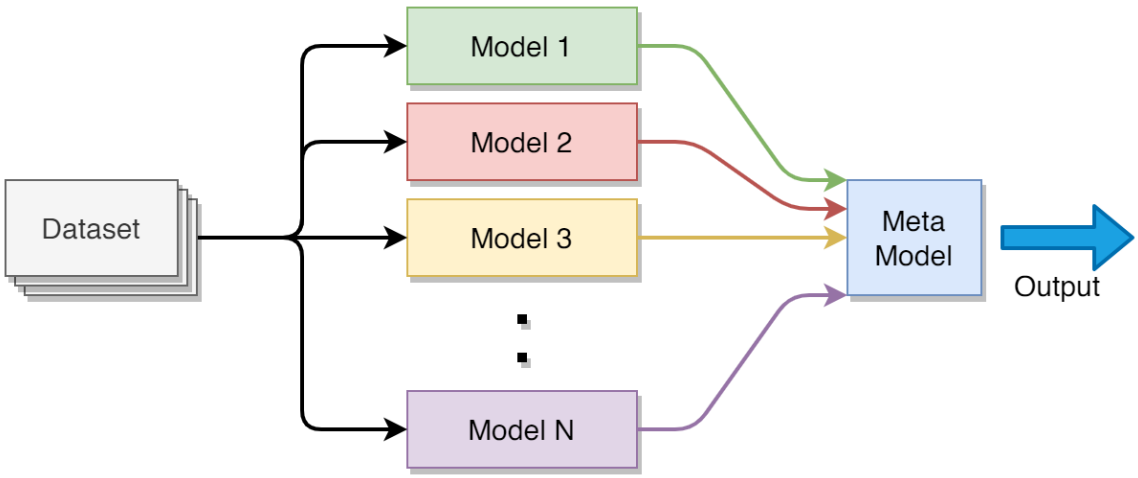

Figure 3.7: Diagram of Stacking ensemble model [8].

3.4 Model Training and Evaluation

In light of the dataset's limited scope, the initial proposition was to employ the entirety of the dataset for the training phase, complemented by a 10-fold cross-validation method for

classifier validation. This methodology aimed to capitalize on every available data point for both training and validation, thereby optimizing data utilization. The underlying premise was to augment the ensemble's learning capacity by exposing it to the full spectrum of data variations.

However, as delineated in the experimental findings section of this thesis 5.4, this approach inadvertently led to data leakage and subsequent model overfitting dew to the LDA and PCA preprocessing. In response to these challenges, a revised methodology was proposed, entailing the segregation of the dataset into distinct training and validation subsets. The validation subset, comprising 26 files, will be reserved exclusively for classifier evaluation and remain unseen during the training phase, which will involve 123 files.

The evaluation of each classifier will be conducted using the validation subset, with key performance metrics such as accuracy, precision, recall, and F1-score meticulously recorded to ascertain the efficacy of the models.

3.5 Quality metrics

As we discussed in the dataset summary in Section 2.6, the biggest gap in the state of the art is the lack of all the metrics necessary for a good model. This thesis will concentrate on improving the quality of the confusion matrix and the derived metrics of Precision, Sensitivity, Specificity and F-1 score.

Confusion matrix

A confusion matrix is a table that summarizes the performance of a classifier by comparing the actual and predicted labels of the test data. For a binary classification problem, the confusion matrix has four cells: true positive (TP), false positive (FP), true negative (TN) and false negative (FN). These cells represent the number of instances that belong to each combination of actual and predicted classes. A confusion matrix can provide more insight into the strengths and weaknesses of a classifier than a single accuracy score [19, 7].

Precision

Precision is a metric that quantifies the number of correct positive predictions made by the classifier. It is calculated as the ratio of TP to the sum of TP and FP 3.5. Precision indicates how reliable the classifier is when it predicts a positive label. A high precision means that the classifier rarely makes false positive errors, but it does not necessarily mean that it covers all the positive instances in the data.

Sensitivity

Sensitivity (or recall) is a metric that quantifies the number of correct positive predictions made out of all the positive instances in the data. It is calculated as the ratio of TP to the sum of TP and FN 3.6. Sensitivity indicates how complete the classifier is when it predicts a positive label. A high sensitivity means that the classifier captures most of the positive instances in the data, but it does not necessarily mean that it avoids false positive errors.

Specificity

Specificity is a metric that quantifies the number of correct negative predictions made by the classifier. It is calculated as the ratio of TN to the sum of TN and FP 3.7. Specificity indicates how well the classifier can identify true negatives and avoid false positives. A high specificity means that the classifier rarely makes false positive errors, but it does not necessarily mean that it covers all the negative instances in the data.

F-1 score

F-1 score is a metric that combines both precision and sensitivity. It is calculated as the harmonic mean of precision and recall 3.8, which gives more weight to low values. F-1 score reaches its best value at 1 and worst value at 0. F-1 score balances the trade-off between precision and recall, and it is useful when the data is imbalanced or when both types of errors are equally important.

$$Precision = \frac{TP}{TP + FP} \tag{3.5}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.6}$$

$$Specificity = \frac{TN}{TN + FP} \tag{3.7}$$

$$F\text{-}1\ score = \frac{2 \times Precision \times Sensitivity}{Precision + Sensitivity} \tag{3.8}$$

3.6 Chapter summary

In this chapter, we outlined a structured pipeline for classifier training, beginning with feature extraction from the EMOTHAW database using a designated library. The extracted features are then normalized using one of four techniques: Z-score, Min-Max, IQR, or logarithmic scaling, to identify the most effective normalization method.

ANOVA is applied to the normalized data to rank features based on their statistical significance for specific tasks and emotions. Features with a p-value less than 0.05 are selected for classification. To address potential issues of insufficient feature selection, a parameter was introduced to ensure a minimum number of features are retained, prioritizing those ranked highest by ANOVA.

Dimensionality reduction is considered with PCA and LDA, as well as the possibility of combining these methods, to evaluate their impact on efficiency. The classifier's structure is built using an ensemble approach, examining various configurations and techniques to determine the optimal model.

To prevent overfitting, the initial proposal included using the entire dataset for 10-fold cross-validation. However, to avoid data leakage from PCA/LDA, we propose a split into training and validation sets, with the validation set reserved exclusively for model performance evaluation.

A visual representation of the pipeline, incorporating each step in Figure 3.8, concludes this summary, providing a clear overview of the proposed methodology.

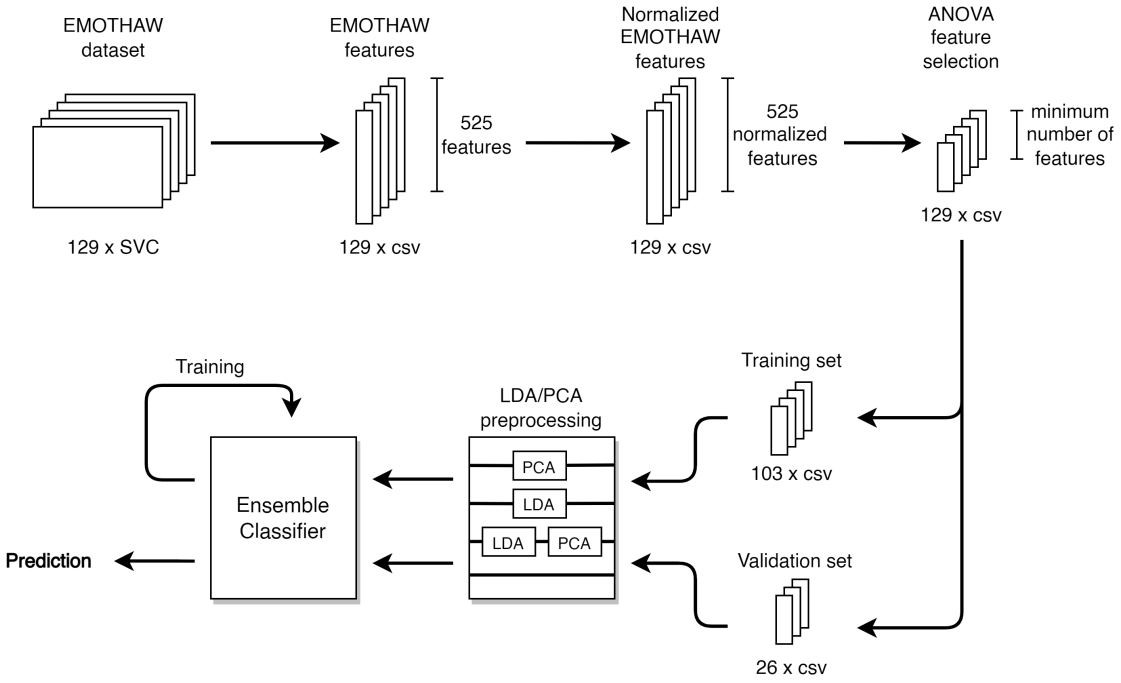

Figure 3.8: The proposed pipline for training a classification model for one emotion.

Chapter 4

Implementation

4.1 Dataset Preprocessing

In the Section on datasets 2.2.4, we concluded that the most comprehensive and reliable dataset for our purposes was the EMOTHAW database [21]. This dataset served as the cornerstone for our primary training and testing regimen. Additionally, we wanted to use The CIU Handwritten database for experimenting with our trained classification model. Both datasets were procured using an INTUOS WACOM digitizing tablet, capturing identical types of online data 2.2.2. Both datasets needed specific adjustments to ensure compatibility with our feature extraction library introduced in Section 3.2.2, as delineated in Section 3.2.1.

- **EMOTHAW database**: The SVC files utilized by the EMOTHAW database were already in the required format for our feature extraction. However, a visual inspection of the data revealed that it was rotated 90° to the right, likely due to the portrait orientation of the task paper, resulting in the tablet being turned. To correct this, we employed a Python script, *convert_emothaw.py*, to transpose the x and y axes. This operation inadvertently mirrored the data horizontally, which we rectified by inverting the sign of the new x values and translating them to positive coordinates.
- **CIU Handwritten database**: This dataset recorded an excess of data points and employed a disparate saving format. We utilized a Python script, *convert_ciu.py*, to discard superfluous data and reformatted the pertinent data to meet our specifications. Moreover, we cropped each data point from the beginning to the initial pen-paper contact, aligning it with the format applied to the EMOTHAW database, thereby ensuring maximal consistency between the datasets.

4.2 Feature Extraction

In Section 2.6, we underscored the pivotal role of feature extraction in emotion classification. Indeed, this step is arguably the most critical in our entire process. For our research, we deliberately pursued an expansive approach by extracting as many features as possible. This strategic decision aimed to enrich the potential of our classification model. The specifics of our feature selection process will be elaborated upon in subsequent sections.

The actual feature extraction process is facilitated by two Python scripts called: *feature_extraction_emothaw.py* and *feature_extraction_ciu.py*. These scripts share similar

functionalities but are fine-tuned for extracting features from each dataset separately. Now, let's delve into the intricacies of the features themselves.

The features that can be extracted form a SVC data file can be seen in Section 3.2.2. Features such as Writing Width or Writing Duration are represented by a single value. However, features that vary over time, like Pressure, can be extracted with multiple statistical representations such as mean, standard deviation, median, first quartile, third quartile, etc. Moreover, certain features, like Velocity or Acceleration, are inherently directional. For these, we can extract information from the entire 2D sample or isolate it to just the x-axis or y-axis. Additionally, some features can differentiate between the pen being in contact with the paper or when the pen is in the air.

We consider each possible combination as one separate feature. By employing this method, we generate a robust set of 525 unique feature values for each data file, providing our classification model with a rich dataset to analyze the nuances of emotional expression through writing. These feature values are saved in CSV format for future use.

4.3 Feature Normalization

The normalization of features, as conceptualized in Section 3.2.3, was implemented through a Python script, *normalization.py*. This script was designed with flexibility in mind, allowing for the application of normalization techniques based on specified arguments during execution.

The script accepts arguments that determine the normalization technique to be applied. It is capable of processing all four normalization methods in a single run, generating four separate CSV files for each method. This feature enhances the efficiency of the normalization process, providing a comprehensive set of normalized data for comparative analysis.

An additional functionality of the script is its ability to recognize pre-existing normalized files. If a file corresponding to a particular normalization technique already exists, the script intelligently skips the normalization calculation for that file, thereby saving computational resources and time.

Upon execution, the script processes each CSV file, which contains features extracted from individual tasks performed by subjects. For each feature, the script aggregates the same feature values from other CSV files within the 'normalization group'—a collection of files from the same task category—to maintain contextual integrity.

The script stores essential parameters such as the mean, standard deviation, minimum, maximum, and quartiles for each feature. These parameters are crucial for normalizing future unseen data.

The output of the script includes new CSV files with normalized values, which are then used for training and testing our classification models.

4.4 Feature Selection

The dimensionality of our feature space, comprising 525 distinct features, presents a challenge when employing basic machine learning algorithms as opposed to deep learning methods. High-dimensional data can lead to model overfitting, where the model learns the noise in the training data rather than the actual signal. Moreover, not all features contribute equally to the prediction task; some may be redundant or irrelevant, potentially obscuring the significant patterns that are crucial for accurate emotion classification.

To address this, we have implemented the Analysis of Variance (ANOVA), a statistical method that assesses the impact of one or more factors by comparing the means of different groups and determining if the observed differences are significant [20, 16]. ANOVA helps in identifying features that have a strong statistical relationship with the emotional states we aim to classify.

Our script, *anova.py*, automates this process by calculating the p-values for each feature. The p-value measures the probability of observing the given data, or something more extreme, under the null hypothesis, which in this context is the assumption that the feature has no effect on the emotion classification. A low p-value suggests that the feature is significant and should be retained for model training.

The classification of emotions is based on the Depression Anxiety Stress Scales test (DASS test), with thresholds defined for binary and trinary classifications as shown in Table 4.1, adapted from the work of Nolazco-Flores et al. [24]. We compute separate p-values for each emotion, considering the DASS score thresholds to determine the relevance of features for each emotional state.

Employing a significance level of 5%, denoted by the threshold $\alpha = 0.05$, we identify features with p-values less than this threshold as statistically significant. These features are likely to have a meaningful contribution to the classification of specific emotions and are selected for inclusion in the model.

Despite the effectiveness of ANOVA in feature selection, subsequent experiments in Section 5.4 indicated that ANOVA might select a suboptimal number of features. To address this, we introduced a new parameter, *minimum_features*, ensuring that at least this minimum number of features is selected. Consequently, our script *anova.py* not only performs the ANOVA test but also saves the p-values of each feature into a CSV file, sorted by the p-value. This allows us to choose a broader set of features beyond those strictly under the $\alpha = 0.05$ threshold, prioritizing the most statistically significant ones for model training.

Binary	**Trinary**	**Depression**	**Anxiety**	**Stress**
Normal	Normal	0–9	0–7	0–14
Above normal	Mild	10–13	8–9	15–18
	Above mild	14+	10+	19+

Table 4.1: DASS score thresholds for binary and ternary classification of emotional states from the work of Nolazco-Flores et al. [24].

4.5 Dimensionality Reduction

The application of ANOVA in feature selection does not ensure a predetermined number of statistically significant features. To accommodate the variability in feature quantity, dimensionality reduction techniques such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are considered within our *ensemble.py* script, which will be further elaborated on in Section 4.6.

The dimensionality reduction step is configurable; it is not a compulsory component of the preprocessing pipeline but is available for use depending on the specific requirements of the dataset and the classifier. The script is parameterized to allow the selection of PCA,

LDA, both PCA and LDA in sequence, or neither, yielding four distinct preprocessing pathways.

For PCA, we have set a threshold to retain 80% of the variance. This level of reduction is applied when the PCA option is selected, aiming to reduce the feature space effectively while maintaining a substantial proportion of the data's variance.

Similarly, LDA can be applied to the features to maximize the separability between the different classes of emotions. When both PCA and LDA are applied, PCA serves as a preliminary step to reduce dimensionality before LDA further refines the feature set to enhance class discrimination.

The script ensures that any transformation matrices generated during PCA or LDA are saved for future use, allowing the classifier to apply the same transformations to new data.

If a matrix already exists, the script is designed to bypass the saving process, thus avoiding unnecessary duplication.

4.6 Ensemble Classification

In accordance with the ensemble learning approach delineated in Section 3.3 and the pipeline visualized in the comprehensive diagram in Figure 3.8, the script *ensemble.py* was developed. This script is adept at executing various ensemble methods, with stacking as the primary technique, while also providing support for bagging and boosting as alternative strategies. The choice of ensemble method is governed by the `-et` argument, allowing for flexibility in the model building process.

4.6.1 Ensemble Method Selection

The choice of ensemble method—stacking, bagging, or boosting—is dictated by the user through the `-et` argument. For stacking, the script evaluates a set of 25 models, selecting the best performers based on individual classification efficacy to serve as inner models. The number of models included in the ensemble is specified by the `-nm` argument. For bagging and boosting, the `-fm` argument determines the base model, with the `-nm` argument indicating the number of estimators.

4.6.2 Meta-Classifier and Base models

In stacking, the meta-classifier, chosen via the `-fm` argument, integrates the predictions from the inner models. This meta-classifier is crucial, as it consolidates diverse predictions into a final decision. For bagging and boosting, where a single model is used, the script adapts the `-fm` model as the base model and utilizes the `-nm` argument to define the ensemble's structure.

4.6.3 Normalization and Feature Selection

The script offers all four normalization methods, z-score, Min-Max, IQR, and Log normalization as shown in Section 4.3, selectable through corresponding arguments. Feature selection is conducted via ANOVA, with a conventional threshold of $p < 0.05$ for statistical significance. If the `-mf` argument is set, the script ensures that at least the specified minimum number of features is selected, even if it requires including features with higher p-values to meet this criterion as explained in Section 4.4.

4.6.4 Data Splitting and Transformation

The dataset is split into training and validation sets, maintaining the same ratio of classified groups as the full dataset. The training set is then subjected to PCA and/or LDA transformations if indicated by the `-pca` and `-lda` arguments, respectively. These transformations are applied solely based on the training data to prevent data leakage, with the derived matrices subsequently used to transform the validation set.

4.6.5 Ensemble Training and Evaluation

Upon preparing the data, the script proceeds to construct the ensemble classifier according to the selected method and trains it using the training set. The validation set is then employed to assess the classifier's performance on unseen data, with metrics such as accuracy, precision, recall, and F1-score being recorded. The results, along with the utilized arguments, are systematically documented in a CSV files for ease of analysis. Additionally, the trained classifier is serialized in pickle (.pkl) format, facilitating future deployment.

4.6.6 Result Documentation and Classifier Deployment

The performance of various classifiers and argument configurations is manually reviewed using the CSV file records. The pickled classifier is intended for standalone use, with a dedicated script that will process new data by extracting features, normalizing, selecting, and classifying, as per the trained model's specifications.

4.7 Deployment

The deployment of our models is orchestrated by the script *classifier.py*, which serves as the operational core for applying the trained classifiers to new data. This process is streamlined through the use of two dedicated directories: *data* for incoming data and *models* for the serialized classifiers.

4.7.1 Model and Data Association

Each model is named with a comprehensive set of arguments that encapsulate its training context, such as emotion, task, normalization group, thresholding method, preprocessing steps, ensemble technique, meta/base model, number of models, and minimum number of features. This naming convention ensures that each piece of data in the *data* folder is classified by the appropriate model. Data filenames include task identifiers (e.g., *task_5*) to link with corresponding task-trained classifiers. In the absence of a task identifier, data is considered generic and is processed by all classifiers. Additionally, DASS scores embedded within the data filenames enable the script to determine the ground truth for performance evaluation.

4.7.2 Feature Extraction

The script iterates over each classifier, identifying the 'linked' data based on task relevance. It consults the *anova_features.csv* file, which lists all features in descending order of their statistical significance as determined by ANOVA. This ordered list acts as a master key,

unlocking the necessary features required by each classifier. Features required for classification are extracted, and processed, ensuring that the input vector for classification is complete and robust against missing or erroneous values.

4.7.3 Normalization

The normalization process begins with the retrieval of task-specific parameters from the *normalization_values.csv* file. This file is a repository of critical statistics—mean, standard deviation, minimum, maximum, first and third quartile values—calculated for each feature across different tasks. These statistics are pivotal for the subsequent z-score, Min-Max, IQR or Log normalization methods applied to the data.

4.7.4 Data Transformation and Classification

For data requiring PCA or LDA transformations, the script fetches the corresponding matrices from *lda_matrices* or *pca_matrices* directories. These matrices are meticulously tailored to each model's specifications, considering factors such as emotion, task, normalization group, thresholding, and minimum number of features. Once the data is normalized and transformed, the script employs the classifier to predict the group classification.

4.7.5 Performance Metrics and Manual Experimentation

Performance metrics such as accuracy, precision, sensitivity, specificity, F1-score, and AUC are calculated for each classifier based on the 'linked' data's classification results. This granular approach allows for an in-depth assessment of each model's efficacy. The deployment phase also facilitates manual experimentation, providing a sandbox environment for exploratory analysis and further refinement of models.

Chapter 5

Experiments

This chapter presents the comprehensive suite of experiments conducted to evaluate the performance of various classification models. It also includes two failed experiments 5.4 performed before our main experiments shoed in this sectrion. These failed experiments showed weak pints in our proposed methodology that had to be rectified before continuing in experimenting.

5.1 Argument Search

As previously discussed in Section 4.6, the script *ensemble.py* is designed to generate a classification model based on the supplied arguments. The selection of these arguments is critical, as they significantly influence the performance of the classifier, yielding results that range from suboptimal to highly precise models.

Given the multitude of arguments, each with its own set of possible values, the task of identifying the ideal combination is non-trivial. The total number of unique classifiers that can be constructed from the argument permutations is substantial to illustrate:

- **Task**: 7 distinct tasks for classification.
- **Emotion**: 3 different emotions to classify: depression, anxiety, stress.
- **Thresholds**: 2 grouping methods for data: binary or trinary classification.
- **Ensemble Type**: 3 ensemble techniques: bagging, boosting, and stacking.
- **Meta/Base Model**: 25 model types for the final ensemble.
- **Number of Models**: 25 settings for the quantity of models in the ensemble.
- **Preprocessing**: 4 preprocessing options; LDA, PCA, both, or none.
- **Minimum number of features**: ± 10 setting values for feature sellection.

This results in $7 \times 3 \times 2 \times 3 \times 25 \times 25 \times 4 \times 10 = 3,150,000$ different argument configurations and, consequently, 3,150,000 potential classifiers.

To expedite the search process, we employed a technique of argument freezing, where all but one argument are held constant to determine the most effective value for the unfrozen argument. This approach is iteratively applied to each argument in succession.

A dedicated script, *argument_experiment.py*, was developed to automate this process. It invokes *ensemble.py* as a subprocess with varying arguments, systematically iterating through all possible combinations. The outcomes of these experiments, including accuracy, precision, recall, and F1-score, are meticulously recorded in a *arguments_results.csv* files. This data repository enables us to discern and eliminate inefficient arguments from the search space.

5.1.1 Adaptation to failed experiments

In light of the accuracy bias identified in Failed Experiment 2 5.4, we introduced a new argument, *minimum_features*, to our classification model pipeline. This experiment was designed to determine the optimal minimum number of features for feature selection required for effective classification.

Experiment 1: Minimum number of features

Classifiers were trained on each of the seven tasks with both binary and trinary thresholds, employing all three ensemble techniques—bagging, boosting, and stacking—and the three meta/base models: Logistic Regression, K-Nearest Neighbors (KNN), and Random Forest. The number of inner models was varied among 5, 10, 15, or 20, and preprocessing was conducted using all combinations.

As we can see in Figure 5.1, the range of *minimum_features* was set to include 0, 20, 30, 40, 60, 80, 100, 125, 150, and 200. Notably, a peak in accuracy was observed at 30 features, indicating an optimal balance between too few and too many features. This peak, while not pronounced, was aligned with the average number of features selected prior to the introduction of the *minimum_features* argument.

By ensuring a baseline number of features, we have mitigated the risk of underfitting due to an insufficient number of features while also avoiding the complexity introduced by an excessive feature set. The result is an enhanced overall effectiveness of our models, as evidenced by the improved accuracy metrics.

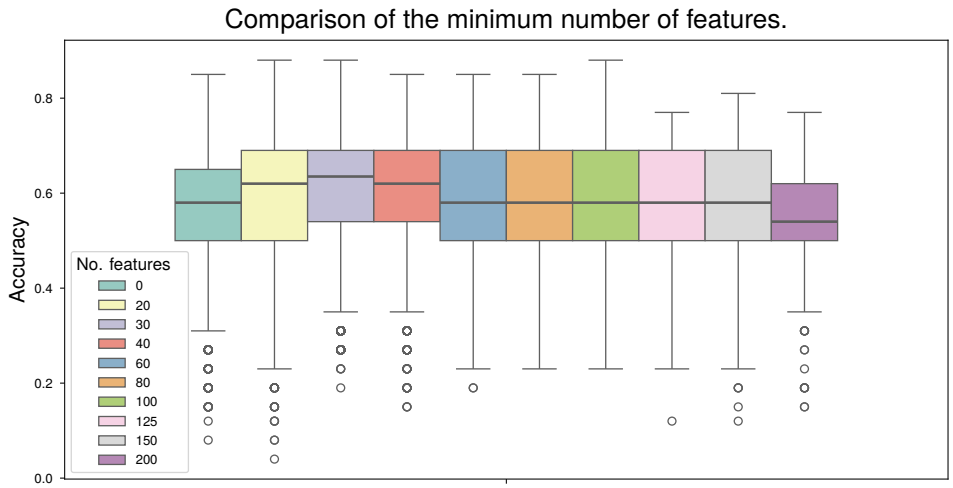

Figure 5.1: The graph showcases the comparative accuracy of classifiers utilizing a different minimum number of features used for feature selection step. We can see that 30 features is our optimal choice for further argument search.

5.1.2 Binary classifier

The next phase of experimentation, conducted via the *argument_experiment.py* script, was concentrated on binary classifiers. This phase is dedicated to refining the selection of arguments, which are instrumental in the construction of robust classification models. Our initial focus was directed towards the preprocessing techniques, as they play a crucial role in shaping the input data for optimal classifier performance.

Experiment 2: Data preprocessing

The initial experiment encompassed all seven tasks, simplifying the meta/base model selection to three established algorithms: Logistic Regression, K-Nearest Neighbors, and Random Forest. We incorporated all three ensemble techniques, varying the number of inner models between 5, 10, 15, and 20.

The execution of *ensemble.py* with these parameters, alongside diverse preprocessing methods, aimed to assess the influence of LDA and PCA transformations, both individually and combined, as well as their absence, on model accuracy. The outcomes, as visualized in Figure 5.2, underscore the enhanced performance of models preprocessed with PCA and the PCA-LDA pipeline, leading to their adoption as preferred preprocessing methods.

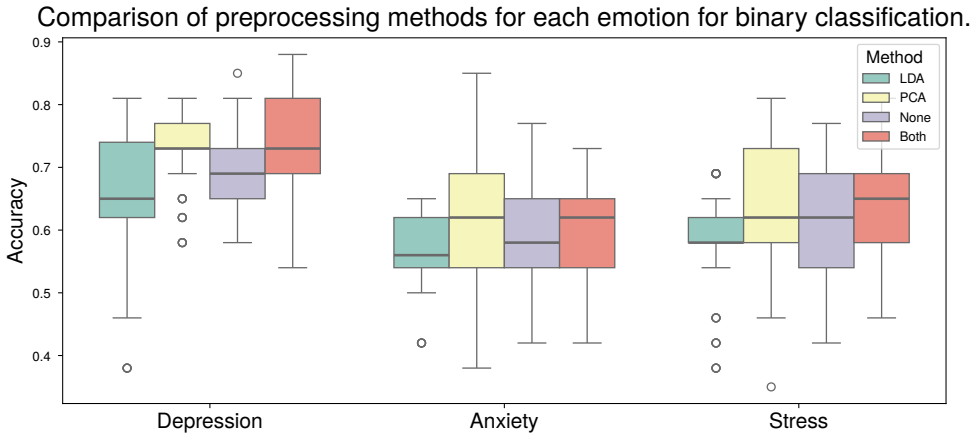

Figure 5.2: The graph compares the accuracy of binary classification models under different preprocessing methods, highlighting the superior performance achieved through PCA and the PCA-LDA pipeline.

Experiment 3: Task-Specific Input Data

Investigating the premise that certain tasks may possess greater discriminative power for emotion classification, we evaluated classifiers using PCA and PCA-LDA preprocessing across all three ensemble techniques. The ensemble configurations, consistent with Experiment 2 5.1.2, included the same three potential meta/base models and varied the number of inner models from 5 to 20. Data was segmented based on input tasks to determine their specific impact on classifier efficacy.

Figure 5.3 encapsulates the insights from this analysis, revealing how task-specific data differentially affects emotion classification success. Notably, task 1 data significantly enhances depression classification, with task 6 as a secondary contributor. For anxiety, tasks 2 and 7 prove most effective, while tasks 2 and 3 lead in stress classification.

The findings from this experiment suggest that within the EMOTHAW database [21], not all tasks carry equal weight in the context of emotion detection. For instance, high-quality binary classifiers for stress and anxiety can be developed using data solely from Task 2, whereas Task 1 is paramount for depression.

In light of these findings, future experiments will leverage the top-performing task inputs for each emotion to streamline our argument search space.

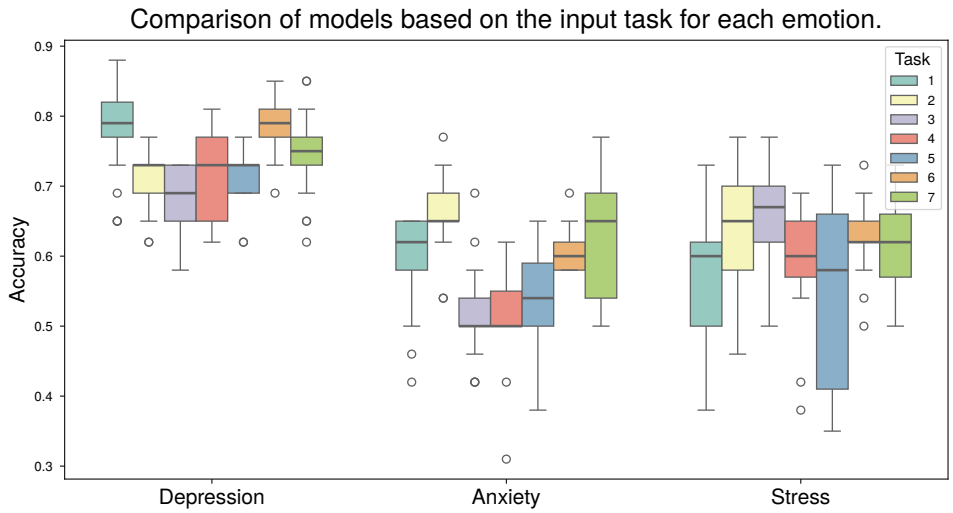

Figure 5.3: This graph presents the accuracy of classifiers for each emotion, showcasing the distinct influence of task-specific input data on model performance.

Experiment 4: Number of Inner Models

In this experiment, we investigated the optimal number of inner models for binary classifiers, with a distinct focus on each ensemble learning technique. We utilized tasks 1 and 6 for depression, tasks 2 and 7 for anxiety, and tasks 2 and 3 for stress applying all 25 possible meta/base models and preprocessing with PCA and PCA-LDA. We observe the impact of 5, 10, 15, 20, 25, 40, 50, 60, and 100 inner models.

- **Bagging Variant** - Figure 5.4 demonstrates that, within the bagging ensemble, an optimal classification accuracy is achieved with approximately 10 inner models.
- **Boosting Variant** - When employing the boosting ensemble, the performance peaked with approximately 50 inner models, as shown in Figure 5.5, suggesting a preference for a higher number of inner models within this technique.
- **Stacking Variant** - With stacking, the analysis revealed that a minimal number of inner models is sufficient, with 5 inner models being adequate for a high-quality classifier, as evidenced by Figure 5.6.

These findings will inform the fine-tuning of our model complexity, allowing us to optimize the classifiers for performance and computational efficiency across different ensemble techniques.

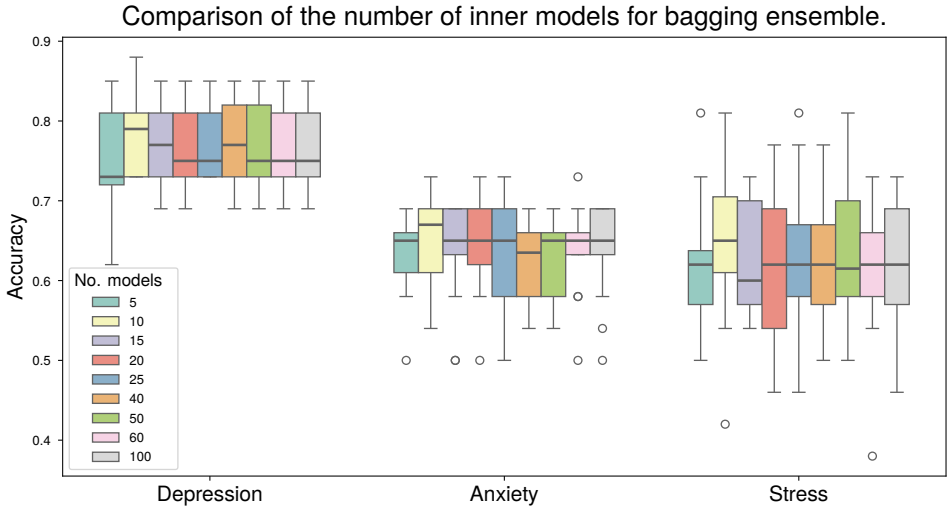

Figure 5.4: Accuracy trends of binary classifiers with varying inner model counts using bagging, highlighting 10 as the optimal number.

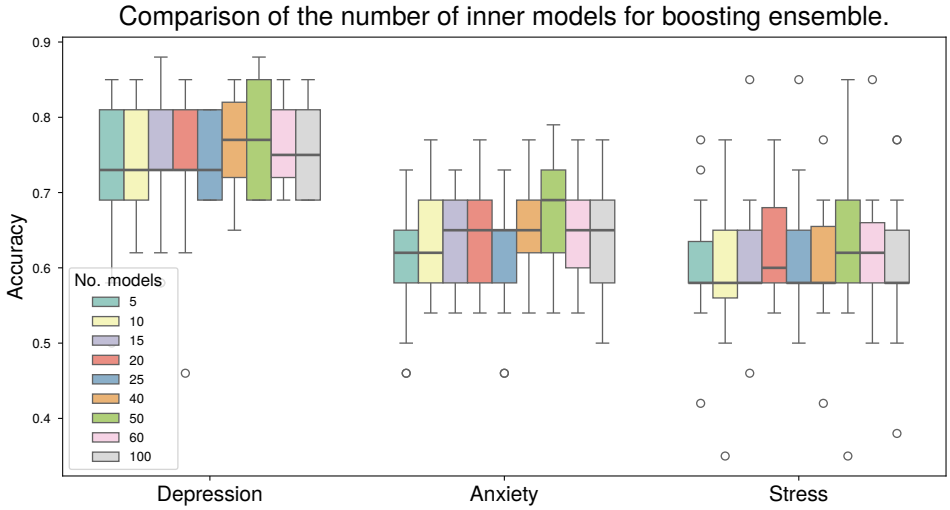

Figure 5.5: Accuracy trends of binary classifiers with varying inner model counts using boosting, highlighting 50 as the optimal number.

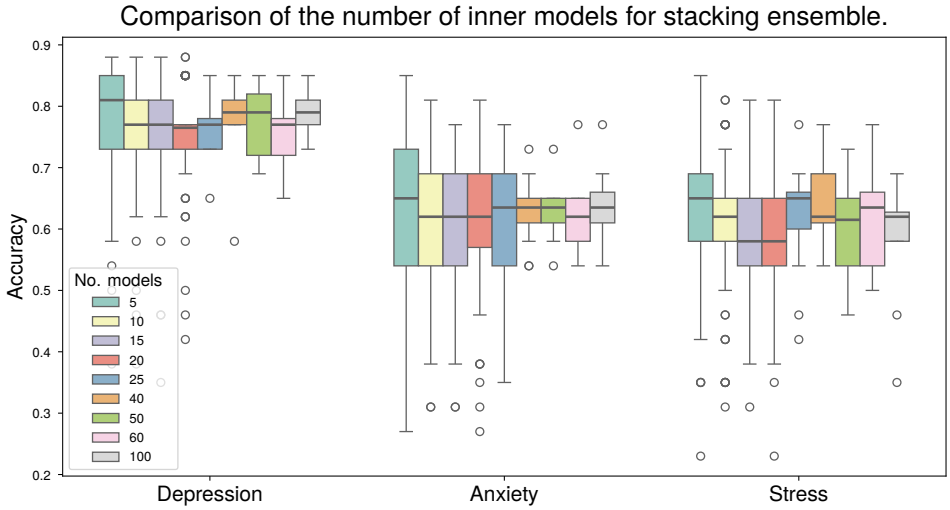

Figure 5.6: Accuracy trends of binary classifiers with varying inner model counts using stacking, highlighting 5 as the optimal number.

Experiment 5: Ensemble Technique

The next experiment in binary classification sought to evaluate the performance differences among various ensemble techniques. For this purpose, we trained classifiers using tasks 1 and 6 for depression, tasks 2 and 7 for anxiety, and tasks 2 and 3 for stress. We applied PCA and PCA-LDA preprocessing and assessed all 25 meta/base models with their corresponding optimal number of inner models.

The results, as visualized in Figure 5.7, did not conclusively favor any single ensemble technique over the others. We will consider all ensemble techniques when searching for optimal classifiers.

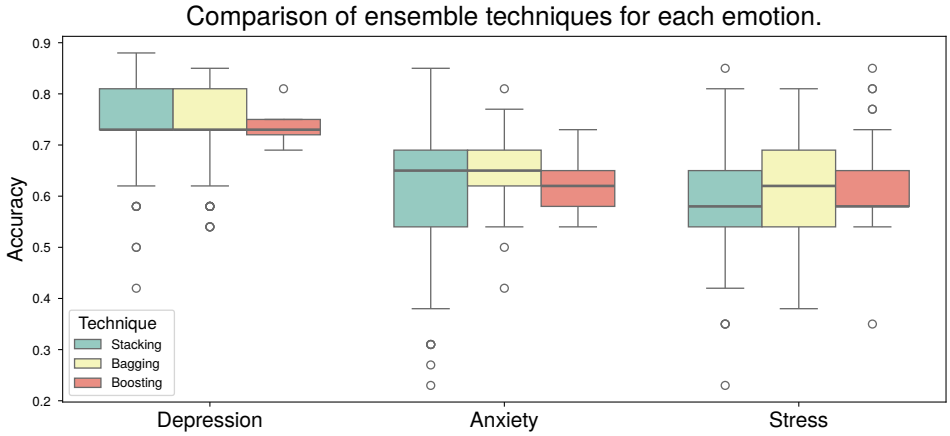

Figure 5.7: The graph compares the accuracy of binary classifiers using different ensemble techniques, showing negligible differences in their performance.

Experiment 6: Meta/Base Model Evaluation

In our sixth experiment, we sought to determine the most effective meta/base model for our classifiers. Each of the 25 models was evaluated for depression, anxiety, and stress classification, using the same arguments as in the previous experiments.

The bar graph, in Figure 5.8, summarized the performance of each model. While some models exhibited marginally better performance, the differences were not substantial enough to draw a definitive conclusion.

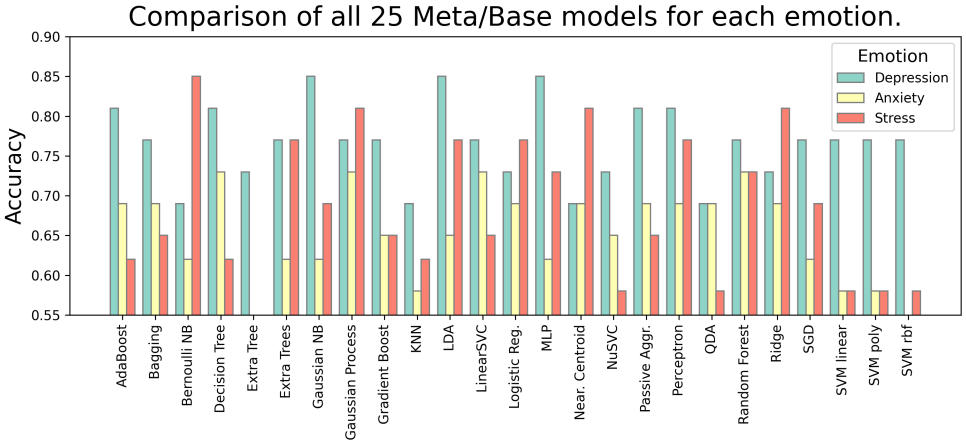

Figure 5.8: The bar graph illustrates the performance of each meta/base model for classifying emotions, with no single model demonstrating clear superiority.

5.1.3 Trinary Classifier

The exploration of trinary classifiers is a natural extension of our binary classification work. This section presents a series of experiments designed to adapt the binary classification framework to the nuances of trinary classification.

Experiment 7: Data Preprocessing

Replicating the binary classification's Experiment 2 5.1.2, we assessed the impact of different preprocessing techniques on trinary classification. Although the differences between preprocessing methods were less pronounced as shown in Figure 5.9, the PCA and PCA-LDA pipeline still proved to be the most effective. Therefore, we will continue to employ these methods in our trinary classification experiments.

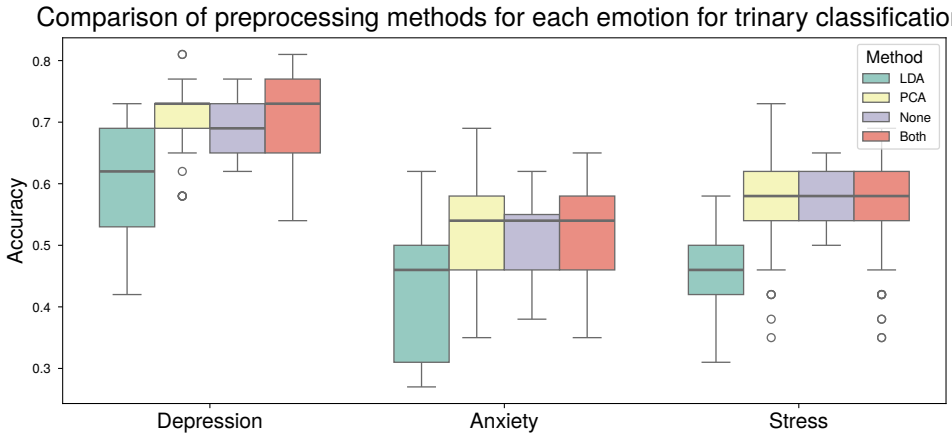

Figure 5.9: The graph compares the accuracy of trinary classification models under different preprocessing methods, highlighting the superior performance achieved through PCA and the PCA-LDA pipeline.

Experiment 8: Task-Specific Input Data

Following the approach of Experiment 3 5.1.2, we conducted a similar analysis for trinary classification. The results in Figure 5.10 indicated that specific tasks correlate with specific emotions. For depression, tasks 1 and 7 were most indicative, with task 6 also showing relevance. Anxiety was best classified using tasks 2 and 6, with task 7 closely behind. For stress, tasks 2 and 7 were chosen. Henceforth, these tasks will be used as inputs for their respective emotions in trinary classification.

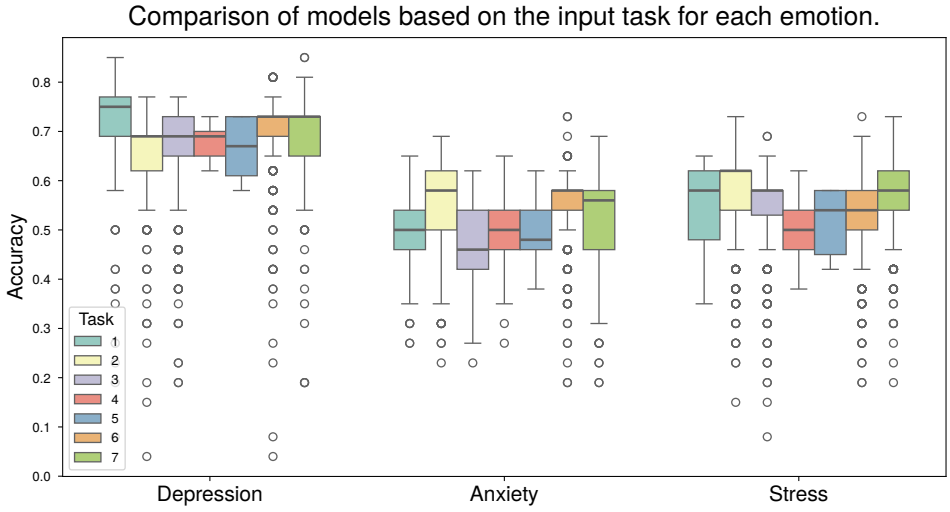

Figure 5.10: This graph presents the accuracy of classifiers for each emotion, showcasing the distinct influence of task-specific input data on model performance in trinary classification.

Experiment 9: Number of Inner Models

In this experiment, we explored the optimal number of inner models for each ensemble method in trinary classification, akin to the binary classification's Experiment 4 5.1.2.

- **Bagging Variant** - Lower numbers of inner models, such as 5, 10, and 15, demonstrated slightly better performance in trinary classification as shown in Figure 5.11.
- **Boosting Variant** - Results from Figure 5.12 were consistent across the board, with a notable bump at 50 inner models. This, coupled with insights from the binary classification, led us to select 50 as the optimal count for boosting.
- **Stacking Variant** - A trend of decreasing accuracy with an increasing number of models was observed, with the best results around 15 inner models as shown in Figure 5.13.

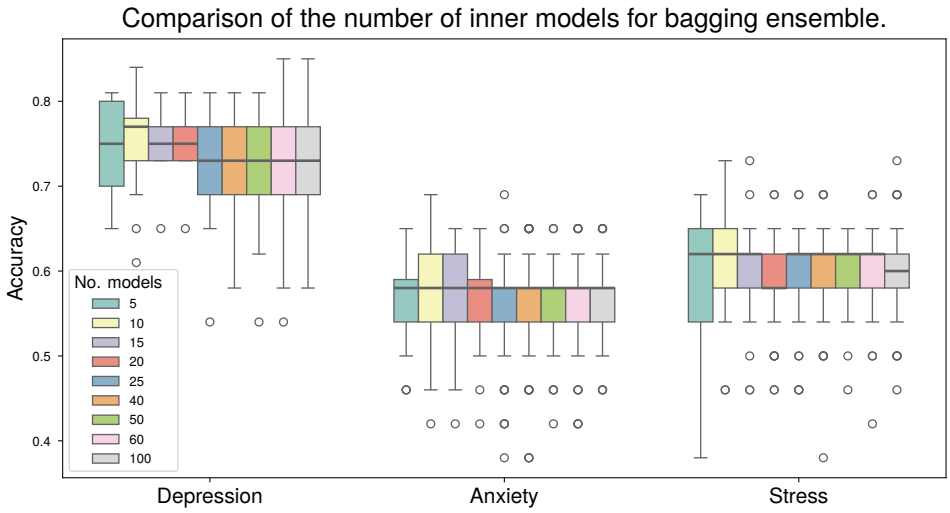

Figure 5.11: Accuracy trends of trinary classifiers with varying inner model counts using bagging, highlighting the optimal lower numbers.

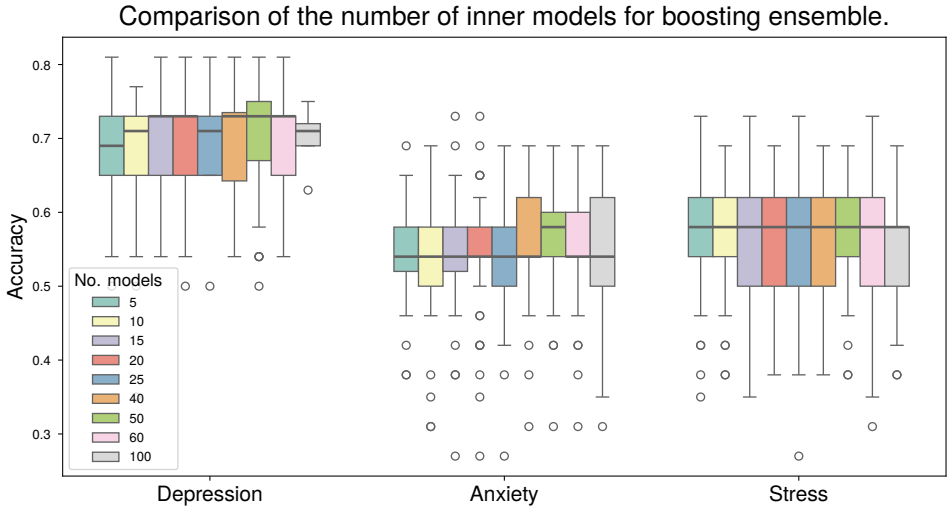

Figure 5.12: Accuracy trends of trinary classifiers with varying inner model counts using boosting, indicating 50 as the optimal number.

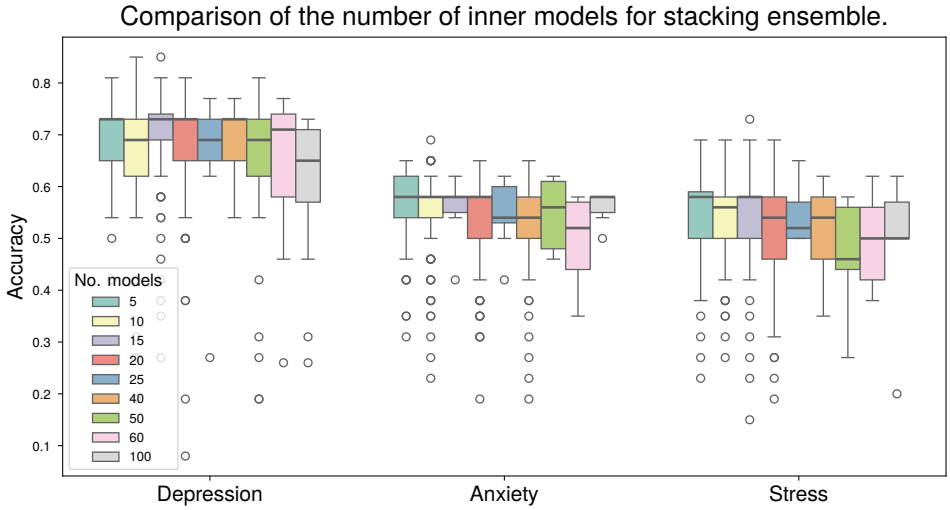

Figure 5.13: Accuracy trends of trinary classifiers with varying inner model counts using stacking, showing a peak at 15 models.

Experiment 10: Ensemble Technique

This experiment aimed to compare the performance of different ensemble techniques in trinary classification. While all techniques yielded similar results as shown in Figure 5.14, stacking exhibited a broader range of accuracies, producing both superior and inferior classifiers compared to the other techniques. Among bagging and boosting, bagging demonstrated a slight edge in performance.

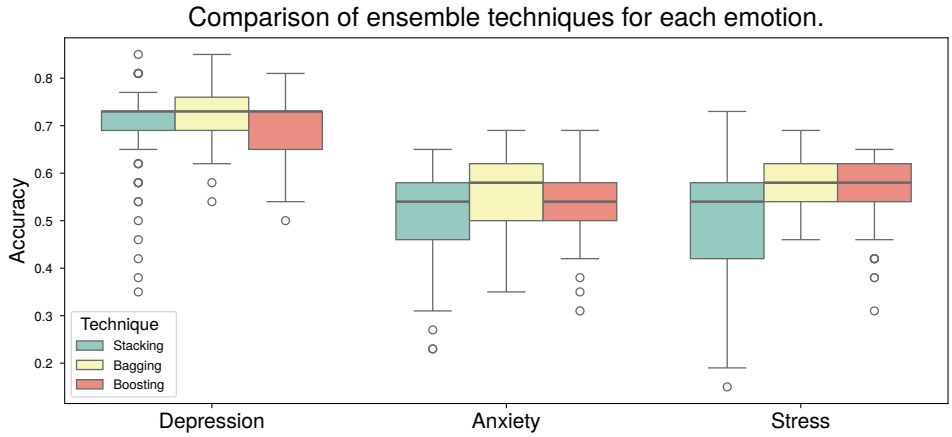

Figure 5.14: The graph compares the accuracy of trinary classifiers using different ensemble techniques, indicating a broader range of results for stacking and a slight advantage for bagging over boosting.

Experiment 11: Meta/Base Model Evaluation

In our final experiment, we evaluated the effectiveness of different meta/base models for trinary classification. Similar to Experiment 6 5.1.2, each of the 25 models was assessed. The results highlighted that some models performed significantly worse than others in the context of trinary classification. We can see the different models performance in Figure 5.15.

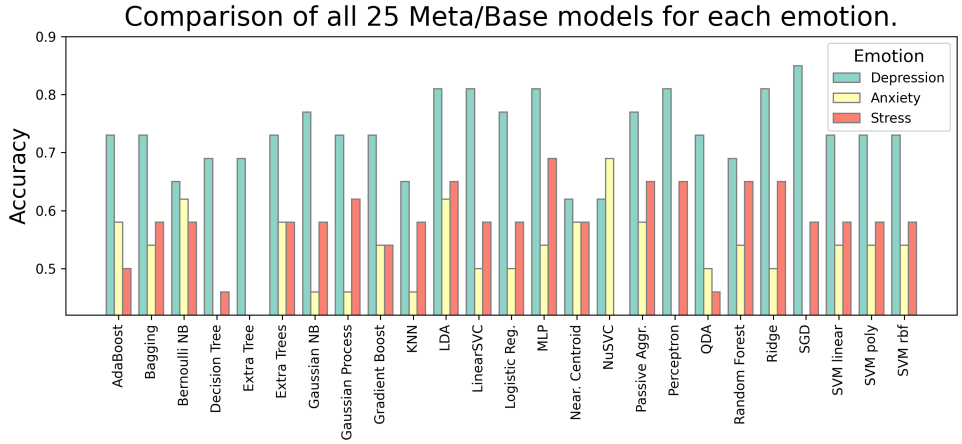

Figure 5.15: The bar graph illustrates the performance of each meta/base model for classifying emotions in trinary classification, with some models showing notably poor performance.

5.2 Feature Importance Analysis

This section delves into the significance of individual features in emotion recognition from handwriting. Utilizing the file *anova_features.csv* from the 'model' folder, we conduct a series of experiments to identify key features that consistently demonstrate importance

across various conditions. The analysis focuses on the average position of features in the ANOVA sorted list and their selection frequency, providing insights crucial for researchers and practitioners in the field.

Experiment 12: Top Features by Average ANOVA Position

In this experiment, we examine the positions of features in the ANOVA sorted list to determine their significance. With 7 different tasks, 3 emotions and the choice of binary or trinary data grouping, we have 42 different rankings of our 525 extracted features. We employed a script *feature_experiment.py* that records the position of each feature in all 42 rankings. The average position is then presented in a sorted list. The top 10 features with the best average position out of 525, are as follows:

- ratio_of_writing_durations: 147.0
- writing_tempo_on_paper: 159.0
- acceleration_on_paper_y_slope_of_linear_regression: 165.4
- vertical_valleys_duration_iqr: 169.4
- acceleration_in_air_y_median: 175.8
- jerk_on_paper_y_cv_parametric: 176.1
- stroke_length_on_paper_iqr: 182.0
- relative_number_of_changes_in_velocity_profile: 183.5
- jerk_on_paper_xy_cv_parametric: 184.4
- vertical_peaks_values_slope_of_linear_regression: 189.5

Experiment 13: Top Features by Selection Frequency

In this experiment, we continue with the feature importance analysis. However, unlike Experiment 12, we score the features in alignment with our feature selection method explained in Section 3.6. Here, we consider not only the position of each feature but also whether it was selected for classification or not. We take in account the argument *minimum_features* that modifies the number of selected features and set it to 30 in this experiment, as was suggested in Experiment 1 5.1.1. The top 10 most frequently selected features are:

- acceleration_on_paper_y_slope_of_linear_regression: 14/42 (33.3%)
- ratio_of_writing_durations: 12/42 (28.6%)
- stroke_length_on_paper_mean: 11/42 (26.2%)
- stroke_length_on_paper_percentile_95: 10/42 (23.8%)
- jerk_on_paper_y_cv_parametric: 10/42 (23.8%)
- writing_tempo_on_paper: 10/42 (23.8%)
- stroke_length_on_paper_median: 10/42 (23.8%)

- stroke_duration_in_air_slope_of_linear_regression: 10/42 (23.8%)
- stroke_width_in_air_slope_of_linear_regression: 10/42 (23.8%)
- relative_number_of_changes_in_velocity_profile: 10/42 (23.8%)

Experiment 14: Top Merged Base Features by Selection Frequency

In this experiment we posed the same question as in Experiment 13 5.2 with the difference being that we merged our features back to their base form. In Section 3.2.2 we mentioned that some features can create multiple values based on the orientation, pen status, various statistical measures. Throughout our search for the best classifier we treated each value as its own feature, but in this experiment, we merge them back together and explore how frequently they were selected. The top 10 most frequently selected base features are:

- ratio_of_writing_durations: 12/42 (28.6%)
- relative_number_of_changes_in_velocity_profile: 10/42 (23.8%)
- relative_number_of_changes_in_y_profile: 6/42 (14.3%)
- relative_number_of_intra_stroke_intersections: 63/462 (13.6%)
- vertical_valleys_velocity: 63/462 (13.6%)
- writing_tempo: 11/84 (13.1%)
- relative_number_of_changes_in_azimuth: 5/42 (11.9%)
- relative_total_number_of_intra_stroke_intersections: 5/42 (11.9%)
- writing_stops: 54/462 (11.7%)
- ratio_of_stroke_durations: 49/462 (10.6%)

5.3 Comparative Analysis of Induced and Measured Stress

This section evaluates the performance of our best-performing binary classifier for stress detection when applied to a different dataset, the CIU Handwritten database. The CIU database differs from the EMOTHAW dataset in that the stress emotion was induced by imposing a time limit on the task as explained in Section 2.2.1, whereas in EMOTHAW, stress was measured based on the DASS test scores as explained in Section 2.2.1.

Experiment 15: Evaluation of CIU Database

The experiment employed our binary classifier, which had previously reported an accuracy of 88%, to classify samples from task 4 of the CIU database, which was the timed task as can be seen in Section 3.1. Participants in the CIU study were asked to write a sentence and self-evaluate their level of stress while writing on a scale from 0-10. For our analysis, we considered values greater than 8 as indicative of stress and values less than 8 as not stressed, excluding the value 8 to broaden the gap between groups as suggested by the Article [3].

Upon applying our classifier to the CIU data, the following metrics were observed:

- Accuracy: 0.29
- Precision: 0.16
- Sensitivity : 0.76
- Specificity: 0.19
- F1 Score: 0.27

The classifier demonstrated significant challenges in accurately classifying the CIU dataset, with a marked decrease in performance metrics compared to the EMOTHAW dataset. This suggests that the input data must be highly precise and consistent across datasets for the classifier to maintain its performance. The disparity in tasks between the datasets likely contributed to the reduced effectiveness of the classifier.

5.4 Failed Experiments

This section is dedicated to the experiments that did not yield the expected results or contribute new insights to our primary research outcomes. However, these unsuccessful attempts were far from futile; they played a crucial role in refining our methodology and steering the research towards more fruitful avenues. By examining the shortcomings and missteps, we gained valuable lessons that ultimately guided us to our successful strategies. For those interested in the granular details of these experiments, a comprehensive account is provided in the appendix in Section A.

Failed Experiment 1: Overfitting

The initial set of experiments, aimed at optimizing the argument selection for our classification models, encountered a critical issue of overfitting. This was identified when the models, trained using 10-fold cross-validation on the entire dataset, achieved implausibly perfect classification results.

The overfitting was traced back to the data preprocessing step, where LDA and PCA were applied. The transformation matrices, derived from the entire dataset, inadvertently introduced information about the whole dataset into each fold of the cross-validation. This led to data leakage, where the training folds contained „hidden“ information from the validation folds, thus compromising the integrity of the validation process.

To rectify this, we altered our approach by ensuring that the transformation matrices for LDA and PCA were computed solely based on the training set. The validation set remained untouched during this preprocessing step, thereby preventing any data leakage and ensuring a more robust and genuine evaluation of the model's performance. The new approach is detailed as our main pipeline in previous Sections 3.8. The comprehensive details of this failed experiment are documented in the appendix in Section A.

Failed Experiment 2: Minimum Number of Features

During the argument search from Experiment 5.4, in addition to discovering the overfitting issue, we also observed an accuracy bias for classifiers trained on certain tasks. Closer inspection revealed the issue lay within the feature selection process. We found that ANOVA

does not select the same amount of features for every classifier; it selects only the statistically significant features. This meant that sometimes ANOVA selected as few as just two features, which were insufficient for classification.

To address this, we implemented a new argument called *minimum_features*, which ensures that at least a minimum number of features are selected. ANOVA then selects the best features up to the defined minimum, as detailed in previous Sections 3.6. The full account of this failed experiment is available in the appendix in Section A.

Chapter 6

Results

This chapter presents the culmination of the extensive research and experimentation conducted throughout this study. It is here that we translate the intricate processes and methodologies of our experiments into tangible outcomes. We will showcase the optimal arguments for constructing both binary and trinary classifiers, derived from the rigorous testing of various configurations. Furthermore, this chapter will detail the specific classifiers that emerged as the most effective, along with their respective performances. By dissecting the results of our experiments, we aim to provide a clear and comprehensive understanding of the factors contributing to the success of our classification models. The insights gained from this analysis are not only pivotal for the current study but also serve as a valuable reference for future research in the field.

6.1 Optimal Arguments for Classifier Search

This summary encapsulates the optimal arguments derived from the comprehensive experiments 2-11 conducted for both binary and trinary classifiers. The findings are instrumental in guiding the construction of robust classification models.

6.1.1 Binary Classifier Arguments

- **Preprocessing**: Employ the PCA or PCA-LDA pipeline for data preprocessing.
- **Tasks for Emotions**:
  - Depression: Utilize tasks 1 and 7.
  - Anxiety: Utilize tasks 2 and 7.
  - Stress: Utilize tasks 2 and 3.
- **Number of Inner Models**:
  - Bagging: Approximately 10 inner models.
  - Boosting: Approximately 50 inner models.
  - Stacking: Approximately 5 inner models.
- **Ensemble Techniques**: Implement all ensemble techniques.
- **Meta/Base Models**: Explore all 25 meta/base models to identify the best classifier.

6.1.2 Trinary Classifier Arguments

- **Preprocessing**: Consistently use the PCA or PCA-LDA pipeline.
- **Tasks for Emotions**:
  - Depression: Utilize tasks 1 and 7.
  - Anxiety: Utilize tasks 2 and 6.
  - Stress: Utilize tasks 2 and 7.
- **Number of Inner Models**:
  - Bagging: Around 10 inner models.
  - Boosting: Maintain 50 inner models as optimal.
  - Stacking: Optimal results with approximately 15 inner models.
- **Ensemble Techniques**: Include all techniques in the search for the optimal classifier.
- **Meta/Base Models**: Assess all 25 meta/base models for the most effective trinary classification.

6.2 Performance of Optimal Classifiers

Following the identification of optimal arguments for classifier construction, we evaluated the performance of the best classifiers for each emotion under binary and trinary data separation. This section presents the configurations and performance metrics of the six classifiers that emerged as the most effective.

6.2.1 Classifier Performance Metrics

Table 6.1: Performance Metrics of Best Classifiers

Classifier Configuration	Accuracy	Precision	Recall	F1 Score
Binary classification of Depression	0.89	0.93	0.79	0.73
Binary classification of Anxiety	0.85	0.89	0.83	0.73
Binary classification of Stress	0.85	0.84	0.84	0.73
Trinary classification of Depression	0.85	0.91	0.75	0.73
Trinary classification of Anxiety	0.73	0.83	0.80	0.73
Trinary classification of Stress	0.73	0.70	0.71	0.73

Each classifier's configuration, which includes the preprocessing method, input task, ensemble technique, meta/base model and number of inner models can be seen in the Table 6.2:

Table 6.2: The Configurations of Best Classifiers

Classifier Config.	Task	Preproces. Method	Ensemble Technique	Meta/Base Model	No. Inner Models
Binary D	1	PCA-LDA	Bagging	Random Forest	5
Binary A	2	PCA	Stacking	Random Forest	5
Binary S	2	PCA	Boosting	Bernoulli NB	5
Trinary D	1	PCA-LDA	Bagging	QDA	10
Trinary A	6	PCA-LDA	Boosting	AdaBoost	15
Trinary S	2	PCA	Boosting	Perceptron	25

The performance metrics reveal that while all classifiers perform well, binary classifier for depression classification exhibits the highest accuracy, precision, and F1 score. The consistent F1 score across classifiers suggests a balance between precision and recall, indicating that the classifiers are well-tuned to their respective tasks.

Conclusion: The results demonstrate the effectiveness of the selected configurations in classifying emotions from handwriting data. The use of PCA or PCA-LDA preprocessing methods, in conjunction with various ensemble techniques and meta/base models, has proven to be successful. The number of inner models also plays a crucial role, with different optimal numbers for bagging, boosting, and stacking ensembles. These findings contribute valuable insights into the development of robust classifiers for emotion recognition in handwriting and can guide future research in the field.

6.3 Feature Importance in Emotion Classification

The exploration of feature importance in handwriting analysis, as detailed in Experiments 12, 13, and 14 has yielded significant insights into the characteristics that most influence emotion classification. This section discusses the key features that have consistently emerged as top indicators of emotional states, regardless of whether they are considered individually or as merged base features.

6.3.1 Key Features for Emotion Classification

The comparison between Experiments 13 and 14 reveals a set of features that are particularly relevant for emotion classification:

- **Ratio of Writing Durations**: The ratio of writing duration on-surface to in-air is a critical indicator. It reflects the time spent on the paper versus above it, suggesting that the length of pauses between strokes is a significant factor in emotion classification.
- **The Slope of Linear Regression of Vertical Acceleration on Paper**: The slope of linear regression for acceleration in the y-direction indicates how quickly acceleration changes when moving up and down on the paper, providing insights into the dynamic aspects of writing that relate to emotional states.
- **Stroke Length on Paper (Mean)**: The average length of each stroke offers information about the extent of pen movement before lifting, which can be indicative of the writer's emotional condition.

- **Stroke Length on Paper (Percentile 95)**: Interestingly, not only the average stroke length but also the length of the longest strokes (top 5%) plays a role in distinguishing emotions, pointing to the importance of extremes in writing behavior.

These features, among others listed in the tables of Experiments 12, 13, and 14, highlight the multifaceted nature of handwriting analysis in the context of emotion recognition. The detailed explanation of each feature can be seen in Section 3.2.2.

Conclusion: The consistent selection of certain features across different analytical approaches underscores their potential as robust markers for emotion classification. By understanding these key features, researchers and practitioners can better interpret handwriting data to assess emotional states. The longer list of features and their detailed analysis are available in the appendix for further exploration in Section B.

6.4 Conclusion on the Comparative Analysis of Stress

The comparative analysis conducted in Experiment 15 5.3 provides a critical evaluation of our classifier's performance on the CIU Handwritten database. The stark contrast in results when compared to the EMOTHAW dataset underscores the challenges in generalizing classifiers across datasets with varying conditions for stress induction.

6.4.1 Implications of the Results

The observed metrics indicate a significant decline in the classifier's accuracy and precision, despite high sensitivity. This discrepancy can be attributed to the fundamental differences in how stress was induced and measured in the CIU and EMOTHAW datasets, respectively. The CIU dataset's induced stress, created by a time constraint, may manifest differently in handwriting compared to the measured stress from the DASS test scores in EMOTHAW, which could explain the classifier's reduced specificity.

6.4.2 Future Directions

To enhance the classifier's applicability and reliability across diverse datasets, future research should focus on the following aspects:

- **Consistency in Task Design**: Ensuring that the tasks used for inducing or measuring stress are consistent across datasets to minimize variability in stress representation.
- **Robust Training**: Incorporating a broader range of data that includes both induced and measured stress conditions to train classifiers that can adapt to different stress manifestations.
- **In-Depth Feature Analysis**: Conducting a thorough analysis of the features that contribute to successful stress classification, particularly those that may be sensitive to the method of stress induction.

6.4.3 Potential for Merging Stress Conditions

The potential to merge induced and measured stress conditions into a unified classification framework remains an open question. The findings from this experiment suggest that

achieving this integration requires a classifier trained on tasks that are identical or highly similar in nature. Such a classifier would offer a more convincing comparison and could potentially lead to a robust solution capable of detecting stress across various conditions.

6.5 Task Effectiveness in Emotion Classification

In this section, we delve into the effectiveness of individual tasks in the classification of emotions. Our experimentation has revealed that certain tasks are more conducive to analyzing specific emotions. This overview provides a comparative analysis of task performance for both binary and trinary classifiers, highlighting the tasks that are most indicative of each emotional state.

6.5.1 Binary Classification Task Effectiveness

Table 6.3: Task Effectiveness for Binary Classification

Emotion	**Task Ranking (Best to Least)**
Depression	1, 6, 7, 4, 2, 5, 3
Anxiety	2, 7, 1, 6, 5, 4, 3
Stress	3, 2, 7, 6, 1, 5, 4

For binary classification we can observe in Figure 5.3, tasks 1 and 6 are optimal for depression, with task 1 being the most effective. Anxiety is best classified using tasks 2 and 7, while stress classification benefits most from tasks 3 and 2.

6.5.2 Trinary Classification Task Effectiveness

Table 6.4: Task Effectiveness for Trinary Classification

Emotion	**Task Ranking (Best to Least)**
Depression	1, 7, 6, 3, 2, 4, 5
Anxiety	2, 6, 7, 1, 4, 5, 3
Stress	2, 7, 3, 6, 1, 5, 4

In the context of trinary classification we can see in Figure 5.10, task 1 stands out as the most effective for depression, with task 7 also being considerable. For anxiety, tasks 2 and 6, and potentially task 7, are most indicative. Stress is best analyzed by task 2, with task 7 also being a viable option.

Chapter 7

Conclusion

This thesis has embarked on a comprehensive journey through the landscape of emotion classification from handwriting, guided by a thorough literature review that identified existing approaches and best practices. Our investigation revealed significant gaps in the state of the art, particularly the lack of sufficient metrics beyond accuracy and the use of an inadequate number of input features for robust model evaluation.

We chose the EMOTHAW dataset as the cornerstone of our research, proposing a novel approach that involved extracting a vast array of features and employing ANOVA to rank their statistical significance. This method allowed us to focus on the most impactful features for emotion classification. We implemented normalization techniques and preprocessing methods, specifically LDA and PCA. Notably, PCA demonstrated high efficacy as a standalone method and also when combined with LDA in a sequential pipeline, enhancing the emotion classification process.

In our quest to mitigate the errors of individual machine learning models, we embraced ensemble learning—a meta approach that amalgamates bagging, boosting, and stacking techniques with various structures and models. This strategy led to the discovery of multiple argument configurations for our classifiers, enabling us to identify the most effective classifiers for each emotion under both binary and trinary classifications.

The classifiers we developed demonstrated commendable performance metrics, fulfilling our objective of creating reliable tools for emotion classification. Throughout this thesis, we conducted a series of experiments that not only refined our understanding but also allowed us to compile a list of the most significant features, laying the groundwork for future research endeavors.

However, our classifiers exhibited sensitivity to specific tasks, which became apparent when we attempted to apply our model to the CIU dataset. The results were less than satisfactory, raising questions about whether the discrepancy was due to the different methods of stress induction or the variation in tasks. This area remains ripe for further exploration.

A pivotal discovery in our research was the occurrence of data leakage during preprocessing with validation data. This served as a cautionary tale for others in the field, emphasizing the importance of preventing information leakage in cross-validation processes. Additionally, we learned the criticality of setting a minimum number of features when dealing with dynamic feature selection to ensure enough information for the classifier.

In conclusion, this thesis has not only contributed to the field of emotion classification from handwriting by developing effective classifiers and identifying key features but has also highlighted areas for future research. The lessons learned from both our successes and

setbacks have provided valuable insights that will undoubtedly benefit subsequent studies in this domain.

As we close this chapter, we acknowledge the iterative nature of scientific research—a path marked by both triumphs and tribulations, each equally important in the pursuit of knowledge.

Bibliography

[1] Alaei, F. and Alaei, A. *Handwriting analysis: Applications in person identification and Forensic*. Springer International Publishing, Jan 1970. Available at: https://link.springer.com/chapter/10.1007/978-3-031-10706-1_7.

[2] Azmi, M., Fathima, S. L. and Wasid, M. Unveiling Emotions through Handwriting: A Data Analysis Approach. *2023 12th International Conference on Advanced Computing (ICoAC)*. 2023, p. 1–6. DOI: 10.1109/ICoAC59537.2023.10250008.

[3] Bay Ayzeren, Y., Erbilek, M. and Çelebi, E. Emotional State Prediction From Online Handwriting and Signature Biometrics. *IEEE Access*. 2019, vol. 7, p. 164759–164774. DOI: 10.1109/ACCESS.2019.2952313.

[4] Bhattacharya, S., Islam, A. and Shahnawaz, S. TEmoDec: Emotion Detection from Handwritten Text using Agglomerative Clustering. In: *2022 First International Conference on Artificial Intelligence Trends and Pattern Recognition (ICAITPR)*. 2022, p. 1–6. DOI: 10.1109/ICAITPR51569.2022.9844210.

[5] Bhattacharya, S., Islam, A. and Shahnawaz, S. TEmoDec: Emotion Detection from Handwritten Text using Agglomerative Clustering. In: *2022 First International Conference on Artificial Intelligence Trends and Pattern Recognition (ICAITPR)*. 2022, p. 1–6. DOI: 10.1109/ICAITPR51569.2022.9844210.

[6] Böck, R., Egorow, O., Siegert, I. and Wendemuth, A. Comparative Study on Normalisation in Emotion Recognition from Speech. In: Horain, P., Achard, C. and Mallem, M., ed. *Intelligent Human Computer Interaction*. Cham: Springer International Publishing, 2017, p. 189–201. ISBN 978-3-319-72038-8.

[7] Brownlee, J. *How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification* [online]. 2020 [cit. 2024-01-12]. Available at: https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/.

[8] Brownlee, J. *A gentle introduction to ensemble learning algorithms*. Apr 2021. Available at: https://machinelearningmastery.com/tour-of-ensemble-learning-algorithms/.

[9] Brownlee, J. *Stacking Ensemble Machine Learning with python*. Apr 2021. Available at: https://machinelearningmastery.com/stacking-ensemble-machine-learning-with-python/.

[10] Chamishka, S., Madhavi, I., Nawaratne, R., Alahakoon, D., De Silva, D. et al. A voice-based real-time emotion detection technique using recurrent neural

network empowered feature modelling. *Multimedia Tools and Applications*. Oct 2022, vol. 81, no. 24, p. 35173–35194. DOI: 10.1007/s11042-022-13363-4. ISSN 1573-7721. Available at: https://doi.org/10.1007/s11042-022-13363-4.

[11] Chaubey, G. *Personality Prediction using Handwriting images*. 2020. DOI: 10.34740/KAGGLE/DSV/1786915. Available at: https://www.kaggle.com/dsv/1786915.

[12] Chernov, Y. and Caspers, C. Formalized Computer-Aided Handwriting Psychology: Validation and Integration into Psychological Assessment. *Behavioral Sciences*. january 2020, vol. 10, p. 27. DOI: 10.3390/bs10010027.

[13] Fairhurst, M., Erbilek, M. and Li, C. Study of automatic prediction of emotion from handwriting samples. *IET Biometrics*. 2015, vol. 4, no. 2, p. 90–97. DOI: https://doi.org/10.1049/iet-bmt.2014.0097. Available at: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-bmt.2014.0097.

[14] Galaz, Z., Mucha, J., Zvoncak, V. and Mekyska, J. *Handwriting features*. Sep 2023. Available at: https://github.com/BDALab/handwriting-features/.

[15] Garcia Garcia, J. M., Penichet, V. M. R. and Lozano, M. D. Emotion detection: a technology review. In: *Proceedings of the XVIII International Conference on Human Computer Interaction*. New York, NY, USA: Association for Computing Machinery, 2017. Interacción '17. DOI: 10.1145/3123818.3123852. ISBN 9781450352291. Available at: https://doi.org/10.1145/3123818.3123852.

[16] Hassan, M. *ANOVA (Analysis of variance) – Formulas, Types, and Examples* [online]. 2024 [cit. 2024-04-25]. Available at: https://researchmethod.net/anova/.

[17] Jia, W., Sun, M., Lian, J. and Hou, S. Feature dimensionality reduction: a review. *Complex & Intelligent Systems*. Jun 2022, vol. 8, no. 3, p. 2663–2693. DOI: 10.1007/s40747-021-00637-x. ISSN 2198-6053. Available at: https://doi.org/10.1007/s40747-021-00637-x.

[18] Kedar, S. and Rokade, S. Recognition of Emotional State Based On Handwriting Analysis and Psychological Assessment. *International Journal of Engineering and Advanced Technology*. august 2019, vol. 8, p. 4395–4402. DOI: 10.35940/ijeat.F8960.088619.

[19] Keldenich, T. *Inside Machine Learning* [online]. 2021 [cit. 2024-01-12]. Available at: https://inside-machinelearning.com/en/recall-precision-f1-score-simple-metric-explanation-machine-learning/.

[20] Kenton, W. *What Is Analysis of Variance (ANOVA)?* [online]. 2024 [cit. 2024-04-25]. Available at: https://www.investopedia.com/terms/a/anova.asp.

[21] Likforman Sulem, L., Esposito, A., Faundez Zanuy, M., Clémençon, S. and Cordasco, G. EMOTHAW: A Novel Database for Emotional State Recognition From Handwriting and Drawing. *IEEE Transactions on Human-Machine Systems*. Apr 2017, vol. 47, no. 2, p. 273–284. DOI: 10.1109/THMS.2016.2635441.

[22] Marano, G., Traversi, G., Gaetani, E., Sani, G., Mazza, S. et al. Graphology: An Interface Between Biology, Psychology and Neuroscience. *Psychological Disorders and Research*. 2020. DOI: http://dx.doi.org/10.31487/j.PDR.2020.03.05. ISSN 2674-2470. Available at: https://www.sciencerepository.org/graphology-an-interface-between-biology-psychology-and-neuroscience_PDR-2020-3-105.

[23] Mucha, J., Galaz, Z., Zvoncak, V. and Mekyska, J. *Handwriting sample*. November 2023. Available at: https://github.com/BDALab/handwriting-sample/.

[24] Nolazco Flores, J. A., Faundez Zanuy, M., Velázquez Flores, O. A., Del Valle Soto, C., Cordasco, G. et al. Mood State Detection in Handwritten Tasks Using PCA–mFCBF and Automated Machine Learning. *Sensors*. 2022, vol. 22, no. 4. DOI: 10.3390/s22041686. ISSN 1424-8220. Available at: https://www.mdpi.com/1424-8220/22/4/1686.

[25] *Oxford Learner's Dictionaries* [online]. 2024 [cit. 2024-01-04]. Available at: https://www.oxfordlearnersdictionaries.com/definition/english/emotion.

[26] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, vol. 12, p. 2825–2830.

[27] PLAMPER, J., REDDY, W., ROSENWEIN, B. and STEARNS, P. THE HISTORY OF EMOTIONS: AN INTERVIEW WITH WILLIAM REDDY, BARBARA ROSENWEIN, AND PETER STEARNS. *History and Theory*. [Wesleyan University, Wiley]. 2024/05/09/ 2010, vol. 49, no. 2, p. 237–265. Full publication date: May 2010. Available at: http://www.jstor.org/stable/40864443.

[28] Pollak, S. D., Camras, L. A. and Cole, P. M. Progress in understanding the emergence of human emotion. *Developmental Psychology*. US: American Psychological Association. 2019, vol. 55, no. 9, p. 1801–1811. DOI: 10.1037/dev0000789. Available at: https://doi.org/10.1037/dev0000789.

[29] Rahman, A. U. and Halim, Z. Identifying dominant emotional state using handwriting and drawing samples by fusing features. *Applied Intelligence*. Feb 2023, vol. 53, no. 3, p. 2798–2814. DOI: 10.1007/s10489-022-03552-x. ISSN 1573-7497. Available at: https://doi.org/10.1007/s10489-022-03552-x.

[30] Smolyakov, V. *Ensemble Learning to Improve Machine Learning Results* [online]. 2017 [cit. 2024-01-22]. Available at: https://medium.com/cube-dev/ensemble-learning-d1dcd548e936.

[31] Turabzadeh, S., Meng, H., Swash, R. M., Pleva, M. and Juhar, J. Facial Expression Emotion Detection for Real-Time Embedded Systems. *Technologies*. 2018, vol. 6, no. 1. DOI: 10.3390/technologies6010017. ISSN 2227-7080. Available at: https://www.mdpi.com/2227-7080/6/1/17.

[32] Zhao, S., Jia, G., Yang, J., Ding, G. and Keutzer, K. Emotion Recognition From Multiple Modalities: Fundamentals and methodologies. *IEEE Signal Processing Magazine*. Institute of Electrical and Electronics Engineers (IEEE). november 2021, vol. 38, no. 6, p. 59–73. DOI: 10.1109/msp.2021.3106895. ISSN 1558-0792. Available at: http://dx.doi.org/10.1109/MSP.2021.3106895.

Appendix A

Detailed Account of Unsuccessful Experiments

This chapter provides an in-depth look at the initial phase of experimentation that ultimately did not yield the desired results, leading to significant changes in our approach. The experiments detailed here were pivotal in highlighting the limitations of our initial methods, particularly concerning overfitting and insufficient feature selection.

A.1 Argument Search Using Cross-Validation

As referenced in Failed Experiment 1 and Failed Experiment 2 5.4, our initial experimentation phase began without the incorporation of the *minimum_features* argument and relied on 10-fold cross-validation. This approach, while standard in many machine learning practices, proved to be inadequate for our specific application.

The following sections provide a comprehensive breakdown of these early experiments, outlining the methodologies employed, the challenges encountered, and the lessons learned that informed the subsequent adjustments to our experimental design.

A.1.1 Binary classifier

The initial phase of experimentation, conducted via the *argument_experiment.py* script, was concentrated on binary classifiers. This phase is dedicated to refining the selection of arguments, which are instrumental in the construction of robust classification models. Our initial focus was directed towards the preprocessing techniques, as they play a crucial role in shaping the input data for optimal classifier performance.

Experiment 1: Data preprocessing

Our first experiment commenced with the whole dataset encompassing all seven tasks. To streamline the complexity, we narrowed the choice of the final meta-model to three well-established algorithms: Logistic Regression, K-Nearest Neighbors, and Random Forest. We adopted stacking as our ensemble learning strategy, with a variation in the number of inner models set at either five or fifteen.

The execution of *ensemble.py* was carried out with these specified arguments, alongside varying preprocessing methods. Our aim was to evaluate the impact of LDA transformation, PCA transformation, the absence of both preprocessing techniques, and the combined effect

of a PCA-LDA pipeline on the model's accuracy. The results of this comparative study are visually represented in Figure A.1, which clearly illustrates the superior performance of models preprocessed with LDA, particularly in the classification of anxiety and depression. The findings from this experiment have led us to adopt **LDA** as our preprocessing method of choice, given its significant contribution to enhancing model accuracy.

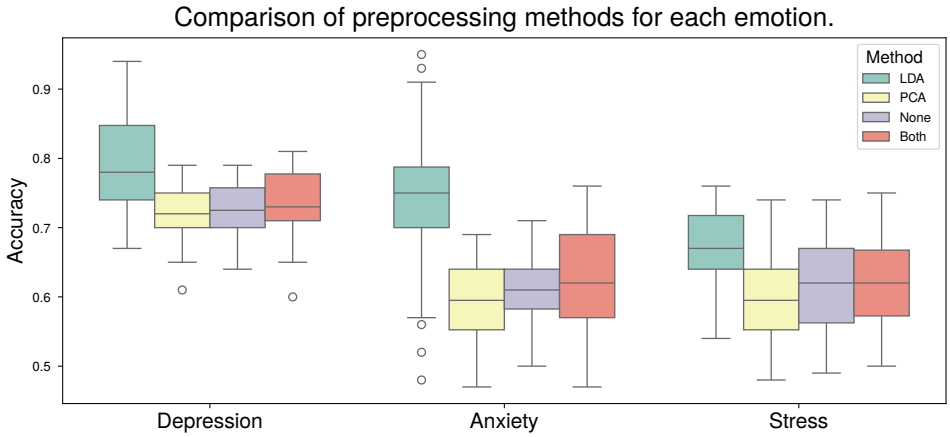

Figure A.1: The graph showcases the comparative accuracy of binary classification models utilizing a stacking ensemble approach. The models, which include either five or fifteen inner models, are evaluated with a meta-model consisting of KNN, Logistic Regression, or Random Forest. The accuracy is measured across different preprocessing techniques, highlighting the effectiveness of LDA in improving model performance.

Experiment 2: Ensemble Technique

With the preprocessing method fixed to Linear Discriminant Analysis (LDA), we embarked on a comparative study of three prominent ensemble techniques: stacking, bagging, and boosting. The objective was to ascertain which technique would yield the most accurate classification models.

In this experiment, the ensemble techniques were scrutinized under similar conditions as in previous experiment A.1.1, using all seven tasks, three possible meta models and either 5 or 15 inner models. The graphical representation detailed in Figure A.2, revealed a nuanced landscape of performance metrics. While no technique emerged as a definitive leader, stacking achieved marginally superior accuracy, with boosting trailing closely behind. Bagging, on the other hand, demonstrated the least favorable results, suggesting its exclusion from future consideration.

The results indicate that while stacking stands out as the preferred technique due to its slightly higher accuracy, the difference is not substantial enough to disregard boosting entirely. Therefore, we conclude that **stacking** will be our primary ensemble method moving forward, but we remain open to further exploration of boosting in subsequent phases of our research.

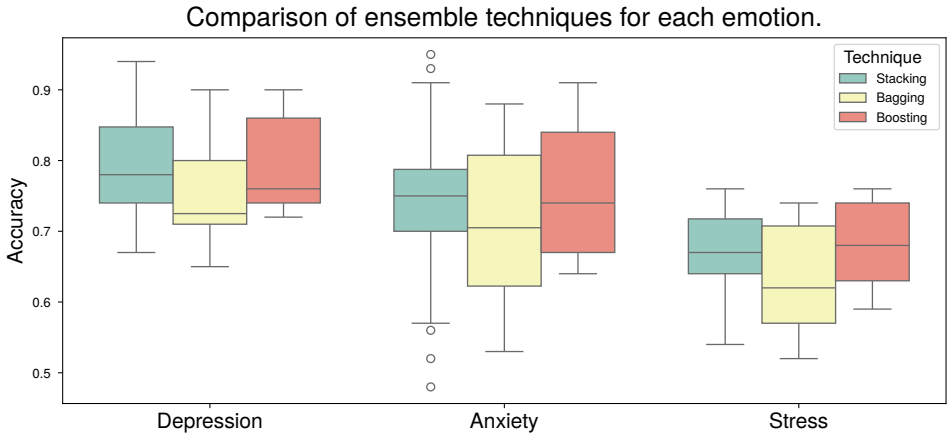

Figure A.2: The graph showcases the comparative accuracy of binary classification models employing different ensemble techniques. Stacking, bagging, and boosting were analyzed under uniform conditions. The models, which include either five or fifteen inner models, are evaluated with a meta-model consisting of KNN, Logistic Regression, or Random Forest. Stacking showing a slight edge in performance.

Experiment 3: Task-Specific Input Data

This experiment probes the hypothesis that certain tasks may yield more potent discriminative power for the classification of specific emotions.

We conducted an evaluation of classifiers employing LDA preprocessing and a stacking ensemble technique. The ensemble was configured with all 25 potential meta-models, and the number of inner models was set to either 5 or 15. The data was partitioned based on the input task to ascertain the impact of task-specific data on classifier performance.

The insights gleaned from this experiment are encapsulated in Figure A.3, which elucidates the differential impact of tasks on the efficacy of emotion classification. A discernible pattern emerges from the analysis: certain tasks significantly bolster the classifier's ability to detect particular emotions. For instance, when classifying depression, input data from task 1 outperforms others, with task 3 following suit. Similarly, for anxiety, task 1 stands out as the most effective, with task 2 as a secondary contributor. In the case of stress, while the results are more homogenous, tasks 3 and 6 emerge as the most promising sources of input data.

The findings from this experiment suggest that within the EMOTHAW database [21], not all tasks carry equal weight in the context of emotion detection. A high-quality binary classifier for depression and anxiety can be achieved predominantly with data from Task 1, whereas for stress, Task 3 is the most informative. As delineated in Table 3.1, these tasks pertain to the drawing of two pentagons and the clock drawing test, respectively. Despite these revelations, our endeavor to construct the most accurate classifier persists, as we continue to refine models for each emotion using data from all tasks, ensuring versatility and robustness in our classification approach.

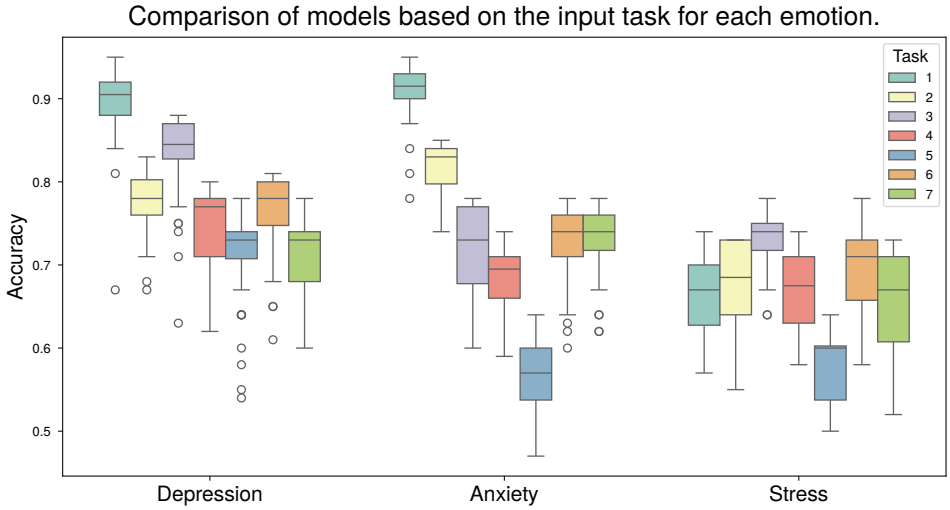

Figure A.3: The graph presents the accuracy achieved by classifiers for each emotion, highlighting the varying impact of task-specific input data on model performance.

Experiment 4: Number of inner models

In this experiment we focus on the number of inner models used in our classifier. We compared classifiers employing LDA preprocessing and a stacking ensemble technique with all 25 possible meta-classifiers. The input data used was from all 7 tasks. We compared classifiers with 5, 10, 15, 20 and 25 inner models.

As we can see in Figure A.4, the difference between different number of inner models doesn't influence the resulting accuracy that much. There is a slight decline of accuracy with growing number of models, but this is almost negligible difference.

The fourth experiment in our series investigates the influence of the number of inner models within our classifier's architecture. We employed LDA preprocessing and a stacking ensemble technique, incorporating all 25 possible meta-classifiers. The dataset again comprised input data from all seven tasks.

We scrutinized classifiers configured with varying quantities of inner models: 5, 10, 15, 20, and 25. This range was selected to cover a spectrum from a minimal ensemble to a substantially large one, allowing us to observe any trends associated with the number of models.

The results, as depicted in Figure A.4, indicate a subtle trend. Contrary to expectations, the variation in the number of inner models does not significantly impact the accuracy of the classifiers. There is a marginal decrease in accuracy as the number of models increases, but this decline is minimal and does not suggest a strong correlation between model quantity and classifier performance.

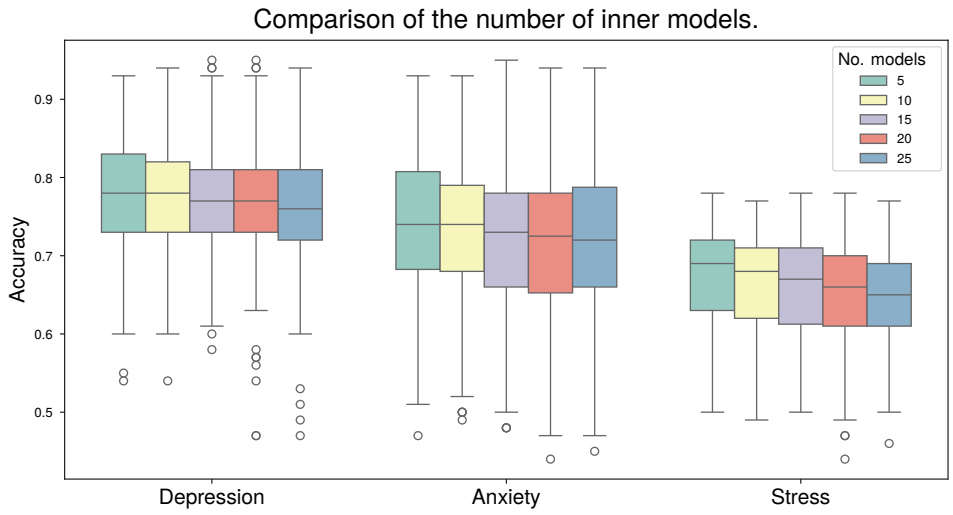

Figure A.4: The graph illustrates the relationship between the number of inner models and the accuracy of the classifiers. It highlights the minimal impact that increasing the number of inner models has on the overall accuracy.

Experiment 5: Meta-Model Selection

The final experiment in this chapter scrutinizes the impact of the meta-model selection within our stacking ensemble framework. This experiment was conducted using a stacking ensemble with 15 inner models, coupled with LDA preprocessing. The tasks selected for input data were those previously shown in Section A.1.1 with the highest efficacy for each emotion: Task 1 for depression and anxiety, and Task 3 for stress.

The classifiers for each emotion were evaluated using every possible meta-model to determine the extent to which the choice of the final model affects the results.

The analysis, as inferred from the Figure A.5, indicates that the differences among the meta-models are minimal, with some variations dipping slightly lower but without significant deviation.

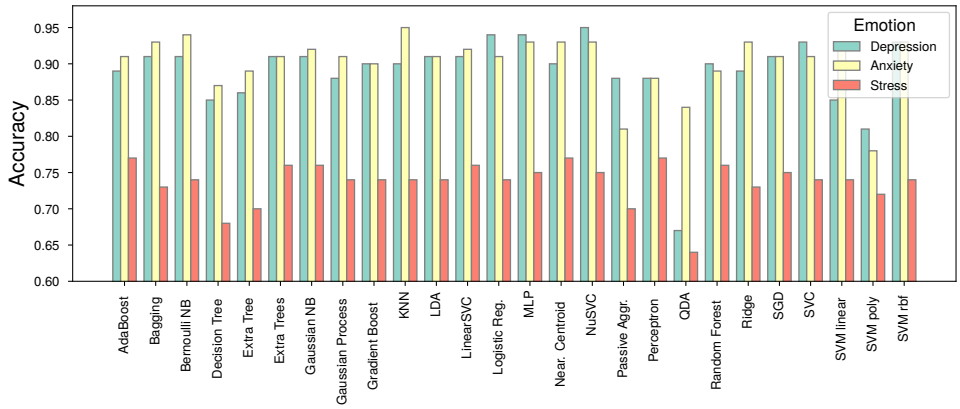

Figure A.5: The graph displays the performance of classifiers across different meta-models, highlighting the negligible differences in accuracy, suggesting the need for further exploration of all meta-models.

A.1.2 Trinary Classifier

This section extends the exploration of argument effects to classifiers trained on trinary data separation. The DASS value thresholds for trinary classification are detailed in Table 4.1. The experimental approach mirrors that of the binary classifiers, utilizing the *argument_experiment.py* script once more.

Experiment 6: Data Preprocessing

The initial trinary experiment evaluated the effectiveness of LDA and PCA preprocessing techniques. Employing the same parameters as before—stacking ensemble with 5 or 15 inner models and a meta-model chosen from KNN, Logistic Regression, or Random Forest—we processed input data from each task with trinary evaluation. Four combinations of preprocessing techniques were tested.

Figure A.6 demonstrates a significant increase in accuracy when using LDA, reaffirming its selection as our primary preprocessing method.

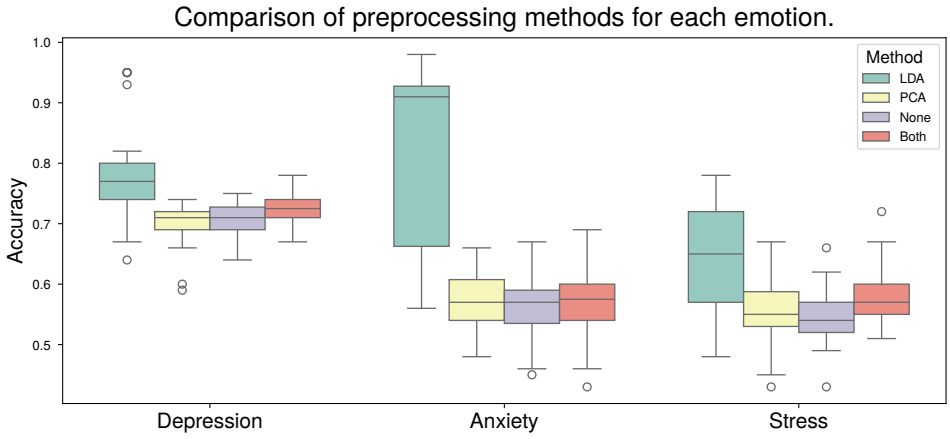

Figure A.6: The graph illustrates the accuracy improvements achieved through LDA preprocessing in trinary classification.

Experiment 7: Ensemble Technique

This experiment highlights the differences in ensemble techniques when applied to trinary data. We conducted runs with data from all tasks preprocessed using LDA. Classifiers with 5 or 15 inner models employed Logistic Regression, KNN, or Random Forest as the meta-model. As depicted in Figure A.7, the mean values suggest stacking's superiority in every case. However, focusing on maximum values for depression, bagging yielded a few superior results. Conversely, for stress, bagging generally underperformed. Stacking remains our primary technique for further argument testing, but we will revisit bagging and boosting in future explorations.

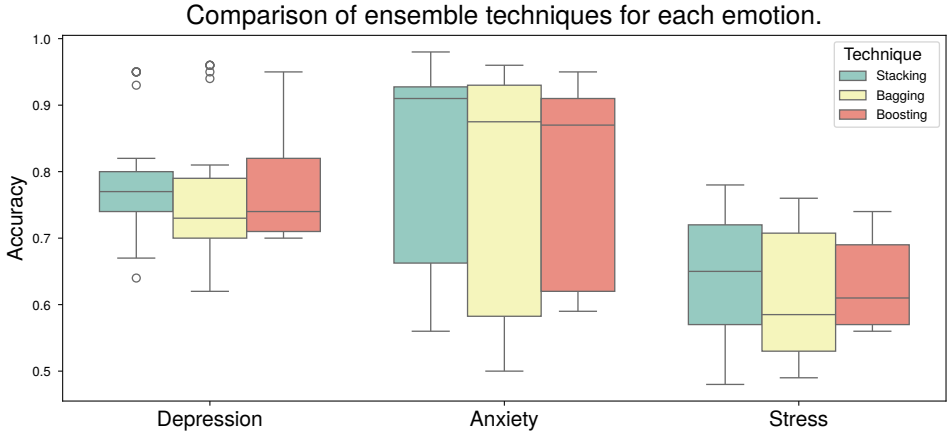

Figure A.7: The graph compares the performance of different ensemble techniques on trinary data, indicating stacking as the most consistent performer.

Experiment 8: Task-Specific Input Data

We revisited the significance of task-specific input data on classification outcomes. All classifiers were evaluated using a stacking ensemble with 5 or 15 inner models across the 25 meta-models. LDA-preprocessed data from each task was examined separately for resulting accuracy. Figure A.8 presents surprising contrasts to binary classification results. For depression, Task 1 remains the clear leader, while for anxiety, Task 7 surpasses with Task 1 closely behind. For stress, Task 3 maintains its lead, with Task 1 now as a strong secondary option.

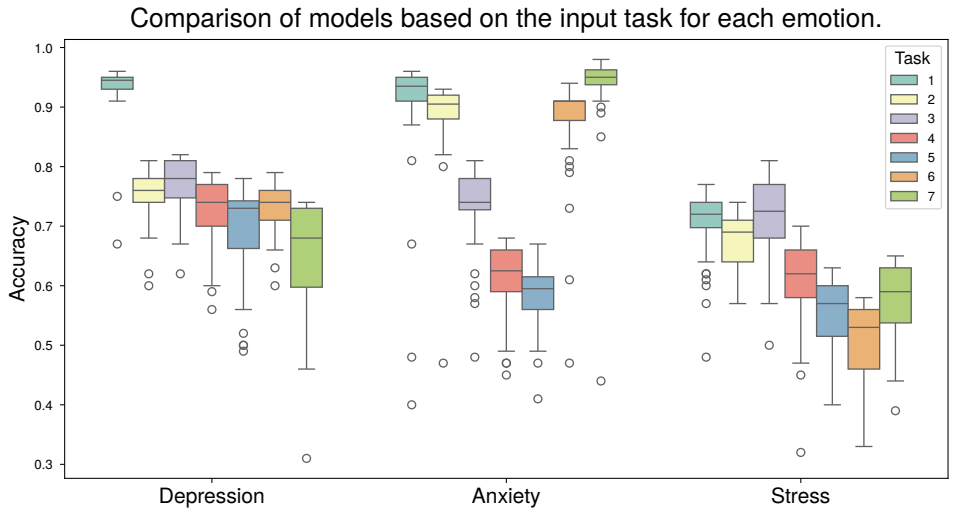

Figure A.8: The graph reveals task-specific input data's impact on trinary classification accuracy, showing distinct preferences for different emotions.

Experiment 9: Number of Inner Models

Mirroring the binary classification experiments, we examined the impact of the number of inner models. Stacking ensemble classifiers with any of the 25 meta-models were evaluated using all tasks preprocessed with LDA. Figure A.9 indicates a slight trend across all emotions, where increasing the number of inner models correlates with a minor drop in accuracy.

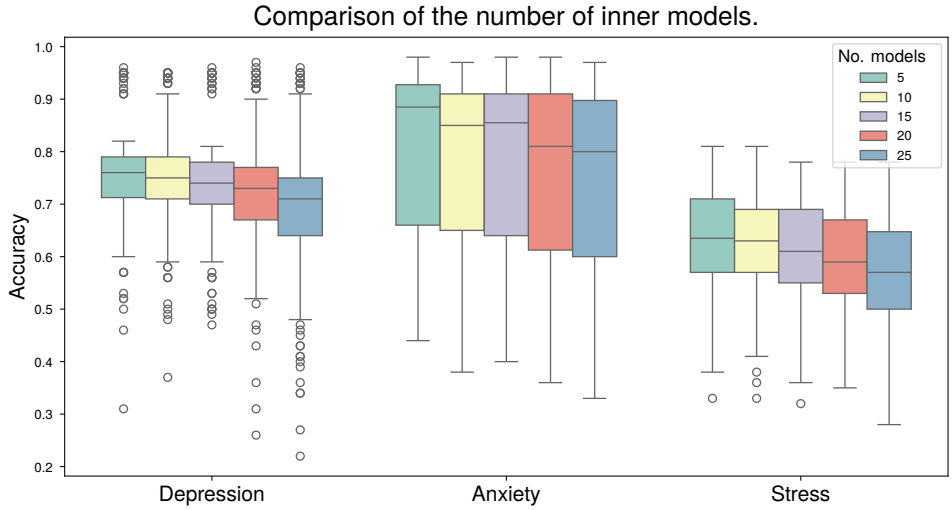

Figure A.9: The graph depicts the subtle effect of varying inner model quantities on the accuracy of trinary classifiers.

Experiment 10: Meta-Model Selection

The concluding trinary experiment assessed the influence of different meta-models. Stacking ensemble models with 15 inner models were evaluated, with input data tailored for each emotion and preprocessed using LDA: Task 1 for depression, Task 7 for anxiety, and Task 3 for stress. The findings in Figure A.10 echo the binary data results, showing that the choice of meta-model has a minimal effect on accuracy. Similar dips in performance were observed for certain models.

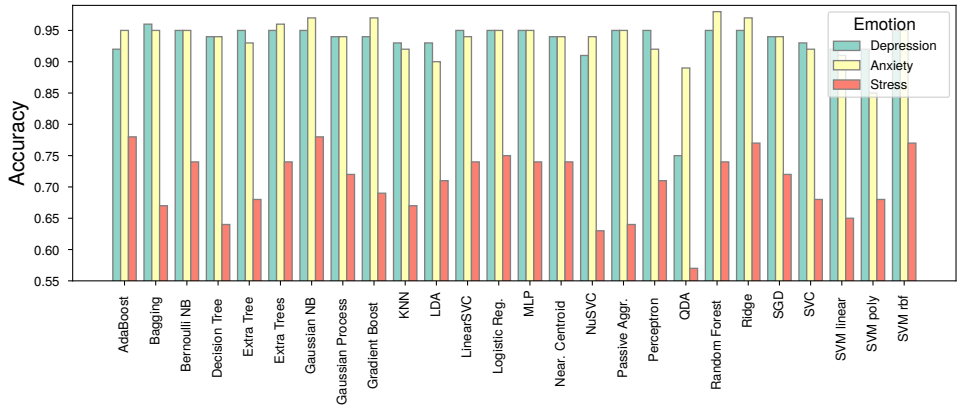

Figure A.10: This graph compares the accuracy of trinary classifiers across various meta-models, highlighting the overall minimal impact on performance.

A.2 Concluding Remarks on Preliminary Experiments

Reflecting on the initial phase of experimentation, we observed significant fluctuations in performance metrics, particularly in the Task-Specific Input Data sections. For instance, Figure A.3 illustrates notable declines in accuracy for task 5 when classifying anxiety and stress in the binary classification setting. A similar trend was evident in the trinary classification, as depicted in Figure A.8, where tasks 3, 4, and 5 for anxiety, task 7 for depression, and tasks 5, 6, and 7 for stress exhibited subpar performance. These inconsistencies were attributed to an insufficient number of features, a challenge we addressed by introducing the *minimum_features* argument.

Towards the conclusion of the binary experimentation phase, detailed in Section A.1.1, our models began to achieve exceptionally high metrics, reaching 100% accuracy, precision, recall, and F1 score. While these results may initially appear promising, they raised concerns about overfitting. Recognizing this, we ceased further experimentation with these configurations to prevent our models from learning the noise and idiosyncrasies in our training data rather than the underlying patterns.

Appendix B

Feature Importance for Classification

In this chapter, we provide detailed lists of features for emotion classification, organized by their significance. These lists rank features based on their average ranking in the ANOVA feature selection process as explored in Experiment 12, selection frequency shown in Experiment 13, and the selection frequency of merged base features from Experiment 14 5.2. The tables are designed to provide a clear and concise overview of the data without excessive length.

B.1 Features by Average ANOVA Position

Feature Name	**Average Position**
ratio_of_writing_durations	147.0
writing_tempo_on_paper	159.0
acceleration_on_paper_y_slope_of_linear_regression	165.4
vertical_valleys_duration_iqr	169.4
acceleration_in_air_y_median	175.8
jerk_on_paper_y_cv_parametric	176.1
stroke_length_on_paper_iqr	182.0
relative_number_of_changes_in_velocity_profile	183.5
jerk_on_paper_xy_cv_parametric	184.4
vertical_peaks_values_slope_of_linear_regression	189.5
tilt_in_air_slope_of_linear_regression	190.7
vertical_valleys_velocity_quartile_3	190.7
vertical_peaks_duration_iqr	193.0
velocity_on_paper_y_slope_of_linear_regression	193.3
acceleration_on_paper_xy_slope_of_linear_regression	194.1
velocity_in_air_xy_percentile_5	195.8
stroke_height_in_air_percentile_5	196.5
relative_number_of_changes_in_azimuth	197.2
vertical_valleys_velocity_std	197.9

Feature Name	Average Position
stroke_length_on_paper_quartile_3	199.3
pressure_slope_of_linear_regression	200.7
number_of_interruptions	200.8
vertical_valleys_velocity_percentile_95	201.2
stroke_height_on_paper_slope_of_linear_regression	202.3
acceleration_on_paper_xy_median	202.5
relative_number_of_changes_in_tilt	202.6
vertical_valleys_distance_quartile_3	203.1
jerk_in_air_xy_cv_nonparametric	205.3
writing_stops_mean	205.7
vertical_peaks_duration_mean	206.2

B.2 Features by Selection Frequency

Feature Name	Selection Frequency
acceleration_on_paper_y_slope_of_linear_regression	14/42 (33.3%)
ratio_of_writing_durations	12/42 (28.6%)
stroke_length_on_paper_mean	11/42 (26.2%)
stroke_length_on_paper_percentile_95	10/42 (23.8%)
jerk_on_paper_y_cv_parametric	10/42 (23.8%)
writing_tempo_on_paper	10/42 (23.8%)
stroke_length_on_paper_median	10/42 (23.8%)
stroke_duration_in_air_slope_of_linear_regression	10/42 (23.8%)
stroke_width_in_air_slope_of_linear_regression	10/42 (23.8%)
relative_number_of_changes_in_velocity_profile	10/42 (23.8%)
vertical_valleys_duration_iqr	9/42 (21.4%)
tilt_in_air_slope_of_linear_regression	9/42 (21.4%)
ratio_of_stroke_durations_percentile_5	9/42 (21.4%)
velocity_in_air_xy_percentile_5	9/42 (21.4%)
relative_number_of_intra_stroke_intersections cv_nonparametric	9/42 (21.4%)
number_of_intra_stroke_intersections_cv_nonparametric	9/42 (21.4%)
number_of_intra_stroke_intersections_quartile_3	9/42 (21.4%)
number_of_intra_stroke_intersections_iqr	9/42 (21.4%)
relative_number_of_intra_stroke_intersections_iqr	9/42 (21.4%)
relative_number_of_intra_stroke_intersections_quartile_3	9/42 (21.4%)
relative_number_of_intra_stroke_intersections_median	9/42 (21.4%)
velocity_in_air_y_quartile_1	9/42 (21.4%)
writing_stops_median	8/42 (19.0%)
writing_stops_mean	8/42 (19.0%)
stroke_duration_on_paper_mean	8/42 (19.0%)
jerk_on_paper_xy_cv_parametric	8/42 (19.0%)

Feature Name	Selection Frequency
writing_stops_iqr	8/42 (19.0%)
vertical_peaks_indices_cv_nonparametric	8/42 (19.0%)
vertical_valleys_velocity_quartile_3	8/42 (19.0%)
vertical_valleys_velocity_percentile_95	8/42 (19.0%)

B.3 Merged Base Features by Selection Frequency

Merged Feature Name	Selection Frequency
ratio_of_writing_durations	12/42 (28.6%)
relative_number_of_changes_in_velocity_profile	10/42 (23.8%)
relative_number_of_changes_in_y_profile	6/42 (14.3%)
relative_number_of_intra_stroke_intersections	63/462 (13.6%)
vertical_valleys_velocity	63/462 (13.6%)
writing_tempo	11/84 (13.1%)
relative_number_of_changes_in_azimuth	5/42 (11.9%)
relative_total_number_of_intra_stroke_intersections	5/42 (11.9%)
writing_stops	54/462 (11.7%)
ratio_of_stroke_durations	49/462 (10.6%)
stroke_length	94/924 (10.2%)
writing_height	8/84 (9.5%)
vertical_valleys_duration	43/462 (9.3%)
stroke_width	84/924 (9.1%)
vertical_peaks_duration	42/462 (9.1%)
pressure	47/546 (8.6%)
number_of_interruptions	7/84 (8.3%)
stroke_height	67/924 (7.3%)
relative_number_of_changes_in_x_profile	3/42 (7.1%)
relative_number_of_changes_in_tilt	3/42 (7.1%)
relative_number_of_changes_in_pressure	3/42 (7.1%)
number_of_interruptions_relative	3/42 (7.1%)
relative_number_of_inter_stroke_intersections	3/42 (7.1%)
tilt	66/1008 (6.5%)
jerk	169/2772 (6.1%)
vertical_peaks_velocity	28/462 (6.1%)
total_number_of_intra_stroke_intersections	5/84 (6.0%)
vertical_valleys_distance	27/462 (5.8%)
acceleration	151/2772 (5.4%)
velocity	205/3780 (5.4%)

Each table provides a detailed account of the features' performance across the experiments, offering insights into their relative importance. For further details on the methodology behind feature selection and the implications for emotion classification, please refer to Section 3.2.2.