COMPARING THE EFFECTIVENESS OF GRAPH NEURAL NETWORKS AND MACHINE LEARNING ALGORITHMS FOR FNIRS-BASED NEUROMARKETING RESEARCH

ATAKAN GÜNGÖR

MEF UNIVERSITY
SEPTEMBER 2024

MEF UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING
MASTER'S IN INFORMATION TECHNOLOGIES

M.Sc. THESIS

COMPARING THE EFFECTIVENESS OF GRAPH NEURAL NETWORKS AND MACHINE LEARNING ALGORITHMS FOR FNIRS-BASED NEUROMARKETING RESEARCH

Atakan GÜNGÖR

ORCID NO: 0000-0001-5411-7392

Thesis Advisor: Asst. Prof. Dr. Tuna ÇAKAR

SEPTEMBER 2024

ACADEMIC HONESTY PLEDGE

I declare that all the information in this study is collected and presented in accordance with academic rules and ethical principles, and that all information and documents that are not original in the study are referenced in accordance with the citation standards, within the framework required by the rules and principles.

Name Surname: Atakan GÜNGÖR

Signature:

ABSTRACT

COMPARING THE EFFECTIVENESS OF GRAPH NEURAL NETWORKS AND MACHINE LEARNING ALGORITHMS FOR FNIRS-BASED NEUROMARKETING RESEARCH

Atakan GÜNGÖR

M.Sc in Information Technologies

Thesis Advisor: Asst. Prof. Dr. Tuna ÇAKAR

September 2024, 56 Pages

Functional near-infrared spectroscopy (fNIRS) has some advantages over other brain imaging methods in terms of cost and portability. For this reason, its use in neuromarketing is increasing. However, fNIRS brings some challenges along with its advantages. Due to features such as multichannel measurement and high temporal resolution, the nature of fNIRS data is complex and multidimensional [7]. Neuromarketing researchers have utilized machine learning algorithms to overcome these challenges. When these studies are analyzed, it is seen that successful results have emerged. Machine learning has influenced researchers working on graphs as well as neuromarketing researchers. Thus, graph neural networks have emerged, which allow the application of artificial neural networks to graph data structures. Thanks to the fact that the brain can be modeled as a graph structure using functional connections [14] and the high temporal resolution of fNIRS [7], there are neuroimaging studies using graph neural networks and fNIRS together. However, despite successful results, there is no neuromarketing research using this combination. Therefore, in this study, the performance of graph neural networks in fNIRS-based neuromarketing was analyzed and compared with machine learning algorithms that have been shown to yield successful results in this context. For the comparison, fNIRS measurements of a neuromarketing experiment conducted to determine perceptions toward brands were used. In the experiment, consumers were asked to decide (yes/no) whether the adjective shown with the brand logo was appropriate for the brand. The data set was obtained by cleaning the obtained measurements. First, a supervised machine learning approach was applied to this dataset. After the dataset went through several data preprocessing stages, various algorithms were trained on it. These were K-Nearest Neighbors, Support Vector

Machines, Random Forest, Naive Bayes, and XGBoost algorithms. Then, two different voting classifiers, one for soft voting and one for hard voting, were created from the algorithms that were more successful than the others. After the machine learning approach was completed, the graph neural network approach was applied. The data obtained through fNIRS was transformed into a graph structure using functional connections in the brain. The Pearson correlation coefficient was used to calculate the functional connections. Since a graph was created for each trial of the participants and each graph had a label (yes/no), classification was performed at the graph level. For graph classification, the generated graphs were given as input to graph neural network architectures. The architectures used in the study consisted of Graph Convolutional Network, Graph Attention Network, and Graph Isomorphism Network. Finally, a soft voting classifier was created by combining these architectures. Test accuracy values of all methods were calculated and binomial confidence intervals were added to these values. The comparison results showed that machine learning algorithms generally outperform graph neural networks. Additionally, machine learning models based on ensemble learning have the best scores.

Keywords : Neuromarketing, fnirs, machine learning, graph, graph neural networks.

Numeric Code of the Field : 92404

ÖZET

FNIRS TABANLI NÖROPAZARLAMA ARAŞTIRMALARINDA ÇİZGE SİNİR AĞLARI İLE MAKİNE ÖĞRENİMİ ALGORİTMALARININ KARŞILAŞTIRILMASI

Atakan GÜNGÖR

Bilişim Teknolojileri Yüksek Lisans Programı

Tez Danışmanı: Dr. Öğr. Üyesi Tuna ÇAKAR

Eylül 2024, 56 Sayfa

Fonksiyonel yakın kızılötesi spektroskopinin (fNIRS) maliyet ve taşınabilirlik açısından diğer beyin görüntüleme yöntemlerine göre bazı avantajları vardır. Bu nedenle nöropazarlama alanında kullanımı gittikçe artmaktadır. Ancak fNIRS, sağladığı avantajların yanında bazı zorlukları da beraberinde getirmektedir. Çok kanallı ölçüm ve yüksek zamansal çözünürlük gibi özellikler nedeniyle fNIRS verilerinin doğası karmaşık ve çok boyutludur [7]. Nöropazarlama araştırmacıları, bu zorlukların üstesinden gelebilmek için makine öğrenimi algoritmalarından yararlanmıştır. Bu çalışmalar incelendiğinde başarılı sonuçların ortaya çıktığı görülmüştür. Makine öğrenimi, nöropazarlama araştırmacılarının yanı sıra çizge üzerinde çalışan araştırmacıları da etkilemiştir. Böylece yapay sinir ağlarının çizge veri yapılarına uygulanmasına izin veren çizge sinir ağları ortaya çıkmıştır. Beynin fonksiyonel bağlantılar kullanılarak çizge yapısı şeklinde modellenebilmesi [14] ve fNIRS'in yüksek zamansal çözünürlüğü sayesinde [7], çizge sinir ağları ile fNIRS'in birlikte kullanıldığı nörogörüntüleme çalışmaları mevcuttur. Ancak başarılı sonuçlara rağmen bu kombinasyona yer veren nöropazarlama araştırmasına rastlanmamıştır. Bu nedenle bu çalışmada, çizge sinir ağlarının fNIRS temelli nöropazarlama alanındaki performansı incelenmiş ve bu bağlamda başarılı sonuçlar verdiği görülen makine öğrenimi algoritmaları ile karşılaştırılması yapılmıştır. Karşılaştırma için, markalara yönelik algıları belirlemek amacıyla yürütülen bir nöropazarlama deneyinin fNIRS ölçümleri kullanılmıştır. Deneyde, tüketicilerden marka logosuyla birlikte gösterilen sıfatın markaya uygun olup olmadığına dair karar vermeleri (evet/hayır) istenmiştir. Elde edilen ölçümler temizlenerek veri seti elde edilmiştir. İlk olarak bu veri setine gözetimli makine öğrenimi yaklaşımı uygulanmıştır. Veri seti birkaç veri ön işleme aşamasından geçirilerek çeşitli

algoritmalar için hazır hale getirilmişir. Bu algoritmalar, K-Nearest Neighbors, Support Vector Machines, Random Forest, Naive Bayes ve XGBoost'tur. Sonrasında ise diğerlerine göre daha başarılı olan algoritmalardan, biri soft voting diğeri hard voting olmak üzere iki farklı voting classifier oluşturulmuştur. Makine öğrenimi yaklaşımı tamamlandıktan sonra çizge sinir ağları yaklaşımına geçilmiştir. fNIRS aracılığı ile elde edilen veriler, beyindeki fonksiyonel bağlantılar kullanılarak çizge yapısına dönüştürülmüştür. Fonksiyonel bağlantıların hesaplanmasında Pearson korelasyon katsayısı kullanılmıştır. Katılımcıların her denemesi için bir çizge oluşturulduğundan ve her çizgenin etiketi (evet/hayır) bulunduğundan, çizge seviyesinde sınıflandırma yapılmıştır. Çizgelerin sınıflandırılması için, oluşturulan çizgeler, çizge sinir ağları mimarilerine girdi olarak verilmiştir. Çalışmada kullanılan mimariler, Graph Convolutional Network, Graph Attention Network ve Graph Isomorphism Network'ten oluşmaktadır. Son olarak, bu mimarilerin bir araya getirilmesiyle bir soft voting classifier oluşturulmuştur. Tüm yöntemlerin test accuracy değerleri hesaplanmış ve bu değerlere güven aralıkları eklenmiştir. Karşılaştırma sonuçları, genel olarak makine öğrenimi algoritmalarının çizge sinir ağlarından daha iyi performans verdiğini göstermiştir. Ek olarak, topluluk öğrenimine dayalı makine öğrenimi modelleri en iyi skorlara sahiptir.

Anahtar Kelimeler : Nöropazarlama, fnirs, makine öğrenmesi, çizge, çizge sinir ağları.

Bilim Dalı Sayısal Kodu : 92404

To my dearest family...

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Asst. Prof. Dr. Tuna Çakar, for his guidance, continuous support, and patience throughout my research journey. His expertise and encouragement have been instrumental in shaping this work.

I am also deeply grateful to the members of my thesis committee, Prof. Dr. Adem Karahoca and Asst. Prof. Dr. Yassine Drias, for their insightful comments and constructive criticism. Their valuable feedback has improved the quality of this thesis.

I would like to extend my thanks to Mehtap Taban for her support throughout my research. Her contributions have been essential to the success of this work.

Finally, and most importantly, I would like to dedicate this work to my beloved family. Their unconditional love, encouragement, and sacrifices have been the driving force behind my academic pursuits. I am eternally grateful for their unwavering support.

TABLE OF CONTENTS

ABSTRACT i
ÖZET iii
ACKNOWLEDGEMENT vi
TABLE OF CONTENTS vii
LIST OF TABLES ix
LIST OF FIGURES x
ABBREVIATIONS xi
INTRODUCTION 1
1. THEORETICAL BACKGROUND 6
1.1. Neuromarketing 6
1.2. Functional Near Infrared Spectroscopy (fNIRS) 7
1.3. Machine Learning Approaches 10
1.4. Graphs and Graph Neural Networks 12
2. METHODOLOGY 15
2.1. Data Source 15
2.2. Machine Learning Approach 17
2.2.1. Data Preparation 18
2.2.2. Data Preprocessing 18
2.2.2.1. Data Cleaning 19
2.2.2.2. Outlier Analysis 19
2.2.2.3. Missing Value Analysis 19
2.2.2.4. Data Reduction 20
2.2.3. Model Development 22
2.2.3.1. K-Nearest Neighbors 22
2.2.3.2. Support Vector Machines 23
2.2.3.3. Random Forests 25
2.2.3.4. Naive Bayes 26
2.2.3.5. XGBoost 27
2.2.3.6. Voting Classifier 28
2.3. Graph Neural Networks Approach 29
2.3.1. Graph Construction 30
2.3.2. Graph Classification 31
2.4. Evaluation and Comparison 34

3. RESULTS AND DISCUSSION 38
3.1. Machine Learning Approach 38
3.2. Graph Neural Networks Approach 46
3.3. Evaluation and Comparison 48
CONCLUSION 50
REFERENCES 51

LIST OF TABLES

Table 2.1 : Description and Types of Features……18
Table 2.2 : Hyperparameter Tuning for K-Nearest Neighbors……..…23
Table 2.3 : Hyperparameter Tuning for Support Vector Machine……25
Table 2.4 : Hyperparameter Tuning for Random Forests……26
Table 2.5 : Hyperparameter Tuning for XGBoost……28
Table 2.6 : Details of GNNs……34
Table 3.1 : Descriptive Statistics of Predictive Variables……..……38
Table 3.2 : Descriptive Statistics of Target Variable……40
Table 3.3 : Outliers in Predictive Variables……..…….41
Table 3.4 : Missing Values in Predictive Variables……..….42
Table 3.5 : Distribution of Target Variable in Train and Test Sets……44
Table 3.6 : VIF Values of Features ……..…45
Table 3.7 : Successes of Machine Learning Algorithms……...46
Table 3.8 : Successes of GNN Algorithms……...47

LIST OF FIGURES

Figure 1.1 : The shape of the path of NIR photons from source to detector including different tissue layers [6]. ……..8
Figure 1.2 : An example of multi-channel fNIR spectroscopy [23]. ….……...9
Figure 1.3 : A summary of machine learning approaches and related tasks [30]. …..12
Figure 1.4 : Examples of Euclidean space data (on the left) and non-Euclidean space data (on the right) [38]. ……..…..13
Figure 2.1 : Examples of images in the 4-second stimulus viewing phase (right) and examples of images in the 4-second decision-making phase (left). E stands for "yes", H for "no", and Fikrim Yok (F) means "no opinion" [40]. ……...…16
Figure 2.2 : An example of the sensor pad in the fNIRS system (left) and the positions of the 16 optodes on the cortical surface (right) [11]. ……17
Figure 2.3 : Two examples of margins with their support vectors [32]. ……24
Figure 2.4 : How the Random Forests method creates the final prediction for classification problems [54]. ……...25
Figure 2.5 : Hard Voting Classifier [52]. ……...29
Figure 2.6 : The attention mechanism in GAT (on the left) and the multihead attention where k = 3 (on the right) [61]. ……...….33
Figure 2.7 : An example of a two-dimensional grid search space [64]. ……35
Figure 2.8 : A visualization of the 5-fold cross-validation used for hyperparameter selection [44]. ……..36
Figure 3.1 : Correlation Heatmap of Predictive Variables. ……40
Figure 3.2 : Cumulative Variance of Components. ……...45
Figure 3.3 : One of the Created Graphs. ……47
Figure 3.4 : Test Accuracy with Confidence Intervals. ……..48

ABBREVIATIONS

CW : Continuous Wave

EEG : Electroencephalography

fMRI : Functional Magnetic Resonance Imaging

fNIRS : Functional Near-Infrared Spectroscopy

GAT : Graph Attention Network

GCN : Graph Convolutional Network

GIN : Graph Isomorphism Network

GNN : Graph Neural Network

GNN_VC : GNN Voting Classifier

HbO : Oxyhemoglobin

HbR : Deoxyhemoglobin

HbT : Hemoglobin

HVC : Hard Voting Classifier

KNN : K-Nearest Neighbors

MEG : Magnetoencephalography

NB : Naive Bayes

Oxy : Oxygen saturation

PCA : Principal Components Analysis

RF : Rondom Forest

SVC : Soft Voting Classifier

SVM : Support Vector Machines

VIF : Variance Inflation Factor

XGB : XGBoost

INTRODUCTION

Neuromarketing, which combines marketing and neuroscience, has been a rapidly growing field recently. This field attracts great attention from marketing researchers due to its potential. Lee et al. [1] define neuromarketing as follows: *"Neuromarketing as a field of study can simply be defined as the application of neuroscientific methods to analyze and understand human behaviour in relation to markets and marketing exchanges."*. The neuroscientific methods mentioned in the definition can be used to obtain objective and unbiased consumer data without the biases of traditional marketing research methods such as surveys and interviews. Moreover, these methods contribute to understanding consumer behavior and enable the development of more effective marketing strategies [2], [3], [4]. Brain imaging techniques constitute an important part of neuroscientific methods used to obtain objective data on consumer behavior. Examples of these techniques include electroencephalography (EEG), magnetoencephalography (MEG), functional magnetic resonance imaging (fMRI), and functional near-infrared spectroscopy (fNIRS) [5]. fNIRS has advantages over other brain imaging modalities because it is noninvasive and affordable. In addition, it is less sensitive to motion artifacts and can be used with diverse and large participant samples [6]. Therefore, this study will analyze the data obtained with fNIRS in the context of neuromarketing.

Although fNIRS can offer several advantages, such as portability and affordability, it also presents some challenges. The nature of fNIRS data is complex and multidimensional due to features such as multichannel measurement and high temporal resolution [7]. Neuromarketing researchers have utilized machine learning algorithms to overcome these challenges and uncover patterns and relationships that traditional methods may miss. For example, Bak et al. [8] emphasized that self-reporting tools used to measure impulse buying behaviors have some disadvantages. To overcome these disadvantages, they used a brain-computer interface method based on fNIRS. Their method involves recording participants' prefrontal cortex activation via fNIRS while generating impulse buying behavior in a computer environment. Using five features obtained from fNIRS signals and support vector machines (SVM), they tried to classify impulse buying and non-impulse buying patterns. The results showed that fNIRS signals can detect impulse buying patterns with an average accuracy of 93.78%. Another study involving purchasing tasks investigated the suitability of fNIRS for neuromarketing research. In the study, participants were asked to make purchase or pass decisions for

products in the food, personal care, and cleaning categories. In addition, participants were given a predetermined budget to create a more realistic purchasing process. Thus, it was aimed to prevent biases that may arise when hypothetical options and rewards are given. The fNIRS signals obtained during decision-making were averaged over twelve-second time intervals. Discriminant analysis was applied to the obtained data to predict purchase or pass decisions and a classification accuracy of 0.62 was found. Afterwards, the respondents were divided into two groups as budget-sensitive or non-budget-sensitive. With this additional information, the classification accuracy increased to 0.77 [9].

In the literature, there are also studies that try to determine whether individuals like a stimulus by using machine learning algorithms trained on the measurements obtained with fNIRS. For example, in one study, prefrontal cortex activations were recorded by fNIRS when participants expressed their liking for the stimuli presented to them. The stimuli used in the study consisted of photographs of animals, cars, landscapes, and food. For these stimuli, three classes were determined as attractive/neutral/unattractive. Support Vector Machine (SVM) algorithm was applied to the preprocessed data and 10-fold cross validation was used to evaluate the results. An average classification accuracy of 72.9% was obtained for attractive and others, and an average classification accuracy of 68.3% was obtained for unattractive and others. At this point, it should be emphasized that the study sample consisted of only five participants [10]. Similarly, in another study, fNIRS and machine learning algorithms were used together to classify the likes of individuals. Participants' prefrontal cortex activity was measured while deciding whether they liked the stimuli aesthetically. The K-Nearest Neighbor (KNN), SVM, Random Forest, XGBoost, and LightGBM algorithms were applied to the data obtained from these measurements and their success was compared. In addition, feature extraction and feature selection methods were utilized to improve the prediction success of the model. The F1 scores were used to compare the success of the models and the scores were tested with leave-one-out cross validation. The results show that the LightGBM algorithm is the most successful algorithm with an average F1-score of 0.66. Moreover, the applied feature extraction and selection methods were not effective in improving the success of the model [11].

Another fNIRS-based neuromarketing research using machine learning focused on the use of retro music in advertisements and chocolate consumption. In the first stage of the research, dimensions were created with the help of content analysis from the data obtained through the questionnaire. Then, these dimensions indicating adjectives such as

"reliable" and "affordable" were shown to the participants. The participants were asked to decide whether the dimensions shown were appropriate for the relevant brands. This procedure was repeated twice and participants were shown advertisements with or without retro music between the procedures. During the experiment, fNIRS measurements were made. Two models were developed using the measurements obtained. The first model aims to classify participants according to whether they are consumers of specific chocolate brands or not. The second model aims to classify participants according to whether they watch advertisements with retro music. Eight different algorithms were applied for both models and the better-performing models were combined using the stacking method. The highest test accuracy of 0.73 was obtained for the first model and the highest test accuracy of 0.66 was obtained for the second model [12]. In another study, neuromarketing and political science were brought together. In this study, individuals' perceptions of political leaders were analyzed. For this purpose, male participants from two different political parties took part in the study. During the experiment, positive and negative adjectives were shown to the participants along with party leaders. The participants were asked to decide whether the adjectives depicted the party leaders or not. During these procedures, participants' prefrontal cortex activations were recorded by fNIRS. Various machine learning algorithms, including stacking, were applied to the data to develop a model that categorizes judgments about political leaders. The results showed that LightGBM was the most successful model with a test classification accuracy of 0.78 [13].

In conclusion, when the success of machine learning algorithms in fNIRS-based neuromarketing research is analyzed, it is seen that the classification success ranges between 0.62 and 0.93. Although some studies show values close to 60%, this approach is promising for developing models that can predict neuromarketing tasks based on fNIRS data.

Similar to machine learning, a recently popular research topic is graph neural networks (GNNs), which enable the application of artificial neural networks to graph data structures. The brain can be modeled as a graph structure through functional connections between different brain regions [14]. Furthermore, fNIRS enables functional connectivity analysis thanks to its high temporal resolution [7]. For these reasons, there are neuroimaging studies in the literature that use GNNs and fNIRS together. For example, Yu et al. [15] by combining these two approaches, aimed to develop a method that automates the diagnosis of depression in individuals. For this purpose, participants were

asked to fill out a questionnaire and were labeled as depressed or not depressed according to the scores obtained. Afterwards, the participants were asked to express the names of vegeTables, furniture, and fruits displayed on a computer screen. During this process, fNIRS measurements were made. The data obtained were preprocessed and temporal and spatial features were determined. A graph was created for each participant. Here, temporal features represent the node features of the graphs and spatial features represent the edge weights of the graphs. Graph neural networks were applied to the generated graphs and the results were compared with classical machine learning methods. When the classification accuracy and F1-scores of the algorithms are examined, it is observed that graph neural networks perform better. In another study, different graph neural network models were combined to analyze data obtained with fNIRS. The study used a publicly available dataset in which participants performed one of three different motor execution tasks. These motor execution tasks were considered as separate classes and tried to be classified using fNIRS measurements. For this purpose, a graph was created for each class and these graphs were fed to graph neural networks. The trained models were combined to create a final prediction model. As in the previous study, the success of the model was compared with classical machine learning algorithms. The ensemble model outperformed the classical algorithms by achieving a classification accuracy of 0.75 (5-fold cross validation) [16].

In conclusion, although there are few studies using graph neural networks to classify fNIRS data, the combination of fNIRS and graph neural networks has been shown to give successful results. At the same time, in these studies, graph neural networks were compared with traditional machine learning algorithms and the former approach outperformed the latter. In the neuromarketing literature, there are no studies that use graph neural networks in combination with fNIRS. However, due to their success in classifying fNIRS measurements obtained in different contexts, it is thought that graph neural networks may be promising for fNIRS-based neuromarketing research. Therefore, the main objective of this study is to investigate how graph neural networks will perform in fNIRS-based neuromarketing research and to compare them with classical machine algorithms that have been shown to give promising results in this context. Thus, it is aimed to reveal which of these two approaches can provide more successful results on the available dataset. In order to create the dataset, fNIRS measurements of a neuromarketing study conducted to determine perceptions toward brands will be used. As a second goal in this direction, a model will be developed to classify participants'

decisions about whether they find the adjectives matched with brands appropriate or not. The findings of this study are expected to provide valuable insights into the use of graph neural networks and machine learning algorithms in fNIRS-based neuromarketing research.

The organization of the current study is as follows. In the first section, the theoretical background, including neuromarketing, fNIRS, machine learning, and graph neural networks, is provided. In the second section, the data source used in the study, the data preparation and modeling stages in both approaches, and the evaluation method are explained in detail. In the third section, the results of the data preprocessing steps in the approaches and comparisons are shared and discussed. Finally, the study is concluded, and recommendations for future research are provided.

1. THEORETICAL BACKGROUND

1.1. Neuromarketing

Neuromarketing is a new field that bridges marketing and neuroscience and provides information exchange in order to understand consumer behavior and guide marketing [17]. The concept of neuromarketing was first used in 2002 with the definition of “studying the brain mechanism to understand consumer behavior and improve marketing strategies” [18]. Although the definition of neuromarketing was first used by Ale Smidts in 2002, it has been observed that companies started to incorporate neuroscience methods such as fMRI into their research strategies to examine the purchasing behavior of consumers at earlier dates, and in fact, the field began to form organically at an earlier date [19].

Billions of dollars are invested in advertising and marketing campaigns every year. Traditional methods for testing and predicting the effectiveness of these investments are known to fall short. This is because traditional methods are influenced by the willingness and competence of the participants. Neuroscience techniques, on the other hand, provide methods that can directly examine how the consumer's mind reacts when exposed to marketing strategies and can measure conscious as well as unconscious reactions [17]. When the techniques used in neuromarketing research are categorized according to the measured activity, they are grouped under 3 different headings: “Cognitive Activity”, “Autonomic Nervous System Activities” and “Somatic Nervous System Activities” [5].

In Lim's study [5], there are twelve methods in total, including six different methods to measure cognitive activities, four different methods to measure autonomic nervous system activities, and two different methods to measure somatic nervous system activities. The techniques measuring cognitive activities are Electroencephalography (EEG), Magneto Encephalography (MEG), Functional Magnetic Resonance Imaging (fMRI), Functional Near Infrared Spectroscopy (fNIRS), Positron Emission Tomography (PET) and Steady State Topography (SST). Techniques used to measure autonomic nervous system activities include Pupil Analysis (Pupilla), Facial Electromyography (fEMG), Heart Rate Response Analysis (Heart Rate) and Galvanic Skin Reaction (GSR). Eye Tracking and Facial Coding (FC) methods are used to measure somatic nervous system activities. Neuroscience methods used in the field of neuromarketing provide

access to background data that is not even recognized at the level of consciousness through the observation of biological reactions instead of directly asking about thoughts, emotions, evaluations, and decision-making processes as used in traditional methods. This data helps to provide a different and new ground for marketing strategies [5].

One of the important topics that neuromarketing focuses on is the reward system in the human brain. It is seen that neuromarketing studies play a major role in designing marketing efforts in a way that encourages the activation of the brain's reward system [20]. The concept examined in neuromarketing research can be implicit concepts involving cognitive and emotional processes or observable concrete behaviors. Research may focus on one or a combination of pre-purchase, intra-purchase, and post-purchase processes, and the concept under investigation may be involved in any one or more of these processes. Concepts such as attitude, attention, trust, choice decision, and recommendation are examples of concepts that are examined in research. These concepts are tested in different ways offered by neuroscience and used to predict and control the effects on outputs such as customer awareness, visits, sales, and brand equity [5].

Although it has been controversial in the years it emerged, it has rapidly gained attention and adoption among marketing experts [17]. Neuromarketing has a significant potential to increase the effectiveness of advertising and other marketing strategies used for many purposes around the world [17]. On the other hand, the information obtained from neuromarketing research can be used in both commercial and non-commercial areas, such as making sales or influencing behavior that contributes to the common good [5]. Studies in the field of neuromarketing continue to increase day by day, applications are becoming widespread, and findings in the field are gaining importance for marketing research [20].

1.2. Functional Near Infrared Spectroscopy (fNIRS)

Functional Near Infrared Spectroscopy (fNIRS) is a neuroimaging method that non-invasively measures changes in cerebral hemodynamic activity to provide information about brain functions [21]. Hemoglobin is a protein found in red blood cells and is called oxyhemoglobin (oxy-Hb) or deoxyhemoglobin (deoxy-Hb) depending on whether it is loaded with oxygen or not [22]. fNIRS can detect changes in oxy-Hb and deoxy-Hb concentration with a high temporal resolution, and can do this thanks to the emission and absorption of near-infrared (NIR) lights [23]. NIR light has the ability to

penetrate biological tissues to varying depths, depending on the tissue type and properties. It can effectively pass through the scalp and skull to reach the underlying brain tissue [24]. Hemoglobin plays an important role in the absorption of NIR light, and this chromophore is the main reason for the relatively high attenuation in the lights [25]. Additionally, oxy-Hb and deoxy-Hb show different absorption properties when exposed to NIR light [6].

When neurons in a certain region of the brain are activated, oxy-Hb decreases and deoxy-Hb increases. As a result, a hemodynamic response occurs within a few seconds. Thus, the blood flow system has been activated to meet the oxygen and glucose needs in the region. The hemodynamic response is followed by an increase in cerebral blood flow and expansion of the peripheral vascular bed. This reduces deoxy-Hb in small vessels and increases oxy-Hb in blood capillaries and vascular beds [26]. Said changes in hemoglobin concentration can be predicted by changes in NIR light attenuation. Changes in attenuation can be measured via fNIRS [6].

Recording of cortical hemodynamic activity is achieved by placing a lot of optical fibers (optodes) on the scalp. NIR lights carried by some of the optical fibers to the scalp are released through sources. Other optical fibers collect the lights coming from the scalp and send them to detectors for measurement. Source and detector pairs form measurement points referred to as channels [23]. When NIR lights emitted from a source reach a detector, they pass through various tissue layers (scalp, skull, cerebrospinal fluid, brain) with different scattering properties. This causes the NIR lights to travel in a diffuse manner and therefore the sensitivities of the source-detector combinations to be banana-shaped (Figure 1.1) [25].

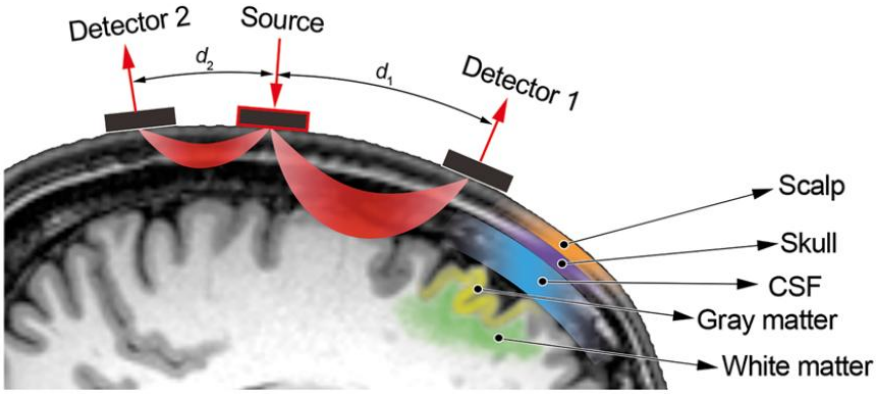

Figure 1.1 : The shape of the path of NIR photons from source to detector including different tissue layers [6].

The fNIRS technology that has become standard today is multi-channel fNIR spectroscopy containing many optodes (Figure 1.2) [23] and has three different types: frequency domain (FD), time domain (TD) and continuous wave (CW) [26]. fNIRS devices using the CW method have some advantages over other methods. Since they are technologically simpler devices, they offer economy and portability [25]. Because they cannot distinguish and measure the contributions of absorption and scattering, they can only provide information about changes in oxy-HB and deoxy-HB concentrations; they cannot determine absolute basal concentrations. However, this limitation does not prevent CW-based fNIRS devices from being ideal for cognitive neuroscience applications because functional activity is often measured with respect to a baseline and absolute concentrations are not needed [6]. In CW-based fNIRS devices, the modified Beer-Lambert law is used to convert changes in oxy-HB and deoxy-HB into oxygenation measurements starting from zero [25].

Figure 1.2 : An example of multi-channel fNIR spectroscopy [23].

fNIRS has been compared not only across its species but also with well-established methods in brain imaging. For example, Lai *et al.* [22] listed the advantages of fNIRS as follows:

- Since it is not a motion-sensitive device, it is applicable to samples that include individuals such as mobile patients and young children.
- Patients can undergo multimodal (e.g., EEG and fNIRS) assessments because other methods such as fMRI, PET, and EEG do not interfere with the light used in fNIRS.
- Although its spatial resolution is not as good as fMRI, it can be used to obtain spatial information about the topography of the brain by matching source and detector probes.

- Although it is not as good as EEG, it can offer a reasonable temporal resolution thanks to its high light speed and high sampling rate.
- Three-dimensional brain mapping software can be used with some fNIRS devices to reconstruct a 3D image through measurement data based on 3D coordinates, which can improve diagnosis by making it easier to see topographic representations of brain activation.

Like other brain imaging methods, fNIRS has some inherent limitations. For example, the depth of penetration is limited (1.5–2 cm) and may not provide detailed information about specific brain regions. Additionally, fNIRS signals can be affected by systemic interferences such as changes in blood pressure, heart rate, and skin blood flow, resulting in noise added to the data [6]. Despite the mentioned disadvantages, fNIRS remains a valuable tool in many areas of research. Thanks to significant advances in hardware and analysis techniques over the last three decades, this method makes significant contributions to various research fields, including psychiatric disorders, neurodevelopment, and cognitive neuroscience [21].

1.3. Machine Learning Approaches

Machine learning, which represents a very important breakthrough in computers learning from data, is defined by Ethem Alpaydın [27] as follows: “Machine learning is programming computers to optimize a performance criterion using example data or past experience". The use of sample data or experience expressed in the definition is included in the training process of machine learning algorithms. In this process, target results are provided along with samples of the input data, and as a result, the algorithms can adjust themselves in the best way possible. Once training is finished, algorithms can gain the ability to generalize and produce the desired result from existing as well as unseen data. Since the process in question does not require explicit coding, these algorithms can be considered "soft coded". Algorithms that perform successfully can detect much more complex patterns in existing data compared to a typical human [28]. For this reason, machine learning can find its place in a wide variety of use cases. In addition to computer vision and natural language processing, it achieves successful results in many fields such as health, cyber security, e-commerce, and finance [29], [28], [30], [31].

Machine learning approaches consist of four different types: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning [30]. The most common of these types is supervised learning [29] and is used to predict unknown input-output pairings from known input-output pairings (each output has a label) [28]. The most popular tasks in supervised learning are classification and regression [30]. In the classification task, the models obtained by training algorithms on labeled training data are used to predict categorical (discrete) labels of previously unseen objects. In the regression task, the trained models try to predict unknown numerical data values [32].

Another machine learning approach is unsupervised learning, in which the label information of the input data is not available. Therefore, the learning process of this approach is called unsupervised [32]. Examples of tasks where unsupervised learning is commonly used include association rule extraction, clustering, anomaly detection, and dimensionality reduction [30]. Semi-supervised learning is a mixture of supervised and unsupervised learning. In this approach, some of the data is labeled and the labeled data helps to learn the unlabeled ones. This process is more similar to the way humans learn new skills and fits most natural processes [28]. Tasks where the semi-supervised learning approach is frequently used include data labeling, machine translation, and fraud detection [30].

Finally, reinforcement learning is a machine learning approach in which an agent learns to make decisions by interacting with a specific environment. The agent performs actions and receives reward or punishment feedback based on these actions. Through this trial and error process, the agent learns to maximize cumulative rewards by discovering the most appropriate strategy for interacting with the environment [33]. The reinforcement learning approach can be applied to improve the efficiency of complex systems through automation in areas such as manufacturing, autonomous driving, and supply chain logistics [30]. A summary of machine learning approaches and related tasks is given in Figure 1.3 below.

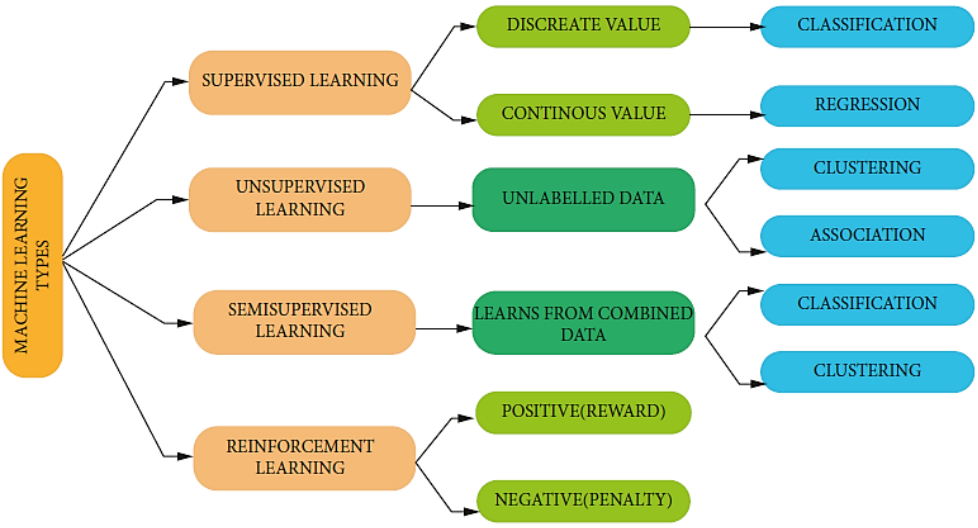

Figure 1.3 : A summary of machine learning approaches and related tasks [30].

1.4. Graphs and Graph Neural Networks

Graphs are structures that contain vertices and edges. In these structures, vertices, also known as nodes or points, are connected by links, called edges. A graph (G) is defined by two sets, one each for vertices (V) and edges (E), and is denoted as G = (V, E) [34]. Graphs are divided into different types according to factors such as whether the edges are directional (directed/undirected), whether the vertices and edges are of different types (homogeneous/heterogeneous). All of these types provide additional information about the graph structure and can be used together because they are independent of each other [35]. As an important data type, graphs can find their place in many different real-world contexts. Examples of these context representations include knowledge graphs, user interest graphs in electronic commerce, citation graphs in research fields, and social graphs in social media networks. Users who can effectively analyze graph data structures can understand the hidden meaning within these structures more deeply [36].

Data produced in areas such as e-commerce and social media can be successfully represented by graphs, but these structures are difficult to analyze with existing machine learning and deep learning algorithms. This is because the graph representations of said fields are in non-Euclidean space. Traditional machine learning and deep learning algorithms are designed for Euclidean space data. Euclidean space states that a group of points in an n-dimensional linear space can be used to represent data [37]. Since graphs are non-Euclidean data, the distance between two nodes may not be equal to the distance between the coordinates of these nodes in Euclidean space [38]. Converting graphs from non-Euclidean space to Euclidean space causes the loss of important information such as

relationships between nodes [37]. The differences between Euclidean space data and non-Euclidean space data are exemplified in Figure 1.4.

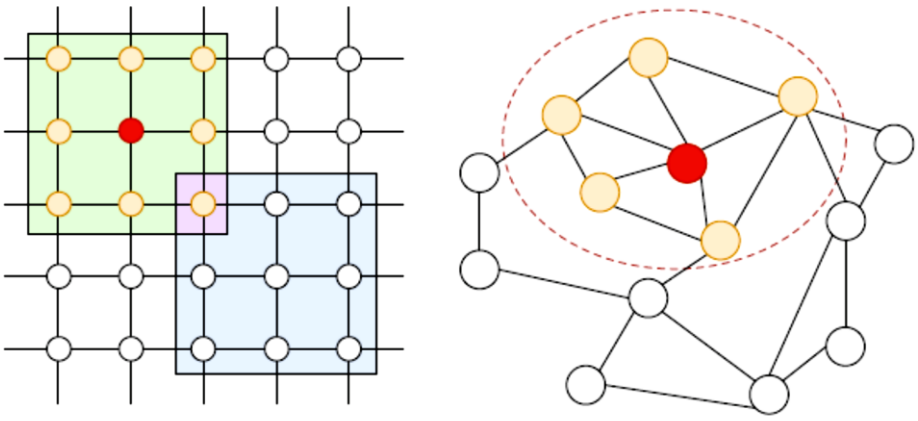

Figure 1.4 : Examples of Euclidean space data (on the left) and non-Euclidean space data (on the right) [38].

Graph neural networks have been developed to solve problems that arise due to the complexity of graph data. This approach extends the capabilities of traditional neural networks, allowing neural networks to be applied to complex graph structures. Thus, the mentioned problems are tried to be solved. Graph neural networks are a class of neural networks that utilize message passing between nodes to detect dependencies in graphs [35]. Message passing scheme is a process that enables nodes in the graph to exchange information with neighboring nodes and can be summarized with the following steps [38]:

- Each node in the graph is represented by a feature vector that characterizes it. These vectors serve as the starting point for the message transfer process.
- At each iteration of the message passing process, all nodes collect information from their neighbors. The information collected includes both structural information about the graph and feature-based information.
- After collecting information from neighboring nodes, each node updates its feature vector according to the collected information. This update function involves combining the existing feature vector of the node with the collected information to create a new feature vector of the node.
- The message passing mechanism is implemented iteratively, and nodes exchange information with their neighbors in multiple rounds. As iterations progress, each node's feature vector contains information from a larger

neighborhood in the graph. Thus, node vectors capture more and more complex relational information.

Training of graph neural networks can be achieved in three different ways. These are supervised learning, where the nodes used in training are labeled, semi-supervised learning, where a small portion of the nodes are labeled but a large portion is unlabeled, and unsupervised learning, where the nodes are unlabeled [35]. The training of graph neural networks can also be classified according to the neighborhood selection. These classes consist of transductive and inductive learning. In transductive learning, neighbors of training nodes can be members of validation and test sets. Here, the task of the model is to try to predict the labels of the nodes that have been seen before in training. In Inductive learning, the neighbors of training nodes include only other training nodes, and testing of the model is performed on unseen data [39]. These learning approaches can be used in node-level, edge-level, and graph-level graph learning tasks. The purpose of node-level tasks is to predict certain properties of nodes, and node classification, node regression, and node clustering are applications at this level. In edge-level tasks, the relationships between nodes in a graph are examined, and edge classification and link prediction are applications at this level. In graph-level tasks, it is aimed to estimate the features related to the entire graph, and graph classification, graph regression, and graph matching are applications belonging to this level [38].

2. METHODOLOGY

The main objective of the current study is to compare traditional machine learning methods and graph neural networks in fNIRS-based neuromarketing research. Firstly, information about the data source used for this purpose will be provided. Then, the machine learning and graph neural network approaches will be explained in detail along with the data preparation and analysis stages. Finally, the evaluation method and the comparison procedure will be introduced.

2.1. Data Source

The data which the approaches were compared were obtained from a neuromarketing experiment using fNIRS brain imaging [40]. Participants in the experiment consisted of men and women without eye disease and mental illnesses and who had not taken psychiatric medication in the past six months. The data of 87 of these participants were used in the analyses in the current study.

The researchers asked the participants to decide whether the brand logo and adjective pairings shown together during the experiment were compatible with each other. Each participant saw 30 brand adjective pairs and indicated their decision by pressing the directional keys of the keyboard (right: yes, bottom: no opinion, left: no). The right and left directional keys were randomly switched on each trial to avoid lateralization of brain function. In each of the trials, participants were given 4 seconds to see the brand-adjective match. They were then asked to make a decision within 4 seconds. Thus, the duration of all trials (periods of tasks) for each participant reached 4 (8 x 30) minutes. Two examples of the images forming the brand adjective pairs are shown in Figure 2.1.

Figure 2.1 : Examples of images in the 4-second stimulus viewing phase (right) and examples of images in the 4-second decision-making phase (left). E stands for "yes", H for "no", and Fikrim Yok (F) means "no opinion" [40].

The researchers used fNIRS brain imaging to measure oxygenation changes in the participants' prefrontal cortex during the stimulus-seeing and decision-making phases of the experiment. The fNIRS system used in the study is from the continuous wave fNIRS class and was developed by fNIR Devices LLC [41]. The system consists of three components: a sensor pad (a flexible head with 4 light sources and 10 detectors for the creation of 16 optodes), a control unit (for hardware management), and a computer with COBI Studio software (for data collection). The distance between the source and detectors on the sensor pad is 2.5 cm, and thanks to this positioning, the lights reach a tissue depth of 1.25 cm. The three-component fNIRS system can perform four different types of measurements with a temporal resolution of 2 Hz (2 samples per second). These are:

- Oxyhemoglobin (HbO), which refers to oxygen-laden hemoglobin,
- Deoxyhemoglobin (HbR), which refers to hemoglobin that releases its oxygen charge,
- Hemoglobin (HbT), which refers to the sum of oxyhemoglobin and deoxyhemoglobin,
- Oxygen saturation (Oxy), which refers to the percentage of oxygenated hemoglobin in the blood.

The cortical surface areas corresponding to the 16 optodes forming the measurement channels are the Brodmann regions BA10, BA44, and BA45 [42]. Figure

2.2 below shows the positions of the 16 optodes on the cortical surface and an example of the sensor pad in the fNIRS system.

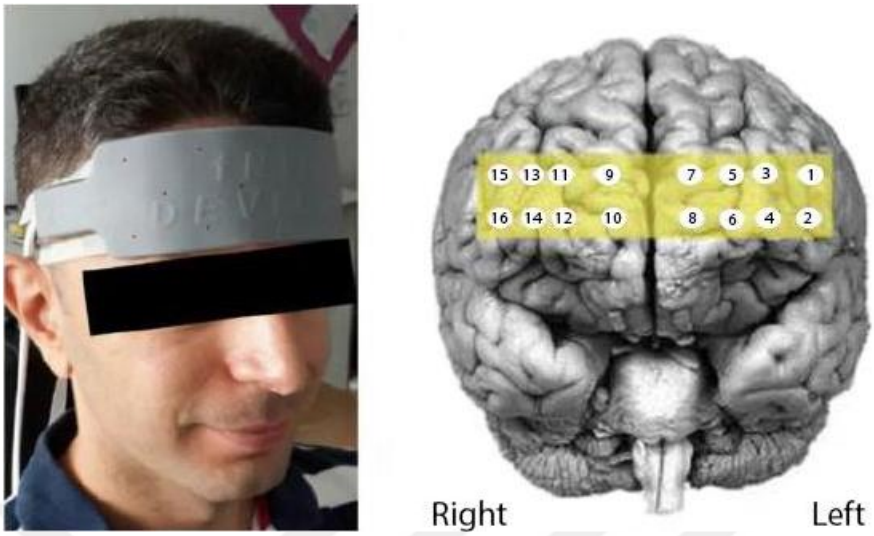

Figure 2.2 : An example of the sensor pad in the fNIRS system (left) and the positions of the 16 optodes on the cortical surface (right) [11].

To obtain the HbO, HbR, Hbt, and Oxy measurements, the researchers subjected the raw data from 16 optodes in the brain imaging system to several pre-processing steps. First, a 20th-order, 0.1 Hz low-pass filter was applied, which plays an important role in reducing high-frequency disturbances caused by respiratory and cardiac activities [43]. In addition to this filter, a sliding windows motion artifact filter [42] was also used to detect and remove motion artifacts.

In the current study, after the acquisition of measurements associated with hemoglobin concentration, additional data preprocessing steps specific to the approaches used to solve the research question were applied. These steps will be detailed separately for both approaches in the following sections.

2.2. Machine Learning Approach

In the previous section, details of the fNIRS-based neuromarketing experiment, which provided the data used in the study, were given. This section will present explanations of how the machine learning approach is applied to the data obtained as a result of the detailed experimental procedure. The data preparation, data preprocessing, and model development stages required for the machine learning approach will be examined respectively.

2.2.1. Data Preparation

As mentioned in the experimental procedure, participants saw the brand-adjective matching image for 4 seconds and then decided whether the match was compatible or not during the next 4 seconds. During this 8-second period, fNIRS was used to continuously measure the oxygenation of the participants' prefrontal cortex. Thus, 8-second HbO, HbR, Hbt, and Oxy values were obtained for each trial of the participants towards the stimuli (brand-adjective images). In the current study, the rows representing the trials were created by averaging the eight-second measurements obtained for each concentration. Since 87 participants in the experiment performed 30 trials, a total of 2610 (87 x 30) rows were obtained. The number of features was determined by the number of channels in the fNIRS device. Since there are 16 optodes on the sensor pad, a total of 64 (4 x 16) attributes were created.

After the attributes were obtained, the target variable was created by coding the yes, no and no opinion decisions indicated by the participants with the directional keys of the keyboard as 0, 1 and 2, respectively. With a three-class categorical variable, the problem at hand became a supervised machine learning task, and the target variable was attempted to be predicted using 64 features obtained from physiological measurements. The dataset took its final form with the addition of participant and stimulus columns to the target variable and 64 physiological measurements. Table 2.1 below provides information about the attributes in the dataset.

Table 2.1: Descriptions and Types of Features

Feature	Type	Description
SUBJECT	C	Numbering of participants from 1 to 96
STIMULUS	C	Numbering of stimuli which were shown to participants from 1 to 30
RESP	C	Decision on brand-adjective matching (yes, no, or no opinion)
HBO_Optode[1-16]	N	Oxyhemoglobin, which refers to oxygen-laden hemoglobin (from 1st optode to 16th)
HBR_Optode[1-16]	N	Deoxyhemoglobin, which refers to hemoglobin that releases its oxygen load (from 1st optode to 16th)
HBT_Optode[1-16]	N	The sum of oxyhemoglobin and deoxyhemoglobin (from 1st optode to 16th)
OXY_Optode[1-16]	N	Oxygen saturation, which refers to the percentage of oxygenated hemoglobin in the blood (from 1st optode to 16th)

2.2.2. Data Preprocessing

In this section, the data preprocessing operations performed on the obtained data set will be detailed. The data preprocessing stages are organized as data cleaning, outlier

analysis, missing value analysis, and data reduction. All of these stages were performed through Python.

2.2.2.1. Data Cleaning

The supervised machine learning task in this study aims to classify consumers' yes or no decisions from the physiological measurements made by fNIRS. For this reason, rows expressing “no opinion” were removed from the dataset. For the same reason, the participant and stimulus columns were also removed from the dataset. It was also examined whether there were rows and columns with more than half missing values. It was found that there were such rows in the dataset, and they were removed from the dataset on the grounds that they did not provide sufficient information.

2.2.2.2. Outlier Analysis

It was aimed to use machine learning algorithms to predict consumers' decisions based on available physiological measurements. Since some machine learning algorithms are sensitive to outliers, outlier analysis was performed. At the same time, it was also aimed to reduce the noise in the dataset. Outlier analysis involves detecting and dealing with outliers. In the detection phase, the IQR method was utilized. The equations of this method are given below.

$$IQR = Q3 - Q1 \tag{2.1}$$

$$Lower\ Boundary = Q1 - 1.5 \times IQR \tag{2.2}$$

$$Upper\ Boundary = Q3 + 1.5 \times IQR \tag{2.3}$$

All predictor variables were examined one by one and values above the upper bound and values below the lower bound were identified as outliers. As a result of this process, it was observed that there were many rows containing outliers in the dataset. Therefore, instead of removing these rows from the dataset, outliers were marked as NA. Thus, outliers were considered as missing data to be assigned later. The assignment process will be described in detail in the next section.

2.2.2.3. Missing Value Analysis

Missing values can occur for many reasons such as loss or forgetting. The implementation of the machine learning algorithms in this study on the existing dataset

was performed with the help of the scikit-learn library [44]. The implementation of most of these algorithms in the scikit-learn library does not accept missing values. Therefore, missing value analysis was applied. When the data set was examined, it was seen that there were a large number of missing values. Due to this high amount of missing values, instead of deleting the rows with missing values, these values were imputed along with the outliers described in the previous section.

Iterative imputation was used as the imputation method. In this method, the feature containing the missing values is considered as Y (target) and the other features as X (predictor). A regressor is trained using features X together with the available values of Y. The trained regressor is used to predict the missing values in Y. These steps are applied for all features in the dataset with missing values. After imputations are made for the features with missing values, these steps are repeated a certain number of times. The results of the last iteration are used as the final imputation [45]. The iterative imputation method was implemented via the IterativeImputer class in the scikit-learn library [44]. The Random Forest [46] algorithm was used as a regressor. This algorithm will be described in detail in the model development section.

2.2.2.4. Data Reduction

As mentioned in the data preparation section, the dataset contains 64 features identified as predictors. High dimensionality increases the likelihood of redundant or irrelevant features being present in the dataset. Such features can cause noise and make it difficult for the model to learn the true relationships between variables. Machine learning algorithms that use the most relevant features can better capture patterns in the dataset. Additionally, thanks to the reduction in the number of features, algorithms require less computing power and can be trained faster. For these reasons, it was decided to apply feature selection on the predictor features. Furthermore, before feature selection, the relationships between the predictor features were examined by Spearman correlation analysis, which showed that there are highly correlated features (results will be presented in the results and discussion section). Feature selection is also suggested as a solution to the problem of multicollinearity that such features can cause.

In the feature selection phase, forward and backward sequential feature selection were used, both in a floating fashion. In forward sequential feature selection, the aim is to iteratively identify the next best feature to be added to the selected list of features.

Initially, there are no features in the feature list. Predictor features are trained with the target feature one by one and their successes are evaluated. The predictor feature with the highest success is selected as the first feature. Then, one more feature is added to the selected feature in the same way and this process is repeated until the predetermined number of features is obtained. Backward sequential feature selection works on the same principle. However, unlike forward selection, all predictor features are added to the feature list and removed one by one until the desired number of features is reached [47].

Forward and backward sequential feature selection methods can be affected by the order of variables since they always move in the same direction. In order to prevent this situation, floating versions of the mentioned methods have been proposed. In these versions, the steps of adding or removing features in the opposite direction have been added to the feature selection method. However, adding the feature in these steps depends on the success of the feature list that follows. Thus, more subsets can be created from the features [48].

The mlxtend library [49] was used to apply the described feature selection methods to the dataset in the study. As a result, 9 features in the forward selection method and 5 features in the backward selection method emerged. These feature sets were determined by the methods according to the selected success metric. The details of the selection process will be given in the results and discussion section. Combining features from different feature selection methods can make models less tendentious to overfit to a particular feature set. Thus, more robust and more generalizable models may emerge. For this reason, the union of the feature sets suggested by both methods was taken. As a result, the number of predictive features was reduced from 64 to 10.

After the feature selection process, the variance inflation factors (VIF) of these 10 features were calculated and a multicollinearity check was performed again. As a result of the examinations, it was observed that some features have VIF values above 5 (the results will be given in the results and discussion section). Therefore, principal component analysis (PCA) was applied to the obtained features. With PCA, a multivariate dataset can be decomposed into a certain number of consecutive orthogonal components that together can explain the maximum percentage of the variance [50].

The scikit-learn library [44] was used to apply PCA to the 10 features from the feature selection phase. As many components as the number of features were obtained.

The components were ranked from largest to smallest according to the amount of variance explained. An XGBoost algorithm [51] was trained using the component with the largest explained variance rate and the target feature together and its success was evaluated. This process was repeated by adding the components with the next highest explained variance rate to the first component one by one. It was observed that the list of components that achieved the highest success consisted of the first 9 components. Therefore, these first 9 components were used for training machine learning algorithms. The XGBoost algorithm involved in the creation of the component list will be explained in the model development section.

2.2.3. Model Development

As mentioned in the previous sections, the aim of this study is to compare the success of machine learning algorithms and graph neural networks in fNIRS-based neuromarketing research. For this reason, various classification algorithms were trained on 9 components obtained after data preprocessing stages and their success was evaluated. These are K-Nearest Neighbours, Support Vector Machines, Random Forest, Naive Bayes, and XGBoost algorithms. Afterward, two different Voting Classifiers, one for soft voting and one for hard voting, were created from the algorithms that are more successful than the others. In this section, the classification algorithms used will be described along with the hyperparameter settings. The details of the success evaluation will be presented in the evaluation and comparison section.

2.2.3.1. K-Nearest Neighbors

The K-Nearest Neighbors (KNN) algorithm is a supervised learning method that classifies data points by considering their similarities. This method searches training data points to estimate the class label of an unknown data point. The aim here is to detect the training points that are closest (most similar) to the unknown data point. The detected training points constitute the nearest neighbors of the point to be classified. The “K” in the algorithm name determines how many nearest neighbors there will be. The most common class label in the neighborhood determines the class label to which the unknown data point will be assigned [32]. In the case of equality between the class labels, different weights are assigned to the labels, or the most common class label is arbitrarily decided. In order to reduce the probability of encountering equality, the number of neighbors is usually chosen as an odd number [27].

KNN may have low accuracy when irrelevant or noisy features are present in the dataset. This is because when making comparisons, it makes use of distance metrics that give equal weights to the features that define the data points. In order to eliminate the mentioned problem, the algorithm has been provided with the ability to prune noisy data points and weight features. At the same time, KNN has the ability to produce predictions for data points with continuous values. During such a task, it assigns the average value of the training data points in the neighborhood to the unknown data point [32].

In the present work, the implementation of the KNN algorithm was performed through the sckit-learn library [44]. Hyperparameter tuning was made for the algorithm used. The tuned hyperparameters consist of the number of neighbors (n_neighbors), the weights given to the data points in proportion to the inverse of their distance (weights), and the distance metric (metric). The tuned values of the hyperparameters are given in Table 2.2 below.

Table 2.2 : Hyperparameter Tuning for K-Nearest Neighbors

n_neighbors	1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21
weights	"uniform", "distance"
metric	"euclidean", "manhattan"

2.2.3.2. Support Vector Machines

Support Vector Machine (SVM), one of the most commonly used machine learning algorithms, can be adapted to many tasks. These tasks include regression, outlier detection, and classification. In classification tasks, it can work powerfully both in cases where the data can be linearly separated and in cases where it cannot be separated [52]. The working principle of the SVM algorithm can be summarized in two steps. In the first step, it transforms the training data points to a higher dimension using a nonlinear mapping. In the second step, it searches for a linear hyperplane within the resulting dimension to be used as the optimal decision boundary. If a sufficiently high dimension is reached with the help of nonlinear mapping, it is always possible for a hyperplane to separate the training data points according to the two classes at hand. To reveal this hyperplane, the SVM algorithm takes certain training data points (support vectors) as a basis and exploits the margins determined by these data points [32]. Figure 2.3 below shows two examples of margins with their support vectors.

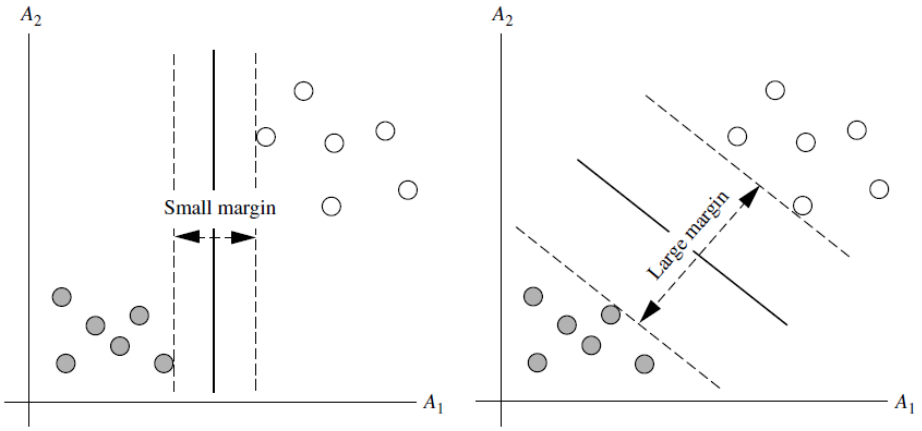

Figure 2.3 : Two examples of margins with their support vectors [32].

The concepts of hard margin and soft margin highlight an important difference in how the SVM algorithm performs the classification task. In hard margin classification, the training data points are not allowed to lie within the margin. This approach is sensitive to outliers and requires the training data points to be linearly separable. In the soft margin approach, the training data points are allowed to lie within the margin even if they are misclassified. Thus, the model is stretched and it is aimed to avoid the problems that arise in hard margin classification [52].

Linear SVM can give good results on linearly separable datasets, but some datasets are not linearly separable. Adding new features can provide a linear separation of such datasets. One way to add new features is to use polynomial features. This approach requires a high polynomial degree to give successful results on complex datasets. However, high polynomial degrees significantly increase the computational cost. A mathematical technique called kernel trick has been proposed to solve this problem. Thanks to this technique, operations can be performed without the need to add polynomial features to the data set (polynomial kernel). Thus, an excessive increase in the number of features is prevented. Another way to add new features is to create features through similarity functions. Gaussian Radial Basis Function (RBF) is one of these similarity functions. As in the first approach, adding features calculated with RBF to the dataset increases the computational cost. In this case, the kernel trick can also be used to perform the operations without the need to add features (RBF Kernel) [52].

In the present work, the implementation of the SVM algorithm was performed through the sckit-learn library [44]. Hyperparameter tuning was made for the algorithm used. The tuned hyperparameters consist of regularisation (C), kernel function (kernel),

and kernel function-specific parameters (degree and gamma). The tuned values of the hyperparameters are given in Table 2.3 below.

Table 2.3 : Hyperparameter Tuning for Support Vector Machine

kernel	**poly**	**C**	0.1, 1, 10, 100
		degree	2, 3, 4, 5, 6
	rbf	**C**	0.1, 1, 10, 100
		gamma	"scale", 1, 0.1, 0.01, 0.001, 0.0001

2.2.3.3. Random Forests

First proposed by Brieman [46], Random Forest (RF) is a learning algorithm that exploits the diversity and collective knowledge provided by many trees to make accurate predictions in classification and regression tasks. Decision trees [53] are one of the most popular combined algorithms, and a decision forest is formed by combining these trees. Random Forests adds randomness to the decision forest to improve accuracy. It trains a large number of decision trees using random subsets of the training data or features in the dataset. It then combines the predictions from the trained decision trees. Thus a large improvement in accuracy can be seen [27]. Combining predictions is accomplished by using majority voting in the classification task, and by averaging the prediction values in the regression task [46]. Figure 2.4 below shows how the Random Forests method creates the final prediction for classification problems.

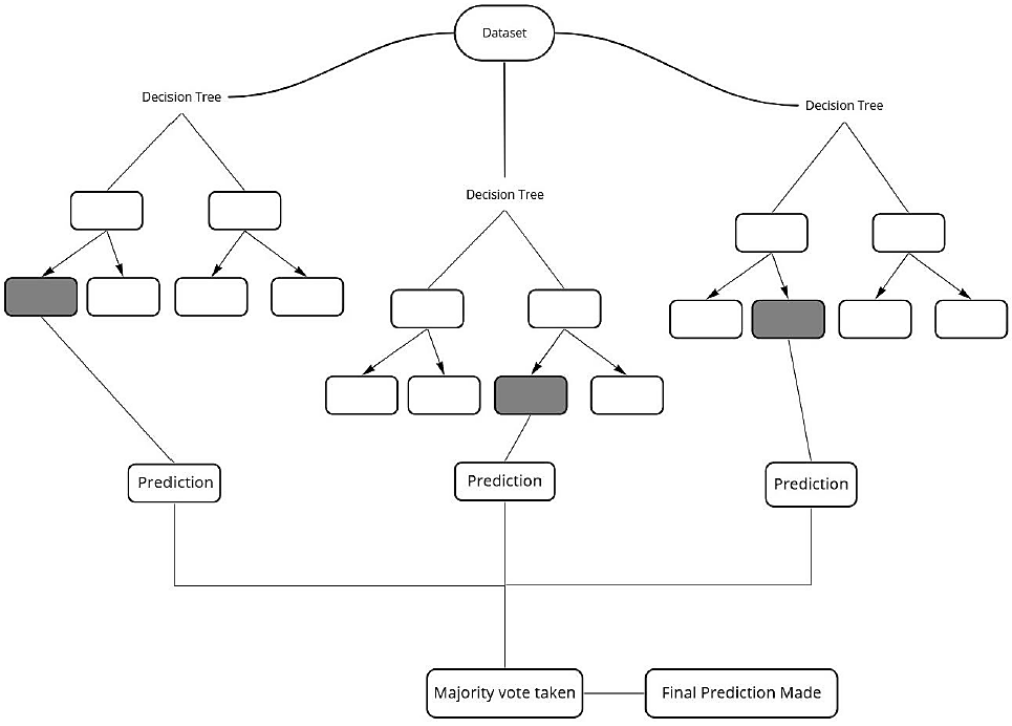

Figure 2.4 : How the Random Forests method creates the final prediction for classification problems [54].

There is no overfit problem in the Random Forest algorithm. This is because generalization error converges when there are many trees in a forest. Another issue is that the strength of the classifiers and their dependencies on each other play an important role in determining the accuracy of the Random Forest. The most desirable situation here is to keep the classifiers strong while reducing their correlations. Furthermore, Random forest algorithms can work efficiently on very large datasets. This is because, in each of the splits, they use very few of the features in the dataset [32].

In the present work, the implementation of the Random Forest algorithm was performed through the sckit-learn library [44]. Hyperparameter tuning was made for the algorithm used. The tuned hyperparameters consist of the number of features used in splits (max_features), the number of trees (n_estimators), the depth that the trees can reach (max_depth), and the function that determines the split quality (criterion). The tuned values of the hyperparameters are given in Table 2.4 below.

Table 2.4 : Hyperparameter Tuning for Random Forests

max_features	1, 2, 3, 4, 5, 6, 7, 8, 9
n_estimators	10, 50, 100, 500, 1000
max_depth	3, 4, 5, 6, 7, 8, 9, None
criterion	"gini", "entropy"

2.2.3.4. Naive Bayes

Another classification method used in the study is Naive Bayes (NB), which is based on Bayes theorem. The reason why it is called Naive is the class conditional independence assumption. In this assumption, the effects of features on a given class are assumed to be independent of each other. Thanks to acceptance, calculations for prediction are made simpler [32]. However, Naive Bayes has been shown to give good results even if the class conditional independence assumption is violated [55]. Naive Bayes classifier makes its predictions through the following equation:

$$R = \frac{P(i|X)}{P(j|X)} = \frac{P(i)\,P(X|i)}{P(j)\,P(X|j)} = \frac{P(i)\prod P(X_r|i)}{P(j)\prod P(X_r|j)} \tag{2.4}$$

Here, if R is greater than 1, class i is predicted. Otherwise, class j is considered prediction [56]. If the predictive features consist of continuous values, these features can be assumed to have a normal distribution. In this case, $P(X_r|i)$ and $P(X_r|j)$ can be calculated with the probability density function of the normal distribution, which takes

the mean and standard deviation of continuous features for a given class as parameters [32].

One of the problems that may be encountered when classifying with Naive Bayes is that one of the probabilities $P(X_r|i)$ or $P(X_r|j)$ is zero. This cancels out the influence of other possibilities on a particular class. The solution to the emerging problem can be provided by Laplace correction. Laplace correction refers to adding 1 to all numerators. What should not be forgotten here is to add the number of added 1 to the denominator [32]. Another problem is that very small numbers may emerge as a result of multiplying the probabilities. The solution to this problem is to convert the product to the sum with logarithms. In addition, the resulting transformation also provides speed in the calculation [56].

In the present study, the Naive Bayes classifier was implemented through the GaussianNB class from the scikit-learn library [44].

2.2.3.5. XGBoost

XGBoost (XGB) is an abbreviation for extreme gradient boosting and was defined by Chen and Guestrin in their 2016 paper [51]. In the XGBoost machine learning method, an initial guess is started with and the residuals are calculated with the help of this guess. Similarity scores are calculated using the obtained residuals, and gains are calculated through the similarity scores. A tree is created by deciding the best splits with the gains. After the tree is obtained, various pruning processes are applied with regularization parameters. Output values are calculated based on the final state of the tree. After this process, new predictions are made using the initial guess, learning rate, and output values and new residuals are calculated from the obtained predictions. Another tree is constructed in the same way and this process is repeated until the residuals become very small or until the predefined number of trees is reached.

As a highly effective machine learning method, XGBoost stands out thanks to its important features. These features give XGBoost scalability (it can scale seamlessly to billions of instances with limited resources), robustness to sparsity (it can optimize learning for high-sparsity datasets), parallel and distributed computing (it can leverage parallel and distributed computing to accelerate the learning process), and out-of-core computing (it can switch to out-of-core computing when faced with memory limitations). These capabilities make it a very versatile method. Due to its versatility and success in

machine learning competitions and real-world applications, XGBoost has become a popular choice among data scientists [51].

In the present study, the implementation of the XGBoost algorithm was performed through the XGBoost python package. Hyperparameter tuning was made for the algorithm used. The tuned hyperparameters consist of the depth that trees can reach (max_depth), the number of trees (n_estimators), the learning rate (learning_rate), the minimum loss reduction used in pruning trees (gamma), and the L2 regularization term applied to the weights (reg_lambda). The tuned values of the hyperparameters are given in Table 2.5 below.

Table 2.5 : Hyperparameter Tuning for XGBoost

max_depth	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
n_estimators	10, 50, 100, 500, 1000
learning_rate	0.01, 0.05, 0.1, 0.3, 0.5, 1
gamma	0, 0.25, 1
reg_lambda	0, 1, 10, 20, 100

2.2.3.6. Voting Classifier

In studies or applications of machine learning, many algorithms are often trained and these algorithms are evaluated on a dataset that is not included in the training. The algorithm that achieves the highest success as a result of the evaluation is selected as the final algorithm. The reason for this process is that none of the machine learning algorithms can consistently produce the best performance in any domain. However, selecting the best algorithm may not be enough. Algorithms make some assumptions and when the assumptions are not met by the data, erroneous results occur. A solution obtained from limited data can cause the algorithm to fail when conditions change. All these illustrate the challenges of learning from data. Even if the hyperparameters of an algorithm are tuned to improve success, success may not occur on some data points. However, other algorithms may give better results on the same data points. Therefore, combining multiple algorithms can contribute to success [27].

The aggregation of algorithms is considered an ensemble, the type of learning obtained through this ensemble is considered ensemble learning, and the methods that enable ensemble learning to be performed are considered ensemble methods. A simple example of these methods is the voting classifier. It consists of two types: hard and soft. Hard voting examines the results of each predictor and accepts the most predicted (most

votes) class label as the final prediction (Figure 2.5). In soft voting, we need methods that give the probability value of each class label. The probability estimates on each of the class labels are averaged and the label with the highest average value is considered as the final prediction. Often the voting classifier is more successful than the best algorithm in the ensemble. However, for this ensemble method to give the best performance, its algorithms need to be independent from each other. One of the methods of ensuring this independence is to include very different algorithms in the ensemble. This is because different algorithms contribute to improving success by making different types of errors [52].

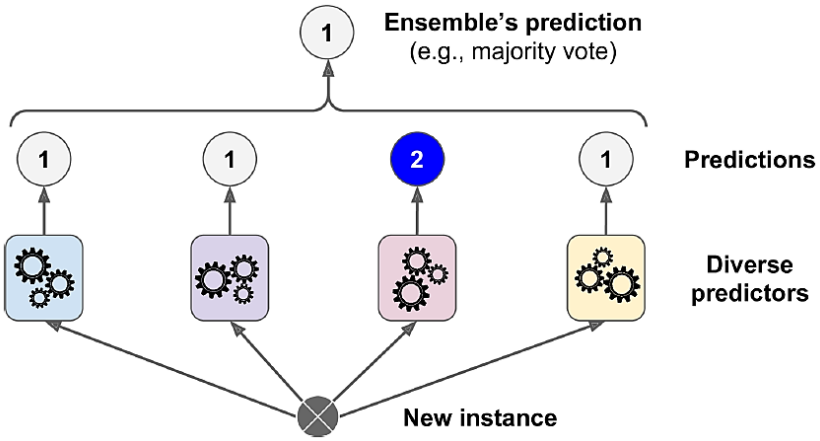

Figure 2.5 : Hard Voting Classifier [52].

In the present study, two voting classifiers, one for hard voting (HVC) and one for soft voting (SVC), were created to predict consumers' yes or no decisions. For both types, Random Forest, Naive Bayes, and XGBoost algorithms were used, which are more successful than the others. For Random Forest, the probability values of the class labels were calculated using the predict_proba method of sci-kit learn. This method, when calculating the class probability for a tree, considers the proportion of data points belonging to the same class in a leaf and then averages all available trees to obtain the probability of the input [44].

2.3. Graph Neural Networks Approach

As mentioned earlier, the main objective of this study is to compare the success of traditional machine learning algorithms and graph neural networks in fNIRS-based neuromarketing research. In line with this objective, data collected from consumers via fNIRS in a neuromarketing experiment was used. The machine learning approach applied to the collected data was described in the previous section. This section will detail the

graph neural network approach applied to the available data to predict consumers' decisions. The framework used to implement this approach consists of graph construction and graph classification. Both steps will be examined respectively.

2.3.1. Graph Construction

The first thing that needs to be done in order to apply graph neural networks to the data obtained is to reveal the graph structure. However, before proceeding to the graph construction phase, the data cleaning procedures in the machine learning section were also applied to this approach in order to obtain graphs from a cleaner dataset. Rows corresponding to "no opinion" and rows with more than half missing values were removed from the dataset. After these operations, the remaining data were used to construct the graphs.

The graph structure requires the identification of nodes, edges, and node feature vectors. In order to obtain these descriptions, the fNIRS measurements from the neuromarketing experiment were divided into 8-second time windows. The 8-second time windows correspond to the participants' trials in the experiment (stimulus seeing and decision-making). As described in the data source section, four different types of hemoglobin-related measurements were obtained through fNIRS. In the 8-second time windows, only oxyhemoglobin (HbO) was used. This is because HbO is the most reliable measurement [57]. Since the fNIRS system used in the experiment can measure twice per second, there are 16 HbO measurements in each 8-second time window. Since the sensor pad contains 16 optodes (channels), 16 x 16 matrices with optodes as columns were created from the time windows. In case of missing values in the resulting matrices, imputation was performed using the KNNImputer class of sci-kit learn [44].

The aim was to create a graph structure for each of the matrices. For this purpose, the optodes in the matrices were considered as nodes of the graphs since they correspond to specific brain regions. Functional connectivities in the brain were used to create the edges of the graphs. Functional connectivity is defined as a measure of the temporal correlation of activation in two different areas of the brain [58]. The Pearson correlation coefficient is the most commonly used method for obtaining functional connectivity [14]. Therefore, Pearson correlation coefficients between optodes in the matrices were calculated. A threshold value was used to reduce noise by eliminating low-level correlations. A value of 0.9 was chosen as higher thresholds are more effective in reducing

noise [57]. Correlations below this threshold were discarded. The remaining correlations were considered as edges of the graphs.

The time series of the optodes in the matrices were used to obtain the node feature vectors in the graphs. For each optode, the mean, skewness, kurtosis, slope, and maximum values were calculated from their 16 rows of HbO measurements. With the calculation of these frequently used temporal features [57], five-dimensional node feature vectors were created. Thus, undirected and homogeneous graphs were obtained.

All the operations to obtain the graphs were performed in Python. Since a graph was created for each matrix (trial), the prediction problem at hand became a graph classification task. Since each graph has a class label (yes/no), supervised learning was used. The details of the graph classification process will be given in the next section.

2.3.2. Graph Classification

To perform graph classification, the graphs detailed in the previous section were given as input to the graph neural network architectures. Training a graph neural network architecture for the graph classification task consists of three steps. These steps include running a graph neural network architecture several times to obtain node embeddings (mapping of node feature vectors to a lower dimensional space), merging the obtained node embeddings to form the graph embedding, and finally training a classifier on the resulting graph embedding [59].

Three different graph neural network architectures were used to provide node embeddings. The first of these architectures is the **Graph Convolutional Network (GCN)** [60]. GCNs have no problem scaling to the number of edges in the graph and can efficiently learn representations that combine information from graph topology and node properties. They accomplish this through the following equation.

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{2.5}$$

In this equation, $H^{(l)}$ is the matrix consisting of input node features, and $W^{(l)}$ is the matrix containing the learned parameters. $\tilde{A}$ is the new graph adjacency matrix obtained by adding self-loops to the graph adjacency matrix ($\tilde{A} = A + I$; A stands for adjacency matrix and I identity matrix). $\tilde{D}$ is the diagonal node degree matrix generated from the new adjacency matrix. Normalization is performed using this matrix

($\widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}$). After the multiplication operations are performed, the result obtained is passed through an activation function (e.g., ReLU). Thus, the hidden features (node embeddings) in a given layer are obtained and the matrix containing these features is denoted by $H^{(l+1)}$ in the present equation.

The other graph neural network architecture used in the study is the **Graph Attention Network (GAT)** [61]. GATs enable the use of the attention mechanism in graph neural networks. Thus, different levels of importance (attention weights) can be set for nodes in a neighborhood with respect to the target node. The importance level of node j for node i is obtained by the following equation.

$$\alpha_{ij} = \frac{e^{LeakyReLU(\vec{a}^T[W\vec{h}_i||W\vec{h}_j])}}{\sum_{k\in N_i} e^{LeakyReLU(\vec{a}^T[W\vec{h}_i||W\vec{h}_k])}} \tag{2.6}$$

Here, a linear transformation is applied to the node features ($W\vec{h}_i$, $W\vec{h}_j$). Afterwards, the resulting values are given as input to the attention mechanism ($\vec{a}^T$), which is a single-layer neural network. The result obtained is passed through the LeakyReLU activation function and finally normalized through the softmax function. Once the importance level (α_{ij}) is determined, the node embedding ($\vec{h}_i'$) in a given layer is obtained by inserting the importance level into the following equation.

$$\vec{h}_i' = \sigma\left(\sum_{j\in N_i} \alpha_{ij} W\vec{h}_j\right) \tag{2.7}$$

Here, neighbor node features (including the self-loop) are multiplied by the parameter matrix (W) and the attention weight (α_{ij}). The obtained results are summed and finally passed through an activation function (σ).

In addition to the listed processes, it has also been suggested to use multiple attention mechanisms in order to make the process more sTable. This method includes k attention mechanisms that are independent of each other. The k attention mechanisms are concatenated in the intermediate layers and averaged in the final layer. Since they are independent of each other, the attention mechanisms can be parallelized. Thus, computational efficiency is achieved. Images of multiple attention and the attention mechanism are shown in Figure 2.6. Also, the equation where multiple attention mechanisms are included in the computation is given below.

$$\vec{h}_i' = \prod_{k=1}^{K} \sigma\left(\sum_{j \in N_i} \alpha_{ij}^k W^k \vec{h}_j\right) \tag{2.8}$$

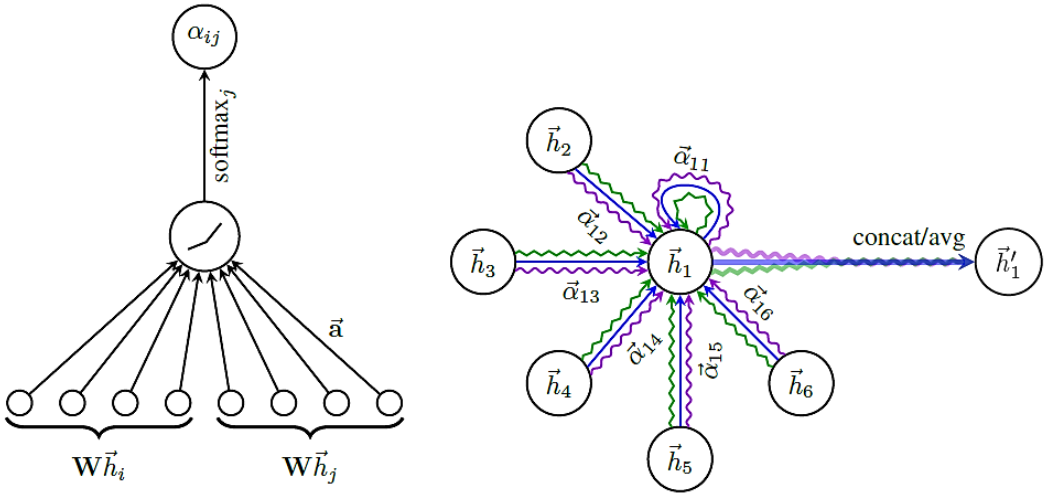

Figure 2.6 : The attention mechanism in GAT (on the left) and the multihead attention where k = 3 (on the right) [61].

The last graph neural network architecture used in this study is **Graph Isomorphism Networks (GIN)** [62]. This architecture has been inspired by the Weisfeiler-Lehman test, which is used to detect whether two graphs are non-isomorphic, and has been proven to be as good a discriminator as the Weisfeiler-Lehman test. In the GIN architecture, it is aimed to obtain different node embeddings when non-isomorphic graphs are encountered. For this purpose, it has been proposed to use two injective functions. A multilayer neural network is exploited to learn the functions. The following equation shows how the GIN architecture obtains the node embedding for a given target node in a given layer.

$$h_i = MLP\left((1+\varepsilon)x_i + \sum_{j \in N_i} x_j\right) \tag{2.9}$$

Here, MLP represents a multilayer neural network used to approximate injective functions. Another detail not found in other graph neural networks is shown by ε in the equation. It expresses the importance of the target node relative to its neighbors and can be chosen as a scalar or a learnable parameter.

After obtaining the node embeddings by running the graph neural network architectures several times, it was aimed to create the graph embedding. For this purpose,

average pooling, which is the most commonly used method for GCN and GAT architectures, was exploited. In this method, the graph embedding is created by taking the average of node embeddings in the last layer [59]. In GIN, as proposed in the original paper [62], the sum of the node embeddings obtained in each layer (summation pooling) was taken and by concatenating these sums in the last layer, the graph embedding was formed. For each architecture, a neural network that acts as the final classifier was trained on the obtained graph embedding. Thus, the steps required for the graph classification task were completed. These three steps were performed through the PyTorch Geometric library. Finally, a soft voting classifier was created by combining GCN, GAT, and GIN architectures.

Details of the graph neural network architectures used in the study are given in Table 2.6 below. Cells with more than one value indicate that the relevant hyperparameter has been tuned.

Table 2.6 : Details of GNNs

	GCN	**GAT**	**GIN**
Layers	2, 3, 4	2, 3, 4	5
Hidden Units	16, 32, 64	8, 16, 32	8, 16, 32
Activation	ReLU	ELU	ReLU
Attention Heads	-	3	-
Epoch	40, 90, 140, 190	40, 90, 140, 190	40, 90, 140, 190
Batch Size	128	128	128
Optimizer	Adam	Adam	Adam
Learning Rate	0.001, 0.01, 0.1	0.001, 0.01, 0.1	0.001, 0.01, 0.1
Dropout Rate	0.5	0.6	0.5
Global Pooling	Average	Average	Summation

2.4. Evaluation and Comparison

Before the training of the algorithms in the approaches used to solve the research problem, the data source introduced in the methodology section was divided into two sets as training and test. 85% of the data (1855 instances and graphs) constituted the training set and 15% (328 instances and graphs) constituted the test set. Stratified partitioning was

used to preserve the distribution of class labels and the data was shuffled before partitioning. Iterative imputation, sequential feature selection (both forward and backward), and PCA operations in the machine learning approach were applied only to the training dataset to prevent data leakage and the results were transferred to the test set.

Hyperparameter optimization was performed using training datasets for both machine learning algorithms and graph neural network architectures. In order to ensure a fair comparison between the approaches, each algorithm was optimized as equally as possible. Grid search was used as the optimization method. Grid search systematically searches the hyperparameter space specific to the algorithm. It gives equal probability to all combinations in the defined hyperparameter space and tries all these combinations one by one [63]. Some of the hyperparameters used to create the combinations can take an unlimited number of values. Therefore, there may be a need to create bounds for the use of grid search optimization. High-dimensional hyperparameter spaces create a large number of combinations, which significantly increases the computational cost and the time required for execution. However, the search operations can be accelerated by parallelizing the hyperparameter values [64]. An example of a two-dimensional grid search space is given in Figure 2.7 below.

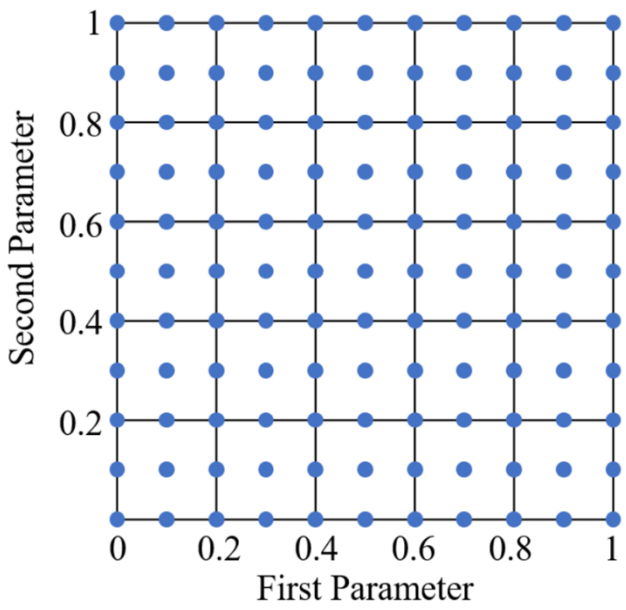

Figure 2.7 : An example of a two-dimensional grid search space [64].

Each hyperparameter combination generated in the grid search optimization was evaluated using k-fold cross-validation. This method ensures that every data point at hand is tested. It works iteratively and creates k subsets from the existing data set in each iteration. It determines one of the subsets it obtains as the validation set and combines the

remaining subsets to form the training set. It trains the algorithm on the training set and evaluates the success of the trained algorithm on the validation set. Since it repeats these processes k times, k trained models and success scores emerge. By taking the arithmetic average of these success scores, the cross-validated successes of the models containing the values in the hyperparameter combinations are obtained [65]. For a better understanding of the method, a visualization of the 5-fold cross-validation used for hyperparameter selection is given in Figure 2.8 below.

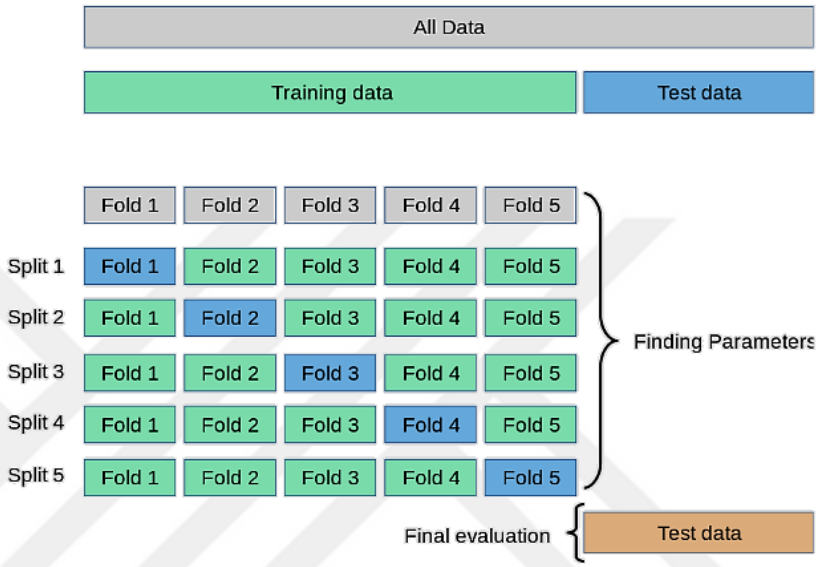

Figure 2.8 : A visualization of the 5-fold cross-validation used for hyperparameter selection [44].

In the current study, the k value was selected as 10 for cross-validation and since the class label ratios are close to each other, the accuracy metric was used for success evaluation. Comparisons were made based on average accuracy values on the same data splits. The hyperparameter combination with the highest average accuracy value was selected as the final model configuration. Accuracy values were calculated using the following equation.

$$Accuracy = \frac{Correct\ Predictions}{All\ Predictions} \tag{2.10}$$

After the completion of the hyperparameter optimization phase, the generalization success of the models with the most successful configuration was evaluated. For generalization success, the accuracy values of these models were calculated on the same test dataset. As a measure of uncertainty, binomial confidence intervals calculated using the sampling distribution of the proportion were added to the obtained accuracy values. The following equation was used to calculate binomial confidence intervals.

$$p \pm Z_{1-\frac{\alpha}{2}} \sqrt{\frac{p(1-p)}{n}} \tag{2.11}$$

In this equation, p represents the interest rate and α represents the previously determined alpha value. $Z_{1-\alpha/2}$ is the z value corresponding to the previously determined confidence level and n stands for the sample size. Comparisons were made using the test accuracy values and the corresponding binomial confidence intervals. Comparison results will be given in the next section.

3. RESULTS AND DISCUSSION

In this section, the findings obtained by applying the procedures described in the methodology section to the available data will be presented and discussed. The results of the study will be analyzed in three groups: machine learning approach, graph neural network approach, and evaluation and comparison.

3.1. Machine Learning Approach

As mentioned in the data preparation section, the fNIRS measurements from the neuromarketing experiment were averaged for each trial (task period) of the participants. Since each participant performed 30 trials, a total of 2610 (87 x 30) rows were obtained. Since there are 16 optode (channels) on the sensor pad of the fNIRS device used in the experiment and 4 measurements related to hemoglobin (HbO, HbR, Hbt, Oxy) were obtained, a total of 64 (4 x 16) features were obtained. Rows corresponding to “no opinion” responses were removed from this dataset. Afterwards, rows with more than half missing values (49) were also removed from the dataset, as they did not contain enough information. Thus, a data set of 2183 x 64 was obtained. The descriptive statistics of the features in this dataset are given in Table 3.1 below.

Table 3.1 : Descriptive Statistics of Predictive Variables

Feature	Count	Mean	Std. Deviation	Min.	50%	Max.	Skewness	Kutrosis
HBO_Optode1	1666	-0.01	0.28	-1.43	-0.01	1.5	0.43	3.8
HBO_Optode2	2013	0.0	0.3	-2.66	-0.0	2.06	0.17	7.13
HBO_Optode3	2139	-0.02	0.29	-2.24	-0.02	1.67	-0.17	5.71
HBO_Optode4	2062	-0.01	0.31	-3.35	-0.01	2.64	-0.36	13.64
HBO_Optode5	2068	-0.0	0.27	-2.42	-0.0	2.37	-0.04	7.95
HBO_Optode6	2137	0.01	0.3	-2.07	0.0	2.27	0.12	5.07
HBO_Optode7	2121	0.01	0.29	-2.28	0.02	1.36	-0.44	5.98
HBO_Optode8	2058	0.03	0.3	-2.06	0.02	1.54	-0.25	3.74
HBO_Optode9	2118	0.02	0.28	-1.43	0.02	2.26	0.02	4.72
HBO_Optode10	2030	0.03	0.32	-1.79	0.02	2.82	0.48	6.14
HBO_Optode11	2051	0.0	0.3	-2.26	0.0	2.17	-0.14	5.08
HBO_Optode12	2097	0.01	0.3	-1.47	0.01	2.86	0.63	6.41
HBO_Optode13	2038	-0.02	0.33	-2.39	-0.01	1.48	-0.58	5.06
HBO_Optode14	2088	-0.01	0.3	-1.47	-0.01	2.63	0.33	4.6
HBO_Optode15	1622	0.0	0.29	-1.38	-0.0	1.7	0.37	3.35
HBO_Optode16	2058	0.0	0.32	-1.47	-0.01	2.61	1.04	8.68
HBR_Optode1	1666	0.01	0.2	-2.18	-0.0	3.5	3.08	67.08
HBR_Optode2	2013	-0.0	0.21	-1.35	-0.01	2.56	1.48	15.42
HBR_Optode3	2139	0.01	0.17	-0.73	0.01	2.32	1.4	17.85

HBR_Optode4	2062	-0.02	0.19	-1.85	-0.02	2.59	0.84	37.32
HBR_Optode5	2068	0.02	0.2	-1.51	0.01	3.5	2.85	55.23
HBR_Optode6	2137	-0.01	0.21	-2.05	-0.01	3.63	2.91	62.15
HBR_Optode7	2121	0.02	0.22	-1.77	0.01	3.38	1.72	31.31
HBR_Optode8	2058	-0.02	0.21	-1.4	-0.02	3.48	1.91	38.99
HBR_Optode9	2118	0.02	0.21	-1.84	0.01	2.27	0.77	14.99
HBR_Optode10	2030	-0.02	0.22	-1.3	-0.02	3.11	2.22	29.27
HBR_Optode11	2051	0.01	0.22	-1.21	0.01	2.31	0.55	8.54
HBR_Optode12	2097	-0.01	0.18	-1.18	-0.01	2.45	1.12	20.67
HBR_Optode13	2038	0.02	0.27	-1.28	0.02	4.21	1.86	32.63
HBR_Optode14	2088	-0.0	0.22	-1.17	-0.01	4.34	3.61	75.12
HBR_Optode15	1622	0.01	0.28	-3.24	0.01	3.0	-0.44	27.11
HBR_Optode16	2058	0.0	0.29	-1.26	-0.01	4.19	3.41	38.12
HBT_Optode1	1666	0.01	0.33	-2.45	-0.01	2.52	0.33	7.93
HBT_Optode2	2013	-0.0	0.4	-4.02	-0.01	3.88	0.82	15.71
HBT_Optode3	2139	-0.01	0.32	-2.79	-0.0	3.99	0.54	19.18
HBT_Optode4	2062	-0.02	0.4	-4.43	-0.03	4.47	-0.43	32.93
HBT_Optode5	2068	0.02	0.35	-3.22	0.01	4.51	1.22	30.12
HBT_Optode6	2137	-0.01	0.39	-2.8	-0.01	4.77	1.21	24.64
HBT_Optode7	2121	0.04	0.38	-3.91	0.03	4.46	-0.02	21.77
HBT_Optode8	2058	0.01	0.39	-3.46	0.01	4.48	0.02	16.14
HBT_Optode9	2118	0.03	0.36	-2.31	0.03	4.53	0.65	16.41
HBT_Optode10	2030	0.01	0.44	-2.37	0.01	5.94	1.93	25.66
HBT_Optode11	2051	0.01	0.34	-3.03	0.01	4.48	0.55	21.78
HBT_Optode12	2097	-0.0	0.37	-1.8	-0.01	4.54	1.24	15.47
HBT_Optode13	2038	-0.0	0.35	-3.12	0.0	4.95	0.48	30.7
HBT_Optode14	2088	-0.01	0.35	-1.78	-0.02	4.92	1.62	23.73
HBT_Optode15	1622	0.02	0.39	-4.46	0.0	3.55	-0.01	25.65
HBT_Optode16	2058	0.0	0.44	-2.37	-0.02	5.36	3.68	35.05
OXY_Optode1	1666	-0.02	0.35	-4.48	-0.01	1.92	-1.43	20.25
OXY_Optode2	2013	0.0	0.33	-1.72	-0.0	2.16	0.21	4.01
OXY_Optode3	2139	-0.04	0.35	-1.99	-0.03	1.54	-0.12	2.41
OXY_Optode4	2062	0.01	0.33	-2.27	0.0	1.92	0.05	4.75
OXY_Optode5	2068	-0.02	0.33	-2.49	-0.01	1.45	-0.36	3.7
OXY_Optode6	2137	0.02	0.34	-2.69	0.02	1.42	-0.29	3.86
OXY_Optode7	2121	-0.01	0.35	-2.3	0.0	1.57	-0.37	2.81
OXY_Optode8	2058	0.04	0.34	-2.48	0.04	1.35	-0.25	3.02
OXY_Optode9	2118	0.0	0.33	-1.73	0.01	2.21	-0.13	3.01
OXY_Optode10	2030	0.05	0.33	-1.98	0.05	1.68	-0.1	2.94
OXY_Optode11	2051	-0.01	0.39	-2.05	-0.01	1.75	-0.24	2.11
OXY_Optode12	2097	0.02	0.34	-1.74	0.03	2.89	0.16	4.36
OXY_Optode13	2038	-0.04	0.49	-3.47	-0.04	2.03	-0.24	3.63
OXY_Optode14	2088	-0.01	0.4	-3.76	-0.0	2.68	-0.39	6.46
OXY_Optode15	1622	-0.01	0.41	-2.46	-0.01	2.01	-0.13	3.31
OXY_Optode16	2058	-0.0	0.41	-3.55	-0.0	2.49	-0.16	6.19

Spearman correlation analysis was used to examine the relationships between the features. This method was preferred because most of the attributes have high kurtosis

values and this method does not require normal distribution. When the values obtained are analyzed, it is seen that some features are highly correlated with each other. This may indicate a multicollinearity problem. Figure 3.1 below shows the heat map of the correlation matrix.

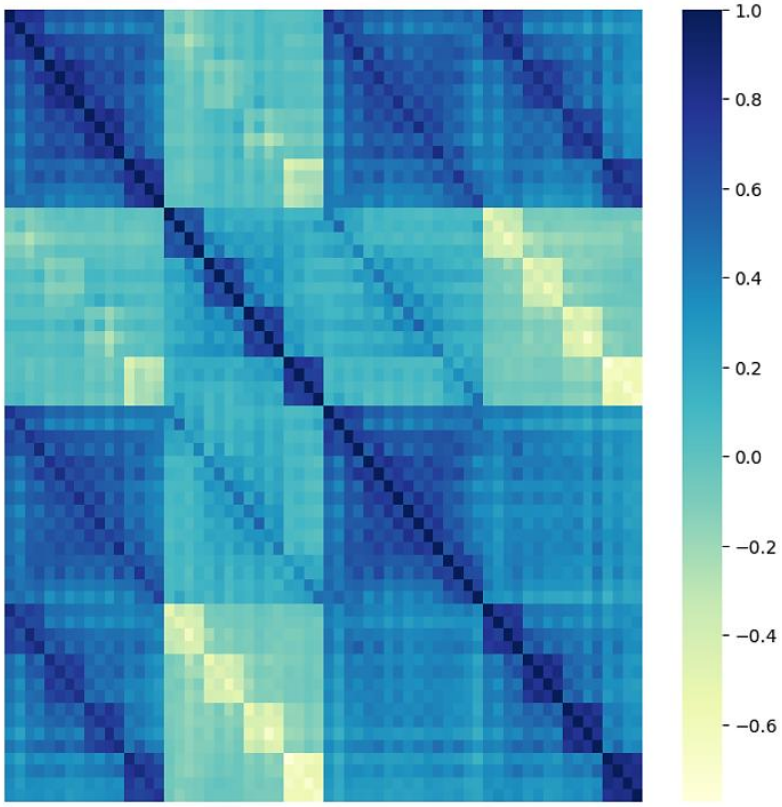

Figure 3.1 : Correlation Heatmap of Predictive Variables.

Yes and no decisions constituting the target variable were coded as 0 and 1, respectively. When the distribution of class labels is examined, it is seen that the rates are close to each other. The descriptive statistics of the target variable are given in Table 3.2 below.

Table 3.2 : Descriptive Statistics of Target Variable

Class Label	Count	Percent
Yes	1166	53.41
No	1017	46.59

Most of the machine learning algorithms used in the study are sensitive to outliers. Therefore, outlier analysis was performed on the obtained predictor variables. At the same time, it is also aimed to reduce the noise in the data set. IQR method was used for outlier detection. The interquartile range was multiplied by 1.5. When all features were analyzed one by one, it was observed that there were many outliers. Therefore, instead of removing

rows with outliers from the dataset, such data values were marked as NA. Table 3.3 below shows the number of outliers found in the features along with their percentages.

Table 3.3 : Outliers in Predictive Variables

Feature	Outliers	Percent
HBO_Optode1	83	3.8
HBO_Optode2	134	6.14
HBO_Optode3	117	5.36
HBO_Optode4	116	5.31
HBO_Optode5	124	5.68
HBO_Optode6	118	5.41
HBO_Optode7	104	4.76
HBO_Optode8	96	4.4
HBO_Optode9	120	5.5
HBO_Optode10	95	4.35
HBO_Optode11	119	5.45
HBO_Optode12	112	5.13
HBO_Optode13	98	4.49
HBO_Optode14	107	4.9
HBO_Optode15	67	3.07
HBO_Optode16	107	4.9
HBR_Optode1	92	4.21
HBR_Optode2	126	5.77
HBR_Optode3	100	4.58
HBR_Optode4	137	6.28
HBR_Optode5	123	5.63
HBR_Optode6	152	6.96
HBR_Optode7	148	6.78
HBR_Optode8	124	5.68
HBR_Optode9	138	6.32
HBR_Optode10	104	4.76
HBR_Optode11	120	5.5
HBR_Optode12	109	4.99
HBR_Optode13	111	5.08
HBR_Optode14	88	4.03
HBR_Optode15	93	4.26
HBR_Optode16	119	5.45
HBT_Optode1	124	5.68
HBT_Optode2	164	7.51
HBT_Optode3	173	7.92
HBT_Optode4	142	6.5
HBT_Optode5	185	8.47
HBT_Optode6	137	6.28
HBT_Optode7	157	7.19
HBT_Optode8	131	6.0
HBT_Optode9	163	7.47
HBT_Optode10	152	6.96

HBT_Optode11	166	7.6
HBT_Optode12	140	6.41
HBT_Optode13	151	6.92
HBT_Optode14	129	5.91
HBT_Optode15	99	4.54
HBT_Optode16	154	7.05
OXY_Optode1	71	3.25
OXY_Optode2	90	4.12
OXY_Optode3	80	3.66
OXY_Optode4	105	4.81
OXY_Optode5	96	4.4
OXY_Optode6	87	3.99
OXY_Optode7	90	4.12
OXY_Optode8	89	4.08
OXY_Optode9	119	5.45
OXY_Optode10	81	3.71
OXY_Optode11	101	4.63
OXY_Optode12	102	4.67
OXY_Optode13	82	3.76
OXY_Optode14	83	3.8
OXY_Optode15	64	2.93
OXY_Optode16	103	4.72

The implementation of most machine learning algorithms used in the study does not accept missing values. Therefore, the dataset was examined for missing values. As a result of the examination, the predictor variables were found to contain missing values. Table 3.4 below shows the number of missing values found in the features along with their percentages.

Table 3.4 : Missing Values in Predictive Variables

Feature	Missing Values	Percent
HBO_Optode1	517	23.68
HBO_Optode2	170	7.79
HBO_Optode3	44	2.02
HBO_Optode4	121	5.54
HBO_Optode5	115	5.27
HBO_Optode6	46	2.11
HBO_Optode7	62	2.84
HBO_Optode8	125	5.73
HBO_Optode9	65	2.98
HBO_Optode10	153	7.01
HBO_Optode11	132	6.05
HBO_Optode12	86	3.94
HBO_Optode13	145	6.64
HBO_Optode14	95	4.35
HBO_Optode15	561	25.7

HBO_Optode16	125	5.73
HBR_Optode1	517	23.68
HBR_Optode2	170	7.79
HBR_Optode3	44	2.02
HBR_Optode4	121	5.54
HBR_Optode5	115	5.27
HBR_Optode6	46	2.11
HBR_Optode7	62	2.84
HBR_Optode8	125	5.73
HBR_Optode9	65	2.98
HBR_Optode10	153	7.01
HBR_Optode11	132	6.05
HBR_Optode12	86	3.94
HBR_Optode13	145	6.64
HBR_Optode14	95	4.35
HBR_Optode15	561	25.7
HBR_Optode16	125	5.73
HBT_Optode1	517	23.68
HBT_Optode2	170	7.79
HBT_Optode3	44	2.02
HBT_Optode4	121	5.54
HBT_Optode5	115	5.27
HBT_Optode6	46	2.11
HBT_Optode7	62	2.84
HBT_Optode8	125	5.73
HBT_Optode9	65	2.98
HBT_Optode10	153	7.01
HBT_Optode11	132	6.05
HBT_Optode12	86	3.94
HBT_Optode13	145	6.64
HBT_Optode14	95	4.35
HBT_Optode15	561	25.7
HBT_Optode16	125	5.73
OXY_Optode1	517	23.68
OXY_Optode2	170	7.79
OXY_Optode3	44	2.02
OXY_Optode4	121	5.54
OXY_Optode5	115	5.27
OXY_Optode6	46	2.11
OXY_Optode7	62	2.84
OXY_Optode8	125	5.73
OXY_Optode9	65	2.98
OXY_Optode10	153	7.01
OXY_Optode11	132	6.05
OXY_Optode12	86	3.94
OXY_Optode13	145	6.64
OXY_Optode14	95	4.35
OXY_Optode15	561	25.7

OXY_Optode16	125	5.73

Instead of deleting rows with missing values, these values were imputed together with outliers marked as NA. As an imputation method, the iterative imputation method using the Random Forest algorithm as a regressor was preferred. Before the imputation process, the dataset was divided into two sets: training and test. 85% of the data (1855 samples) constituted the training set and 15% (328 samples) constituted the test set. Stratified partitioning was used and the dataset was shuffled before partitioning. To avoid data leakage, the imputation and subsequent preprocessing steps were performed on the training dataset. The obtained results were transferred to the test dataset. Table 3.5 below shows the distribution of class labels of the target variable in the training and test sets.

Table 3.5 : Distribution of Target Variable in Train and Test Sets

	Train Set	**Test Set**
Yes	991	175
No	864	153

As mentioned before, there are 64 predictor features in the dataset. Feature selection was applied to reduce the number of features. At the same time, it was aimed to eliminate the multicollinearity problem. For feature selection, two different types of sequential feature selection were used, one floating forward and one floating backward. As a result of the comparison of the feature sets according to the accuracy values obtained, the first method suggested 9 features and the second method suggested 5 features. Afterward, a feature set containing 10 features was obtained by the union of the two sets. This set consists of HBO_Optode7, HBO_Optode14, HBT_Optode7, HBT_Optode9, HBT_Optode12, OXY_Optode1, OXY_Optode3, OXY_Optode7, OXY_Optode14, OXY_Optode15 variables.

After the feature selection process, the VIF values of the variables obtained were calculated. As a result of the examinations, it was seen that some of the variables had VIF values above 5. This showed that the multicollinearity problem persisted. The VIF values of the variables are given in the Table 3.6 below.

Table 3.6 : VIF Values of Features

Feature	VIF
HBO_Optode7	22.53
HBO_Optode14	6.83
HBT_Optode7	8.73
HBT_Optode9	3.59
HBT_Optode12	2.92
OXY_Optode1	2.97
OXY_Optode3	3.25
OXY_Optode7	10.17
OXY_Optode14	7.1
OXY_Optode15	3.08

In order to eliminate the multicollinearity problem, PCA was applied to the features. After comparing the sets formed by the sequential addition of the components according to the obtained accuracy values, the first 9 components were selected for training the machine learning algorithms. The cumulative variance ratios explained by the components are shown in the Figure 3.2 below.

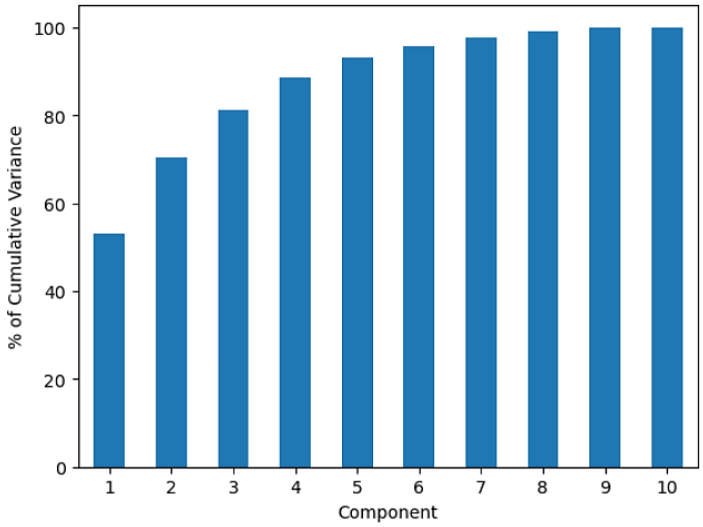

Figure 3.2 : Cumulative Variance of Components.

Various machine learning algorithms were trained on the 9 components obtained from fNIRS measurements to predict participants' compatibility (yes/no) decisions for brand adjective pairs. These are K-Nearest Neighbors, Support Vector Machines, Random Forest, Naive Bayes, and XGBoost algorithms. For algorithms other than Naive Bayes, hyperparameter optimization was performed on the training dataset. Grid search was used as the optimization method. Each hyperparameter combination was evaluated with 10-fold cross-validation and the best combination was selected according to the average accuracy value. The generalization success of the algorithms was evaluated based on the

accuracy scores obtained on the test set allocated at the beginning of the study. Naive Bayes, Random Forest, and XGBoost algorithms, which performed better than the others, were combined to create two different voting classifiers, soft and hard. The selected hyperparameter combinations and test accuracy values of the algorithms are given in Table 3.7 below. The highest test accuracy value is highlighted in the Table.

Table 3.7 : Successes of Machine Learning Algorithms

Algorithm	**Best Hyperparameter Combination**	**Test Accuracy**
RF	criterion: entropy, max_depth: 3, max_features: 9, n_estimators: 10	0.57
KNN	metric: euclidean, n_neighbors: 5, weights: uniform	0.50
SVM	C: 10, gamma: scale, kernel: rbf	0.48
NB		0.56
XGB	gamma: 1, learning_rate: 0.01, max_depth: 10, n_estimators: 1000, reg_lambda: 1	0.58
HVC		**0.60**
SVC		0.59

3.2. Graph Neural Networks Approach

As mentioned in the Methods section, graphs were created from the fNIRS measurements. In these graphs, the optodes corresponding to specific brain regions represented nodes. Edges were created by calculating functional connections in the brain as Pearson correlation coefficients and using a threshold value (0.9). Time series of 8-second HBO measurements were used for node feature vectors. The mean, skewness, kurtosis, slope, and maximum values were calculated to obtain 5-dimensional node feature vectors. This resulted in undirected and homogeneous graphs for each trial. One of the generated graphs is shown in Figure 3.3 below. This graph has 16 nodes and 84 edges.

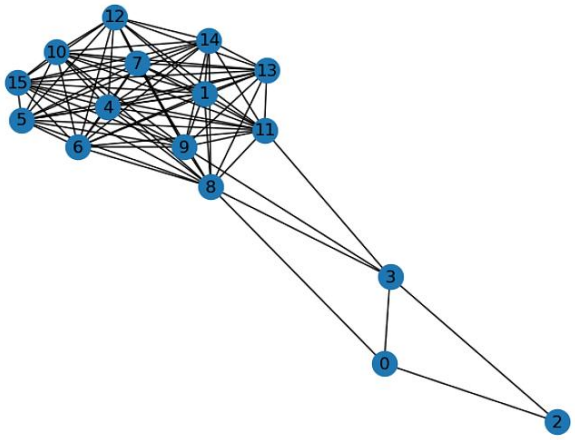

Figure 3.3 : One of the Created Graphs.

Since the resulting graphs represent participants' trials, each graph has a class label (yes/no). Therefore, the task at hand became the graph classification task. In order to classify the graphs, the training graphs were given as input to graph neural network architectures. These architectures consist of Graph Convolutional Network, Graph Attention Network, and Graph Isomorphism Network. For each architecture, hyperparameter optimization was performed on the training dataset. Grid search was used as the optimization method. Each hyperparameter combination was evaluated with 10-fold cross-validation and the best combination was selected based on the average accuracy. Then, a soft voting classifier (GNN_VC) was created by combining GNN architectures. The generalization success of all models was evaluated based on the accuracy scores obtained on the test dataset allocated at the beginning of the study. The selected hyperparameter combinations and test accuracy values of the models are given in Table 3.8 below. The highest test accuracy value is highlighted in the Table.

Table 3.8 : Successes of GNN Algorithms

Algorithm	**Best Hyperparameter Combination**	**Test Accuracy**
GCN	epochs: 190, hidden_units: 64, layers: 3, learning_rate: 0.01	0.53
GAT	epochs: 140, hidden_units: 32, layers: 2, learning_rate: 0.01	0.52
GIN	epochs: 190, hidden_units: 32, learning_rate: 0.001	**0.55**
GNN_VC		0.51

3.3. Evaluation and Comparison

As stated in the previous sections, the main objective of this study is to compare the performance of traditional machine learning algorithms and graph neural networks in fNIRS-based neuromarketing research. Therefore, data from a neuromarketing experiment utilizing fNIRS brain imaging is used. In order to make a fair comparison on the data, as much as possible equal amount of hyperparameter optimization was performed for the algorithms in both approaches. Furthermore, the generalization performance of all models was evaluated on the same test set. Binomial confidence intervals, calculated using the sampling distribution of the proportion, were added to the resulting test accuracies as a measure of uncertainty. Figure 3.4 below shows the test accuracy values of the algorithms and the corresponding confidence intervals.

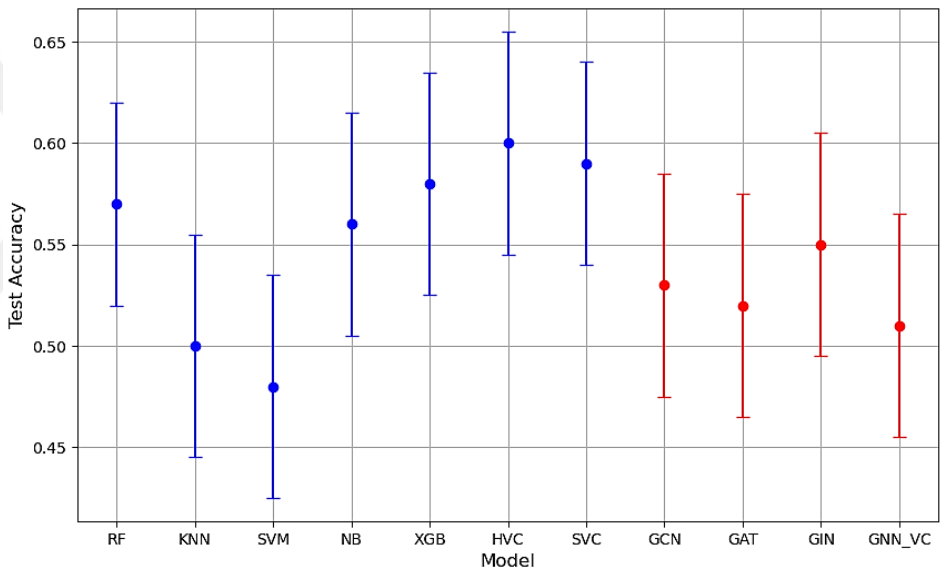

Figure 3.4 : Test Accuracy with Confidence Intervals.

The comparison results show that all machine learning algorithms except KNN and SVM have higher accuracy values than graph neural network architectures. Graph neural networks were not able to make sufficient use of structural information (functional connections). Machine learning models based on ensemble learning gave the most successful results on the available dataset. A similar result was seen in a study comparing the approaches in a different context. In this study, the success of GCN and the ensemble-based RF algorithm was compared on a public startup database, and the former showed similar or lower performances than the latter [66]. However, the results in the current study differed from the studies described in the introduction section [15], [16]. In both articles, the average of the scores provided by the k-fold cross-validation used to evaluate

the hyperparameter values was taken as the final performance. Furthermore, there is no indication that hyperparameter optimization was performed for the machine learning algorithms used as baselines for comparison. For these reasons, it may be difficult to uncover the reason for the discrepancy with the findings in the current study.

CONCLUSION

The main purpose of this study was to investigate how graph neural networks would perform in analyzing fNIRS data in the context of neuromarketing and to compare them with classical machine algorithms. Thus, it was aimed to determine which of these approaches would yield more successful results. For this comparison, fNIRS data from a neuromarketing experiment conducted to determine perceptions of brands were used. In the machine learning approach, various preprocessing steps were applied to the data. After these steps, KNN, SVM, RF, NB, and XGB algorithms were trained. Two different voting classifications (soft and hard) were created by combining the algorithms with better performance. In the graph neural network approach, a graph was created for each trial of the participants from the fNIRS data. In order to obtain these graphs, the functional connections of the brain calculated by Pearson correlation coefficient were used. The resulting graphs were given as input to GCN, GAT, and GIN architectures and tried to be classified. In addition, a voting classifier was created by combining these architectures.

To ensure fair comparisons, the hyperparameters of the models in the approaches were optimized as equally as possible. The generalization success for each model was calculated as accuracy scores on the same test set. Binomial confidence intervals were added to the obtained scores as a measure of uncertainty. The comparison results showed that machine learning algorithms generally outperform graph neural network architectures on the current dataset. Machine learning models based on ensemble learning have the best scores.

A review of the literature reveals that there is no study that uses graph neural networks and fNIRS together in the context of neuromarketing. Therefore, it is thought that the current study can provide important information on the use of graph neural networks in fNIRS-based neuromarketing research. The same comparisons can be repeated on the data of different neuromarketing studies using fNIRS to understand brand perceptions. In addition, fNIRS measurements from neuromarketing research involving different tasks such as purchase behavior can also be used for this purpose. Thus, the findings of the current study can be improved.

REFERENCES

[1] N. Lee, A. Broderick, and L. Chamberlain, “What is ‘neuromarketing’? A discussion and agenda for future research,” *International Journal of Psychophysiology*, vol. 63, no. 2, pp. 199-204, Feb. 2007, doi: 10.1016/j.ijpsycho.2006.03.007.

[2] L. Bell, J. Vogt, C. Willemse, T. Routledge, L. T. Butler, and M. Sakaki, “Beyond self-report: A review of physiological and neuroscientific methods to investigate consumer behavior,” *Frontiers in Psychology*, vol. 9, Sep. 2018, doi: 10.3389/fpsyg.2018.01655.

[3] J. Gountas, S. Gountas, J. Ciorciari, and P. Sharma, “Looking beyond traditional measures of advertising impact: Using neuroscientific methods to evaluate social marketing messages,” *Journal of Business Research*, vol. 105, pp. 121–135, Aug. 2019, doi: 10.1016/j.jbusres.2019.07.011.

[4] Y. Zhang, P. Thaichon, and W. Shao, “Neuroscientific research methods and techniques in consumer research,” *Australasian Marketing Journal (AMJ)*, vol. 31, no. 3, pp. 211–227, Mar. 2022, doi: 10.1177/14413582221085321.

[5] W. M. Lim, "Demystifying neuromarketing," *Journal of Business Research*, vol. 91, no. 4, p. 205-220, Oct. 2018, doi: 10.1016/j.jbusres.2018.05.036.

[6] P. Pinti, *et al*., “The present and future use of functional near-infrared spectroscopy (fNIRS) for cognitive neuroscience,” *Annals of the New York Academy of Sciences*, vol. 1464, no. 1, pp. 5-29, Aug. 2018, doi: 10.1111/nyas.13948.

[7] S. Tak and J. C. Ye, “Statistical analysis of fNIRS data: A comprehensive review,” *NeuroImage*, vol. 85, pp. 72–91, Jun. 2013, doi: 10.1016/j.neuroimage.2013.06.016.

[8] S. Bak, Y. Jeong, M. Yeu, and J. Jeong, “Brain–computer interface to predict impulse buying behavior using functional near-infrared spectroscopy,” *Scientific Reports*, vol. 12, no. 1, pp. 18-24, Oct. 2022, doi: 10.1038/s41598-022-22653-8.

[9] M. P. Çakir, T. Çakar, Y. Girisken, and D. Yurdakul, “An investigation of the neural correlates of purchase behavior through fNIRS,” *Eur J Mark*, vol. 52, no. 1–2, pp. 224–243, Feb. 2018, doi: 10.1108/EJM-12-2016-0864.

[10] S. M. H. Hosseini, Y. Mano, M. Rostami, M. Takahashi, M. Sugiura, and R. Kawashima, “Decoding what one likes or dislikes from single-trial fNIRS measurements,” *Neuroreport*, vol. 22, no. 6, pp. 269–273, Apr. 2011, doi: 10.1097/wnr.0b013e3283451f8f.

[11] M. Y. Köksal, “Predicting the preference of liking using fNIRS and machine learning algorithms,” M.S. thesis, Dept. Information Technologies, MEF Univ., İstanbul, Turkey, 2023.

[12] G. Filiz, T. Çakar, T. E. Soyaltın, Y. Girişken, and C. A. Türkyılmaz, "Analyzing consumer behavior: The impact of retro music in advertisements on a chocolate brand and consumer engagement," 2023 Innovations in Intelligent Systems and Applications Conference (ASYU), Sivas, Turkiye, 2023, pp. 1-6, doi: 10.1109/ASYU58738.2023.10296776.

[13] T. Çakar and G. Filiz, "Unraveling neural pathways of political engagement: Bridging neuromarketing and political science for understanding voter behavior and political leader perception," *Front. Hum. Neurosci.*, vol. 17, Dec. 2023, doi: 10.3389/fnhum.2023.1293173.

[14] S. Zhang, *et al*., "The combination of a graph neural network technique and brain imaging to diagnose neurological disorders: A review and outlook," *Brain Sciences*, vol. 13, no. 10, pp. 1462, Oct. 2023, doi: 10.3390/brainsci13101462.

[15] Q. Yu, R. Wang, J. Liu, L. Hu, M. Chen, and Z. Liu, "GNN-based depressionr recognition using spatio-temporal information: A fNIRS study," *in IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 10, pp. 4925-4935, Oct. 2022, doi: 10.1109/JBHI.2022.3195066.

[16] M. Seo, E. Jeong, and K.-S. Kim, "Multi-class fNIRS classification using an ensemble of GNN-based models," *in IEEE Access*, vol. 11, pp. 137606-137620, Dec. 2023, doi: 10.1109/ACCESS.2023.3339647.

[17] C. Morin, "Neuromarketing: The new science of consumer behavior," *Soc*, vol. 48, pp. 131-135, Jan. 2011, doi: 10.1007/s12115-010-9408-1.

[18] V. Boricean, "Brief history of neuromarketing," in *The International Conference on Economics and Administration*, *ICEA -FAA*, Nov. 2009, pp. 119–121.

[19] B. C. Iloka and K. J. Onyeke, "Neuromarketing: A historical review," *Neurosci Res Notes*, vol. 3, no. 3, pp. 27–35, Sep. 2020, doi: 10.31117/neuroscirn.v3i3.54.

[20] Z. O. Touhami, L. Benlafkih, M. Jiddane, Y. Cherrah, H. O. EL Malki, and A. Benomar, "Neuromarketing: Where marketing and neuroscience meet," *African Journal of Business Management*, vol. 5, no. 5, pp. 1528-1532, Mar. 2011, doi: 10.5897/AJBM10.729.

[21] D. Highton, D. Boas, Y. Minagawa, R. Mesquita, and J. Gervain, "Thirty years of functional near-infrared spectroscopy," *Neurophotonics*, vol. 10, no. 2, Apr. 2023, doi: 10.1117/1.nph.10.2.023501.

[22] C. Y. Y. Lai, C. S. H. Ho, C. R. Lim, and R. C. M. Ho, "Functional near-infrared spectroscopy in psychiatry," *BJPsych Advances*, vol. 23, no. 5, pp. 324–330, Sep. 2017, doi: 10.1192/apt.bp.115.015610.

[23] S. Cutini, S. Moro, and S. Bisconti, "Functional near infrared optical imaging in cognitive neuroscience: An introductory review," *J. Near Infrared Spectrosc.*, vol. 20, no. 1, pp. 75-92, Mar. 2012, doi: 10.1255/jnirs.969.

[24] F. Jöbsis, “Noninvasive, infrared monitoring of cerebral and myocardial oxygen sufficiency and circulatory parameters,” *Science*, vol. 198, no. 4323, pp. 1264-1267, Dec. 1977, doi: 10.1126/science.929199.

[25] M. Ferrari and V. Quaresima, “A brief review on the history of human functional near-infrared spectroscopy (fNIRS) development and fields of application,” *NeuroImage*, vol. 63, no. 2, pp. 921-935, Nov. 2012, doi: 10.1016/j.neuroimage.2012.03.049.

[26] S. Koike, Y. Nishimura, R. Takizawa, N. Yahata, and K. Kasai, “Near-infrared spectroscopy in schizophrenia: A possible biomarker for predicting clinical outcome and treatment response,” *Frontiers in Psychiatry*, vol. 4, no. 145, Nov. 2013, doi: 10.3389/fpsyt.2013.00145.

[27] E. Alpaydin, *Introduction to Machine Learning*, 3rd ed. Cambridge, MA, USA: MIT Press, 2014.

[28] I. Naqa and M. Murphy, “What is machine learning?,” *Machine Learning in Radiation Oncology: Theory and Applications*, pp. 3-11, 2015, doi: 10.1007/978-3-319-18305-3_1.

[29] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436-444, May. 2015, doi: 10.1038/nature14539.

[30] R. H. Jhaveri, A. Revathi, K. Ramana, R. Raut, and R. K. Dhanaraj, “A review on machine learning strategies for real-world engineering applications,” *Mobile Information Systems*, vol. 2022, no. 1833507, Aug. 2022, doi: 10.1155/2022/1833507.

[31] K. Teja and B U. J. Reddy, “Review and applications of machine learning,” *IJSREM*, vol. 7, no. 2, Feb. 2023, doi: 10.55041/IJSREM17823.

[32] J. Han, M. Kamber and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. Waltham, MA, USA: Morgan Kaufmann Publishers, 2012.

[33] L. Kaelbling, M. Littman, and A. Moore, “Reinforcement learning: A survey,” *Journal of Artificial Intelligence Research*, vol. 4, pp. 237-285, May. 1996, doi: 10.1613/jair.301.

[34] M. Rodriguez and P. Neubauer, “The graph traversal pattern,” *Graph Data Management: Techniques and Applications*, pp. 29-46, Apr. 2010, doi: 10.4018/978-1-61350-053-8.ch002.

[35] J. Zhou, *et al*., “Graph neural networks: A review of methods and applications,” *AI Open*, vol. 1, pp. 57-81, 2020, doi: 10.1016/j.aiopen.2021.01.001.

[36] H. Cai, V. W. Zheng and K. C. -C. Chang, “A comprehensive survey of graph embedding: Problems, techniques, and applications,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616-1637, Sep. 2018, doi: 10.1109/TKDE.2018.2807452.

[37] F. Liang, C. Qian, W. Yu, D. Griffith, and N. Golmie, “Survey of graph neural networks and applications,” *Wireless Communications and Mobile Computing*, vol. 2022, Jul. 2022, doi: 10.1155/2022/9261537.

[38] B. Khemani, S. Patil, K. Kotecha, and S. Tanwar, “A review of graph neural networks: Concepts, architectures, techniques, challenges, datasets, applications, and future directions,” *Journal of Big Data*, vol. 11, no. 18, Jan. 2024, doi: 10.1186/s40537-023-00876-4.

[39] G. Lachaud, P. Conde-Cespedes, and M. Trocan, “Comparison between ınductive and transductive learning in a real citation network using graph neural networks,” in *2022 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2022, pp. 534-540, doi: 10.1109/ASONAM55673.2022.10068589.

[40] T. Çakar, Y. Girişken, E. Tuna, G. Filiz, and Y. Drias, "Neural decoding of brand perception and preferences: Understanding consumer behavior through fNIRS and machine learning," in *32nd Signal Processing and Communications Applications Conference (SIU)*, May. 2024, pp. 1-4, doi: 10.1109/SIU61531.2024.10601072.

[41] Functional Brain Imaging in Natural Environments. fnirdevices.com. https://fnirdevices.com (accessed May. 12, 2024).

[42] H. Ayaz, P. Shewokis, A. Curtin, M. İzzetoğlu, K. İzzetoğlu, and B. Onaral, “Using MazeSuite and functional near infrared spectroscopy to study learning in spatial navigation,” *Journal of Visualized Experiments*, vol. 56, Oct. 2011, doi: 10.3791/3443.

[43] H. Ayaz, S. Bunce, P. Shewokis, K. İzzetoğlu, B. Willems, and B. Onaral, “Using brain activity to predict task performance and operator efficiency,” in *Advances in Brain Inspired Cognitive Systems. BICS 2012. Lecture Notes in Computer Science*, 2012, vol. 7366, pp. 147-155, doi: 10.1007/978-3-642-31561-9_16.

[44] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825-2830, 2011, [Online]. Available: http://jmlr.org/papers/v12/pedregosa11a.html

[45] Scikit-learn User Guide. “6.4.3. Multivariate feature imputation.” scikit-learn.org. https://scikit-learn.org/stable/modules/impute.html#iterative-imputer (accessed May. 12, 2024).

[46] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5-32, Oct. 2001, doi: 10.1023/A:1010933404324.

[47] Scikit-learn User Guide. “1.13.5. Sequential Feature Selection.” scikit-learn.org. https://scikit-learn.org/stable/modules/feature_selection.html#sequential-feature-selection (accessed May. 12, 2024).

[48] S. Raschka. “SequentialFeatureSelector: The popular forward and backward feature selection approaches (including floating variants).” mlxtend. https://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/ (accessed May. 12, 2024).

[49] S. Raschka, “Mlxtend: providing machine learning and data science utilities and extensions to python’s scientific computing stack,” *Journal of Open Source Software*, vol. 3, no. 24, pp. 638, Apr. 2018, doi: 10.21105/joss.00638.

[50] Scikit-learn User Guide. “2.5.1. Principal component analysis (PCA).” scikit-learn.org. https://scikit-learn.org/stable/modules/decomposition.html#pca (accessed May. 12, 2024).

[51] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2016, pp.785-794, doi: 10.1145/2939672.2939785.

[52] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*, 2nd ed. Sebastopol, CA, USA, O’Reilly Media, Inc., 2019.

[53] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*, 1st ed. Boca Raton, FL, USA. Chapman and Hall/CRC, 1984.

[54] A. Abylkassymova, “Machine learning method for detecting botnet attacks originated from the IoT networks,” M.S. thesis, Dept. Cyber Security, Univ. Tartu, Tartu, Estonia, 2022.

[55] P. Domingos and M. Pazzani, “On the optimality of the simple bayesian classifier under zero-one loss,” *Machine Learning*, vol. 29, pp. 103-130, Nov. 1997, doi: 10.1023/a:1007413511361.

[56] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, “Machine learning: A review of classification and combining techniques,” *Artif Intell Rev*, vol. 26, no. 3, pp. 159–190, Nov. 2006, doi: 10.1007/s10462-007-9052-3.

[57] A. Zafar, *et al*., “A hybrid gcn and filter-based framework for channel and feature selection: An fnirs-bci study,” *International Journal of Intelligent Systems*, vol. 2023, Mar. 2023, doi: 10.1155/2023/8812844.

[58] J. Nylén, “Exploring ways of visualizing functional connectivity,” M.S. thesis, Dept. Applied Physics and Electronic, Umea Univ., Umea, Sweden, 2017.

[59] PyG Colab Notebooks and Video Tutorials. “Graph Classification with Graph Neural Networks.” pytorch-geometric. https://pytorch-geometric.readthedocs.io/en/latest/get_started/colabs.html (accessed May. 28, 2024).

[60] T. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *ICLR 2017*, 2017, doi: 10.48550/arxiv.1609.02907.

[61] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lió, and Y. Bengio, “Graph attention networks,” in *ICLR 2018,* 2018, doi: 10.48550/arxiv.1710.10903.

[62] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," in *ICLR*, Feb. 2019, doi: 10.48550/arxiv.1810.00826.

[63] H. Alibrahim and S. A. Ludwig, "Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, 2021, pp. 1551-1559, doi: 10.1109/CEC45853.2021.9504761.

[64] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: a big comparison for nas," Dec. 2019, doi: 10.48550/arxiv.1912.06059.

[65] S. Raschka, “Model evaluation, model selection, and algorithm selection in machine learning,” Nov. 2018, doi: 10.48550/arXiv.1811.12808.

[66] C. Gastaud, T. Carniel, and J. Dalle, "The varying importance of extrinsic factors in the success of startup fundraising: competition at early-stage and networks at growth-stage," Jun. 2019, doi: 10.48550/arxiv.1906.03210.