SANTANDER CUSTOMER TRANSACTION PREDICTION: A NEURAL NETWORKS APPROACH

Cesar Rene Pabon Bernal
Professor Iordan Slavov
Stat 724
City University of New York, Hunter College
Sunday, May 19th, 2019

TABLE OF CONTENTS

SANTANDER CUSTOMER TRANSACTION PREDICTION

Cesar Rene Pabon Bernal
Professor Iordan Slavov
Stat 724
City University of New York, Hunter College

ABSTRACT

Neural Networks is a well documented technique for classification problems,[1] in particular, identifying customer behavior in baking.[2] In this report, we explore a neural network (NN) statical model that accurately predicts whether or not a consumer will make a transaction and benchmark its results with Logistic Regression (LR) and Random Forests (RFs). NN and RFs both were equally accurate in prediction at 90.5% in comparison to LR 82.5%. However, NN was much faster in CPU time versus RFs, 14.5sec and 1min 52sec respectively, and therefore serves as the better model.

INTRODUCTION

Based out of Boston Massachusetts, Santander Bank is a whole subsidiary of a Spanish Santander group.[3] With over $57.5 billion in deposits, it offers a plethora of services which include retail banking, mortgages, corporate banking, credit card, and more. On April 3rd, 2019, Santander published a competition, labeled “Can you identify who will make a transaction?”, and its respective data, on Kaggle.com. They proposed a prize for helping the bank identify which customers will make a specific transaction in the future, irrespective of the amount of money transacted. The data provided for the competition has the same structure as the real data Santander receives on daily bases. Although the competition has expired, our objective in this study, is to utilize the provided data to create NN model to accurately predict a transaction. Python was utilized for the entire project.

DATA

Data was collected from the Kaggle Repository. The training set has dimensions of (200,000 X 201) and the test set of (200,00 X 201). A summary of the data reveals:

1) 201 Features are not labeled
2) There are not missing values
3) No meaningful correlation between variables— The R-value for all 201 features is < 0.5; possible negative correlation

Figure 1 (A) shows the distribution transaction responses for target. 89.951% of the observations are composed of future transactions. This is a problem for our model because it will internalize a bias towards the data points where a future transaction was not made, hindering any acceptable rate of accuracy and grossly misleading metric results. Evidently, the model will not be trained sufficiently and we thus state, presence of an imbalance dataset and explore in detail this important factor.

[1] Hoskins, J. C., et al. “Fault Diagnosis in Complex Chemical Plants Using Artificial Neural Networks.” *AIChE Journal*, vol. 37, no. 1, 1991, pp. 137–141., doi:10.1002/aic.690370112.

[2] Ogwueleka, Francisca Nonyelum, et al. “Neural Network and Classification Approach in Identifying Customer Behavior in the Banking Sector: A Case Study of an International Bank.” *Human Factors and Ergonomics in Manufacturing & Service Industries*, 2012, doi:10.1002/hfm.20398.

[3] “Santander Bank.” *Wikipedia*, Wikimedia Foundation, 14 May 2019, en.wikipedia.org/wiki/Santander_Bank.

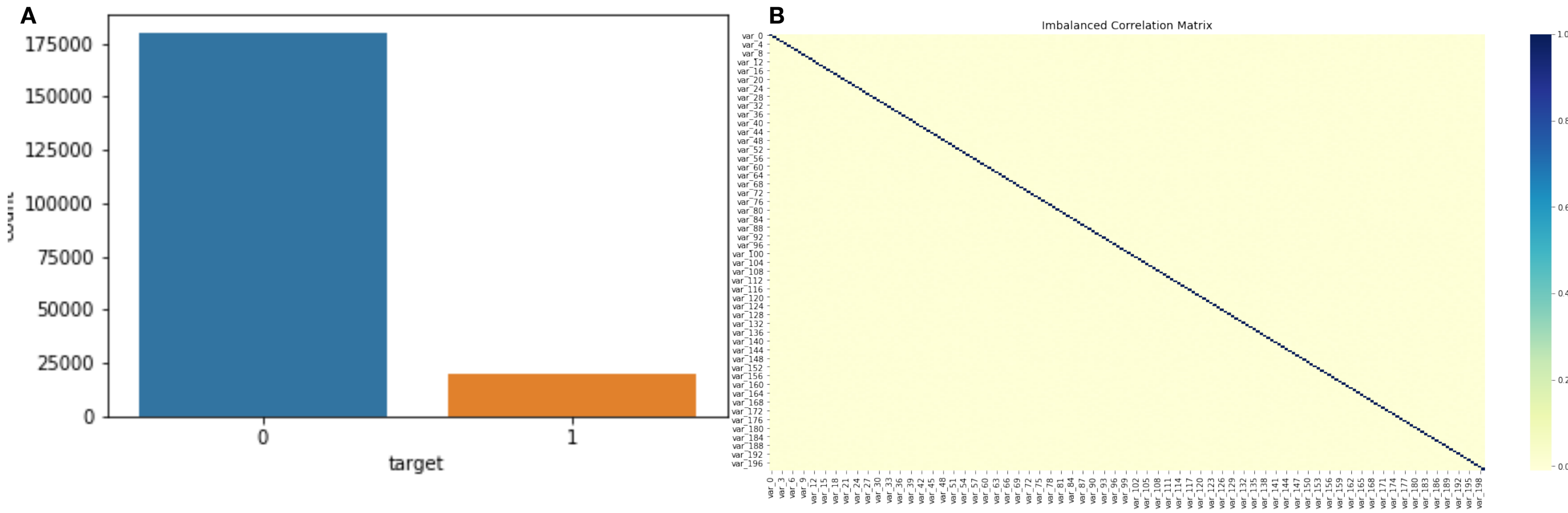

Figure 1 (A) Response for future transactions in data (B) Correlation matrix of 200 variables

I.EXPLORATORY ANALYSIS

There are several ways we may mitigate the effects of this imbalance:[4]

1) When assessing the performance of the models, consulting accuracy alone will not be enough. We will need to look at **precision**, the fraction of the positive predictions that were correct predictions, and **recall/sensitivity**, the fraction of the positive cases that were correctly predicted to be positive. The negative class possesses analogous metrics known as **negative predicted value** and **specificity**, respectively. Additionally, the AUC, the area under the ROC curve, will be important in determining the best overall model. The **AUC** takes into account how well the model predicts both, or all, cases/classes of the response variable. This provides a way to evaluate all of our classification models no matter the distribution of the response variable.
2) **Under & Oversampling**: We could balance out the classes of the response variable by either sampling from the overrepresented class, until we have a number of observations equal to that of the underrepresented class, or we could sample, with replacement, the underrepresented class until we obtain a number of observations equal to that of the overrepresented class. The problem with this is that we are not introducing new data into the model but repeating observations it has seen before. Therefore, training on multiple cases of the same observation runs the risk of overfitting the model.
3) **Adjusting the decision threshold**: Many classification models conclude by providing the probability estimate that an observation belongs to a particular class. The default is to use 0.5, or 50%, as the cutoff for determining how to classify new observations.
4) **Employing an Anomaly Detection framework**: Instead of treating the problem as one of classification, it can be reframed as an anomaly detection. The idea here is that since a majority of the observations are of a single class, we could create a model that is sensitive to certain conditions to detect the minority class or an ‘anomaly'. Two models were effective in dealing with anomalies, iForest[5] and a nearest neighbor ensemble model[6] (an improvement on the iForest model).
5) **Adjust the class weight/misclassification error**: Many machine learning packages allow for the specification of a 'class weight' parameter. This parameter allows the engineer to increase the weight applied to the misclassification error of one class over the other. In this way, more importance is put on correcting the errors involving the minority class and the model's

[4] “Learning from Imbalanced Classes.” *Silicon Valley Data Science*, 25 Sept. 2017, www.svds.com/learning-imbalanced-classes/#fn4.

[5] Liu, Fei Tony, et al. “Isolation-Based Anomaly Detection.” *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, 2012, pp. 1–39., doi: 10.1145/2133360.2133363.

[6] Bandaragoda, Tharindu R., et al. “Efficient Anomaly Detection by Isolation Using Nearest Neighbour Ensemble.” *2014 IEEE International Conference on Data Mining Workshop*, 2014, doi:10.1109/icdmw.2014.70.

separation barrier is thought to correct itself accordingly. Four popular options are: MinMaxScaler (Scaling with range of 0 to 1 w/out changing distribution), RobustScaler (transforms the feature vector by subtracting the median and then dividing by the interquartile range (75% value — 25% value)), StandardScaler (changing the values so that the distribution SD from the mean=1 with an output close to normal distribution) , and Normalizer (can be standardized or scaling or a mix) are scikit-learn methods to preprocess data for machine learning. Which method you need, if any, depends on your model type and your feature values.

We summarize this section and state that will we explore an adjustment of class weight and evaluate four packages to deal with data imbalance: MinMaxScaler, Normalizer , RobustScale, and StandardScaler

II.MODEL SELECTION

In order to evaluate the imbalance dataset, 179902-target "0" versus 20098-target "1", we explore normalization of four packages and determine which is best to use for model selection.

II. (a) *MinMaxScaler*

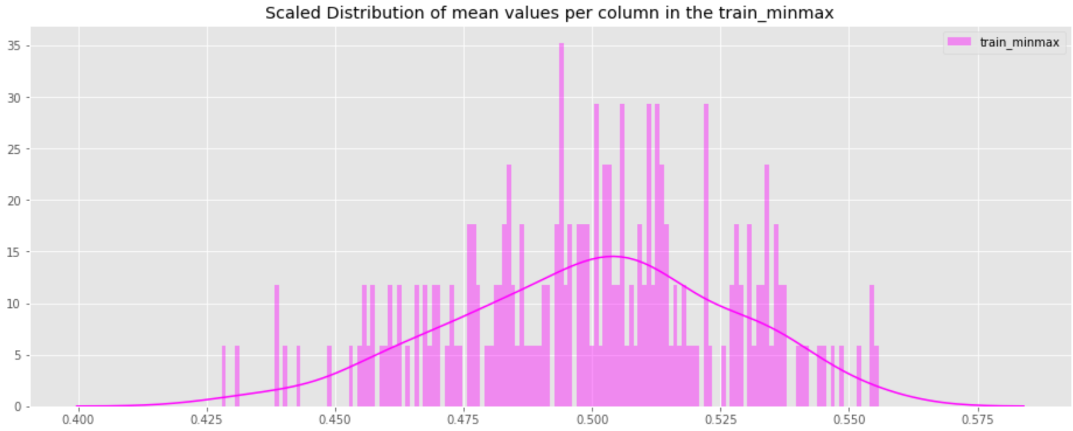

Figure 2 Distribution using MinMaxScaler

II. (b) *Normalizer*

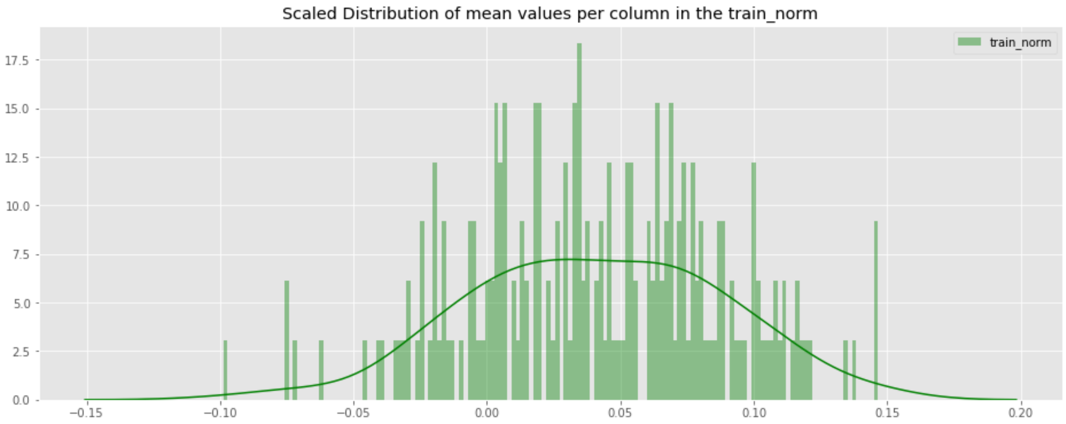

Figure 3 Distribution using Normalizer

III. (c) *RobustScaler*

(d) *StandardScaler*

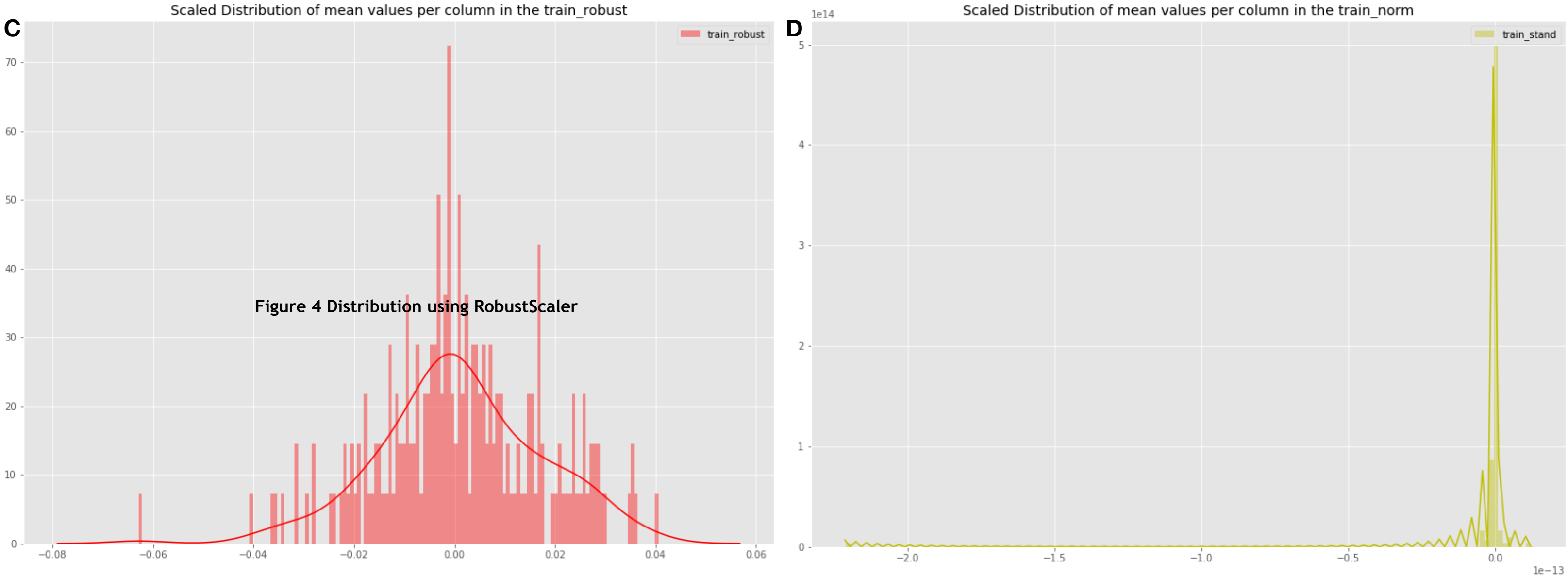

Figure 4 (C)Distribution using *RobustScaler* (D) Distribution using *StandardScaler*

IV. Balanced Data

Upon thorough analysis of the imbalanced output, standardization using the *Normalizer* package was chosen as the best option. Data was saved and analysis was performed. **Figure 5** provides the results of the new balanced target training set.

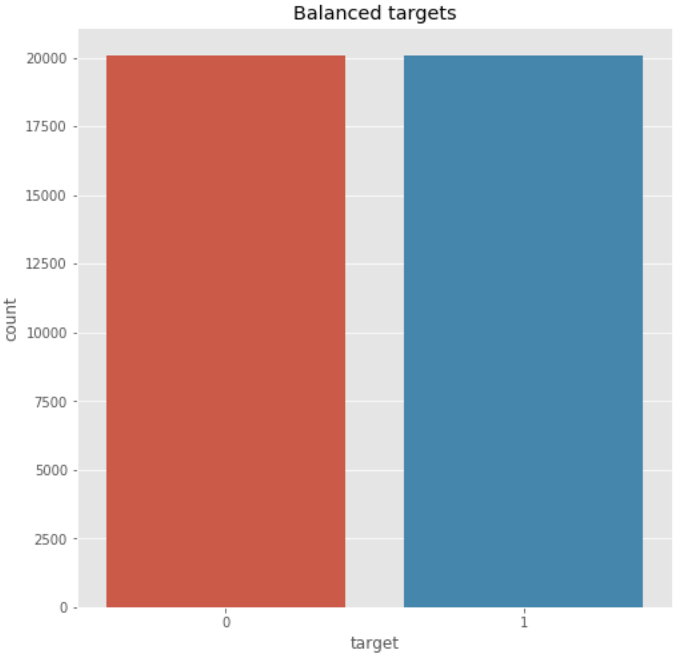

Figure 5 Balanced target data set

ANALYSES

Balanced Data was split into a train and test set (1:5) to prevent the injection of subjective bias into the analysis and model. We first analyze the default models without optimization.

V. WITHOUT OPTIMIZATION

(a) NEURAL NETWORKS

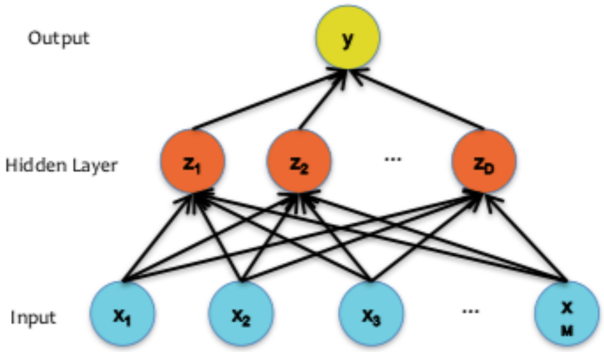

Figure 6 Framework for NN

Figure 6 shows the NN framework utilized in this classification problem of transaction (Input->hidden layers->number of units/ layer->activation function->output). Its measured fit (for weights) is automatic by using “cross-entropy.” **Table A** summaries the results without optimization

Table A. CLASSIFICATION REPORT FOR NNs

	Precision	*Recall*	*f-1 score*	*support*
0	*0.85*	*1.00*	*0.81*	*181000*
1	*0.00*	*0.00*	*0.00*	*1900*
avg/total	*0.65*	*0.73*	*0.67*	*200000*

We summarize the NN model with an accuracy score of 84.6%, AUC of 0.5 and CPU time of 14.42sec

(b) LOGISTIC REGRESSION

Figure 7 shows the LR framework utilized in this classification problem of transaction.[7] The Logistic function has the typical sigmoid shape where the probabilities are between 0 and 1. **Table B** summaries the results without optimization

[7] James, Gareth, et al. An Introduction to Statistical Learning: with Applications in R. Springer, 2017.

Table B. CLASSIFICATION REPORT FOR LR

	Precision	*Recall*	*f-1 score*	*support*
0	*0.91*	*0.89*	*0.90*	*181000*
1	*0.17*	*0.21*	*0.19*	*1900*
avg/total	*0.65*	*0.73*	*0.67*	*200000*

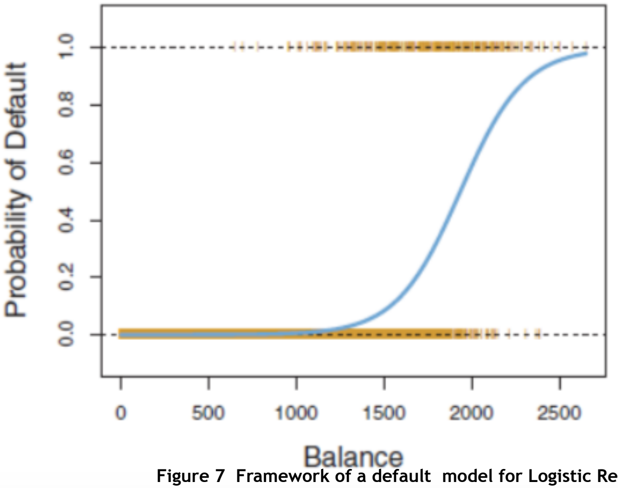

Figure 7 Framework of a default model for Logistic Regression

We summarize the LR model with an accuracy score of 79% , AUC of 0.554 and CPU time of 50.5microseconds

(c) RANDOM FOREST

The RF framework utilizes decision trees and splits for the classification problem of transaction. Since we have 200 predictors, we used 200 splits. Table C summaries the results without optimization

Table C. CLASSIFICATION REPORT FOR RF

	Precision	*Recall*	*f-1 score*	*support*
0	*0.91*	*0.65*	*0.76*	*181000*
1	*0.11*	*0.42*	*0.18*	*1900*
avg/total	*0.84*	*0.63*	*0.71*	*200000*

We summarize the RF model with an accuracy score of 63%, AUC of 0.5365 and CPU time of 48microseconds

VI. WITH OPTIMIZATION

Appropriate functions were modified with defined thresholds for respective models. Classifiers were optimized and their results are presented.

(a) NEURAL NETWORKS

Table D. CLASSIFICATION REPORT FOR NNs W/ OPTIMIZATION

	Precision	*Recall*	*f-1 score*	*support*
0	*0.91*	*1.00*	*0.95*	*181000*
1	*0.00*	*0.00*	*0.00*	*1900*
avg/total	*0.82*	*0.91*	*0.86*	*200000*

We summarize the optimized NN model with an accuracy score of 90.5% and AUC of 0.5 and CPU time of 14.5seconds

(b) LOGISTIC REGRESSION

Table E. CLASSIFICATION REPORT FOR LR W/ OPTMIZATION

	Precision	*Recall*	*f-1 score*	*support*
0	*0.91*	*0.89*	*0.90*	*181000*
1	*0.17*	*0.24*	*0.19*	*1900*
avg/total	*0.84*	*0.82*	*0.83*	*200000*

We summarize the LR model with an accuracy score of 82.5% , AUC of 0.55 and CPU time of 3.4 sec

(c) RANDOM FOREST

Table F. CLASSIFICATION REPORT FOR RF W/ OPTMIZATION

	Precision	*Recall*	*f-1 score*	*support*
0	*0.91*	*1.0*	*0.95*	*181000*
1	*0.00*	*0.00*	*0.00*	*1900*
avg/total	*0.82*	*0.91*	*0.86*	*200000*

We summarize the RF model with an accuracy score of 90.5%, AUC of 0.5 and CPU time of 1 min 52 seconds

CONCLUSIONS

Exploratory analysis of the database revealed improper responses between train (89.95%) and target variables (10.05%). We decided to approach this problem two ways:

1) By standardizing the imbalanced dataset and normalizing its structure
2) Application of careful analysis between accuracy percentage and AUC.

However, it’s important to state that further analysis needs to be done on the imbalanced data. Santander, being a banking enterprise, values prediction and precision over simple inference. The dataset was composed of 200 features which created no correlation and little variance. In addition, anonymity was provided to the variables to keep Santander's inner workings a secret further complicating the exploratory analysis of the data.

That being said, we conclude our report and state that optimized NN and RF models both were equally accurate in prediction at 90.5% in comparison to LR 82.5%. However, NN was much faster in CPU time versus RFs, 14.5sec and 1min 52sec respectively, and therefore serves as the better model tp predict transactions.

REFERENCES

1) Hoskins, J. C., et al. “Fault Diagnosis in Complex Chemical Plants Using Artificial Neural Networks.” *AIChE Journal*, vol. 37, no. 1, 1991, pp. 137–141., doi:10.1002/aic.690370112.

2) Ogwueleka, Francisca Nonyelum, et al. “Neural Network and Classification Approach in Identifying Customer Behavior in the Banking Sector: A Case Study of an International Bank.” *Human Factors and Ergonomics in Manufacturing & Service Industries*, 2012, doi:10.1002/hfm.20398.

3) “Santander Bank.” *Wikipedia*, Wikimedia Foundation, 14 May 2019, en.wikipedia.org/wiki/Santander_Bank.

4) “Learning from Imbalanced Classes.” *Silicon Valley Data Science*, 25 Sept. 2017, www.svds.com/learning-imbalanced-classes/#fn4.

5) Wallace, Byron C., et al. “Class Imbalance, Redux.” 2011 IEEE 11th International Conference on Data Mining, 2011, doi: 10.1109/icdm.2011.33.

6) Liu, Fei Tony, et al. “Isolation-Based Anomaly Detection.” *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, 2012, pp. 1–39., doi:10.1145/2133360.2133363.

7) Bandaragoda, Tharindu R., et al. “Efficient Anomaly Detection by Isolation Using Nearest Neighbour Ensemble.” *2014 IEEE International Conference on Data Mining Workshop*, 2014, doi:10.1109/icdmw.2014.70.

8) James, Gareth, et al. An Introduction to Statistical Learning: with Applications in R. Springer, 2017.