Using Neural Nets and Machine Learning to Detect Credit Card Fraud

STP 494/598 Final Project

Sara Abrams
Department of Psychology
Arizona State University
Tempe, Arizona
sarbrams5@asu.edu

Ashley Megumi Satkowski
School of Arts, Media, and Engineering
Arizona State University
Tempe, Arizona
asatkows@asu.edu

Calvin Dae-sung Norman
Fulton School of Engineering
Arizona State University
Tempe, Arizona
cdnorman@asu.edu

***Abstract*—This report documents the analysis of a credit card fraud dataset and the algorithm's prediction of fraud by using neural nets and machine learning. This document is a research paper for a final project for the class STP 494/598: Machine Learning and Statistical Analysis taught by Dr. Robert McCulloch. The students in this group had no previous experience in R, Machine Learning, or Neural Nets and used the project as a test of how far they have progressed throughout the semester.**

***Index Terms*—smote, neural network, machine learning**

I. Introduction

Millions of credit card transactions are made everyday which makes it critical that credit card companies are able to detect fraudulent charges so that customers are not charged for the items that they did not purchase. Currently, there are datasets, competitions, and kernels online in which people can use Machine Learning and Data Science to perform some sort of analysis. The purpose of this data being available publicly is to give others a chance to use various AI and data mining techniques to come up with an improved solution.

We have decided to take a dataset online from Kaggle (https://www.kaggle.com/mlg-ulb/creditcardfraud/feed) [1] to be used in our final project. The dataset contains credit card transactions made by European cardholders. We are attempting to predict credit card fraud from factors from 28 features obtained from a PCA, time elapsed between each transaction, and the amount charged in the transaction. A full description of abbreviations, acronyms, and variables is listed below.

II. Data Set

This dataset is a record of transactions that occurred in the span of two days with the results of 492 frauds out of 284,807. Notably, the dataset is highly unbalanced with the positive class indicating frauds to be 0.172% for all transactions that occurred. Due to security complications, the data contains numerical outputs from the result of a PCA transformation. Original features and other background information about the data were not provided because it would be a breach in confidentiality.

A. Dataset Variables

Term	Definition
V1..V28 (Variable1...28)	28 variables; Each number is feature that is obtained from a PCA transformation.
Time	Variable in the dataset that indicates the seconds elapsed between each transaction and the first transaction in the dataset.
Amount	Variable in the dataset that indicates the amount of money spent during the transaction.
Class	Variable in the dataset that indicates the response variable that indicates a binary value to show fraud (1) or a genuine transaction (0).

B. Abbreviations and Acronyms

AUPRC (Area Under the Precision-Recall Curve or AUC)	The AUC is from the dimension of precision multiplied by recall and is the area that is under the receiver operation curve. It is a method to commonly used to classify binary data by displaying a curve that is able to better represent the algorithm's performance. Method to measure precision, recall, sensitivity, and specificity. Classifies data into true positive, false negative, false positive, and true negative.
Bootstrapping	Also known as bagging and it is a method in machine learning to try to improve the stability and accuracy of algorithms that are used in classification and regression.

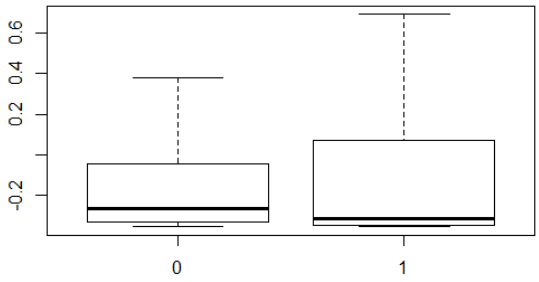
B. (continued)

Term	Definition
PCA (Principal Component Analysis)	A statistical procedure that uses an orthogonal transformation to convert a set of observations of potentially related variables into a set of linearly uncorrelated variables called principal components. [2] The first transformation (principal component) has the largest possible variance and each component after that has the highest possible variance that is under the constraint of the orthogonal of its previous components. The vectors that result create an uncorrelated orthogonal basis set.
SMOTE (Synthetic Minority Over-sampling Techniques)	SMOTE is a function that supersamples rare events in R, also referred to any variable that happens less than 15% of the time. It works by oversampling rare events, which in this case is the detected fraud, and uses the methods of bootstrapping and KNN to artificially create additional observations of that event.
LASSO	Method measuring the prediction error of machine learning models utilizing bootstrap aggregation to the sub-samples of the data samples used for training. OOB is the average prediction error on each training sample using the trees that did not have it in their bootstrap sample.[4]
MSE (Mean Squared Error)	The average of the squared error which is used as the loss function for least squares regression also known as the unbiased estimate of error variance.
NB (Naive Bayes Classifier)	A probabilistic classifier which uses Bayes' theorem to assign class probabilities for each possible outcome.
KNN (k-nearest neighbors)	A non-parametric method that is used for classification and regression in which k is the number of closest training samples that is in the feature space. It is a type of instance based learning where the computations are deferred until classification and the function is approximated locally.
NN (Neural Net)	A computer system model on the human brain and nervous system–a variety of deep learning technologies and methods. Similar to optic nerves in the human brain, the first tier receives raw input informaiton and then each following tier receives the output from the tier before instead of from the raw input. Just like the human brain, in a NN the neurons furthest from the optic nerve (first tier) takes signals closest to it and then produces the output of the system from the last tier.

C. Processing Data

The dataset that was organized into one csv file, with no missing cases. Given the imbalanced ratio of the class, we used Area Under The Curve (AUC) as a measure of accuracy instead of RMSE. Further, the bootstrapping Synthetic Minority Over-sampling (SMOTE) was used to simulate additional positive cases, as the dataset was highly imbalanced. For NN it is optimal to scale and normalize variables around 0. The PCA variables are already scaled, so the only other variable to scale was the transaction amount, which we normalized and scaled around 0. Below shows the box plot of the (normalized) transaction amount and the classes. Outlier cases were removed so the graph could fit on this paper.

Fig. 1. Normalized Transaction Amount of Classes



Further, to handle the low ratio of positive to negative cases, we use SMOTE (which uses KNN and bootstrapping) to simulate more positive cases. Without the additional cases, training any model on a sample of the data would result in poor prediction power.

III. Modeling Data

We tried methods from STP494: Machine Learning along with other resources that found online through stack exchange. We focused on Naive Bayes classification, Logistic Regression, and Single-Layer Neural Net. Further description of each method is provided below.

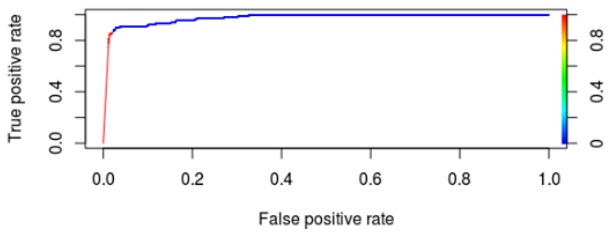
A. Naive Bayes and knn

The initial testing was done with Naive Bayes, with the desire to compare the accuracy to the Neural Net's. The Naive Bayes analysis was run in a loop, with changing Laplace values, in a desire to see how results would change. However, in what was somewhat of a surprise, changing the Laplace values had no effect on the results. The Naive Bayes had an AUC of 0.97555238. With an accuracy of 0.9777. The question testing the Naive Bayes raised was why changing the Laplace value had no effect on the AUC.

After the Naive Bayes testing a KNN model was built, to compare accuracy. It was run with a k value of 5, and had an accuracy of 0.9983. This accuracy was the highest accuracy

measured so far. The figure below shows the result of the Naive Bayes.

Fig. 2. Naive Bayes



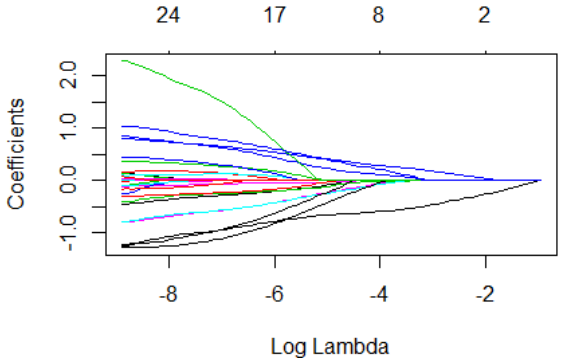
B. Logistic Regression

To give a basic starting point for comparison in accuracy and OOB performance, a logistic regression model was chosen using the R package **glmnet**. Then, the textbflasso feature selection method with cross validation was used to find an optimal model. The output gave the results of the adjusted coefficients. The baseline model is given below.

Class V1:V28 + Amount

The figure below shows the Coefficient selection for the model predictions, using lasso (L1) regularization.

Fig. 3. Coefficients of the Logistic Lasso Regression



C. AOC

One measure of accuracy we are especially interested in is sensitivity. Sensitivity measures the ratio of true positives detected compared to the total number of true positives and false negatives.

$$sensitivity = = \frac{\#\,of\,true\,positives}{\#\,of\,true\,positives + \#\,of\,false\,negatives} \quad (1)$$

Fig. 4. Parameter Selection of Lambda from MSE

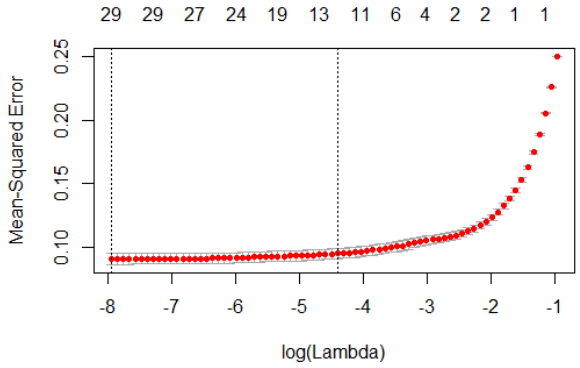
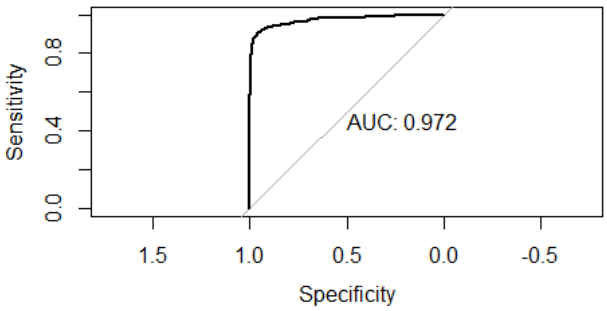


Fig. 5. AUC for the Logistic Regression



The figure below shows the area under the curve for the optimal logistic regression model in glmnet.

D. Neural Net

A neural network (NN) is a machine learning method that uses a series of interconnected processing nodes to predict outcomes using different activation functions. The method is roughly based off of the same learning methods of the human brain.

E. Explanation of why NN was used

As mentioned previously, these will be the rules that set the number of layers and size (neurons/layer) for both the input and output layers of the neural network. Next, we had to figure out how many hidden layers to add. If data is linearly separable (which can be found out when starting to code the NN) then hidden layers are not needed. Although NN are not necessary to find a solution, it can be one of the many methods for prediction and classification. Every NN has three types of layers: *input, output* and *hidden*.

F. Activation Function and Optimization for NN

In a NN, the activation function transforms the net by using scalar-to-scalar functions and introduces nonlinearity into the network. Without this, the hidden layers would not make the NN more powerful than just plain computations without hidden units. Nonlinearity is what makes the multiple layer NN work powerfully. A sigmoid function is used during the computation because the small change in the weights produces a change in the outputs that displays if the difference is positive or negative. Because the output class if binary, we used the logistic function for the activation function that will be applied to the distribution of target values. [8]

The input layer is decided with respect to the number of neurons in this layer and correlates to the shape of the training data. The number of neurons in the input layer is analogous to the number of feature variables (or columns) in the data. Occasionally, an additional bias node is added but for our dataset this was left out. The size of the output layer is set by the configuration of the chosen model. If the NN is a regressor there will be a single node in the output layer and if NN is a classifier then it will either have a single node or one node per class label in the model. For our data, the output contains one node that has a binary value.

There are various opinions for hidden layer configuration in NN but there is a general consensus in the issue of performance differences (which are very few) from adding additional hidden layers. One hidden layer is generally found to be sufficient enough for a large majority of problems. For size of the hidden layer(s)–how many neurons?[7] There are some empirically-derived rules-of-thumb, of these, the most commonly relied on is ”the optimal size of the hidden layer is usually between the size of the input and size of the output layers” [6].

We tested 10 different decay values (0.5, 0.1, .05, 0.025, 0.01, 0.005, 0.0025, 0.001, 0.0001, 0.00001) with a single hidden layer with 2-29 units in the hidden layer. The optimal decay value and layer size was determined by choosing the model with the lowest AUC.

The next figure shows the area under the curve for the optimal NN with 29 units and a decay value of 0.1

The following figure plots the final structure of the neural net.

By using various techniques to trim the network size, it can sometimes improve both the computational and resolution performance. The SMOTE function is used to se the rules of the dat. The function includes calls to boostrapping and KNN. The figure below shows a lift chart, which is slightly AUC chart. The lift chart indicates the prediction gain from the models above baseline, or a simple regression model. In

Fig. 6. AUC for NN

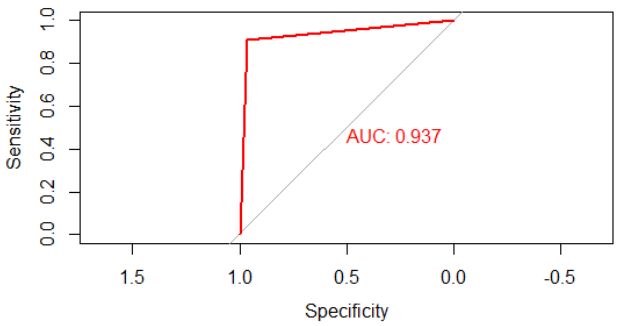
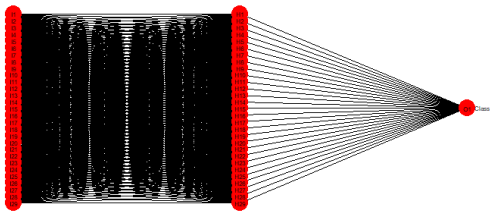
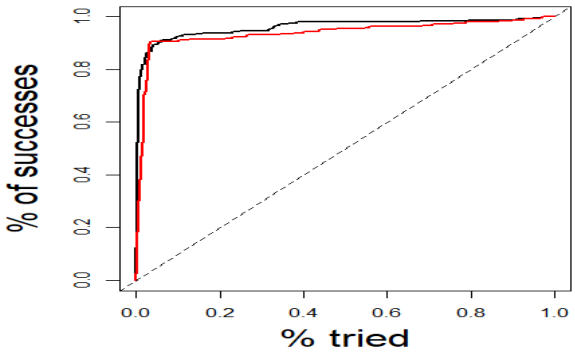


Fig. 7. Visualization of the Neural Net



this model, the neural net is represented by the red line and the logistic regression is represented by the black line.

Fig. 8. Lift of the NN (red) compared to Logistic Regression (black)



As you can see, their prediction improvements are extremely similar. This might be explained by the sigmoid functions that both models use to classify binary outcomes.

IV. Bottlenecks and Challenges

While working on this project there were some issues we encountered. Although the dataset was formatted cleanly, there was rare occurrence data and extreme outliers that needed to be dealt with. Thus, we had to do research and found that SMOTE was the best method to create a new training set. Further, other researchers suggesting avoiding using a confusion matrix alone

to measure model accuracy. Instead, we opted to use the AUC variable to find the optimal solution.

V. Conclusion and Future Work

We were curious to see how the methods would compare and surprised to see that the logistic regression had better performance than the neural net. This may be due to the fact that both the logistic regression model and the neural net use sigmoid functions as a method

In there future, additional research can be conducted by potentially using additional layer in a different NN model instead of the ——that we used should be considered. To compare the results and accuracy for more analysis, classification techniques such as random forests and classification trees can also be used in future work.

Acknowledgment

The dataset used was taken from Kaggle and was collected during a research collaboration of Worldline and the Machine Learning Group of Universite Libre de Bruxelles on big data mining and fraud detection. Special thanks to professor Dr. Robert McCulloch and the Graduate Math Tutoring Center in Wexler Hall at Arizona State University.

References

[1] Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson and Gianluca Bontempi. Calibrating Probability with Undersampling for Unbalanced Classification. In Symposium on Computational Intelligence and Data Mining (CIDM), IEEE, 2015.

[2] Abdi. H., & Williams, L.J. (2010). "Principal component analysis". Wiley Interdisciplinary Reviews: Computational Statistics. 2 (4): 433?459. doi:10.1002/wics.101

[3] Bowyer W. Kevin, Chawla V. Nitesh, Hall O. Lawrence, Kegelmeyer P. W. "SMOTE: Synthetic Minority Over-sampling Technique". pp. 321-357. 2012. Journal of Artificial Intelligence Research.

[4] Gareth James; Daniela Witten; Trevor Hastie; Robert Tibshirani (2013). An Introduction to Statistical Learning. Springer. pp. 316?321.

[5] Davis, J., Goadrich, M. "The Relationship Between Precision-Recall and ROC Curves". Department of Computer Sciences and Department of Biostatistics and Medical Informatics, University of Wisconsin-Madison, 1210 West Dayton Street, Madison, WI, 53706 USA

[6] Jeff Heaton, author of Introduction to Neural Networks in Java offers a few more

[7] D. Hunter, H. Yu, M. S. Pukish III, J. Kolbusz, and B. M. Wilamowski, ?Selection of proper neural network sizes and architectures: a comparative study,? IEEE Transactions on Industrial Informatics, vol. 8, no. 2, pp. 228?240, 2012

[8] Jordan, M.I. (1995), "Why the logistic function? A tutorial discussion on probabilities and neural networks", MIT Computational Cognitive Science Report 9503, http://www.cs.berkeley.edu/ jordan/papers/uai.ps.Z.