Usage of Machine Learning to reduce Fraudulent Audits

Gokulkumar Krishnakumar
Stevens Institute of Technology

Nick Benelli
Stevens Institute of Technology

Samruth Vennapusala
Stevens Institute of Technology

***Abstract*—It is proposed that machine learning can be used to help auditing firms by creating an automation system that allows for deep learning, which allows firms to produce accurate audits and efficient speeds. By using various classification methods it is hypothesized that such a program can be created. Success in doing so will help create more accurate audit reports which in turn will help protect more people from fraudulent companies.**

I. Introduction

With the over reliance on companies to provide customers with goods, fraudulent practices start to have bigger and bigger consequences. In order to prevent that, auditing firms are created in order to help keep in check other companies. To Audit is to, in essence, to collect and analyze a company's financial statements in accordance to standard procedure. The auditing firm will then give the company their findings, alongside any discrepancies for the parent company to fix. Failure to fix said discrepancies is usually met with government intervention. As stated before this is an extremely important process so failure to accurately assess a company would have severe consequences. As a result there have been efforts to see if it is possible to increase accuracy.

The goal of the research is to help the auditors by building a classification model that can predict the fraudulent firm on the basis of the present and historical risk factors. The information about the sectors and the counts of firms are listed respectively as Irrigation (114), Public Health (77), Buildings and Roads (82), Forest (70), Corporate (47), Animal Husbandry (95), Communication (1), Electrical (4), Land (5), Science and Technology (3), Tourism (1), Fisheries (41), Industries (37), Agriculture (200).

The Data set has 18 attributes dubbed risk factors. Many risk factors are examined from various areas like past records of audit office, audit-paras, environmental conditions reports, firm reputation summary, on-going issues report, profit-value records, loss-value records, follow-up reports etc. After an in-depth interview with the auditors, important risk factors are evaluated and their probability of existence is calculated from the present and past records. The data has been taken from a one year non- confidential year from 2015 to 2016 collected from the Auditor Office of India.

One way to solve classification problems within machine learning is by using decision trees. Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a data set, branches represent the decision rules and each leaf node represents the outcome. By constructing an accurate tree you give the program a highly accurate means to predict future behavior. However it must be noted that the decision tree is fairly static when it comes to attributes. However auditing does deal with fairly static attributes so working with Decision Trees fits well here. To help remove some of these limitations we can instead use Random Forests which act as multiple Decision Trees to help obtain more accurate results. If all else fails Ensemble methods can also be used to help train the model.

II. Related Work

The closest solution other than machine learning is traditional automation. That is using traditional programming strategies to create a program that is fed data and outputs the correct audit output. While this solution is efficient at doing what it needs to do, it lacks the ability to learn. If some new type of company or fraud comes up that a traditional auditing program does not understand or know it won't be able to correctly audit the information. This is where machine learning comes in. With its ability to learn from past and current data we can effectively remove this drawback from traditional automation. Now if new information is presented, while it will fail at first, as it keeps using said information it will start to learn what to do in those situations, sort of like how a human behaves.

III.Our Solution

A. Description of Dataset

The dataset is compiled by the Comptroller and Auditor General (CAG) of India. CAG is the audit office of the Indian government tasked with tracking receipts and expenditures of the national government and India's local governments. They are in charge of managing the finances (balance sheets, trading, manufacturing) of different departments in the government, and overseeing government grants and loans.

The dataset was acquired from the UCI Machine Learning repository website and then imported into our notebook. We received 2 sets of data from this source. Audit Data and the Trial data. We will be using the Audit Data for the majority of this project. The shape of the Audit Data set is 27 columns and 776 rows. We have variables that are float types, int types and object types. Here is a brief summary of all the variables.

Sector Score – Historical risk score value for each target sector

LOCATION ID – Unique ID of the city or province

PARA A - Discrepancy found in the planned expenditure of inspection and summary report A in Rs (in crore)

SCORE A, RISK A – These columns are formulaic scores that can be derived from the PARA A PARA B - Discrepancy found in the unplanned expenditure of inspection and summary report B in Rs (in crore)

SCORE B, RISK B - These columns are formulaic scores can be derived from the PARA B. TOTAL - Total amount of discrepancy found in other reports Rs (in crore) numbers - Historical discrepancy score

Score B.1, Risk C - These columns are outputs that can be derived from the numbers

Money Value - Amount of money involved in misstatements in the past audits

SCORE MV, Risk D - These columns are formulaic scores that can be derived from the Money Value

District Loss - Historical risk score of a district in the last 10 years

PROB – probability of District Loss

Risk E – It is the product of District Loss and PROB

History - Average historical loss suffered by firm in the last 10 years

Prob – Probability of Historical Loss score

Risk F – It is the product of History and prob

Score – It is a deciding factor in classifying a firm as 'Fraud' or 'Not Fraud', In the trial file, if the score is less than or equal to 2, the firm is labelled 'Not Fraud', if it is greater than 2, it is labelled 'Fraud'

Inherent Risk - this is an output of the risk present due to the discrepancies present in the transactions

CONTROL RISK - this is an output of the risk due to the discrepancies which are left undetected by an internal control system.

Detection Risk - this is an output of risk of discrepancies present in the firm which are not even detected by the audit procedures

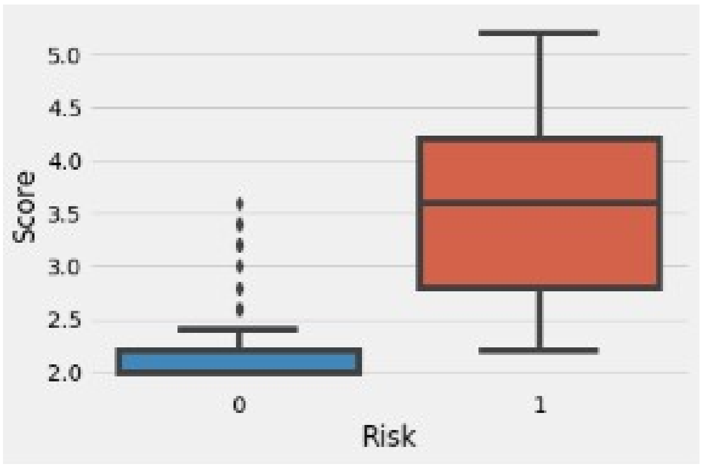
Audit Risk – this is an output of the product of Inherent, control and detection risks

Risk – This is a binary output dependent on Audit Risk, If the Audit Risk is less than or equal to 1, the firm is labelled 'Not Fraud' and if it is greater than 1, it is labelled 'Fraud'.

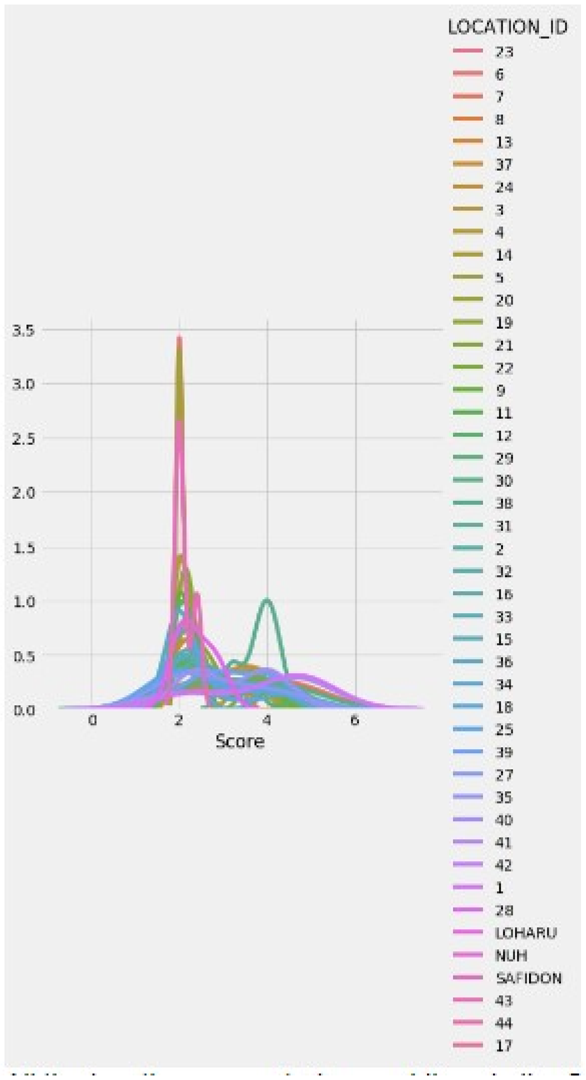
The dataset includes several inputs and outputs. "Score", "Inherent Risk", "Control Risk", "Detection Risk", "Audit Risk", and "Risk" are all outputs derived formulaically by CAG. The goal of the project is to take the input values and predicted the proper binary risk output. Therefore, other nonbinary outputs will be ignored. The input columns used in the dataset are "Sector Score", "Location ID", "Para A", "Para B", "numbers", "Money Value", "District Loss" and "History". These data points build the output variables. They provide a good representation of the current status, and historical status of the business. These factors will determine if the business in question is an audit risk (1) or not an audit risk (0.) If the business is an audit risk it should be further investigated by forensic accountants. The model will allow CAG to deploy its audit teams wisely without wasting resources on non-fraudulent businesses.

Pre-process of Data

We looked at the head and the tail of the data to make sure all the variables and rows are visually correct. Then we look at what type of variables we are dealing with. We look at their D-type and if it is a quantitative variable or a categorical variable. Then we process the quantitative variables in a correlation matrix to see if the variables are independent and the correlation. Following this we do more exploratory data analysis to makes sure we get a better understanding of the variables and see what data we want to keep and which variables we want to drop so that we can get the best possible result from our model.



It is a clear trend that a lower score is more likely to get labeled a risk.



All the locations seem to be providing similar Score outputs. There is no distinguishing pattern for a certain location.

After preprocessing, it was decided that locations will be ignored as inputs in the dataset. Categorical data is not always optimal for classification algorithm. Since there is no clear relation between audit risk and location, it will be ignored.

Finally, any missing values in the dataset will be replaced with the mean value of the dataset. This decision will allow for all instance in the dataset to be used. Removing blank data from the dataset can lead to a reduced size which may lead to overfitting.

B. *Machine Learning Algorithms*

There is no shortage of classification algorithms that can be used on the binary classifier problem. Decision trees, random forests, AdaBoost, neural networks will all have advantages and disadvantages to solving this problem.

Decision Tree

The data contains numerous independent input variables where the target function returns a discrete output variable. The decision tree will take the quantitative data and create binary thresholds. The model will use those thresholds to make classification decisions. Those thresholds are “nodes”, and the inputs are considered “branches.” The nodes are ranked by importance to make a hierarchical structure called a decision tree. This hierarchical structure allows for the model to be easily visualized. Classification datasets tend to have human errors and misclassifications. When a decision tree is simplified and pruned, it will reduce overfitting of the data. By reducing overfitting, the decision tree can handle misclassifications in the dataset and still produce a high accuracy.

Random Forest

”Wisdom of the crowd” is one of the strongest concepts in data science. This theory states the collective group will be “smarter” than one individual expert in terms of predicting. Instead of choosing one of four experts’ predictions, it would be more accurate to average the 4 expert’s predictions. Ensemble methods are built of this theory and results can be averaged to yield a more accurate result. Random Forest is an ensemble method and an offshoot of the decision tree. The random forest algorithm creates a large sample of pruned decision trees. Aggregating these trees will lead to a reduced variance without increasing bias. Along with a large number of trees, the bagging method will reduce overfitting by allowing data to be reused during training. In turn a better model can be created that any one tree.

AdaBoost

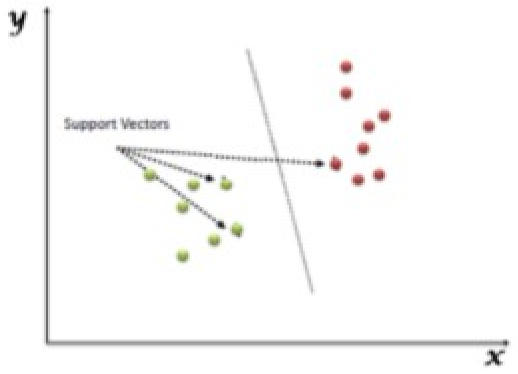
AdaBoost (Adaptive Boosting) is another Ensemble method where several weak learns can create one strong learner. AdaBoost was created in the early 21st Century, but it won the 2003 computer science Gödel Prize. The AdaBoost algorithm is most commonly used in conjunction with decision trees and random forests. AdaBoost tends to use 1 node decision trees which are considered weak learners. Unlike a random tree forest (that equally weights each tree), AdaBoost weights a tree based on its accuracy. More accurate trees will have a large say in the final outcome. The final difference between AdaBoost and random forest is the order in which trees are made. Random forest randomly and independently creates each tree. AdaBoost creates the first tree randomly and independently, but each subsequent tree is influenced by the error of previous trees. This is done by weighted error rate and predictor weight. These values are constantly being updated in each iteration.

Neural Networks

The human brain classifies data constantly throughout the day. The neural network is designed to mimic the human brain and classify information in the dataset. The neural network will have three layers, an input layer, an output layer, and one hidden layer. Each input parameter is considered a neuron. The neurons are weighted and summed and that weighted summed is put into a threshold function. The threshold function makes a determination (true, false) based on the weighted sum input. The weights will start with an initial guess, then slowly be corrected through backpropagation. The neurons in this case will be the risk factors like location, paras, valuations, profits, score formulas.

SVM

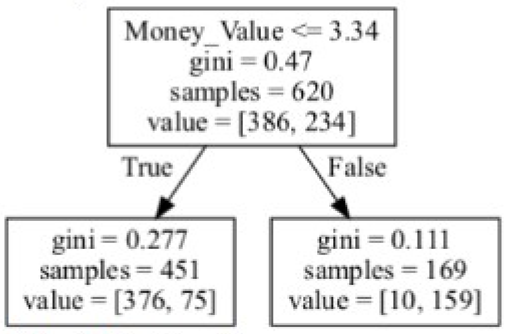
Support Value Machine algorithm is a supervised machine learning algorithm that can be used for both classification or a regression problem. Most of the time however SVM is used for classification problems. In this algorithm we plot each of the data points in n-dimensional space with the value of each feature being the value of a particular coordinate. Then you perform a classification by finding the hyper-plane that differentiates the classes very well. An example of these linear hyper-planes would look as such.



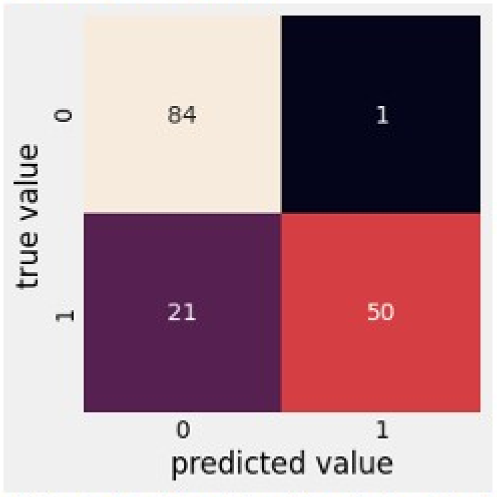
SVM can also work for more complex non linear data sets. We can use different shapes as hyperplanes. For example a circle (z = x² + y²) can be a hyperplane to classify items inside and outside the circle. When solving a classification problem it is very important to pick the right hyperplanes to result in the best possible result.

C. Implementation Details

First, the data was split into testing data and training data. 80% of the 776 rows will be used to train the model, while the other 20% will be used for testing. The predicted outputs will be compared to the actual outputs of the testing data. Accuracy will be measured with different methods. A simple accuracy score will determine the%age of predicted values exactly match the actual values. A confusion matrix will determine the number of true negative, false positive, false negative, and true positive values are predicted. The confusion matrix can then calculate precision (true positive: predicted positive values), recall (true positive: actual positive values), and F1 score (combination of precision and recall).



1 Node Decision Tree

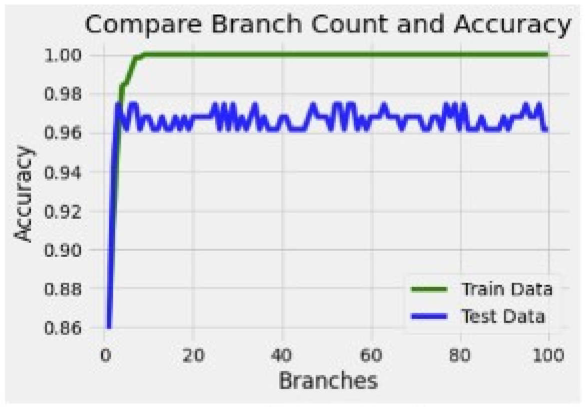


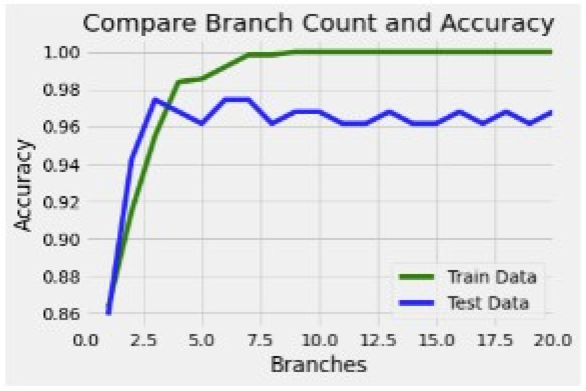
1 Node Decision Tree Confusion Matrix

Decision Tree

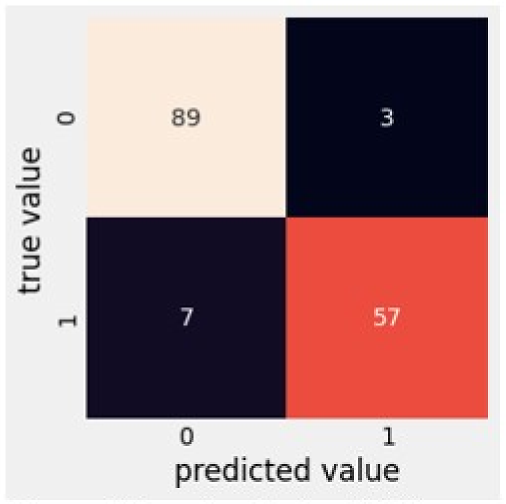
First, a simple 1 node decision tree (decision stump) was created to find the strongest input variable and its accuracy. The tree's node is the "Money Value" with a threshold of 3.34. If the money value is greater or equal to 3.34 the tree will classify the instance as an audit risk. A money value less than that will be classified as not an audit risk.

The overall accuracy of the simple stump was 86 %, with a precision of 88% and recall of 86%. This is promising because this is considered a weak classifier. The model struggles the most with predicting audit risks to falsely be non-audit risk. Any classification algorithm for this dataset will likely use Money Value as its strongest predictor. Next, the number of branches can be compared to the accuracy of the model. The test data and train data are plotted simultaneously. This plot illustrates as the number of branches increases the test data and train data grow in accuracy until the two diverge around 6 nodes.

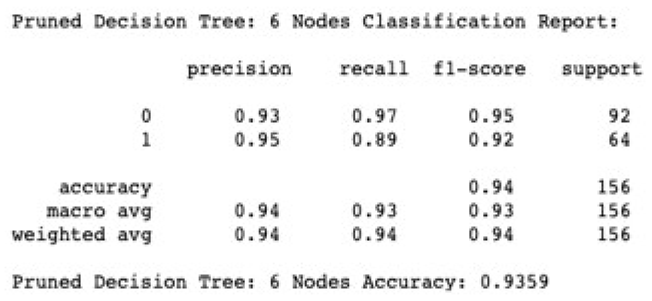




This divergence gives the optimal number of nodes that can be used for a pruned decision tree. Any more nodes and the model will most likely overfit the data and be a bad predictor for the testing data and any new data that is introduced. A Grid Search CV uses cross validation to find the optimal criterion information gain function between gini impurity and entropy. The search yielded gini impurity measure the frequency of mislabeled data instead of measuring correctly labeled data and mislabeled data like entropy.



Pruned Tree Confusion Matrix

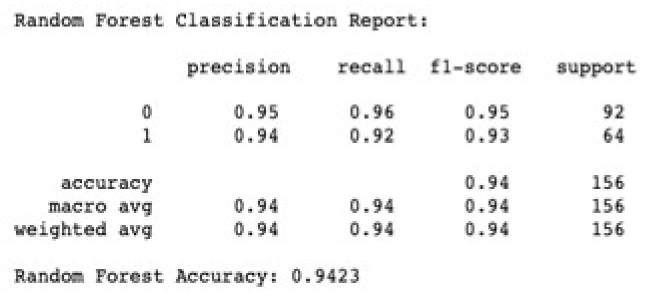


```
Pruned Decision Tree: 6 Nodes Classification Report:

              precision    recall  f1-score   support

           0       0.93      0.97      0.95        92
           1       0.95      0.89      0.92        64

    accuracy                           0.94       156
   macro avg       0.94      0.93      0.93       156
weighted avg       0.94      0.94      0.94       156

Pruned Decision Tree: 6 Nodes Accuracy: 0.9359
```

The newly pruned tree's main node is still Money Value, but the threshold is only 3.165 now. The pruned tree is very accurate with a precision, recall, and f1 score of 97%. There might be some overfitting with this tree. This can be solved by using these parameters in a random forest classifier.

Random Forest

The random forest uses the same parameters of the max nodes being 6 and the measure being the gini impurity, but it will create trees of different shapes and sizes. The trees will be averaged together to get a final predictor. 50 tree estimators were chosen in the model. The number can be higher, but it is more computationally expensive. The model predicted the testing data with a precision, recall, and f1 score of 94%.



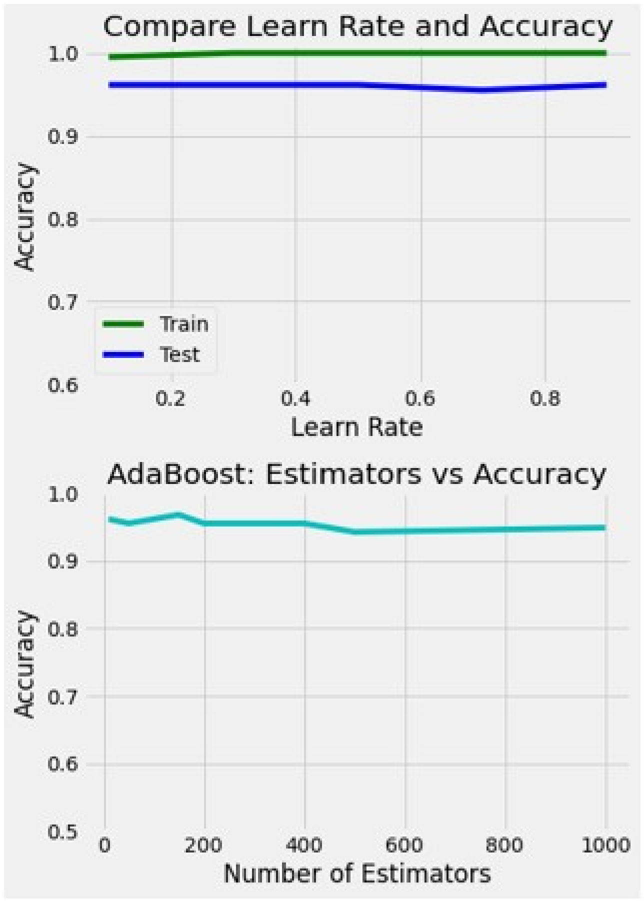
```
Random Forest Classification Report:

              precision    recall  f1-score   support

           0       0.95      0.96      0.95        92
           1       0.94      0.92      0.93        64

    accuracy                           0.94       156
   macro avg       0.94      0.94      0.94       156
weighted avg       0.94      0.94      0.94       156

Random Forest Accuracy: 0.9423
```

This might be slightly lower that the one pruned decision tree, but this model decreases the variance without increasing the bias. This means there is less overfitting in the data. The random forest can be experimented with different max nodes (between 6 and 11) and a different number of estimators.

AdaBoost

AdaBoost builds upon the methods of the decision tree and random forest. The classifying method, the number of estimators, and the learn rate are the main hyperparameters in the algorithm. The Stagewise Additive Modeling using a Multiclass Exponential Loss Function (SAMME) is used to make the classifications in the algorithm. First the AdaBoost model was plotted with a decision tree of 1 node with a learn rate of 0.5.

The model's hyperparameters learn rate and estimators should be kept around 0.5 and 200. Although the data does not show a major change, too big of a learn rate would open up the threat of the model not converging. To many estimators can also lead to overfitting.

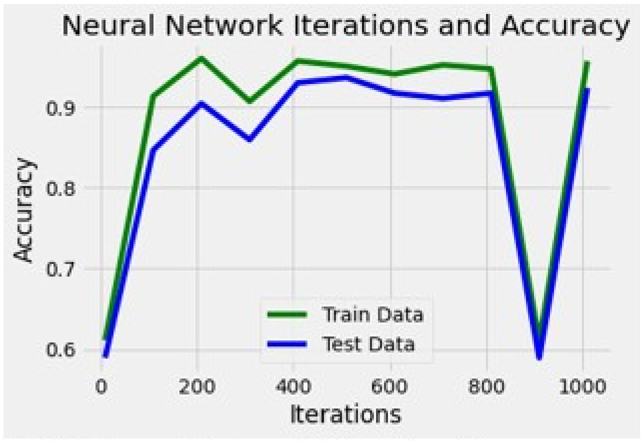


Plot 1: AdaBoost classifier accuracy with varying learn rates. Plot 2: AdaBoost classifier accuracy with varying estimators.

The AdaBoost algorithm can be used in conjunction with a pruned decision tree or a random forest classifier. This will be more computationally expensive and the model might be more likely to overfit. The overall accuracy of the AdaBoost is around 96 percent with a single node decision treen and 98 percent with a random forest. These results are extremely promising.

Neural Network

The same data can be used to implement a neural network MLP classifier with the data to build a model. Only 1 hidden layer will be used, because it is a binary output model. Cross entropy can be used to find the best activation function, regularization term (alpha), learning rate, and weight optimization solver. Next, the accuracy of the model can used to plot iterations of optimized model versus the accuracy.



MLP Neural Network Classifier Max Iterations vs. Accuracy Score

The model seems to be optimized around 200 iterations. Any more iterations will be computationally too expensive and the date might be overfitted.

SVM

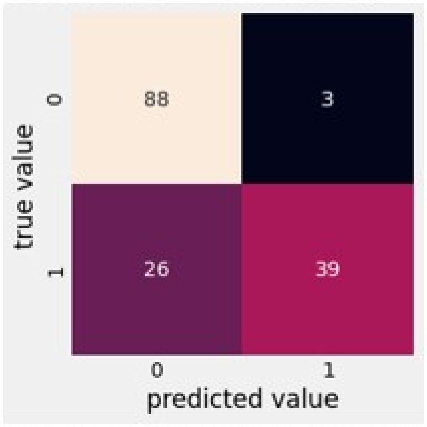
We used the same 6 variables to predict the Risk using SVM. We use the same training and testing sets to formulate and test our classifications. We initially performed a linear SVM method and get a 96.15% accuracy. Then we try a non-linear hyperplane to see if we can receive an even better result. We use Polynomial SVM method that uses a polynomial hyperplane to receive a different result. The polynomial SVM model receives a 99.36% accuracy which is better than the linear model. So we can say that a non-linear hyper plane was better for classification in this data-set.

IV. Results

Decision Tree and Random Forest:

Decision Tree and Random Forest

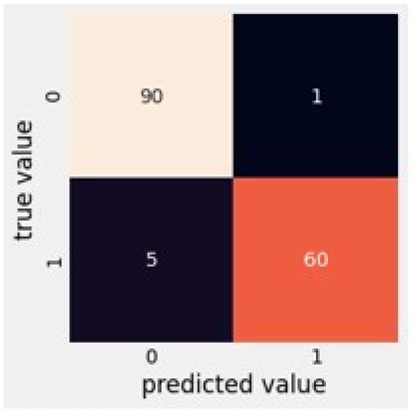
```
Decision Tree: 1 Node  Classification Report:

              precision    recall  f1-score   support

           0       0.77      0.97      0.86        91
           1       0.93      0.60      0.73        65

    accuracy                           0.81       156
   macro avg       0.85      0.78      0.79       156
weighted avg       0.84      0.81      0.80       156

Decision Tree: 1 Node  Accuracy: 0.8141
```



1 Node Decision Stump Accuracy Report

The simple 1 Node decision stump is a strong predictor for a “weak learner.” The algorithm fails mainly when classifying true values as false values. This would likely mean there are companies that have lower Money Values, but are still audit risks.

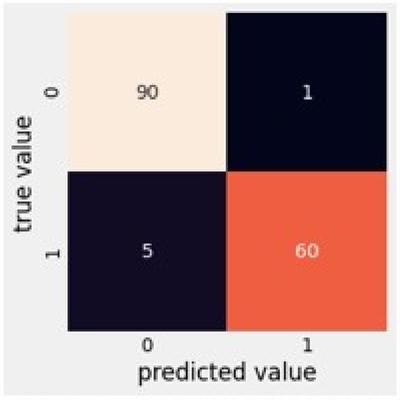
```
Decision Tree: No Max Nodes Classification Report:

              precision    recall  f1-score   support

           0       0.95      0.99      0.97        91
           1       0.98      0.92      0.95        65

    accuracy                           0.96       156
   macro avg       0.97      0.96      0.96       156
weighted avg       0.96      0.96      0.96       156

Decision Tree: No Max Nodes Accuracy: 0.9615
```



43 Node Decision Tree Accuracy Report

A decision tree with 43 max nodes has an accuracy precision, recall, and F1 score of 0.96. The overall accuracy is very high which is both a good and bad thing. Obviously, it is very good as our model is producing accurate results but the overly high accuracy may be due to overfitting which causes our model to be overly static. To make sure we shall prune our original decision tree and test again.

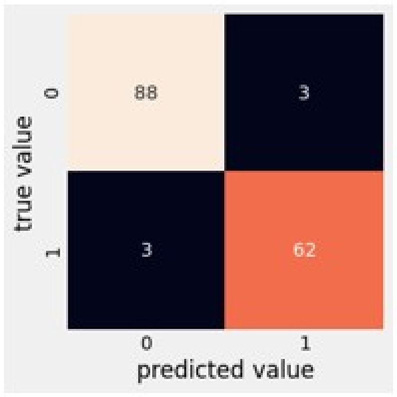
```
Pruned Decision Tree: 6 Nodes Classification Report:

              precision    recall  f1-score   support

           0       0.95      0.99      0.97        91
           1       0.98      0.92      0.95        65

    accuracy                           0.96       156
   macro avg       0.97      0.96      0.96       156
weighted avg       0.96      0.96      0.96       156

Pruned Decision Tree: 6 Nodes Accuracy: 0.9615
```



Pruned (6 Node) Decision Tree Accuracy Report

The accuracy of a pruned tree is similar to a non-pruned tree. There is less overfitting in a pruned true and accuracy did not need to be sacrificed. A random forest ensemble method can decrease the variance without increasing bias and further handle overfitting.

```
Random Forest Classification Report:

              precision    recall  f1-score   support

           0       0.97      0.97      0.97        91
           1       0.95      0.95      0.95        65

    accuracy                           0.96       156
   macro avg       0.96      0.96      0.96       156
weighted avg       0.96      0.96      0.96       156

Random Forest Accuracy: 0.9615
```



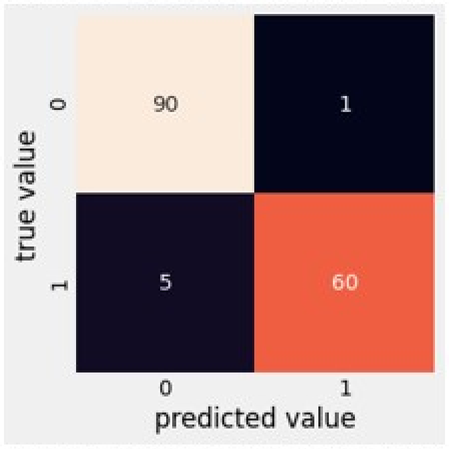
Random Forest Accuracy Report

As we can see here the accuracy is slightly lower than compared to our decision tree values. However, this is due to a reduced variance and similar bias. As a result, this model shall work better compared to the decision tree if more data is to be introduced. Which in turn allows our model to become more dynamic, slightly solving the overfitting issue.

AdaBoost

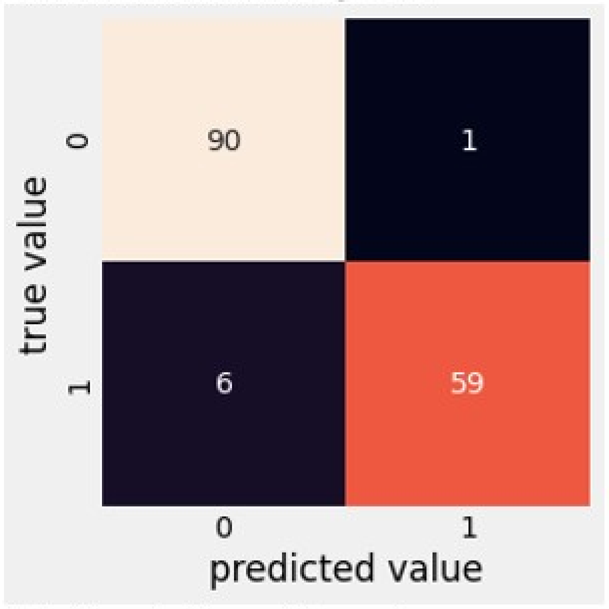
Now that we have our trees let us run Adaboost on them to see if we can improve our results

```
AdaBoost: 1 Node Tree Classification Report:

              precision    recall  f1-score   support

           0       0.95      0.99      0.97        91
           1       0.98      0.92      0.95        65

    accuracy                           0.96       156
   macro avg       0.97      0.96      0.96       156
weighted avg       0.96      0.96      0.96       156

AdaBoost: 1 Node Tree Accuracy: 0.9615
```



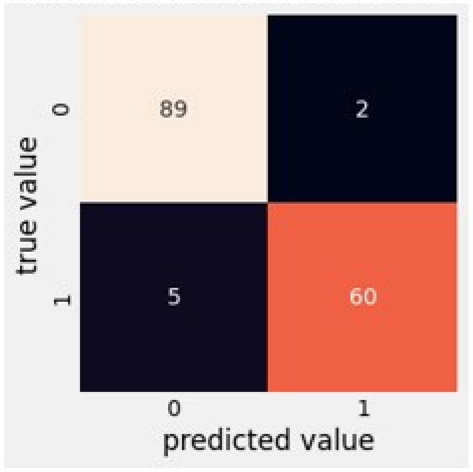
AdaBoost: 1 Node Tree Accuracy Report

```
AdaBoost: Pruned Tree Classification Report:

              precision    recall  f1-score   support

           0       0.94      0.99      0.96        91
           1       0.98      0.91      0.94        65

    accuracy                           0.96       156
   macro avg       0.96      0.95      0.95       156
weighted avg       0.96      0.96      0.95       156

AdaBoost: Pruned Tree Accuracy: 0.9551
```



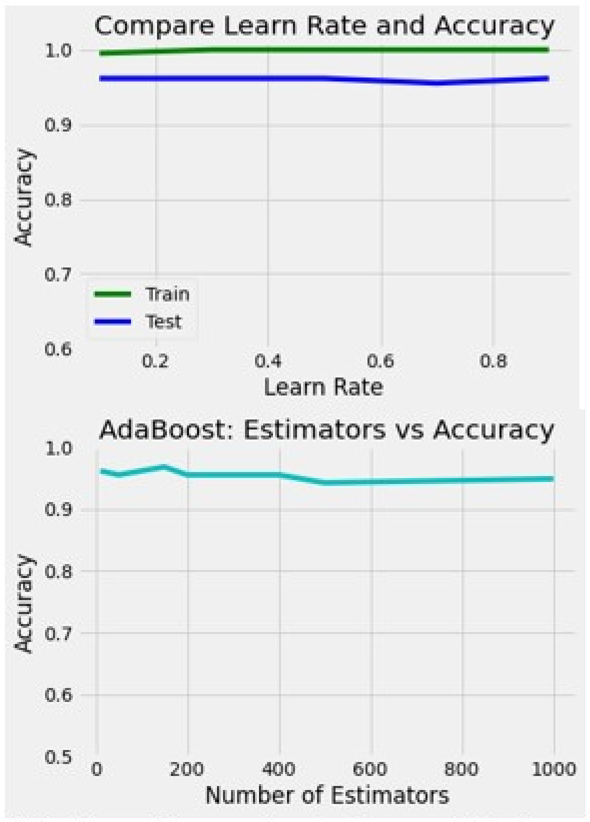
AdaBoost: Pruned Tree Accuracy Report

```
AdaBoost: Random Forest Classification Report:

              precision    recall  f1-score   support

           0       0.95      0.98      0.96        91
           1       0.97      0.92      0.94        65

    accuracy                           0.96       156
   macro avg       0.96      0.95      0.95       156
weighted avg       0.96      0.96      0.95       156

AdaBoost: Random Forest Accuracy: 0.9551
```



AdaBoost: Random Forest Accuracy Report

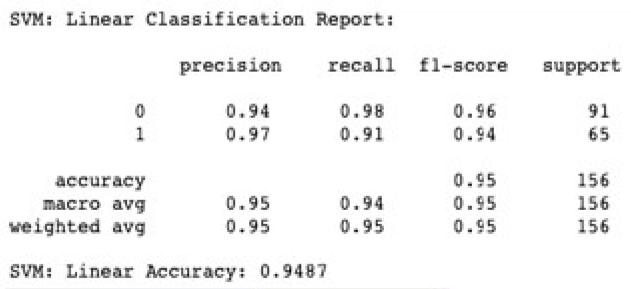
Here we can see a tie between the regular decision tree and the random forest, the only major difference being the types of errors we get. That being said the fear of overfitting is still a valid concern seeing as our accuracy's are still unusually high for these types of models. In fact no matter what learning rate or estimators we choose it seems this value stays the same as evidenced here:



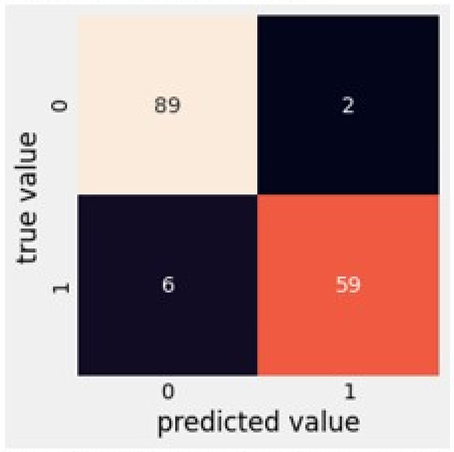
AdaBoost Learning Rate and Estimators vs Model Accuracy

Support Vector Machine:

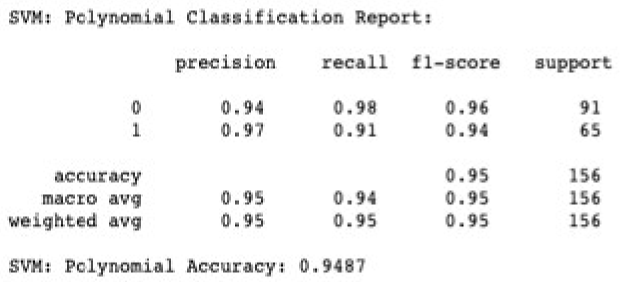
Then we compare SVM to the other methods. We can see that the SVM method is good but the SVM method using the polynomial hyperplane has one of the best results. It made only 1 error with its predictions.



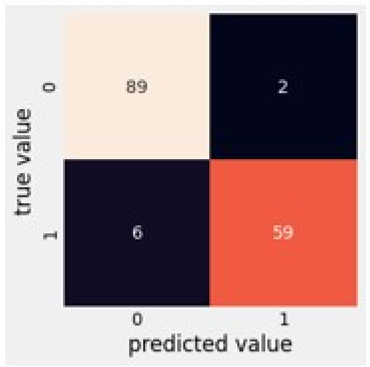
```
SVM: Linear Classification Report:

              precision    recall  f1-score   support

           0       0.94      0.98      0.96        91
           1       0.97      0.91      0.94        65

    accuracy                           0.95       156
   macro avg       0.95      0.94      0.95       156
weighted avg       0.95      0.95      0.95       156

SVM: Linear Accuracy: 0.9487
```



SVM: Linear Accuracy Report



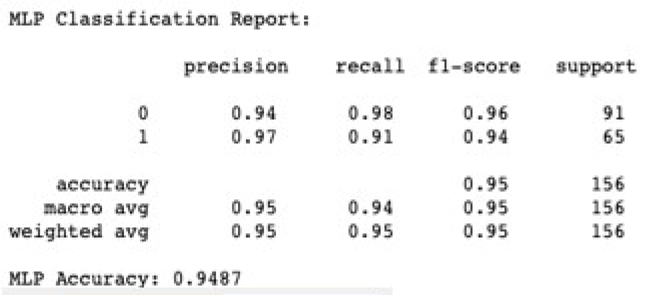
```
SVM: Polynomial Classification Report:

              precision    recall  f1-score   support

           0       0.94      0.98      0.96        91
           1       0.97      0.91      0.94        65

    accuracy                           0.95       156
   macro avg       0.95      0.94      0.95       156
weighted avg       0.95      0.95      0.95       156

SVM: Polynomial Accuracy: 0.9487
```



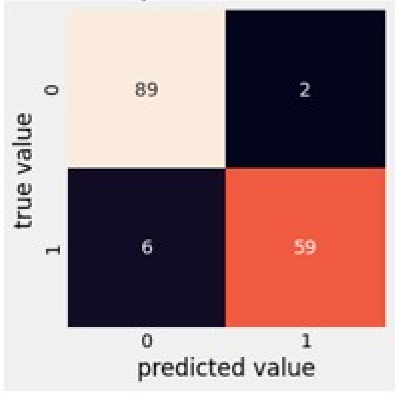
SVM: Polynomial Accuracy Report

Artificial Neural Network MLP

Lastly, we compare the neural network method to the others. The MLP and optimized MLP classifiers have good accuracy's. However there are other models that do a better job of predicting than either on of these.

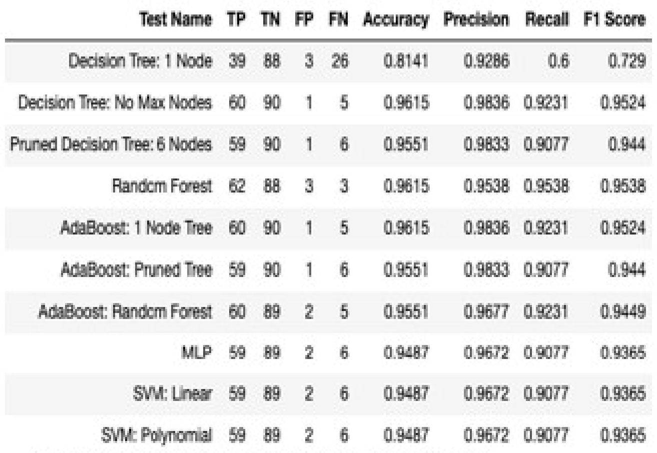


```
MLP Classification Report:

              precision    recall  f1-score   support

           0       0.94      0.98      0.96        91
           1       0.97      0.91      0.94        65

    accuracy                           0.95       156
   macro avg       0.95      0.94      0.95       156
weighted avg       0.95      0.95      0.95       156

MLP Accuracy: 0.9487
```



Neural Network Accuracy Report

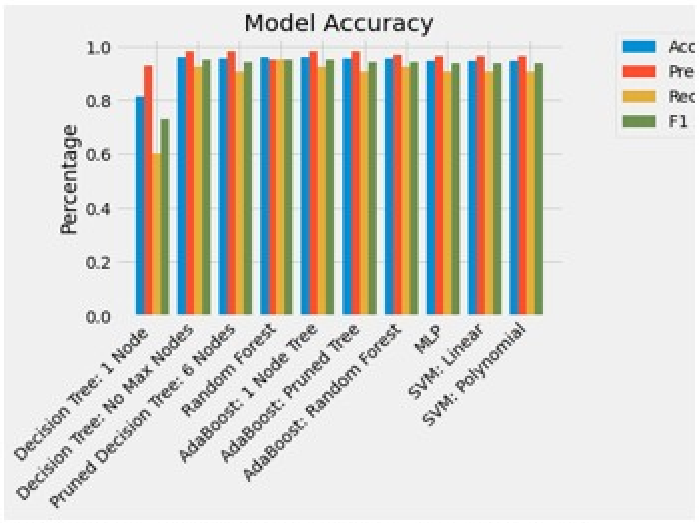
The models' accuracy values can be compiled into a single table for comparison. Since the models use the same testing data, we can compare the True Positive, True Negative, False Positive, and False Negatives.



Test Name	TP	TN	FP	FN	Accuracy	Precision	Recall	F1 Score
Decision Tree: 1 Node	39	88	3	26	0.8141	0.9286	0.6	0.729
Decision Tree: No Max Nodes	60	90	1	5	0.9615	0.9836	0.9231	0.9524
Pruned Decision Tree: 6 Nodes	59	90	1	6	0.9551	0.9833	0.9077	0.944
Random Forest	62	88	3	3	0.9615	0.9538	0.9538	0.9538
AdaBoost: 1 Node Tree	60	90	1	5	0.9615	0.9836	0.9231	0.9524
AdaBoost: Pruned Tree	59	90	1	6	0.9551	0.9833	0.9077	0.944
AdaBoost: Random Forest	60	89	2	5	0.9551	0.9677	0.9231	0.9449
MLP	59	89	2	6	0.9487	0.9672	0.9077	0.9365
SVM: Linear	59	89	2	6	0.9487	0.9672	0.9077	0.9365
SVM: Polynomial	59	89	2	6	0.9487	0.9672	0.9077	0.9365

Table Comparison of Models' Accuracy Values

The table can be converted to a bar graph for comparison. Ultimately the AdaBoost algorithms have the highest accuracy values.



Accuracy, Precision, Recall, F1 Score by Model

V. Future Directions

This project can be expanded upon if given another 6 months to year. Progress can be made on the data and the classifying algorithms.

The data gives multiple inputs and one output if the subject is an audit risk. Being an audit risk does necessarily mean the company has or is going to commit fraud. Further studies would need to be done to determine how many audit risks commit fraud, compared to non-audit risks. The models can then be tweaked to account for actual fraud data.

Next, the model can be applied to different companies in other countries. It would be interesting to see if the model could detect fraud in countries other than India without a large undertaking of completely re-training the model. If the model holds up, audit practices can be standardized around the world. More fraudulent business practices can be caught at a minimal audit cost.

Finally, Naive Bayes Theorem can be applied if more time were given. Naive Bayes Theorem would encompass each classifier weighted by its accuracy from the confusion matrix. The new Naive Bayes classifier model would be able to pull from each accurate model and use the power of ensemble learning.

VI.Conclusion

Going through the various machine learning algorithms it seems the combination that produced the best results was using AdaBoost with the random forest as that produced a 0.95 accuracy rate. However, we cannot 100% claim that this would produce the best results as we had taken some liberties with the data (removing some columns) as well as we believe there is a chance some overfitting is occurring which muddles the model a bit. In this case we believe we can fix these issues in future experiments by normalizing the data before creating the decision tree and running AdaBoost. We believe that doing so will help reduce the overfitting issue which will in turn lead to a more overall accurate model.

References

[18711()] I. 18711. How to adjust the hyperparameters of mlp classifier to get more perfect performance. . Retrieved December 14, 2021, from https://datascience.stackexchange.com/questions/36049/how-to-adjust-the-hyperparameters-of-mlp-classifier-to-get-more-perfect-performa.

[Aurelien()] G. Aurelien. Hands-on machine learning with scikit-learn, keras, and tensorflow concepts, tools, and techniques to build intelligent systems. o'reilly. . (2020).

[Found()] N. A. Found. Classification with neural nets using mlpclassifier. evening session. (2016, october 5). Retrieved December 14, 2021, from https://sdsawtelle.github.io/blog/output/week4-andrew-ng-machine-learning-with-python.html.

[Freund()] . S. R. E. Freund, Y. 2002, may 25). a decision-theoretic generalization of on-line learning and an application to boosting. journal of computer and system sciences. Retrieved December 14, 2021, from https://www.sciencedirect.com/science/article/pii/S002200009791504X.

[Marsland()] S. Marsland. Machine learning: An algorithmic perspective. crc press. (2015).

[Mitchell()] T. M. . Mitchell. Machine learning. MacGraw-Hill.

[(n.d.).()] L. (n.d.). Lavanyabk/fraudulent-firm-detection: Building a predictive model that predicts the audit risk of a firm in the form of a score and this score can in turn be used to classify a firm as fraudulent or otherwise. GitHub. Retrieved December 14, 2021, from https://github.com/lavanyabk/Fraudulent-Firm-Detection.

[of India. CAG of India()] C. of India. CAG of India. Comptroller and auditor general of india. (n.d.). Retrieved December 14, 2021, from https://cag.gov.in/en/page-cag-of-india.

[Quantdare()] Q. Quantdare. Decision trees: Gini vs entropy . Retrieved December 14, 2021, from https://quantdare.com/decision-trees-gini-vs-entropy/.

[Repository()] U. M. L. Repository. Audit data data set. (n.d.). Retrieved December 14, 2021, from https://archive.ics.uci.edu/ml/datasets/Audit+Data.