Caroline Sklaver

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

George Washington University, Department of Data Science
carolinesklaver@gmail.com; csklaver@gwu.edu

Acknowledgements:

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

Abstract

Depression is characterized as a long-term affect in mood is caused by a complex combination of genetic, physiological, environmental, and social factors. It is a leading cause of disability and a major contributor to the global burden of disease. This work aims to improve the classification of depression using deep neural networks by implementing progressive machine learning imputation techniques and feature engineering with demographic, heath, and laboratory data from the National Health and Nutrition Examination Survey (NHAENS). Both a deep artificial neural network (DANN) and a multi-layer 1-dimensional (1D) convolutional neural network (CNN) are proposed for binary classification with this highly imbalanced data. Using F1-scores for evaluation, the CNN outperforms the DANN and demonstrates the potential applications of CNNs with standard structured datasets. Additionally, this work demonstrates the advantages of progressive machine learning imputation techniques for missing values. Lastly, slight improvements are achieved using feature engineering including a combination of feature creation, selection, and transformation.

Key words: feature engineering, imbalance, depression, neural networks,

Declarations

Funding: Not applicable

Conflicts of interest/Competing interests: Not applicable

Availability of data and material: All data can be downloaded from NHANES webpage at https://wwwn.cdc.gov/nchs/nhanes/

Code availability: Code is available on GitHub at https://github.com/csklaver/Capstone-Group6

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).

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 paper, I 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 unspecified causes.

1.3 Related work

Previous work 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

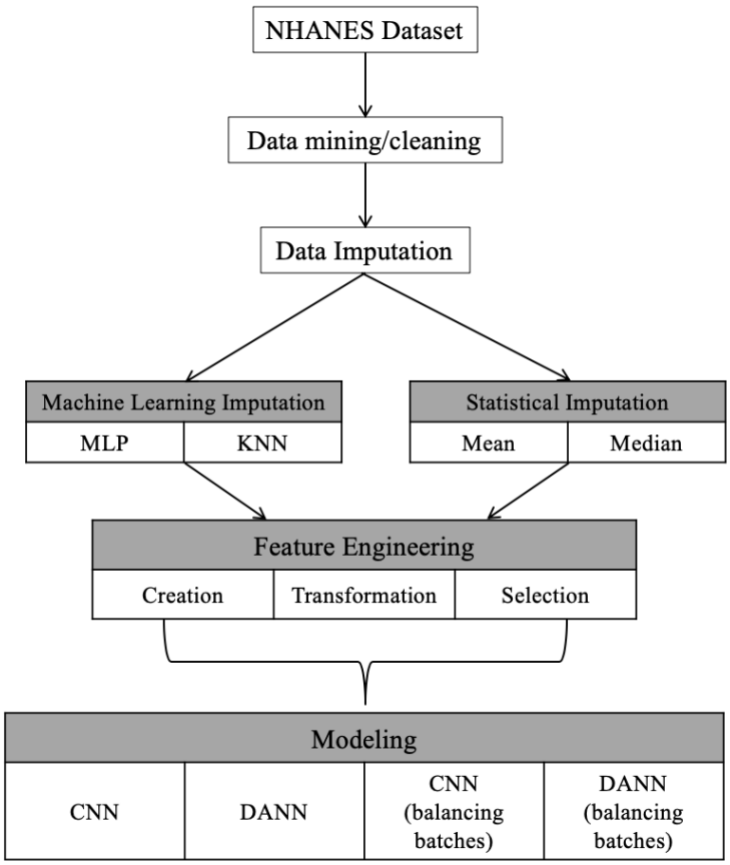
[25], and osteoporosis [26, 27]. Lopez-Martinez et. al. 2020 [2], presented a multi-layer neural network architecture with 3 hidden layers to identify hypertension in NHANES patients. This neural network achieved a lower error rate compared to that of Logistic Regression (LR), SVM, and Decision Tree (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. Data & Methods

Figure 1 outlines the methodology of this paper which includes data cleaning, data imputation methods, feature engineering, and ANN modeling.

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. This data is highly imbalanced with only 7.49% of the observations being positive for depression.

The breakdown of this extended number of features by NHANES component is given in Table 1, along with their data types (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 2.4).

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 Data 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.

The detailed explanation of an MLP in this section is paramount in understanding the function and implementation of the DNNs described in Section 2.3. MLP is a supervised machine learning algorithm 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. MLP can be used for either classification or regression, depending on the nature of the target feature. Both classification and regression use backpropagation, a form of gradient-descent, to repeatedly adjust the weights to minimize loss.

Rectified Linear Unit (ReLU) is the activation function implemented for the hidden layers in both MLP regression and classification. ReLU outputs all positive values as identity and all negative values as 0. In this data imputation approach, if the data type of the target variable is discrete, the loss function is cross-entropy (logarithmic loss). Meanwhile if the data type is continuous, the loss function is mean squared error.

The output layer of an MLP transforms the values from the final hidden layer into output values [34]. In binary classification, MLP uses the logistic function, where 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]. In multi-class classification, MLP uses the SoftMax function in the output layer, which calculates the probabilities of each class outcome and assigns the final output to the class with the greatest probability. 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 column (continuous, binary, or multi-class). A simple architecture consisting of two hidden layers was used in this process.

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 Neural Networks

Neural networks can have many different architectures. Section 2.2.2 describes the process of a feed-forward neural network and the corresponding functions 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, the same DANN and CNN outlined in the following sections, with the same set of hyper-parameters was implemented.

2.3.1 DANN

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

Fig. 2 DANN architecture [37]

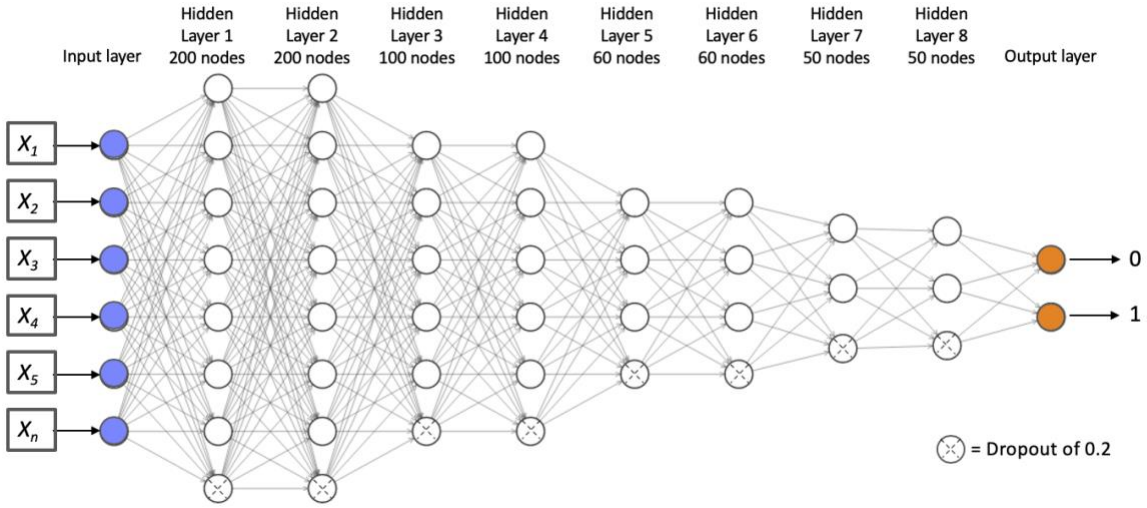


Table 2 DNN Model Architecture and Parameters

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	Confusion matrix, Accuracy, Loss, Recall, Precision, F1-score, AUC

The transformation of the input features using weights and biases in each hidden layer along with the ReLU activation function are explained in Section 2.2.2. 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 given by Eq. (1), which is a permutation cross-entropy, where the number of classes is equal to 2, N is the total number of samples, y_i is the target output, and $\hat{y}_i$ is the predicted output after ReLU activation.

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

In this DANN the number of input nodes is equal to the number of features in each permutation of the engineered data. Two output nodes and the SoftMax output function 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 [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.

2.4.2 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 that contain biases and weights [38].The hidden layers of a CNN are convolutional, pooling, and fully connected feed-forward layers [39]. 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 [40]. 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 or ANN previously described. Lastly, the fully connected layers map the extracted features into the final classification outputs [41].

Figure 3 represents the CNN model architecture implemented in this project and Table 3 outlines the parameters of this architecture.

Fig. 3 CNN architecture [37]

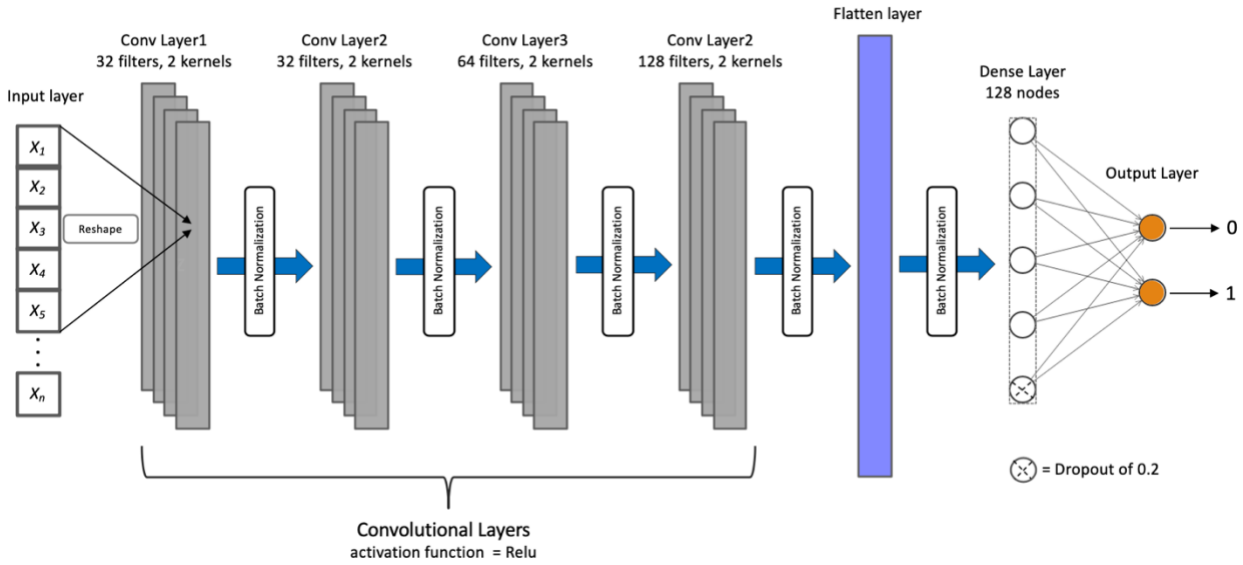


Table 3: CNN Model Architecture and Parameters

Parameter	**Value**
Reshaped input	(# observations, # features, 1)
Number of hidden layers	5
Conv1D layer 1 filters	32
Conv1D layer 1 filter size	2
Conv1D layer 1 activation function	Relu
Conv1D layer 2 filters	32
Conv1D layer 2 filter size	2
Conv1D layer 2 activation function	Relu
Conv1D layer 3 filters	64
Conv1D layer 3 filter size	2
Conv1D layer 3 activation function	Relu
Conv1D layer 4 filters	128
Conv1D layer 4 filter 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	Confusion matrix, Accuracy, Loss, Recall, Precision, F1-score, AUC

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, 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. The loss function implemented is binary cross-entropy (Eq. 1). Lastly, the output layer contains 2 nodes representing the binary output and uses SoftMax as the activation function. After ReLU activation, between each convolutional layer, batch normalization is implemented to normalize the output tensors to a mean of 0 and standard deviation of 1.

2.4 Feature Engineering

In an effort to enhance the performance of the DNNs described in the previous section, various feature engineering techniques were implemented. 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 the sub-setting of features from the data based on feature importance, correlations, associations, or other statistical methods. In this paper, features selection involved the inclusion of features with proven prior associations with depression and the removal 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 raw features were 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 feature engineering 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

2.5 Balancing Batches

For each data imputation method and feature engineering experiment, both the DANN and CNN models were trained with normal training data, which is highly imbalanced, and then with balanced mini batches. These balanced mini batches used random under-sampling of the majority class without replacement to create a random, balanced batches of size 32 [42]. This technique was implemented in order to evaluate the testing results when models are trained on balanced data, as resampling methods are commonly used with imbalanced data classification [28, 43].

2.6 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

Table 5 in the appendix includes all F1-scores of each imputation and feature engineering method with both DANN and CNN models with and without balanced batches (BB). In Table 6, 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 CNN training (0.3069). Machine learning imputation techniques (MLP and KNN) have better performance on the 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 6 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 7 provides the average F1-score results of each network 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 7 Comparison of Average Neural Network Performances

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

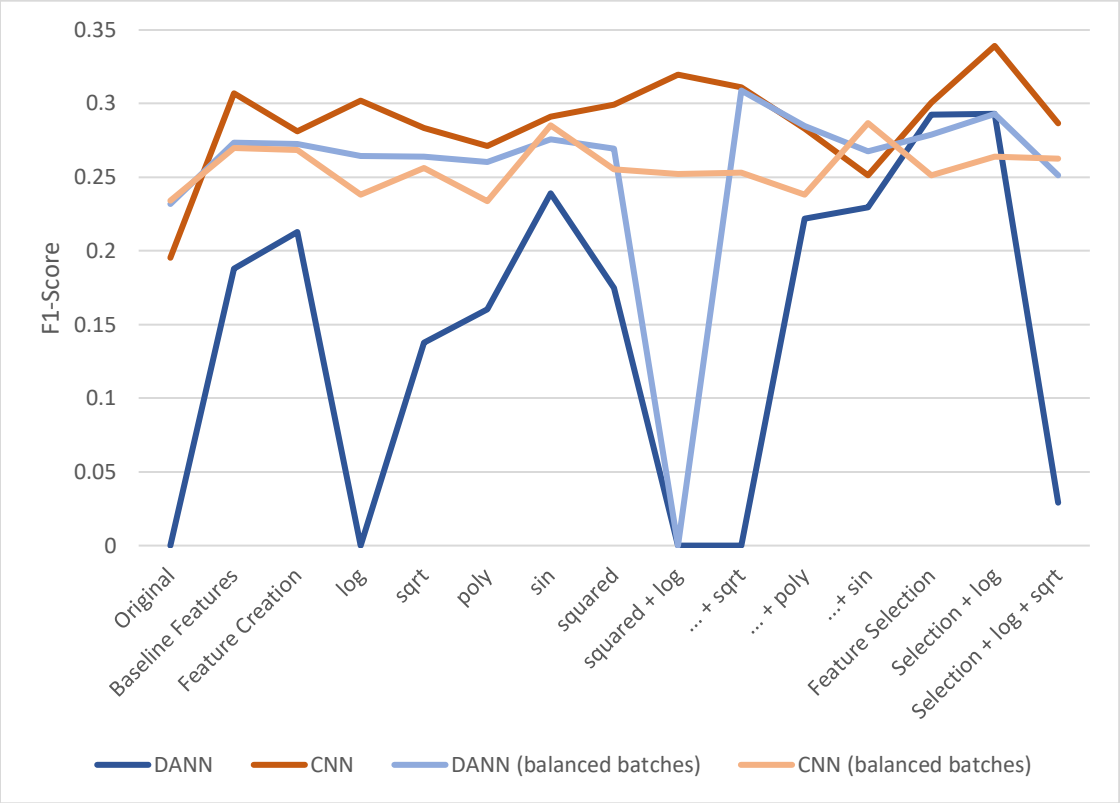
To compare imputation methods, Table 8 presents the average F1-score of each imputation type across all model and feature engineering experiments (Column 1). In this case, KNN provides the highest average F1-score. However, taking the average F1-score results from just the CNN results (the highest performing model shown in Table 7), across all feature engineering methods, MLP imputation has the best F1-score (0.274), followed by KNN (Column 2).

Table 8 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

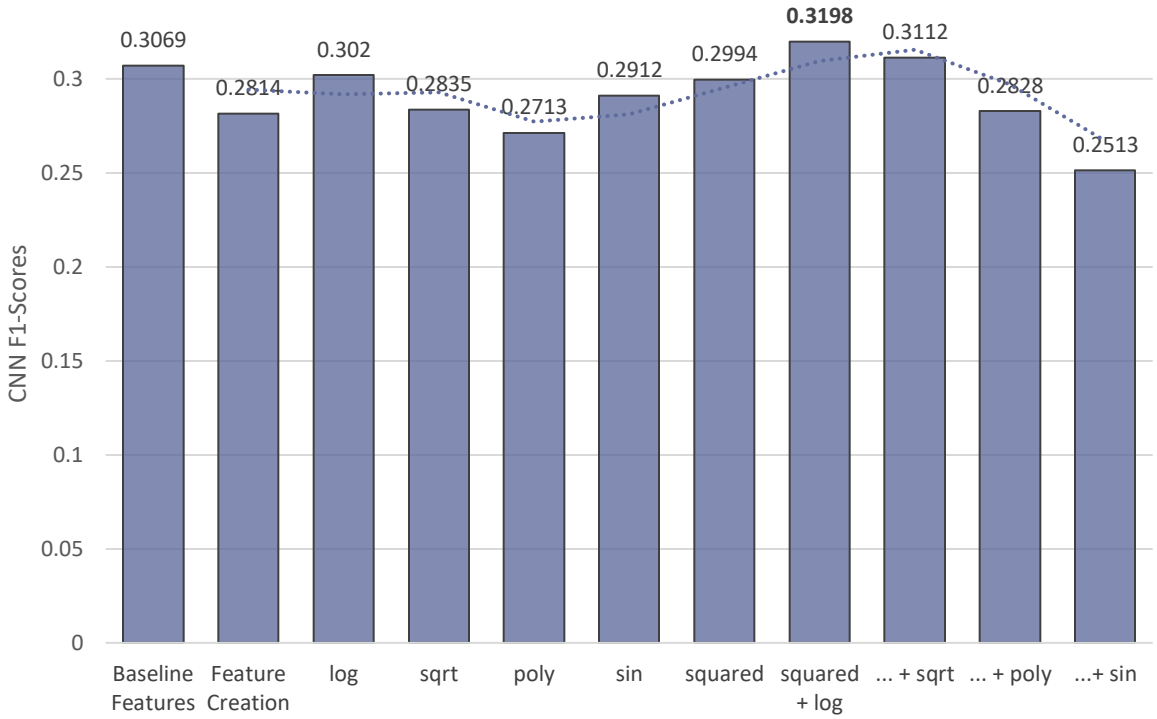
Next, to examine the effects of the feature engineering techniques on each model type (DANN, CNN, and BB), Fig. 4 depicts the fluctuations in F1-scores as a result of feature engineering with the MLP imputed data. The DANN has the greatest fluctuations and is the least reliable model, however, it is 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 DNN with MLP imputated data



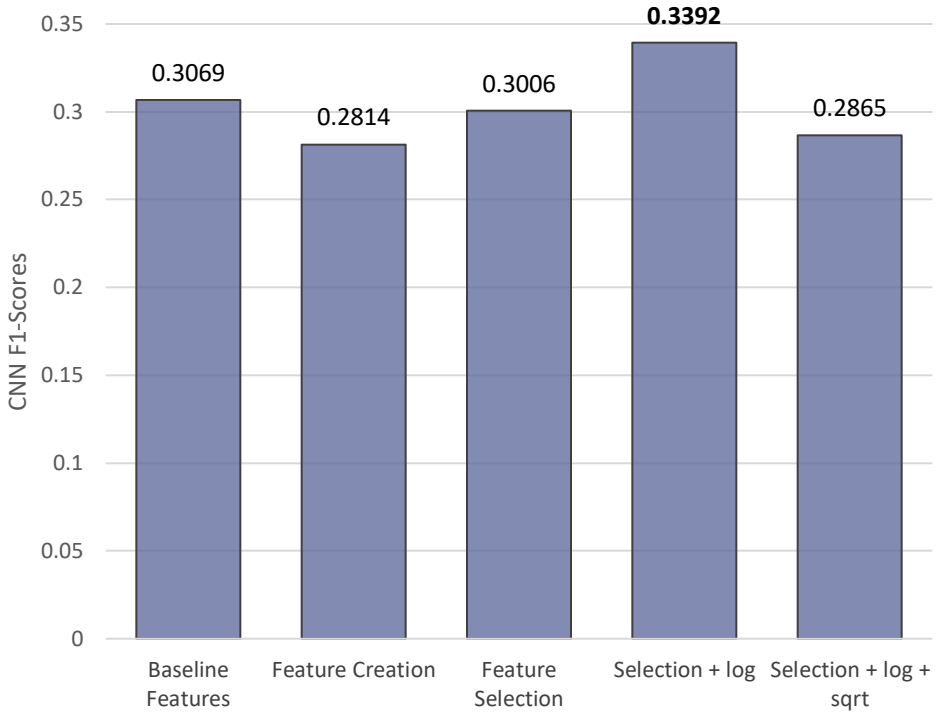
To take a closer look at the effects of feature engineering on the best model with the best imputation method (Tables 6 and 7), Fig. 5 depicts the changes in F1-scores after feature creation, transformations, and continuous augmentations with MLP imputed data and the CNN without balancing batches. In 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 is with the augmentation the square and the 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 each 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 9. This model is achieved with the CNN without balancing batches, MLP imputation, and a subset of the extended number of NHANES features along with the 15 created features and log of the continuous features (Fig. 6).

Table 9 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 a 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 shown to slightly improve model results.

4. Discussion & Conclusions

Depression is a complex mental disease with unspecified causes. Despite the unique combination of demographic, dietary, clinical, and questionnaire data from NHANES, we are 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 multi-layer ANNs to classify depression. 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 for other types of imbalanced data classifications [28, 43], training the DNNs in this paper on balanced mini-batches 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. However, when training the DANN, balanced batches may have improved the model's ability to distinguish between positive and negative cases, indicated by an increased 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 employed as a tool in clinical diagnosis, this detail 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 but should be considered in clinical applications.

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

Lastly, feature engineering techniques including feature creation, 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 generally 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 out of a machine learning binary classification should experiment with the feature engineering techniques outlined in this paper.

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

38. 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

39. 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

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

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

42. 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

43. 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

Appendix

Table 5 F1-score Results of All Imputation and Feature Engineering Experiments on both DNNs

	MLP Imputation				**KNN Imputation**				**Mean Imputation**				**Median Imputation**			
Feature Engineering Method	DANN	CNN	DANN BB	CNN BB	NN	CNN	DANN BB	CNN BB	NN	CNN	DANN BB	CNN BB	NN	CNN	DANN BB	CNN BB
NHANES minimal features	0.0000	0.0000	0.0000	0.0000	0.0474	0.2336	0.2339	0.2327	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
NHANES baseline features	0.1881	*0.3069*	0.2736	0.2698	0.0000	0.2841	0.2777	0.2694	0.0000	0.2712	0.2618	0.2647	0.1703	0.2356	0.2595	0.2573
Baseline + Created features (BCF)	0.2128	0.2814	0.2725	0.2685	0.2357	*0.3193*	0.2525	0.2629	0.1258	0.2766	0.2736	0.2544	0.1845	**0.3377**	0.2923	0.2648
BCF + log(cont)	0.0000	*0.3020*	0.2643	0.2381	0.2749	0.2713	*0.3091*	0.2661	0.2642	0.2795	0.2515	0.2247	0.2307	0.2764	0.2676	0.2545
BCF + $(cont)_{1/2}$	0.1378	0.2835	0.2642	0.2565	0.0944	0.2914	0.2569	0.2627	0.0000	0.2818	0.2491	0.2378	0.1542	0.2997	0.2440	0.2788
BCF + $(cont)_2$	0.1602	0.2713	0.2606	0.2338	0.1651	0.2530	0.2792	0.2758	0.2224	*0.3144*	0.2914	0.2754	0.1852	0.2718	0.2405	0.2737
BCF + polynomial	0.2392	0.2912	0.2759	0.2853	0.2782	0.2747	0.2841	0.2778	0.2416	*0.3159*	0.2818	0.2782	0.1915	0.2908	0.2827	0.2977
BCF + sine(cont)	0.1747	0.2994	0.2696	0.2553	0.1498	0.2701	0.2797	0.2482	0.0000	0.2566	0.2396	0.2822	0.2047	*0.3016*	0.2691	0.2622
BCF + $(cont)_2$ + log(cont)	0.0000	*0.3198*	0.0000	0.2521	0.2609	0.2925	0.2869	0.2581	0.0000	0.2269	0.2795	0.2529	0.1689	0.2638	0.2927	0.2289
BCF + $(cont)_2$ + log(cont) + $(cont)_{1/2}$	0.0000	*0.3112*	*0.3089*	0.2532	0.2748	0.2815	0.2295	0.2535	0.0000	0.2708	0.2765	0.2699	0.1827	0.2771	0.2856	0.2556
BCF + $(cont)_2$ + log(cont) + $(cont)_{1/2}$ + polynomial	0.2220	0.2828	0.2847	0.2384	0.1994	0.2967	0.2564	0.2553	0.0000	0.2498	0.2601	0.2861	0.1797	0.2760	0.2543	0.2615
BCF + $(cont)_2$ + log(cont) + $(cont)_{1/2}$ + polynomial + sine(cont)	0.2295	0.2513	0.2676	0.2868	0.2390	*0.3033*	0.2719	0.2693	0.1858	*0.3103*	0.2686	0.2446	0.0000	0.2441	0.2415	0.2420
Subset of BCF	0.2925	*0.3006*	0.2790	0.2515	0.0000	0.2988	0.2827	0.2752	0.2359	0.2904	*0.2990*	0.2712	0.2969	0.2632	0.2601	0.2660
Subset + log(cont)	0.2931	**0.3392**	0.2931	0.2640	0.2617	0.2679	*0.3010*	0.2546	0.0000	*0.2999*	0.2876	0.2900	0.2322	0.2865	0.2705	0.2580
Subset + log(cont) + $(cont)_{1/2}$	0.0293	0.2865	0.2512	0.2627	0.2983	**0.3354**	0.2508	0.2783	0.0000	**0.3369**	0.2561	0.2808	0.2586	0.2843	0.2746	0.2569