Outlier Prediction Using Random Forest Classifier

Divya Pramasani Mohandoss
College of Computing and Software Engineering
Kennesaw State University
Atlanta, USA
dpramasa@students.kennesaw.edu

Yong Shi, Kun Suo
College of Computing and Software Engineering
Kennesaw State University
Atlanta, USA
yshi5, ksuo@ kennesaw.edu

Abstract— Random forest is an ensemble learning method for classification, regression and other tasks that operate by constructing a multiple decision trees using training data and majority of the class will be consider as output. Out-of-Bag (OOB) takes the samples from the training set with replacement. In random forests, if you choose oob to true then there is no need for a separate test set to validate the model. It is estimated internally when the forest is built on training data, and each tree is tested on one-third of the samples not used in building that tree. Out of bag estimate an internal estimate of a random forest as it is being constructed. In this paper we propose two approach to implement outlier prediction by applying random forest classifier and LSTM model with experiments.

Keywords— Random forests, OOB, LSTM.

I. INTRODUCTION

Outlier is the abnormal behavior in the dataset or any business. Outlier prediction refers to the problem of predicting patterns in data that are not consistent with the expected behavior. In data mining, outlier detection is a significant concept. Outlier identification is a commonly investigated issue for a range of applications, such as detection of credit card fraud, detection of enemy activity in military surveillance, insurance fraud and many other fields. Outlier is present in a wide number of areas that have resulted in a lot of outlier identification and analysis. Many of these approaches have been developed to fix oriented issues. Issues concerning a specific problem domain, although others have been built in a more general way. The purpose of this research work is to provide a detailed insight into the different ways in which two approaches have been taken to help predict outliers and potential areas for future research.

II. RELATED WORK

In Data Analysis, outliers are often referred to in the outlier predictions as deviations or deviants. The data comes from various processes in most applications that may represent the system's operation or observation in terms of outliers. Ensemble is commonly used in a variety of Data Mining and Machine Learning applications[1] as clustering, grouping, outlier detection, etc. Based on the Decision Tree algorithm [2], Random Forests are created. In classification or prediction, we include information from explanatory variables to categorize data samples into two or more different classes[4]. The most frequently used data mining activity in medical diagnostics is grouping. There are many ways to build models for classification. The problem with the Decision Tree is that as the complexity of the problem increases, the robustness of its future forecasts decreases, particularly when dealing with a small dataset[3].

Random forests are the ensemble learning methods used for classification and regression problem. Random forest is an ensemble collection of multiple decision trees that are constructed from the bootstrap samples. While constructing each tree as high variance, random forests take the average of multiple different trees, which in turn reduces the high variance and leave us with a powerful classifier.

Random forests require no parameter tuning. They are different from many Machine Learning models used today. Tree-based models like random forest are constructed from samples out of the dataset, picking a less features, and finding the value that makes the best split in our dataset.

In this model, the original dataset is split into training and test set. From the training data samples are randomly taken with replacement. These subsets of samples are known as bootstrap samples. Using each bootstrap, we construct a Decision Tree. Each Decision Tree is trained separately on these bootstrap samples. The aggregation of these Decision Tree is called the Random Forest ensemble (which is collection of trees called forest).

Since each decision tree takes a different set of training data as input, the original training dataset do not impact the result obtained from the aggregation of Decision Trees. Therefore, bagging concept reduces overfitting variance without changing the bias of the complete ensemble. Random forests have evolved from a single algorithm to a complete model framework[5] and have been commonly used in a wide range of fields. The mathematical forces behind their performance are not well known, despite the widespread use of random forests in nature. The early theoretical work of[6] is basically focused on inference and heuristics of mathematics and has not been rigorously formalized until very recently[7]. Random forests are correlated with two key properties of theoretical importance. The first is the accuracy of estimators generated by the algorithm, which asks if, as the data set grows infinitely large, we can guarantee convergence to an optimal estimator. In addition to accuracy, we are also interested in convergence metrics gives some insights into the behavior of the random forest-based variable significance index and proposes to explore two classical variable selection problems.

2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC) | 978-1-6654-1490-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/CCWC51732.2021.9376077

978-0-7381-4394-1/21/$31.00 ©2021 IEEE

In this paper, we focus solely on the OOB vs. Test Partition Efficiency and LSTM model of three Different Datasets. The objective was to measure the OOB score on the training set, but to include only the trees in a row's measurement and that row was not included in that tree's training. This helps one to see if the model, without having a different test range, is overfitting.

There is an overview of the simulated and actual data in the models in [11]. Simulated data were used to analyze the behavior of an OOB error in which all predictors are not associated. This delivers insight pathways that contribute to bias in the OOB error. Based on these observations, there is a probability of bias in the OOB error. Subsequently, nuanced real-world data are used to determine the degree of bias in these environments in practice. OOB error will overestimate true prediction error depending on the choice of RF parameters[12].

The Random Forest with 500 trees and 1000 trees created very similar OOBs. In compliance with these results, the number of trees has been set at 1000. Every setting was replicated 500 times to obtain stable results[13]. Experiments on many datasets reveal that the out-of-bag estimation and 10-fold cross-validation have comparable results but are both skewed [15, 16]. The bias is significantly minimized by subsampling without substitution and by selecting the same number of observations from each group. However, even after these changes, there is a limited degree of prejudice. The residual bias exists because when trees have the same predictive capabilities, the one that performs better on the in-bag samples will do worse on the out-of-bag samples.

The idea for this research was partly taken from [14] the author used the dense neural network cells in the autoencoder model in that article. In our auto-encoder model, we use Long Short-Term Memory (LSTM) neural network cells. A subtype of the more popular recurrent neural networks, LSTM networks are (RNN). A core feature of recurrent neural networks is their capacity to survive for subsequent use in the network knowledge or cell state. This makes them particularly well suited for temporal data processing that grows over time. In activities such as speech recognition, text translation, LSTM networks are used to evaluate sequential sensor readings for anomaly detection.

III. RANDOM FOREST IMPLEMETATION

In this section, we define the random forest algorithm processes. Each tree in the random forest is developed separately.

We do the following steps:

1. Draw the bootstrap samples from the original training set.
2. For each bootstrap sample, build the decision tree follow the steps to find the split in a tree by repeating the steps.
   i. Calculate the entropy of the class, if the entropy is 0 or 1 that means the class is pure.
   ii. Calculate the information gain after each split, if information gain is higher, then split the tree further.
3. Grow the tree until following conditions are met.
   i. Until the feature = square root (all features)
   ii. Compute the information gain for each value in bootstrap samples.
   iii. Split the node into two child nodes
   iv. Split until the maximum depth is reached.
4. Output the collection of trees as forest.

1. Entropy

Entropy is a measure of ambiguity or impurity using the formula below.

$$E(X) = -\sum Prob_j \, logProb_j \qquad (1)$$

such that $Prob_j$ is the probability of class j. [20] In the case of classification entropy takes on the form

$$E(X) = -Prob \, log_2 Prob - (1 - prob) log_2 (1 - Prob) \qquad (2)$$

If the sample is absolutely in one class, the entropy is zero and if the sample is equal to or greater than two classes, the entropy is one. Split is achieved when the entropy is low.

2. Information Gain

Information gain [21] tells us how important the feature vectors are to a given attribute. The Information Gain is calculated using the formula below,

$$I(N) = I(D_{parent}) - \frac{Node_{left}}{Node_{parent}} I(D_{left}) - \frac{Node_{right}}{Node_{parent}} I(D_{right}) \qquad (3)$$

Where D_{parent} , D_{left} , D_{right} represent the datasets from the parent, left, and right children nodes, $Node_{parent}$, $Node_{left}$, $Node_{right}$ represent the number of observations in the parent, left and right children nodes and I(N) denotes the entropy for that node.

This formula can be interpreted as

$$I(N) = E_{before} - E_{after} \qquad (4)$$

where E_{before} is a measure of how uncertain we were with our data before the split and E_{after} is a measure for how uncertain we are after split the data.

3. Bootstrapping

Due to the random injection into each tree, random forests are powerful. Based on the bootstrapped samples[22], each individual decision tree is constructed.

The method of taking the random sample points with replacement from the training set is bootstrapping. This implies that some data points will be selected more than once in our data set and some will not be selected at all. We can calculate the probability a data points that was omitted from our bootstrapped dataset is $(1 - \frac{1}{N})^N$.

Approximately one-third of the data points in each separate tree will be left out by Bootstrapping N samples with substitution. Since only two-thirds of the data is used to build each individual tree, most of the trees will vary from one another.

4. Bagging

Bagging is the process of creating a bootstrap decision tree from bootstrap samples. To find the best split in the data at that node, sample each feature. For all trees, this process is replicated. Random forests follow the bagging method.
However, for a dataset with N features, each tree will look at a subset of features i.e.(√N) [23]. We are looking at (√N) features at one time, and many of the trees will look at different features from one another.

5.Out-Of-Bag Estimate

The samples of the OOB (out-of-bag) are taken from the data points that are not chosen to build a tree. On samples not seen during training, the Out of Bag Score is simply calculated. In bagging techniques, it has an important role, as it constructs a new set by drawing with replacement at random. The average OOB estimate is above these scores. While out-of-bag is a kind of estimate of the single tree in the ensemble over all possible selections of the training set maintaining a separate test, set requires a significant number of points so that the remaining data is reasonably estimated[24]. Out-of-bag estimate gives the ability to tell how well the model behaves while at the same time uses all data available. After computing an OOB rate for each tree and take the average of all those scores to get an estimate to see how accurate our random forests perform, this is essentially leave one out cross validation (i.e., we do not need separate cross validation).

Fig. 1. shows the pseudo code of our approach:

```
Algorithm: Random-Forest Classifier
for i = 1 → N
Bootstrapping ($B_i$)
 do
    Randomly select the samples from training data D with replacement to produce $D_i$ and $B_i$ where $D_i$ is bootstrap, and $B_i$ is out-of-bag
    Create a root node with $D_i$
    Call Build-Tree ($D_i$)
  end for
    Build-Tree (T):
      calculate the entropy of T
      if T contains value of only one class:
        then terminate the node
       else
         Randomly select s of the possible split from the features in T
         Calculate the information gain for features in T
         Select feature n with highest information gain to split the tree
         Create f child nodes of T, such as $T_1$, ..., $T_n$ where n has values to max-depth
       for i = 1 → n
       do
         Set the contents of $T_i$ →$D_i$, where Di is all samples in T that match $N_i$
         Call Build-Tree ($N_i$)
       end for
      end if
```

```
OOB-Score:
    for i = 0 → L
    do
      if A equals to L value:
        then increase the counter
        calculate the score S using the counter and A, where A is actual values
        return S
    end for
```

IV. LSTM MODEL

A form of recurrent neural network is LSTM-Long Short-Term Memory (RNN). [24] RNN suffered from two difficulties as the gradient is fading and the gradient is exploding, rendering it unacceptable. Later, by the clear integration of the memory unit called Cell into this LSTM network, LSTM was implemented to solve these two problems by RNN. Below fig [5] is the LSTM building block which contains Input, Output, Non-linearities (like sigmoid, hyperbolic tangent, bias) , and Vector operations like (+, x).

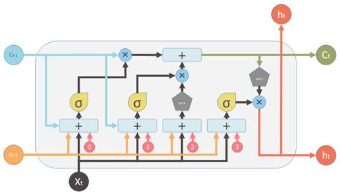


Fig. 2: LSTM Building block

So, this unit works by taking input, previous output, and previous memory and this generates new output, and it changes the memory.

LSTM architecture has three gates [25] (or valves) output valve, memory valve, forget valve. Here, the output valve determines how much fresh memory this valve can deliver to the next layer of LSTM. The memory valve governs how much fresh memory can affect the old memory. Forget valve control by one-layer neural network.

V. DATASET DESCRIPTION

This section explains the experimental methods used to evaluate the appropriateness of the Random Forest Classifier and the LSTM, as well as the experimental performance. Three different datasets were used for the experiment: Satellite, Shuttle, and Skin dataset from UCI Machine Learning Repository [17-19] each are multi-dimensional dataset. Following are dataset descriptions.

1. Satellite

The dataset was taken from the UCI Machine Learning Repository. The characteristics of dataset is multivariate. The data collection operates in a satellite image of 3x3 neighborhoods and the designation correlated with the central pixel in each neighborhood. Here we are forecasting, given multi-spectral values. The pixel class is defined as a number in the data collection. It has 36 numerical properties, in the range of 0 to 255, and a total of 6435 instances.

2. Shuttle

The original shuttle dataset is taken from UCI Machine Learning repository is a multi-class classification dataset with dimensionality 9. The preparation and evaluation details are merged here. There are five classes 2, 3, 5, 6, 7 which are grouped to form the outlier class, while class 1 forms the inlier class. The Class 4 mark has been discarded. There are 58,000 examples like this.

3. Skin

Skin dataset is obtained by random sampling of B, G, R values from face photographs of different age groups i.e., young, medium, and old, race groups i.e., white, black, and Asian and genders from the FERET database and the PAL database. The overall sample size is 245057, of which 50859 are skin samples and 194198 are non-skin samples.

TABLE 1. SHOWS THE RECORDS AND DIMENSION OF EACH DATASETS.

Dataset	Dimensions	No. of Points
Satellite	36	6435
Shuttle	9	49097
Skin	4	245057

Our studies have been performed using a random forest model and LSTM using the datasets described above. We measure the accuracy score of these datasets while increasing the number of trees, checked with 3 trees to equate the accuracy score with the OOB score. We've also measured the running time of each tree. The random forest forecast approach takes the test set along with the trained tree as input and returns the maximum vote of trees in the forest, weighted by their probability estimates. Thus, the expected class is the one with the largest average tree-wide likelihood estimation.

VI. EXPERIMENTAL RESULT

Our goal to compare the model accuracy and the OOB score in table 2 - 4 we list trees and how much time-taken to predict the tree. We use 3 trees for each dataset and compared the result.

TABLE 2. SATELLITE DATASET RUNTIME COMPARISON

Dataset	No. of Trees	Run Time(min)
Satellite	1	6.39
	2	13.69
	3	26.80

TABLE 3. SHUTTLE DATASET RUNTIME COMPARISON

Dataset	No. of Trees	Run Time(min)
Shuttle	1	12.74
	2	29.03
	3	41.30

TABLE 4. SKIN DATASET RUNTIME COMPARISON

Dataset	No. of Trees	Run Time(min)
Skin	1	15.52
	2	31. 13
	3	43. 35

Table 5-7 displays the training, test accuracy and OOB score. Here we note that the model measures findings that are not educated for of decision tree in the forest and aggregates over all so that there should be no prejudice, hence the name out-of-bag. As regards the OOB score as an estimation of the accuracy of the test, even though each tree in the forest is trained on a subset of training data, all training data is also used to construct the forest.

TABLE 5. SATELLITE ACCURACY AND OOB SCORE COMPARISON

Dataset	No. of Trees	Accuracy	OOB Score
Satellite	1	33.47	37
	2	34.32	39
	3	35.31	41

TABLE 6. SHUTTLE ACCURACY AND OOB SCORE COMPARISON

Dataset	No. of Trees	Accuracy	OOB Score
Shuttle	1	13.39	15
	2	14.29	21
	3	15.21	25

TABLE 7. SKIN DATASET ACCURACY AND OOB SCORE COMPARISON

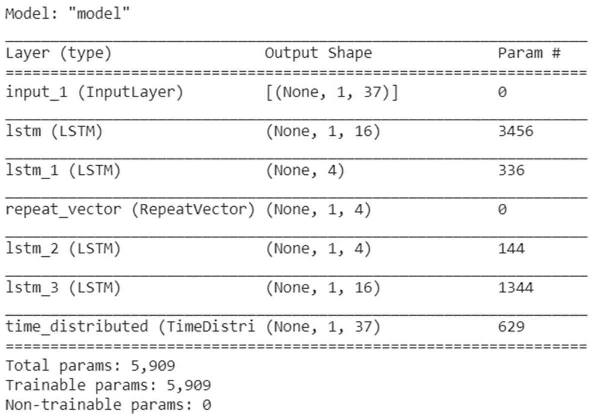
Dataset	No. of Trees	Accuracy	OOB Score
Skin	1	18.14	20
	2	19.21	23
	3	19.94	25

The outcome of this experiment was that the random forest worked well, if we increase the trees, it gives better accuracy and OOB score. The OOB allows less generalization error and prevents over-fitting than the normal decision tree. It's a stronger metric to verify a random forest classifier than standard accuracy.

We finish the pre-processing of our data for the LSTM model experiment, we will first normalize it to a range between 0 and 1. We then reshape our data into a format that is appropriate for an LSTM network input. LSTM cells consider the shape [data samples, time steps, attributes] to have a 3 dimensional tensor.

The first couple of neural network layers construct the compact representation of input data, the encoder, in the LSTM autoencoder network architecture. To spread the compact representational vector through the decoder's time measures, we then use a repeat vector layer fig 3. The decoder's final output layer gives us the restored input data.

Using Adam as our neural network optimizer, we then instantiate the model and compile it and mean absolute error for calculating our loss function.



```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 1, 37)]           0
_________________________________________________________________
lstm (LSTM)                  (None, 1, 16)             3456
_________________________________________________________________
lstm_1 (LSTM)                (None, 4)                 336
_________________________________________________________________
repeat_vector (RepeatVector) (None, 1, 4)              0
_________________________________________________________________
lstm_2 (LSTM)                (None, 1, 4)              144
_________________________________________________________________
lstm_3 (LSTM)                (None, 1, 16)             1344
_________________________________________________________________
time_distributed (TimeDistri (None, 1, 37)             629
=================================================================
Total params: 5,909
Trainable params: 5,909
Non-trainable params: 0
```

Fig. 3: LSTM model summary

Finally in fig 4, we fit the model to our training data and train it for 10 epochs. To evaluate our model's success, we then plot the training and test accuracy.

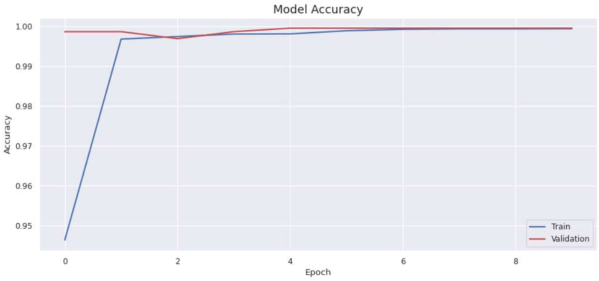


Fig. 4: LSTM Training and Test Accuracy

Fig 5, shows the training and test loss for 10 epochs.we then plot the training and test losses the train and test loss was reduced to 0.005.

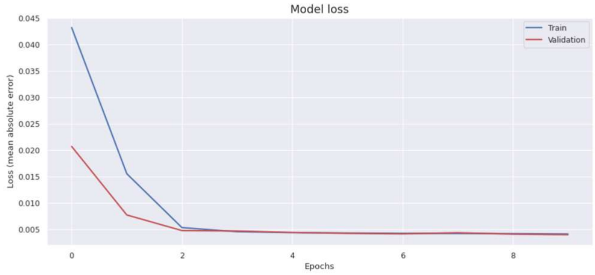


Fig. 5: LSTM Training and Test loss(mae)

VI. ADVANTAGES AND DISADVANTAGES OF LSTM

1. It has a cell state memory unit that has memory that can store previous time stage data from which it can learn well.
2. It's able to withstand noise.
3. By utilizing the Dropout layer, it prevents overfitting.
4. For sequential sensor data for anomaly detection, it is very useful.
5. The ability to use multivariate features is one of the benefits of using LSTM cells.
6. It takes time for the algorithm to train the model.
7. It takes more memory than machine learning models to practice.

VII. APPLICATIONS OF LSTM

1. The learning of grammar
2. Prediction of time series
3. Anomaly Detection of Time Series
4. Compositing songs
5. Predicting diagnostics for wellbeing
6. Predicting inventory

VIII. CONCLUSION

In this article, to estimate the outlier, we proposed a random forest implementation and LSTM model. When we have a small dataset, the OOB score used in random forest is useful and separating the data collection into training and validation set takes out some useful data that can be used to train the models. Thus, by taking the samples that were not used for the model testing, we chose to use the training data as the validation package. We also used some rows for each tree which were not used to train the tree. Thus, for each study, it only considered trees that did not use that sample to train themselves when testing training data for prediction. We found from our experiment that when we have tiny datasets, the OOB score is beneficial. On average, since we use less trees to get the predictions for each study, the OOB score indicates less generalization than the test score. In general, we get a higher predictive accuracy score as the number of trees increases. We get marginally less reliable results if we use less trees than we have available. Using the out-of-bag calculation, thus, eliminates the need for a separate test range. The benefit of this approach needs relatively little estimation and helps us to assess the model as it was trained.

The reason for using this LSTM architecture is that we train the model and determine the resulting reconstruction error on the "ordinary" data. Then, as the model finds data that is outside the standard and tries to replicate it, the reconstruction error(loss) will increase since the model has never been trained to reproduce items from outside the standard correctly. The LSTM network is good by this experiment conclusion, and training the network is better than the other models of machine learning. On the other hand, the Random Forest behaved well with labels for structured data, and the LSTM model can work well on the sequential data.

VIII. REFERENCES

[1] C. Aggarwal, "Outlier analysis", Data Mining, November 25, 2016, pp. 185-215.

[2] Ho, T. K. (1995). Random decision forests. In Document analysis and recognition, 1995, Proceedings of the third international conference, Montreal, Quebec, Canada . pp. 278–282.

[3] Song, Yan-Yan & Lu, Ying. (2015). Decision tree methods: applications for classification and prediction. Shanghai archives of psychiatry, 2015 Apr 25

[4] Vijay Kotu and Bala Deshpande,"Predictive Analytics and Data Mining",Morgan Kaufmann Publication,2015,pp.63-163

[5] Antonio Criminisi,Jamie Shotton annd Ender Konukoglu,"Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning", ACM Digital Library,Foundations and Trends® in Computer Graphics and Vision,February 2012

[6] Misha Deni,David Matheson and Nando de Freitas,"Narrowing the Gap: Random Forests In Theory and In Practice",International Conference on Machine Learning (ICML),4 Oct 2013

[7] Biau, G." Analysis of a random forests model",Journal of Machine Learning Research, 13:1063–1095, 2012.

[8] L. Breiman,"Bagging predictors", Machine Learning, 24:123–140, 1996.

[9] L. Breiman,"Random forests", Machine Learning, 45:5–32, 2001.

[10] L. Breiman,"Consistency For a Simple Model of Random Forests",UCBerkeley, 2004.

[11] Silke Janitza,Roman Hornung,"on the overestimate of the random forest ooberror",Plos One,August 6, 2018.

[12] Mitchell MW,"Bias of the Random Forest out-of-bag (OOB) error for certain input parameters",Research Gate,January 2011,pp.205–211

[13] Leo Breiman,"Out-Of-Bag Estimation",UC Berkeley Statistics.

[14] Dr. Vegard Flovik's, "Machine learning for anomaly detection and condition monitoring" 23 April 2019

[15] Tom Bylander ,"Estimating Generalization Error on Two-Class Datasets Using Out-of-Bag Estimates",Springer Link,July 2002

[16] Matthew W. Mitchell,"Bias of the Random Forest Out-of-Bag (OOB) Error for Certain Input Parameters",Scientific Research,October 2011

[17] Ashwin Srinivasan," Statlog (Landsat Satellite) Data Set",UCI Machine learning Repository

[18] Jason Catlett,"Statlog (Shuttle) Data Set", UCI Machine learning Repository

[19] Rajen Bhatt, Abhinav Dhall, " Skin Segmentation Data Set", UCI Machine learning Repository

[20] Sam T,"Entropy: How Decision Trees Make Decisions",TowardsDatascience,Jan 10, 2019

[21] "Information gain in decision trees",Wikipedia the free encyclopedia,https://en.wikipedia.org/wiki/Information_gain_in_decision_trees.

[22] "Bootstrap Aggregating",Wikipedia the free encyclopedia, https://en.wikipedia.org/wiki/Bootstrap_aggregating

[23] Jason Brownlee ," Bagging and Random Forest Ensemble Algorithms for Machine Learning", April 22, 2016

[24] Sepp Hochreiter and Jurgen Schmidhuber, "Long- Short-term memory" Neural computation 9(8): 1735- 1780, 1997

[25] Michael Phi, "Illustrated guide to LSTM's and GRU's: A step by step explanation". Sep 24, 2018