2023-08-31

Prediction of Cognitive Status of Alzheimer's Disease Using Machine Learning Methods with Structural Magnetic Resonance Imaging

Chen, Zhenyu

UNIVERSITY OF CALGARY

Prediction of Cognitive Status of Alzheimer's Disease Using Machine Learning Methods with Structural Magnetic Resonance Imaging

by

Zhenyu Chen

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN MATHEMATICS AND STATISTICS

CALGARY, ALBERTA

AUGUST, 2023

Abstract

Alzheimer's disease is a progressive form of dementia. A patient diagnosed with Alzheimer's disease usually underwent three types of cognitive status: normal cognitive (NC), mild cognitive impairment (MCI), and Alzheimer's disease (AD). This thesis studies two machine learning models to predict cognitive status based on the region of interest features extracted from structural magnetic resonance imaging. The support vector machine (SVM) model uses recursive feature elimination to select the proper number of features and the multilayer perceptron (MLP) neural network model uses Bayesian optimization to tune the hyperparameters. When applied to a real data set with 1684 observations, the SVM model achieved 68.84% accuracy while the MLP model achieved 70.92% accuracy for multi-classification. We further compared the two models in three binary classifications, namely, NC vs MCI, MCI vs AD, and NC vs AD. The SVM model achieved the area under the receiver operating characteristic curves (AUCs) of 72%, 71%, 86%, while the MLP model achieved the AUCs of 68%, 73%, 88%, respectively. In general, the MLP model performs better than the SVM model. Both models can be used to assist in the early diagnosis of Alzheimer's disease.

Preface

This thesis is an original work by the author under the supervision of Dr. Gemai Chen. No part of this thesis has been previously published.

Acknowledgements

The completion of this dissertation has been made possible through the guidance of my supervisor, Dr. Gemai Chen. I am grateful for Dr. Chen's unwavering support and mentorship throughout this journey, which has immensely helped in shaping my academic and research capabilities.

I would like to extend my gratitude to all my classmates and teachers who have generously provided their assistance in my studies. Your continued support has been instrumental in my pursuit of this academic endeavor. I would also like to express my heartfelt gratitude to the Department of Mathematics and Statistics and the Faculty of Graduate Studies for their unwavering financial support throughout the completion of this dissertation.

Lastly, I owe a tremendous amount of thanks to my family. Your enduring love, encouragement, and support have been my foundation, providing me with the strength to surmount the challenges along this journey.

To my beloved family...

Table of Contents

List of Figures and Illustrations

List of Tables

List of Symbols, Abbreviations and Nomenclature

Symbol or abbreviation	Definition
AD	Alzheimer's disease
MCI	Mild cognitive impairment
NC	Normal cognitive
MRI	Magnetic resonance imaging
SVM	Support Vector Machine
KKT	Karush-Kuhn-Tucker conditions
RFE	Recursive feature elimination
MLP	Multilayer Perceptron
BayesOpt	Bayesion Optimization
GP	Gaussian Process
SHAP	SHapley Additive exPlanations
NACC	National Alzheimer's Coordinating Center
UDS	Uniform Data Set

Chapter 1

Introduction

1.1 Background

Alzheimer's disease (AD) is a progressive disease and the most common form of dementia, accounting for over 50% of all cases. As mentioned in World Alzheimer Report 2022, Alzheimer's Disease International (ADI) estimates the population of people who live with dementia will rise to 139 million in 2050. The navigation to diagnose post-dementia is challenging and arduous[1]. Research on Alzheimer's Disease has occupied a vital role in health care and development of epidemic studies worldwide.

The causes of Alzheimer's disease may be multifaceted, with many lifestyle-related factors in addition to genetic inheritance. Lifestyle-related factors, such as diabetes, obesity, physical and mental inactivity, depression, smoking, low educational attainment, and diet, may contribute to the development of dementia. The potential for primary prevention through the modification of such modifiable risk factors is substantial, however, it remains incompletely explored[2].

The neuropathology of Alzheimer's disease typically initiates and disseminates predictably, with pathological tangles exhibiting the strongest association with clinical symptoms. From a clinical point of view, patients with Alzheimer's disease may have impaired function in infor-

mation acquisition, logical inference, visuospatial process, expression disorder, and emotional fluctuations[3].

In the United States, the prevalence of Alzheimer's dementia increases with age, affecting approximately 1 in 9 individuals (10.8%) aged 65 years and older. The percentage of individuals with Alzheimer's dementia rises with advancing age, with 5.0% of those aged 65 to 74 years, 13.1% of those aged 75 to 84 years, and 33.3%of those aged 85 years and older experiencing the condition. Notably, Alzheimer's dementia can also develop in individuals younger than 65 years, with limited prevalence studies indicating that about 110 out of every 100,000 individuals between the ages of 30 and 64 years, or roughly 200,000 Americans in total, are affected by younger-onset dementia[4].

According to the National Institute on Aging-Alzheimer's Association, the symptomatic predementia phase of Alzheimer's dementia is defined as mild cognitive impairment (MCI) due to AD[5]. Patients typically undergo a long-term process of transitioning from normal cognitive (NC) functioning to mild cognitive impairment before progressing to Alzheimer's disease. The boundaries that distinguish these states are often unclear and difficult to define. As such, non-invasive methods for determining a patient's cognitive status are becoming increasingly popular, in the hope of avoiding the need for post-mortem examination.

1.2 Literature Review

Currently, two pharmacologic therapy classes are available for AD patients: cholinesterase inhibitors and memantine[6]. Early detection of prodromal symptoms that may correspond to dementia in patients can facilitate subsequent medical intervention. At present, Alzheimer's disease can only be definitively diagnosed post-mortem. Consequently, proposing statistical methods for distinguishing between MCI and AD can provide clinical physicians with an auxiliary tool from a statistical perspective.

Structural magnetic resonance imaging (sMRI) technology is important for statistical

study of AD detection. The research on Alzheimer's disease classification using sMRI relies on AD biomarkers. Cerebrospinal fluid (CSF) composition provides valuable insight into soluble pathogenic biomarkers that may be diagnostic for brain disease[7]. In CSF, biomarkers of Amyloid-β (Aβ) plaques and tau-related neurodegeneration are very important in clinical analysis[8].

AD classification through sMRI images mainly differentiates from the feature extraction techniques. Feature extraction methods are used to extract informative features from images, including pre-defined region of interest (ROIs) approaches, voxel-based morphometry (VBM) or voxel-based segmented tissue density maps, cortical surface or vertex-based method, wavelet transform-based method and texture-based method[9].

Chupin *et al.*[10] employed probabilistic and anatomical priors as part of their approach for hippocampus segmentation. Their method yielded an accuracy of 76% in segmenting the hippocampus in patients with AD and 71% in those with MCI out of a total cohort comprising 145 AD patients, 294 individuals with MCI, and 166 elderly subjects considered cognitively normal. Liu *et al.*[11] use a multi-model deep learning framework based on CNN for joint automatic hippocampal segmentation and AD classification with 97 AD, 233 MCI, and 119 NC subjects, achieving an accuracy of 88.9% and an AUC of 92.5% for classifying AD vs. NC subjects, and accuracy of 76.2% and an AUC of 77.5% for classifying MCI vs. NC subjects. Pennanen *et al.*[12] use stepwise discriminant function analysis to achieve accuracy between NC and AD patients with 90.7%, and between MCI and AD patients with 82.3% by exploring hippocampal volumes. Savio *et al.*[13] achieve an accuracy result of 86% with the AdaBoost applied to the SVM with radial basis function kernel trained independently at each VBM detected cluster, with 49 AD cases and 49 NC cases. Zhang *et al.*[14] use extreme learning machine with 58 patients with AD and 94 NCs to achieve an high accuracy of 96% based on VBM obtained from MRI. Park *et al.*[15] use 25 AD, 25 MCI, and 50 NC patients to achieve an accuracy of 85% between AD and NC, 79% between MCI and NC, and 69% between MCI and AD by using SVM and principal component analysis. Shuyu *et*

al.[16] uses SVM to analyze 24 amnestic MCI participants and 26 normal controls to achieve a classification accuracy in discriminating the two groups 76% in the left hemisphere and 80% in the right hemisphere.

1.3 Research Motivation

Many previous studies have encountered the issue of data leakage. Some studies[11] have overlooked the proper partitioning of the training, validation, and test sets, utilizing portions of the test set for model training. In some research[10,11,15,16], only a limited amount of data has been used to train machine learning models, resulting in a significantly small dataset for model evaluation. Additionally, certain studies[17] have failed to consider the autocorrelation present within an individual patient's MRI scans, thus neglecting to ensure that the dataset retains only one MRI per patient.

In contrast, the experimental data utilized in this thesis avoids such leakage problems through careful considerations in areas such as sample size and methodology for dataset partitioning. Furthermore, during the process of dataset selection and cleansing, only the data from each patient's first MRI scan was chosen. Therefore, we selected over 1300 data samples for training and validation and over 300 data samples for testing the model.

This thesis aims to propose methods to distinguish the status of experiment subjects with sMRI data. Exploring the mental status of experiment subjects can be achieved by utilizing the intrinsic relationship with the brain's structural gray matter volumes and cortical thicknesses, which have been processed through image analysis. Further analysis can be conducted by examining the numerical features extracted from different brain regions, which can shed light on the relative importance of these regions for the classification of Alzheimer's disease. An examination of the performance of both linear and non-linear machine learning methods on the dataset allows for a comparative analysis of potential advantages and disadvantages.

Although there have been numerous applications of machine learning methods for predicting the status of Alzheimer's disease, they often lack attention to the specific details of applying the models to the actual data. This thesis mainly focuses on exploring the differences between support vector machine and multilayer perceptron models in their application to Alzheimer's disease status prediction in both multi-class and binary classification scenarios. The thesis aims to investigate the generalizability of different models under varying data conditions and identify the challenges in prediction.

Chapter 2

Methodology

In this chapter, we outline the methodologies to be used in Chapter 3. We introduce and develop two machine learning methods-support vector machine and multilayer perceptron neural network for the later experiment.

2.1 Support Vector Machine

2.1.1 Introduction

Support vector machine (SVM) is a supervised learning method to make decision boundaries for classification problems. For two-class problems, given a training dataset $D = \{(\boldsymbol{x_i}, y_i)\}_{i=1}^{n}$, where $\boldsymbol{x_i} \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, the decision boundary for the problem is a hyperplane $H = \{\boldsymbol{x} : \boldsymbol{w}^\top \boldsymbol{x} - b = 0\}$. To classify all the points correctly, we need to make all the points from the positive side of the hyperplane be classified as $+1$ and all the points from the negative side be classified as -1.

Hard-Margin SVMs

Generally, there are two types of SVM: hard-margin SVMs and soft-margin SVMs. The mission of hard-margin SVMs is to maximize the margin, which is defined as the minimum

distance from the decision boundary to the training points. Set $\boldsymbol{w}$ as a vector which is perpendicular to H, for any point $\boldsymbol{z}$, the distance from $\boldsymbol{z}$ to H is determined by a scaled size of $\boldsymbol{w}$. If $\boldsymbol{x_0}$ is an arbitrary point on the hyperplane H, then the distance D from $\boldsymbol{z}$ to H is the length of the projection from $\boldsymbol{z} - \boldsymbol{x_0}$ to $\boldsymbol{w}$:

$$D = \frac{|\boldsymbol{w}^\top(\boldsymbol{z} - \boldsymbol{x_0})|}{\|\boldsymbol{w}\|_2} = \frac{|\boldsymbol{w}^\top \boldsymbol{z} - b|}{\|\boldsymbol{w}\|_2}.$$

If d is the margin size, for any training point $\boldsymbol{x_i}$,

$$y_i \frac{\boldsymbol{w}^\top \boldsymbol{x_i} - b}{\|\boldsymbol{w}\|_2} \geq d.$$

Our optimization problem is:

$$\begin{aligned} &\max_{d,\boldsymbol{w},b} \quad d, \\ &\text{s.t.} \quad \begin{cases} y_i \frac{\boldsymbol{w}^\top \boldsymbol{x_i} - b}{\|\boldsymbol{w}\|_2} \geq d \quad \forall i, \\ d \geq 0. \end{cases} \end{aligned} \tag{2.1}$$

Support vectors are the points whose distance to the hyperplane is equal to $\frac{d}{2}$. The current optimization formulation is not unique for a combination of $\boldsymbol{w}$ and b. Let $d = \frac{1}{\|\boldsymbol{w}\|_2}$, we change this formulation to the standard formulation:

$$\begin{aligned} &\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2, \\ &\text{s.t.} \quad y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b) \geq 1 \quad \forall i. \end{aligned} \tag{2.2}$$

Soft-Margin SVMs

The hard-margin SVMs are only available to the data which is linearly separable due to the sensitivity to outliers. However, soft-margin SVMs allow some points to violate the margin.

The changing part to hard-margin SVMs is the slack variables ξ in the constraints:

$$
\begin{aligned}
& y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b) \geq 1 - \xi_i \quad \forall i, \\
& \xi_i \geq 0 \quad \forall i.
\end{aligned}
$$

In case the slack variables become very large to make $\boldsymbol{w}$ meaningless, we need to penalize the slacks by setting the soft-margin SVM optimization problem as:

$$
\begin{aligned}
& \min_{\boldsymbol{w},b,\xi_i} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i, \\
& \text{s.t.} \quad \begin{cases} y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b) \geq 1 - \xi_i \quad \forall i, \\ \xi_i \geq 0 \quad \forall i. \end{cases}
\end{aligned}
\tag{2.3}
$$

When the hyperparameter C is relatively small, it will maximize the margin and may cause the underfitting problem. When the hyperparameter C is relatively large, it will keep the slack variables small or even zero, and may cause the overfitting problem. The large C is more sensitive to outliers compared to the small C.

Loss Function

Changing the order for the constraints of soft-margin SVM optimization, we can get:

$$
\xi_i \geq \max(1 - y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b), 0).
$$

Consider that our goal is to minimize the objective function with ξ_i, therefore, the optimization problem can be written as:

$$
\begin{aligned}
& \min_{\boldsymbol{w},b,\xi_i} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i, \\
& \text{s.t.} \quad \xi_i = \max(1 - y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b), 0) \quad \forall i.
\end{aligned}
\tag{2.4}
$$

Now we can put the constraints to the objective function:

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{n}\max(1 - y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b), 0).$$

Let $\lambda = \frac{1}{2Cn}$, the objective function of the optimization problem can be transformed to this form:

$$\min_{\boldsymbol{w},b} \quad \lambda\|\boldsymbol{w}\|_2^2 + \frac{1}{n}\sum_{i=1}^{n} L(y_i, \boldsymbol{w}^\top \boldsymbol{x_i} - b), \tag{2.5}$$

where L is the hinge loss:

$$L = L_{hinge}(y_i, \boldsymbol{w}^\top \boldsymbol{x_i} - b) = \max(1 - y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b), 0).$$

Duality

The standard form of a convex optimization problem is[18]:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f_0(\boldsymbol{x}), \\ \text{s.t.} \quad & \begin{cases} f_i(\boldsymbol{x}) \le 0 \quad i = 1, 2, \cdots, m, \\ g_j(\boldsymbol{x}) = 0 \quad j = 1, 2, \cdots, n. \end{cases} \end{aligned} \tag{2.6}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the optimization variable; the objective function $f_0(\boldsymbol{x})$ and inequality constraint functions $f_i(\boldsymbol{x})$ are convex functions; the equality constraint functions $g_j(\boldsymbol{x})$ are affine transformations.

Like what we did to the loss function of SVMs, we can transform the constrained problem to an unconstrained problem:

$$\min_{\boldsymbol{x}} L(\boldsymbol{x}),$$

where

$$L(\boldsymbol{x}) \quad = \quad \begin{cases} f_0(\boldsymbol{x}) & \text{if } f_i(\boldsymbol{x}) \leq 0,\ \forall i \in \{1, 2, \cdots, m\} \text{ and } g_j(\boldsymbol{x}) = 0,\ \forall j \in \{1, 2, \cdots, n\}, \\ \infty & \text{otherwise.} \end{cases}$$

To fix the uncertainty generated by $L(\boldsymbol{x})$, we introduce the dual variables λ_i for inequality constraints and v_j for equality constraints. The new optimization problem is:

$$p^* = \min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda} \geq \boldsymbol{0}, \boldsymbol{v}} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{v}), \tag{2.7}$$

where $L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{v}) = f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i f_i(\boldsymbol{x}) + \sum_{j=1}^{n} v_j g_j(\boldsymbol{x})$ is the Lagrangian function.

The form (2.7) is the primal problem. If $g_j(\boldsymbol{x}) > 0$, then for the inner maximum problem, take v_j as a large number approaching ∞; if $g_j(\boldsymbol{x}) < 0$, then for the inner maximum problem, take v_j as a small number approaching $-\infty$. If $f_i(\boldsymbol{x}) > 0$, then for the inner maximum problem, take λ_i as a large number approaching ∞. All of these results would be vetoed by the outer minimum problem. Therefore, $\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda} \geq \boldsymbol{0}, \boldsymbol{v}} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{v})$ is equivalent to $\min_{\boldsymbol{x}} L(\boldsymbol{x})$, which is equivalent to the original problem (2.6).

The dual of this optimization problem is:

$$d^* = \max_{\boldsymbol{\lambda} \geq \boldsymbol{0}, \boldsymbol{v}} \min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{v}). \tag{2.8}$$

Weak duality is $p^* \geq d^*$ and strong duality is $p^* = d^*$.

KKT Conditions

The Karush-Kuhn-Tucker (KKT) conditions are:

(1) Primal feasibility:

$$f_i(\boldsymbol{x}) \leq 0, \forall i \in \{1, 2, \cdots, m\},$$

$$g_j(\boldsymbol{x}) = 0, \forall j \in \{1, 2, \cdots, n\}.$$

(2) Dual feasibility:

$$\lambda_i \geq 0, \forall i \in \{1, 2, \cdots, m\}.$$

(3) Complementary Slackness:

$$\lambda_i f_i(\boldsymbol{x}) = 0, \forall i \in \{1, 2, \cdots, m\}.$$

(4) Stationarity:

$$\nabla_{\boldsymbol{x}} f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i \nabla_{\boldsymbol{x}} f_i(\boldsymbol{x}) + \sum_{j=1}^{n} v_j \nabla_{\boldsymbol{x}} g_j(\boldsymbol{x}) = 0.$$

The following theorems reveal the relationship between the strong duality and KKT conditions[19,20]:

Theorem 2.1. *If $\boldsymbol{x}^*$ is the primal solution and $\boldsymbol{\lambda}^*, \boldsymbol{v}^*$ are the dual solutions with strong duality holds, then $\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{v}^*$ satisfy the KKT conditions.*

Theorem 2.2. *If $\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{v}^*$ satisfy the KKT conditions and the primal problem is convex, then $\boldsymbol{x}^*$ is the primal solution and $\boldsymbol{\lambda}^*, \boldsymbol{v}^*$ are the dual solutions with strong duality holds.*

Dual SVMs

We can write the soft-margin SVM optimization problem (2.3) in the standard form of the convex optimization problem:

$$\begin{aligned} \min_{\boldsymbol{w}, b, \xi_i} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n} \xi_i, \\ \text{s.t.} \quad & \begin{cases} 1 - \xi_i - y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b) \leq 0 \quad \forall i, \\ -\xi_i \leq 0 \quad \forall i. \end{cases} \end{aligned} \tag{2.9}$$

The Lagrangian function for the SVM problem is:

$$\begin{aligned} L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i(1-\xi_i - y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b)) + \sum_{i=1}^{n}\beta_i(-\xi_i) \\ &= \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n}\alpha_i y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b) + \sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n}(C-\alpha_i-\beta_i)\xi_i. \end{aligned} \tag{2.10}$$

Let $h(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \min_{\boldsymbol{w},b,\boldsymbol{\xi}} L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})$, then the dual of the Lagrangian function is:

$$\max_{\boldsymbol{\alpha},\boldsymbol{\beta}\geq \mathbf{0}} h(\boldsymbol{\alpha}, \boldsymbol{\beta}). \tag{2.11}$$

We know the primal problem of SVM is convex, to apply Theorem 2.2, we need to verify the KKT conditions for the optimization problem (2.9):

(1) Primal feasibility: This is satisfied by default of constraints.

(2) Dual feasibility:

$$\alpha_i^* \geq 0, \beta_i^* \geq 0.$$

(3) Complementary Slackness:

$$\alpha_i^*(1-\xi_i^* - y_i(\boldsymbol{w^*}^\top \boldsymbol{x_i} - b^*)) = \beta_i^*(-\xi_i^*) = 0.$$

(4) Stationarity:

$$\frac{\partial L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial w_i} = \frac{\partial L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial b} = \frac{\partial L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \xi_i} = 0.$$

To satisfy the KKT conditions, the results are:

$$\begin{cases} \boldsymbol{w^*} = \sum_{i=1}^{n}\alpha_i^* y_i \boldsymbol{x_i}, \\ \sum_{i=1}^{n}\alpha_i^* y_i = 0, \\ C - \alpha_i^* - \beta_i^* = 0. \end{cases} \tag{2.12}$$

Use optimal primal values (2.12) to simplify the Lagrangian function (2.10):

$$\begin{aligned} L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha^*}, \boldsymbol{\beta^*}) &= \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n} \alpha_i^* y_i(\boldsymbol{w}^\top \boldsymbol{x_i} - b) + \sum_{i=1}^{n} \alpha_i^* + \sum_{i=1}^{n}(C - \alpha_i^* - \beta_i^*)\xi_i \\ &= \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n} \alpha_i^* y_i(\boldsymbol{w}^\top \boldsymbol{x_i}) + \sum_{i=1}^{n} \alpha_i^*. \end{aligned} \tag{2.13}$$

From (2.12) and (2.13), we can deduce that:

$$\begin{aligned} h(\boldsymbol{\alpha^*}, \boldsymbol{\beta^*}) &= \min_{\boldsymbol{w}, b, \boldsymbol{\xi}} L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha^*}, \boldsymbol{\beta^*}) \\ &= L(\boldsymbol{w^*}, b^*, \boldsymbol{\xi^*}, \boldsymbol{\alpha^*}, \boldsymbol{\beta^*}) \\ &= \frac{1}{2}\boldsymbol{w^*}^\top \boldsymbol{w^*} - \sum_{i=1}^{n} \alpha_i^* y_i((\sum_{i=1}^{n} \alpha_i^* y_i \boldsymbol{x_i})^\top \boldsymbol{x_i}) + \sum_{i=1}^{n} \alpha_i^* \\ &= \sum_{i=1}^{n} \alpha_i^* + \frac{1}{2}(\sum_{i=1}^{n} \alpha_i^* y_i \boldsymbol{x_i})^\top (\sum_{j=1}^{n} \alpha_j^* y_j \boldsymbol{x_j}) - (\sum_{i=1}^{n} \alpha_i^* y_i \boldsymbol{x_i})^\top (\sum_{j=1}^{n} \alpha_j^* y_j \boldsymbol{x_j}) \\ &= \boldsymbol{\alpha^*}^\top \mathbf{1} - \frac{1}{2}\boldsymbol{\alpha^*}^\top \boldsymbol{Q} \boldsymbol{\alpha^*}, \end{aligned} \tag{2.14}$$

where $Q_{ij} = y_i y_j(\boldsymbol{x_i}^\top \boldsymbol{x_j})$.

From the dual feasibility and stationarity, we know $\sum_{i=1}^{n} \alpha_i^* y_i = 0$ and $0 \leq \alpha_i^* \leq C$.

Therefore, from (2.11) and (2.14), the dual form of the SVM optimization problem is:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^\top \boldsymbol{Q} \boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \mathbf{1}, \\ \text{s.t.} \quad & \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \quad \forall i, \\ 0 \leq \alpha_i \leq C \quad \forall i. \end{cases} \end{aligned} \tag{2.15}$$

Prediction for test data

From the Complementary Slackness and Stationarity conditions for the problem (2.9), we have the following results:

$$C - \alpha_i^* - \beta_i^* = 0,$$
$$\alpha_i^*(1 - \xi_i^* - y_i(\boldsymbol{w^*}^\top \boldsymbol{x_i} - b^*)) = \beta_i^*(-\xi_i^*) = 0.$$

(1) If $\alpha_i^* = 0$, then $\beta_i^* = C$, $\xi_i^* = 0$. There is no slack for point i, which means point i lies on or outside the margin;

(2) If $0 < \alpha_i^* < C$, then $\beta_i^* \neq 0$, $\xi_i^* = 0$, $y_i(\boldsymbol{w^*}^\top \boldsymbol{x_i} - b^*) = 1$. In this case, point i lies on the margin;

(3) If $\alpha_i^* = C$, then $\beta_i^* = 0$, $\xi_i^* = 1 - y_i(\boldsymbol{w^*}^\top \boldsymbol{x_i} - b^*) \geq 0$. $y_i(\boldsymbol{w^*}^\top \boldsymbol{x_i} - b^*) \leq 1$, which means point i lies on or inside the margin.

Based on (2.12) and the discussion above, we have the following results:

$$\begin{cases} \boldsymbol{w^*} = \sum_{i=1}^n \alpha_i^* y_i \boldsymbol{x_i}, \\ y_i(\boldsymbol{w^*}^\top \boldsymbol{x_i} - b^*) = 1, \text{ if } 0 < \alpha_i^* < C, \\ \xi_i^* = \begin{cases} 1 - y_i(\boldsymbol{w^*}^\top \boldsymbol{x_i} - b^*), \text{ if } \alpha_i^* = C, \\ 0, \text{ otherwise.} \end{cases} \end{cases} \tag{2.16}$$

Therefore, the prediction strategy for test data is:

$$\boldsymbol{w}^\top \boldsymbol{x} - b = \sum_{i=1}^n \alpha_i y_i \boldsymbol{x_i}^\top \boldsymbol{x} - b. \tag{2.17}$$

The predicted class is based on the sign of the outcome.

SVM for Multi-classification

One of the popular methods for multi-class SVM is the one-vs-one (OvO) strategy. If there are $N(N > 2)$ classes, the OvO strategy will conduct two steps to classify:

(1) Construct the binary classifier for each of two classes, $\frac{N(N-1)}{2}$ classifiers in total.

(2) Run the test data through all the binary classifiers. The class that wins the most duels is the one assigned to the test data point.

2.1.2 Recursive Feature Elimination

Recursive feature elimination (RFE) is an iterative procedure to select proper features and reduce the demension by getting a subset of all features. The iterative procedure is as follow[21]:

(1) Train the classifier;

(2) Compute the ranking criterion for all features;

(3) Remove the feature with the smallest ranking criterion.

RFE intends to remove several features at a time, which produces a feature subset ranking. Combining SVM and RFE to select proper features, we have the algorithm on the next page.

There are some notices for the algorithm:

(1) The stopping criterion for the algorithm is the model performance evaluated by the cross-validation score of each step. When the cross-validation score of current sorted features is not improved, the loop of RFE will stop and give the best-ranked features.

(2) The **Accuracy** in the algorithm is the classification accuracy with cross-validation data by the SVM model.

(3) From (2.1-2.3), SVM intends that the hyperplane will maximize the margin between different classes. $\boldsymbol{w}$ is the coefficient of the hyperplane. The larger the absolute value of a coefficient, the more significant the impact a slight change in that feature will have on the outcome, hence, such a feature is deemed more important. From (2.12), we know $\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x_i}$.

Algorithm 1: SVM-RFE

Input : $\boldsymbol{X}_{train} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_k}]^\top$, $y_{train} = [y_1, y_2, \cdots, y_k]^\top$.
Classifier : $\mathbf{SVM}_{train}(\boldsymbol{X}, y) = \min_{\boldsymbol{\alpha}} \frac{1}{2}, \boldsymbol{\alpha}^\top \boldsymbol{Q}\boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \mathbf{1}$,
subject to $\sum_{i=1}^{n} \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$.
$\boldsymbol{s} = [1, 2, \cdots, n]$;
$\boldsymbol{r} = []$;
$\boldsymbol{X} = \boldsymbol{X}_{train}$;
$score_{cv}^{(1)} = 0, score_{cv}^{(2)} = 1$;
while $score_{cv}^{(2)} > score_{cv}^{(1)}$ **do**
  $\boldsymbol{X} = \boldsymbol{X}(:, \boldsymbol{s})$;
  $score_{cv}^{(1)} = \mathbf{Accuracy}(\boldsymbol{X}, y)$;
  $\boldsymbol{\alpha} = \mathbf{SVM}_{train}(\boldsymbol{X}, y)$;
  $\boldsymbol{w} = \sum_i \alpha_i y_i \boldsymbol{x_i}$;
  $f = \arg\min(\|\boldsymbol{w}\|^2)$;
  $\boldsymbol{r} = [\boldsymbol{s}(f), \boldsymbol{r}]$;
  $\boldsymbol{s} = \boldsymbol{s}(1 : f-1, f+1 : \text{length}(\boldsymbol{s}))$;
  $score_{cv}^{(2)} = \mathbf{Accuracy}(\boldsymbol{X}(:, \boldsymbol{s}), y)$
end
$\boldsymbol{r} = [\boldsymbol{s}, \boldsymbol{r}]$;
Output : $\boldsymbol{r}$

2.2 Multilayer Perceptron Neural Network

2.2.1 Introduction

Multilayer perceptron (MLP) is a class of feedforward neural networks. A neural network has three basic types of layers: the input layer, the hidden layer, and the output layer. Each layer has multiple perceptrons (also known as nodes or neurons) to process weighted information. MLP is a fully connected neural network, which means every node from the previous layer is connected to every node in the current layer. For a classification problem, the number of nodes in the input layer depends on the number of features of the dataset; the number of nodes in the output layer depends on the number of classes of the dataset.

Each node after the input layer receives information by computing a weighted and biased sum of its inputs and then applying the activation function to deliver the non-linearity. Let $\boldsymbol{x}$ represent the information of all nodes from the previous layer. If $\boldsymbol{w_i}$ and b_i are the weights

and bias of all connections between one node i and its inputs, and σ_i is the activation function of i, the mapping process to node i is:

$$\boldsymbol{x} \mapsto \sigma_i(\boldsymbol{w_i}^\top \boldsymbol{x} - b_i).$$

Generally, if the neural networks have $L+1$ layers in total, the number of layers in order are denoted by $n_0, n_1, \cdots, n_L$. The weights for every nodes in layer i is a matrix $\boldsymbol{W_i} \in \mathbb{R}^{n_i \times n_{i-1}}$ and bias is $\boldsymbol{b_i} \in \mathbb{R}^{n_i}$. The activation functions for layer i are $\boldsymbol{\sigma_i} : \mathbb{R}^{n_i} \mapsto \mathbb{R}^{n_i}$, then the mapping process to layer i is:

$$\boldsymbol{x} \mapsto \boldsymbol{\sigma_i}(\boldsymbol{W_i x} - \boldsymbol{b_i}).$$

Let $\boldsymbol{x}$ be the information of the input layer, then the neural networks can be represented by:

$$\boldsymbol{x} \mapsto \boldsymbol{\sigma_L}(\boldsymbol{W_L \sigma_{L-1}}(\cdots \boldsymbol{\sigma_2}(\boldsymbol{W_2 \sigma_1}(\boldsymbol{W_1 x} - \boldsymbol{b_1}) - \boldsymbol{b_2}) \cdots) - \boldsymbol{b_L}). \tag{2.18}$$

In this thesis, the activation function is the same map for a layer. For instance, the layer i has activation functions:

$$\boldsymbol{\sigma_i}(\boldsymbol{x}) = [\sigma_i(x_1), \sigma_i(x_2), \cdots, \sigma_i(x_{n_i})]^\top.$$

Activation Function

The activation function applied on hidden layers is the rectified linear unit (ReLU) function:

$$\sigma(x) = \max\{0, x\}.$$

If N is the number of classes, given the input vector $\boldsymbol{z} = [z_1, z_2, \cdots, z_N]$ for the output layer, the softmax function $\sigma(z)$ outputs a new vector $\boldsymbol{y} = [y_1, y_2, \cdots, y_N]$ where each element

y_i is computed as follows:

$$y_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}.$$

Loss Function

The loss function applied to the output layer is cross entropy loss or log loss. The binary cross-entropy loss is used in a two-class case. For the node i, let $y_i \in \{0, 1\}$ denote the true labels, and $\hat{y_i}$ denotes the predicted probability after the softmax function in the output layer. The binary cross-entropy loss is calculated by the average log loss:

$$\mathcal{L}(y) = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log \hat{y_i} + (1 - y_i)\log(1 - \hat{y_i})).$$

Next, we set up the categorical cross-entropy loss for a multi-classification problem. Considering the sample size is different for each class, let w_i denotes the weight for the node i and $\hat{y_i}$ denotes the predicted probability after the softmax function in the output layer. The categorical cross-entropy loss function is defined as:

$$\mathcal{L}(y) = -\sum_{i=1}^{N} \frac{w_i}{\sum_{j=1}^{N} w_j} \log \hat{y_i}.$$

Gradient Descent

Gradient descent is an optimization strategy to take small steps to find the local minimum in the direction of the steepest descent of the objective function in the optimization problem. If the initialization point is $\boldsymbol{\theta}^{(0)}$, with a learning rate α, then the iteration in timestep $t + 1$ is:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha \nabla f(\boldsymbol{\theta}^{(t)}),$$

α is the adaptive stepsize which is decided before the optimization. If α is too large, then the

algorithm may be divergent; if α is too small, then the algorithm may converge too slowly.

With the basic idea of gradient descent, we can explore batch gradient descent, mini-batch gradient descent, and stochastic gradient descent. The process of using the entire training set in one epoch is called batch gradient descent. For each training example, the forward propagation through the neural network generates a gradient to optimize the parameters. If the loss f can be decomposed as $f(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{\theta})$, we can average all the gradients to get the batch gradient:

$$\nabla f(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta}).$$

Compared to the batch gradient descent, the mini-batch gradient descent is more computationally efficient. Let the gradient $\nabla G(\boldsymbol{\theta}) = \frac{1}{k}\sum_{i=1}^{k} \nabla f_i(\boldsymbol{\theta})$, where $k < n$. Notice that $\nabla f(\boldsymbol{\theta})$ is the expectation of the stochastic gradient $\nabla G(\boldsymbol{\theta})$. Therefore, the stochastic gradient is an unbiased estimate of the true gradient. When $k = 1$, the special case of the mini-batch gradient descent is called stochastic gradient descent (SGD): $\nabla G(\boldsymbol{\theta}) = \nabla f_i(\boldsymbol{\theta})$, where $i \in \{1, 2, \cdots, n\}$.

The stochastic gradient update rule is:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha \nabla G(\boldsymbol{\theta}^{(t)}).$$

Adam Optimizer

The "rolling downhill" process of gradient descent can be slow and prevented by plateau parts of the objective function. The approach to the optima is usually tortuous if we only choose a few dimensions each time. Momentum is a method that combines the previous approach in the current timestep. The idea of the update rule of momentum is:

$$g_t = \beta g_{t-1} + \nabla f(\boldsymbol{\theta}^{(t)}),$$
$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha g_t.$$

where g_t is the velocity term at time t, β is the decay rate, α is the learning rate.

Momentum leads to faster convergence than standard SGD and helps the model overcome small local minima during training. On the other hand, SGD can lead to suboptimal updates where the learning rate might become too small. This issue can be improved by root mean square propagation (RMSProp)[22]. RMSProp is a method that adjusts the learning rate for each of the weights in the model. The update rule for RMSProp is:

$$
\begin{aligned}
S_t &= \beta S_{t-1} + (1-\beta)(\nabla f(\boldsymbol{\theta}^{(t)}))^2, \\
\boldsymbol{\theta}^{(t+1)} &= \boldsymbol{\theta}^{(t)} - \alpha \frac{\nabla f(\boldsymbol{\theta}^{(t)})}{\sqrt{S_t}}.
\end{aligned}
$$

where S_t is the accumulated square gradient at time t that is greater than 0, β is the decay rate, α is the learning rate.

RMSProp can take larger steps when the gradients are small, and smaller steps when the gradients are large, thus speeding up the learning process in flat regions and slowing it down in steep regions. Therefore, we use adaptive moment estimation (Adam) optimizer[23] to combine the advantages of both momentum and RMSProp for the optimization strategy of the backpropagation of the neural network. Adam can compute the exponentially decaying average of past gradients and past squared gradients. The update rule for Adam is:

$$
\begin{aligned}
m_t &= \beta_1 m_{t-1} + (1-\beta_1)\nabla f(\boldsymbol{\theta}^{(t)}), \\
v_t &= \beta_2 v_{t-1} + (1-\beta_2)(\nabla f(\boldsymbol{\theta}^{(t)}))^2, \\
\hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \\
\hat{v}_t &= \frac{v_t}{1-\beta_2^t}, \\
\boldsymbol{\theta}^{(t+1)} &= \boldsymbol{\theta}^{(t)} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},
\end{aligned}
\tag{2.19}
$$

where m_t is the first-moment estimate corresponding to momentum and v_t is the second-moment estimate corresponding to RMSProp. β_1 and β_2 are the decay rates corresponding to the first-moment estimate m_t and the second-moment estimate v_t. $\hat{m}_t$ and $\hat{v}_t$ are the

correction for m_t and v_t. α is the learning rate. ϵ is a small constant to prevent division by zero and to provide numerical stability.

Backpropagation

If we start from the hidden layers, we have L layers including the output layer. Let $\boldsymbol{z^{(i)}}$ be the linear mapping result in the layer i, $\boldsymbol{a^{(i)}}$ be the non-linear mapping (activation function) result in the layer i:

$$\begin{aligned}\boldsymbol{z^{(i)}} &= \boldsymbol{W^{(i)}}\boldsymbol{a^{(i-1)}} - \boldsymbol{b^{(i)}},\\ \boldsymbol{a^{(i)}} &= \boldsymbol{\sigma^{(i)}}(\boldsymbol{z^{(i)}}),\end{aligned}$$

where $i \in \{1, 2, \cdots, L\}$, $\boldsymbol{a^{(0)}}$ is the input layer information $\boldsymbol{x}$. The loss function is $\mathcal{L}(\boldsymbol{a^{(L)}})$.

Apply the chain rule for the backpropagation:

$$\begin{aligned}\frac{\partial \mathcal{L}(\boldsymbol{a^{(L)}})}{\partial \boldsymbol{W^{(i)}}} &= \frac{\partial \boldsymbol{z^{(i)}}}{\partial \boldsymbol{W^{(i)}}}\frac{\partial \boldsymbol{a^{(i)}}}{\partial \boldsymbol{z^{(i)}}}\frac{\partial \boldsymbol{z^{(i+1)}}}{\partial \boldsymbol{a^{(i)}}}\cdots\cdots\frac{\partial \boldsymbol{z^{(L)}}}{\partial \boldsymbol{a^{(L-1)}}}\frac{\partial \boldsymbol{a^{(L)}}}{\partial \boldsymbol{z^{(L)}}}\frac{\partial \mathcal{L}(\boldsymbol{a^{(L)}})}{\partial \boldsymbol{a^{(L)}}},\\ \frac{\partial \mathcal{L}(\boldsymbol{a^{(L)}})}{\partial \boldsymbol{b^{(i)}}} &= \frac{\partial \boldsymbol{z^{(i)}}}{\partial \boldsymbol{b^{(i)}}}\frac{\partial \boldsymbol{a^{(i)}}}{\partial \boldsymbol{z^{(i)}}}\frac{\partial \boldsymbol{z^{(i+1)}}}{\partial \boldsymbol{a^{(i)}}}\cdots\cdots\frac{\partial \boldsymbol{z^{(L)}}}{\partial \boldsymbol{a^{(L-1)}}}\frac{\partial \boldsymbol{a^{(L)}}}{\partial \boldsymbol{z^{(L)}}}\frac{\partial \mathcal{L}(\boldsymbol{a^{(L)}})}{\partial \boldsymbol{a^{(L)}}}.\end{aligned}$$

For this specific neural network, the activation function in the hidden layers is the ReLU function. For the jth node in the hidden layer i:

$$a_j^{(i)} = \sigma_j^{(i)}(z_j^{(i)}) = \max\{0, z_j^{(i)}\}.$$

The activation function in the output layer is softmax. For the jth node in the output layer L:

$$a_j^{(L)} = \sigma_j^{(L)}(z_j^{(L)}) = \frac{e^{z_j^{(L)}}}{\sum_{m=1}^{N} e^{z_m^{(L)}}}.$$

The loss function is cross-entropy loss:

$$\mathcal{L}(\boldsymbol{a^{(L)}}) = -\sum_{j=1}^{N} \frac{w_j}{\sum_{m=1}^{N} w_m} \log \hat{a_j}^{(L)}.$$

More specifically, for the node i in a certain layer s: $z_i^{(s)} = \sum_k W_{ik}^{(s)} a_k^{(s-1)} - b_i^{(s)}$, where k is the dimension of $a^{(s-1)}$. We have:

$$\begin{aligned}
\frac{\partial z_i^{(s)}}{\partial W_{jk}^{(s)}} &= \begin{cases} a_k^{(s-1)}, & \text{if } j = i, \\ 0, & \text{if } j \neq i, \end{cases} \\
\frac{\partial z_i^{(s)}}{\partial b_j^{(s)}} &= \begin{cases} -1, & \text{if } j = i, \\ 0, & \text{if } j \neq i, \end{cases} \\
\frac{\partial z_i^{(s)}}{\partial a_j^{(s-1)}} &= W_{ij}^{(s)}.
\end{aligned}$$

2.2.2 Bayesian Optimization

Bayesian optimization (BayesOpt) is a class of optimization methods for black-box derivative-free global optimization[24]. The goal of BayesOpt is simply maximizing the objective function f. The objective function f is continuous and lacks linearity. The dimension of input data for BayesOpt has to be less than 20. The general idea of BayesOpt is to place a Gaussian Process prior on f and use training data to update the posterior probability distribution on f. Use the current posterior distribution to compute the acquisition function and decide where the next data point samples from. Continue the process until reaching the specified number of iterations.

Surrogate Model

The most common surrogate model for BayesOpt is Gaussian Process (GP). GP is a Bayesian approach for modeling functions. For a set of data points $\boldsymbol{x}$: $\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_n} \in \mathbb{R}^d$. The corresponding function values over f are $f(\boldsymbol{x_1}), f(\boldsymbol{x_2}), \cdots, f(\boldsymbol{x_n})$. For GP, the most important part is to choose the proper mean function and covariance or kernel function. Usually, we

set our mean function as a constant value: $m(\boldsymbol{x}) = \mu$. The common kernel function used is the Màtern kernel[25]:

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \frac{2^{1-v}}{\Gamma(v)}(\sqrt{2v}\frac{\|\boldsymbol{x} - \boldsymbol{x'}\|}{\rho})^v K_v(\sqrt{2v}\frac{\|\boldsymbol{x} - \boldsymbol{x'}\|}{\rho}),$$

where $\|\boldsymbol{x} - \boldsymbol{x'}\|$ is the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{x'}$: $\|\boldsymbol{x} - \boldsymbol{x'}\| = \sqrt{\sum_{i=1}^{n}(x_i - x_i')^2}$. v is a smoothness parameter and ρ is a length scale parameter. K_v is a modified Bessel function of the second kind[26]. Γ is the Gamma function.

For this thesis, the smoothness parameter $v = \frac{5}{2}$, which gives

$$k(\boldsymbol{x}, \boldsymbol{x'}) = (1 + \frac{\sqrt{5}\|\boldsymbol{x} - \boldsymbol{x'}\|}{\rho} + \frac{5\|\boldsymbol{x} - \boldsymbol{x'}\|^2}{3\rho^2}) \exp(-\frac{\sqrt{5}\|\boldsymbol{x} - \boldsymbol{x'}\|}{\rho}).$$

Let $\boldsymbol{X} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_n}]$, $y = f(\boldsymbol{x}) + \epsilon$, where $\epsilon \sim N(0, \sigma_n^2)$. The prior distribution on the function value $f(\boldsymbol{x})$ is

$$f(\boldsymbol{x}) \sim N(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x'}) + \sigma_n^2 \boldsymbol{I}).$$

If we have a new data $(\boldsymbol{x'}, \boldsymbol{y'})$, then the posterior distribution[25] is

$$\boldsymbol{y'}|\boldsymbol{X}, \boldsymbol{y} \sim N(\mu(\boldsymbol{x'}), \sum(\boldsymbol{x'})), \tag{2.20}$$

where

$$\begin{aligned} \mu(\boldsymbol{x'}) &= k(\boldsymbol{x'}, \boldsymbol{X})[k(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}]^{-1}(\boldsymbol{y} - m(\boldsymbol{X})), \\ \sum(\boldsymbol{x'}) &= k(\boldsymbol{x'}, \boldsymbol{x'}) - k(\boldsymbol{x'}, \boldsymbol{X})[k(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}]^{-1} k(\boldsymbol{x'}, \boldsymbol{X}). \end{aligned} \tag{2.21}$$

However, two parameters in the prior distribution are still left to choose. There are two popular methods to choose these parameters. Let $\boldsymbol{D} = [\boldsymbol{X}, \boldsymbol{y}]$, the first method is to estimate $\boldsymbol{\theta} = (\mu, \rho)$ with the maximum a posteriori (MAP) estimate:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} P(\boldsymbol{D}|\boldsymbol{\theta})P(\boldsymbol{\theta}).$$

The second method is the fully Bayesian approach:

$$\begin{aligned} P(f(\boldsymbol{x'})|\boldsymbol{D}) &= \int P(f(\boldsymbol{x'})|\boldsymbol{D},\boldsymbol{\theta})P(\boldsymbol{\theta}|\boldsymbol{D})d\boldsymbol{\theta} \\ &\approx \frac{1}{J}\sum_{j=1}^{J} P(f(\boldsymbol{x'})|\boldsymbol{D},\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_j), \end{aligned}$$

where $\hat{\boldsymbol{\theta}}_j$ are sampled from $P(\boldsymbol{\theta}|\boldsymbol{D})$ via an MCMC method.

Acquisition Function

The most common acquisition function is the expected improvement. Assume the observations f are noiseless, our optimal goal is to find the largest observed value from what we evaluated previously. Let $f_n^* = \max_i f(\boldsymbol{x_i})$ be the optimal value, where $i = 1, 2, \cdots, n$ represents the previous evaluated points. If our new sample point is $\boldsymbol{x}$, we wish our improvement $f(\boldsymbol{x}) - f_n^* \geq 0$. Write $\boldsymbol{x}^+ = \max\{\boldsymbol{x}, 0\}$, define the expected improvement as:

$$EI_n(\boldsymbol{x}) := E_n[[f(\boldsymbol{x}) - f_n^*]^+],$$

where $f(\boldsymbol{x})$ is the posterior distribution given by (2.20).

The expected improvement can be solved as a closed form[27,28]:

$$EI_n(\boldsymbol{x}) = [\Delta_n(\boldsymbol{x})]^+ + \sum_{\boldsymbol{n}}(\boldsymbol{x})\phi(\frac{\Delta_n(\boldsymbol{x})}{\sum_{\boldsymbol{n}}(\boldsymbol{x})}) - |\Delta_n(\boldsymbol{x})|\Phi(\frac{\Delta_n(\boldsymbol{x})}{\sum_{\boldsymbol{n}}(\boldsymbol{x})}),$$

where $\Delta_n(\boldsymbol{x}) = \mu_n(\boldsymbol{x}) - f_n^*$. $\mu_n(\boldsymbol{x})$ and $\sum_{\boldsymbol{n}}(\boldsymbol{x})$ are the expressions from (2.21). $\phi(\cdot)$ and $\Phi(\cdot)$ are the standard normal density and distribution function.

The next point is sampled with the largest expected improvement[28,29]:

$$\boldsymbol{x_{n+1}} = \arg\max_{\boldsymbol{x}} EI_n(\boldsymbol{x}).$$

2.2.3 SHapley Additive exPlanations

SHapley Additive exPlanations (SHAP) is an additive feature attribution method that provides a more detailed and intuitive explanation of black-box model predictions. The method ensures that the sum of all the feature importance values equals the difference between the prediction for the current input and the expected prediction over all inputs. SHAP is based on the concept of Shapley values from cooperative game theory.

Assume f is the prediction model, the Shapley value for the i-th feature is the average marginal contribution of the feature across all possible coalitions[30]:

$$\phi_i(\boldsymbol{x}) = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(M - |S| - 1)!}{M!} [g(S \cup \{i\}) - g(S)],$$

where $|S|$ is the number of elements in S. M is the total number of features. F is the set of all features. S is a coalition of features that is a subset of F. $\boldsymbol{x} \in \mathbb{R}^M$ is the input data point to compute SHAP values. $g(S)$ is the value function that measures the change in the model's prediction when the features in coalition S are present.

However, the computational cost for iterating over all 2^M coalitions of features is expensive. It's better to resort to approximation methods when we deal with large datasets. KernelSHAP is a method to approximate SHAP values. It creates a binary representation for the presence or absence of features, a coalition denoted by $\boldsymbol{z}$. The ith feature of $\boldsymbol{z}$ is z_i: 1 for the present feature of the dataset and 0 for the absent feature of the dataset. To map the presence or absence of features to the output of the model f, we use a mapping function for instance $\boldsymbol{x}$:

$$h_{\boldsymbol{x}}(\boldsymbol{z}) = f(\boldsymbol{x} \odot \boldsymbol{z} + \boldsymbol{x'} \odot (\mathbf{1} - \boldsymbol{z})),$$

where $\boldsymbol{x'}$ is a synthetic instance where the value of some features is replaced by their baseline values. $\odot$ represents element-wise multiplication.

As an additive feature attribution method, KernelSHAP gives a linear function of binary

variables z_i as the local surrogate model:

$$g(\boldsymbol{z}) = \phi_0 + \sum_{i=1}^{M} \phi_i z_i,$$

where ϕ_i are the SHAP values.

The SHAP values are found by solving the following weighted least squares problem:

$$\min_{\phi} \sum_{\boldsymbol{z} \in \{0,1\}^M} \pi_{\boldsymbol{x}}(\boldsymbol{z})[g(\boldsymbol{z}) - h_{\boldsymbol{x}}(\boldsymbol{z})]^2 + \lambda[\phi_0 + \sum_{i=1}^{M} \phi_i - f(\boldsymbol{x})]^2,$$

where λ is a regularization parameter. $\pi_{\boldsymbol{x}}(\boldsymbol{z})$ is a kernel function that provides weights for each $\boldsymbol{z}$[29]:

$$\pi_{\boldsymbol{x}}(\boldsymbol{z}) = \frac{M-1}{\binom{M}{|\boldsymbol{z}|}|\boldsymbol{z}|(M-|\boldsymbol{z}|)}.$$

Chapter 3

Experiment

In this chapter, we apply the SVM model and the MLP model to real sMRI datasets to illustrate the usefulness of statistical methodologies in predicting the cognitive status of Alzheimer's disease.

3.1 Data Resource

Datasets have been collected and organized across the Alzheimer's Disease Research Centers Program by the National Alzheimer's Coordinating Center (NACC). This thesis mainly uses two datasets: the Uniform Data Set (UDS) with basic descriptions (demographics, physical/neurological exam findings, clinician diagnosis, etc.) and the Imaging Data (ID) of UDS subjects (structural MRIs, calculated MRI summary data, and positron emission tomography scans). We use some features from UDS to set up 3 classes as target variables based on the clinical diagnosis and the diagnosis guideline[31]. Since we aim to explore the feature importance of structural regions of the human brain, we use the regional gray matter volumes and the regional cortical thicknesses from ID as our explanatory variables. The region of interest data was processed and obtained by the IDeA Laboratory at the University of California, Davis.

3.2 Class Determination

In this thesis, we use different methods to carry out classification for the three statuses of AD patients: NC, MCI, and AD. The NC and MCI cases are identified by the variable "NACCUDSD" which is the cognitive status at a UDS visit. Subjects with a clinical diagnosis of normal cognition have NACCUDSD = 1. Subjects with either amnestic or non-amnestic MCI have NACCUDSD = 3. The AD cases are identified by three variables "NACCUDSD", "NACCALZD" and "NACCALZP". "NACCALZD" is the presumptive etiologic diagnosis of the cognitive disorder - Alzheimer's disease and "NACCALZP" is the primary, contributing, or non-contributing cause of observed cognitive impairment - Alzheimer's disease. Subjects with a clinical diagnosis of Alzheimer's disease have NACCUDSD = 4, NACCALZD = 1 and NACCALZP = 1.

3.3 Data Wrangling and Data Exploration

In the UDS, each patient is assigned an identification code which is stored in the variable "NACCID". By using the unique "NACCID", we match the structural MRI data with every patient who participated. We selected patients who underwent MRI with a sequence type of T1 and a magnetic field strength of either 1.5 or 3, wherein the duration between the MRI examination and the physician's diagnosis did not exceed six months. Among all the MRI scans taken by the patient, we only chose the first-time scan without duplication to avoid potential leakage. After omitting all the observations with missing values, we had 1684 subjects from the original datasets. Table 3.1 reveals the basic demographic characteristics information of the subjects.

From Table 3.1, our dataset has more female subjects than male subjects and the number of subjects whose age is beyond 65 is larger than the number of subjects whose age is below 65.

Demographic Characteristics	NC (n=967)	MCI (n=390)	AD (n=327)
Male	338	224	174
Female	629	166	153
Age below 65	365	52	47
Age beyond 65	602	338	280

Table 3.1: Demographic characteristics of the subjects

3.4 Experiment Results and Analyses

This section uses two methods to explore the Alzheimer's disease dataset, from both linear and non-linear perspectives. For SVM, it sorts out the rank for feature importance, then uses the RFE to choose the best number of features based on cross-validation and uses the selected features to construct the model and tune the hyperparameters with grid research and forms the final model to make the prediction. For MLP, all the features are used in the training of forward propagation and backpropagation. The neural network uses Bayesian optimization to set up multiple trials to tune the hyperparameters. The SHAP method sorts out all feature importance based on the final model.

All the models are used for three-class and binary classification problems. We start with a three-class scenario and then discuss the binary situation. The dataset was split randomly into the training dataset and the test dataset with a proportion of 80% and 20%, respectively. Before any training, we standardize features by removing the mean and scaling to unit variance. Therefore, the data has a mean of 0 and a standard deviation of 1.

3.4.1 Linear-SVM

Feature Importance and Selection

The Linear-SVM model sorts out the feature importance by the weights of the hyperplane, the calculation formula is in (2.12). After a complete rank list of the features is obtained, RFE trains the model with cross-validation and saves the score. In every training with SVM,

RFE removes one or more weakest features until the model accuracy is no longer increasing. Figure 3.1 shows the cross-validation score variation. Figure 3.2 shows the feature importance sorted by the SVM-RFE model.

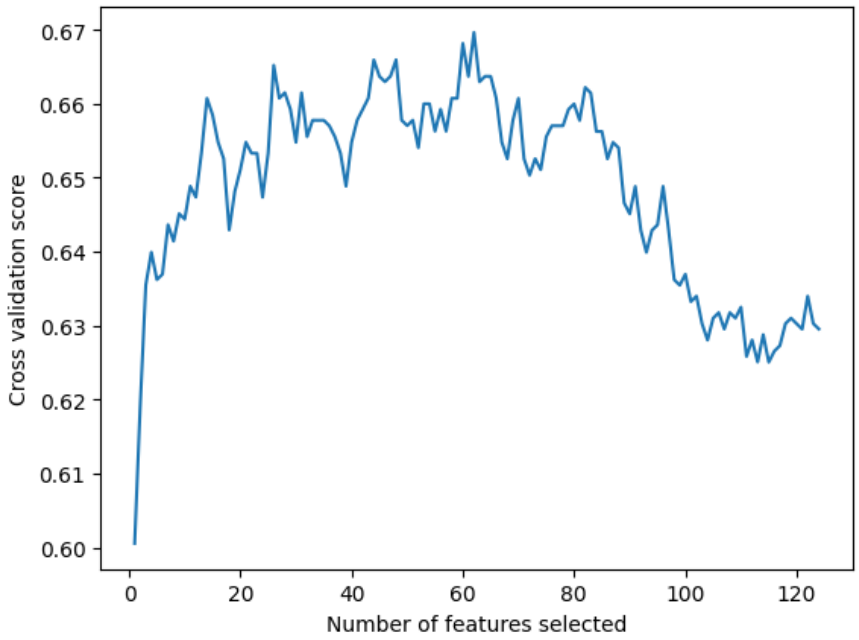


Figure 3.1: Cross validation score. This figure illustrates the variation of the cross-validation score with respect to the number of selected features.

From the curve of Figure 3.1, the optimal number of selected features is 62 with the largest cross-validation score. Therefore, in Figure 3.2, the most important 62 features have the same relative importance as 1. However, these 62 features are still ranked by the SVM model. The most important 10 features are as follows in Table 3.2.

From Table 3.2, left transverse temporal mean cortical thickness is the most important feature for predicting the patient's status of Alzheimer's disease by the SVM model. Left pars, right lateral, and left postcentral have more than one feature to make a big contribution to the prediction model.

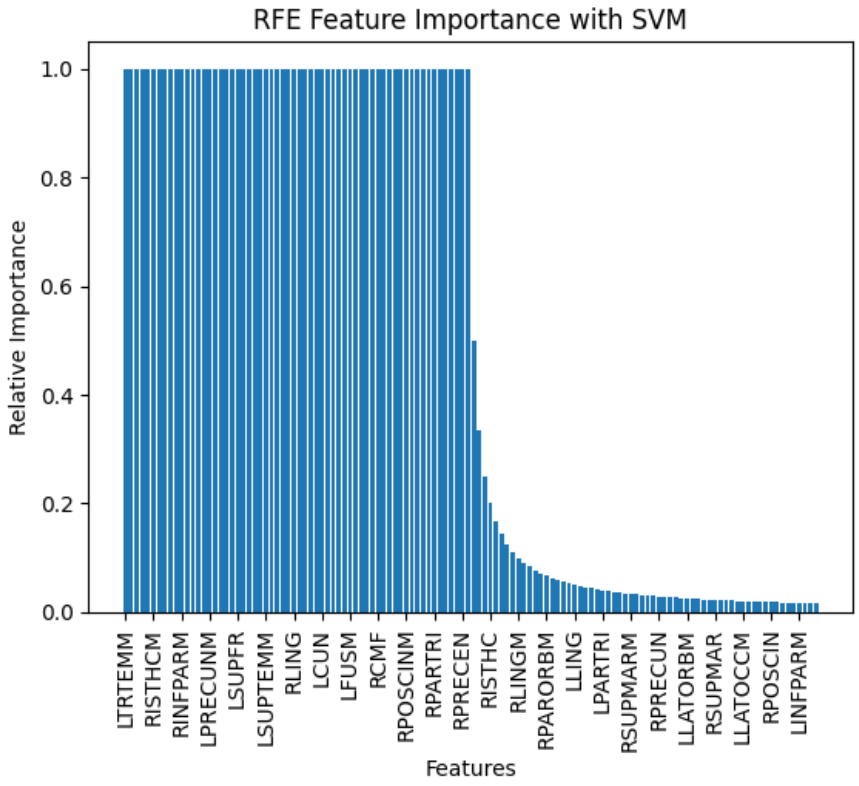


Figure 3.2: Feature importance by SVM with all 3 classes. This figure shows the feature importance rank.

Variable	Description
LTRTEMM	Left transverse temporal mean cortical thickness (mm)
LPARSOPM	Left pars opercularis mean cortical thickness (mm)
LPARORB	Left pars orbitalis gray matter volume (cc)
RLATORBF	Right lateral orbitofrontal gray matter volume (cc)
RLATOCCM	Right lateral occipital mean cortical thickness (mm)
RISTHCM	Right isthmus cingulate mean cortical thickness (mm)
LPERCAL	Left pericalcarine gray matter volume (cc)
RINFTEMP	Right inferior temporal gray matter volume (cc)
LPOSCEN	Left postcentral gray matter volume (cc)
LPOSCENM	Left postcentral mean cortical thickness (mm)

Table 3.2: Top 10 important features by SVM

Hyperpatameter-tuning

The main hyperparameters for SVM are the regularization parameter C and the tolerance T for stopping criteria. The size of C could cause the model from underfitting to overfitting, as discussed after (2.3). The tolerance T controls when the SVM model stops. If T is too high, the SVM might stop too early before finding the optimal solution. If T is too low, the SVM might run for too long, wasting computational resources without significantly improving the model.

The strategy for tuning hyperparameters in an SVM model is grid search. Grid search is a brute-force exhaustive search paradigm using all the parameter combinations from a list of specified values. With a range of C: 0.1, 1, 10, 100 and a range of T: 0.0001, 0.001, 0.01, 0.1, the best parameters for the model are C: 0.1 and T: 0.01.

Classification Report

After the training part is finished and the parameters for the model are estimated, we apply the model to the test dataset. Table 3.3 shows the confusion matrix for our model. For example, 174 is the number of instances that belong to NC and were correctly predicted as NC. 12 is the number of instances that belong to NC and were incorrectly predicted as MCI.

From Table 3.3, the total test accuracy is 68.84%, which is a good result for three-class

		Predicted(Class Accuracy)		
		NC	**MCI**	**AD**
Actual	**NC**	174(88.32%)	12	11
	MCI	46	20(24.39%)	16
	AD	13	7	38(65.52%)

Table 3.3: Confusion Matrix for 3 classes in SVM

classifications. Let TP, TN, FP, and FN represent the cases that are true positive, true negative, false positive, and false negative, respectively. The most common classification evaluation metrics are:

(1) Precision: the ability of a classifier not to label an instance positive that is negative.

$$Precision = \frac{TP}{TP + FP}.$$

(2) Recall (Sensitivity): the ability of a classifier to find all positive instances.

$$Recall = \frac{TP}{TP + FN}.$$

(3) F1 score: a weighted harmonic mean of precision and recall.

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

(4) Support: the number of actual occurrences of the class in the test dataset.

$$Support(Class) = \sum Actual(Class).$$

(5) Macro Average: computes the metric independently for each class and then takes the

average.

$$Macro - Avg = \frac{1}{N}\sum_{i=1}^{N} Metric(Class_i).$$

(6) Weighted Average: computes the average metric, giving to each class a weight that is equal to the number of instances of that class.

$$Weighted - Avg = \frac{1}{sum\{Support(Class_i)\}}\sum_{i=1}^{N}[Metric(Class_i) \times Support(Class_i)].$$

Based on the confusion matrix in Table 3.3, we can calculate all the evaluation metrics in Table 3.4.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
NC	0.75	0.88	0.81	197
MCI	0.51	0.24	0.33	82
AD	0.58	0.66	0.62	58
Macro Avg	0.61	0.59	0.59	337
Weighted Avg	0.66	0.69	0.66	337

Table 3.4: Classification Report for SVM

We aspire for all metrics to be as close to 1 as possible. In the context of predicting Alzheimer's disease, recall is the most paramount metric of all. This is because minimizing instances where patients with Alzheimer's are mistakenly diagnosed as healthy is of utmost importance. From Table 3.4, the values of recall are higher than other metrics for NC and AD, which means the SVM model is better at predicting NC and AD situations rather than MCI. The Macro average results are lower than the corresponding metrics for NC and AD which suggests that the SVM model may be biased towards these two classes, especially NC. The Weighted average results are beyond 0.65 which means the SVM model performs reasonably well.

Binary Classification Results

In addition to exploring the multi-class scenario, we have also analyzed our model's predictive capabilities in binary classification. The cross-validation score plots, the feature importance plots, and the ranks are in the appendix. For the binary pairs of NC vs MCI, MCI vs AD, and NC vs AD, the optimal number for selected features are 13, 22, and 15, respectively. Table 3.5 shows the best hyperparameters. Table 3.6 shows the confusion matrix. The classification reports for each pair are in the appendix.

	C	**T**
NC vs MCI	10	0.01
MCI vs AD	1	0.0001
NC vs AD	100	0.01

Table 3.5: Best hyperparameters for the pairs of binary classification in SVM

		Predicted	
		NC	**MCI**
Actual	**NC**	188	8
	MCI	61	15

		Predicted	
		MCI	**AD**
Actual	**MCI**	47	31
	AD	20	46

		Predicted	
		NC	**AD**
Actual	**NC**	174	15
	AD	24	46

Table 3.6: Confusion matrices for binary classification problems in SVM. From left to right: NC vs MCI, MCI vs AD, NC vs AD.

According to Table 3.6, the class accuracies for the pair of NC vs MCI are 95.92% in NC and 19.74% in MCI, the class accuracies for the pair of MCI vs AD are 60.26% in MCI and 69.70% in AD, the class accuracies for the pair of NC vs AD are 92.06% in NC and 65.71% in AD. The total test accuracies for the pairs of NC vs MCI, MCI vs AD, and NC vs AD are 74.63%, 64.58%, and 84.94%, respectively. The SVM model can predict pretty well between NC cases and AD cases but relatively worse between MCI cases and AD cases.

In a binary classification problem, there are two more important evaluated metrics: the receiver operating characteristic (ROC) curve and the area under the ROC curve (AUC). The ROC curve illustrates the performance of a binary classifier system as its discrimination

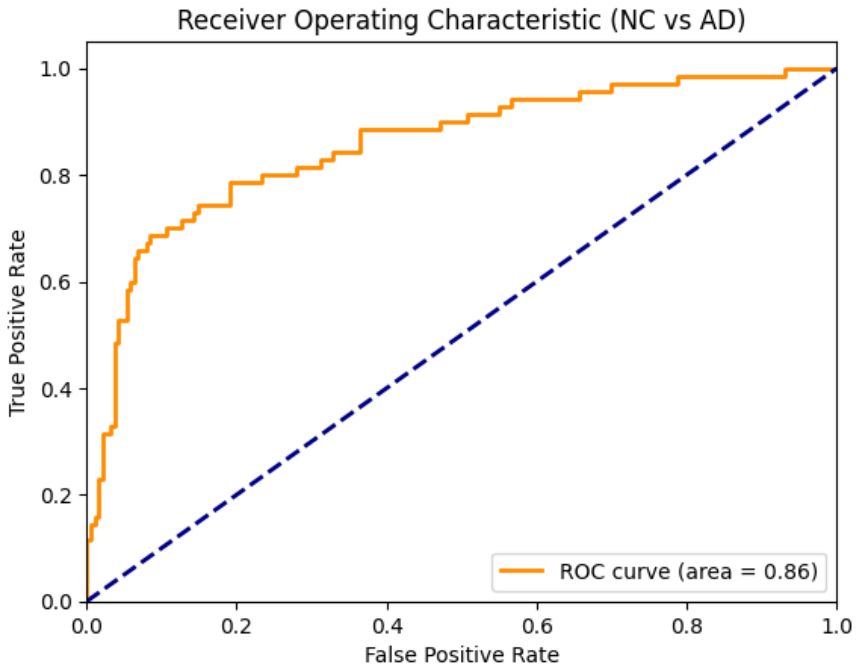


Figure 3.3: ROC curve by SVM for NC vs AD. This figure shows the true positive rate vs the false positive rate.

threshold is varied. The AUC provides an aggregate measure of the performance across all possible classification thresholds. Figure 3.3 includes the ROC curve and the AUC value for NC vs AD, the other two figures are in the appendix.

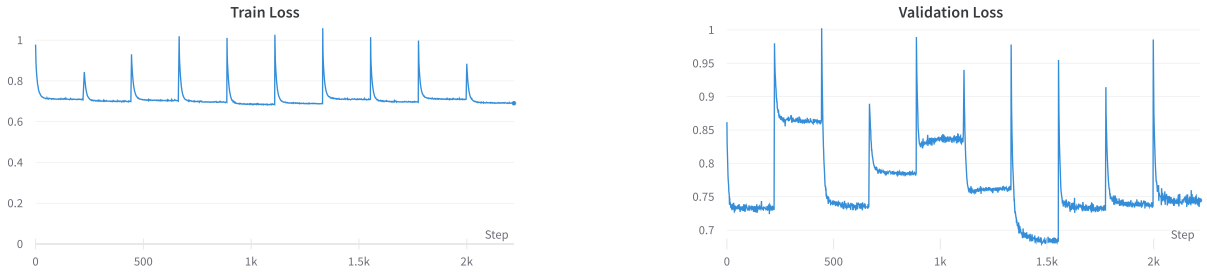
The ROC curve is close to the top-left corner of the plot which indicates a good model. The AUC = 0.86 which means there is 86% chance that the model will be able to distinguish between positive class and negative class.

3.4.2 MLP with Bayesian Optimization

Training Performance

The training process of a neural network is relatively complex, necessitating a careful observation of model performance during training. Initially, we set the number of trials for Bayesian Optimization to 50, with 10-fold cross-validation. Each trial was conducted with different hyperparameters. The training loss, validation loss, and accuracy of each epoch

were recorded. The average accuracy across all epochs in a single trial was considered the validation accuracy for that trial. Figure 3.4 shows the training loss vs the validation loss curve for the last trial. After recording the training loss and validation loss for all trials, we apply the moving average which smooths out fluctuations in the loss and makes the overall trends more apparent. Let moving window size = 10, which means that each point on the smoothed loss curves is the average of 10 consecutive points on the original loss curves. Figure 3.5 shows the average training loss and average validation loss through all trials as the epoch increases.



(a) Train loss

(b) Validation loss

Figure 3.4: Train loss and validation loss for the last trial. They both decrease in a certain step period.

Figure 3.5 indicates that the MLP model is learning and generalizing well to unseen data.

Hyperpatameter-tuning

For MLP, there are 6 hyperparameters to tune in Bayesian Optimization:

(1) Learning rate: a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated.

(2) Batch size: the number of samples that will be passed through the network at once. It directly affects the gradient estimation and consequently, the learning dynamics of the weights.

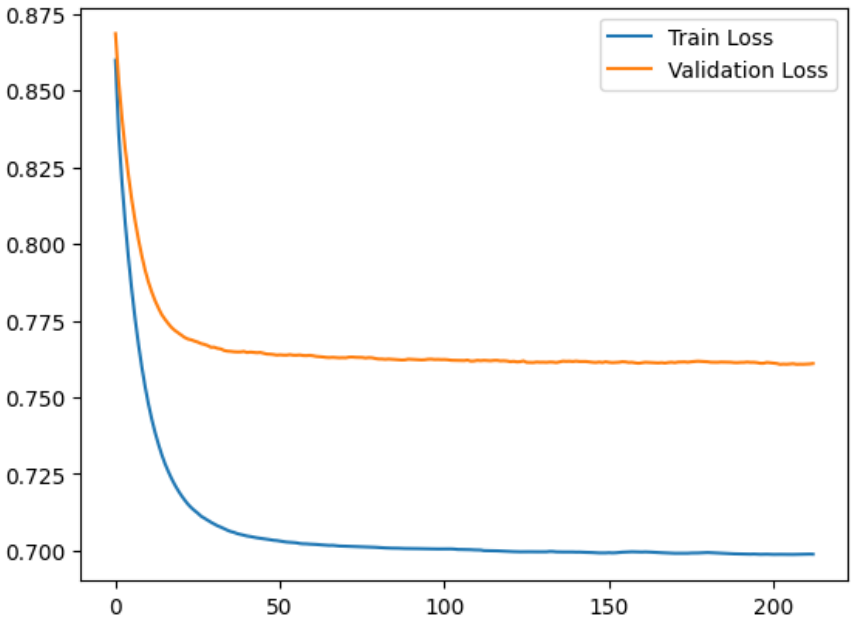


Figure 3.5: Train loss and validation loss in the average of all trials. They both decrease as the epoch increases.

(3) Hidden size: the number of nodes or neurons in each hidden layer. The number of these nodes is a measure of the model's capacity.

(4) Epochs: an epoch is one complete pass through the entire training dataset.

(5) Number of layers: This is the number of hidden layers in the neural network.

(6) Weight decay: a regularization technique by adding a small penalty, the L2 norm of the weights, to the loss function to reduce overfitting by preventing the weights from growing too large.

Table 3.7 shows the range of each hyperparameter and the best hyperparameters chosen by the MLP model.

Feature Importance

For the SHAP method, features with large absolute Shapley values are important. Let $\phi_j^{(i,c)}$ be the SHAP value for instance j to the class c and feature i, the importance for feature i

	learning rate(Lr)	**batch size(Bs)**	**hidden size(Hs)**
Range	[1e-6, 0.1]	[64, 512]	[64, 512]
Optimum	3.8082e-4	64	185
	epochs(Eps)	**number of layers(Nl)**	**weight decay(Wd)**
Range	[64, 256]	[1, 4]	[1e-5, 0.1]
Optimum	221	1	0.1

Table 3.7: Hyperparameter-tuning for MLP

in class c is:

$$I_{i,c} = \frac{1}{N} \sum_{j=1}^{N} |\phi_j^{(i,c)}|.$$

Figure 3.6 shows the top 30 feature importance in MLP.

From Figure 3.6, we can gather the top 10 most important features ranked by the MLP model in Table 3.8.

Variable	**Description**
LENTM	Left entorhinal mean cortical thickness (mm)
RENTM	Right entorhinal mean cortical thickness (mm)
LISTHCM	Left isthmus cingulate mean cortical thickness (mm)
RPARCENM	Right paracentral mean cortical thickness (mm)
RENT	Right entorhinal gray matter volume (cc)
LENT	Left entorhinal gray matter volume (cc)
LINFTEMP	Left inferior temporal gray matter volume (cc)
LLING	Left lingual gray matter volume (cc)
RISTHCM	Right isthmus cingulate mean cortical thickness (mm)
LMIDTEMP	Left middle temporal gray matter volume (cc)

Table 3.8: Top 10 important features by MLP

According to Table 3.8, both left and right entorhinal regions, as well as mean cortical thickness and gray matter volume, are crucial factors for predicting the patient's status of Alzheimer's disease using the MLP model.

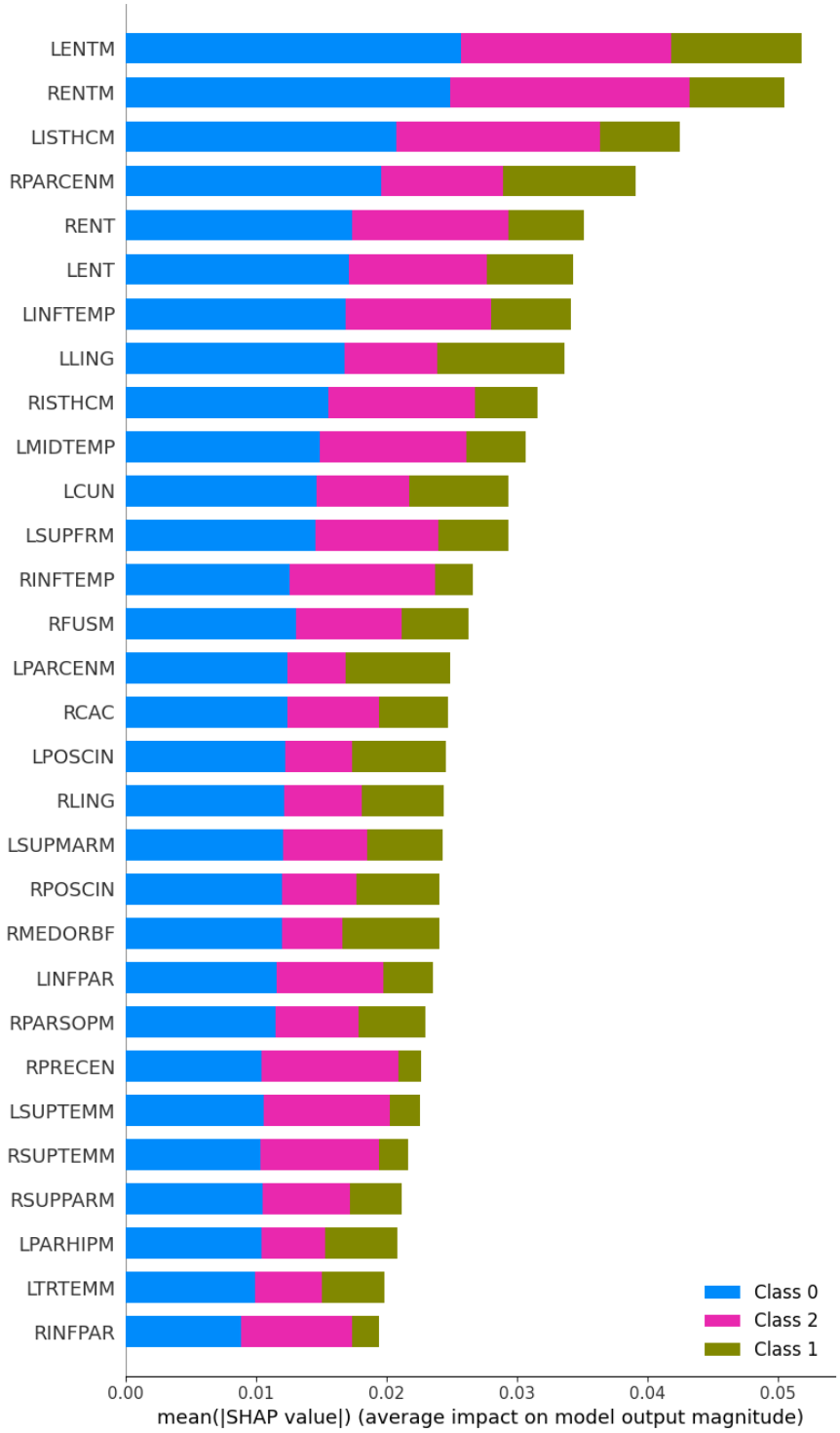


Figure 3.6: Top 30 feature importance in MLP. Each feature has different impacts on distinct classes. The blue bar is for NC, the olive bar is for MCI, and the pink bar is for AD.

Classification Report

For the MLP model, Table 3.9 shows the confusion matrix information. Table 3.10 shows the evaluation metrics.

		Predicted(Class Accuracy)		
		NC	**MCI**	**AD**
Actual	**NC**	185(93.91%)	3	9
	MCI	53	9(10.98%)	20
	AD	13	0	45(77.59%)

Table 3.9: Confusion Matrix for 3 classes in MLP

Class	**Precision**	**Recall**	**F1-Score**	**Support**
NC	0.74	0.94	0.83	197
MCI	0.75	0.11	0.19	82
AD	0.61	0.78	0.68	58
Macro Avg	0.70	0.61	0.57	337
Weighted Avg	0.72	0.71	0.65	337

Table 3.10: Classification Report for MLP

From Table 3.10, almost all the metrics are improved compared to the SVM model in Table 3.4 except the recall and F1-Score for MCI. It illustrates that the MLP model has a better ability to predict NC and AD situations than the SVM model but worse in MCI situation.

Binary Classification Results

In the MLP model, we add more methods to prevent severe overfitting problems. First, we apply an early stopping technique which halts training once the model's performance starts to deteriorate on a validation dataset. We also add the dropout rate to hidden layers where during training, a certain fraction of neurons are randomly ignored to improve generalization. We add the $L1$ penalty that discourages complex or excessive features in a model by adding the absolute value of the coefficients to the loss function. Consistent with all the results from

the SVM model, Table 3.11 and Table 3.12 show the best hyperparameters and confusion matrices for each pair, respectively. The rest of the classification reports are in the appendix.

	Lr	**Bs**	**Hs**	**Eps**	**Nl**	**Wd**	**Do**	**L1**
NC vs MCI	1.78e-03	151	56	209	1	3.83e-03	0.3535	1.60e-03
MCI vs AD	6.73e-03	512	23	183	1	1.33e-05	0.4446	6.87e-05
NC vs AD	1.59e-03	257	61	205	1	6.89e-03	0.3600	9.77e-05

Table 3.11: Best hyperparameters for the pairs of binary classification in MLP

		Predicted	
		NC	**MCI**
Actual	**NC**	168	28
	MCI	53	23

		Predicted	
		MCI	**AD**
Actual	**MCI**	51	27
	AD	20	46

		Predicted	
		NC	**AD**
Actual	**NC**	171	18
	AD	26	44

Table 3.12: Confusion matrices for binary classification problems in MLP. From left to right: NC vs MCI, MCI vs AD, NC vs AD.

From Table 3.12, the class accuracies for the pair of NC vs MCI are 85.71% in NC and 30.26% in MCI, the class accuracies for the pair of MCI vs AD are 65.38% in MCI and 69.70% in AD, the class accuracies for the pair of NC vs AD are 90.48% in NC and 62.86% in AD. The total accuracies for the pairs of NC vs MCI, MCI vs AD, and NC vs AD are 70.22%, 67.36%, and 83.01%, respectively. The results to distinguish between MCI cases and AD cases are less ideal than the other two, which is similar to the results of the SVM model.

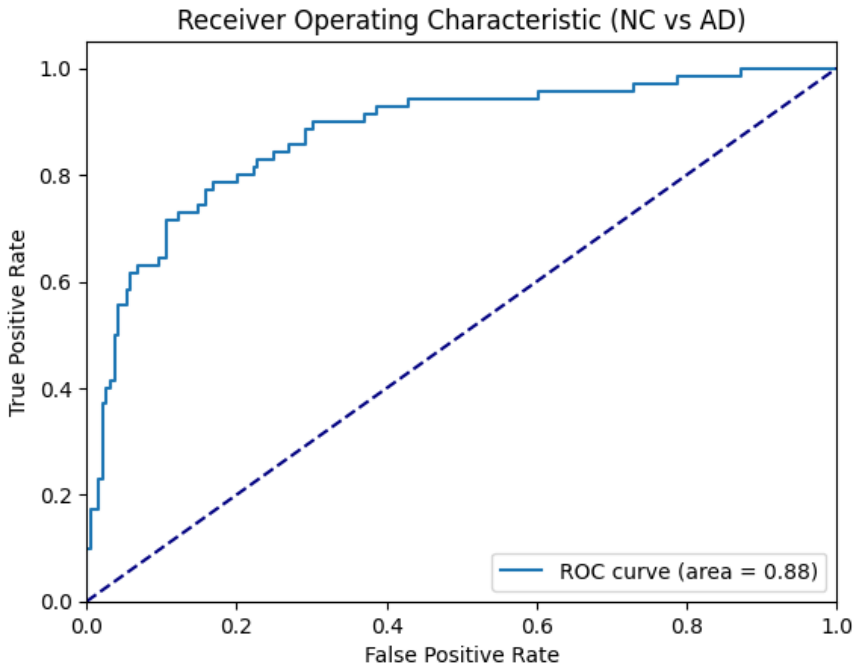


Figure 3.7: ROC curve by MLP for NC vs AD.

From Figure 3.7, the AUC = 0.88 which means there is 88% chance that the model will be able to distinguish between positive class and negative class.

Chapter 4

Conclusions

This thesis endeavored to predict different stages of Alzheimer's disease by investigating two machine learning models SVM and MLP. The results are shown in Table 4.1.

Accuracy(%)	**SVM**	**MLP**
Three classes	68.84	70.92
NC vs MCI	74.63	70.22
MCI vs AD	64.58	67.36
NC vs AD	84.94	83.01

Table 4.1: Test accuracy of SVM and MLP

Generally speaking, the MLP model exhibits superior performance in predicting the NC and AD states, while the SVM model demonstrates enhanced prediction for the MCI state. As a linear model, SVM offers better reproducibility than the nonlinear MLP model.

The structural MRI dataset used in this study was sourced from NACC. By appropriately selecting and partitioning the dataset, we managed to avoid the common leakage problem. When compared with peer research, MCI emerges as the most challenging class to define and predict. The final results also reflect the inherent imbalance in the dataset from other metrics.

The novelty of this thesis lies in the comparative analysis of the predictive efficacy of the SVM and MLP models on structural MRI data. By applying methods such as recursive

feature elimination and Bayesian optimization, we managed to optimize the models, resulting in improved accuracy. From the results, it can be observed that the SVM model outperforms the MLP model in binary classification tasks. Due to its higher complexity, the MLP model tends to overfit imbalanced data, making it more suitable for handling ternary classification problems than binary classification problems.

The results obtained in this thesis are based on larger data sets compared to those in the literature. While this leads to slightly lower accuracy rates compared to those on smaller data sets, it enhances the model's generalizability for predicting unknown data. The predictive outcomes for new samples serve as a reference for medical professionals in patient diagnosis. When supplemented with statistical insights, these findings can further aid clinicians in making more accurate diagnostic assessments. Future directions for this research include collecting more data on MCI and AD to balance the data set for more accurate predictions.

Bibliography

[1] Gauthier, S., Webster, C., Servaes, S., Morais, J. A., & Rosa-Neto, P. (2022). World Alzheimer report 2022: life after diagnosis: navigating treatment, care and support. London, England: Alzheimer's Disease International.

[2] Scheltens, P., Blennow, K., Breteler, M. M., De Strooper, B., Frisoni, G. B., Salloway, S., & Van der Flier, W. M. (2016). Alzheimer's disease. The Lancet, 388(10043), 505-517.

[3] McKhann, G. M., Knopman, D. S., Chertkow, H., Hyman, B. T., Jack Jr, C. R., Kawas, C. H., ... & Phelps, C. H. (2011). The diagnosis of dementia due to Alzheimer's disease: Recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease. Alzheimer's & dementia, 7(3), 263-269.

[4] 2023 Alzheimer's disease facts and figures. (2023). Alzheimer's & dementia : the journal of the Alzheimer's Association, 19(4), 1598–1695. https://doi.org/10.1002/alz.13016

[5] Albert, M. S., DeKosky, S. T., Dickson, D., Dubois, B., Feldman, H. H., Fox, N. C., ... & Phelps, C. H. (2011). The diagnosis of mild cognitive impairment due to Alzheimer's disease: recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease. Alzheimer's & dementia, 7(3), 270-279.

[6] Weller, J., & Budson, A. (2018). Current understanding of Alzheimer's disease diagnosis and treatment. F1000Research, 7.

[7] Alexandrov, P. N., Dua, P., Hill, J. M., Bhattacharjee, S., Zhao, Y., & Lukiw, W. J. (2012). microRNA (miRNA) speciation in Alzheimer's disease (AD) cerebrospinal fluid (CSF) and extracellular fluid (ECF). International journal of biochemistry and molecular biology, 3(4), 365.

[8] Jack, C. R., & Holtzman, D. M. (2013). Biomarker modeling of Alzheimer's disease. Neuron, 80(6), 1347-1358.

[9] Garg, N., Choudhry, M. S., & Bodade, R. M. (2023). A review on Alzheimer's disease classification from normal controls and mild cognitive impairment using structural MR images. Journal of neuroscience methods, 384, 109745.

[10] Chupin, M., Gérardin, E., Cuingnet, R., Boutet, C., Lemieux, L., Lehéricy, S., ... & Colliot, O. (2009). Fully automatic hippocampus segmentation and classification in Alzheimer's disease and mild cognitive impairment applied on data from ADNI. Hippocampus, 19(6), 579-587.

[11] Liu, M., Li, F., Yan, H., Wang, K., Ma, Y., Shen, L., ... & Alzheimer's Disease Neuroimaging Initiative. (2020). A multi-model deep convolutional neural network for automatic hippocampus segmentation and classification in Alzheimer's disease. Neuroimage, 208, 116459.

[12] Pennanen, C., Kivipelto, M., Tuomainen, S., Hartikainen, P., Hänninen, T., Laakso, M. P., ... & Soininen, H. (2004). Hippocampus and entorhinal cortex in mild cognitive impairment and early AD. Neurobiology of aging, 25(3), 303-310.

[13] Savio, A., García-Sebastián, M. T., Chyzyk, D., Hernández, C., Graña, M., Sistiaga, A., ... & Villanúa, J. (2011). Neurocognitive disorder detection based on feature vectors extracted from VBM analysis of structural MRI. Computers in biology and medicine, 41(8), 600-610.

[14] Zhang, F., Tian, S., Chen, S., Ma, Y., Li, X., & Guo, X. (2019). Voxel-based morphometry: improving the diagnosis of Alzheimer's disease based on an extreme learning machine method from the ADNI cohort. Neuroscience, 414, 273-279.

[15] Park, H., Yang, J. J., Seo, J., & Lee, J. M. (2012). Dimensionality reduced cortical features and their use in the classification of Alzheimer's disease and mild cognitive impairment. Neuroscience letters, 529(2), 123-127.

[16] Li, S., Yuan, X., Pu, F., Li, D., Fan, Y., Wu, L., ... & Han, Y. (2014). Abnormal changes of multidimensional surface features using multivariate pattern classification in amnestic mild cognitive impairment patients. Journal of Neuroscience, 34(32), 10541-10553.

[17] Liu, S., Masurkar, A. V., Rusinek, H., Chen, J., Zhang, B., Zhu, W., Fernandez-Granda, C., & Razavian, N. (2022). Generalizable deep learning model for early Alzheimer's disease detection from structural MRIs. Scientific reports, 12(1), 17106.

[18] Karush, W. (1939). Minima of functions of several variables with inequalities as side constraints. M. Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago.

[19] Boyd, S. P., & Vandenberghe, L. (2004). Convex optimization. Cambridge university press.

[20] Kuhn, H., & Tucker, A. (1951). Nonlinear programming In Proceedings of 2nd Berkeley symposium (pp. 481–492). Berkeley: University of California Press.

[21] Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. Machine learning, 46, 389-422.

[22] Tieleman, T., & Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, 4(2), 26-31.

[23] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[24] Frazier, P. I. (2018). A tutorial on Bayesian optimization. arXiv preprint arXiv:1807.02811.

[25] Williams, C. K., & Rasmussen, C. E. (2006). Gaussian processes for machine learning (Vol. 2, No. 3, p. 4). Cambridge, MA: MIT press.

[26] Spanier, J., Oldham, K. B., & Romer, R. H. (1988). An atlas of functions.

[27] Clark, C. E. (1961). The greatest of a finite set of random variables. Operations Research, 9(2), 145-162.

[28] Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. Journal of Global optimization, 13, 455-492.

[29] Mockus, J. (1975). On the Bayes methods for seeking the extremal point. IFAC Proceedings Volumes, 8(1), 428-431.

[30] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. Advances in neural information processing systems, 30.

[31] McKhann, G. M., Knopman, D. S., Chertkow, H., Hyman, B. T., Jack Jr, C. R., Kawas, C. H., ... & Phelps, C. H. (2011). The diagnosis of dementia due to Alzheimer's disease: Recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease. Alzheimer's & dementia, 7(3), 263-269.

Appendix A

Data Source and Acknowledgment

The NACC database is funded by NIA/NIH Grant U24 AG072122. NACC data are contributed by the NIA-funded ADRCs: P30 AG062429 (PI James Brewer, MD, PhD), P30 AG066468 (PI Oscar Lopez, MD), P30 AG062421 (PI Bradley Hyman, MD, PhD), P30 AG066509 (PI Thomas Grabowski, MD), P30 AG066514 (PI Mary Sano, PhD), P30 AG066530 (PI Helena Chui, MD), P30 AG066507 (PI Marilyn Albert, PhD), P30 AG066444 (PI John Morris, MD), P30 AG066518 (PI Jeffrey Kaye, MD), P30 AG066512 (PI Thomas Wisniewski, MD), P30 AG066462 (PI Scott Small, MD), P30 AG072979 (PI David Wolk, MD), P30 AG072972 (PI Charles DeCarli, MD), P30 AG072976 (PI Andrew Saykin, PsyD), P30 AG072975 (PI David Bennett, MD), P30 AG072978 (PI Neil Kowall, MD), P30 AG072977 (PI Robert Vassar, PhD), P30 AG066519 (PI Frank LaFerla, PhD), P30 AG062677 (PI Ronald Petersen, MD, PhD), P30 AG079280 (PI Eric Reiman, MD), P30 AG062422 (PI Gil Rabinovici, MD), P30 AG066511 (PI Allan Levey, MD, PhD), P30 AG072946 (PI Linda Van Eldik, PhD), P30 AG062715 (PI Sanjay Asthana, MD, FRCP), P30 AG072973 (PI Russell Swerdlow, MD), P30 AG066506 (PI Todd Golde, MD, PhD), P30 AG066508 (PI Stephen Strittmatter, MD, PhD), P30 AG066515 (PI Victor Henderson, MD, MS), P30 AG072947 (PI Suzanne Craft, PhD), P30 AG072931 (PI Henry Paulson, MD, PhD), P30 AG066546 (PI Sudha Seshadri, MD), P20 AG068024 (PI Erik Roberson, MD, PhD), P20 AG068053

(PI Justin Miller, PhD), P20 AG068077 (PI Gary Rosenberg, MD), P20 AG068082 (PI Angela Jefferson, PhD), P30 AG072958 (PI Heather Whitson, MD), P30 AG072959 (PI James Leverenz, MD).

Appendix B

Additional figures

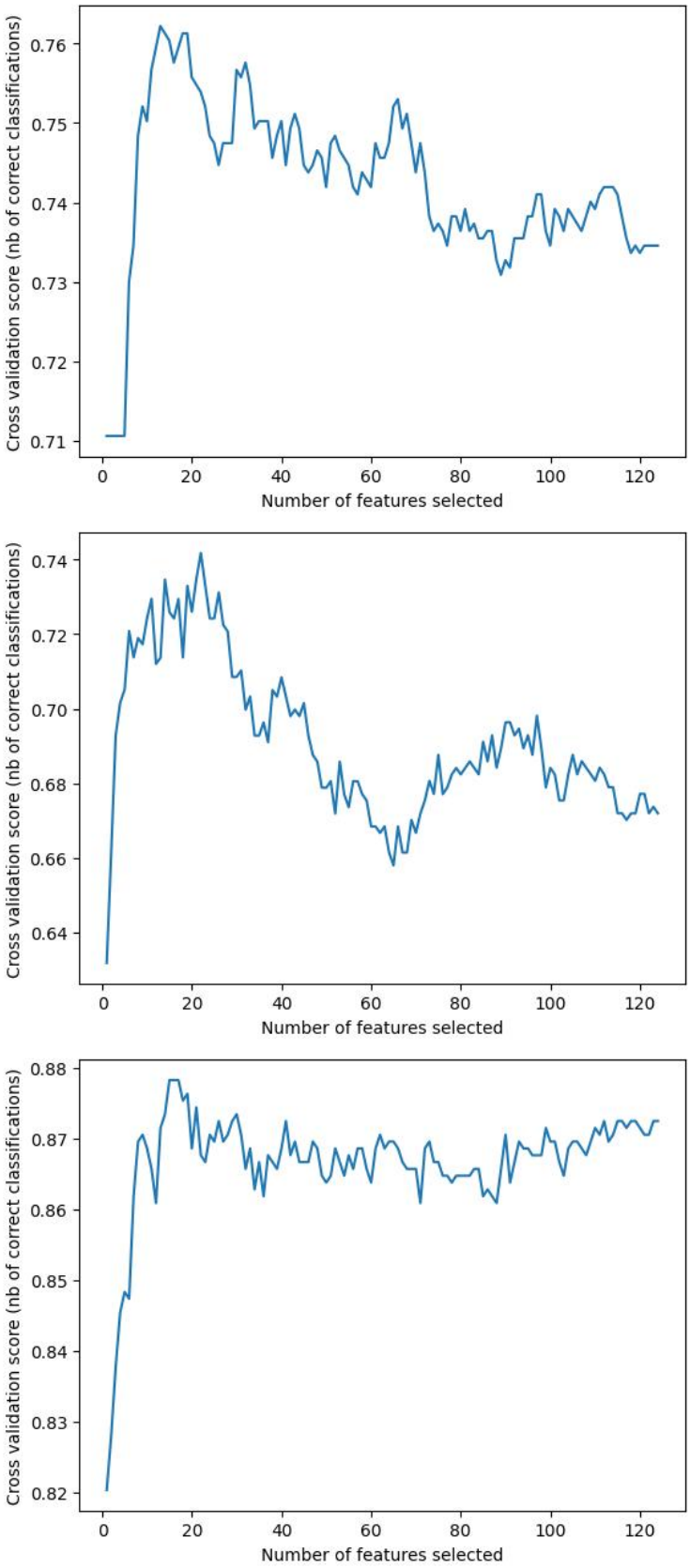


Figure B.1: Cross validation scores for SVM model. From top to bottom: NC vs MCI, MCI vs AD, NC vs AD.

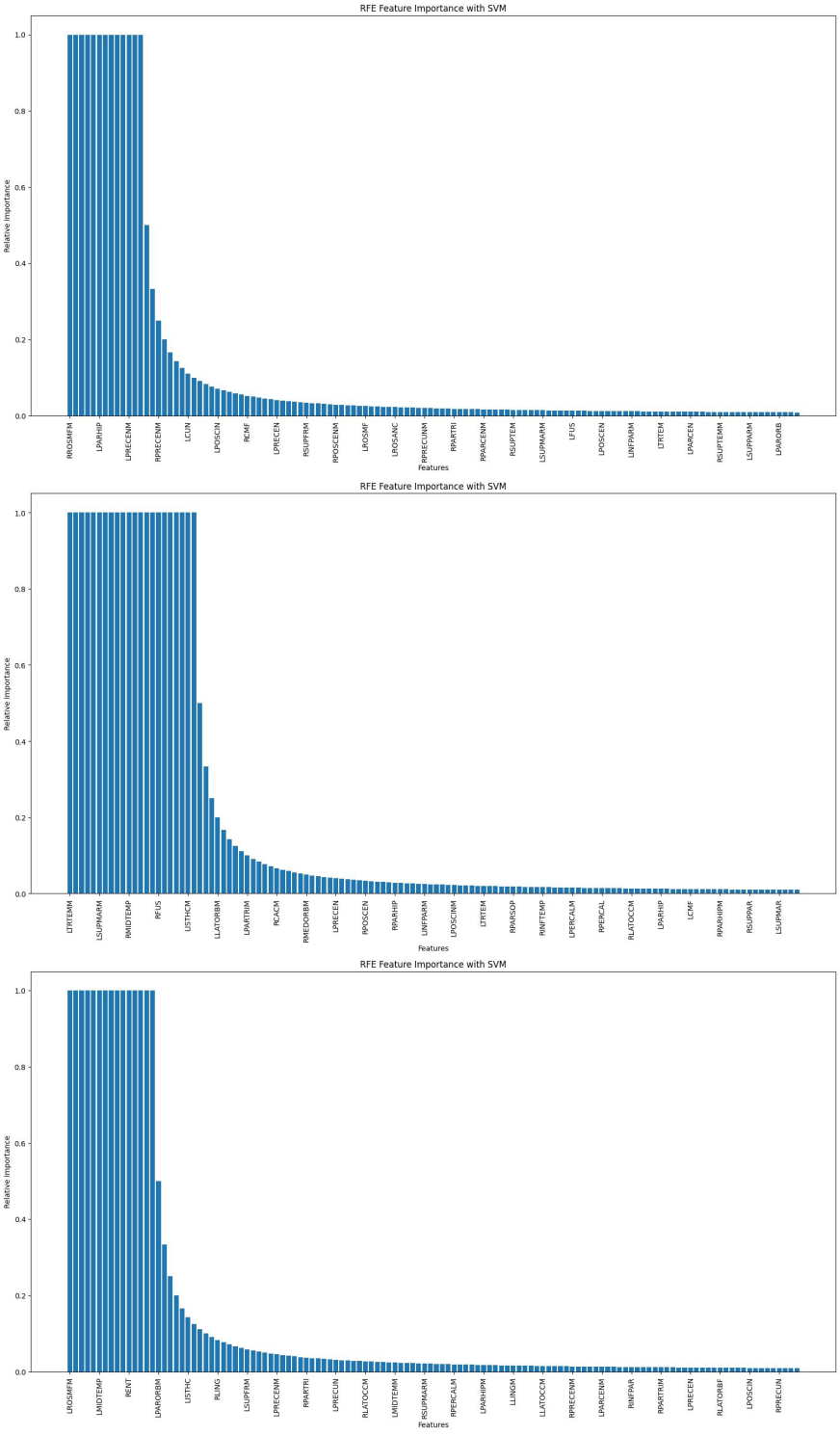


Figure B.2: Feature importance by SVM models. From top to bottom: NC vs MCI, MCI vs AD, NC vs AD.

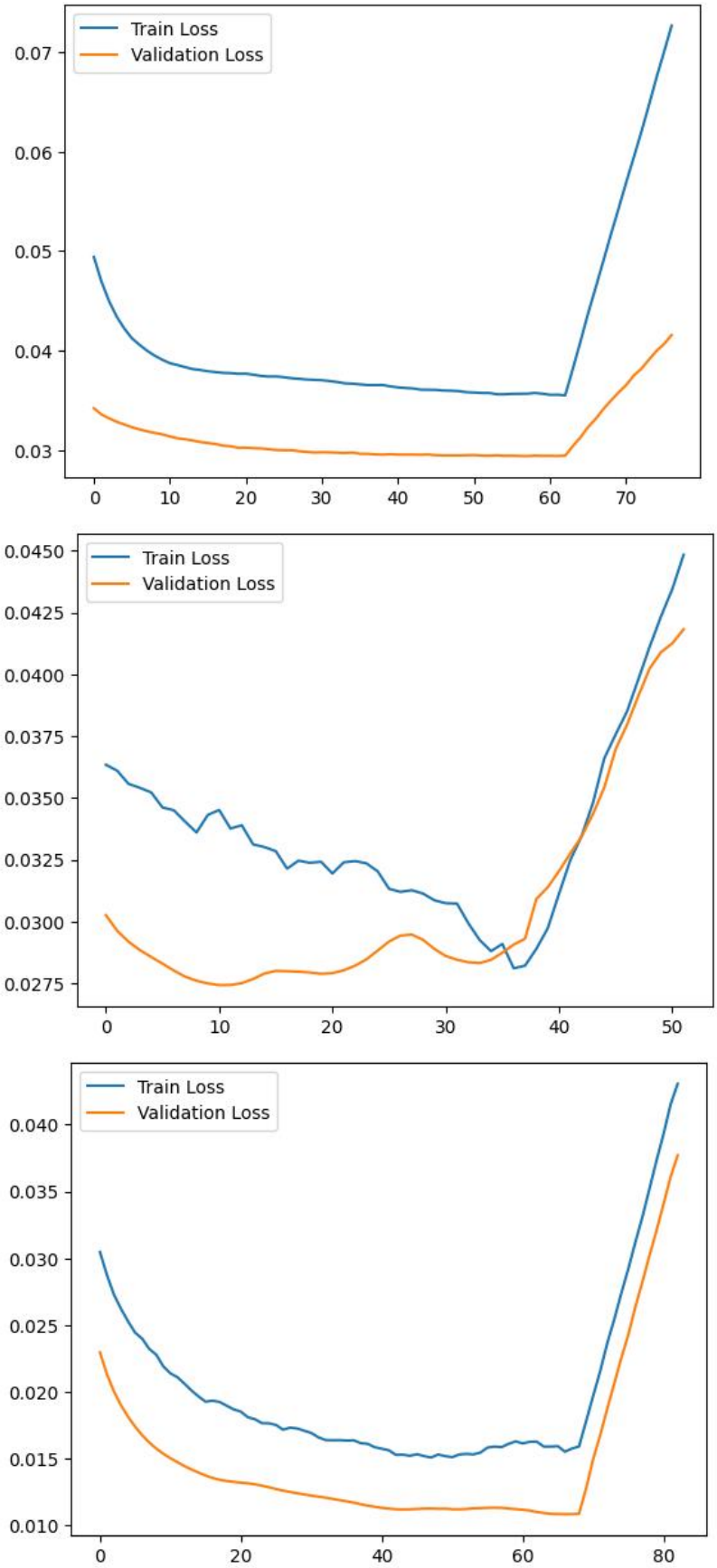


Figure B.3: Train loss and validation loss in the average of all trials by MLP models. From top to bottom: NC vs MCI, MCI vs AD, NC vs AD.

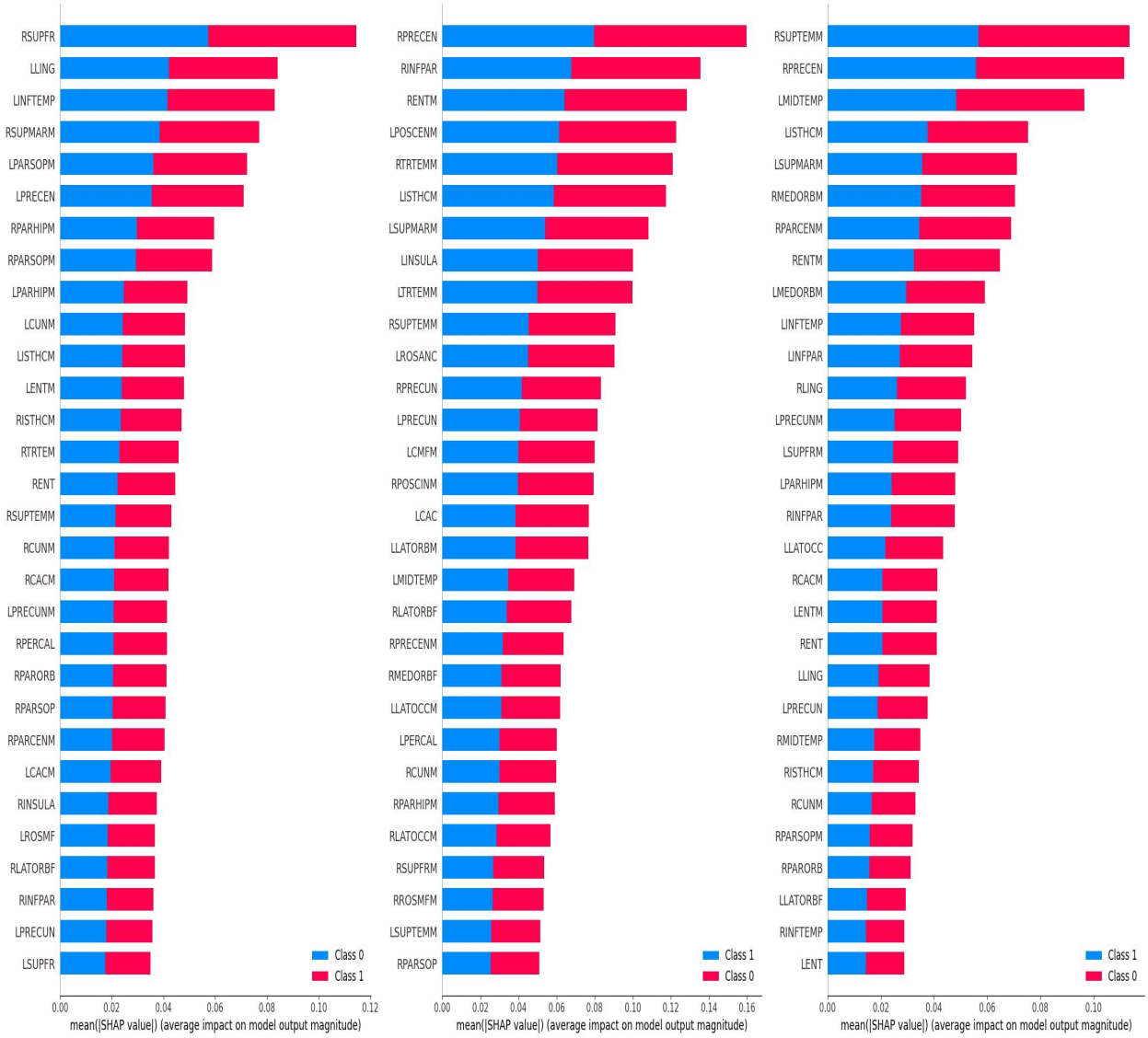


Figure B.4: Top 30 feature importance in MLP. From left to right: NC vs MCI, MCI vs AD, NC vs AD, corresponding to 0 vs 1.

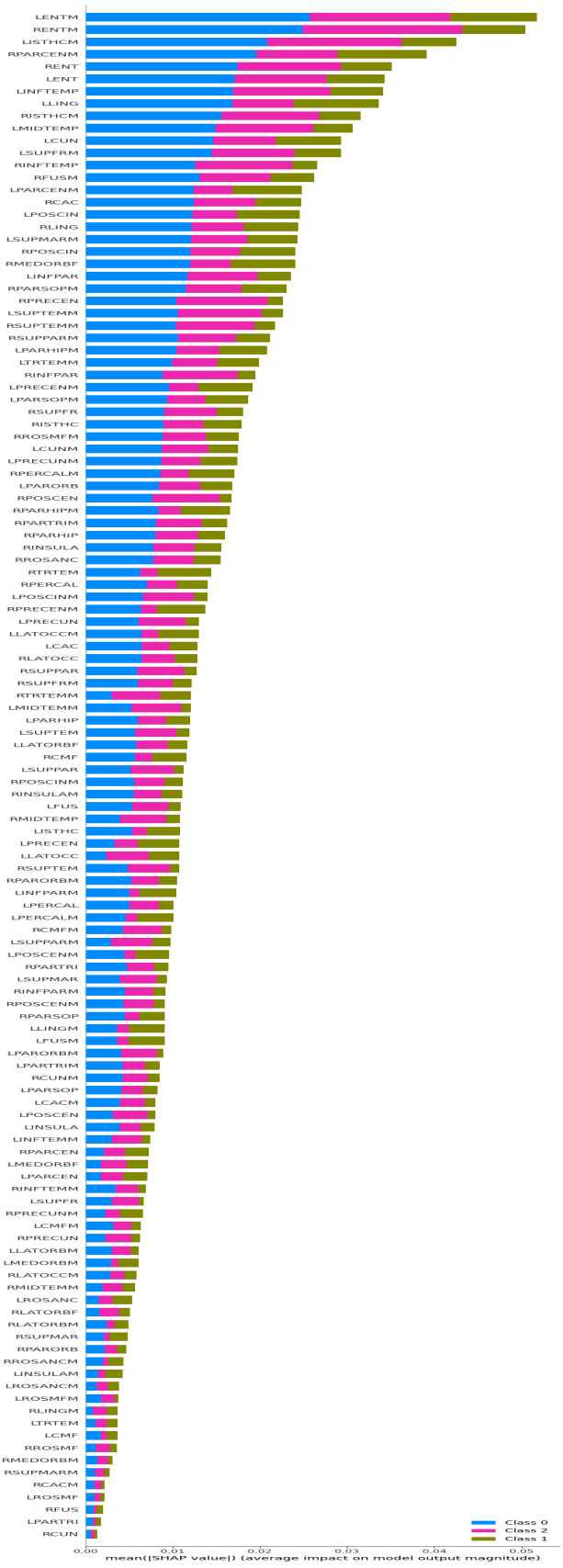


Figure B.5: Feature importance in MLP for all 3 classes. The blue bar is for NC, the olive bar is for MCI, and the pink bar is for AD.

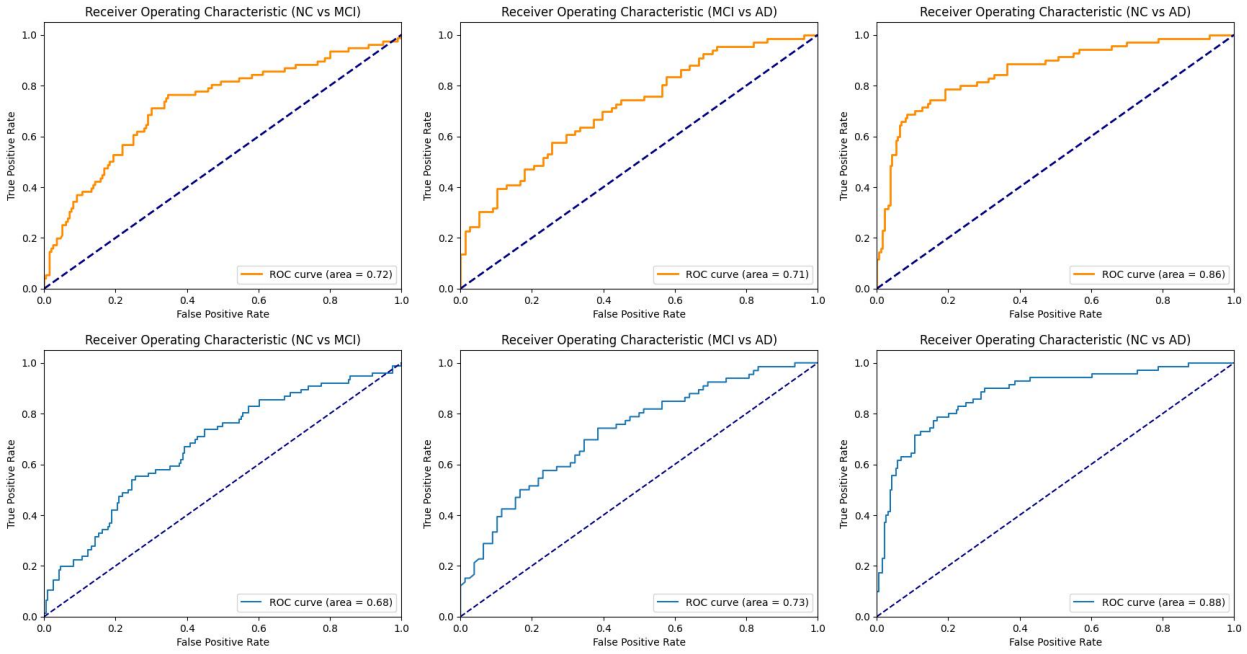


Figure B.6: ROC curves. The first row is by SVM models and the second row is by MLP models. From left to right: NC vs MCI, MCI vs AD, NC vs AD.

Appendix C

Additional tables

Class	Precision	Recall	F1-Score	Support
NC	0.76	0.96	0.84	196
MCI	0.65	0.20	0.30	76
Macro Avg	0.70	0.58	0.57	272
Weighted Avg	0.73	0.75	0.69	272

Table C.1: Classification Report for SVM: NC vs MCI

Class	Precision	Recall	F1-Score	Support
MCI	0.70	0.60	0.65	78
AD	0.60	0.70	0.64	66
Macro Avg	0.65	0.65	0.65	144
Weighted Avg	0.65	0.65	0.65	144

Table C.2: Classification Report for SVM: MCI vs AD

Class	Precision	Recall	F1-Score	Support
NC	0.88	0.92	0.90	189
AD	0.75	0.66	0.70	70
Macro Avg	0.82	0.79	0.80	259
Weighted Avg	0.85	0.85	0.85	259

Table C.3: Classification Report for SVM: NC vs AD

Class	Precision	Recall	F1-Score	Support
NC	0.76	0.86	0.81	196
MCI	0.45	0.30	0.36	76
Macro Avg	0.61	0.58	0.58	272
Weighted Avg	0.67	0.70	0.68	272

Table C.4: Classification Report for MLP: NC vs MCI

Class	Precision	Recall	F1-Score	Support
MCI	0.72	0.65	0.68	78
AD	0.63	0.70	0.66	66
Macro Avg	0.67	0.68	0.67	144
Weighted Avg	0.68	0.67	0.67	144

Table C.5: Classification Report for MLP: MCI vs AD

Class	Precision	Recall	F1-Score	Support
NC	0.87	0.90	0.89	189
AD	0.71	0.63	0.67	70
Macro Avg	0.79	0.77	0.78	259
Weighted Avg	0.83	0.83	0.83	259

Table C.6: Classification Report for MLP: NC vs AD