Expert Systems With Applications 135 (2019) 273–286

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A Recursive General Regression Neural Network (R-GRNN) Oracle for classification problems

Dana Bani-Hani*, Mohammad Khasawneh

Department of Systems Science and Industrial Engineering, State University of New York at Binghamton, Binghamton, NY 13902, USA

ARTICLE INFO

Article history:
Received 20 October 2018
Revised 17 May 2019
Accepted 6 June 2019
Available online 8 June 2019

Keywords:
GRNN Oracle
Data Mining, Machine Learning
Classification
Ensemble
Prediction Model

ABSTRACT

This research introduces the Recursive General Regression Neural Network Oracle (R-GRNN Oracle) and is demonstrated on several binary classification datasets. The traditional GRNN Oracle classifier (Masters et al., 1998) combines the predictive powers of several machine learning classifiers by weighing the amount of error each classifier has on the final predictions. Each classifier is assigned a weight based on the percentage of errors it contributes to the final predictions as the classifiers evaluate the dataset. The proposed R-GRNN Oracle is an enhancement to the GRNN Oracle in which the proposed algorithm consists of an oracle within an oracle – where the inner oracle acts as a classifier with its own predictions and error contribution. By combining the inner oracle with other classifiers, the R-GRNN Oracle produces superior results. The classifiers considered in this study are: Support Vector Machine (SVM), Multilayer Perceptron (MLP), Probabilistic Neural Network (PNN), Gaussian Naïve Bayes (GNB), K-Nearest Neighbor (KNN), and Random Forest (RF). To demonstrate the effectiveness of the proposed approach, several datasets were used, with the primary one being the publicly available Spambase dataset. The predictions of SVM, MLP, KNN, and RF were used to create the first GRNN Oracle, which was then enhanced with the high performances of SVM and RF to create the second oracle, the R-GRNN Oracle. The combined recursive model was 93.24% accurate using 10-fold cross validation, higher than the 91.94% of the inner GRNN Oracle and the 91.29% achieved by RF, the highest performance by a stand-alone classifier. The R-GRNN Oracle was not only the most accurate, but it also had the highest AUC, sensitivity, specificity, precision, and F1-score (97.99%, 91.86%, 94.40%, 93.28%, and 92.57%, respectively). The research contribution of this paper is introducing the concept of recursion (a concept not fully explored in machine learning models and applications) and testing this structure's ability on further enhancing the performance of the traditional oracle. The recursive model has also been applied to several other datasets: The Human Resources, Bank Marketing, and Monoclonal Gammopathy of Undetermined Significance (MGUS) datasets. The results of these implementations are summarized in this paper.

1. Introduction

The world today produces huge and complex amounts of data every second. In this new era of big data, advanced analytic methods can extract valuable information, patterns, trends, and associations to provide meaningful insights. Processing such data manually would be impractical if not impossible, therefore, the need to automate such processes is needed. Tasks too complex for humans to code and process directly require machine learning. Machine learning helps analyze big data by focusing on designing algorithms that can learn patterns in the data to make predictions. It is a branch of artificial intelligence that teaches machines how to learn from experiences and adapt. Successful data mining requires effective machine learning techniques. Data mining is defined as the process of discovering properties and extracting valuable information from large, incomplete, and noisy raw data that is stored in databases, data warehouses, or other information repositories. In data mining, the data is stored electronically and is processed through computers (Witten, Frank, Hall, & Pal, 2016). It is about solving problems by analyzing data already present in databases, where some of its tasks include association rule learning, clustering, classification, and regression (Esfandiari, Babavalian, Moghadam, & Tabar, 2014). It is applied to various disciplines and industries such as manufacturing, customer relationship management, fraud detection, banking, marketing, and healthcare.

The General Regression Neural Network Oracle (GRNN Oracle), developed by Masters et al. in 1998, combines the predictions of individually trained classifiers and outputs one superior prediction.

* Corresponding author.
E-mail addresses: dbaniha1@binghamton.edu (D. Bani-Hani), mkhasawn@binghamton.edu (M. Khasawneh).

https://doi.org/10.1016/j.eswa.2019.06.018
0957-4174/

A classifier is a machine learning model that is used to predict a categorical output variable while a regressor predicts output variables that take on continuous values. The GRNN Oracle determines the error rate for each classifier and assigns weights based on them. The classifiers with lower error rates have a greater weight which leads to a greater influence on the final prediction. The final prediction for an unknown observation is calculated by summing each classifier's prediction for belonging in a certain class for that unknown observation multiplied by the classifier's weight. As the weights reflect the percentage of errors, the total weight is equal to one.

Because of the strong capabilities of the oracle, in this research, it has been enhanced and developed to consist of two GRNN Oracles; one within the other. The first oracle is created through its own combination of algorithms and acts as a classifier with its own predictions and error contribution to a set of unknown observations. It is then combined with other classifiers to create a new, outer oracle that has been named the Recursive General Regression Neural Network Oracle (R-GRNN Oracle). To demonstrate the effectiveness of the proposed oracle, it has been applied to the Spambase dataset Hopkins et al. (1999) that is publicly available from the UCI (University of California, Irvine) machine learning repository, where the model's performance is compared to the performance of other classifiers through the performance metrics of accuracy, the area under the receiver operating characteristic (ROC) curve (AUC), sensitivity, specificity, precision, and F1-score. These performance metrics are standard in machine learning when assessing a model's classification ability. For more details, the reader is referred to the books "Machine Learning in Python: Essential Techniques for Predictive Analysis" by Bowles (2015) and "Data Mining: Practical Machine Learning Tools and Techniques" by Witten et al. (2016).

To further demonstrate the power of the model, it has also been applied to several other publicly available datasets in which the results are summarized. The datasets are Human Resources, Bank Marketing Moro et al. (2014), and MGUS (Monoclonal Gammopathy of Undetermined Significance) Kyle et al. (2002).

There are many machine learning algorithms used for classification, regression, or both. Each algorithm has a set of strengths and weaknesses that distinguish it from other algorithms. A specific algorithm can perform better than another based on several factors, such as the nature of the dataset, but no one algorithm can outperform the rest in all or most prediction problems. This study addresses this problem by combining the strengths of several powerful machine learning algorithms to create a R-GRNN Oracle in which its predictive capability outweighs the predictive capabilities of the other demonstrated classifiers, including the traditional oracle that envelops its own combination of classifiers.

The remainder of this paper is organized as follows: Section 2 presents literature review on machine learning, ensemble learning, and the GRNN Oracle, it also addresses the literature gap and motivations behind this study. Section 3 explains the methodology followed in this study and the methodology behind the R-GRNN Oracle. It also presents the related work to the Spambase dataset. Section 4 includes the experimental analysis and results on the application of the recursive model on the Spambase dataset, as well as the final results of its application on the other datasets. Finally, Section 5 presents the conclusion, limitations, and future work related to this study.

2. Literature review

Machine learning techniques are generally separated into three categories: supervised learning, unsupervised learning, and reinforcement learning (Witten et al., 2016). Supervised learning deals with data that is already labeled (that has a dependent variable: an output), wherein the algorithm tries to find the relationships between the inputs and outputs in order to predict unknown observations. An example would be the prediction and diagnosis of diabetes for a certain patient based on the attributes of previously recorded patients. In unsupervised learning however, there are no labels; therefore, the given algorithm tries to sperate the observations into clusters that share the same characteristics. An appropriate example to give for unsupervised learning is the segmentation of customers for marketing purposes in order to target each segment with specific products based on their common interests. Reinforcement learning on the other hand deals with algorithms that can learn based on their surrounding environments and adjust accordingly. An example would be a self-driving car; based on its surroundings, it adjusts its algorithm to maximize the reward needed to be obtained.

This study focuses on supervised learning, more specifically, the classification task in which classification is a widely used technique where the algorithm learns how data can be assigned to certain categories or classes. The process uses patterns it finds to map new data into the class it fits best. These algorithms differ in how they can separate, distinguish, and map the data into their corresponding classes. Techniques such as ensembles may boost the ability of prediction models to classify new observations. Ensembles are multiple algorithmic systems made up of several "base learners" where their ability to predict is stronger than that of a single algorithm (Zhang & Ma, 2012). The two most commonly used methods for ensemble learning are voting and averaging. Voting chooses the class of the unknown observation from the majority votes of the algorithms that make up the ensemble – this is used for classification tasks. However, with regression tasks, the average output of the algorithms is considered the final output of the ensemble. The three types of ensembles most used for combining predictions are boosting, bagging (bootstrap aggregating), and stacking; the reader is referred to the book "Ensemble Machine Learning: Methods and Applications" by Zhang and Ma (2012) for details on these methods. Many studies tend to use ensemble learning because of its desirable performance.

The remainder of this section provides a review of the literature relevant to the topic of this research. In Section 2.1, some of the recent studies on classification and ensemble learning are presented and discussed. In Section 2.2, the studies on the original GRNN Oracle are reviewed. In Section 2.3, the literature gap and motivating factors behind this research are explained. A detailed look on how the GRNN Oracle works is discussed in Section 3.1. The proposed R-GRNN Oracle is tested on multiple datasets, including the Spambase dataset. For ease of reference in the comparative study, a more detailed review of the literature related to the application of various algorithms on the Spambase dataset is provided in Section 3.4.

2.1. Classification and ensemble learning

Chaurasia (2017) applied classification and regression trees (CART), iterative dichotomized 3 (ID3), and decision tables to predict early heart disease where CART achieved the highest accuracy of 83.49%. Zheng, Yoon, and Lam (2014) used a hybrid of K-means and support vector machine (SVM) algorithms to extract useful features and diagnose breast cancer. The K-SVM obtained patterns of malignant and benign tumors while reducing computation time significantly without losing diagnosis accuracy. Zheng et al. (2015) applied machine learning algorithms and metaheuristics to predict hospital readmissions. The study applied neural network (NN), random forest (RF), and a hybrid of particle swarm optimization and SVM, where SVM was tuned by PSO. The proposed model (PSO-SVM) outperformed the other models and achieved an accuracy of 78.40%. In Agrawal et al., 2011 used

classifier and ensemble schemes such as SVM, NN, J48 decision tree (DT), RF, LogitBoost, decision stump, random subspace, reduced error pruning tree, alternating DT, and voting to predict risk or mortality for lung cancer patients. The ensemble voting of five DT-based classifiers and ensembles resulted in the highest accuracy. They also developed an online lung cancer outcome calculator based on the results of their study from the predictors and risk factors found. Tan and Gilbert (2003) applied ensemble learning, bagged and boosted DTs namely, and observed that they perform better than single DTs (C4.5). They applied their work on seven gene expression datasets for cancer classification: acute lymphoblastic leukemia, breast cancer outcome, central nervous system, colon tumor, lung cancer, prostate cancer, and prostate cancer outcome.

Bagging and boosting were also applied by Sun, 2002 through CART to predict pitch accents. They used the acoustic features individually for prediction, text features individually, and both acoustic and text features. They concluded that while using both types of features, ensemble learning with CART-AdaBoost achieved the highest accuracy of 87.17%, while CART-bagging with both features were at an accuracy of 86.89%, and 84.26% through using CART alone. Tsoumakas and Vlahavas (2002) presented an alternative to the stacking method to combine classifiers that holds a high accuracy and a low computational complexity. Their model learns by averaging the base learners and overcomes problems caused by the stacking method when scaling up, especially when there are several classes in a dataset are involved, while keeping the advantage of modeling complex classifier ensemble behavior. A study conducted by Liu and Yao in 1999 presented negative correlation learning for NN ensembles. This method encourages different individual NNs in the ensemble to learn different parts of the training data, which makes the ensemble learn more. The NNs are trained simultaneously through correlation penalty terms in their error functions such that their errors are negatively correlated. Xia, Liu, Da, and Xie (2018) proposed the method bstacking; a novel heterogeneous ensemble credit model that combines the stacking and bagging methods. Their research focused on pool generation, selection of base learners, and trainable fuser. Their base learners included four types of classifiers, SVM, Gaussian process, RF, and extreme gradient boosting (XGBoost). They compared their ensemble with benchmark models on four credit datasets and found that their method is superior. Zięba, Tomczak, and Tomczak (2016) proposed a novel approach for bankruptcy prediction that utilizes XGBoost for learning an ensemble of DTs and applied it to Polish companies. Their method proved to be significantly better than existing benchmark classifiers. They also introduced the concept of synthetic features that improved the accuracy of predictions.

2.2. GRNN Oracle

Masters, Land, and Maniccam (1998) proposed the GRNN Oracle by modifying the original GRNN. They had emphasized that the results obtained by combining the output of several classifiers are generally superior to the results obtained by using one classifier. Land, Masters, and Lo (2000) applied the GRNN Oracle to maximize the predictive power of mammographic screening data for breast cancer in which the oracle intelligently combined the output of four models trained to classify breast lesions as benign or malignant. The models included are evolutionary programming (EP), probabilistic neural network (PNN), NN, and linear discriminant analysis (LDA). The database consisted of 500 cases. Individually, the EP had an accuracy of 73.2% and an AUC of 80.7%, PNN had an accuracy of 77.2% and AUC of 82.43%, NN at 74.4% for accuracy and 85.64% for AUC, and LDA with an accuracy of 76.8% and AUC of 85.23%. Also, they averaged their performance by a simple technique where their combined accuracy was 76.4% and their AUC was 85.9%. However, the accuracy and AUC of the GRNN Oracle created were at 76.4% and 87.70%, respectively. Campbell, Land, Margolis, Mathur, and Schaffer (2013) applied the GRNN Oracle to predict colon cancer recurrence in less than five years from gene microarrays. They fed 38 classifiers into the oracle to combine their predictions by using error and weight formulas. They resulted in an AUC for the validation dataset and the training dataset of 76% and 95%, respectively. This was an improvement to their previous study that used the voting-majority ensemble method in which it resulted in an AUC for the validation dataset and the training dataset at 73% and 95%, respectively. Xiang et al. (2016) also applied the oracle for the rapid detection and classification of food pathogens. The pathogens needed to classify were: Escherichia coli (ATCC#25922), E. coli (ATCC#11775), and Staphylococcus epidermidis (ATCC#12228). By using the differences in metabolic rates between the pathogens, they combined the predictions of PNN, SVM radial basis function (RBF), SVM 2-Order Poly, and SVM 3-Order Poly. The PNN had the highest accuracy among the individual classifiers with 84.4%, however, the GRNN Oracle achieved the highest accuracy at 85.4%.

2.3. Literature gap and motivating factors

The traditional GRNN Oracle has not received much attention in the literature as only very few researchers have published work on the algorithm, even with its desirable results. The way the oracle classifies an unknown observation in which it assigns a certain weight to each one of its classifiers based on their overall error contribution, makes the oracle unique.

To the best of the authors' knowledge, the recursion concept has not been fully explored in machine learning models and applications. The research contribution of this paper is the integration of the recursion concept with the GRNN Oracle. Because of the traditional oracle's formation, this study implements recursion, by having one oracle inside another, and tests this structure's ability on further enhancing the performance of the GRNN Oracle.

Many real-life problems that apply machine learning models to automate classifying unknown observations are problems that require accurate predictions. Tasks such as diagnosing diseases entail precision to avoid serious issues such as false positives and false negatives which could potentially lead to problems such as lawsuits or even deaths. When an organization applies machine learning to create a prediction model, commonly used classifiers are generally used such as logistic regression, SVM, or NN. Because of their inadequate performance, stronger classifiers are needed. Since the traditional oracle is unbiased to any classifier it encompasses and its results are encouraging, it has been enhanced using the recursion concept, and this new enhancement has been named the R-GRNN Oracle.

3. Research methodology

The methodology adopted in this study involves several steps and phases, as seen in Fig. 1. The proposed methodology represents a broad and general view on the creation of the R-GRNN Oracle. As shown in this high-level research framework, the proposed methodology is divided into four phases: preprocessing, individual classifiers, GRNN Oracle development, and Recursive GRNN Oracle design and implementation. The first three phases describe the process and steps taken to create the traditional GRNN Oracle. The novelty of the proposed work lies in the last phase, the Recursive GRNN Oracle design and implementation phase. This phase makes parallel use of the previous phases to create one oracle within another.

To demonstrate the effectiveness of the proposed model, the R-GRNN Oracle has been tested on several datasets. It is

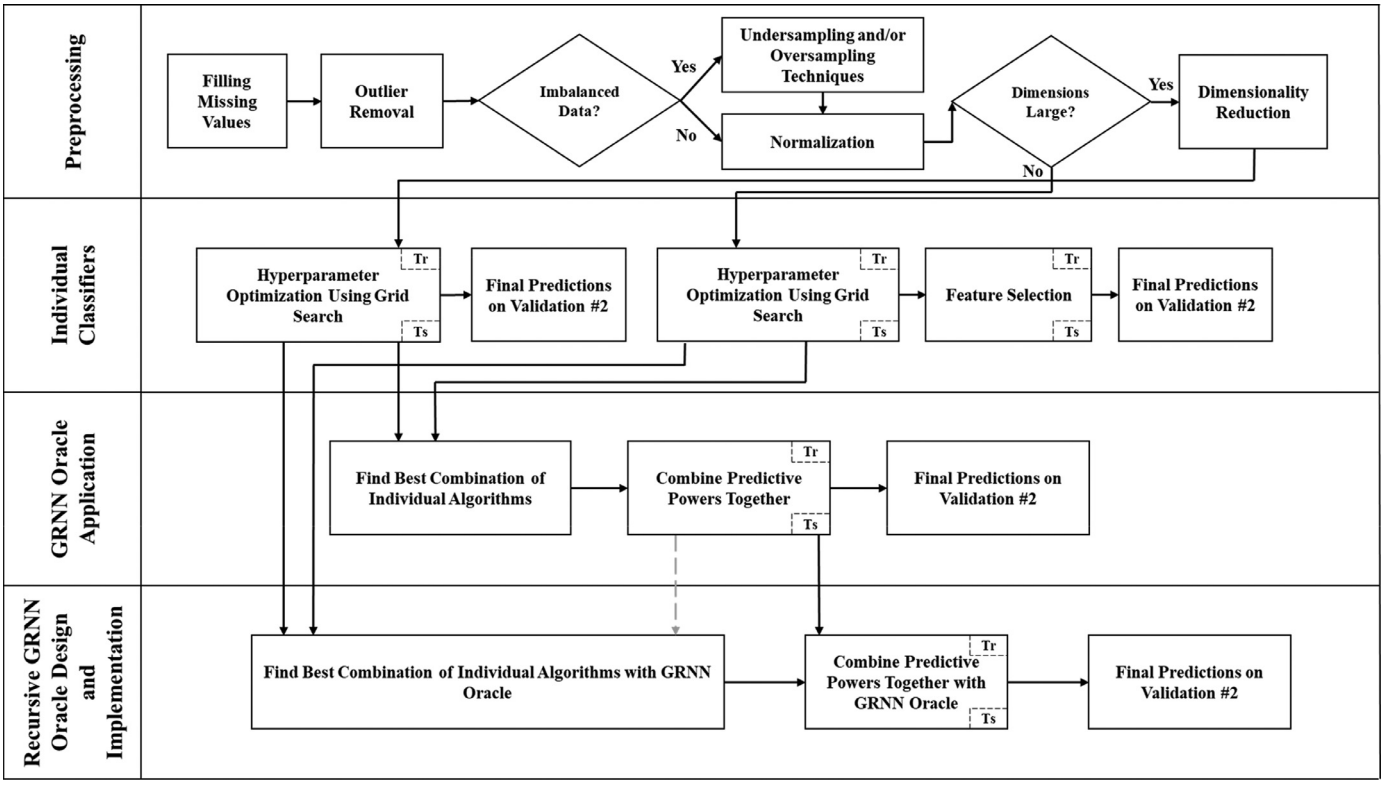


Fig. 1. An illustration of the overall research methodology (Tr: training; Ts: testing).

noteworthy that not all steps in the preprocessing and individual classifiers phases were involved in every dataset as each has its own properties. Therefore, each dataset may need unique preprocessing and preparation to apply the oracles. Six individual classifiers were used in this research: support vector machine (SVM), multilayer perceptron (MLP), probabilistic neural network (PNN), Gaussian Naïve Bayes (GNB), k-nearest neighbor (KNN), and random forest (RF), in which some were used to create the GRNN Oracle, and some were combined with the first oracle to create the R-GRNN Oracle. The programming language used for this study was Python 2.7 and the hardware specifications were Intel® Core™ i7-6700HQ CPU @ 2.60 GHz with 16.0 GB RAM.

The preprocessing phase includes cleaning the dataset by examining the data for missing values and removing any that could affect the model's accuracy. Datasets with imbalanced classes, where one class is more frequent than the other, were approached based on how severe the disparity is. If the disproportion was not considered severe, undersampling was used, but if the disproportion was considered large, SMOTE and TOMEK methods were applied. While undersampling removes observations from the majority class at random to obtain a balanced or near-balanced dataset, SMOTE (synthetic minority oversampling technique) is an oversampling technique where synthetic observations are created for the minority class, and TOMEK is a guided undersampling technique. Data were then normalized to a range between 0 and 1, where 0 indicates the lowest value in a specific feature and 1 represents the highest. This allows fair and equal weights for the features that create the model. The formula of normalization is given in Formula 1 where v_j is the j-th value in feature V_i that needs to be normalized, $\bar{v}_j$ is the new, normalized j-th value in feature V_i, $\min V_i$ is the minimum value in the set of values in feature V_i, and $\max V_i$ is the maximum value in feature V_i. As for the dimensions of a dataset, a phenomenon known as the curse of dimensionality means the more features in a dataset, the more dimensions it has and more complex it gets. Machine learning algorithms tend to perform poorly on high-dimensional data so there is a need to keep the number of dimensions low. The Hughes effect (Hughes, 1968) states that the predictive abilities of an algorithm decrease as the number of dimensions increase. Since many machine learning algorithms rely on similarity-based reasoning (Domingos, 2012) such as measuring the distances between observations using vectors, the high number of features impact their performance greatly, including prediction accuracy, effectiveness, and computational time (Chu et al., 2007).

This paper uses the technique principal component analysis (PCA) to reduce the number of features by creating a new set of orthogonal variables called principal components. The goal of PCA is to explain the maximum amount of variance with the fewest number of principal components. The drawback of applying PCA is the loss of information while compressing the data. The amount of variance explained by the principal components is based on the amount of data information left after the data was compressed. There is a tradeoff between reducing dimensions and losing information; the more dimensions reduced, the more information representing the data is lost.

$$\bar{v}_j = (v_j - \min V_i)/(\max V_i - \min V_i) \quad (1)$$

In the individual classifiers phase, it regards acquiring final predictions for each classifier performed solely. If PCA had been carried out, hyperparameter optimization for each classifier is performed through grid search. However, if the number of features could be handled without PCA, a feature selection method was applied, and with grid search, the hyperparameters of each classifier were optimized. With these optimized classifiers, final predictions were obtained. The optimized classifiers were tested on a left-out validation data subset that was not used for the training nor testing steps of any classifier to avoid overfitting.

The GRNN Oracle application phase includes creating the oracle, as described earlier, through some chosen classifiers. The proposed

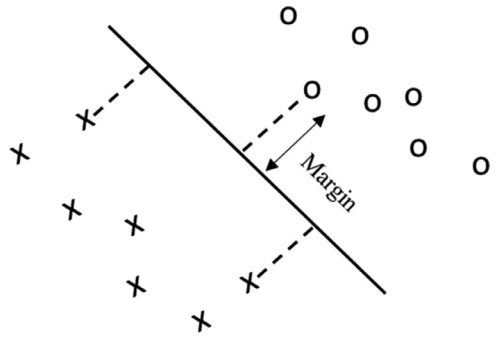


Fig. 2. A simple linear SVM.

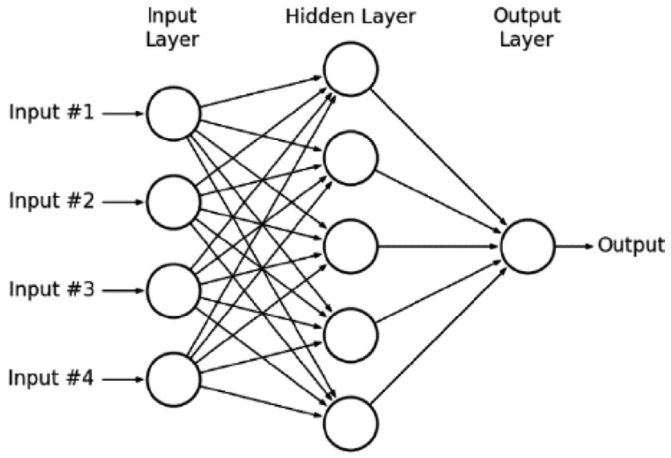


Fig. 3. An MLP NN with one hidden layer (Mohamed, Negm, Zahran, & Saavedra, 2015).

methodology behind which classifiers are chosen is primarily based on high accuracies and high AUCs. After the combination is chosen, they are encased and the GRNN Oracle is created. It is tested on the same left-out validation data subset as the previous phase to ensure a fair comparison.

The R-GRNN Oracle design and implementation phase is built with the same steps as the previous phase but instead, one of the classifiers that will be enveloped is the traditional oracle in which it is chosen with other compatible classifiers. This new combination is called the R-GRNN Oracle. It produces a superior predictive capability, as shown by tests on the same left-out validation data subset.

3.1. Classifiers and hyperparameter optimization

The classifiers used in this study are SVM, MLP, PNN, GNB, KNN, RF, and the traditional GRNN Oracle. Some of the hyperparameters for some of the classifiers were optimized using grid search, which is also discussed in this section. The following subsections introduce each algorithm used and briefly describe the way they work.

Support Vector Machine (SVM): The original SVM was introduced by Vapnik and Chervonenkis in 1963; however, in Boser et al., 1992 suggested a way to create nonlinear classifiers by introducing the kernel trick which produces higher dimensions to better divide the classes. SVMs are supervised machine learning algorithms that are used for classification and regression tasks. They are widely used and are based on statistical learning theory (Cholette, Borghesani, Di Gialleonardo, & Braghin, 2017). SVM maximizes the margin between classes in the feature space and classifies new samples by building a mechanism to separate data into different categories by a n-dimensional hyperplane that computes from a given training dataset (Al-Yaseen, Othman, & Nazri, 2017) as seen in Fig. 2, where support vectors are points in the dataset that help create and locate the hyperplane. SVMs help solve many problems including text and hypertext categorization, image classification and recognition, and they have been applied in biology and other sciences.

Multilayer Perceptron (MLP): MLP is a feedforward NN that is a modification of the standard linear perceptron and can solve problems that are not linearly separable. It is the most frequently used NN (Hossain, Ong, Ismail, & Khoo, 2017) and is widely-used for classification, regression, recognition, prediction, and approximation tasks. It consists of an input layer, a hidden layer(s), and an output layer. MLP uses a supervised machine learning technique called backpropagation for training and adjusting the weights of the model. The term perceptron describes a single-layer network of binary threshold neurons. Fig. 3 illustrates an example of a NN with one hidden layer with five hidden nodes.

Probabilistic Neural Network (PNN): PNN is a feedforward NN which is used in classification and recognition problems. It is a classifier version, which combines the Baye's strategy for decision-making with a non-parametric estimator for obtaining the probability density function (PDF) (Karthikeyan, Gopal, & Venkatesh, 2008). Every such PDF is estimated through a kernel density estimation technique that is known as the Parzen method (Berno et al., 2003). It consists of three layers: an input layer, a hidden layer, and an output layer.

Gaussian Naïve Bayes (GNB): GNB is a widely used supervised learning algorithm which uses Bayes theorem as its framework for classification (Griffis, Allendorfer, & Szaflarski, 2016) and has strong independence assumptions between the features. It assigns the label of the class that maximizes the posterior probability of each observation, under the assumption that the voxel contributions are conditionally independent and follow a Gaussian distribution (Ontivero-Ortega, Lage-Castellanos, Valente, Goebel, & Valdes-Sosa, 2017). Parameter estimation for naïve Bayes models uses the method of maximum likelihood. One advantage of GNB is that it could estimate the parameters necessary for classification by training on a small training set. It is widely used for classification problems because of its simplicity and accurate results (Farid, Zhang, Rahman, Hossain, & Strachan, 2014).

K-Nearest Neighbor (KNN): KNN is a lazy learning method for classification and regression tasks (Zhang et al., 2016) where lazy learning refers to when the target function is approximated locally making it successful for changes in the data. k represents the number of known observations closest to the unknown observation mapped out in the feature space. For classification tasks, the class of the new observation is based on the majority class of k neighbors surrounding it. For regression tasks, the new observation is taken as the average of its k neighbors.

Random Forest (RF): RF is an ensemble created by Ho (1995) that consists of many DTs and exploits the bagging technique to improve the model's performance by decreasing the model's variance without increasing the bias. The trees are created by drawing a subset of the training data through replacement (Belgiu & Drăguţ, 2016). Each tree consists of a random subsample of features (Wang, Lin, & Ho, 2018). RF is used for classification and regression tasks with one of its main advantages is the ability to be robust against DTs habit of overfitting. Because of its simplicity and superior performance, it is widely used in various research fields such as biological and biomedical research.

Grid Search: Grid search is an exhaustive search and a traditional approach to manual hyperparameter tuning in which all possible combinations of the parameters selected are tested. It is guided by a performance metric and typically measured by cross validation on the training set or an evaluation on a validation subset (Hsu, Chang, & Lin, 2003).

Table 1
Computing final prediction ($\hat{y}$) for unknown observation through GRNN Oracle.

For each unknown observation, for each classifier (k)	
I: Train Classifier (k)	Each classifier (k) is trained on the training subset and applied on the testing subset to obtain predictions for the observations
II: Mean Squared Error (MSE)	From the testing data subset, the MSE of each observation (i) for each classifier (k) is calculated through its actual prediction (actual class) and its predicted output (probability of belonging to each class) $error_{i,k} = \sum_{m=1}^{num\_classes} (AP_m - PP_{m,k})^2 / num\_classes$
III: Distances and Weights	The distance between each unknown observation in the validation set and all the known samples in the testing subset is calculated, and each known observation has a particular weight for each unknown observation $D(\vec{x}, \vec{x}_i) = \frac{1}{p} \sum_{j=1}^{p} ((x_j - x_{ij})/\sigma_j)^2$ $weight_i = e^{-D(\vec{x}, \vec{x}_i)}$
IV: Predicted Squared Error	For each unknown observation, for each classifier (k), the predicted squared error is attained through the MSE and weight of each known observation $error_k(\vec{x}) = (\sum_{i=1}^{n} error_{i,k} * weight_i) / \sum_{i=1}^{n} weight_i$
V: Classifier's Trust (Weight)	Each classifier (k) has an amount of trust for the final prediction of the unknown observation where the higher the weight, the more likely the classifier can output an accurate prediction $w_k = (1/error_k)/(\sum_{l=1}^{L} 1/error_k)$ Where: $\sum_{k=1}^{L} w_k = 1$
VI: Final Prediction	Through the amount of error each classifier (k) contributes, their trust/weight is multiplied by the unknown observation's prediction, and summed up to form the final prediction for that particular unknown observation $\hat{y} = \sum_{k=1}^{L} w_k * q_k$

Table 2
List of mathematical notations and their descriptions.

	Description
$error_{i,k}$	Mean squared error of a known observation (i) from classifier (k)
$num\_classes$	The total number of classes (two in binary classification)
AP_m	The actual probability of the known observation (i) for being in class (m)
$PP_{m,k}$	The predicted probability of being class (m) from classifier (k)
$\vec{x}$	The vector of features belonging to the unknown observation, [feature 1, feature 2, ..., feature p]
p	The total number of features
$\vec{x}_i$	The feature vector for the known observation
x_j	The j-th feature of the unknown observation
x_{ij}	The j-th feature of the known observation
σ_j	An adjustable sigma parameter for the j-th feature
w_k	The weight (trust) of classifier (k) on the prediction of the unknown observation
L	The total number of classifiers
l	Indicates classifier (l) from all classifiers (L)
$\hat{y}$	The prediction of the unknown observation outputted by the GRNN Oracle represented as a class membership vector
q_k	The predicted class membership vector for the unknown observation given by classifier (k)

GRNN Oracle: The GRNN Oracle combines the predictive powers of several machine learning classifiers that were trained independently to form one superior prediction through the amount of error each classifier contributes. For example, if observations A, B, and C were correctly classified by classifier X and misclassified by classifier Z, and observations D, E, and F were classified correctly by classifier Z and misclassified by classifier X, the GRNN Oracle may improve the predictions by combining the predictions of both classifiers X and Z. This also means that if the predictions of both classifiers are identical, the GRNN Oracle's accuracy would not be significantly different as there is not much difference in the information or classification provided by the classifiers (Li, 2014). Table 1 shows the logic behind the GRNN Oracle through a pseudocode in which it explains the oracle's prediction for one unknown observation. It shows the high-level formulation of the GRNN Oracle developed by Masters et al. (1998). A subset of the data (its majority) is used for training and testing the model to gain the amount of error each classifier has from the testing set's predictions – as the actual predictions exist. Another subset of data, the validation set, is set aside for the GRNN Oracle to perform on. The reader is referred to Land et al. (2000), Masters et al. (1998), and Campbell et al. (2014), for more detailed explanation.

Table 2 shows the list of the mathematical notations concerning Table 1 and their descriptions.

In this study, the performance of each individual classifier, with respect to its accuracy, AUC, sensitivity, and specificity, directs the choosing of which classifiers should be fed into the GRNN Oracle and which should be left out. The best combination of classifiers that complement each other and individually provides a strong performance regarding a dataset are chosen to create either or both oracles. Fig. 4 illustrates the formulation of the oracle (presented in Table 1) through a flowchart.

3.2. Strength and weaknesses of the classifiers

Table 3 lists the significant and fairly standard strengths and weaknesses of the classifiers: SVM, MLP, PNN, GNB, KNN, RF, and the GRNN Oracle. For more details, the reader is referred to books such as "Machine Learning: An Algorithmic Perspective" by Marsland (2011), "Foundations of Machine Learning" by Mohri, Rostamizadeh, and Talwalkar (2012), and "Machine Learning: Algorithms and Applications" by Mohammed, Khan, and Bashier (2016).

3.3. Recursive GRNN Oracle

The best combination of classifiers that were trained and tested individually and independently with respect to accuracy, AUC, sensitivity, and specificity was used to make the first oracle. By having predictions outputted from the oracle, it now acts as any other machine learning classifier would. The best combination of classifiers that would enhance the performance of the first GRNN Oracle

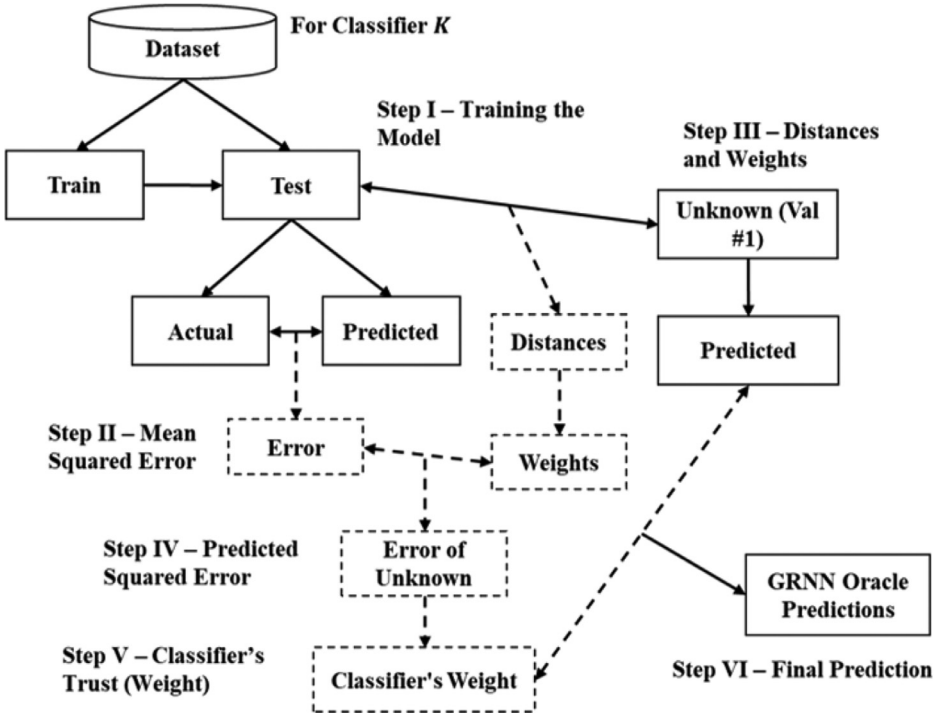


Fig. 4. A detailed flowchart of the GRNN Oracle binary classification formulation.

Table 3
Strength and weaknesses of the classifiers used in this study.

	Strengths	Weaknesses
SVM	• Uses a subset of observations (support vectors) to define the margin between classes. • Effective in high-dimensional spaces.	• Less effective as the data size grows. • Sensitive and less effective on noisy data.
MLP	• Handles nonlinearly separable data. •Effective with large datasets.	• It has a "black box" nature. • Requires large amounts of data for training. • Training is computationally expensive (slow).
PNN	• Handles nonlinearly separable data. • Effective with large datasets. • Trains faster than an MLP. • Generates predicted target probability scores.	• Its "black box" nature. • Requires large amounts of data for training. • Training is computationally expensive (slow). • Predicts slower than an MLP.
GNB	• Ease of implementation. • Fast training time. • Works well with small training sets. • Effective when assumption of feature independence holds.	• Works under the assumption of feature independence. • Unable to make a prediction if a testing label was not observed during training (the zero-frequency problem).
KNN	• Simple and intuitive. • Ease of implementation. • Lazy learner (the target function is approximated locally making it successful for changes in the data). • Handles multi-class problems well.	• Classification is highly dependent on the number of neighbors (k). • Sensitive and less effective on noisy data. • Does not work well with high-dimensional datasets. • Less effective on imbalanced data.
RF	• Effective with large datasets. • Robust against overfitting. • Handles missing data well. • Does not require data scaling.	• Complex and time-consuming as the number of trees grow. • Hard to interpret.
GRNN O.	• High accuracy. • Effective with large datasets. • Unbiased classifier that distributes prediction weight based on error contribution.	• Training is computationally expensive (slow).

is selected and then, this selected combination, including the first oracle, creates the second oracle, the R-GRNN Oracle. The combination of classifiers that are used to enhance the first oracle is chosen based on the overall performance of each classifier; classifiers that have high accuracies and AUCs are able to help boost the performance of the first oracle. This is also how the traditional oracle works; the classifiers chosen to create the oracle are chosen based on which classifiers perform best. Another consideration that is taken, which is dependent on the results of the individual classifiers, is that if the inner oracle has a lower sensitivity than its specificity. For example, it would be favorable to combine it with a classifier that has high sensitivity to help even out the difference in both metrics (this concept also applies to when the inner oracle has a lower specificity than its sensitivity).

The accuracy, AUC, sensitivity, specificity, precision, and F1-score of the R-GRNN Oracle's final predictions are taken, along with the same performance metrics of the inner GRNN Oracle and the individual classifiers for the final comparison. Fig. 5 presents

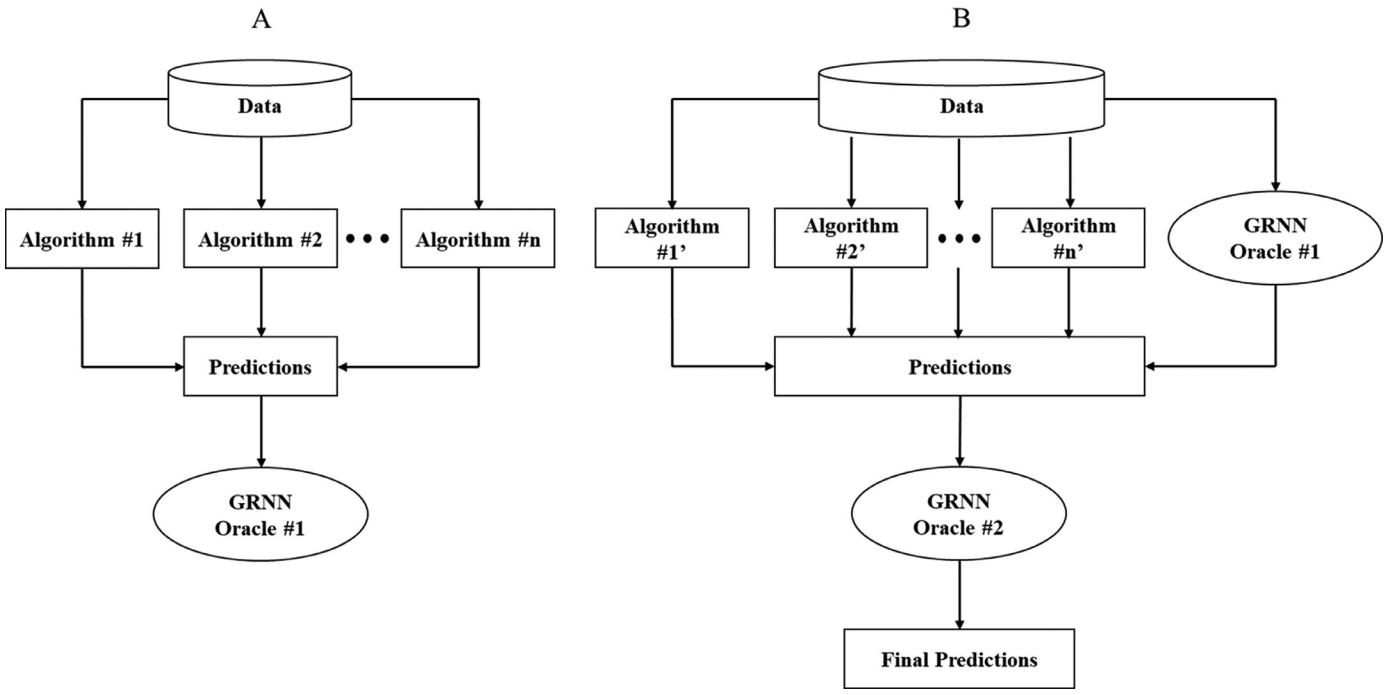


Fig. 5. An overview of the first (traditional) GRNN Oracle (A) and the second (proposed) GRNN Oracle, the R-GRNN Oracle (B).

the formation of the traditional oracle and the proposed recursive oracle.

3.4. Related work to the spambase dataset

Many studies have been carried out on the Spambase dataset – and of these studies several validation methods have been used. As this study applies the 10-fold cross validation method to validate the performance of the model, only the studies that have applied k-fold cross validation were chosen as the baseline to evaluate the recursive model to ensure a fair comparison of the performance metrics. Readers interested in published work that did not use k-fold cross validation are referred to Caruana, Li, and Liu (2013), Ferreira and Figueiredo (2012), and Tabakhi, Moradi, and Akhlaghian (2014). The k-fold cross validation method produces a lower accuracy rate than other methods; however, using it to evaluate a model minimizes biases and gives an accurate and consistent view of performance. Testing the performance of a certain model on the same subset of data that the model was trained on would most likely generate a high accuracy because of overfitting. To avoid this problem, the model should be tested on data not involved in the training phase. The k-fold cross validation works through splitting the dataset into a k number of same-sized partitions (folds) in which training is done on k-1 subsets and tested on the left-out k and is repeated k times until each fold has been trained k-1 times and tested on once. The accuracy (or error) is computed by taking the average accuracy (or error) of all the folds.

In terms of benchmark studies that have applied various classifiers on the Spambase dataset, Covões, Hruschka, de Castro, and Santos (2009) proposed a filter-based method for feature selection, the simplified silhouette filter (SSF), and applied the method to ten datasets including the Spambase dataset. They performed a 10-fold cross validation for the GNB and KNN algorithms on the datasets. With their proposed feature selection method, the GNB achieved an accuracy of 58.60% while KNN was at 65.01%. Nonparametric statistical techniques have been used by García, Fernández, Luengo, and Herrera (2010) in the comparison of several machine learning algorithms where a control treatment is compared to other treatments. The positive definite fuzzy classifier (PDFC) had the highest accuracy of 92.4% with 10-fold cross validation. A study conducted by Mitra, Murthy, and Pal (2002) introduced a new feature similarity measure for feature selection called maximum information compression index (MICI) and applied it to various datasets through GNB and KNN. Their resulted accuracy for the Spambase dataset was 90.01% for KNN and 88.19% for GNB. Guan et al. (2018) proposed a noise filtering technique, named enhanced soft majority voting by exploiting unlabeled data (ES-MVU), which is an ensemble that adopts the soft majority voting technique. On the Spambase dataset, they reached an accuracy of 89.23% using their proposed method; however, the highest accuracy they obtained was using the consensus filter (CF) at 90.34%.

Wei et al. (2017) also used unlabeled data for noise filtering. The proposed the multiple filtering with the aid of unlabeled data using confidence measurement (MFUDCM). Regarding the Spambase dataset, their applied method gave them an accuracy of 89.23%, however, as the previous study mentioned, they applied CF which resulted in the same accuracy of the previous study at 90.34%. Unler and Murat (2010) proposed a modified particle swarm optimization algorithm for feature selection and compared it with tabu search and scatter search algorithms using logistic regression and applied them on several datasets. Their proposed modification on the testing set for the Spambase dataset was the highest at 90.20% at $i=8$, where i is the number of features selected from the original 57 independent variables. Sharma and Sahni (2011) applied four DT algorithms to the Spambase dataset. They used: ID3, J48, CART, and ADTree. With 10-fold cross validation, they achieved accuracies of 92.76%, which was the highest, with the J48 algorithm, 92.63% with CART, 90.91% with ADTree, and ID3, with the lowest accuracy, was at 89.11%. Panagopoulos, Pappu, Xanthopoulos, and Pardalos (2016) applied a new binary classification method called constrained subspace classifier (CSC) for high dimensionality datasets. They tested their methodology on six datasets including the Spambase dataset and reached an accuracy of 87.90%, however, SVM obtained a higher accuracy at 91.00%.

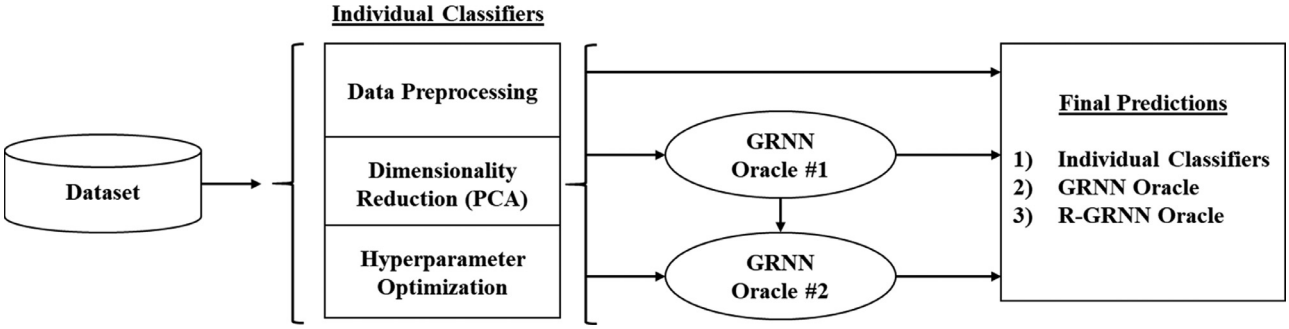


Fig. 6. The methodology followed for the Spambase dataset through a high-level flowchart.

Table 4
The Spambase dataset feature description.

	Description	Details
1–48	Word Frequency	The frequency with which the word appears (measured in percentage)
49–54	Character Frequency	The frequency with which the character appears (measured in percentage)
55–57	Capital Run Length	The length of consecutive capital letters
58	Spam/Non-Spam	Whether the email was classified as spam or non-spam

Table 5
Mean run time and standard deviation for all classifiers in seconds (using PCA).

	Mean Run Time	Standard Deviation
SVM	2.896	0.110
MLP	7.087	0.347
PNN	1.274	0.061
GNB	0.055	0.006
KNN	0.674	0.032
RF	1.512	0.030
GRNN O.	51.665	0.689
R-GRNN O.	716.951	11.501

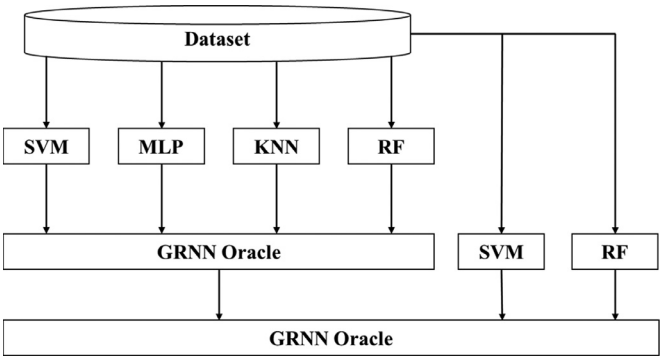


Fig. 7. Overview of the Recursive GRNN Oracle on the Spambase dataset.

Wahbeh, Al-Radaideh, Al-Kabi, and Al-Shawakfa (2011) conducted a comparison study between four data mining programs: WEKA (Waikato Environment for Knowledge Analysis), Orange, Tanagra, and KNIME (Konstanz Information Miner). They applied nine publicly available datasets on the four programs and compared their accuracies. They used the holdout cross validation method, in which they gave a 66% split for the training set and utilized the 10-fold cross validation method. Their 10-fold cross validation regarding the Spambase dataset had the highest accuracy of 92.98% using the C4.5 classifier on the WEKA program. They concluded that Weka had the best results regarding applicability and accuracies when compared to Orange, Tanagra, and KNIME. Wang and Witten (2002) obtained an 88.70% accuracy by using the maximum likelihood estimator (MLE) but the method they suggested in their study yielded a lower accuracy of 86.90%. The proposed method adopts the MLE's well-known asymptomatic normality property to transform the original parameters into dummy ones and reduce the model's dimensions. Lu, Wang, and Yoon (2018) proposed a genetic algorithm-based online gradient boosting (GAOGB) and applied it to three datasets including the Spambase dataset. They obtained an accuracy of 88.79%.

4. Experimental analysis and results

To illustrate the performance of the proposed algorithm, the R-GRNN Oracle was applied on four datasets: Spambase, Human Resources, Bank Marketing, and MGUS. The detailed application of the recursive model was applied to the Spambase dataset; a classification dataset of spam and non-spam emails. This dataset is used as a detailed example to test the proposed algorithm to compare its performance with the performances of other traditional approaches in several metrics such as classification accuracy. The performance of the R-GRNN Oracle on the other datasets has been summarized. To reach a more accurate evaluation for all the classifiers, 10-fold cross validation was used.

4.1. Dataset description

The Spambase dataset consists of 4601 observations, each classifying an email as spam or not based on 57 independent variables (features) that describe the content of a specific email (Table 4). The methodology followed in this study for the Spambase dataset is shown in Fig. 6.

4.2. Data preprocessing

Outliers were removed to increase the robustness and accuracy of the proposed model and undersampling was carried out as a data balancing technique. All independent variables have been normalized and scaled down to the range of [0,1], this helps in creating a more accurate model because of having all features weigh similarly before the model creation. The resulted dataset contained 3891 emails.

Table 6
Performance metrics for all classifiers on the Spambase dataset.

	Accuracy	AUC	Sensitivity	Specificity	Precision	F1-Score
SVM	0.9175	0.9682	0.8984	0.9337	0.9198	0.9090
MLP	0.9071	0.9636	0.8873	0.9239	0.9091	0.8974
PNN	0.8814	0.9378	0.8196	0.9337	0.9128	0.8637
GNB	0.8493	0.9128	0.8631	0.8376	0.8182	0.8399
KNN	0.9046	0.9499	0.8676	0.9358	0.9197	0.8929
RF	0.9129	0.9657	0.8974	0.9260	0.9113	0.9043
GRNN O.	0.9194	0.9736	0.9043	0.9321	0.9189	0.9114
R-GRNN O.	**0.9324**	**0.9799**	**0.9186**	**0.9440**	**0.9328**	**0.9257**

Table 7
The performance of the R-GRNN Oracle on other datasets.

		Accuracy	AUC	Sensitivity	Specificity	Precision	F1-Score
Human Resources	SVM	0.9337	0.9624	0.9301	0.9372	0.9357	0.9329
	MLP	0.9429	0.9722	0.9237	0.9618	0.9597	0.9413
	PNN	0.8975	0.9212	0.9155	0.8798	0.8821	0.8985
	GNB	0.6644	0.8196	0.8662	0.4663	0.6146	0.7190
	KNN	0.8990	0.9420	0.9073	0.8910	0.8911	0.8991
	RF	0.9604	0.9804	0.9365	0.9838	0.9827	0.9590
	GRNN O.	0.9535	0.9803	0.9302	0.9763	0.9748	0.9520
	R-GRNN O.	**0.9685**	**0.9919**	**0.9506**	**0.9860**	**0.9853**	**0.9676**
Bank Marketing	SVM	0.8020	0.8767	0.8061	0.7986	0.7693	0.7872
	MLP	0.7911	0.8704	0.8144	0.7717	0.7499	0.7786
	PNN	0.6972	0.7178	0.6655	0.7236	0.6675	0.6664
	GNB	0.7424	0.8130	0.7077	0.7712	0.7206	0.7141
	KNN	0.7558	0.8175	0.6872	**0.8129**	0.7540	0.7189
	RF	0.7989	0.8766	0.8233	0.7787	0.7562	0.7882
	GRNN O.	0.8031	0.8834	0.8185	0.7903	0.7653	0.7906
	R-GRNN O.	**0.8209**	**0.8929**	**0.8344**	0.8097	**0.7852**	**0.8090**
MGUS	SVM	0.8446	0.9380	0.8264	0.8633	0.8615	0.8435
	MLP	0.8356	0.9426	0.8556	0.8150	0.8513	0.8414
	PNN	0.9079	0.9709	**0.9940**	0.8194	0.8499	0.9163
	GNB	0.7479	0.8313	0.8610	0.6316	0.7061	0.7759
	KNN	0.8682	0.9437	0.9648	0.7688	0.8110	0.8812
	RF	0.9227	0.9806	0.9367	**0.9083**	**0.9132**	0.9247
	GRNN O.	0.9113	0.9781	0.9561	0.8652	0.8798	0.9162
	R-GRNN O.	**0.9364**	**0.9860**	0.9729	0.8988	0.9083	**0.9394**

4.3. Dimensionality reduction and hyperparameter optimization

To reduce the number of dimensions while maintaining most of the dataset's information, PCA was applied to the 57 independent variables. At an explained variance of 82%, 26 principal components were retained, where the user-set threshold was set to no less than 80%. Regarding computational-time before and after applying PCA with 3891 observations, 15 runs were executed for each method and the average run time was recorded. After the use of PCA, the average time noted was 11 min and 57 s, with a standard deviation of 11.5 s. When executing the algorithm without applying PCA, the average run time was 18 min and 39 s, with a standard deviation of seven seconds. Table 5 displays the mean run time and their standard deviation for all classifiers in seconds after the use of PCA. It clearly shows a significant difference in the R-GRNN Oracle's computational-time in comparison to the rest of the classifiers involved including the traditional GRNN Oracle. Unfortunately, this is a major drawback for the proposed model. The hardware specifications used for running the algorithms were Intel® Core™ i7-6700HQ CPU @ 2.60 GHz with 16.0 GB RAM.

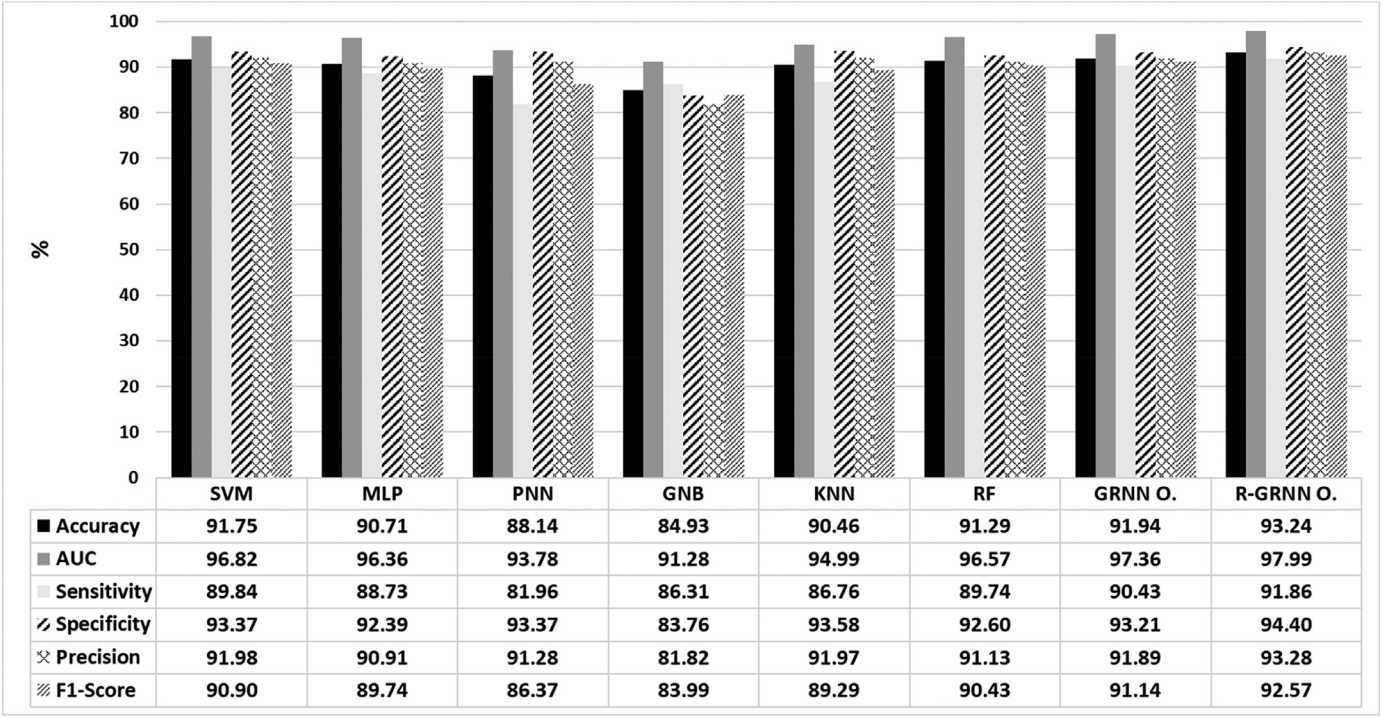
To guarantee better accuracy, the hyperparameters for SVM, MLP, KNN, and RF were optimized using grid search through 10-fold cross validation. The optimized hyperparameters for SVM were the regularization hyperparameter (c) and the Gaussian kernel hyperparameter (γ), with the kernel set to RBF. The MLP's optimized hyperparameters included the number of hidden layers, the number of nodes in each hidden layer, the learning rate, and the momentum, where the activation function was set to ReLU (Rectified Linear Unit). And the hyperparameters optimized for KNN and RF were the number of neighbors (k) and the number of decision trees, respectively. Hyperparameter optimization was not performed on GNB due to its non-existence, nor was it applied to PNN due to its insignificance.

4.4. Recursive GRNN Oracle

For the first GRNN Oracle (the inner oracle), the classifiers fed into it were SVM, MLP, KNN, and RF. The oracle performed slightly better than the performance of each classifier modeled separately. The accuracy and AUC for all the classifiers are as follows, respectively: SVM: 91.75% and 96.82%; MLP: 90.71% and 96.36%; KNN: 91.29% and 94.99%, and RF: 91.29% and 97.26%. The performance of the first oracle had an accuracy of 91.94%, AUC of 97.36%, sensitivity of 90.43%, and specificity of 93.21%. PNN and GNB were not chosen because of their inferior performances when compared to the others.

For the R-GRNN Oracle, the first GRNN Oracle, which now acts as a classifier with its own predictions, was combined with SVM and RF, because the strategy of choosing a combination was based on high accuracies, AUCs, sensitivities, and specificities. As SVM and RF had better performances than others, they were chosen as a match with the first oracle to create the second oracle. The sensitivity and specificity of SVM, respectively, were 89.84% and 93.37%, and RF was at 89.74% and 92.60% respectively. Fig. 7 illustrates the classifiers feeding into each one of the two oracles.

The first oracle achieved an accuracy of 91.94% which was slightly above the highest of the individual classifiers, RF, which



	SVM	MLP	PNN	GNB	KNN	RF	GRNN O.	R-GRNN O.
Accuracy	91.75	90.71	88.14	84.93	90.46	91.29	91.94	93.24
AUC	96.82	96.36	93.78	91.28	94.99	96.57	97.36	97.99
Sensitivity	89.84	88.73	81.96	86.31	86.76	89.74	90.43	91.86
Specificity	93.37	92.39	93.37	83.76	93.58	92.60	93.21	94.40
Precision	91.98	90.91	91.28	81.82	91.97	91.13	91.89	93.28
F1-Score	90.90	89.74	86.37	83.99	89.29	90.43	91.14	92.57

Fig. 8. Graphical representation of the performance metrics for all classifiers on the Spambase dataset (in percentage).

was at 91.29%. However, the recursive model had the highest in all the metrics (accuracy at 93.24% and AUC at 97.99%). Table 6 and Fig. 8 show the accuracy, AUC, sensitivity, specificity, precision, and F1-score of all the classifiers: six individual classifiers (performing on their own), the GRNN Oracle, and the R-GRNN Oracle. As it can be seen, the R-GRNN Oracle outperformed every other model in every performance metric. The recursive model's AUC was at 97.99% demonstrating a strong capability in distinguishing between the two classes. The sensitivity, specificity, precision, and F1-score were 91.86%, 94.40%, 93.28%, and 92.57%, respectively, where the sensitivity indicates that the model is able to predict 91.86% of actual spam emails as spam, and the specificity indicates that the model is able to predict 94.40% of actual non-spam emails as non-spam.

One-way analysis of variance (ANOVA) was carried out in which it showed statistical and significant differences in the accuracies of all the classifiers involved including the R-GRNN Oracle. A pairwise comparison using *t*-test and the Holm-Bonferroni method at a significance level of 0.05 showed that the R-GRNN Oracle performed statistically better than the rest of the classifiers (p-value < 0.05).

4.5. Additional evaluation of the recursive GRNN Oracle

The recursive model was also applied to other datasets: Human Resources dataset, acquired from Kaggle, a platform for predictive modeling and analytics competitions, Bank Marketing dataset, available on the UCI dataset repository, and MGUS dataset, available through datasets distributed by R language.

The Human Resources dataset holds 14,999 observations and consists of nine independent variables and one dependent variable. The task is to predict whether an employee would leave their work or not based on several factors. The independent variables are: the department the employee worked at, their salary, their satisfaction level, last evaluation score, number of projects they worked on, average monthly hours, time spent at the company, if they had a work accident, and if they were promoted in the last five years. The dependent variable is whether the employee left their work or not.

The Bank Marketing dataset originally has 11,162 observations, however, after preprocessing, the dataset has been reduced to 7842 observations. The dataset consists of 16 independent variables and one dependent variable. The task is to predict if the client will subscribe to a term deposit during a marketing campaign. The independent variables are: client's age, job, marital status, education, whether they have credit in default, balance, whether they have a housing loan, whether they a personal loan, contact communication method, day of the month they were last contacted, last contact month of the year, contact duration, number of contacts performed during the campaign, number of days that passed by after the client was last contacted from a previous campaign, number of contacts performed before this campaign, and the outcome of the previous marketing campaign. The dependent variable is whether the client subscribed to the term deposit or not.

The MGUS dataset has 1384 observations with eight independent variables and one dependent variable. The task is to predict the progression of patients diagnosed with MGUS to multiple myeloma. The independent variables are: the patient's age, sex, hemoglobin level, creatinine level, M-spike (myeloma gamma globulin), time from MGUS until diagnosis of plasma cell myeloma, months from diagnosis to last follow-up or death, and whether a patient died or not. The dependent variable is whether MGUS progressed to multiple myeloma.

Concerning the Human Resources dataset, the R-GRNN outperformed all the other classifiers in all performance metrics, as shown in Table 7. As for the Bank Marketing dataset, the recursive model outperformed the rest in all performance metrics except specificity, where it came in second after KNN. Lastly, for the MGUS dataset, it outperformed the rest in the accuracy, AUC, and F1-score, however, for sensitivity, it came in second best after PNN; for specificity, it came in second best after RF; and for precision, it also came in second best after RF. The results for the proposed model indicate practical significance; in the context of these datasets, predicting if an employee would leave their work would

Table 8
Comparison of the accuracy in the literature with this study.

	Method	Accuracy
This Study	**R-GRNN Oracle**	**93.24%**
Wahbeh et al. (2011)	C4.5 Classifier	92.98%
Sharma and Sahni (2011)	J48 Classifier	92.76%
García et al. (2010)	PDFC	92.40%
Panagopoulos et al. (2016)	SVM	91.00%
Guan et al. (2018)	CF	90.34%
Wei et al. (2017))	CF	90.34%
Unler and Murat (2010)	LR with PSO	90.20%
Mitra et al. (2002)	KNN with MICI	90.01%
Lu et al. (2018)	GAOGB	88.76%
Wang and Witten (2002)	MLE	88.70%
Covões et al. (2009)	KNN with SSF	65.01%

help any company's human resources department plan ahead and accordingly. The recursive model would also help banks by targeting their marketing campaigns to clients who are more susceptible to subscribing to certain products which would evidently help decrease marketing costs. The R-GRNN Oracle can also be extremely valuable and advantageous in the medical field to diagnose diseases and track their progression such as the MGUS dataset example.

4.6. Summary of the results

Tables 6 and 7 show the performance metrics achieved by all classifiers, including the R-GRNN Oracle, on the four datasets used in this study. In the Spambase and Human Resources datasets, the recursive model outperformed the rest of the classifiers in every performance metric. For the dataset Bank Marketing, the proposed model surpassed the rest in all metrics expect for specificity in which it was still able to come in second best. However, for the MGUS dataset, it had the highest accuracy, AUC, and F1-Socre, but came in second in the sensitivity, specificity, and precision metrics. The results on these four datasets are promising; the R-GRNN Oracle has great potential to be considered for use in real-life problems, and since the proposed model displayed remarkable results, the benefits of the recursive structure should be further investigated in machine learning.

5. Conclusion and future work

This study proposed the R-GRNN Oracle classifier as an enhancement to the GRNN Oracle introduced by Masters et al. (1998) and was evaluated on four datasets in which it proved its strong capability. This study introduced the GRNN Oracle's background through literature, and literature regarding classification and ensemble learning and the applied dataset, the Spambase dataset. The classifier proposed in this research, the R-GRNN Oracle, proved its superiority among all the classifiers considered: SVM, MLP, PNN, GNB, KNN, RF, and GRNN Oracle, regarding the Spambase dataset. It proved to be robust as no matter how the data was shuffled (different sets each run), the recursive model would always generate superior predictions. Its detailed application was tested on the Spambase dataset regarding classifying emails to spam and non-spam. The recursive model outperformed the rest in all the evaluation metrics. The R-GRNN Oracle combined the predictive powers of SVM, RF, GRNN Oracle, and the classifiers within the GRNN Oracle: SVM, MLP, KNN, and RF. PNN and GNB performed poorly on the dataset and were excluded from both oracles. Compared to the literature, the R-GRNN Oracle obtained an accuracy of 93.24% using 10-fold cross validation while the highest in the literature studied, to the best of the authors' knowledge, which also used 10-fold cross validation, was at 92.98% reached by Wahbeh et al. in 2011 (Table 8). However, for the other three classification datasets, the R-GRNN Oracle outperformed all the other classifiers in the Human Resources dataset; outperformed the rest in all metrics expect specificity in the Bank Marketing dataset; and outperformed the rest in the accuracy, AUC, and F1-score in the MGUS dataset.

The research contribution of this paper was introducing the recursion concept to the GRNN Oracle to further enhance its performance. Because of the way the traditional oracle works, integrating the recursive structure was possible. The GRNN Oracle works through assigning weights to the classifiers it encompasses based on their predictive abilities; the more error a classifier contributes to the overall error all classifiers give, the less weight it is assigned, hence, the less contribution it has towards the final prediction. This unique way of classification makes the final prediction unbiased, and also, suitable to integrate the recursion concept by treating the whole oracle as any other machine learning classifier, therefore feeding it to a new oracle, creating a recursive structure which has been named as the R-GRNN Oracle.

Insightful implications of the proposed model include its consideration when building a classification prediction model, and possibly designing a decision support system (DSS) that automates its implementation and result interpretation that would be most useful to those who are not experts in data science and machine learning. The recursive model can be applied to any domain, especially domains where model performance metrics are crucial and paramount, such as disease diagnostics in medicine. In addition, machine learning programming languages, such as Python, have algorithms predefined and written by some users to help others use their block of code to execute algorithms in a few lines rather than program the whole algorithm from the ground up. These blocks of codes that are reused by others are often called packages and libraries, and it would be beneficial if one was created for the R-GRNN Oracle.

Limitations regarding this study include the size of the dataset as the R-GRNN Oracle requires five splits of data; the size of the dataset should be large enough to train on an acceptable subset to obtain more accurate results and to avoid overfitting. The leave-one-out cross validation method could be used for smaller datasets, as the previous literature showed, however, the way the recursive model works requires two independent training sets for each oracle and two independent testing sets along with a left-out validation set which calls for the reason that requires the oracle to need a relatively large dataset. Computational-time has also been found to be a major drawback of the recursive model; this is especially the case when dealing with extremely large datasets, whether the large size relates to the number of observations or to the number of features. Another limitation is finding the best combination of classifiers to create each oracle. There are many classifiers that were not considered in this study, which means different combinations may yield even better results. This same problem applies to finding the optimal set of hyperparameters for each classifier. Grid search was used for hyperparameter optimization, but this involves having predetermined values and sets to choose from, which results in the problem of getting stuck in a local minimum regarding the loss or cost function. Because of this reason, grid search has a great disadvantage of not searching the entire search space for all possible solutions, as the actual search space is infinite, and the number of combinations grid search gives is a very small finite number in comparison.

Future work might include applying the proposed model to more datasets including real-life datasets and applying it to non-binary (multi-class) classification to assess its capabilities further. Along with that, applying the recursive model to smaller datasets and using the leave-one-out cross validation method for training, testing, and validation, while making sure there is a left-out data

subset for validation purposes. Another recommendation includes the use of metaheuristics for hyperparameter optimization instead of grid search. A few of the metaheuristics that could be considered are those that work well on continuous problems, such as genetic algorithm, simulated annealing, and particle swarm optimization. A hybrid of metaheuristics would also be interesting to implement to evaluate its performance in comparison with individual metaheuristics.

Author Contribution

Dana Bani-Hani: Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing - original draft, Writing - review & editing

Mohammad Khasawneh: Project administration, Resources, Supervision, Validation, Writing - original draft, Writing - review & editing

Declaration of Competing Interest

None

References

Agrawal, A., Misra, S., Narayanan, R., Polepeddi, L., & Choudhary, A. (2011, August). A lung cancer outcome calculator using ensemble data mining on SEER data. In *Proceedings of the Tenth International Workshop on Data Mining in Bioinformatics* (p. 5). ACM.

Al-Yaseen, W. L., Othman, Z. A., & Nazri, M. Z. A. (2017). Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system. *Expert Systems with Applications, 67*, 296–303.

Belgiu, M., & Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing, 114*, 24–31.

Berno, E., Brambilla, L., Canaparo, R., Casale, F., Costa, M., Pepa, C. D., et al. (2003, July). Application of probabilistic neural networks to population pharmacokineties. In *Neural Networks, 2003. Proceedings of the International Joint Conference on: Vol. 4* (pp. 2637–2642). IEEE.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory* (pp. 144–152). ACM.

Bowles, M. (2015). *Machine learning in Python: essential techniques for predictive analysis*. Hoboken, New Jersey (NJ): John Wiley & Sons.

Campbell, A. S., Land, W. H., Margolis, D., Mathur, R., & Schaffer, D. (2013). Investigating the GRNN Oracle as a method for combining multiple predictive models of colon cancer recurrence from gene microarrays. *Procedia Computer Science, 20*, 374–378.

Caruana, G., Li, M., & Liu, Y. (2013). An ontology enhanced parallel SVM for scalable spam filter training. *Neurocomputing, 108*, 45–57.

Chaurasia, V. (2017). Early prediction of heart diseases using data mining techniques.

Cholette, M. E., Borghesani, P., Di Gialleonardo, E., & Braghin, F. (2017). Using support vector machines for the computationally efficient identification of acceptable design parameters in computer-aided engineering applications. *Expert Systems with Applications, 81*, 39–52.

Chu, C. T., Kim, S. K., Lin, Y. A., Yu, Y., Bradski, G., Olukotun, K., et al. (2007). Map-reduce for machine learning on multicore. In *Advances in Neural Information Processing Systems* (pp. 281–288).

Covões, T. F., Hruschka, E. R., de Castro, L. N., & Santos, Á. M. (2009, June). A cluster-based feature selection approach. In *International Conference on Hybrid Artificial Intelligence Systems* (pp. 169–176). Springer.

Domingos, P. M. (2012). A few useful things to know about machine learning. Communications of the ACM, *55*(10), 78–87.

Esfandiari, N., Babavalian, M. R., Moghadam, A. M. E., & Tabar, V. K. (2014). Knowledge discovery in medicine: Current issue and future trend. *Expert Systems with Applications, 41*(9), 4434–4463.

Farid, D. M., Zhang, L., Rahman, C. M., Hossain, M. A., & Strachan, R. (2014). Hybrid decision tree and naïve Bayes classifiers for multi-class classification tasks. *Expert Systems with Applications, 41*(4), 1937–1946.

Ferreira, A. J., & Figueiredo, M. A. (2012). An unsupervised approach to feature discretization and selection. *Pattern Recognition, 45*(9), 3048–3060.

García, S., Fernández, A., Luengo, J., & Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences, 180*(10), 2044–2064.

Griffis, J. C., Allendorfer, J. B., & Szaflarski, J. P. (2016). Voxel-based Gaussian naïve Bayes classification of ischemic stroke lesions in individual T1-weighted MRI scans. *Journal of Neuroscience Methods, 257*, 97–108.

Guan, D., Wei, H., Yuan, W., Han, G., Tian, Y., Al-Dhelaan, M., et al. (2018). Improving label noise filtering by exploiting unlabeled data. *IEEE Access, 6*, 11154–11165.

Ho, T. K. (1995, August). Random decision forests. In *Proceedings of the Third International Conference on Document Analysis and Recognition: 1* (pp. 278–282). IEEE.

Hopkins, M., Reeber, E., Forman, G., & Suermondt, J. (1999). Spambase Data Set. Hewlett-Packard Labs. Accessed May 2017. Available at http://archive.ics.uci.edu/ml/datasets/Spambase.

Hossain, M. S., Ong, Z. C., Ismail, Z., & Khoo, S. Y. (2017). A comparative study of vibrational response-based impact force localization and quantification using radial basis function network and multilayer perceptron. *Expert Systems with Applications, 85*, 87–98.

Hsu, C. W., Chang, C. C., & Lin, C. J. (2003). A practical guide to support vector classification. *Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan (www. csie. ntu. edu. tw/~cjlin/papers/guide/guide. pdf)*.

Hughes, G. (1968). On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory, 14*(1), 55–63.

Karthikeyan, B., Gopal, S., & Venkatesh, S. (2008). Partial discharge pattern classification using composite versions of probabilistic neural network inference engine. *Expert Systems with Applications, 34*(3), 1938–1947.

Kyle, R. A., Therneau, T. M., Rajkumar, S. V., Offord, J. R., Larson, D. R., Plevak, M. F., et al. (2002). A long-term study of prognosis in monoclonal gammopathy of undetermined significance. *New England Journal of Medicine, 346*(8), 564–569.

Land, W. H., Masters, T. D., & Lo, J. Y. (2000, June). Application of a GRNN oracle to the intelligent combination of several breast cancer benign/malignant predictive paradigms. In *Medical Imaging 2000: Image Processing: 3979* (pp. 77–86). International Society for Optics and Photonics.

Li, Y. (2014). *A complex adaptive system for accurate detection of multiple species of pathogens using multiple machine learning techniques*. New York (NY): State University of New York at Binghamton.

Liu, Y., & Yao, X. (1999). Ensemble learning via negative correlation. *Neural Networks, 12*(10), 1399–1404.

Lu, H., Wang, H., & Yoon, S. W. (2018). *A dynamic gradient boosting machine using genetic optimizer for practical breast cancer prognosis*. Expert Systems with Applications.

Marsland, S. (2011). *Machine learning: an algorithmic perspective*. Boca Raton, Florida (FL): Chapman and Hall/CRC.

Masters, T., Land, W. H., & Maniccam, S. (1998, October). An oracle based on the general regression neural network. In *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on: 2* (pp. 1615–1618). IEEE.

Mitra, P., Murthy, C. A., & Pal, S. K. (2002). Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*(3), 301–312.

Mohamed, H., Negm, A., Zahran, M., & Saavedra, O. C. (2015). Assessment of artificial neural network for bathymetry estimation using high resolution satellite imagery in Shallow Lakes: Case study El Burullus Lake. In *International Water Technology Conference* (pp. 12–14).

Mohammed, M., Khan, M. B., & Bashier, E. B. M. (2016). *Machine learning: algorithms and applications*. Boca Raton, Florida (FL): Crc Press.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). Foundations of machine learning. In *Adaptive computation and machine learning: 31* (p. 32). Cambridge, Massachusetts (MA): MIT Press.

Moro, S., Cortez, P., & Rita, P. (2014). A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems, 62*, 22–31.

Ontivero-Ortega, M., Lage-Castellanos, A., Valente, G., Goebel, R., & Valdes–Sosa, M. (2017). Fast Gaussian Naïve Bayes for searchlight classification analysis. *Neuroimage, 163*, 471–479.

Panagopoulos, O. P., Pappu, V., Xanthopoulos, P., & Pardalos, P. M. (2016). Constrained subspace classifier for high dimensional datasets. *Omega, 59*, 40–46.

Sharma, A. K., & Sahni, S. (2011). A comparative study of classification algorithms for spam email data analysis. *International Journal on Computer Science and Engineering, 3*(5), 1890–1895.

Sun, X. (2002). Pitch accent prediction using ensemble machine learning. *Seventh International Conference on Spoken Language Processing*.

Tabakhi, S., Moradi, P., & Akhlaghian, F. (2014). An unsupervised feature selection algorithm based on ant colony optimization. *Engineering Applications of Artificial Intelligence, 32*, 112–123.

Tan, A. C., & Gilbert, D. (2003). Ensemble machine learning on gene expression data for cancer classification. *Applied Bioinformatics, 2*(Suppl 3), S75–S83.

Tsoumakas, G., & Vlahavas, I. (2002). Distributed data mining of large classifier ensembles. In *Proceedings of Companion Volume of the Second Hellenic Conference on Artificial Intelligence*.

Unler, A., & Murat, A. (2010). A discrete particle swarm optimization method for feature selection in binary classification problems. *European Journal of Operational Research, 206*(3), 528–539.

Wahbeh, A. H., Al-Radaideh, Q. A., Al-Kabi, M. N., & Al-Shawakfa, E. M. (2011). A comparison study between data mining tools over some classification methods. *International Journal of Advanced Computer Science and Applications, 8*(2), 18–26.

Wang, X., Lin, P., & Ho, J. W. (2018). Discovery of cell-type specific DNA motif grammar in cis-regulatory elements using random Forest. *BMC Genomics, 19*(1), 929 https://bmcgenomics.biomedcentral.com/track/pdf/10.1186/s12864-017-4340-z.

Wang, Y., & Witten, I. H. (2002). Modeling for optimal probability prediction. In *Proceedings of the 19th International Conference in Machine Learning* (pp. 650–657). Sydney, Australia.

Wei, H., Zhu, Q., Guan, D., Yuan, W., Khattak, A. M., & Chow, F. (2017, December). Improved label noise identification by exploiting unlabeled data. In *Security, Pattern Analysis, and Cybernetics (SPAC), 2017 International Conference on* (pp. 284–289). IEEE.

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). Data mining: Practical machine learning tools and techniques. *Morgan kaufmann*. Massachusetts (MA): Morgan Kaufmann Publishers in Burlington.

Xia, Y., Liu, C., Da, B., & Xie, F. (2018). A novel heterogeneous ensemble credit scoring model based on bstacking approach. *Expert Systems with Applications, 93*, 182–199.

Xiang, K., Li, Y., Ford, W., Land, W., Schaffer, J. D., Congdon, R., et al. (2016). Automated analysis of food-borne pathogens using a novel microbial cell culture, sensing and classification system. *Analyst, 141*(4), 1472–1482.

Zhang, C., & Ma, Y. (2012). *Ensemble machine learning: methods and applications*. Boston, Massachusetts (MA): Springer Science & Business Media.

Zhang, Y., Lu, S., Zhou, X., Yang, M., Wu, L., Liu, B., et al. (2016). Comparison of machine learning methods for stationary wavelet entropy-based multiple sclerosis detection: Decision tree, k-nearest neighbors, and support vector machine. *Simulation, 92*(9), 861–871.

Zheng, B., Yoon, S. W., & Lam, S. S. (2014). Breast cancer diagnosis based on feature extraction using a hybrid of K-means and support vector machine algorithms. *Expert Systems with Applications, 41*(4), 1476–1482.

Zheng, B., Zhang, J., Yoon, S. W., Lam, S. S., Khasawneh, M., & Poranki, S. (2015). Predictive modeling of hospital readmissions using metaheuristics and data mining. *Expert Systems with Applications, 42*(20), 7110–7120.

Zięba, M., Tomczak, S. K., & Tomczak, J. M. (2016). Ensemble boosted trees with synthetic features generation in application to bankruptcy prediction. *Expert Systems with Applications, 58*, 93–101.