Proceedings of the 5th International Conference on Statistics: Theory and Applications (ICSTA'23)
Brunel University, London, United Kingdom – August 03-05, 2023
Paper No. 141
DOI: 10.11159/icsta23.141

Deception Detection using Random Forest-based Ensemble Learning

Kun Bu[1], Kandethody Ramachandran[1]
[1] University of South Florida, Department of Mathematics & Statistics,
Tampa, FL 33620-5700, USA
kunbu@usf.edu; ram@usf.edu

Abstract **-** The purpose of this work is to detect people lying using different ensemble machine learning algorithms to conclude a better classification model through comparison. Random forest (RF) performed efficient work while dealing with both classification and regression problems. In this paper, we proposed random forest-based ensemble learning, which is the combination of RF with SVM, GLM, KNNs, and GBM to improve the model performance. The data set that we used to fit into the machine learning models is the Miami University Deception Detection Database (MU3D). MU3D is a free resource containing 320 videos of Black and White targets, female and male, telling truths and lies. We fit the MU3D video level data set into random forest-based ensemble learning models, which include RF+SVM. Linear, RF+SVM. Poly, RF+GLM, RF+KNNs, RF+GBM (stochastic gradient boosting) and RF+WSRF (weighted subspace random forest). As a comprehensive comparison of the model performance, we conclude that our new combination of algorithms performs better than the traditional machine learning models. Our contribution in this work provides a robust classification method that improves the predicted performance while avoiding model overfitting.

Keywords**: Ensemble learning, deception detection, exploratory data analysis (EDA), weighted subspace random forest, generalized linear model, stochastic gradient boosting.**

1. Introduction

The traditional lie detection machine is a polygraph, which can provide people with an averaging accuracy between 78% and 90%.[1] With 90% accuracy, it seems to do a very good job at detecting lying; however, with 78% accuracy, we can hardly have much confidence in saying a person is lying. In other words, the polygraph test is easy to pass for well-trained people (i.e., company spies or country spies). Even for ordinary people who search for the word "polygraph" online, the next search suggestion would be "How to Pass a Polygraph Test?" Since the polygraph operating principle is to detect lies by looking for signs of an examinee's physiological changes. Once the examinee lies, it puts a blip on the polygraph machine that serves as a signature of that examinee's lies. In addition, the polygraph test is a time-based test that only captures the examinee's body reaction in each specific question, which means that the examinees themselves know that they are being tested whether they are lying. Therefore, polygraphs are not useful for underground and secret cases. Therefore, artificial intelligence (AI) approaches have come to scientists' minds. Why do we not just detect lying by applying machine learning algorithms to see if the accuracy of deception detection would be improved.

The Miami University Deception Detection Database (MU3D)[2] is a free resource containing 320 videos of Black and White targets, female and male, telling truths and lies. Eighty (20 Black female, 20 Black male, 20 White female, and 20 White male) targets were recorded speaking honestly and dishonestly about their social relationships. Each target generated four different videos (i.e., positive truth, negative truth, positive lie, negative lie), yielding 320 videos fully crossing target race, target gender, statement valence, and statement veracity. In previous studies of MU3D, scholars conducted research using standardized stimuli that can aid in building comprehensive theories of interpersonal sensitivity, enhance replication among labs, facilitate the use of signal detection analyses, and promote consideration of race, gender, and their interactive effects in deception detection research1. Our motivation also comes from those previous studies and aims to develop better deception detection via machine learning tools.

Ensemble learning, sometimes referred to as a multiclassifier system, builds and combines multiple classifiers to complete the learning task. There are two choices for obtaining multiple classifiers. The first is supposed that all individual classifiers are of the same type or homogenous. For example, both decision tree individual classifiers or both neural network individual classifiers (i.e., bagging and boosting, for example, random forest). The second is supposed that all individual classifiers are not homogeneous or heterogeneous. For example, in this paper, we have a classification problem of deception detection. We use a support vector machine (SVM) individual learner, logistic regression (LR) individual learner and k-nearest neighbours (KNNs) individual learner to learn the training set and then determine the final strong classifier by some combination strategy. This integration is called Stacking[3]. In the experimental section, we applied bagging, boosting, and stacking and selected a better ensemble model to classify people lying.

2. Methods

MU3D is collected by recording 80 targets speaking honestly and dishonestly about their social relationships. The dataset was divided into two parts: video level and target level. In the video level dataset, information such as the valence indicates whether the statement in the video is negative or positive; VidLength ms and VidLength sec indicate the length of video in milliseconds and seconds, respectively. There are a total of 12 variables with 1 label variable called Veracity in the video level dataset, and a short variable description is shown in Table 1.

Table 1. MU3D video-level database variable descriptions (Hugenberg et al. (2017)).

Video-Level Variables	Mean (±SD)	Description
VideoID	/	ID associated with the video.
Veracity	/	Indicates whether the statement in the video is a truth or a lie: value of 0 indicates a lie, value of 1 indicates a truth.
Valence	/	Indicates whether the statement in the video is negative or positive: value of 0 a indicates negative statement, value of 1 indicates a positive statement.
Sex	/	Indicates target’s sex: value of 0 indicates a female target, value of 1 indicates a male target.
Race	/	Indicates target’s race: value of 0 indicates a Black target, value of 1 indicates a White target.
VidLength_ms	35728.86±3491.95	Indicates length of the video in milliseconds.
VidLength_sec	35.73±3.49	Indicates length of the video in seconds.
WordCount	106.69±23.48	Indicates the number of words contained in the full transcription of the video.
Accuracy	0.52±0.21	Indicates average accuracy (i.e., proportion correct) across raters who viewed the video.
TruthProp	0.59±0.18	Indicates average truth proportion (i.e., proportion of truth responses) across raters who viewed the video.
Attractive	4.08±0.58	Indicates average attractiveness ratings (measured on a scale ranging from 1 ”Not at all” to 7 ”Extremely”) across raters who viewed the video.

Trustworthy	4.16±0.52	Indicates average trustworthiness ratings (measured on a scale ranging from 1 ”Not at all” to 7 ”Extremely”) across raters who viewed the video.
Anxious	3.04±0.65	Indicates average anxiousness ratings (measured on a scale ranging from 1 ”Not at all” to 7 ”Extremely”) across raters who viewed the video.
Transcription	/	Full transcription of the video.

2.1. Exploratory Data Analysis

Due to the data properties, normalizing the data so that they are of the same order of magnitude is much better for machine learning. Except for VideoID and the last variable Transcription, a normalized boxplot was created for all other variables by Veracity. As shown in Figure 1, only the variables Accuracy, TruthProp and Trustworthy have differences in Veracity, which makes it difficult for us to start choosing features to train a classification prediction model.

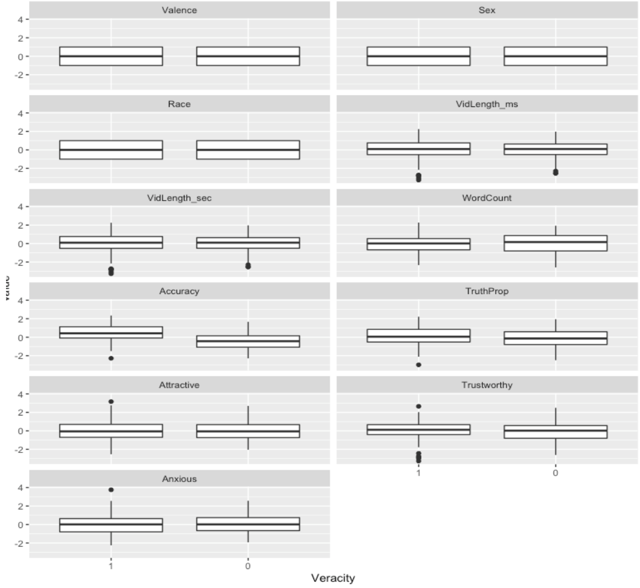


Figure 1. Boxplot for video-level dataset variables.

A correlation scatter plot is shown in Figure 2, which indicates that VidLength ms and VidLength sec, variable TruthProp and Trustworthy, and variable Accuracy and TruthProp are highly correlated; however, the effect here is similar to that of multicollinearity in linear regression. Our learned model may not be particularly stable against small variations in the training set because different weight vectors will have similar outputs. The training set predictions, however, will be stable, and so will test predictions if they come from the same distribution. Based on the variable’s linear correlation relationship in Figure 2, we can reduce the feature dimension from 11 to 9, where VidLength_ms and TruthProp were removed because they are linearly correlated with VidLength_sec and Accuracy, respectively.

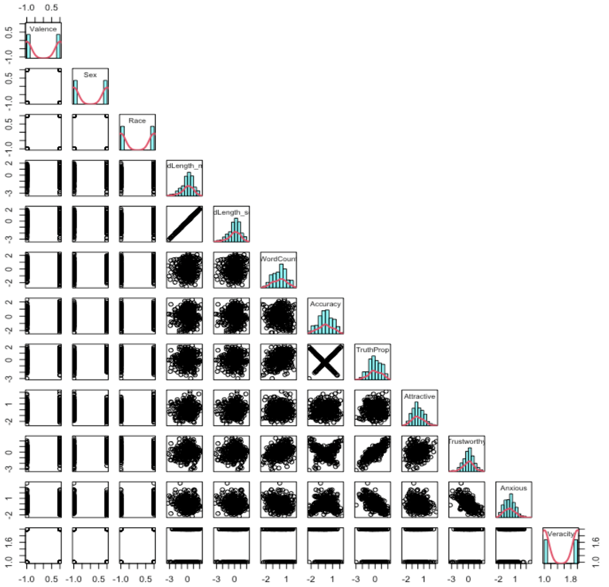


Figure 2. Correlation scatter plot for video level dataset.

2.2. Ensemble Learning for Deception Detection

The next step is to fit MU3D into a machine learning model and see how the computer performs in detecting lies. We first trained three different models based on the data properties, including support vector machine (SVM), binary logistic regression (BLR) and random forest (RF), to predict deception. The basic idea of these three selected algorithms is based on the algorithm flowchart in Figure 3. The purpose of this algorithm flowchart is to create a tool that helps not only select the possible modeling techniques but also better understand the problem itself. As shown in Figure 3, by answering the following questions from the flowchart, we initially decided the main three modeling techniques to be applied in this prediction. In the next subsection, we explain why these three machine learning algorithms were chosen based on the data properties and the model assumptions.

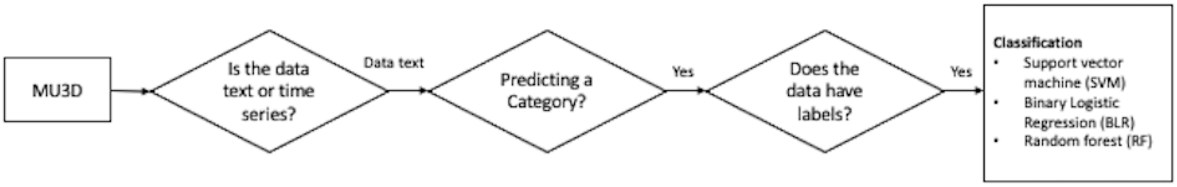

MU3D
Is the data text or time series?
Data text
Predicting a Category?
Yes
Does the data have labels?
Yes
Classification
• Support vector machine (SVM)
• Binary Logistic Regression (BLR)
• Random forest (RF)

Figure 3. Flowchart of the Basic Ideas in Deception Detection.

2.3. Preliminaries Machine Learning Algorithms

Support vector machines are based on a decision plane concept that defines decision boundaries, and SVMs have been shown to perform well in a variety of settings and are often considered one of the best “out of the box” classifiers according to James, G. et al. (2013)[4]. A decision plane is a separation plane between a set of objects with different types of membership.

SVM is a supervised learning method used to perform binary classification on data. According to the statistical data analysis section, the data properties show that we have exactly two classes: Lies or Truth. In addition, SVM can deal with real valued features, which means there are no categorical variables in the data, such as our dataset above. All the features except the Transcription are numerical numbers, which are much fittable by using SVM. Moreover, SVM can perform well on many features; for example, it works with tens, hundreds, and thousands of features. In our dataset, we have more than 10 features, which motivates us to choose SVM according to Chapelle et al. (2002)[5]. Another reason is that SVM has simple decision boundaries, indicating that there are no issues with overfitting. The

SVM can be defined as a linear classifier under the following two assumptions[6]: 1) The distance from the SVM's classification boundary to the nearest data point should be as large as possible; the distance formulas include Euclidean distance, Manhattan distance, Chebyshev distance and Minkowski distance. where the Euclidean distance and Manhattan distance are special forms of the Minkowski distance[7], and 2) the support vectors are the most useful data points because those points are the ones most likely to be incorrectly classified. This means that the primary goal of training SVMs is to find support vectors in the dataset that both separate the data and find the maximum margin between classes.

Binary logistic regression (LR) is a regression model where the target variable is binary, that is, it can take only two values, 0 or 1. It is the most utilized regression model in deception prediction, given that the output is modeled as truth (1) or lie (0). BLR is a statistical tool that classifies the MU3D target person in a video to either lie or not. BLR has two stages: training and evaluation. At the training stage, it uses video-level data from both the lie and truth and builds a detection module. At the evaluation stage, data that were not used in the training stage are used to evaluate the detection model.

Logistic regression is also called logit (log unit) regression, and we usually build a logistic regression model by starting to build the model by combining the generalized linear model with the logit function or from the point of a random variable following the logistic distribution. Binary logistic regression has the following assumptions: 1) adequate sample size, 2) absence of multicollinearity and 3) no outliers. Note that according to EDA, the outliers in our dataset need to be removed before fitting this model so that it does not violate the assumptions.

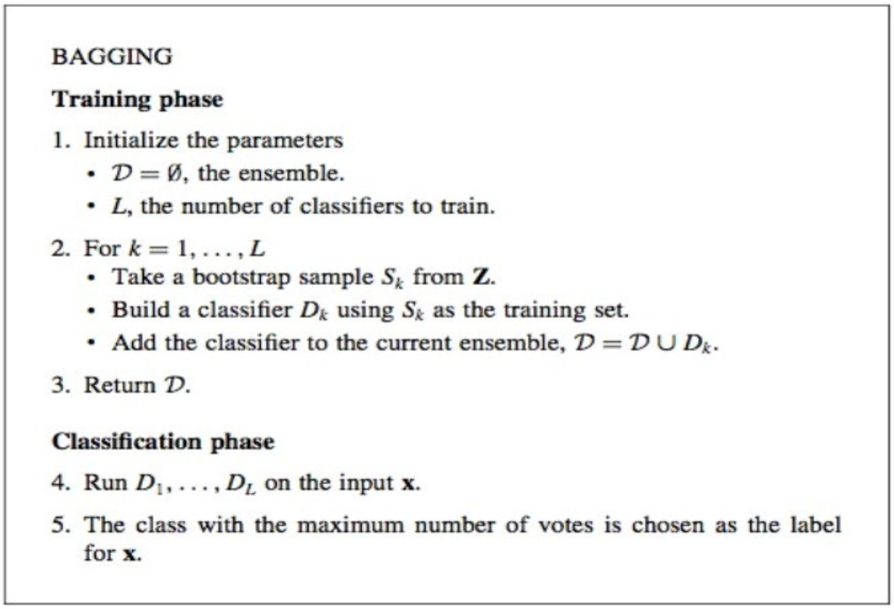
The mathematical expression of the logistic regression is:

$$g(y) = ln(\frac{y}{1-y}) = \widehat{w}^T \cdot \hat{x}$$

where $g(y) = ln(\frac{y}{1-y})$ is called the logit function and is the link function for the generalized linear model. This logistic regression model can be expressed as

$$y = \frac{1}{1+e^{-x}}$$

where e is the natural logarithm, and the above function is called the sigmoid function.



```
BAGGING
Training phase
1. Initialize the parameters
    • $\mathcal{D} = \emptyset$, the ensemble.
    • $L$, the number of classifiers to train.
2. For $k = 1, \ldots, L$
    • Take a bootstrap sample $S_k$ from $\mathbf{Z}$.
    • Build a classifier $D_k$ using $S_k$ as the training set.
    • Add the classifier to the current ensemble, $\mathcal{D} = \mathcal{D} \cup D_k$.
3. Return $\mathcal{D}$.
Classification phase
4. Run $D_1, \ldots, D_L$ on the input $\mathbf{x}$.
5. The class with the maximum number of votes is chosen as the label
   for $\mathbf{x}$.
```

Figure 4. Bagging Algorithms by Kuncheva, Ludmila I [10].

According to Fern´Andez-Delgado et al. (2014)[8], "The classifier most likely to be the best are the random forest versions, the best of which (implemented in R and accessed via caret), achieves 94.1% of the maximum accuracy overcoming 90% in the 84.3% of the data sets." This quote clearly pointed out the power of RF in the classification field. The basic idea of RF is to produce numerous trees and combine the results. The random forest technique does this by applying two different tricks in model development. The first is the use of bootstrap aggregation or, in short, bagging. In the bagging process, a single decision tree is built on a random sample of the dataset, which accounts for approximately two-thirds of the total observations (note that the remaining third is called out-of-bag (oob)). This is repeated dozens or hundreds of times, and then the average of the results is calculated. The growth and pruning of each tree are not based on any error measure, which means that the variance of each tree is high. However, by averaging the results, the variance can be reduced without increasing the bias according to Merentitis et al. (2014)[9].

The next thing that random forest brings to the table is that concurrently with the random sample of the data, that is, bagging[10] according to Kuncheva, Ludmila I. It also takes a random sample of the input features at each split, and the original bagging algorithm is presented in Figure 4. We will use the default random number of the predictors that are sampled, which, for classification problems, is the square root of the total predictors. The advantage of RF is that by using this random sample of the features at each split and incorporating it into the methodology, one can mitigate the effect of a highly correlated predictor, becoming the main driver in all the bootstrapped trees. The subsequent averaging of the trees that are less correlated to each other is more generalizable and robust to outliers than if only bagging is performed.

Thomas G. Dietterich[11] noted that the effectiveness of ensemble learning can be attributed to both statistical and computational reasons. Therefore, in this paper, we applied random forest-based ensemble learning to improve the prediction performance. We fit the MU3D video level data set into random forest-based ensemble learning models, which include RF+SVM. Linear (SVM with linear kernel), RF+SVM. Poly (SVM with polynomial kernel), RF+GLM (Generalized Linear Model), RF+KNNs (k-Nearest Neighbors), RF+GBM (Stochastic Gradient Boosting) and RF+WSRF (Weighted Subspace Random Forest). We keep our ensemble learning as simple as just combining one algorithm with RF at each time to avoid complicated models because the more complicated the model is, the easier it will cause overfitting. Section 5 explains a comprehensive comparison of the model performance based on the experiment, and then we conclude that our new combination of algorithms performs better than the traditional machine learning models.

3. Results

3.1 Preliminaries Machine Learning Algorithms

As mentioned in the Data description section, the MU3D contains 320 data cells that tell truths and lies. Eighty targets were recorded as speaking honestly and dishonestly about their social relationships. Each target generated 4 different levels (i.e., positive truth, negative truth, positive lie, negative lie), yielding 320 video data cells. Before building our models with SVM and BLR, we choose a common split ratio of 80:20 for training/validation and testing. For the RF, we do not split our dataset because the RF does not require a split sampling method to assess the accuracy of the model. It performs internal validation as 2-3rd of available training data is used to grow each tree and the remaining one-third portion of training data is always used to calculate out-of-bag error to assess model performance.

3.2 Normalization and Feature Engineering

Normalization is a data preparation technique that is frequently used in machine learning. The process of transforming the columns in a dataset to the same scale is referred to as normalization. Every dataset does not need to be normalized for machine learning. It is only required when the ranges of characteristics are different. Standardization Scaling, that is, centering variables at zero and standardizing the variance at one. Subtracting the mean of each observation and then dividing by the standard deviation is the procedure as follows:

$$X' = \frac{X - \mu}{\sigma}$$

For example, the ranges of variables VidLength ms, VidLength sec and Attractive, Trustworthy, and Anxious are varied. Therefore, normalization was applied for the MU3D. Note that normalization was performed after splitting the data between the training and test sets, using only the data from the training set. This is because when normalizing the test set, one should apply the normalization parameters previously obtained from the training set as is. Recalculating them on the test set would be inconsistent with the model, which would produce incorrect predictions, and the test set plays the role of fresh unseen data, so it is not supposed to be accessible at the training stage. Using any information from the test set before or during training is a potential bias in the evaluation of the performance.

Since there are more than 10 dimensions in this MU3D, as shown in Table 1 and Figure 2, multicollinearity needs to be addressed and removed. We applied principal component analysis (PCA), which takes advantage of multicollinearity and combines highly correlated variables into a set of uncorrelated variables. Therefore, PCA can effectively eliminate multicollinearity between features. Figure 5 shows the scree plot and the cumulative variance plot from PCA, which indicated that the cut-off number of PCs was 5.

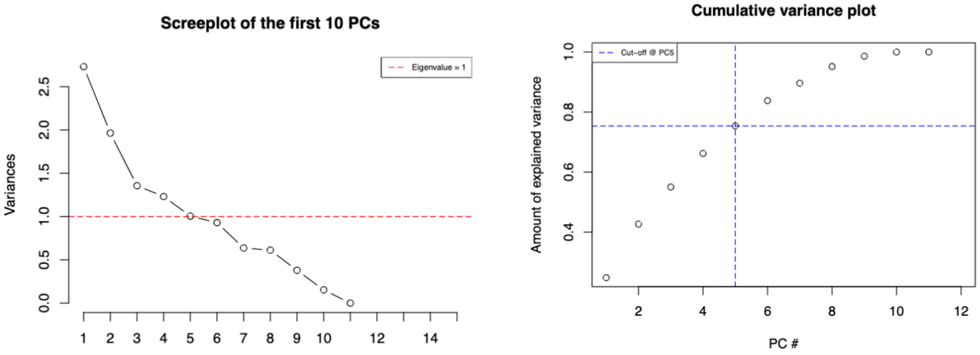


Figure 5. Scree plot and the cumulative variance plot with cut-off = 5 of the first 10 PCs based on the PCA.

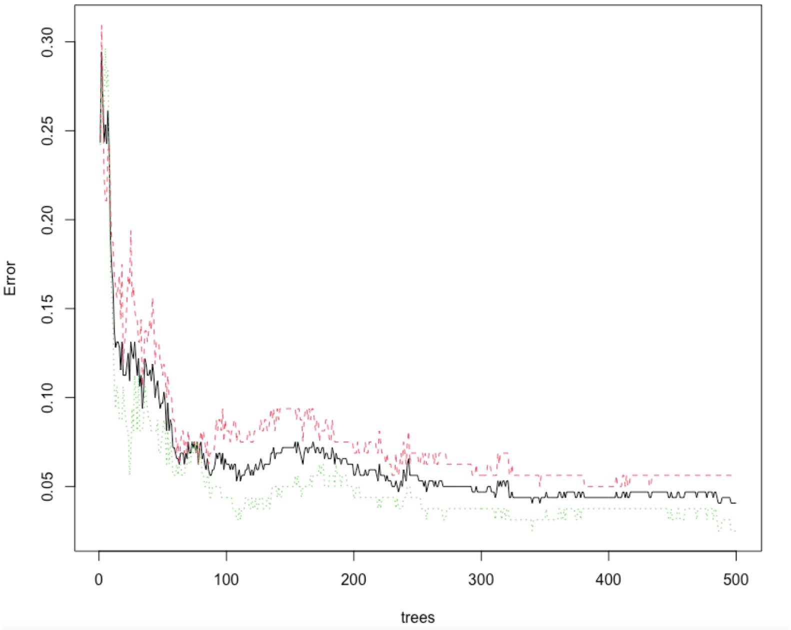


Figure 6. MSE by the number of trees in the random forest model.

Figure 6 shows the MSE by the number of trees in the model. We can see that as the trees are added, significant improvement in MSE occurs early on and then flatlines just before 400 trees are built in the forest, and the optimal tree can be specified in the model. Here, we split the data as 80/20, and our model's optimal number of trees is ntree = 334. Based on this optimal ntree, the error rate in the training set is just 0.78%. Figure 7 shows the random forest variance importance plot and Gini plot of the random forest, which indicated that the 5 selected features are accuracy, trustworthiness, anxiety, word count and attractiveness.

3.3 Ensemble learning with selected algorithms

Parmar et al. (2018)[12] explained that since RF itself is an ensemble learning method, the widely discussed problem for RF is its overfitting problem. Segal, Mark R (2004)[13] provided the machine learning benchmarks on RF, and he mentioned that some researchers believe that RF does not overfit because of the two random processes in random sampling for each tree node and random selecting features for each splitting. However, since the random forest is based on the decision tree, the decision tree has been proven to have an overfit problem. In addition, most machine learning scientists believe that those two random processes in RF can only help to reduce the chance of overfitting but cannot avoid it. Based on this opinion, this paper implements the random forest-based ensemble learning method, which helps improve the model prediction performance while reducing the chance of model overfitting. The core idea is by the different types of ensembles, such as averaging, majority vote, and weighted average.

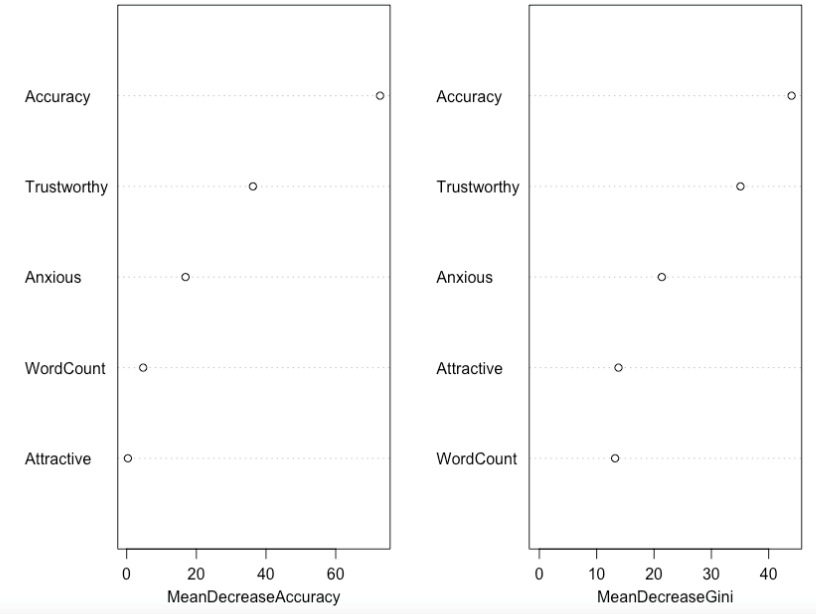


Figure 7. RF variance importance plot and Gini plot.

Ensemble learning is the key ingredient for winning almost all machine learning hackathons and makes the model more robust and stable, thus ensuring decent performance on the test cases in most scenarios. One can use ensembling to capture linear and simple as well nonlinear complex relationships in the data. This can be done by using two different models and forming an ensemble of two. In this paper, we include 6 different random forest-based ensemble learning models: RF+SVM. Linear (SVM with linear kernel), RF+SVM. Poly (SVM with polynomial kernel), RF+GLM (Generalized Linear Model), RF+KNNs (k-Nearest Neighbors), RF+GBM (Stochastic Gradient Boosting) and RF+WSRF (Weighted Subspace Random Forest).

Overall, the ensemble method is a meta-algorithm that combines several machine learning techniques into a prediction model to achieve the effect of reducing variance (bagging), boosting (boosting), or improving prediction (stacking).

3.4 Computational Results

Our results show that RF+GBM (stochastic gradient boosting) provides the highest prediction among the other ensemble learning methods. Table 2 shows the experimental results with accuracy, sensitivity, specificity, and kappa value for each ensemble learning.

According to Table 2 and Figure 8, our result shows that RF has a better performance among all the individual classifiers, followed by the WSRF model and KNNs. Although GBM does not show good performance as an individual classifier, it achieves the highest accuracy when ensemble with RF compared with other RF-based ensemble learning classifiers. In addition, the computational time is higher in all the ensemble studies compared to the individual classifier, which is inevitably due to the model complexity; however, RF+GBM has the highest performance with acceptable model running time at the same time, which stands out from all the classifiers.

Table 2. Evaluation of Random-forest based Ensemble Learning Methods on MU3D Video Level Database.

Method	Accuracy	Sensitivity	Specificity	Kappa	Computational Time (Sec)
RF	0.9531	1.0000	0.9062	0.9062	0.1481
GLM	0.6875	0.7812	0.5938	0.3750	0.0199
KNNs	0.8281	0.9062	0.7500	0.6562	0.0166
SVM	0.7031	0.4722	1.0000	0.4391	0.0185
WSRF	0.9375	0.9688	0.9062	0.6171	0.5292
GBM	0.6875	0.6774	0.6970	0.3744	4.7694
RF+GLM	0.7047	0.7406	0.6687	0.4094	5.9365
RF+KNNs	0.7266	0.7344	0.7188	0.4531	5.9884
RF+SVM.Poly	0.7906	0.9719	0.6094	0.5812	18.2516
RF+SVM.Linear	0.8062	1.0000	0.6125	0.6125	6.4943
RF+WSRF	0.9609	0.9781	0.9437	0.9219	10.9316
RF+GBM	0.9750	0.9875	0.9625	0.9500	7.2205

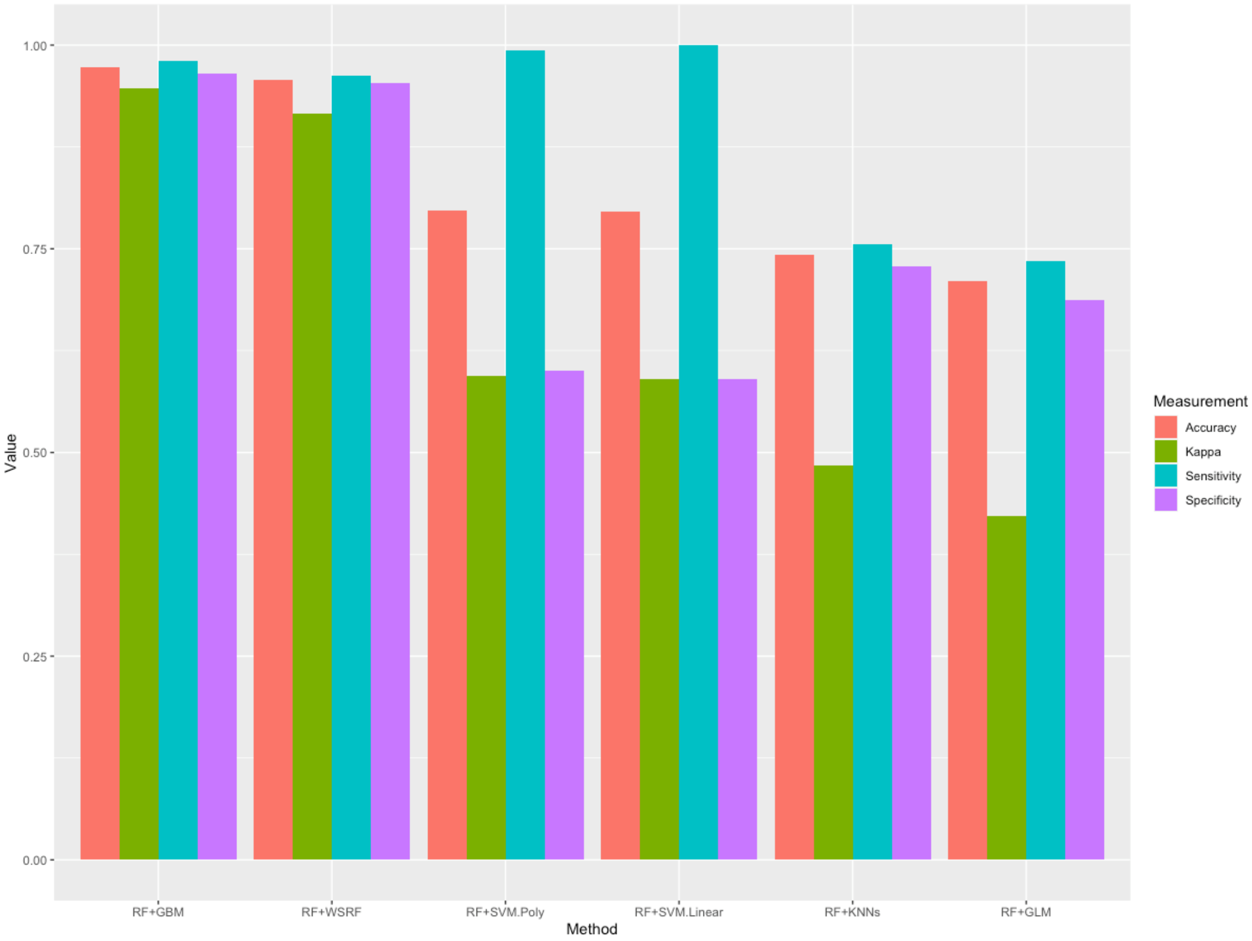


Figure 8. RF-based ensemble learning performance comparison.

4. Conclusion

In conclusion, RF+GBM and RF+WSRF ranked the top 2 among the other ensemble learning models, with overall accuracies of 0.9750 and 0.9609, sensitivities of 0.9875 and 0.9781, specificities of 0.9625 and 0.9437, and kappa values of 0.9500 and 0.9219, respectively. Although RF+SVM. Linear has the highest sensitivity, and the specificities are the lowest among all six methods, which means there are few false negative results and more false positive results; that is, the model correctly predicts the lie cases but misclassifies the truth.

References

[1]. Spence, Sean A., and Catherine J. Kaylor-Hughes. "Looking for truth and finding lies: The prospects for a nascent neuroimaging of deception." Neurocase 14.1 (2008): 68-81.

[2]. Lloyd, E. P., Deska, J. C., Hugenberg, K., McConnell, A. R., Humphrey, B., & Kunstman, J. W. (2017). Miami University deception detection video database. Manuscript under review.

[3]. Ting, Kai Ming, and Ian H. Witten. "Stacking bagged and dagged models." (1997).

[4]. Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. An introduction to statistical learning, volume 112. Springer, 2013.

[5]. Chapelle, Olivier, et al. "Choosing multiple parameters for support vector machines." Machine learning 46.1 (2002): 131-159.

[6]. Steinwart, Ingo, and Clint Scovel. "Fast rates for support vector machines using Gaussian kernels." The Annals of Statistics 35.2 (2007): 575-607.

[7]. Walters-Williams, Janett, and Yan Li. "Comparative study of distance functions for nearest neighbors." Advanced techniques in computing sciences and software engineering. Springer, Dordrecht, 2010. 79-84.

[8]. Manuel Fern ́andez-Delgado, Eva Cernadas, Sen ́en Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? The journal of machine learning research, 15(1):3133–3181, 2014.

[9]. A. Merentitis, C. Debes and R. Heremans, "Ensemble Learning in Hyperspectral Image Classification: Toward Selecting a Favorable Bias-Variance Tradeoff," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 7, no. 4, pp. 1089-1102, April 2014, doi: 10.1109/JSTARS.2013.2295513.

[10]. Kuncheva, Ludmila I. Combining pattern classifiers: methods and algorithms. John Wiley & Sons, 2014.

[11]. Dietterich, Thomas G. "Ensemble learning." The handbook of brain theory and neural networks 2.1 (2002): 110-125.

[12]. Parmar, A., Katariya, R., & Patel, V. (2018, August). A review on random forest: An ensemble classifier. In International Conference on Intelligent Data Communication Technologies and Internet of Things (pp. 758-763). Springer, Cham.

[13]. Segal, Mark R. "Machine learning benchmarks and random forest regression." (2004).