Feature Engineering and Missing Value Imputation for Classification of Depression with Deep Artificial Neural Networks

Caroline Sklaver
George Washington University, Department of Data Science
carolinesklaver@gmail.com

Acknowledgements:

I would like to thank and acknowledge
Amir Jafari as an advisor and mentor
throughout this project.

1. Introduction

With the dramatic increase in availability of healthcare data, clinical applications of artificial neural networks (ANNs) have become more prevalent [1]. ANNs can be used to predict diagnoses and therapeutic outcomes or to analyze images and waveforms. Along with other machine learning algorithms, ANNs have proven to be powerful tools in uncovering patterns and finding trends to enhance medical practices [2, 3]. Yet, this rapid increase in clinical data comes with challenges and limitations. One of the major challenges, is the class imbalance between positive and negative cases of diseases or other clinical outcomes. Additionally, this data can be replete with missing values. Despite these challenges, neural networks [1], support vector machines (SVM) [4–6], tree-based models [7, 8], and other machine learning algorithms have successfully predicted common diseases using clinical data. In the case of mental health diseases, machine learning methods are much less explored, and perhaps more complex due to the biological and environmental nature of these conditions. This report focuses on optimizing deep neural networks (DNNs) using data imputation techniques and feature engineering to classify depression using demographic, heath, and laboratory data from the National Health and Nutrition Examination Survey (NHANES). This report begins with background on depression, an explicit problem statement, and a literature review. Next, within the Methods section, the dataset, data imputation techniques, preliminary analyses, and feature engineering will be explained, followed by a detailed description of the DNNs. All results will be presented in table or figure format and thoroughly discussed. Lastly, this report will conclude with a summary of all findings and future applications.

1.1 Depression

According to the Centers for Disease Control and Prevention (CDC), approximately 1 in every 6 adults in the US will experience depression during their lifetime. This equates to about 16 million adults each year. Depression is characterized as a long-term affect in mood that can take form in a variety of symptoms such as feeling sad, loss of interest in activities, changes in appetite and sleeping patterns, increased fatigue, feeling worthless, difficulty concentrating, or thoughts of suicide [9]. Depression is caused by a complex combination of genetic, physiological, environmental, and social factors and is a leading cause of disability and a major contributor to the global burden of disease [10]. Depression is more common in women and is associated with physical comorbidities such as cardiovascular disease, stroke, cancer and other chronic conditions as well as mental health comorbidities such as anxiety disorder, and alcoholism [11]. Despite the widespread prevalence of depression, it frequently goes undiagnosed and untreated [12]. The NHANES conducted by the National Center for Health Statistics (NCHS) provides a depression-screening survey along with a vast amount of demographic, dietary, health-survey, examination, and laboratory features.

1.2 Problem statement

In this report, I explore missing value imputation methods, resampling techniques, feature engineering, and neural networks to predicting depression using NHANES data from 2005 to 2016. Ultimately, I will evaluate how statistical and machine learning imputation methods along with feature engineering techniques can improve classification results of both a deep ANN (DANN) and a multi-layer CNN. NHANES data is highly imbalanced with respect to depression (7.49% positive class) and contains many missing values. The primary goal of this paper is to identify the best imputation and feature engineering methods to optimize classification ability of these DNNs with the challenges of a highly imbalanced dataset and a target disease with imprecise causes.

1.3 Related work

Previous research has shown associations between depression and demographics such as age, race, gender, and income level [12–14]. Health characteristics including physical activity [15], metabolic syndrome, BMI [16], and other chronic conditions such as asthma, kidney disease [17], and osteoporosis [18] are correlated with depression rates. Additionally, dietary practices including caffeine consumption [19], household food insecurities [20], and Vitamin B levels [21] may be associated to depression. These associations were found mainly using linear regression analysis to identify single variable correlations with depression.

Jerez et. al 2010 [22] evaluated the performance of several statistical and machine learning imputation methods including mean/mode, multi-layer perceptron (MLP), self-organization maps (SOM), and k-nearest neighbors (KNN). The performance of each imputation method was evaluated using ANNs to predict cancer relapse using healthcare and demographic data of breast cancer patients. The results reveal improved prognosis accuracy using machine learning imputation, which significantly outperformed statistical imputation methods [22].

Heaton et. al. 2016 [23] reviewed feature engineering techniques for machine learning and found a DANN able to handle addition, summation and multiplication of features. Additionally, feature counts, differences, power transformations, and rational polynomial transformations were all synthesized by the DANN relatively easily [23].

ANNs have demonstrated accurate classifications of many diseases including heart disease [24], caries [25], and osteoporosis [26, 27]. Lopez-Martinez et. al. 2020 [2], presented a multi-layer neural network architecture with 3 hidden layers to identify hypertension in NHANES patients. This neural network achieved a lower error rate compared to that of Logistic Regression (LR), SVM, and Decision Tree (DT) algorithms. To handle the imbalanced data, synthetic minority oversampling techniques (SMOTE) was used for augmentation and produced the best results (AUC—0.77) [2]. Similarly, Yildirim et. al. 2017 [28], found that random sub-sampling to create a balanced training set improved MLP results in predicting chronic kidney disease. Meanwhile, Dutta et. al 2020 [29], utilized sub-sampling along with a multi-layer CNN to predict coronary heart disease with NHANES data.

According to Kiranyaz et. al 2019 [30], in a review of CNN applications, 1-dimenstional (1D) CNNs have recently achieved high performance in biomedical data classification and anomaly detection studies with the advantage of the simple and compact configuration [30]. However, within the literature, applications of 1D CNNs for binary classification are lacking.

2. Methods

Fig.1 outlines the condensed methodology of this report which includes data cleaning, data imputation methods, feature engineering, and implementation of DNNs. Additional methods absent from Fig.1 include the implementation and evaluation of commonly used machine learning algorithms, re-sampling techniques, and the verification of the proposed DNNs with a different target variable and a different dataset.

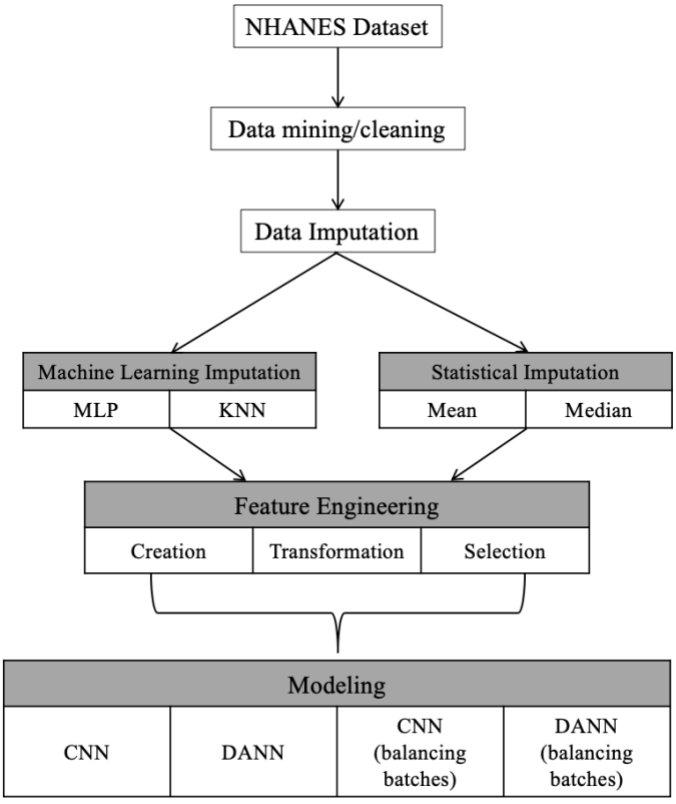

Fig. 1. Methodology flowchart

2.1 NHANES Data

The CDC's NCHS provides NHANES data every two years dating back to 1999. The NHANES is a stratified, multistage probability sample of the civilian, non-institutionalized US population amounting to about 5,000 participants each two-year period. This survey is unique in that it combines both interviews and physical examinations divided into 4 categories including demographic, dietary, laboratory, and examination data. For this project, I have included data from the 2005-2006 to the 2015-2016 surveys as they are the most heterogenous and complete with respect to the depression questionnaire. Wyatt et. al 2016 [31] created python scripts to download all NHANES data from the CDC website and convert it from XPT to CSV files, which were executed in this project and retrieved from GitHub.

Within the Questionnaire component lies the 'Mental Health – Depression Screener' comprised of the Patient Health Questionnaire nine-item (PHQ-9) depression screening instrument that assesses the frequency of depression symptoms over the past 2 weeks [32]. Response categories include "not at all," "several days," "more than half the days," and "nearly every day" given a point ranging from 0 to 3. A total score was calculated ranging from 0 to 27, with a score of 10 or higher marked as 'depressed'. This method using the PHQ-9 survey and a threshold value of 10 has been proven extensively in the literature and is commonly used in clinical studies to define depression [33]. Only data with complete PHQ-9 questionnaires were used in this study. Fig. 2 depicts the imbalance in the data with only 7.49% of the observations being positive for depression.

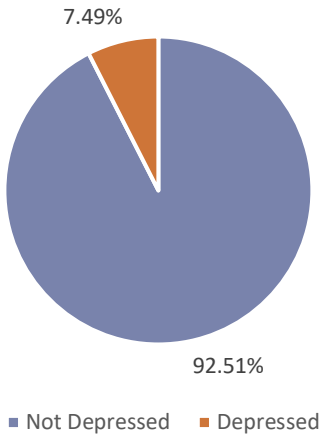

Fig. 2. Percentage of positive and negative depression cases in NHANES data

Originally, only 53 features were chosen for this project and were chosen based off associations found in the literature (Section 1.3). These original features were used for preliminary analyses described in Section 2.3. However, after preliminary models were unable to accurately predict depression, the number of features imported from NHANES was more than doubled to 112 features. Table 1 provides a breakdown these features by NHANES component and data type (binary, multi-class, or continuous). Most features are from the Questionnaire component, which makes this data unique when combined with demographics, dietary, and clinical data. Additionally, of note, there are 48 continuous features, which become central in the feature transformation portion of this project (Section 2.6).

Table 1: Number of Features by NHANES Component and Data Type

Survey Component	Number of Features	Binary	Multi-class	Continuous
Demographics	7	1	2	4
Dietary	16	1	0	15
Examination	7	0	1	6
Laboratory	15	2	0	13
Questionnaire	67	43	14	10
Total	112	47	17	48

The complete data is 31,357 observations with 112 columns including respondent sequence number, and survey year. Responses of '1 – yes' and '2 – no' were converted to binary 1/0, respectively. All responses of '7 – don't know' or '9 – refused' were marked as an N/A value. Prior to the implementation of any algorithm predicting depression, all multi-class features are encoded, and continuous features are standardized to a mean of 0 and standard deviation of 1.

2.2 Missing Value Imputation

2.2.1 Statistical methods

Missing data was imputed using two different methods: statistical and machine learning. The statistical methods include simple mean, median, and mode missing value imputation. Statistical imputations were implemented after dropping columns with a proportion of missing values exceeding an input threshold. Mean and median are simple imputations in which the missing values of continuous features are replaced by the mean or

median value of that feature column. With discrete variables, missing values were imputed using the most frequent value or mode of that feature column. Mean and median imputations with a 75% threshold were evaluated.

2.2.2 Machine Learning methods

The machine learning imputation methods aim to use the available information within the dataset to estimate and substitute missing values. In this way, the data was separated into complete data and incomplete data with missing values. The complete data (excluding the target feature 'depressed', ensuring the imputed values are independent of the final target) was used to predict values of the next-most-complete feature column. These predicted values were then imputed into the column and used with the complete data to predict the missing values of the subsequent column. Thus, progressively predicting missing values with the complete and imputed data. The machine learning models implemented in this section are MLP and k-nearest neighbors KNN.

MLP

The detailed explanation of an MLP in this section is paramount in understanding the function and implementation of the DNNs described in Section 2.5.

MLP is a supervised machine learning model that can estimate a nonlinear function using multiple layers of computational units connected in a feed-forward way [22, 34]. The input layer of an MLP is a set of nodes representing the input features. These input nodes pass through hidden layers, which transform the values from the previous layer using a weighted linear summation and a non-linear activation function. This is shown in Eq. (1) where w_i represents the weights, x_i the inputs, b the bias, $f(z)$ the activation function, and $\hat{y}$ the predicted output. Both classification and regression use backpropagation, a form of gradient-descent, to repeatedly adjust the weights to minimize loss.

$$z = \sum_{i=1}^{n} w_i x_i + b$$
$$\hat{y} = f(z) \tag{1}$$

Rectified linear unit function (ReLU) is the activation function implemented for the hidden layers in both MLP regression and classification. The ReLU function is shown in Eq. (2). For all positive values, the output is identity, while for all negative values, the output is 0.

$$\hat{y} = f(z) = \begin{cases} 0 \; if \; z < 0 \\ z \; if \; z \geq 0 \end{cases} \tag{2}$$

In this data imputation approach, if the data type of the target variable is discrete, the loss function is cross-entropy (logarithmic loss), given in Eq. (3), where y_i is the ground truth, $\hat{y}_i$ is the predicted value for each output class i in M. This is implemented with a learning rate of 0.001.

$$CE = -\sum_{i}^{M} y_i \ln(\hat{y}_i) \tag{3}$$

Meanwhile while in MLP regression, the loss function is mean squared error which is calculated by taking the average of the squared differences between the predicted ($\hat{y}_i$) and actual (y_i) values. Thus, as the difference between predicted and actual values increases, the square of that value results in a larger error.

Lastly, the output layer of an MLP transforms the values from the final hidden layer with the number of nodes given by the desired output, into output values [34]. In binary classification MLP uses the logistic function $g(\hat{y})$ in the output layer given by Equation 4. A threshold of 0.5 assigns outputs greater than or equal to 0.5 to positive class (1), and outputs less than 0.5 to the negative class (0) [35].

$$g(\hat{y}) = \frac{1}{(1 + e^{a})} \tag{4}$$

In multi-class classification, MLP uses the SoftMax function in the output layer shown in Eq. (5), where, again, M represents the number of output classes [34]. This function calculates the probabilities of each class outcome and

assigns the output to be the class with the greatest probability.

$$SoftMax_i(\vec{\hat{y}}) = \frac{e^{\hat{y}_i}}{\sum_k e^{\hat{y}_M}} \quad (5)$$

In MLP regression, the output activation function is just the identity function (the output remains as is). In this MLP progressive data imputation approach, the different error and output functions were implemented as described depending on the datatype of target variable (continuous, binary, or multi-class). A simple architecture consisting of two hidden layers was used in this process.

KNN

KNN is another supervised learning algorithm that predicts target values based on its resemblance to other values in the training set [36]. In comparison to MLP, KNN does not use the entire data, but rather uses the mean (in regression) or majority vote (in classification) of k-closest complete cases [22]. These k cases are found by minimizing a distance measure, Manhattan, Euclidian, or Minkowski distance. The KNN model used in this portion of the project uses $k = 5$ with uniform weights and Euclidian distance to progressively predict and impute missing values. Depending on the datatype of the target variable, the mean of k neighbors values (continuous target), or majority of k neighbors values (discrete target) were used to predict and impute missing values.

2.3 Preliminary Analysis

As mentioned in Section 2.1, the analyses performed in this section used the original data containing a much smaller sample of 53 features from NHANES. In these introductory attempts at classifying depression, seven supervised machine learning algorithms were applied: LR, Naïve Bayes (NB), DT, Random Forest (RF), Extreme Gradient Boosting (XGBoost), a single Perceptron (PPN) and a 2-layer MLP. Each algorithm will be briefly outlined in this section. All training data was standardized, encoded, and split into 33% testing data. Hyperparameters were not tuned and cross-validation did not occur in this process as it was a preliminary analysis to determine the next steps.

LR was applied as a baseline standard statistical tool, which is a linear algorithm with a non-linear output [37]. NB is an algorithm based on Bayes' theorem that assumes all features are independent of each other and each pair of features are independent of each other given the value of the class variable. NB uses maximum likelihood for parameter estimation and is a simple, fast model. While it has been used for disease prevention [37], the major limitation lies in the feature-independence assumption, thus it is sensitive to highly correlated features.

DT is an algorithm that predict the target value by using tree-like graphs based on sorting and learning decision rules from the data features. Each tree node represents the feature to be classified, while each branch is the value that node can assume. DTs can be used for both classification and regression analysis and are commonly used in disease prevention [33]. Next, RF is an algorithm based on ensemble learning. A RF classifier fits a number of DTs to a sample of the training data and uses majority vote to get predictions. RF uses averaging to improve the accuracy of prediction and to limit over-fitting. RF have proven to be successful in predicting Parkinson's Disease [37, 38]. Similarly, XGBoost is a boosting model that utilizes tree ensembles at its base. Boosting is a process of improving prediction accuracy by assigning weights to previous outcomes, beginning with a weak learner. XGBoost grows classification trees and attempts to minimize the misclassification rate in subsequent trees. This is a powerful method in machine learning.

Lastly, both a simple PPN and a 2-layer MLP (described in Section 2.2.2) were implemented during this stage of analysis. The PPN differs from the MLP in that it is comprised of just a single perceptron, opposed to multiple layers of perceptrons.

2.4 Resampling

The initial attempts at combatting the imbalance within the data involved three resampling techniques. More specifically, implementing random over sampling (ROS), random under sampling (RUS), and synthetic minority over sampling (SMOTE) to balance the training data of a 2-layer MLP. ROS entails randomly selecting

observations with replacement from the minority class to match the length of the majority class. Conversely, RUS is the random selection from the majority class to match the length of the minority class. Lastly, SMOTE is a resampling method that creates synthetic minority samples that are halfway between the existing instances and *k* other minority instances. Each of these resampling techniques were used to train a 2-layer MLP using the original 53 NHANES features with 5-fold cross validation and 20 epochs for each missing value imputation type.

2.5 Neural Networks

Given the outcomes of the preliminary algorithms and resampling techniques, the next step was to test the abilities of DNNs to classify depression. Neural networks can have many different architectures. Section 2.2.2 describes the process of a feed-forward neural network and the corresponding equations used in both regression and classification. In predicting depression, deeper, more complex architectures are implemented to enhance the likelihood of the model's ability to learn the data from such a highly imbalanced and potentially ambiguous data set. For each imputation type and feature engineering method (Section 2.7), the same DANN and CNN outlined in the following sections, with the same set of hyper-parameters was implemented.

2.5.1 DANN

Figure 3 represents the DANN architecture and Table 2 presents the parameters of this architecture.

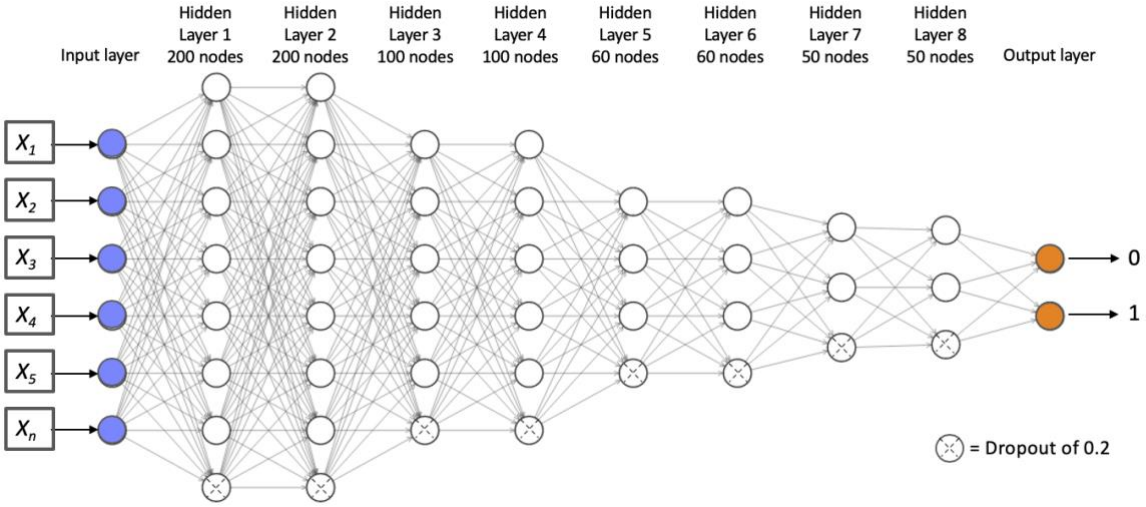

Fig. 3. DANN architecture [39]

Table 2: DNN Model Architecture and Parameters

Parameter	Value
Input dimension	# features in training set
Number of hidden layers	8
Hidden layer 1 dimension	200
Layer 1 activation function	ReLU
Dropout	0.2
Hidden layer 2 dimension	200
Layer 2 activation function	ReLU

Dropout	0.2
Hidden layer 3 dimension	100
Layer 3 activation function	ReLU
Dropout	0.2
Hidden layer 4 dimension	100
Layer 4 activation function	ReLU
Dropout	0.2
Hidden layer 5 dimension	60
Layer 5 activation function	ReLU
Dropout	0.2
Hidden layer 6 dimension	60
Layer 6 activation function	ReLU
Dropout	0.2
Hidden layer 7 dimension	50
Layer 7 activation function	ReLU
Dropout	0.2
Hidden layer 8 dimension	50
Layer 8 activation function	ReLU
Dropout	0.2
Number of output classes	2
Output function	SoftMax
Batch size	32
Number of training samples	21,009
Loss function	Binary Cross-Entropy
Optimizer	Root mean squared propagation
Learning rate	0.001
Evaluation metrics	F1-Score, Precision, Recall, AUC, Confusion Matrix

The transformation of the input features using weights and biases occurring in each hidden layer along with the ReLU activation function are explained in section 2.2.2 with Eq. (1) and Eq. (2). In this DANN, root mean squared propagation is the optimizer and implements a learning rate of 0.001. A smaller learning rate may result in higher accuracy; however, it also significantly increases learning time. The loss function is binary cross-entropy, which is a permutation of Eq. (3), where $M = 2$ and N is the total number of samples, given by Eq. (6).

$$BCE = -\frac{1}{N}\sum_{i=1}^{N} y_i \ln \hat{y}_i + (1 - y_i) \ln(1 - \hat{y}_i) \tag{6}$$

The number of input nodes is equal to the number of features in each permutation of the data. Similarly, two output nodes and the SoftMax output function (Eq. 5) were used to generate the model predictions. There are 8

hidden layers that, when viewed from input to output contain 200, 200, 100, 100, 60, 60, 50, and 50, nodes respectively. The activation function for each hidden layer is ReLU (Eq. 2) [21], making each hidden node a rectified linear unit [23]. Between each hidden layer there is dropout with a value of 0.2, meaning during the training phases of each layer, individual nodes (including their incoming and outgoing edges) are dropped out of the net with the probability of 0.2 to prevent overfitting. The goal in building this DANN architecture was to create enough depth and nodes to best learn the NHANES data. During training, the model underwent 25 epochs with 3-fold cross validation, and finally, the trained models were evaluated on the reserved testing set.

2.5.2 CNN

CNNs are a type of feed-forward DNN often used for Computer Vision applications of two-dimensional data such as image classification and object detection. 1D CNNs are typically applied to signal data (e.g. EKGs, audio recordings), time-series data, or natural language processing. Similar in concept to ANNs, CNNs consist of hidden layers that are connected to the input layer and contain biases and weights [40].The hidden layers of a CNN are convolutional, pooling, and fully connected feed-forward layers [41]. Convolutional layers perform feature extraction by sliding a kernel (similar to a typical neuron of an ANN) over input signals and applying cross-correlation to get the outputs. The dimensionality of this output can then be reduced in the pooling layer [42]. The output of the convolution or pooling layers is often passed through a flattening layer that reshapes the output to be fed into the final fully connected layers. These fully connected layers are analogous to the hidden layers of any MLP previously described. Lastly, the fully connected layers map the extracted features into the final classification outputs [43]. Figure 4 presents the CNN model architecture implemented in this project and Table 4 outlines the parameters of this architecture.

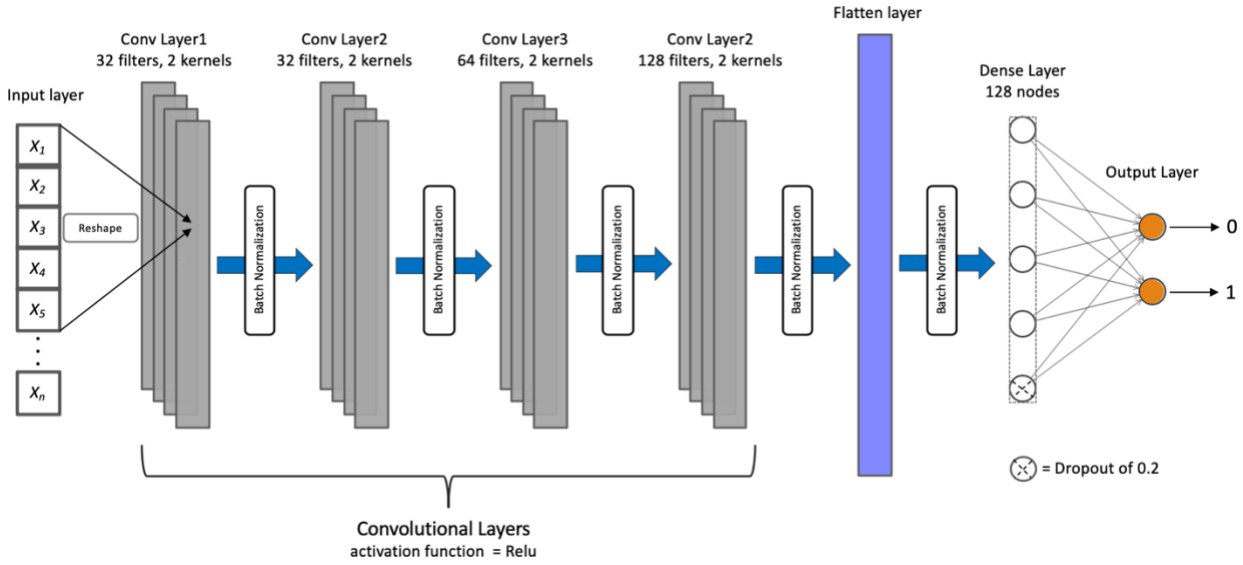

Fig. 4. CNN architecture [39]

Table 3: CNN Model Architecture and Parameters

Parameter	**Value**
Reshaped input	(# observations, # features, 1)
Number of hidden layers	5
Conv1D layer 1 filters	32
Conv1D layer 1 kernel size	2
Conv1D layer 1 activation function	ReLU

Conv1D layer 2 filters	32
Conv1D layer 2 kernel size	2
Conv1D layer 2 activation function	ReLU
Conv1D layer 3 filters	64
Conv1D layer 3 kernel size	2
Conv1D layer 3 activation function	ReLU
Conv1D layer 4 filters	128
Conv1D layer 4 kernel size	2
Conv1D layer 4 activation function	ReLU
Flatten shape	(# observations, # features)
Dense Hidden layer 5 dimensions	128
Hidden layer 5 activation function	ReLU
Dropout	0.2
Number of output classes	2
Output function	SoftMax
Batch size	32
Number of training samples	21,009
Loss function	Binary Cross-Entropy
Optimizer	Adam
Learning rate	0.001
Evaluation metrics	F1-Score, Precision, Recall, AUC, Confusion Matrix

This model first reshapes the input data into three dimensions then contains four 1D convolutional layers with 32, 32, 64, and 128 filters of size 2, respectively. Each of these layers use ReLU as the activation function (Eq. 2), which converts any negative values to 0. Next, the data is flattened to enter a fully connected dense layer with 128 nodes, ReLU activation, and a dropout of 0.2. Lastly, the output layer contains 2 nodes representing the binary output and uses SoftMax as the activation function (Eq. 5). Additionally, after ReLU activation, between each hidden layer, batch normalization is implemented to normalize the output tensors to a mean of 0 and standard deviation of 1.

2.6 Verification of Neural Networks on Different Data

To verify the architectures and implementations of the DANN and CNN described in section 2.5, these networks were tested on both a different target within the NHANES data, as well as the UCI breast cancer dataset [44]. The DANN underwent 25 epochs with 3-fold cross validation and the CNN underwent 10 epochs in these verification implementations. With the NHANES data, the target of '*overweight*' was chosen as it contains 33.79% positive observations. Though the predictive features were not chosen specifically to classify '*overweight*', this application will indicate whether the models are functioning, and/or if the data itself is inadequate for classifications in general. Progressive MLP-imputed data was used for the training and testing of these models with this target.

Secondly, the UCI breast cancer dataset has 37% positive breast cancer observations. The imbalance in this data not extreme, and the features have proven to be highly predictive using many machine learning algorithms [45–

47]. While there may be overfitting due to the depth and complexity of the DNNs, the performance of these networks on this data should be highly accurate, otherwise it is a clear indication that the networks are not implemented correctly.

2.7 Feature Engineering

In an effort to enhance the performance of the DNNs described above, various feature engineering techniques were implemented. In this paper, each feature engineer method is implemented as an experiment to improve classification abilities of the proposed DNNs (Section 2.5). These methods include feature creation, transformation, and selection. Feature creation refers to the addition of feature columns by either counting, summing, subtracting, dividing, or multiplying two or more existing features. Feature transformation refers to the application of mathematical functions such as square root, log, power, polynomials, and sine to the continuous numerical features (referred to as '*cont*') in the existing data. Lastly, feature selection refers to the sub-setting of features based on known prior associations with depression along with the elimination of highly correlated features.

This study begins with 50 raw features originally chosen from the NHANES (excluding depression, respondent sequence number, and year). Subsequently, 61 more features are added to this original data directly from the NHANES to create the complete baseline feature set (breakdown in Table 1). Next, 15 features were created through counts, summations, and quotients of two or more existing features. After feature creation, transformations of the continuous variables were also appended to the existing features. Once all NHANES and created features were included, feature selection (sub-setting) was also tested. Each method with the number of features used in training (after one-hot-encoding) are shown in Table 4.

Table 4: Feature Engineering Methods

Engineering Type	**Feature Engineering Procedure**	**Number of Features (after encoding)**
Baseline	NHANES minimal features	79
	NHANES baseline features	166
Creation	Baseline + Created features (BCF)	186
Transformation	BCF + log(cont)	240
	BCF + $(cont)_{1/2}$	240
	BCF + $(cont)_2$	240
	BCF + polynomial	240
	BCF + sine(cont)	240
Transformation and continual augmentation	BCF + $(cont)_2$ + log(cont)	293
	BCF + $(cont)_2$ + log(cont) + $(cont)_{1/2}$	346
	BCF + $(cont)_2$ + log(cont) + $(cont)_{1/2}$ + polynomial	399
	BCF + $(cont)_2$ + log(cont) + $(cont)_{1/2}$ + polynomial + sine(cont)	452
Selection	Subset of BCF	105
Selection and transformation	Subset + log(cont)	135
Selection, transformation, and continual augmentation	Subset + log(cont) + $(cont)_{1/2}$	165

2.8 Balancing Batches

For each data imputation method and feature engineering experiment, both the DANN and CNN models were trained with standard training data, which is highly imbalanced, and then with balanced mini batches. These balanced mini batches used RUS of the majority class without replacement to create a random, balanced batches of size 32 [48]. This technique was implemented in order to evaluate the DNN performances when trained on balanced data to determine whether resampling methods are beneficial in DNN classifications. [28, 49].

2.9 Evaluation methods

In each experiment, 33% of the data, 10,348 observations, were set aside and used to evaluate the trained models. In this evaluation of the proposed experiments, this paper focuses on the F1-score. F1-score is a weighted average of precision and recall. With imbalanced data, the typical accuracy score does not capture the model's ability to distinguish between positive and negative cases. Precision conveys the proportion of positive predictions that were correctly classified (the positive predictive value), while recall measures the proportion of the actual positive values that were correctly predicted (sensitivity). Thus, the F1-Score will provide information about both the model's false positive and false negative rates. The goal in this project is to maximize the F1-score.

3. Results

Preliminary model results using the original 50 NHANES features with each imputation technique are shown in Table 5. With such low F1-scores, and accuracies averaging at 92%, the challenge with imbalanced data is highlighted in these results. NB has the highest F1-scores, yet, they are still extremely low. Furthermore, no imputation type stands out as either a superior or inferior method.

Table 5: Preliminary Algorithm F1-Scores

	F1-Score							
Imputation Type	LR	DT	RF	XGBoost	KNN	NB	PPN	MLP
MLP	0.1575	0.0129	0.0129	0.1839	0.0946	0.2465	0.2822	0.1489
KNN	0.1430	0.0077	0.0077	0.1630	0.0943	0.2459	0.0303	0.1675
Mean	0.1398	0.0154	0.0154	0.1833	0.0953	0.2473	0.1171	0.2346
Median	0.1398	0.0154	0.0154	0.1833	0.0953	0.2473	0.1171	0.0832

It is clear from the results in Table 5 that training models on the original data from NHANES is not sufficient when aiming to accurately discriminate between positive and negative depression cases. Thus, to determine whether the high level of imbalance is the root cause of poor performances, resampling techniques were tested. F1-score results of each resampling technique using a simple 2-layer MLP are presented in Table 6. These F1-scores are accompanied by high recall scores and correspondingly low precision scores. This indicates that when balancing the training data, this 2-layer MLP predicts many more positive cases; however, these predictions are not based off learned discrimination of positive cases within the data, but rather, is a direct reflection of the proportions seen during training.

Table 6: 2-Layer MLP F1-Scores with Resampling and Imputation Methods

	Resampling Method F1-Score		
Imputation Type	ROS	RUS	SMOTE
MLP	0.14635	0.134151	0.137677
KNN	0.139518	0.140855	0.134487
Mean	0.146516	0.135999	0.139016
Median	0.14319	0.141022	0.137006

The results insofar led to the acquisition and augmentation of more features from the NHANES and the construction of DNNs to increase the complexity of the models in attempts to improve classification results. In verifying the architecture and implementation of these DNNs, the F1-scores of NHANES MLP imputed data with '*overweight*' as the target variable, and UCI Wisconsin Breast Cancer with target of '*diagnosis*' are provided in Table 7.

Table 7: Verification of DNNs with Different Target and Breast Cancer Data

			F1-Score	
Data	**Target Variable**	**% positive cases**	**DANN**	**CNN**
UCI Wisconsin Breast Cancer	*diagnosis*	37.26	0.8859	0.9385
NHANES MLP imputed	*overweight*	33.75	0.7143	0.7538

These model-verification results ultimately reveal that the DANN and CNN were implemented correctly with sufficient architectures. Interestingly, the CNN out-performs the DANN in both cases. With high F1-scores using the slightly imbalanced Breast Cancer data, the ability of these networks to make accurate classifications is confirmed. However, when testing the models on the NHANES data using a different target of '*overweight*', the F1-scores are not quite as impressive. This is likely due to the fact that the features in the NHANES dataset have much less predictive power than those in the Breast Cancer dataset. The NHANES data may, by nature, not be highly discriminatory or indicative for any imbalanced classification problem. Nevertheless, these DNNs can predict the target of '*overweight*' with much higher F1-scores than in the case of depression (Table 8). This disparity between the two targets' outcomes can be attributed to the disparities in class imbalances as well as the nature of the two variables, '*overweight*' having more recognizable qualities.

In all, these results, confirm the proper functioning of the proposed that the DNNs and that the high level of class imbalance with respect to depression is a significant hindrance. Lastly, these results indicate that the NHANES dataset and depression itself may be too abstruse for any accurate classifications.

In the final attempts to enhance performance of the DNNs in predicting depression, feature engineering methods were tested. In Table 8, the F1-scores of the 'baseline' models using the extended number of features imported from NHANES are shown for each imputation type. As shown in bold, the best baseline results are achieved with MLP imputation and training the CNN (0.3069). Machine learning imputation techniques (MLP and KNN) have better performance on this baseline data compared to the statistical imputation techniques (mean and median). Additionally, the DANN is unable to learn the data properly without any feature engineering.

Table 8: Baseline NHANES F1-Score Results

Model	**MLP**	**KNN**	**Mean**	**Median**

DANN	0.1881	0.0000	0.0000	0.1703
CNN	**0.3069**	0.2841	0.2712	0.2356
DANN (BB)	0.2736	0.2777	0.2618	0.2595
CNN (BB)	0.2698	0.2694	0.2647	0.2573

Taking a closer look at the separate DNN performances, Table 9 provides the average F1-score results of each DNN across all imputation and feature engineering methods. CNN has the highest average F1-score (0.2640), thus, averages the best performance across all experiments.

Table 9: Comparison of Average Neural Network Performances

Model	**Average F1 Score**
DANN	0.1458
CNN	**0.2640**
DANN BB	0.2513
CNN BB	0.2488

To compare imputation methods, Table 10 presents the average F1-score of each imputation type across all model and feature engineering experiments (Column 1). In this case, KNN provides the highest average F1-score. However, taking the average F1-score results from just the CNN testing results (model averaging the highest F1-scores as shown in Table 9), across all feature engineering methods, MLP imputation has the best F1-score (0.274), followed by KNN (Column 2).

Table 10: Average F1-Scores Across Imputation Methods

Imputation Method	**Average F1 Score (all models)**	**CNN Average F1 Score**
MLP	0.2312	**0.2794**
KNN	**0.2429**	0.2726
Mean	0.2083	0.2521
Median	0.2274	0.2521

Next, to examine the effects of the feature engineering techniques on each model type (DANN, CNN, and BB), Fig. 5 depicts the fluctuations in F1-scores as a result of feature engineering with the MLP imputed data. The DANN has the greatest fluctuations and is the least reliable model, however, it is improved by training on balanced batches. All of the models benefit from the addition of features from ‘original’ (50 NHANES features) to ‘baseline features’ (112 NHANES features). In general, feature transformations can make slight differences in each model’s performance, however, continual augmentation of these transformed features can have varying effects based on the mathematical equation and model being tested. Lastly, most of the models benefit from some feature selection and subsequent transformations (e.g. ‘Feature Selection’, ‘Selection + log’).

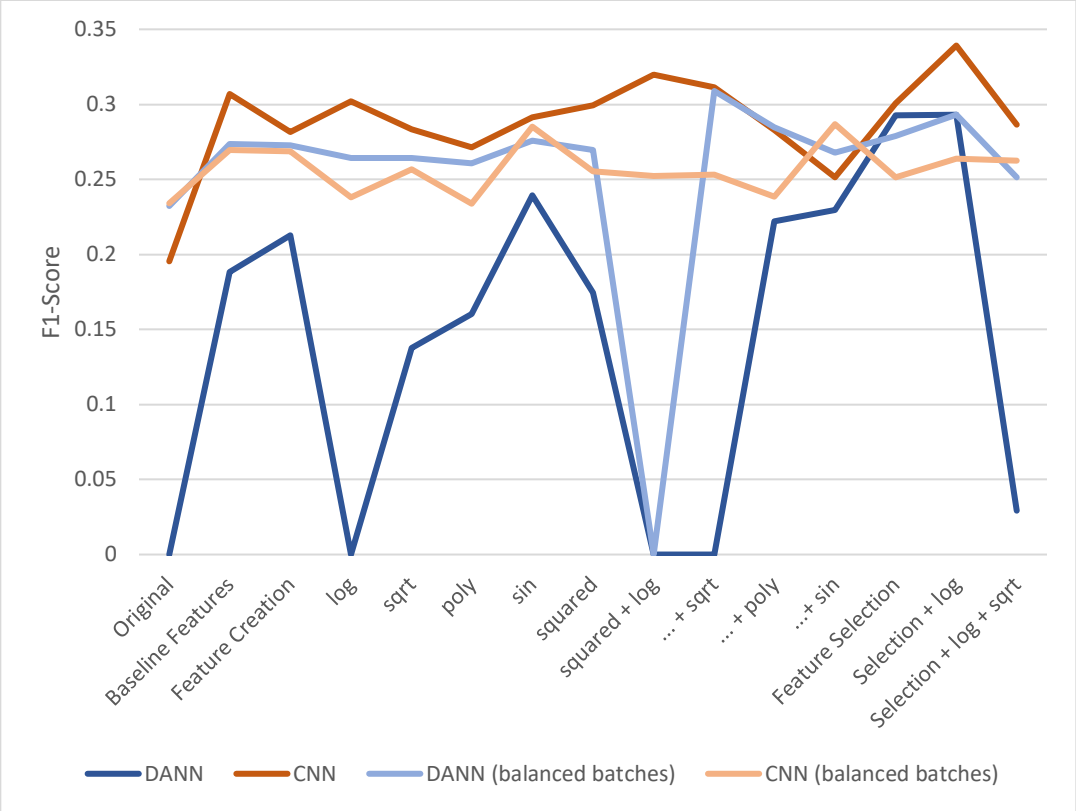

Fig. 4. Effects of feature engineering on each network with MLP imputated data

To take a closer look at the effects of feature engineering on the best model with the best imputation method (Tables 9 and 10), Fig. 5 depicts the changes in F1-scores after feature creation, transformations, and continuous augmentations with MLP imputed data and the CNN without balancing batches. In this case, feature creation results in a decrease in the F1-score. Meanwhile, log, sin, and squared transformations have better outcomes than polynomial and square root. The best transformation and augmentation combination includes both the square and log of continuous features resulting in a higher F1-score than the baseline (0.3198). However, subsequent augmentations negatively affect performance.

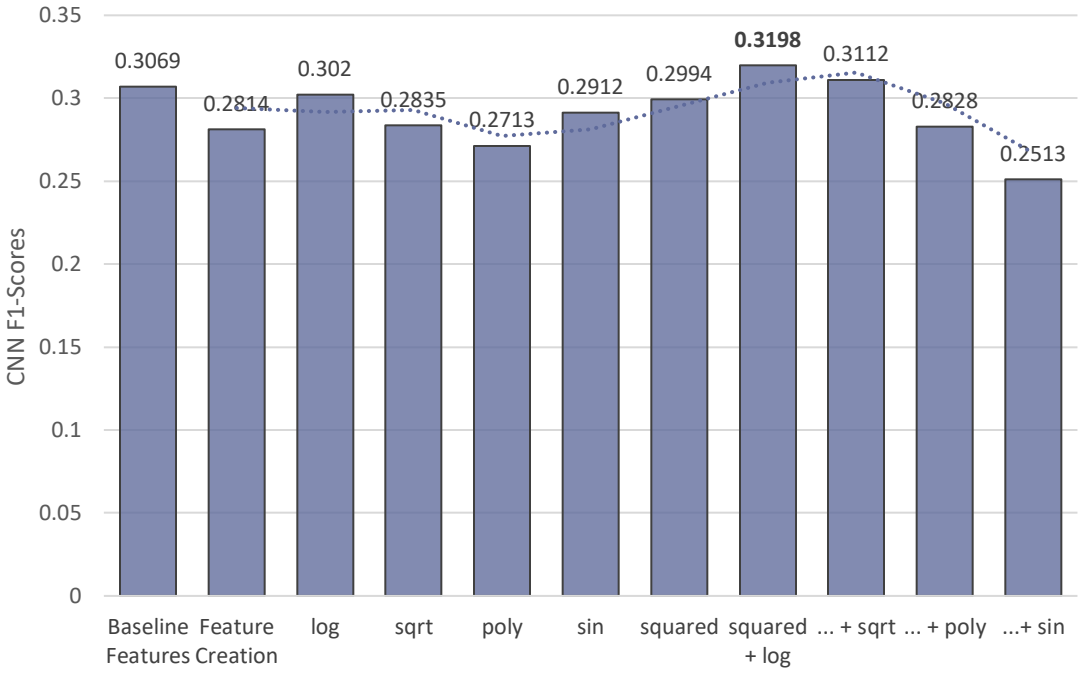

Fig. 5. Comparison of MLP imputed CNN F1-scores with feature creation, transformation, and augmentation methods

Lastly, Fig. 6 presents the results of feature selection with both transformations and subsequent augmentations. Feature selection itself leads to improvement from feature creation. Yet, the addition of the log-transformed features provides the best results (0.3392) among all F1-scores of every imputation, model, and feature engineering combination.

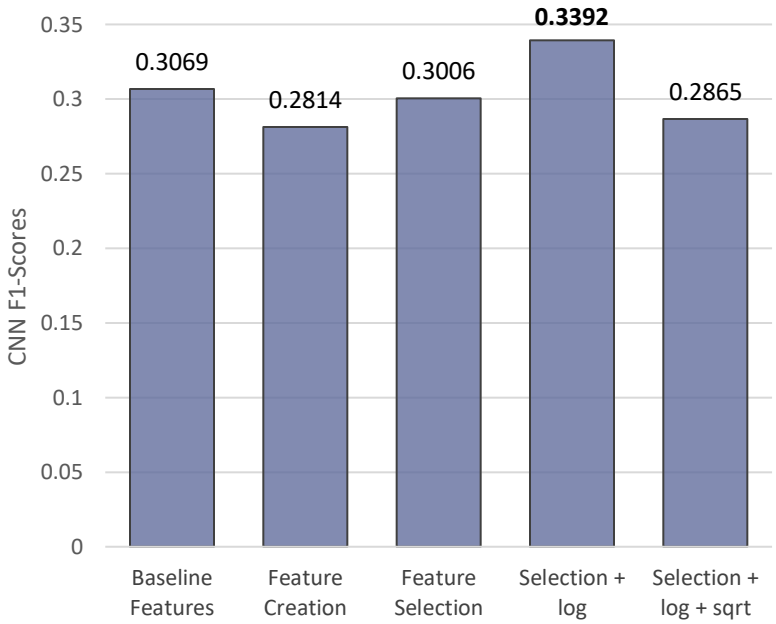

Fig. 6. Feature selection results of CNN with and MLP imputed data

The confusion matrix of the model with the best F1-score is given in Table 11. These results are achieved with the CNN without balancing batches, MLP imputation, and a subset of the NHANES features along with the 15 created features and log of the continuous features (Fig. 6).

Table 11: Confusion matrix of experiment with highest F1-score

	Predicted Negative	**Predicted Positive**
Actual Negative	9036	545
Actual Positive	499	268

This confusion matrix outcome result in the highest F1-score of 0.3392, a precision value of 0.3296, recall of 0.3494, and an AUC score of 0.64626. This F1-score is an 11% increase from that of the baseline features model.

On average, CNN without balancing batches produces the best F1-score results across all imputation and feature engineering methods. Additionally, machine learning imputation methods out-perform statistical methods on average. Lastly, a combination of feature creation, selection, transformation, and augmentation was proven to slightly improve model results.

4. Summary & Conclusions

Depression is a complex mental disease with unspecified causes. Despite the unique combination of demographic, dietary, clinical, and questionnaire data from the NHANES, I am unable to build a model to accurately predict depression with both high levels of precision and recall. The nature of this data with only 7.49% positive cases and many missing values did not provide enough information for DNNs to classify depression in the NHANES population. Therefore, missing value imputation, feature engineering, and re-sampling techniques were tested to evaluate potential improvements in classification accuracies with data of this nature.

Progressive machine learning imputation out-performed standard statistical imputation techniques. Thus, while machine learning imputation has greater time complexity than statistical imputation, future work using data with a large proportion of missing values may consider implementing progressive MLP imputation. Additionally, while re-sampling may be a useful method with less-highly imbalanced data [28, 49], training the DNNs on balanced mini-batches mainly exemplified the trade-off between precision and recall. Networks trained on these balanced batches generally resulted in very high recall values, with correspondingly low precision values. Nonetheless, when training the DANN, balanced batches may have slightly improved the model's ability to distinguish between positive and negative cases, indicated by an increased average F1-score (Fig. 4). Additionally, given the clinical nature of this data, false positive diagnoses (higher recall) may be preferred to false negative diagnoses (higher precision). Had our results been accurate enough to implement these DNNs in a clinical diagnosis setting, this trade-off would be taken into account when fine-tuning the model. In all, resampling techniques may be better suited for classification of less-highly imbalanced and ambiguous data and should be considered in clinical applications.

A major finding of this paper was the success of a 1D multi-layer CNN in comparison to a more typical DANN. Currently, CNNs are not widely used for binary classification using data of this nature, yet in this paper, we found the CNN's ability to learn the highly ambiguous data superior to the more commonly used DANN. Future work could compare the outcomes of previously successfully binary classifications using ANNs and other machine learning algorithms with a multi-layer 1D CNN. These 1D CNNs could generate improved accuracies in disease prediction and should be studied further.

Lastly, feature engineering techniques including feature selection and augmentation of transformed features afforded slight improvements to each model's performance. The addition of features from NHANES generated the greatest increase in F1-scores verifying the need for more data and information to accurately predict depression. In the same vein, feature creation resulted in some improved results by providing information in a different way. The addition of transformed features had varying effects on each model as did the continual augmentation of these transformed features. Feature selection or sub-setting showed slight increases in performance which is a result of a

decrease in correlation and noise within the data. In the end, a combination of feature creation, selection, and transformation boosted the F1-score of the baseline model by 11%. Future studies aiming to extract more information and a higher performance during binary classification should experiment with the feature engineering techniques outlined in this paper.

Depression widely goes undiagnosed and untreated in the US, and though there have been many identified risk factors and co-morbidities, these DNNs are unable to accurately predict depression using NHANES data. However, this paper demonstrates the advantages of progressive machine learning imputation techniques for missing values, the capabilities of a 1D CNN in binary classification, and lastly, the slight improvements gained with feature engineering when dealing with indirect or ambiguous data.

References

1. Baxt WG (1995) Application of artificial neural networks to clinical medicine. Lancet 346:1135–1138. https://doi.org/10.1016/S0140-6736(95)91804-3
2. López-Martínez F, Rolando Núñez-Valdez E, Crespo RG, García-Díaz V (2020) An artificial neural network approach for predicting hypertension using NHANES data. 10:10620. https://doi.org/10.1038/s41598-020-67640-z
3. Mantzaris DH, Anastassopoulos GC, Lymberopoulos DK (2008) Medical disease prediction using artificial neural networks. In: 8th IEEE International Conference on BioInformatics and BioEngineering, BIBE 2008
4. Yu W, Liu T, Valdez R, et al (2010) Application of support vector machine modeling for prediction of common diseases: The case of diabetes and pre-diabetes. BMC Med Inform Decis Mak 10:1–7. https://doi.org/10.1186/1472-6947-10-16
5. Jegan C, Kumari VA, Chitra R Classification Of Diabetes Disease Using Support Vector Machine Development of an Intelligent system for the diagnosis of cardiovascular diseases View project Classification Of Diabetes Disease Using Support Vector Machine. 3:1797–1801
6. Mohan V, Vijayarani S, Dhayanand MS, Research Scholar MP Kidney Disease Prediction Using SVM and ANN Algorithms. Int J Comput Bus Res
7. Khalilia M, Chakraborty S, Popescu M (2011) Predicting disease risks from highly imbalanced data using random forest. BMC Med Inform Decis Mak 11:51. https://doi.org/10.1186/1472-6947-11-51
8. Maji S, Arora S (2019) Decision Tree Algorithms for Prediction of Heart Disease. In: Lecture Notes in Networks and Systems. Springer, pp 447–454
9. (2017) What Is Depression? In: Am. Psychiatr. Assoc. https://www.psychiatry.org/patients-families/depression/what-is-depression. Accessed 2 Jul 2020
10. (2020) Depression. In: World Heal. Organ. https://www.who.int/news-room/fact-sheets/detail/depression. Accessed 2 Jul 2020
11. Belmaker RH, Agam G (2008) Major Depressive Disorder. N Engl J Med 358:55–68. https://doi.org/10.1056/NEJMra073096
12. Majidi M, Khadembashi N, Etemad K, et al (2019) Associated factors with major depression: a path analysis on NHANES 2013–2014 study. Int J Cult Ment Health. https://doi.org/10.1080/17542863.2018.1563623
13. Van de Velde S, Bracke P, Levecque K (2010) Gender differences in depression in 23 European countries. Cross-national variation in the gender gap in depression. Soc Sci Med 71:305–313. https://doi.org/10.1016/j.socscimed.2010.03.035
14. Patel V, Burns JK, Dhingra M, et al (2018) Income inequality and depression: a systematic review and meta-analysis of the association and a scoping review of mechanisms. World Psychiatry 17:76–89. https://doi.org/10.1002/wps.20492
15. Liu M, Wu L, Yao S (2016) Dose-response association of screen time-based sedentary behaviour in children and adolescents and depression: a meta-analysis of observational studies. Br J Sports Med 50:1252–1258. https://doi.org/10.1136/bjsports-2015-095084
16. Liu Y, Ozodiegwu ID, Yu Y, et al (2017) An association of health behaviors with depression and metabolic risks: Data from 2007 to 2014 U.S. National Health and Nutrition Examination Survey. J Affect Disord 217:190–196. https://doi.org/10.1016/j.jad.2017.04.009
17. Patel PO, Patel MR, Baptist AP (2017) Depression and Asthma Outcomes in Older Adults: Results from the National Health and Nutrition Examination Survey. J Allergy Clin Immunol Pract 5:1691-1697.e1. https://doi.org/10.1016/j.jaip.2017.03.034
18. Cizza G, Primma S, Csako G (2009) Depression as a risk factor for osteoporosis. Trends Endocrinol Metab 20:367–373. https://doi.org/10.1016/j.tem.2009.05.003
19. Iranpour S, Sabour S (2019) Inverse association between caffeine intake and depressive symptoms in US

adults: data from National Health and Nutrition Examination Survey (NHANES) 2005–2006. Psychiatry Res 271:732–739. https://doi.org/10.1016/j.psychres.2018.11.004

20. Leung CW, Epel ES, Willett WC, et al (2015) Household Food Insecurity Is Positively Associated with Depression among Low-Income Supplemental Nutrition Assistance Program Participants and Income-Eligible Nonparticipants. J Nutr 145:622–627. https://doi.org/10.3945/jn.114.199414

21. Hvas A-M, Juul S, Bech P, Nexø E (2004) Vitamin B₆ Level Is Associated with Symptoms of Depression. Psychother Psychosom 73:340–343. https://doi.org/10.1159/000080386

22. Jerez JM, Molina I, García-Laencina PJ, et al (2010) Missing data imputation using statistical and machine learning methods in a real breast cancer problem. Artif Intell Med 50:105–115. https://doi.org/10.1016/j.artmed.2010.05.002

23. Heaton J (2016) An empirical analysis of feature engineering for predictive modeling. In: Conference Proceedings - IEEE SOUTHEASTCON. Institute of Electrical and Electronics Engineers Inc.

24. Yan H, Jiang Y, Zheng J, et al (2006) A multilayer perceptron-based medical decision support system for heart disease diagnosis. Expert Syst Appl 30:272–281. https://doi.org/10.1016/j.eswa.2005.07.022

25. Zanella-Calzada L, Galván-Tejada C, Chávez-Lamas N, et al (2018) Deep Artificial Neural Networks for the Diagnostic of Caries Using Socioeconomic and Nutritional Features as Determinants: Data from NHANES 2013–2014. Bioengineering 5:47. https://doi.org/10.3390/bioengineering5020047

26. Chiu J-S, Li Y-C, Yu F-C, Wang Y-F (2006) Applying an Artificial Neural Network to Predict Osteoporosis in the Elderly

27. Lemineur G, Harba R, Kilic N, et al (2007) Efficient estimation of osteoporosis using Artificial Neural Networks. In: IECON Proceedings (Industrial Electronics Conference). pp 3039–3044

28. Yildirim P (2017) Chronic Kidney Disease Prediction on Imbalanced Data by Multilayer Perceptron: Chronic Kidney Disease Prediction. In: Proceedings - International Computer Software and Applications Conference. IEEE Computer Society, pp 193–198

29. Dutta A, Batabyal T, Basu M, Acton ST An Efficient Convolutional Neural Network for Coronary Heart Disease Prediction

30. Kiranyaz S, Avci O, Abdeljaber O, et al (2019) 1D Convolutional Neural Networks and Applications: A Survey

31. Wyatt MR, Papas M, Johnston T, Taufer M (2016) Development of a scalable method for creating food groups using the NHANES dataset and MapReduce. In: ACM-BCB 2016 - 7th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics. Association for Computing Machinery, Inc, New York, NY, USA, pp 118–127

32. Kroenke K, Spitzer RL, Williams JBW (2001) The PHQ-9: Validity of a brief depression severity measure. J Gen Intern Med 16:606–613. https://doi.org/10.1046/j.1525-1497.2001.016009606.x

33. NHANES 2009-2010: Mental Health - Depression Screener Data Documentation, Codebook, and Frequencies. https://wwwn.cdc.gov/Nchs/Nhanes/2009-2010/DPQ_F.htm. Accessed 29 Jun 2020

34. Pedregosa F, Varoquaux G, Gramfort A, et al (2011) Scikit-learn: Machine Learning in Python. J Mach Learn Res 12:2825–2830

35. 1.17. Neural network models (supervised) — scikit-learn 0.23.1 documentation. https://scikit-learn.org/stable/modules/neural_networks_supervised.html. Accessed 30 Jun 2020

36. Kana M (2020) Handling Missing Data for Advanced Machine Learning | Towards AI—Multidisciplinary Science Journal. In: Towar. AI. https://medium.com/towards-artificial-intelligence/handling-missing-data-for-advanced-machine-learning-b6eb89050357. Accessed 2 Jul 2020

37. Razzak MI, Imran M, Xu G (2020) Big data analytics for preventive medicine. Neural Comput Appl 32:4417–4451. https://doi.org/10.1007/s00521-019-04095-y

38. Geetha R, Professor R, Head &, Sivagami G (2011) Parkinson Disease Classification using Data Mining

Algorithms

39. LeNail A (2019) NN-SVG: Publication-Ready Neural Network Architecture Schematics. J Open Source Softw 4:747. https://doi.org/10.21105/joss.00747

40. Acharya UR, Oh SL, Hagiwara Y, et al (2017) A deep convolutional neural network model to classify heartbeats. Comput Biol Med 89:389–396. https://doi.org/10.1016/j.compbiomed.2017.08.022

41. Nigam V Understanding Neural Networks. From neuron to RNN, CNN, and Deep Learning | by vibhor nigam | Towards Data Science. https://towardsdatascience.com/understanding-neural-networks-from-neuron-to-rnn-cnn-and-deep-learning-cd88e90e0a90. Accessed 11 Aug 2020

42. Srinivasamurthy RS (2018) TigerPrints Understanding 1D Convolutional Neural Networks Using Multiclass Time-Varying Signals

43. Yamashita R, Nishio M, Do RKG, Togashi K (2018) Convolutional neural networks: an overview and application in radiology. Insights Imaging 9:611–629

44. Dua D, Graff C (2019) UCI Machine Learning Repository. In: Irvine, CA Univ. California, Sch. Inf. Comput. Sci.

45. Asri H, Mousannif H, Al Moatassime H, Noel T (2016) Using Machine Learning Algorithms for Breast Cancer Risk Prediction and Diagnosis. In: Procedia Computer Science. Elsevier, pp 1064–1069

46. Mazurowski MA, Habas PA, Zurada JM, et al (2008) Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. Neural Networks 21:427–436. https://doi.org/10.1016/j.neunet.2007.12.031

47. Islam MM, Iqbal H, Haque MR, Hasan MK (2018) Prediction of breast cancer using support vector machine and K-Nearest neighbors. In: 5th IEEE Region 10 Humanitarian Technology Conference 2017, R10-HTC 2017. Institute of Electrical and Electronics Engineers Inc., pp 226–229

48. imblearn.keras.BalancedBatchGenerator — imbalanced-learn 0.5.0 documentation. https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.keras.BalancedBatchGenerator.html. Accessed 11 Aug 2020

49. Khalilia M, Chakraborty S, Popescu M (2011) Predicting disease risks from highly imbalanced data using random forest. BMC Med Inform Decis Mak 11:1–13. https://doi.org/10.1186/1472-6947-11-51

Code references:

https://github.com/mrwyattii/NHANES-Downloader

https://gist.github.com/michelkana/238fe080594525a86b84195c184ba5c0#file-missing_data_logistic_regression-py

https://www.topbots.com/handling-missing-data-for-machine-learning/

https://github.com/amir-jafari/Deep-Learning/blob/master/Keras_/CNN/2_SignalClassification_1D/example_ActivityPrediction.py

All documented computer listings/code can be found at https://github.com/csklaver/Capstone-Group6/tree/master/Code