**ABSTRACT**

**ADVANCING PREDICTION OF STIMULANT MEDICATION MISUSE THROUGH GRAPH REPRESENTATION LEARNING**

**by**

**Hamid Razavi**

The misuse of stimulant prescription medications poses a significant and escalating public health concern in the United States, particularly among young adults. Addressing this issue requires sophisticated methodologies capable of uncovering complex patterns and relationships in data. Geometric Deep Learning, a paradigm designed to analyze data with non-Euclidean structures, has achieved remarkable success across various domains, offering a powerful framework for tackling complex graph structure data challenges.

This study leverages Graph Convolutional Networks (GCNs) to predict the likelihood of stimulant medication misuse using data from the National Survey on Drug Use and Health (NSDUH). Individuals are represented as nodes in a graph, while their relationships form edges, capturing social, behavioral, and contextual factors. Experimental results demonstrate the superior performance of the GCN model, achieving 96.1% accuracy, 92.93% precision, and 96.40% recall. can you add the precision and recall too? when compared to traditional machine learning approaches -- Support Vector Machines, k-Nearest Neighbors, Multi-Layer Perceptron, Random Forests, Gaussian Naïve Bayes, and Logistic Regression. The findings underscore the transformative potential of GNNs in predicting stimulant misuse and guiding the design of data-driven prevention strategies. To enhance interpretability, the study integrates GNN Explainer, a tool designed to identify feature importance and explain the model's predictions. By analyzing the critical features influencing misuse predictions, this approach provides great insights into the 20 key factors driving stimulant misuse.

**ADVANCING PREDICTION OF STIMULANT MEDICATION MISUSE THROUGH GRAPH REPRESENTATION LEARNING**

by

**Hamid Razavi**

A Thesis

Submitted to the Faculty of

New Jersey Institute of Technology

In Partial Fulfillment of the Requirements for the Degree of

**Master of Science in Data Science**

**Department of Data Science**

**December 2024**

APPROVAL PAGE

**ADVANCING PREDICTION OF STIMULANT MEDICATION MISUSE THROUGH GRAPH REPRESENTATION LEARNING**

**Hamid Razavi**

Mengjia Xu, Dissertation Advisor

Assistant Professor, Data Science Department

Hai Phan, Committee Member

Associate Professor, Data Science Department

Mengnan Du, Committee Member

Assistant Professor, Data Science Department

Lijing Wang, Committee Member

Assistant Professor, Data Science Department

**BIOGRAPHICAL SKETCH**

**Author:** Hamid Razavi

**Degree:** Master of Data Science

**Date:**  December 2024

**Date of Birth:** July 10, 1985

**Place of Birth:** Tehran, Iran

**Undergraduate and Graduate Education:**

* Master of Science in Data Science

New Jersey Institute of Technology, Newark, New Jersey 2024

* Bachelor of Science in NanoEngineering

University of California, San Diego, La Jolla, California 2018

* Bachelor of Arts in Political Science

University of California, San Diego, La Jolla, California 2018

ACKNOWLEDGMENTS

I wish to convey my heartfelt appreciation to my advisor, Dr. Mengjia Xu, for her supportive nature, valuable insights, and understanding during my academic pursuits, particularly considering my demanding work commitments. Her encouragement has been crucial to my development and the successful completion of this thesis.

I am also incredibly thankful to the members of my defense committee, Dr. Hai Phan, Dr. Mengnan Du, and Dr. Lijing Wang, for their thoughtful feedback, constructive critiques, and the time they invested in guiding my research.

I am immensely thankful to my mother, Fara, whose sacrifices, love, and steadfast faith have instilled in me resilience and compassion, propelling my journey forward. To my partner, Sahar, I am grateful for your constant support and encouragement, which have helped me navigate challenges and celebrate my successes. My sister, Melika, has served as a source of positivity and strength, providing wise advice and unwavering faith in my abilities. I also want to extend my heartfelt appreciation to my friends, whose support through conversations, encouragement, and shared experiences has been invaluable. Lastly, I recognize the professors, staff, and everyone else who played a role behind the scenes in my success—your contributions have made this milestone achievable, and I am deeply appreciative.

Table of Contents

Chapter Page

[1. INTRODUCTION 1](#_Toc185020464)

[1.1 Background 1](#_Toc185020465)

[1.2 Objective 2](#_Toc185020466)

[2. METHODOLOGIES 4](#_Toc185020467)

[2.1 Classic Machine Learning Models 7](#_Toc185020469)

[2.1.1 Random Forest 7](#_Toc185020470)

[2.1.2 Feature Importance Random Forest 8](#_Toc185020471)

[2.1.3 Gaussian NB 9](#_Toc185020472)

[2.1.4 Logistic Regression 11](#_Toc185020473)

[2.1.5 Support Vector Machine 12](#_Toc185020474)

[2.1.6 K-Nearest Neighbors 14](#_Toc185020475)

[2.1.7 Multi-Layer Perceptron 15](#_Toc185020476)

[2.2 Graph Convolutional Networks 17](#_Toc185020477)

[2.2.1 GCN Model Architecture 17](#_Toc185020478)

[2.2.2 Graph Construction with KNNs 18](#_Toc185020479)

[2.2.3 Data Preparation for the GCN 18](#_Toc185020480)

[2.2.4 GNN Explainer 19](#_Toc185020481)

[3. RESULTS AND DATA ANALYSIS 21](#_Toc185020482)

[3.1 Dataset Descriptions 21](#_Toc185020483)

[3.2 Random Forest 27](#_Toc185020484)

Table of contents

(continued)

Chapter Page

[3.3 Gaussian NB 31](#_Toc185020485)

[3.4 Logistic Regression 33](#_Toc185020486)

[3.5 SVM 34](#_Toc185020487)

[3.6 KNN 36](#_Toc185020488)

[3.7 MLP 37](#_Toc185020489)

[3.8 GCN 39](#_Toc185020490)

[3.9 Feature Importance 44](#_Toc185020491)

[3.10 Explainer Subgraph Visualization 47](#_Toc185020492)

[4. CONCLUSIONS 51](#_Toc185020493)

[4.1 Potential Future Work 53](#_Toc185020494)

[APPENDIX A 55](#_Toc185020495)

[PARAMETER SENSITIVITY ANALYSIS RESULTS 55](#_Toc185020496)

[REFERENCES 60](#_Toc185020497)

|  |  |  |
| --- | --- | --- |
|  | LIST OF TABLES |  |
| Table | Page | |
| 3.1 | Confusion Matrix for the Random Forest Model Highlighting Performance Metrics | 27 |
| 3.2 | Confusion Matrix for the Gaussian NB Model Highlighting Performance Metrics | 31 |
| 3.3 | Confusion Matrix for the Logistic Regression Model Highlighting Performance Metrics | 33 |
| 3.4 | Confusion Matrix for the SVM Model | 35 |
| 3.5 | Confusion Matrix for the SVM Model Highlighting Performance Metrics | 36 |
| 3.6 | Confusion Matrix for the MLP Model | 38 |
| 3.7 | List of the Top 20 Most Important Features and Their Descriptions | 47 |
| 4.1 | Comparison of Performance for the Seven Models Compared to One Another | 54 |
| A.1 | List of the Attributes Used in the Thesis | 56 |

|  |  |  |
| --- | --- | --- |
|  | LIST OF FIGURES |  |
| Figure |  | Page |
| 3.1 | Histogram of the AGE3 variable, showcasing the distribution of age in the dataset. | 24 |
| 3.2 | Correlation Heatmap highlighting the correlation between features of the dataset. | 25 |
| 3.3 | Feature importance highlighting the top twenty most important attributes of the dataset. | 29 |
| 3.4 | Visualization of the graph structure constructed using the KNNs for the NSDUH dataset. | 40 |
| 3.5 | Train and Test Accuracy plot for the GCN model. | 44 |
| 3.6 | GCN Explainer showcasing the top 20 most important features. | 47 |
| 3.7 | Subgraph visualization highlights the important nodes and edges. | 51 |

**CHAPTER 1**

# INTRODUCTION

## Background

Prescription stimulants, such as methylphenidate (for instance, Ritalin) and amphetamine derivatives (like dextroamphetamine; Adderall), have received approval from the U.S. Food and Drug Administration for treating attention deficit hyperactivity disorder (ADHD). Stimulants have been shown to be the most effective option among pharmacological treatments for the condition, while non-stimulants like atomoxetine, guanfacine, and clonidine are regarded as alternative treatment options (Zhao et al., 2022).

Attention-deficit hyperactivity disorder (ADHD) is a neurodevelopmental disorder characterized by inattention, hyperactivity, and impulsivity, causing functional impairment (National Institute of Mental Health, n.d.). Its prevalence lies at approximately 5% in children and adolescents and at approximately 2.5% in adults. The disorder follows a multifactorial etiology and shows a high heritability. Patients show a high interindividual and intraindividual variability of symptoms, with executive deficits in several cognitive domains. Overall, ADHD is associated with high rates of psychiatric comorbidities, and insufficient treatment is linked to adverse long-term outcomes (Kipf & Welling, 2021).

It is thought that stimulants promote their effects by affecting the dopaminergic and noradrenergic systems and fostering an increase in these neurotransmitters within the synaptic cleft. When taken as directed, prescription stimulants do not pose major health risks to users (U.S. Drug Enforcement Administration, n.d.). The side effects of these stimulants depend on the dosage, with the most frequently reported ones being reduced appetite, weight reduction, headaches, insomnia, stomach pain, dizziness, nervousness, mood instability, and dry mouth. More severe reactions, such as psychosis, seizures, and cardiovascular events like rapid heartbeat, high blood pressure, heart attacks, and sudden death, are rarely documented in individuals taking the therapeutic doses of these medications orally. The method of administration also influences the potential negative effects. Taking the drugs via intravenous or intranasal routes (i.e., snorting) significantly increases the risks associated with prescription stimulants (Teter et al., 2006). Additionally, prolonged use of high doses of prescription stimulants may raise the risk of cardiovascular adverse effects; however, studies involving children and young adults were deemed underpowered to identify such an increased risk.

The United States Drug Enforcement Administration (DEA) categorizes the stimulants as Schedule II drugs, indicating a significant risk for abuse and dependency (Drug Enforcement Administration [DEA], U.S. Department of Justice, 2015). Despite the risks to health and legal issues, the misuse of prescription stimulants—which generally refers to consuming them without a legitimate prescription or using them in ways other than prescribed—has emerged as a considerable concern in both the United States and internationally, particularly on college campuses (Weyandt et al., 2016).

Given the various negative consequences of stimulant misuse, it is imperative that we create solutions to tackle this problem at the individual, public health, and societal levels. On an individual basis, decreasing stimulant misuse can lead to a reduction in health complications, dependency issues, and legal troubles, ultimately enhancing overall well-being. From a public health standpoint, successful prevention efforts ease the burden on healthcare systems and foster improved health results for communities. At the societal level, reducing misuse helps to lessen legal problems and provides valuable data to guide policy decisions. Thus, it is critical to formulate strategies to combat this challenge. A vital part of this effort is recognizing the behaviors and trends related to medication misuse, which allows for the implementation of focused and effective prevention initiatives (United Nations Office on Drugs and Crime [UNODC], 2019).

## Objective

The thesis proposed the first graph-based machine learning model for predicting stimulant misuse, alongside various machine learning techniques applied to the 2022 National Survey on Drug Use and Health (NSDUH) dataset. The main goal is to identify distinctive patterns and risk factors associated with stimulant misuse, utilizing advanced Graph Convolutional Networks (GCNs) and traditional machine learning algorithms such as the Logistic Regression, Random Forest, Multi-Layer Perceptron (MLP), Support Vector Machine (SVM), k-Nearest Neighbors (KNN), and Gaussian Naïve Bayes. Among the models, the GCN model emerged as the most successful, achieving a notable accuracy of 96.1% with 91.21% precision, and 95.50% recall. In addition to the prediction results, the project also further investigated the GNN explainability. This is crucial for understanding the underlying decision-making processes of the GCN model, identifying key features or relationships influencing predictions of stimulant misuse, and ensuring transparency. By integrating innovative graph-based methods with well-established machine learning techniques, this study makes meaningful contributions to predictive modeling in public health, setting the stage for more focused prevention and intervention strategies.

**CHAPTER 2**

# METHODOLOGIES

The approach for forecasting stimulant misuse employs a dual methodology that integrates both traditional machine learning (ML) models and sophisticated graph-based methods. Traditional ML models such as Logistic Regression, Random Forest, Multi-Layer Perceptron (MLP), Support Vector Machine (SVM), k-Nearest Neighbors (KNN), and Gaussian Naïve Bayes are used to set a benchmark for prediction accuracy and evaluate their effectiveness on structured, tabular datasets. Concurrently, advanced Graph Convolutional Networks (GCNs) utilize the existing relationships and dependencies in the data by representing individuals as nodes and their interactions as edges within a graph framework. This method allows for the identification of intricate, non-linear patterns that traditional approaches might overlook. To further boost understanding and confidence in the predictions made by the GCN, interpretability is examined through the GNN Explainer, which highlights the key graph features and connections that influence the model's choices. This all-encompassing methodology ensures a strong balance between effective predictive capability and transparency in interpretation.

## Classic Machine Learning Models

### 2.1.1 Random Forest

Random Forests are ensemble learning techniques that merge multiple decision trees to boost predictive performance and reliability, especially for classification and regression tasks. By combining the results from different trees, Random Forests reduce the problem of overfitting and enhance stability when compared to single decision trees. The fundamental principle behind Random Forests is ensemble learning, where various models come together to establish a more dependable predictive framework. In this case, the ensemble contains numerous decision trees, leading to more generalized and precise predictions than a standalone tree. Each tree in the Random Forest is trained independently, and their collective output lessens overfitting by balancing the biases of individual trees (Breiman, 2001).

The Random Forest algorithm follows three primary steps. Initially, during the training phase, multiple decision trees are created using distinct subsets of the data that are sampled with replacement, a method referred to as bootstrap sampling. This technique fosters diversity in the datasets used for each tree, which enhances the overall robustness of the forest and decreases model variance. Next, each tree is trained on its respective bootstrapped dataset, selecting a random subset of features at each split. This random selection guarantees the uniqueness of each tree and prevents dependence on particular features. Lastly, the combined results from all the trees yield the ultimate prediction. For classification problems, the model decides based on the majority vote among all trees, while for regression scenarios, it calculates the average of the trees' outputs.

Random Forests provide numerous benefits. By consolidating predictions from multiple trees, the algorithm significantly lowers the likelihood of overfitting, a prevalent challenge with standalone decision trees. The random feature selection during training also equips the model to effectively manage high-dimensional data. Furthermore, since each tree is trained on diverse data subsets, Random Forests maintain resilience against outliers. Another advantage is that Random Forests can evaluate the relative importance of features, making them useful for feature selection and offering interpretative insights.

Nonetheless, Random Forests have certain drawbacks. Training a large number of trees can demand significant computational resources, particularly with extensive datasets, rendering the model resource heavy. Additionally, the ensemble characteristic of Random Forests diminishes interpretability in contrast to single decision trees because the roles of individual trees are less clear. Moreover, the requirement to consider multiple trees for predictions can delay prediction times, making Random Forests less appropriate for applications that require real-time results (Louppe, 2014).

Crucial hyperparameters of Random Forests consist of the number of trees, maximum depth of the trees, minimum samples necessary per leaf, and the count of features analyzed at each split. Tuning these hyperparameters enables control over the model's complexity and aids in reducing the risk of overfitting.

### 2.1.2 Feature Importance Random Forest

In a Random Forest model, feature importance quantifies the significance of each feature when it comes to making predictions. This metric shed light on which features most significantly affect the model's decisions, serving as a valuable resource for both comprehending the model and choosing relevant features. Feature Importance in Random Forest is determined by analyzing how each feature contributes to the reduction of impurity across all the trees in the forest. In this context, impurity refers to how mixed the classes are within a particular node. Common methods to measure impurity include Gini impurity for classification tasks and mean squared error for regression tasks. When a feature facilitates a node split, it typically leads to a decrease in impurity, resulting in a more distinct and informative division.

The process of calculating feature importance involves several steps. Initially, every time a feature contributes to a node split, the resulting drop in impurity is documented. This reduction is then accumulated across all nodes and trees in the forest for each feature. Ultimately, these total reductions are normalized to yield a relative importance score for each feature. All of these scores total to one, simplifying the comparison of each feature's importance. Feature importance scores play a crucial role in pinpointing the most influential predictors within a dataset. Features with high importance have a greater impact on the model’s accuracy and provide critical insights into the data. Additionally, these scores aid in feature selection: by eliminating less significant features, we can streamline the model without a notable loss in performance, which may help reduce noise and improve interpretability (Agarwal et al., 2023).

### 2.1.3 Gaussian NB

Gaussian Naive Bayes (Gaussian NB) is a classifier that uses probability and is founded on Bayes' theorem, operating under the assumption that features are conditionally independent when the class label is known. This technique is particularly effective for classification tasks involving continuous data that can be approximated as normally distributed. The term "Gaussian" in Gaussian NB signifies that the model presumes each feature adheres to a Gaussian (normal) distribution. For instance, if a dataset contains two features, Gaussian NB assumes that for each class, both features independently conform to a bell-shaped normal distribution (Webb, 2010).

To generate predictions, Gaussian NB assesses the probability of each class for a specific data point by evaluating the likelihood of the feature values according to the Gaussian distribution assumption. This likelihood is derived using the Gaussian probability density function, which is defined by the mean and variance of the features for each class. During the training phase, Gaussian NB estimates these mean and variance values from the training dataset. When a new instance needs to be classified, Gaussian NB calculates how likely it is to observe that instance's feature values for each class by applying the Gaussian distribution formula, while assuming the independence of each feature. This process is repeated for all classes, and the class that has the highest probability is chosen as the prediction.

Gaussian NB finds frequent use in practical applications where the features are approximately normal and can be evaluated independently, such as spam detection or document categorization (Jurafsky & Martin, 2021). Although it operates under the premise of feature independence—an assumption that seldom holds true in actual datasets—Gaussian NB often yields satisfactory results in practice, particularly when normality is somewhat valid. The model is quick to train, necessitates minimal configuration, and performs well even with high-dimensional datasets. However, Gaussian NB may face challenges with datasets that significantly deviate from the assumptions of independence or normality, and it is typically surpassed by more advanced models in intricate tasks. Nevertheless, its ease of use and speed make Gaussian Naive Bayes a useful method in numerous machine learning applications.

### 2.1.4 Logistic Regression

Logistic regression is a popular statistical technique for binary classification, aiming to predict one of two outcomes based on input variables. In contrast to linear regression, which forecasts continuous values, logistic regression calculates the likelihood of an input belonging to a specific category, making it ideal for classification tasks. A defining characteristic of logistic regression is its utilization of the logistic (or sigmoid) function, which converts predictions into probabilities ranging from 0 to 1 (Google Developers, 2024).

The model functions by computing a linear combination of the input features alongside their respective coefficients. This combination is then fed into the logistic function to generate a probability score. Based on this score, the model assigns a class label to the input, typically applying a threshold of 0.5. During the training phase, logistic regression fine-tunes the model's coefficients to align the predicted probabilities with the actual class labels in the dataset. This adjustment process employs a technique known as Maximum Likelihood Estimation (MLE) to determine the optimal set of coefficients.

One major benefit of logistic regression is its ease of interpretation. Each coefficient signifies the relationship between an input variable and the outcome, specifically illustrating how changes in the variable influence the likelihood of the result. For instance, a positive coefficient suggests that an increase in the predictor enhances the probability of the event occurring. This transparency renders logistic regression particularly valuable in fields such as healthcare, social sciences, and business analytics, where comprehending the effects of variables is essential (Owen & Roediger, 2014).

Logistic regression can also be adapted to address classification problems involving more than two classes. This adaptation can be accomplished through techniques such as one-vs-rest (where a binary classifier is trained for each class) or multinomial logistic regression, which addresses all classes at once. Although logistic regression is efficient when a linear relationship exists between the predictors and the outcome, it may encounter difficulties with more intricate, non-linear patterns. In these situations, more sophisticated models may yield improved results.

Despite its straightforwardness, logistic regression continues to be a robust and frequently utilized method for both binary and multiclass classification tasks, owing to its combination of efficiency, precision, and interpretability.

### 2.1.5 Support Vector Machine

The Support Vector Machine (SVM) is a robust supervised machine learning algorithm designed for both classification and regression tasks, although it is predominantly recognized for its exceptional performance in classification scenarios. The fundamental concept of SVM is to identify a hyperplane that optimally separates data points into distinct categories with the largest possible margin. This margin represents the distance between the support vectors, which are the closest data points from each class, and the hyperplane. By striving to maximize this margin, SVM seeks to establish a boundary that generalizes well to new, unseen data, thereby improving its reliability and accuracy (Scikit-learn, n.d.).

For basic binary classification tasks, SVM operates by determining a linear hyperplane (a line in two dimensions or a plane in three dimensions) if the data can be cleanly separated. The goal of the algorithm is to maximize the margin to position the decision boundary as far away from the nearest data points of each category as possible. These closest points, known as support vectors, are vital as they determine the precise location of the hyperplane. Other data points that are located further away from the boundary do not influence the model (IBM, 2023).

In situations where the data cannot be linearly separated, SVM employs a method referred to as the kernel trick. This technique transforms the data into a higher-dimensional space, making it easier to separate using a hyperplane. For example, employing a radial basis function (RBF) kernel or a polynomial kernel can project the data into higher dimensions, enabling SVM to manage complex, nonlinear relationships among data points. The versatility in selecting different kernels enhances SVM's effectiveness in a wide array of complicated tasks, even with elaborate data structures.

A significant advantage of SVM is its capability to perform effectively on high-dimensional datasets, particularly in scenarios where the number of features surpasses the number of samples, as it can adeptly navigate large feature spaces. Moreover, SVM shows a relatively high resistance to overfitting, particularly with an adequate margin, although it is crucial to fine-tune parameters such as the regularization parameter C and kernel parameters to achieve optimal performance (Shamsi & Beheshti, 2023).

Nonetheless, SVM has its drawbacks. It can become computationally demanding with large datasets, especially when complex kernels are involved, and it is sensitive to the selection of kernel and tuning parameters. Additionally, compared to simpler models, SVM can be more difficult to interpret. Despite these challenges, SVM remains a potent tool for classification tasks, delivering high accuracy and adaptability for a broad range of applications, including image recognition and bioinformatics (Scikit-learn, n.d.).

### 2.1.6 K-Nearest Neighbors

The K-Nearest Neighbors (KNN) algorithm is a straightforward, yet powerful machine learning method frequently utilized for both classification and regression problems. As a non-parametric, instance-based learning technique, it does not rely on a specific data distribution and predicts outcomes by comparing the similarity between new inputs and existing instances. Essentially, KNN functions by locating the ‘k’ nearest data points to a specific input and generating predictions based on their attributes, typically through majority voting for classification or averaging for regression (Brownlee, 2020).

In the classification context, the algorithm determines a class for a new data point by analyzing its k nearest neighbors within the feature space. For example, if k=3, KNN will assess the three closest data points to the input instance. The class that appears most frequently among these three neighbors is assigned as the prediction for the new instance. This characteristic makes KNN a logical and transparent approach, as predictions are directly affected by nearby data points. Similarly, in the regression task, KNN derives a numerical prediction for a new instance by averaging the values of its k nearest neighbors (GeeksforGeeks, 2023).

The choice of distance metric is vital in KNN, as it specifies which points are considered nearest. Commonly applied distance metrics include Euclidean distance for continuous data and Manhattan distance for categorical or grid-like variables. Choosing a suitable distance metric is crucial, as it significantly influences the model's performance and accuracy. Furthermore, the selection of k the number of neighbors can greatly affect the algorithm. A small k value can lead to excessive sensitivity to noise, while a larger k value might cause over-generalization. Practitioners often employ cross-validation to determine an optimal k value that strikes a balance between these factors.

One notable advantage of KNN is its straightforwardness and minimal training time, as it does not utilize a training phase in the conventional way. Instead, all calculations take place during the prediction stage. Nonetheless, this can also be a drawback since generating predictions can be computationally intensive for large datasets, requiring KNN to compute distances from the input to all training data points (RoboticsBiz, 2022).

Despite its simplicity, KNN can yield strong performance in numerous real-world applications, particularly in cases with irregular or complex class boundaries. It is regularly used in fields such as recommendation systems, image classification, and anomaly detection. However, KNN is sensitive to feature scaling, making data normalization necessary in many situations. Overall, KNN serves as a useful tool, especially when interpretability and simplicity are paramount, though it may need modifications for better scalability and efficiency in large, high-dimensional datasets.

### 2.1.7 Multi-Layer Perceptron

A Multi-Layer Perceptron (MLP) is a kind of artificial neural network frequently utilized for supervised learning applications such as classification and regression. An MLP consists of several layers of interconnected nodes (neurons), with each layer fully connected to the following one. These layers comprise an input layer, one or more hidden layers, and an output layer. MLPs are particularly adept at identifying intricate patterns in data through these multiple layers, especially when non-linear activation functions are employed (Scikit-learn: Machine Learning in Python, n.d.).

In an MLP, data travels through the network in a single direction—from the input layer to the output layer—characterizing it as a "feedforward" network. Each neuron in the input layer corresponds to a specific feature of the dataset. These neurons transmit their values to the hidden layer, where each neuron calculates a weighted sum of its inputs, applies an activation function (such as ReLU, sigmoid, or tanh), and then forwards the result to the next layer. Activation functions introduce non-linearity, allowing MLPs to recognize more complex connections.

During training, the MLP improves its performance by modifying the weights in each layer to lessen the discrepancy between its predictions and the actual outcomes. This method is known as backpropagation, which employs a gradient-based optimization technique (usually stochastic gradient descent). In backpropagation, the error from the output layer is sent backward through the network, enabling each neuron to adjust its weights to reduce the error in future predictions (Cohn, 2017).

One benefit of MLPs is their adaptability and capacity to approximate a diverse range of functions, which makes them well-suited for demanding tasks like image and speech recognition. However, they demand large quantities of labeled data and substantial computational resources, particularly as the number of layers and neurons grows. Additionally, MLPs may have difficulties dealing with sequential data or spatial relationships, which is why alternative architectures, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are often favored for specific tasks (33rd Square, 2023).

## Graph Convolutional Networks

### 2.2.1 GCN Model Architecture

The concept of a graph's structure can be grasped through the principle of Message Passing. Message Passing refers to the ability of a node within a graph to communicate and exchange information with its neighboring nodes. This occurs in two phases: initially, nodes transmit messages about themselves to neighboring nodes, followed by receiving messages from those neighbors to refine their understanding of the environment. Each node collects all attribute vectors from its adjacent nodes and utilizes an aggregation function, like averaging. This compiled vector is then processed through a neural network, achieved by multiplying it with a matrix and subsequently applying an activation function, such as ReLU in this scenario. The output from this dense layer forms a new vector representation for the node. In the subsequent layer, the input consists of the updated vectors from the first layer. The number of nodes resulting from the GCN layer corresponds to the count of columns or features, in other terms. The binary cross-entropy loss function is employed in conjunction with backpropagation to optimize all parameters (Fey & Lenssen, 2019).

The Graph Convolutional Network model initiates with the first convolution layer, which performs graph convolutions to capture node-level features while considering the structure of the graph. Subsequently, a ReLU activation function is applied to introduce non-linearity, thereby improving feature learning. To mitigate overfitting, a dropout layer is utilized, randomly omitting 50% of the neurons during the training process. The second convolution layer subsequently produces the final node embeddings and output logits for classification. These logits are then transformed into probabilities using the SoftMax function. The model undergoes training for 1000 epochs within a loop, where the cross-entropy loss is minimized, and the weights are adjusted using the Adam optimizer.

### Graph Construction with KNNs

To construct the graph, an adjacency matrix A is created to depict connections, where non-zero entries indicate the distances between connected nodes. A weighted K-Nearest Neighbors (KNN) graph is formed using the kneighbors\_graph function, setting k=11 to link each node to its 11 closest neighbors based on distance. The resulting adjacency matrix has a size of 5000×5000 and is symmetric. This sparse adjacency matrix is then transformed into a NetworkX graph G, which simplifies the examination of edges and their weights, thereby allowing for visualization and manipulation. Following this, the NetworkX graph is converted into a PyTorch Geometric Data object to enable additional processing. Edge weights that reflect distances are assigned to the data.edge\_weight attribute, which supports weighted graph operations. PyTorch Geometric represents graphs in a tensor format that is optimized for GPU processing, a key requirement for training Graph Neural Networks (GNNs). Given that the graph is undirected, edge weights are replicated to align with the bidirectional representation of edge\_index in PyTorch Geometric (PyTorch Geometric, 2023).

### Data Preparation for the GCN

The node features (X\_less\_rows) and labels (y\_less\_rows) are initially transformed into PyTorch tensors to enable compatibility with PyTorch functions. The features are expressed as FloatTensor, while the labels are converted to LongTensor to meet the GCN model's specifications. In a transductive GCN setup, the division of training and testing is done at the node level. This implies that the entire set of nodes and edges is utilized in the graph during training, with only certain nodes (test nodes) designated for evaluation. The model learns by capitalizing on the complete graph structure, although it does not account for the labels of test nodes throughout the training process. In contrast, when it comes to an inductive GCN setup, the split for training and testing occurs at the graph level, meaning that the test nodes or subgraphs are completely unobserved during training. This necessitates that the model effectively generalizes to new graphs or nodes. Transductive models leverage the overall graph structure by using all nodes to understand relationships, whereas inductive models showcase adaptability by generalizing to new graph data without prior exposure, making them appropriate for situations involving unseen data (Kipf & Welling, 2017).

### 2.2.4 GNN Explainer

The fusion of feature information with intricate combinatorial graph structures has led to the emergence of complex non-linear GNN models. This increased complexity has made it more challenging to understand the workings of GNNs and the rationale behind their predictions. To address this concern, numerous explainability techniques have been introduced to shed light on the internal functions of GNNs. Explainable GNNs improve safety and enhance trust in their recommendations (Chen et al., 2023).

In Graph Convolutional Networks (GCNs), it is essential to comprehend feature significance—the role that each feature plays in the predictions made by the model—for interpreting how the model draws information from graph data. One approach to evaluate feature significance in GCNs involves utilizing Graph Explainers. Graph Explainers are dedicated techniques aimed at pinpointing which features and nodes are pivotal in a GCN’s decision-making, thus shedding light on the model’s internal processes (Xu et al., 2019).

Graph Explainers like GNNExplainer and GraphLIME operate by highlighting the most relevant segments of a graph that pertain to a specific prediction. These techniques identify the key features and structural elements (nodes and edges) that affect the model’s output. For instance, GNNExplainer generates explanations by modifying node features and edges to observe the effect of these changes on predictions. By examining which alterations lead to significant shifts in predictions, GNNExplainer can determine the features and edges that are most critical for the GCN’s output. Likewise, GraphLIME employs a local surrogate model to simulate the GCN’s decision boundary and delivers an explanation at the feature level, providing insights into the significance of features across nodes (Huang et al., 2020).

These explainers provide importance scores for each feature within the graph. A higher importance score signifies features that considerably influence the prediction, assisting researchers in understanding which information the GCN prioritizes when making decisions. For instance, in a social network graph, Graph Explainers may indicate that attributes like connectivity and shared interests are more influential than other features, depending on the specific task at hand (Ying et al., 2019).

Graph Explainers improve the interpretability of GCNs, making them essential tools for elucidating intricate relationships in graph data. By delivering clear insights into which features drive predictions, they support feature engineering, bolster model reliability, and enable domain experts to make well-informed choices based on the model’s explanations (Pope et al., 2019).

**CHAPTER 3**

# RESULTS AND DATA ANALYSIS

## Dataset Descriptions

Every year, the United States civilian, noninstitutionalized population aged 12 and up is surveyed by the NSDUH. It is the primary source of statistics on the use of alcohol, tobacco, prescription psychotherapeutic medicines (such as sedatives, painkillers, tranquilizers, and stimulants), and other substances (such as cocaine and marijuana) by individuals in that population who are 12 years of age or older. In-depth data on substance use disorders (SUDs), substance use treatment, mental health conditions, and mental health therapy are also included in the survey (MarlonDaniel & samhsahhsgov, 2023).

Annual estimates regarding drug use and mental health among civilian individuals in the noninstitutionalized population of the United States mainly come from NSDUH, which serves as a nationally representative source. The substantial and widely dispersed NSDUH sample enables the generation of estimates at national, state, and substate levels. Most of the NSDUH questions administered in person utilize ACASI, which is designed to provide respondents with a very confidential and personal method for answering questions. This approach leads to more honest reporting of illegal drug use and other sensitive or stigmatized behaviors. Similarly, in online surveys, participants are asked to verify that they are in a private area of their home before starting the survey.

For the 2022 study, a final sample of 71,369 interviews was collected. The 2022 NSDUH had a weighted screening response rate of 25.46 percent and a weighted interview response rate of 47.43 percent, thanks to strategies for guaranteeing the highest participation rates. Forty-two percent of the interviews in the final sample were conducted online, while fifty-eight percent were done in person. Response rates from web-based data collection were still lower than those from in-person data gathering.

By keeping identifying information separate from survey responses, respondent anonymity and response privacy were maintained throughout the study. Respondents received assurances that their answers and names would be treated strictly in accordance with federal law. As was previously mentioned, both the questionnaire and the interviewing techniques were created to improve the confidentiality of answers, particularly during parts of the interview where delicate questions were asked. ACASI was used to collect sensitive interview questions during in-person interviews. However, during the online interview, the ACASI technology was unavailable. After consulting with subject matter experts, the decision was made to exclude ACASI from the web interviews. They concluded that providing audio would increase the risk of confidentiality breaches (allowing others to hear the questions being asked or the responses being entered) compared to not having any audio at all.

To generate objective estimates for survey outcomes in the population represented by the 2022 NSDUH, the data must be weighted because the estimates provided by NSDUH are based on sample survey data rather than complete data for the entire population. The number of sampling units in the NSDUH target population that the ith respondent represents can be seen as the “final analysis weight” of that respondent. The size of the entire target population is estimated by adding the weights of all respondents:

**= estimated size of the target population**, where the sum is over all respondents in the 2022 NSDUH.

This weight modification obtained an intermediate subsample weight (ANALWTADJ) by multiplying the initial analysis weight (ANALWT2)32. This phase ensures that the subsample's total weight (ANALWTADJ) is almost equal to the original sample's total weight (ANALWT2) while accounting for the respondents' varying subsampling rates.

In order for the sum of the final PUF weights (ANALWT2\_C) to equal the sum of ANALWT2 from the initial analytic data for a set of primary study variables and auxiliary variables (which were the chosen identifying variables, such as demographic variables), ANALWTADJ was further adjusted during the calibration step.

The study utilized a dataset comprising 5,000 individuals and an initial set of 2,606 variables. However, due to redundancy and minimal contribution to predictive models, only 204 features were selected before applying one-hot encoding. Categorical variables underwent one-hot encoding to convert them into a numerical format suitable for machine learning algorithms. This transformation significantly raised the total feature count to 1,304, as each categorical feature expanded into multiple binary columns, each signifying a distinct category. Although this led to an increase in dimensionality, it ensured that categorical data could be properly used in the models without introducing biases from ordinal interpretations. As illustrated in Figure 1 below, the majority of respondents are found in category 9, which represents the age range of 35 to 49 years old.

The dataset was divided into training and testing subsets with an 80/20 ratio, establishing a balanced framework for training the models while reserving a portion for assessment. Cross-validation was conducted using 3, 5, and 10 folds to evaluate model performance consistently across various data partitions, enhancing robustness and mitigating the risks of overfitting. Binary classification aimed to predict one of two possible outcomes: "Yes" or "No." While the overall dataset contained missing values, the features chosen were meticulously selected to exclude those with incomplete data. This strategy eliminated the necessity for imputation techniques or the removal of rows with NaN values, simplifying the preprocessing stage. Table 1A in the appendix section provides a detailed summary and description of the 204 features utilized in the analysis.

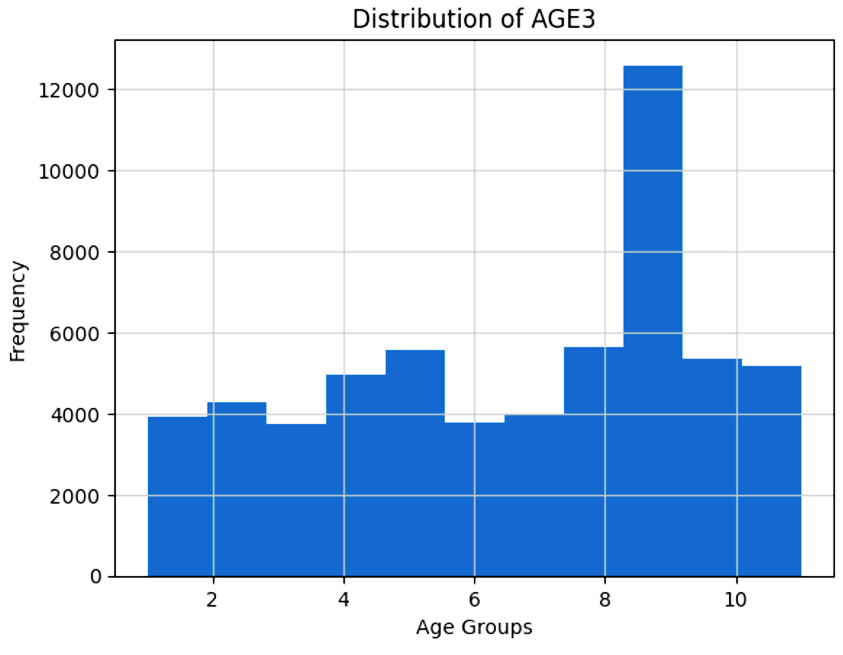


Figure 3.1 Histogram of the AGE3 variable, showcasing the distribution of age in the dataset.

**Feature correlation:**

**A screenshot of a computer

Description automatically generated**

Figure 3.2 Correlation Heatmap highlighting the correlation between features of the dataset.

The heatmap shown in figure 2 demonstrates the connections between various features within a dataset, with correlation coefficients ranging from -1 to 1. A coefficient of 1 indicates a perfect positive relationship, suggesting that both features increase simultaneously. Conversely, a coefficient of -1 represents a perfect negative correlation, meaning one feature's increase corresponds with a decrease in the other. Values approaching 0 imply a weak or non-existent correlation between the features. The diagonal line of the heatmap showcases values of 1, as each feature is perfectly correlated with itself by definition.

An important observation is the discovery of clusters of features that show a correlation of 1, indicating strong positive relationships among these variables. For example, features like "trqrsgdfx\_91.0" and "trqwygnmt\_91.0" fall within this category and are perfectly correlated, which suggests redundancy as they likely convey similar information.

Additionally, there are strong negative correlations to note, such as the -0.68 correlation between "stmnmflag" and "trqnmrec\_3.0". A significant negative correlation implies that an increase in one variable typically corresponds with a decrease in the other, potentially uncovering a meaningful inverse relationship between these features.

Certain features display lower positive correlations, exemplified by the approximately 0.27 correlation for "ecstmolly\_91.0", "peyote\_91.0", and "pcp\_91.0". These weaker correlations reflect a slight connection among the features, which could still be relevant for analysis or modeling objectives.

Moreover, some features show minimal significant correlation with others, exhibiting values close to 0. These uncorrelated features may offer distinct information and could be advantageous for predictive modeling, as they are not influenced by other variables in the dataset.

The heatmap also reveals clusters or groups of features that are heavily interconnected. For instance, the collection of features that begin with "trq..." indicates strong internal correlations, implying that these features are closely related and may represent similar aspects of the dataset.

These findings lead to several conclusions. Features with exact or nearly identical correlations can cause redundancy in a model. It might be beneficial to remove some of these variables to simplify the model and mitigate issues associated with multicollinearity. In contrast, features showing notable positive or negative correlations could uncover important relationships worth investigating further for deeper insights into the dataset. Features with correlations near 0 may provide unique information for modeling due to their independence.

To tackle these insights, techniques like Principal Component Analysis (PCA) could be used to address highly correlated features. Alternatively, directly eliminating redundant features may be an option. Methods for feature selection, such as statistical testing or evaluating feature importance, can help in identifying the most critical variables to keep. Focusing on features with strong correlations, whether positive or negative, may also highlight significant patterns and improve model performance.

## Random Forest

**Table 3.1** Confusion Matrix for the Random Forest model highlighting Performance Metrics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Test-Set (No CV)** | **3-Fold CV** | **5-Fold CV** | **10-Fold CV** |
| **Accuracy** | 0.9571 | 0.9591 ± 0.0013 | 0.9592 ± 0.0019 | 0.9592 ± 0.0024 |
| **Precision** | 0.9485 | 0.9508 ± 0.0022 | 0.9508 ± 0.0027 | 0.9510 ± 0.0038 |
| **Recall** | 0.9571 | 0.9591 ± 0.0013 | 0.9592 ± 0.0019 | 0.9592 ± 0.0024 |
| **F1-Score** | 0.9487 | 0.9511 ± 0.0010 | 0.9513 ± 0.0023 | 0.9515 ± 0.0030 |

Before executing the Random Forest model, hyperparameter tuning was conducted through Grid Search to determine the best settings for the model. Grid Search explored various combinations of hyperparameters, including the number of trees (n\_estimators), the maximum depth of each tree (max\_depth), and the minimum number of samples needed to split a node (min\_samples\_split). The optimal hyperparameters identified were max\_depth=16, min\_samples\_split=3, and n\_estimators=200.

The Random Forest model was then trained to forecast stimulant misuse using the chosen features. The performance of the model was assessed using a test set and cross-validation with different fold sizes (3, 5, and 10). Weighted metrics such as accuracy, precision, recall, and F1 score were computed to tackle any class imbalances present in the dataset.

The results shown in Table 1 from the test set reveal an accuracy of 0.9571, precision of 0.9485, recall of 0.9571, and an F1 score of 0.9487. These metrics indicate that the model performs effectively with new, unseen data. However, the results from cross-validation provide more detailed insights into the model's reliability. In the 3-fold CV, the average accuracy was 0.9591 with a standard deviation of 0.0013, while precision, recall, and F1 scores were similarly high, indicating minimal variability. The 5-fold and 10-fold CV results were also consistent, with mean accuracies of 0.9592 and standard deviations of 0.0019 and 0.0024, respectively. Precision scores fluctuated between 0.9508 and 0.9510, with equally low standard deviations, suggesting strong performance across all folds.

Despite its overall strength, the model may encounter difficulties in accurately predicting instances of stimulant misuse, potentially due to limitations in feature selection or unreported cases in the dataset. For example, although both accuracy and recall are high, precision indicates that there is still potential for improvement in recognizing true positive cases.

In summary, the Random Forest model shows impressive predictive performance and generalization abilities, as shown by the consistent metrics across both the test set and cross-validation folds. Future initiatives could aim at enhancing feature selection and addressing class imbalance to further improve the model's capacity to predict stimulant misuse.

**Random Forest Feature Importance**

A graph with blue and white stripes

Description automatically generated

**Figure 3.3** Feature importance highlighting the top twenty most important attributes of the dataset.

The bar chart in figure 3 presents the most significant features for a Random Forest model aimed at predicting stimulant misuse, using the Mean Decrease in Impurity (MDI) approach. This metric assesses the importance of each feature in reducing impurity within the decision trees, which in turn affects the model's predictions. The y-axis lists the various features, while the x-axis denotes their mean decrease in impurity, emphasizing their relevance to the model.

**Key Contributing Features:**

The top features include "trqrshook\_91.0", "hallucevr\_91.0", and "TRQNM30D\_91.0". These features show the highest mean decrease in impurity, revealing a strong connection with stimulant misuse. For instance, "trqrshook\_91.0" appears to encapsulate important behavioral trends or traits that affect misuse predictions. Other significant features, such as "mesc\_91.0" and "pnrrsdgfx\_91.0", highlight behavioral aspects and potentially substance-related factors.

**Behavioral and Substance Use Indicators:**

Features like "peyote\_2.0" and "psilcy\_91.0" indicate a link between particular substance use and stimulant misuse, emphasizing the impact of past or simultaneous substance experiences on the risk of stimulant misuse. Similarly, "hallucevr\_91.0", associated with hallucinogen experiences, ranks among the top features, showcasing its predictive capability.

**Demographic and Health Features:**

Features such as "cocrec\_nan" and "ecstmorrec\_91.0" reflect contextual health or demographic aspects, suggesting underlying socio-economic or health factors that affect stimulant misuse patterns. The presence of "trqnrec\_3.0" and "trqwyamnt\_91.0" underlines health or treatment-related indicators, indicating connections between medical or intervention histories and the likelihood of misuse.

**Socioeconomic and Contextual Factors:**

Features like "ccbstway\_91.0" and "cocever\_2.0" may capture socio-economic or access-related dynamics, illuminating external factors that influence stimulant misuse behavior.

**General Observations and Implications:**

The analysis of feature importance highlights the strong predictive capacity of behavioral and substance-use-related features in detecting stimulant misuse. The elevated position of features like "trqrshook\_91.0" and "hallucevr\_91.0" implies that behavioral patterns, substance use history, and treatment records are crucial for predictions. Health and demographic features, although significant, rank slightly lower than substance-specific elements, indicating a largely behavior-driven model. To improve predictive accuracy, investigating additional features or fine-tuning the contributions of lower-ranked features could help bridge gaps in identifying stimulant misuse among targeted subpopulations.

## Gaussian NB

**Table 3.2** Confusion Matrix for the Gaussian NB model highlighting Performance Metrics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Test-Set (No CV)** | **3-Fold CV** | **5-Fold CV** | **10-Fold CV** |
| **Accuracy** | 0.9080 | 0.9079 ± 0.0019 | 0.9079 ± 0.0020 | 0.9079 ± 0.0023 |
| **Precision** | 0.9523 | 0.9522 ± 0.0009 | 0.9523 ± 0.0016 | 0.9523 ± 0.0019 |
| **Recall** | 0.9080 | 0.9079 ± 0.0019 | 0.9079 ± 0.0020 | 0.9079 ± 0.0023 |
| **F1 Score** | 0.9250 | 0.9250 ± 0.0015 | 0.9250 ± 0.0014 | 0.9250 ± 0.0017 |

The results in table 2 are from the Gaussian Naive Bayes model, as shown in the table, display consistent performance metrics both before and after applying cross-validation with 3, 5, and 10 folds. An analysis of the metrics—accuracy, precision, recall, and F1 score—provides valuable insights into the model's reliability and suitability for the dataset.

Starting with accuracy, the model achieves a score of 0.9080 prior to cross-validation. Following the application of 3-fold, 5-fold, and 10-fold cross-validation, the accuracy reflects slight fluctuations around 0.9079, with standard deviations of 0.0019, 0.0020, and 0.0023, respectively. This consistency in accuracy across all folds indicates that the model performs well in generalizing to the dataset, with minor variations resulting from different train-test splits. The somewhat higher standard deviation with an increased number of folds is a predictable outcome due to the smaller test sets in each fold, resulting in slightly greater variance.

Precision remains steady, starting at 0.9523 before cross-validation and showing minimal changes across all folds. For 3-fold, 5-fold, and 10-fold cross-validation, the average precision values are 0.9522, 0.9523, and 0.9523, accompanied by standard deviations of 0.0009, 0.0016, and 0.0019, respectively. This consistency indicates that the model reliably identifies positive cases while minimizing false positives, a crucial element in datasets where the cost of classification errors differs.

Similarly, recall values demonstrate stability, beginning at 0.9080 before cross-validation and remaining consistent throughout all folds, with an average of 0.9079 and standard deviations of 0.0019, 0.0020, and 0.0023 for 3-fold, 5-fold, and 10-fold cross-validation, respectively. This performance highlights the model's ability to detect the majority of positive cases within the dataset with minimal variation, showcasing its effectiveness in reducing false negatives.

The F1 score, which reflects a balance between precision and recall, also displays solid performance, recording a score of 0.9250 before cross-validation and maintaining stable values across all folds. The average F1 score remains unchanged at 0.9250 for 3-fold, 5-fold, and 10-fold cross-validation, with standard deviations of 0.0015, 0.0014, and 0.0017, respectively. These results suggest that the model effectively balances both true positives and false negatives, maintaining an equilibrium between precision and recall.

In conclusion, the Gaussian Naive Bayes model exhibits consistency across all metrics, with minimal variability across cross-validation folds. This level of performance suggests that the model can generalize well to the dataset and can adeptly handle both class imbalance and feature interactions. The low standard deviations further confirm that the model's performance is stable despite varying train-test divisions, establishing it as a reliable option for classification tasks involving this dataset.

Given these results, the Gaussian Naive Bayes model is well-suited for the dataset, showcasing high precision and F1 scores that effectively address the trade-offs between false positives and false negatives. Future enhancements might include exploring various preprocessing techniques, adding new features, or engaging in feature engineering to uncover additional patterns within the data.

## Logistic Regression

**Table 3.3** Confusion Matrix for the Logistic Regression model highlighting Performance Metrics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Test-Set (No CV)** | **3-Fold CV** | **5-Fold CV** | **10-Fold CV** |
| **Accuracy** | 0.9587 | 0.9602 ± 0.0025 | 0.9607 ± 0.0028 | 0.9604 ± 0.0031 |
| **Precision** | 0.9512 | 0.9531 ± 0.0036 | 0.9537 ± 0.0038 | 0.9533 ± 0.0043 |
| **Recall** | 0.9587 | 0.9602 ± 0.0025 | 0.9607 ± 0.0028 | 0.9604 ± 0.0031 |
| **F1 Score** | 0.9517 | 0.9541 ± 0.0024 | 0.9546 ± 0.0032 | 0.9544 ± 0.0037 |

The evaluation of model performance in table 3 is using logistic regression with cross-validation (CV) employs various classification metrics: accuracy, precision, recall, and F1-score. The model undergoes training and evaluation across three different cross-validation folds: 3, 5, and 10. This methodology aids in assessing the model’s stability and reliability across varying data splits. The function compute\_metrics is responsible for calculating metrics on the test set, while cross\_validate\_and\_compute\_stats carries out CV, computes the average and standard deviation of each metric over the folds, and delivers a performance summary.

The Logistic Regression model shows strong and reliable performance in both the test set and cross-validation assessments. The test set accuracy stands at 0.9587, while similar figures are noted during cross-validation: 0.9602 for 3-fold, 0.9607 for 5-fold, and 0.9604 for 10-fold, all exhibiting low standard deviations. This consistency emphasizes the model's capability to generalize effectively to new data. Precision is also notably high, with a test set value of 0.9512 and slightly better averages achieved during cross-validation (0.9531 for 3-fold, 0.9537 for 5-fold, and 0.9533 for 10-fold). The stable precision results across assessments highlight the model’s competence in reducing false positives. Likewise, recall figures align closely with accuracy, at 0.9587 for the test set and averages of 0.9602, 0.9607, and 0.9604 for 3-fold, 5-fold, and 10-fold cross-validation, respectively, demonstrating the model's ability to effectively identify true positives.

The F1 Score, which serves to balance precision and recall, is recorded at 0.9517 on the test set and is slightly elevated during cross-validation, with averages of 0.9541, 0.9546, and 0.9544 for the 3-fold, 5-fold, and 10-fold splits, respectively. These findings suggest that the model sustains a robust equilibrium between minimizing false positives and false negatives across various data segments. In summary, the Logistic Regression model's steady metrics, alongside low variability across evaluations, underscore its dependability and appropriateness for the classification task, positioning it as a strong candidate for predictive modeling in this scenario.

## SVM

The SVM model with a linear kernel and 𝐶 = 1.0 was trained on the dataset and evaluated through various metrics, such as accuracy, precision, recall, and F1-score. The model achieved an accuracy of 0.9150, a precision of 0.9441, a recall of 0.9150, and an F1-score of 0.9284 on the test set, as shown in table 4. These results reflect strong performance, particularly due to high precision, indicating that the model is effective in reducing false positives.

**Table 3.4** Confusion Matrix for the SVM Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Test-Set (No CV)** | **3-Fold CV** | **5-Fold CV** | **10-Fold CV** |
| **Accuracy** | 0.9150 | 0.9475 ± 0.0030 | 0.9400 ± 0.0064 | 0.9437 ± 0.0170 |
| **Precision** | 0.9441 | 0.9460 ± 0.0051 | 0.9415 ± 0.0051 | 0.9469 ± 0.0190 |
| **Recall** | 0.9150 | 0.9475 ± 0.0030 | 0.9400 ± 0.0064 | 0.9437 ± 0.0170 |
| **F1 Score** | 0.9284 | 0.9467 ± 0.0040 | 0.9407 ± 0.0054 | 0.9431 ± 0.0175 |

To further evaluate the model's generalizability, cross-validation was conducted with 3-fold, 5-fold, and 10-fold splits. In the case of 3-fold cross-validation, the model reached an accuracy of 0.9475 ± 0.0030, a precision of 0.9460 ± 0.0051, a recall of 0.9475 ± 0.0030, and an F1-score of 0.9467 ± 0.0040. This suggests a slight improvement in performance on the test set, highlighting the model's robustness across diverse data splits.

During the 5-fold cross-validation, the model recorded an accuracy of 0.9400 ± 0.0064, a precision of 0.9415 ± 0.0051, a recall of 0.9400 ± 0.0064, and an F1-score of 0.9407 ± 0.0054 as highlighted in table 4. These findings indicate stable performance with slightly greater variability compared to the 3-fold CV, which is anticipated due to the increased number of splits.

Lastly, in the 10-fold cross-validation, the model obtained an accuracy of 0.9437 ± 0.0170, a precision of 0.9469 ± 0.0190, a recall of 0.9437 ± 0.0170, and an F1-score of 0.9431 ± 0.0175. The outcomes from 10-fold CV reveal consistent performance with somewhat higher variability owing to the smaller size of each fold, yet the overall metrics are still strong.

In conclusion, the SVM model shows outstanding performance across both the test set and the cross-validation folds. The high precision and recall indicate that the model effectively balances the minimization of false positives and false negatives, making it suitable for applications where this balance is vital. The consistency of the metrics across various cross-validation methods further underscores the model’s reliability and robustness.

## KNN

**Table 3.5** Confusion Matrix for the SVM Model Highlighting Performance Metrics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Test-Set (No CV)** | **3-Fold CV** | **5-Fold CV** | **10-Fold CV** |
| **Accuracy** | 0.9200 | 0.9550 ± 0.0170 | 0.9512 ± 0.0214 | 0.9525 ± 0.0278 |
| **Precision** | 0.8843 | 0.9541 ± 0.0164 | 0.9323 ± 0.0508 | 0.9299 ± 0.0595 |
| **Recall** | 0.9200 | 0.9550 ± 0.0170 | 0.9512 ± 0.0214 | 0.9525 ± 0.0278 |
| **F1** | 0.8946 | 0.9429 ± 0.0297 | 0.9390 ± 0.0350 | 0.9377 ± 0.0430 |

As it can be seen in table 5, The k-Nearest Neighbors (k-NN) algorithm, set with 𝑘 = 5, was assessed for its ability to predict stimulant misuse through the use of both cross-validation and metrics from the test set. The findings indicate that the model demonstrates strong accuracy across both the test dataset and the cross-validation splits, reflecting reliable performance in predicting instances of stimulant misuse.

For the test dataset, the model recorded an accuracy of 92.00%, precision of 88.43%, recall of 92.00%, and an F1 score of 89.46%. The elevated accuracy and recall suggest that the model is efficient in accurately identifying cases of both stimulant misuse and non-misuse. However, the relatively lower precision indicates a higher occurrence of false positives, which affects the F1 score.

Throughout the cross-validation process, the model exhibited stable performance across the different folds. The accuracy varied between 95.12% (5-fold CV) and 95.50% (3-fold CV), accompanied by low standard deviations, suggesting reliability across various data splits. Precision values displayed greater fluctuation, especially in the 5-fold and 10-fold CV, with a mean precision of 93.23% ± 5.08% and 92.99% ± 5.95%, respectively. This fluctuation may point to class imbalance or influence from specific data subsets on the predictions.

The average F1 score across the cross-validation splits ranged from 93.77% (10-fold CV) to 94.29% (3-fold CV), underscoring the model's ability to maintain a good equilibrium between precision and recall. Nevertheless, the standard deviation for the F1 score was higher for the 10-fold CV (±4.30%) in contrast to the 3-fold CV (±2.97%), indicating potential inconsistencies in predictions across smaller test segments.

In summary, the k-NN algorithm showcases impressive predictive capability for stimulant misuse, particularly regarding its accuracy and recall scores. However, the discrepancies seen in precision and F1 scores imply that further enhancements, such as fine-tuning the k value, tackling possible class imbalances, or engaging in feature engineering, could bolster the overall robustness of the model.

## MLP

**Table 3.6** Confusion Matrix for the MLP Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Test-Set (No CV)** | **3-Fold CV** | **5-Fold CV** | **10-Fold CV** |
| **Accuracy** | 0.9450 | 0.9540 ± 0.0013 | 0.9580 ± 0.0093 | 0.9580 ± 0.0125 |
| **Precision** | 0.9233 | 0.9454 ± 0.0045 | 0.9543 ± 0.0128 | 0.9528 ± 0.0212 |
| **Recall** | 0.9450 | 0.9540 ± 0.0013 | 0.9580 ± 0.0093 | 0.9580 ± 0.0125 |
| **F1** | 0.9307 | 0.9473 ± 0.0056 | 0.9514 ± 0.0139 | 0.9521 ± 0.0173 |

The Multi-Layer Perceptron (MLP) model was trained and assessed using a sample of 1,000 data points to forecast stimulant misuse. It features a single hidden layer containing 100 neurons and was trained for a maximum of 500 iterations. The findings indicate robust predictive performance across the test set and during cross-validation.

As seen in table 6, on the test set the model reached an accuracy of 94.50%, a precision of 92.33%, a recall of 94.50%, and an F1 score of 93.07%. These metrics suggest that the MLP model is proficient in predicting both categories, with high recall indicating that it effectively identifies cases of stimulant misuse and non-misuse. Yet, the slightly lower precision may point to possible false positives, warranting additional investigation to address this issue.

Throughout cross-validation, the model displayed reliable performance across various folds. The accuracy varied from 95.40% in 3-fold CV to 95.80% in 5-fold and 10-fold CV, accompanied by low standard deviations, reflecting stable predictions across the splits. The precision values were somewhat lower than the accuracy, with a peak mean precision of 95.43% ± 1.28% noted in 5-fold CV, indicating the model’s effectiveness in predicting both classes.

The mean F1 score across the cross-validation folds fluctuated between 94.73% (3-fold CV) and 95.21% (10-fold CV), underscoring the model's capability to balance precision and recall proficiently. However, the standard deviations for precision and F1 were greater in 10-fold CV, indicating some variability in predictions across smaller test subsets.

To summarize, the MLP model demonstrates outstanding performance in predicting stimulant misuse, achieving high accuracy and recall in all assessments. For further enhancement, efforts could target increasing precision by minimizing false positives through feature engineering or hyperparameter optimization. The model's strong generalization on a limited dataset emphasizes its robustness, positioning it as a valuable tool for stimulant misuse prediction initiatives.

## GCN

In addition to applying the previous six classical machine learning algorithms (mentioned in the Sections 3.2 - 3.7) that operate on feature vectors within the Euclidean space, this thesis also employed an advanced graph-based machine learning method. This approach leverages the relational structure of data to enhance the prediction of stimulant misuse, providing a complementary perspective to traditional ML methods.

A blue and black dots

Description automatically generated

**Figure 3.4** Visualization of the graph structure constructed using the KNNs for the NSDUH dataset.

A two-layer Graph Convolutional Network (GCN) was implemented with PyTorch and PyTorch Geometric 2.6.1 to carry out node classification on a weighted graph. The data is structured using a PyTorch Geometric Data object, which encompasses node features (X\_tensor), node labels (y\_tensor), and edge weights (if they are present). This graph-based methodology takes advantage of the interconnections between nodes, facilitating the learning process on structured data.

The main architecture of the GCN model is made up of two convolutional layers. The initial layer is a GCNConv layer that receives the input dimension (input\_dim) and produces a hidden dimension of size 16 (hidden\_dim=16). The subsequent layer is another GCNConv layer that takes the hidden dimension as input and generates an output dimension equivalent to the number of distinct classes in the labels (output\_dim). After the first layer, a ReLU activation function is applied to introduce non-linearity, followed by a dropout layer (with a rate of 0.5) to help prevent overfitting. At the final layer, a log softmax activation function is employed to perform multi-class classification.

The model is configured with particular hyperparameters to strike a balance between performance and computational efficiency. The hidden layer dimension is fixed at 16, which offers adequate capacity while maintaining a manageable computational cost. The Adam optimizer is set with a learning rate of 0.01, a frequently utilized value for GCNs. A dropout rate of 0.5 is applied to alleviate overfitting, which is especially crucial for small datasets or when dealing with high-dimensional feature spaces. Finally, the model undergoes training for 100 epochs, providing enough iterations for convergence while avoiding prolonged training duration.

The Graph Convolutional Network (GCN) model employed an inductive learning strategy for splitting the training and test datasets. This approach entails dividing the data into separate training and test groups, ensuring that the test group consists of nodes that the model has not encountered during the training phase. Specifically, 80% of the nodes were designated for training, while the remaining 20% were set aside for testing, achieved through a random division with a fixed seed (random\_state=42) to maintain consistency. The partitioning was carried out using the train\_test\_split function from scikit-learn, and the node indices were transformed into PyTorch tensors to be compatible with the PyTorch Geometric framework.

Within the inductive learning paradigm, the GCN model is trained on a selection of nodes and subsequently assessed on completely novel nodes during the evaluation phase. This arrangement guarantees that the model's capacity to generalize to unfamiliar data is assessed, reflecting realistic scenarios where predictions must be made for unseen nodes or new data points.

The architecture of the model comprises two GCN layers, paired with a ReLU activation function and a dropout layer to mitigate overfitting. The forward pass integrates edge weights when present in the graph, offering additional contextual insights regarding the relationships among nodes. The training process employs the Adam optimizer with a learning rate set at 0.01, and the loss is determined using the cross-entropy loss function. Throughout 100 epochs, both the loss and accuracy metrics were monitored for the training and test datasets.

Assessing the GCN model showed consistent outcomes throughout the epochs, with stable training and testing accuracies indicating robust generalization capabilities. The inductive split method, alongside the model's solid architecture, enhances its proficiency in accurately predicting stimulant misuse for nodes that were excluded from the training graph. This inductive framework assures that the model's predictions remain dependable in real-world scenarios involving previously unseen data.

The assessment of the Graph Convolutional Network (GCN) model for predicting stimulant misuse indicates strong performance in both the training and testing datasets. Throughout more than 100 epochs, the model's training accuracy consistently hovered around 95.47%, suggesting that it effectively captured patterns from the training data without indications of underfitting. Likewise, the test accuracy leveled off at 96.10%, illustrating the model's excellent ability to generalize and its capability to make reliable predictions on new data.

The training loss started at around 0.1177 and progressively declined to 0.1136 by the last epoch, demonstrating the model's enhanced adjustment to the training data as the optimization process advanced. The test loss showed slight variations at first but settled at roughly 0.0987, indicating precise predictions and no signs of overfitting. The narrow margin between training and test accuracy, along with low loss values, underscores the GCN model's stability and successful convergence during the training phase.

The findings highlight the model's strong generalization ability, as shown by the minimal difference in accuracy between training and testing. The predictive strength of the GCN model is evidenced by its accuracy figures (training: 95.47%, test: 96.10% highlighted in figure 5) and low loss metrics (training: 0.1136, test: 0.0987). Moreover, the consistent performance plateau after a few epochs suggests that the model achieves optimal performance relatively early in the training process, presenting a chance for applying early stopping in future training sessions to conserve computational resources.

To further improve the model's performance, several suggestions can be taken into account. First, assessing feature importance may yield insights into whether the existing set of input features is ideal for predicting stimulant misuse or if including additional domain-specific features could enhance predictions. Second, even though the model exhibits no signs of overfitting, the integration of regularization strategies, such as dropout layers or L2 regularization, could ensure robustness when applied to more intricate or noisy datasets. Lastly, hyperparameter optimization, which may involve modifying learning rates, adding more GCN layers, or trying out various convolutional processes, might provide additional enhancements in predictive accuracy.

In summary, the GCN model showcases remarkable performance in predicting stimulant misuse, supported by high accuracy and low loss figures. Its consistency across both training and testing phases highlights its efficiency and potential for practical application. Future endeavors should prioritize refining feature selection and exploring advanced configurations to further maximize its predictive capabilities.

A graph with numbers and lines

Description automatically generated

**Figure 3.5** Train and Test Accuracy Plot for the GCN Model.

The model includes a model explainer, specifically GNNExplainer, to clarify the predictions made by a trained Graph Convolutional Network (GCN). This interpretation centers on examining the significance of features and edges for a particular node within the graph. To successfully incorporate the explainer, the GCN implementation has been modified to explicitly distinguish between node features (data.x) and edge indices (data.edge\_index), which are essential inputs for the Explainer class.

The explanation process starts with initializing the Explainer. The Explainer is set up with the trained GCN model and employs the GNNExplainer algorithm. The configuration designates the mode as multiclass\_classification, suitable for tasks that involve categorizing nodes into several classes. The task level is defined as node, guiding the explainer to assess the significance of features and edges on a node-by-node basis. Furthermore, the output type of the model is established as log\_probs, a typical choice for classification tasks.

To create explanations, the explainer is utilized on node\_index = 10 in the graph. The generated explanations comprise an edge\_mask, which emphasizes the significance of edges, and a node\_mask, which pinpoint the most relevant features that influence the model's prediction for the designated node. This approach allows for a comprehensive and interpretable insight into the model's decision-making process at the node.

## Feature Importance

The feature importance visualization derived from the code emphasizes the top 20 features shown in figure 6 and table 7 that significantly impact the model's prediction for the designated node (node\_index = 10). This bar chart organizes these features according to their importance scores, determined by the GNNExplainer, with higher scores indicating a greater effect on the model's prediction for the node.

The feature with the highest importance is "trqanylif\_2.0", which has an importance score of 50.252, suggesting it has the most substantial effect on the prediction for node 10. The next most critical feature is "trqrsgdfx\_91.0", scoring 48.386, followed by "alcdays\_nan" at 43.502 and "trqrsrelx\_91.0" at 42.964. The importance scores vary from roughly 32.503 for the least significant feature in the top 20 ("pnranylif\_2.0") to 50.252 for the most significant feature ("trqanylif\_2.0"). This variation indicates that while all selected features are notably impactful, some have a considerably greater effect on the model's predictions.

The identified features comprise both binary attributes (e.g., "trqanylif\_2.0") and continuous variables (e.g., "alcdays\_nan"), representing a blend of behavioral, demographic, and possibly survey-related information. Numerous features bear suffixes like \_nan, likely signifying missing-value indicators, which implies that the absence of data could also carry predictive significance within the dataset. For instance, "alcdays\_nan" may pertain to patterns of alcohol usage or absent data regarding alcohol consumption, while "pnrnayrec\_91.0" and similar features may denote survey inquiries or documented behaviors over a certain period.

The existence of features ending in \_nan signifies that missing data may play a crucial role in predictions, potentially serving as proxies for other identifiable patterns or behaviors within the dataset. Behavioral features such as "trqanylif\_2.0" and "pnrnayrec\_91.0" could be linked to stimulant misuse or related risk factors, rendering them especially important for the model's predictions.

These recognized features are essential for interpreting the model's prediction for node 10. Features like "trqanylif\_2.0" and "trqrsgdfx\_91.0" likely encompass vital behavioral or survey-based indicators, while the presence of \_nan features highlights the need to understand and manage patterns of missing data. These insights are beneficial for elucidating and validating the model's predictions, particularly in contexts like predicting stimulant misuse, where behavioral, demographic, and survey features significantly contribute to evaluating individual risk levels.

**A graph with blue and white stripes

Description automatically generated**

**Figure 3.6** GCN Explainer showcasing the top 20 most important features.

**Table 3.7** List of The Top 20 Most Important Features and Their Descriptions

|  |  |  |
| --- | --- | --- |
| **Feature** | **Description** | **Importance Score** |
| trqanylif | Ever used any prescription tranquilizer | 50.252 |
| alcdays | Main reason for using tranquilizers | 48.386 |
| alcdays | The number of days alcohol consumed in the past 30 days. | 43.502 |
| trqrsrelx | Tranquilizers were used to relax or relieve tension. | 42.964 |
| pnranyrec | When for most recent pain reliever used. | 41.635 |
| AL30EST | The number of days had one or more drinks. | 41.075 |
| pnrnrec | Recent nonmedical use of pain relievers. | 40.161 |
| inhalrec | Whether inhalants were recently used. | 39.984 |
| herever | If heroin has ever been used. | |  | | --- | |  |   38.929 |
| cigarrec | Time since last smoked Cigar. | 38.778 |
| mjyrsmoke | Did you smoke marijuana within last year. | 38.078 |
| crkever | Ever used crack. | 35.674 |
| mjrec | When last smoked marijuana. | 35.561 |
| mrbstway | Most recent method marijuana was used. | 34.655 |
| ecstmolly | Ever used MDMA/ecstasy | 34.573 |
| peyote | Ever used peyote (is a hallucinogen) | 34.302 |
| PNRNM30D | Pain reliever nonmedical use in the past 30 days | 34.130 |
| NRNM30AL | Nonmedical alcohol use in the past 30 days. | 33.814 |
| cocever | Indicates if cocaine was ever used. | 33.197 |
| pnranylif | Indicates lifetime use of pain relievers. | |  | | --- | |  |  |  | | --- | | 32.503 | |

The subgraph illustration found in the accompanying PDF which is shown below as figure 7 has been produced by the GNNExplainer applied to a particular node in the graph, referred to as node\_index = 10. This illustration emphasizes a portion of the graph that significantly relates to the model's prediction for the chosen node. The subgraph is centered on the given node, and the associated nodes and edges represent the critical elements that affect the prediction.

The focal point of interest, node\_index = 10, acts as the center of explanation. The visualization includes nodes and edges that have the greatest influence on the model’s prediction for this specific node. By using the edge\_mask\_type='object' option in the explainer setup, the significance of edges is assessed, ensuring that only the most pertinent edges are featured in the subgraph. These edges are selected based on their calculated importance scores, with those scoring higher being included in the visualization.

The subgraph simplifies the overall complexity of the full graph, making it easier to understand the relationships deemed essential by the model for its decision-making process. Nodes and edges in the subgraph are trimmed down to concentrate solely on the most influential ones, thereby offering a clearer perspective on the reasoning behind the model's conclusions. If the edges or nodes in the subgraph have distinctive visual characteristics, such as variations in color intensity or thickness, these differences indicate their relative significance according to the model's interpretation.

The visualization of subgraphs produced by the GNNExplainer proves to be advantageous in situations like assessing the likelihood of stimulant medication misuse for a particular individual. By centering the analysis around a designated node, such as one that represents a person in a dataset, the subgraph emphasizes the relationships and features that significantly impact the model's prediction. This method facilitates an understandable, individualized examination, illustrating how the connections between the individual and others, alongside the individual's characteristics, play a role in the model's decision-making.

In the scope of forecasting stimulant medication misuse, the subgraph can unveil essential factors such as social connections, demographic characteristics, or behavioral trends that affect risk evaluation for a specific individual. By depicting these connections, the subgraph provides actionable insights into the most crucial elements influencing the prediction, allowing for targeted interventions or further investigation of the underlying risk factors. The ability to concentrate on specific nodes enhances the visualization's effectiveness as a potent instrument for applications that demand personalized and comprehensible predictions, such as public health initiatives or individual risk evaluations.

In the subgraph visualization, the edges connecting node 10 to other nodes, like node 249, are emphasized because the GNNExplainer has identified these edges as important for the model's prediction regarding node 10. The GNNExplainer analyzes how each edge contributes to the model's decision by giving importance scores to the edges. Edges that are highlighted carry the highest scores, signifying their strong impact on the prediction.

The significance of an edge can stem from various reasons. One reason is the strong influence of features, where the edge denotes a relationship between nodes that possess highly relevant attributes. For example, in a social or behavioral network, an edge linking node 10 to node 249 could represent shared traits or interactions that are especially indicative of the target outcome, such as the misuse of stimulant medication. Another reason is the structural importance of the edge; it could be vital in the graph’s architecture, such as connecting node 10 to a group of related nodes or bridging key subcommunities, making it essential for the flow of information throughout the graph.

Edge weight may also influence its importance. If the graph utilizes edge weights, a greater weight between nodes (for example, between node 10 and node 249) could indicate a stronger bond or interaction, thus enhancing the edge's significance to the model. Moreover, the model's sensitivity to particular edges during the message-passing phase in the GNN can increase certain connections' importance. Highlighted edges illustrate that the information conveyed through these connections plays a crucial role in shaping node 10's final embedding and, in turn, its prediction.

By marking these vital edges, the visualization offers a glimpse into how the model utilizes the structure of the graph to inform its predictions. In the scenario of predicting stimulant medication misuse, the edge between node 10 and node 249 might signify a relevant relationship, such as shared demographic, social, or behavioral traits that are essential for evaluating the individual's risk. These highlighted links not only clarify the model’s decision-making process but also illuminate the factors that influence its predictions.

Overall, the subgraph aids in clarifying how the model reaches its prediction for the specified node. It highlights essential relationships within the graph's structure, demonstrating the model's focus on particular connections and attributes. By examining the subgraph, one can derive insights into the decision-making mechanism of the Graph Neural Network (GNN), as it uncovers which aspects of the graph are most pertinent to the task and their contributions to the prediction. This illustration proves to be a valuable resource for comprehending and validating the model's actions in an organized and interpretable format.

**A drawing of a diagram

Description automatically generated**

**Figure 3.7** Subgraph visualization highlights the important nodes and edges.

**CHAPTER 4**

# CONCLUSIONS

This thesis investigates the application of Graph Convolutional Networks (GCNs) to predict the misuse of stimulant prescription medications using data from the National Survey on Drug Use and Health (NSDUH) 2022. The primary contribution of this research lies in demonstrating how GCNs can effectively model complex relationships within survey data by conceptualizing individuals as nodes and their connections as edges, offering a novel perspective for understanding patterns of medication misuse.

**Table 4.1** Comparison of the GCN accuracy Versus other Benchmarks

|  |  |  |  |
| --- | --- | --- | --- |
| **Rank** | **Model** | **Test-Set Accuracy** | **Training Accuracy** |
| 1 | GCN | 0.9610 | 0.9547 |
| 2 | Logistic Regression | 0.9587 | 0.9604 |
| 3 | Random Forest | 0.9571 | 0.9592 |
| 4 | MLP | 0.9450 | 0.9567 |
| 5 | KNN | 0.9200 | 0.9529 |
| 6 | SVM | 0.9150 | 0.9437 |
| 7 | Gaussian NB | 0.9080 | 0.9079 |

The thorough assessment of the models and their outcomes, as presented in the table and thesis document, uncovers several key insights related to predicting stimulant medication misuse. Per table 8, the Graph Convolutional Network (GCN) achieved the highest performance, with a test accuracy of 96.10% and a training accuracy of 95.47%, illustrating its enhanced capability to understand complex relationships and interdependencies in graph-structured data. This aligns with the thesis’ objective to utilize GCN in identifying intricate patterns that standard machine learning models might miss. The GCN's outstanding performance highlights its appropriateness for detailed tasks like predicting stimulant misuse, where relational and contextual factors are crucial.

Conventional models such as Random Forest and Logistic Regression also demonstrated impressive outcomes, with test accuracies of 95.71% and 95.87%, respectively. Their uniform cross-validation results underscore their dependability and ability to generalize across various data subsets. Moreover, the interpretability offered by Random Forest's feature importance analysis aligns with the thesis’ aim of identifying significant predictors of stimulant misuse.

More basic algorithms like Gaussian Naive Bayes (test accuracy: 90.80%) and k-Nearest Neighbors (test accuracy: 92.00%) showcased satisfactory performance, although they were less competitive compared to more advanced models. These simpler models are still beneficial in situations where computational efficiency or straightforward interpretability is prioritized over optimal accuracy. In contrast, the Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP) exhibited considerable performance variability. The SVM performed particularly well in certain cross-validation scenarios, demonstrating its sensitivity to the data distribution. The MLP recorded a test accuracy of 94.50%, showing minor precision concerns, indicating a need for improvements to reduce false positives.

The study's emphasis on graph-structured data and feature importance analysis highlights the importance of comprehending behavioral and demographic factors. Significant features, such as "trqanylif" (prescription tranquilizer usage) and "trqrsgdfx" (primary reason for utilizing tranquilizers), emerged as vital predictors, aligning with the specialized insights examined in the thesis. These results emphasize the practical implications of machine learning in public health, particularly the promise of graph-based approaches like GCN to deliver actionable insights into individual risk factors and their contextual relationships, thereby facilitating more targeted interventions.

In summary, this study underscores the importance of choosing models that are suited to the data's structure and the complexity of the problem at hand. While traditional models show reliability and effectiveness, the GCN's higher accuracy and capability to model relationships make it particularly beneficial for intricate, relational tasks such as predicting stimulant medication misuse. Future research could focus on enhancing feature engineering, investigating hybrid modeling techniques, and incorporating additional contextual data to improve predictions and increase model interpretability.

## Potential Future Works

Despite the strengths of this research, there remain areas for future exploration:

* **Improve Feature Selection:** Using Feature Importance by GNN Explainer or Random Forest, you can identify highly correlated (redundant) or irrelevant features, reducing dimensionality and noise, which improves model efficiency and generalization. Feature importance highlights the most predictive features, allowing you to focus on key contributors for better performance and enhanced interpretability.
* **Medication Categories**: Utilize the GCN model to predict misuse across various categories of medications, such as painkillers, sedatives, and tranquilizers, by tailoring the input features and graph structure to capture category-specific patterns.
* **Target-Specific Focus:** Apply the GCN model to focus on specific medications, such as Adderall, by refining the graph structure and features to include factors particularly relevant to the misuse of the targeted drug.
* **Alternative Graph Construction:** Investigating alternative graph construction methods, such as cosine similarity or to further refine graph representation. Cosine similarity serves as a substitute for KNN. It can be utilized to establish edges or weights according to the similarity of nodes regarding their feature directionality for constructing our GCN model architecture.
* **Regularization:** While the model does not currently show signs of overfitting, implementing regularization techniques, such as dropout layers or L2 regularization, could ensure robustness, especially when applied to complex or noisy datasets.
* **Hyperparameter Tuning:** Conducting hyperparameter optimization, including adjusting learning rates, adding more GCN layers, or experimenting with different convolutional processes, could further enhance the model's predictive accuracy.

# APPENDIX A

# PARAMETER SENSITIVITY ANALYSIS RESULTS

**Table A.1** List of The Attributes Used in The Thesis

|  |  |
| --- | --- |
| **Variable Name** | **Summary Description** |
| PDEN10 | Population density classification from the 2010 Census. |
| COUTYP4 | Metro/nonmetro county classification. |
| MAIIN102 | Indicator for majority American Indian areas. |
| AGE3 | Recoded age categories. |
| speakengl | English language proficiency. |
| irsex | Gender of the respondent. |
| irmarit | Marital status of the respondent. |
| IREDUHIGHST2 | Highest education level attained. |
| NEWRACE2 | Recoded race and ethnicity. |
| HEALTH2 | Self-reported health status. |
| IRHHSIZ2 | Household size. |
| IRKI17\_2 | Number of children under 17 in the household. |
| IRHH65\_2 | Number of adults aged 65 or older in the household. |
| irfamsoc | Indicator for receiving Supplemental Security Income. |
| irfamssi | Family participation in Social Security programs. |
| irfstamp | Indicator for receiving food stamps. |
| irfampmt | Family income payment methods. |
| irfamsvc | Use of family services such as counseling or support. |
| IRPINC3 | Recoded personal income level. |
| IRFAMIN3 | Recoded family income level. |
| govtprog | Participation in government assistance programs. |
| POVERTY3 | Poverty level of the respondent's household. |
| cigever | Ever used cigarettes. |
| irpmnicdep | Nicotine dependence level. |
| MEDMJPA2 | State medical marijuana policy indicator. |
| MOVSINPYR2 | Movement between states in the past year. |
| sexident | Sexual identity of the respondent. |
| service | Service in military or related groups. |
| eduschlgo | Currently enrolled in school. |
| miltfamly | Military family member status. |
| WRK35WKUS | Worked 35 weeks or more in the past year. |
| wrkdrgpol | Workplace drug policy awareness. |
| wrkdrgedu | Workplace drug education participation. |
| wrkdrghlp | Workplace drug help resources availability. |
| wrktstalc | Workplace alcohol testing policy. |
| wrktstdrg | Workplace drug testing policy. |
| wrktstrdm | Workplace random drug testing. |
| irwrkstat | Employment status. |
| hltinalc | Health impact of alcohol use. |
| hltindrg | Health impact of drug use. |
| hltinmnt | Health impacts of mental conditions. |
| ANYHLTI2 | Any health impacts reported. |
| cellwrkng | Use of a cellphone while working. |
| booked | History of being booked or arrested. |
| mxmjpnlt | Maximum penalty for marijuana possession. |
| DRVINALCO2 | Driving under the influence of alcohol. |
| DRVINMARJ2 | Driving under the influence of marijuana. |
| drvindrg | Driving under the influence of other drugs. |
| drvindrotmj | Driving under influence of a combination of drugs. |
| drvinaldrg | Alcohol and drug-related driving violations. |
| parol | Current or past parole status. |
| prob | Current or past probation status. |
| cigmfu | Monthly frequency of cigarette use. |
| cigrec | Recent cigarette use. |
| CIG30USE | Cigarette use in the past 30 days. |
| CG30EST | Estimated cigarette consumption in 30 days. |
| CIG30AV | Average daily cigarette consumption. |
| CIGMENTH30 | Menthol cigarette use in the past 30 days. |
| CIGROLL30 | Roll-your-own cigarette use in the past 30 days. |
| cigdlymo | Daily smoking behavior in the past month. |
| cigage | Age at first cigarette use. |
| cigdlyfu | Frequency of daily cigarette use. |
| cigdlmfu | Duration of daily cigarette use. |
| CIG100LF | Lifetime use of 100 or more cigarettes. |
| nicvapever | Ever used nicotine vapes. |
| nicvapage | Age at first nicotine vape use. |
| nicvapyfu | Frequency of nicotine vape use. |
| nicvapmfu | Monthly frequency of nicotine vaping. |
| nicvaprec | Recent nicotine vaping behavior. |
| NICVAP30N | Nicotine vape use in the past 30 days. |
| smklssevr | Lifetime use of smokeless tobacco. |
| smklsstry | Ever tried smokeless tobacco. |
| smklssyfu | Yearly frequency of smokeless tobacco use. |
| smklssmfu | Monthly frequency of smokeless tobacco use. |
| smklssrec | Recent smokeless tobacco use. |
| SMKLSS30N | Smokeless tobacco use in the past 30 days. |
| SMKLSS30E | Estimated smokeless tobacco consumption in 30 days. |
| cigarevr | Lifetime use of cigars. |
| cigartry | Ever tried cigars. |
| cigaryfu | Yearly frequency of cigar use. |
| cigarmfu | Monthly frequency of cigar use. |
| cigarrec | Recent cigar use. |
| CGR30USE | Cigar use in the past 30 days. |
| CI30EST | Estimated cigar consumption in 30 days. |
| pipever | Lifetime use of pipes for smoking. |
| PIPE30DY | Pipe use in the past 30 days. |
| cigcragp | Cigarette craving frequency. |
| cigregdy | Regular daily cigarette use. |
| cigwake | Smoking within 30 minutes of waking up. |
| ndssdepnd | Nicotine dependence score. |
| ftnddepnd | Nicotine dependence level based on FTND score. |
| pmnicdep | Past month nicotine dependence. |
| alcever | Lifetime alcohol use. |
| alcyfu | Yearly frequency of alcohol use. |
| alcrec | Recent alcohol use. |
| albstway | Method of alcohol consumption. |
| alcdays | Number of days alcohol consumed in the past 30 days. |
| AL30EST | Estimated alcohol consumption in 30 days. |
| ALCBNG30D | Binge drinking in the past 30 days. |
| cadrpeop | Social interactions with individuals who drink. |
| CADROTS2 | Routes used to obtain alcohol. |
| cabuyfre | Frequency of purchasing alcohol. |
| cabundag | Purchase of alcohol in bulk quantities. |
| cafrewho | Relationship to individuals providing alcohol. |
| CAFRESP2 | Frequency of sharing alcohol with others. |
| cadrkdrug | Use of alcohol with drugs. |
| cabingevr | Lifetime binge drinking. |
| cabingyfu | Yearly frequency of binge drinking. |
| cabingmfu | Monthly frequency of binge drinking. |
| uadpeop | Social interactions with underage drinkers. |
| uadpaid | Financial contribution to underage drinking. |
| cadrkmarj | Use of alcohol with marijuana. |
| cbdhmpevr | Lifetime use of CBD or hemp products. |
| cbdhmprec | Recent use of CBD or hemp products. |
| mjever | Lifetime marijuana use. |
| mjmfu | Monthly frequency of marijuana use. |
| mjrec | Recent marijuana use. |
| mrbstway | Methods of consuming marijuana. |
| MR30EST | Estimated marijuana use in 30 days. |
| mjyrsmoke | Yearly frequency of smoking marijuana. |
| mjyrvape | Yearly frequency of vaping marijuana. |
| mjyrdab | Yearly frequency of dabbing marijuana. |
| mjyreat | Yearly frequency of consuming marijuana edibles. |
| mjmonsmoke | Monthly frequency of smoking marijuana. |
| mjanymedyr | Yearly use of marijuana for medical purposes. |
| mkmbghtyr | Marijuana purchases in the past year. |
| MKMBGHT30N2 | Number of marijuana purchases in the past 30 days. |
| MKMAMTPDCOM2 | Amount spent on marijuana purchases. |
| mkmbghtdisp | Dispensary use for purchasing marijuana. |
| MKMLOOGMCOM2 | Marijuana purchases made online. |
| mkmsellmrj | Selling marijuana to others. |
| cocever | Lifetime cocaine use. |
| cocrec | Recent cocaine use. |
| ccbstway | Methods of consuming cocaine. |
| crkever | Lifetime crack cocaine use. |
| crakrec | Recent crack cocaine use. |
| crbstway | Methods of consuming crack cocaine. |
| herever | Lifetime heroin use. |
| herrec | Recent heroin use. |
| hrbstway | Methods of consuming heroin. |
| lsd | Lifetime use of LSD. |
| pcp | Lifetime use of PCP. |
| peyote | Lifetime use of peyote. |
| mesc | Lifetime use of mescaline. |
| psilcy | Lifetime use of psilocybin. |
| ecstmolly | Lifetime use of ecstasy or MDMA (Molly). |
| ketminesk | Lifetime use of ketamine. |
| dmtamtfxy | Lifetime use of DMT, AMT, or Foxy. |
| salviadiv | Lifetime use of salvia divinorum. |
| hallucoth | Lifetime use of other hallucinogens. |
| hallucevr | Lifetime use of hallucinogens. |
| hallucrec | Recent hallucinogen use. |
| lsdrec | Recent LSD use. |
| pcprec | Recent PCP use. |
| ecstmorec | Recent ecstasy or MDMA use. |
| ketminrec | Recent ketamine use. |
| damtfxrec | Recent DMT, AMT, or Foxy use. |
| salviarec | Recent salvia divinorum use. |
| inhalever | Lifetime use of inhalants. |
| inhalrec | Recent inhalant use. |
| inheaswy | Method of using inhalants. |
| methamevr | Lifetime use of methamphetamine. |
| methamyfu | Yearly frequency of methamphetamine use. |
| methamrec | Recent methamphetamine use. |
| metheaswy | Method of consuming methamphetamine. |
| oxcnanyyr | Yearly use of OxyContin. |
| pnranylif | Lifetime use of pain relievers for any reason. |
| pnrnmlif | Lifetime nonmedical use of pain relievers. |
| pnranyrec | Recent use of pain relievers. |
| oxcnnmyr | Yearly nonmedical use of OxyContin. |
| pnrnmrec | Recent nonmedical use of pain relievers. |
| PNRNM30D | Nonmedical pain reliever use in the past 30 days. |
| PNRNM30AL | Estimated nonmedical pain reliever use in 30 days. |
| pnrwynorx | Pain reliever use without a prescription. |
| pnrwygamt | Amount of nonmedical pain relievers used. |
| pnrwylngr | Duration of nonmedical pain reliever use. |
| pnrwyotwy | Other ways of obtaining pain relievers. |
| pnrrspain | Use of pain relievers to relieve physical pain. |
| pnrrsrelx | Use of pain relievers to relax. |
| pnrrsexpt | Use of pain relievers to experiment. |
| pnrrshigh | Use of pain relievers to get high. |
| pnrrsslep | Use of pain relievers to aid sleep. |
| pnrrsemot | Use of pain relievers for emotional relief. |
| pnrrsdgfx | Experiencing side effects from pain relievers. |
| pnrrshook | Feeling hooked or addicted to pain relievers. |
| pnrrssor | Use of pain relievers as a last resort. |
| trqanylif | Lifetime use of tranquilizers. |
| trqanyrec | Recent tranquilizer use. |
| trbzoanyyr | Yearly use of benzodiazepines. |
| trqnmlif | Lifetime nonmedical use of tranquilizers. |
| trqnmrec | Recent nonmedical use of tranquilizers. |
| TRQNM30D | Nonmedical tranquilizer use in the past 30 days. |
| trqwynorx | Tranquilizer use without a prescription. |
| trqwygamt | Amount of nonmedical tranquilizers used. |
| trqwyoftn | Frequency of nonmedical tranquilizer use. |
| trqrsrelx | Use of tranquilizers to relax. |
| trqrsexpt | Use of tranquilizers to experiment. |
| trqrshigh | Use of tranquilizers to get high. |
| trqrsslep | Use of tranquilizers to aid sleep. |
| trqrsemot | Use of tranquilizers for emotional relief. |
| trqrsdgfx | Side effects experienced from tranquilizers. |
| trqrshook | Feeling addicted to tranquilizers. |
| trqrssor | Use of tranquilizers as a last resort. |
| trbzonmyr | Yearly nonmedical use of benzodiazepines. |

Source: [2022 NSDUH Public Use File Codebook](https://www.samhsa.gov/data/system/files/media-puf-file/NSDUH-2022-DS0001-info-codebook.pdf)

# REFERENCES

1. National Institute of Mental Health. (n.d.). *Attention-deficit/hyperactivity disorder (ADHD)*. U.S. Department of Health and Human Services. <https://www.nimh.nih.gov/health/topics/attention-deficit-hyperactivity-disorder-adhd>
2. Kipf, T. N., & Welling, M. (2021). Semi-supervised classification with graph convolutional networks. *arXiv*. <https://arxiv.org/abs/1609.02907>
3. U.S. Drug Enforcement Administration. (n.d.). *Prescription for disaster*. U.S. Department of Justice. <https://www.dea.gov/sites/default/files/resource-center/Publications/DEA_Prescription-For-Disaster_508ver.pdf>
4. Teter, C. J., McCabe, S. E., LaGrange, K., Cranford, J. A., & Boyd, C. J. (2006). Illicit use of specific prescription stimulants among college students: Prevalence, motives, and routes of administration. *Pharmacotherapy, 26*(10), 1501-1510. https://doi.org/10.1592/phco.26.10.1501
5. Drug Enforcement Administration [DEA], U.S. Department of Justice. (2015). *National drug threat assessment summary 2015*. U.S. Department of Justice. https://www.dea.gov/documents/2015/NDTA-summary
6. Weyandt, L. L., Oster, D. R., Marraccini, M. E., Gudmundsdottir, B. G., Munro, B. A., Rathkey, E. S., & McCallum, A. (2016). Prescription stimulant medication misuse: Where are we and where do we go from here? *Experimental and Clinical Psychopharmacology, 24*(5), 400-414. https://doi.org/10.1037/pha0000093
7. United Nations Office on Drugs and Crime. (2019). *World drug report 2019*. United Nations. https://wdr.unodc.org/wdr2019/
8. Breiman, L. & Kluwer Academic Publishers. (2001). Random forests (R. E. Schapire, Ed.; Vol. 45, pp. 5–32) [Journal-article].
9. Louppe, G. (2014). *Understanding Random Forests: From Theory to Practice*. arXiv. <https://arxiv.org/abs/1407.7502>
10. Agarwal, A., Kenney, A. M., Tan, Y. S., Tang, T. M., & Yu, B. (2023). MDI+: A Flexible Random Forest-Based Feature Importance Framework. arXiv. <https://arxiv.org/abs/2307.01932>
11. Webb, G. I. (2010). Naïve Bayes. In Encyclopedia of Machine Learning (pp. 713–714). Springer. <https://doi.org/10.1007/978-0-387-30164-8_576>
12. Jurafsky, D., & Martin, J. H. (2021). Speech and Language Processing (3rd ed.). Draft chapters available at <https://web.stanford.edu/~jurafsky/slp3/>
13. Google Developers. (2024, October 15). *Logistic regression: Calculating probability with the sigmoid function*. <https://developers.google.com/machine-learning/crash-course/logistic-regression/sigmoid-function>
14. Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression (3rd ed.). Wiley. <https://doi.org/10.1002/9781118548387>
15. Owen, A. B., & Roediger, P. A. (2014). *The sign of the logistic regression coefficient*. arXiv. <https://arxiv.org/abs/1402.0845>
16. Scikit-learn. (n.d.). *Support vector machines*. Retrieved from <https://scikit-learn.org/stable/modules/svm.html>
17. IBM. (2023, December 27). What Is Support Vector Machine? <https://www.ibm.com/topics/support-vector-machine>
18. Shamsi, M., & Beheshti, S. (2023). *Separability and Scatteredness (S&S) Ratio-Based Efficient SVM Regularization Parameter, Kernel, and Kernel Parameter Selection*. arXiv. <https://arxiv.org/abs/2305.10219>
19. Brownlee, J. (2020, August 15). *K-Nearest Neighbors for machine learning*. Machine Learning Mastery. Retrieved from <https://machinelearningmastery.com/k-nearest-neighbors-for-machine-learning/>
20. GeeksforGeeks. (2023, July 20). *K-Nearest Neighbor (KNN) Algorithm*. <https://www.geeksforgeeks.org/k-nearest-neighbours/>
21. RoboticsBiz. (2022, September 28). *Pros and cons of the K-Nearest Neighbors (KNN) algorithm*. <https://roboticsbiz.com/pros-and-cons-of-the-k-nearest-neighbors-knn-algorithm/>
22. Scikit-learn: Machine Learning in Python. (n.d.). Neural network models (supervised). Retrieved <https://scikit-learn.org/stable/modules/neural_networks_supervised.html>
23. Cohn, T. (2017). *Lecture 7: Multilayer Perceptron. Backpropagation*. University of Melbourne. <https://trevorcohn.github.io/comp90051-2017/slides/07_backpropagation.pdf>
24. 33rd Square. (2023, November 15). *CNN vs RNN vs MLP: Analyzing 3 Key Neural Network Architectures in Deep Learning*. <https://www.33rdsquare.com/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/>
25. Fey, M., & Lenssen, J. E. (2019). *Fast Graph Representation Learning with PyTorch Geometric.* arXiv. Retrieved from <https://arxiv.org/abs/1903.02428>
26. Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). How powerful are graph neural networks? arXiv. <https://arxiv.org/abs/1903.03894>
27. Chen, J., Li, C., & Wang, X. (2023). Exploring the potential of graph neural networks in advanced applications. *arXiv*. <https://arxiv.org/pdf/2306.01958>
28. Substance Abuse and Mental Health Services Administration. (2022). National Survey on Drug Use and Health: 2022 codebook. Substance Abuse and Mental Health Services Administration. <https://www.samhsa.gov/data/>
29. PyTorch Geometric. (2023). *Creating Message Passing Networks*. <https://pytorch-geometric.readthedocs.io/en/latest/tutorial/create_gnn.html>
30. Kipf, T. N., & Welling, M. (2017). *Semi-Supervised Classification with Graph Convolutional Networks*. arXiv. <https://arxiv.org/abs/1609.02907>
31. Pope, P. E., Kolouri, S., Rostami, M., Martin, C. E., & Hoffmann, H. (2019). *Explainability Methods for Graph Convolutional Neural Networks*. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 10772–10781. <https://openaccess.thecvf.com/content_CVPR_2019/papers/Pope_Explainability_Methods_for_Graph_Convolutional_Neural_Networks_CVPR_2019_paper.pdf>
32. Ying, R., Bourgeois, D., You, J., Zitnik, M., & Leskovec, J. (2019). *GNNExplainer: Generating Explanations for Graph Neural Networks*. arXiv. <https://arxiv.org/abs/1903.03894>
33. Huang, Q., Yamada, M., Tian, Y., Singh, D., & Chang, Y. (2020). *GraphLIME: Local Interpretable Model Explanations for Graph Neural Networks*. arXiv. <https://arxiv.org/abs/2001.06216>