Term Project Report
Perceptrons

Problem Statement

Credit card fraud is the unauthorized use of a payment card or similar payment tool that is often part or result of an identity theft scheme[1]. In 2018, credit card fraud from both new and existing accounts comprised 50.5% of the 323,000 reports of top five types of identity theft[2]. Understanding the credit card fraud data and examining important factors may help in reducing such instances and protecting consumer data. We wanted to classify if a transaction is a fraud or not based on attributes such as transaction type and amount, and initial and new balance.

Collection and Description of Datasets

We found two credit card transaction datasets from Kaggle:

1) https://www.kaggle.com/mlg-ulb/creditcardfraud#creditcard.csv
2) https://www.kaggle.com/ntnu-testimon/paysim1

Our first dataset is a record of credit card transactions over a two-day period in September 2013 by European cardholders. The dataset is highly unbalanced, with 492 out of approximately 284,000 transactions being fraudulent. Feature “Time” contains the seconds elapsed between each transaction and the first transaction in the dataset. Feature “Amount” is the transaction amount. Due to confidentiality issues, descriptions on features V1 to V28 are not available. The response variable is “Class,” which takes value 1 for fraudulent transactions and 0 otherwise.

Due to a lack of such datasets that are available to the public, our second dataset is a synthetic dataset generated using the PaySim simulator. The synthetic dataset has 11 attributes and over 1 million observations. It is based on a sample of real transactions extracted from one month of financial logs from a mobile money service implemented in an African country. The attributes are a unit of time, transaction type, transaction amount, and name and initial and new balance for both sender and recipient. Two binary variables are also included, each indicating whether the transaction is fraudulent or flagged (which is characterized by illegal attempts). The response variable is “isFraud.”

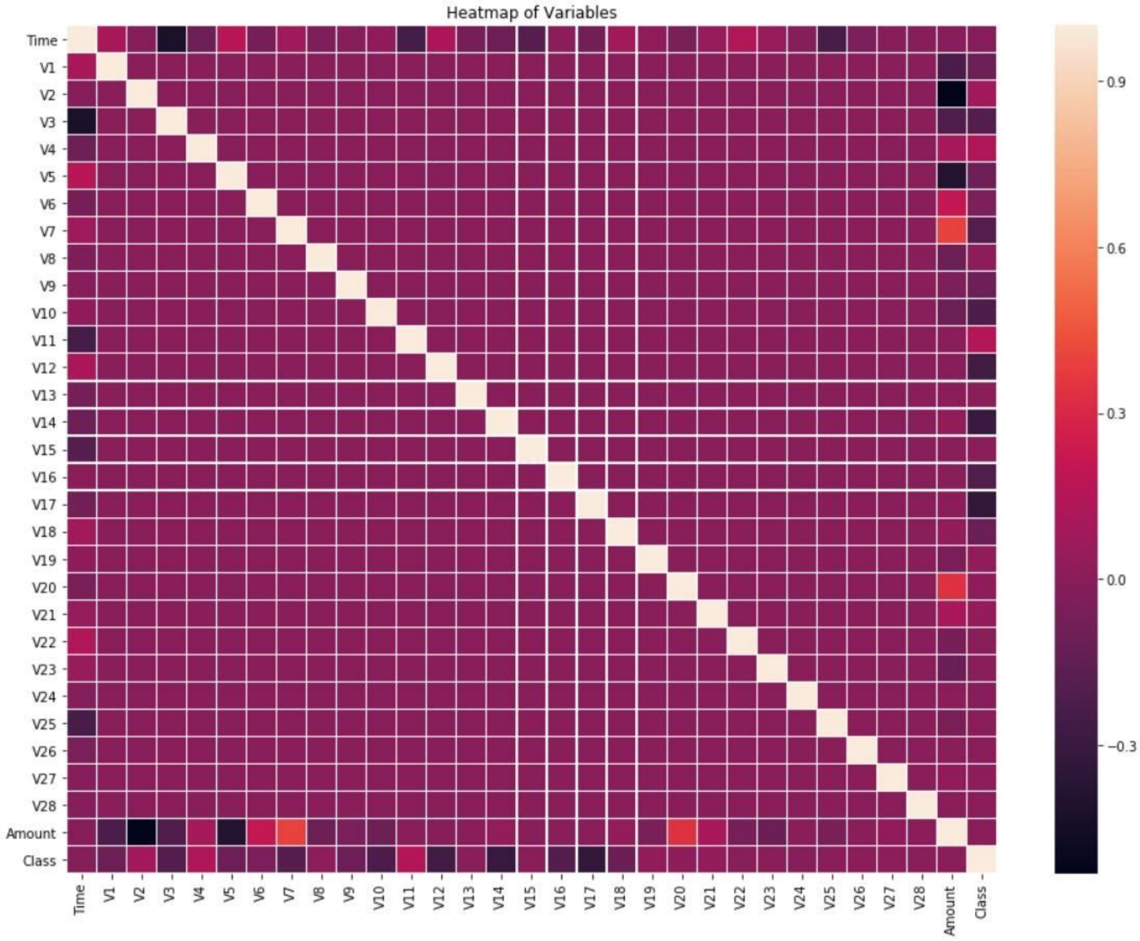
Both datasets have the same binary response variable. Because the datasets each target a different population, it would be interesting to see the similarities and differences in the results. The patterns from the second dataset may give us insight to the sensitive features in the first dataset.

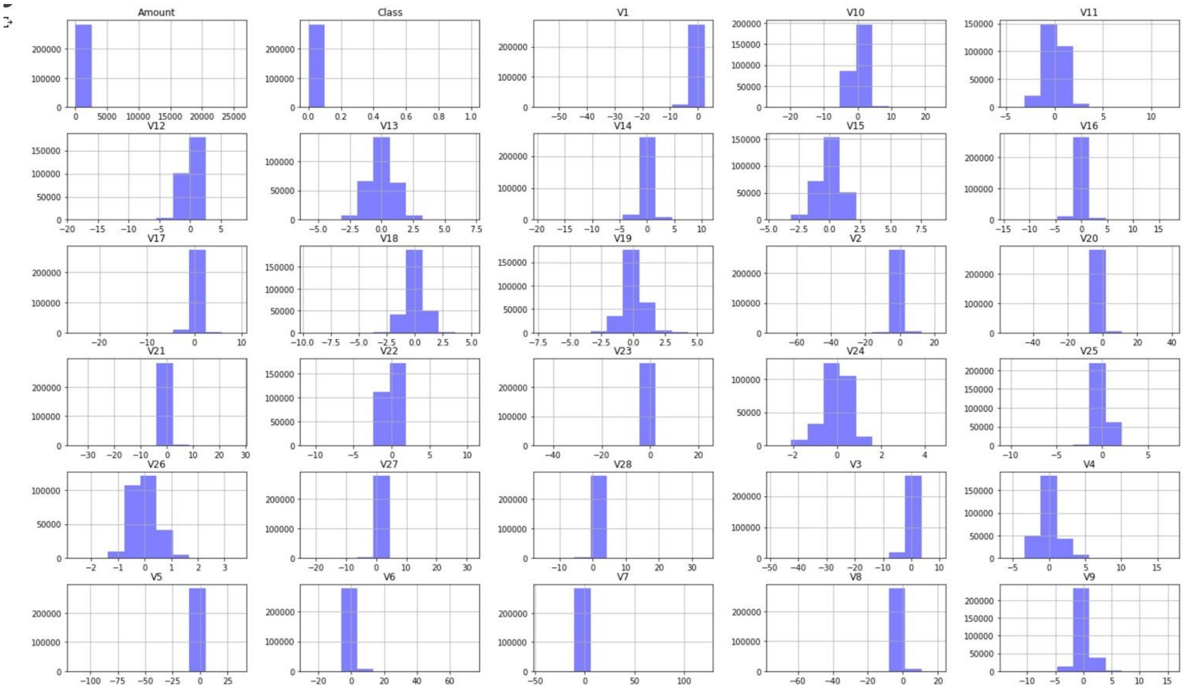
Data Preprocessing

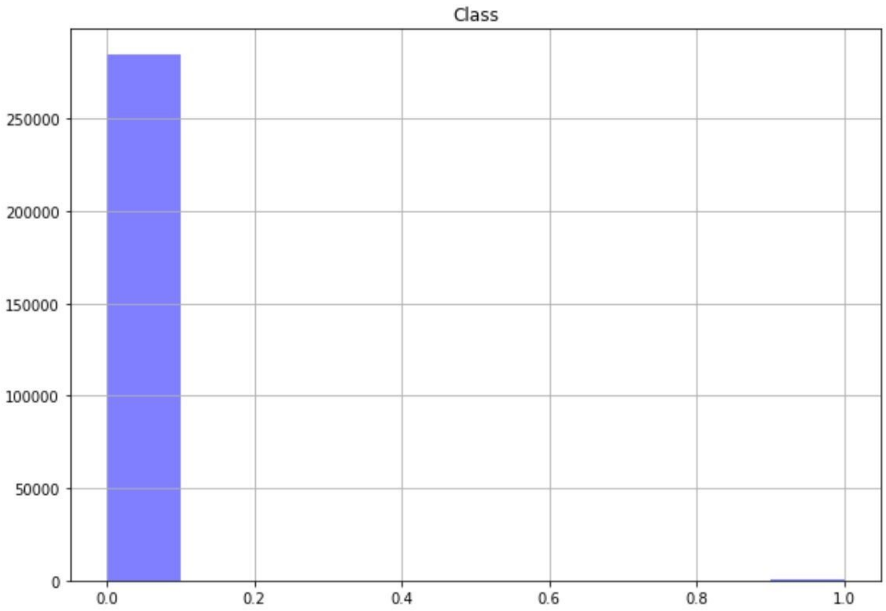
For the first dataset, we checked to see if the dataset had any missing values by using dropNan, but we found that the data had no missing values for both datasets. We also used the method “getDummies” in order to make some categorical variables numeric in nature. There were also some string objects such as “newbalanceDest” that had to be converted using getDummies. The time variable may be converted so that they are not based on the time of the first transaction. Because all the variables are already numerical, not a lot of data preprocessing is needed. The only column that had not been scaled for the Credit Card data was the amount column, which we were able to scale for some of the models. The correlation matrix also indicates that there is not a strong correlation between and on the input variables, except perhaps with amount. This suggests that most of the input variables are independent of each other. The data was skewed towards transactions that weren’t fraud, and while we did not use SMOTE, this could be a possible way to improve our analysis in the future.

Visual Examination

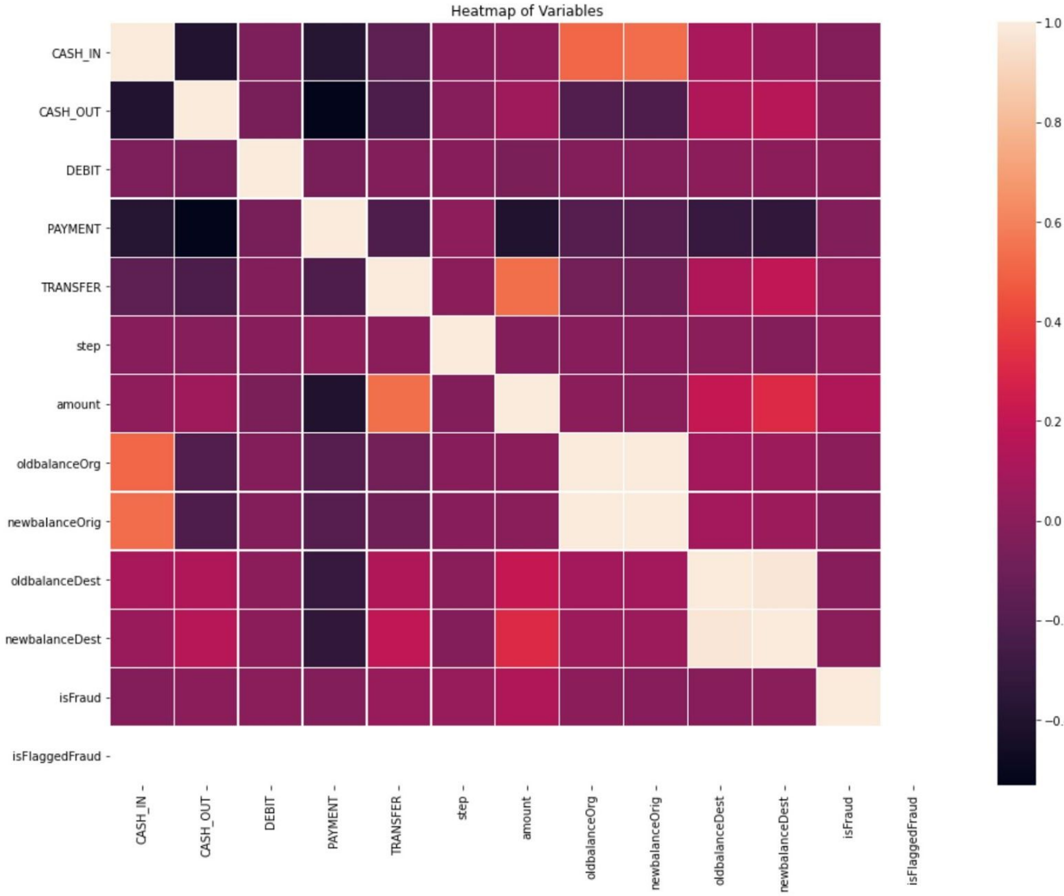
We first created a correlation matrix to get an idea of how well each attribute correlates with each other and also the label of the observation. While most attributes seem to have very little correlation with each other, the attributes appear to have slight correlations with the Class (fraud or not fraud). We printed out histograms of each input variable to show the distribution, which center around 0 and seem to appear fairly normal. This is good for when we use the gaussian NaiveBayes model, as this model assumes variables are normally distributed. When creating a bar graph for the distribution of fraud and non-fraud transactions, we noticed a clear imbalance. There is a much greater number of non-fraud transactions than fraud ones, which could affect the accuracy of model results. This is displayed in the third figure below.

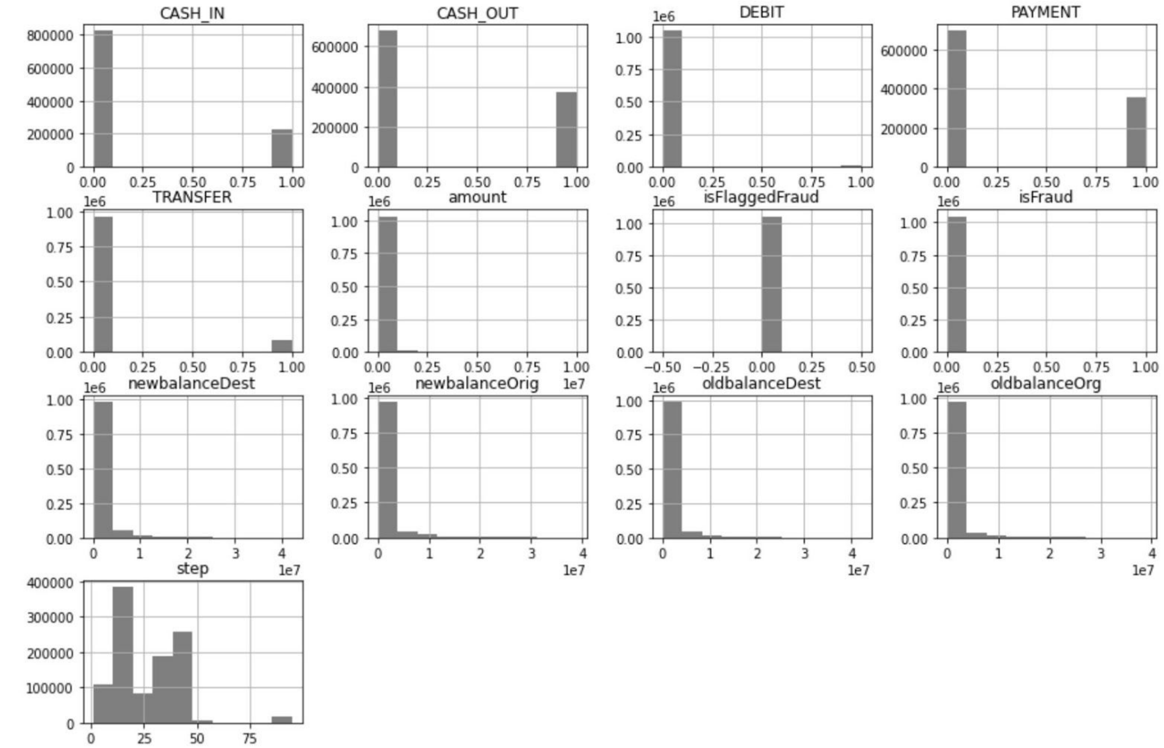

Heatmap of Variables
Time
V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
V15
V16
V17
V18
V19
V20
V21
V22
V23
V24
V25
V26
V27
V28
Amount
Class
0.9
0.6
0.3
0.0
−0.3

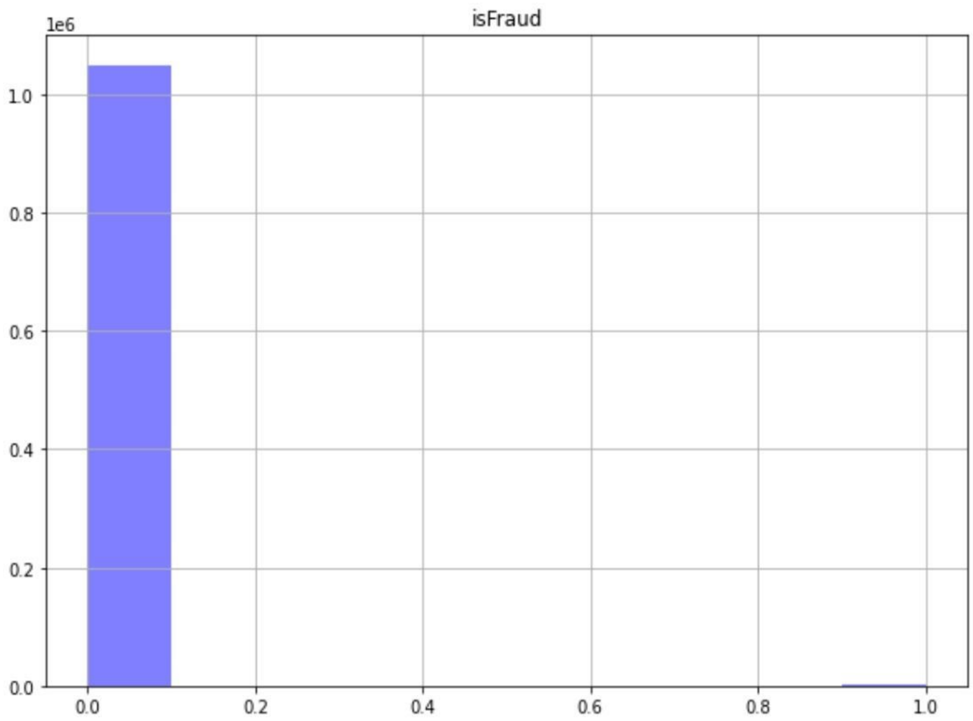

Amount
0
100000
200000
0 5000 10000 15000 20000 25000
Class
0.0 0.2 0.4 0.6 0.8 1.0
V1
−50 −40 −30 −20 −10 0
V10
50000
150000
−20 −10 0 10 20
V11
−5 0 5 10
V12
−20 −15 −10 −5 0 5
V13
−5.0 −2.5 0.0 2.5 5.0 7.5
V14
−20 −15 −10 −5 0 5 10
V15
−5.0 −2.5 0.0 2.5 5.0 7.5
V16
−15 −10 −5 0 5 10 15
V17
−20 −10 0 10
V18
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0
V19
−7.5 −5.0 −2.5 0.0 2.5 5.0
V2
−60 −40 −20 0 20
V20
−40 −20 0 20 40
V21
−30 −20 −10 0 10 20 30
V22
−10 −5 0 5 10
V23
−40 −20 0 20
V24
−2 0 2 4
V25
−10 −5 0 5
V26
−2 −1 0 1 2 3
V27
−20 −10 0 10 20 30
V28
−10 0 10 20 30
V3
−50 −40 −30 −20 −10 0 10
V4
−5 0 5 10 15
V5
−100 −75 −50 −25 0 25
V6
−20 0 20 40 60
V7
−50 0 50 100
V8
−60 −40 −20 0 20
V9
−10 −5 0 5 10 15



For the second dataset, we would first set the name features aside, but may further analyze later for any duplicated values. The transaction type attribute may be converted into integers or be converted into dummy variables with one hot encoding. Any missing values would be handled by imputing some values or removing certain observations. We may use a random sample of each dataset if they are determined to be too big.


Heatmap of Variables
CASH_IN
CASH_OUT
DEBIT
PAYMENT
TRANSFER
step
amount
oldbalanceOrg
newbalanceOrig
oldbalanceDest
newbalanceDest
isFraud
isFlaggedFraud
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4


CASH_IN
800000
600000
400000
200000
0
0.00 0.25 0.50 0.75 1.00
CASH_OUT
600000
400000
200000
0
0.00 0.25 0.50 0.75 1.00
DEBIT
1e6
1.00
0.75
0.50
0.25
0.00
0.00 0.25 0.50 0.75 1.00
PAYMENT
600000
400000
200000
0
0.00 0.25 0.50 0.75 1.00
TRANSFER
1e6
1.00
0.75
0.50
0.25
0.00
0.00 0.25 0.50 0.75 1.00
amount
1e6
1.00
0.75
0.50
0.25
0.00
0.00 0.25 0.50 0.75 1.00
1e7
isFlaggedFraud
1e6
1.00
0.75
0.50
0.25
0.00
−0.50 −0.25 0.00 0.25 0.50
isFraud
1e6
1.00
0.75
0.50
0.25
0.00
0.00 0.25 0.50 0.75 1.00
newbalanceDest
1e6
1.00
0.75
0.50
0.25
0.00
0 1 2 3 4
1e7
newbalanceOrig
1e6
1.00
0.75
0.50
0.25
0.00
0 1 2 3 4
1e7
oldbalanceDest
1e6
1.00
0.75
0.50
0.25
0.00
0 1 2 3 4
1e7
oldbalanceOrg
1e6
1.00
0.75
0.50
0.25
0.00
0 1 2 3 4
1e7
step
400000
300000
200000
100000
0
0 25 50 75



Modeling Techniques

We used classification to predict whether the transaction is fraud or is not fraud. Because we used classification, we chose a null model technique, a Gaussian NaiveBayes technique for simple complexity, logistic regression for standard complexity, a random forest technique for intermediate complexity, and a NeuralNet_Classif_XL for the complex model. We hope to create a model that can predict if a transaction isFraudulent (1) or isNotFraudulent (0) with high accuracy.

Explanation of Techniques Chosen

We had to use methods to predict classification. We used a Gaussian NaiveBayes model because many of the attribute variables were numeric and generally followed a normal distribution. We used logistic regression for classification because it is useful for classification when there's a decision threshold. It also accounts for outliers well.

For the more complex models:

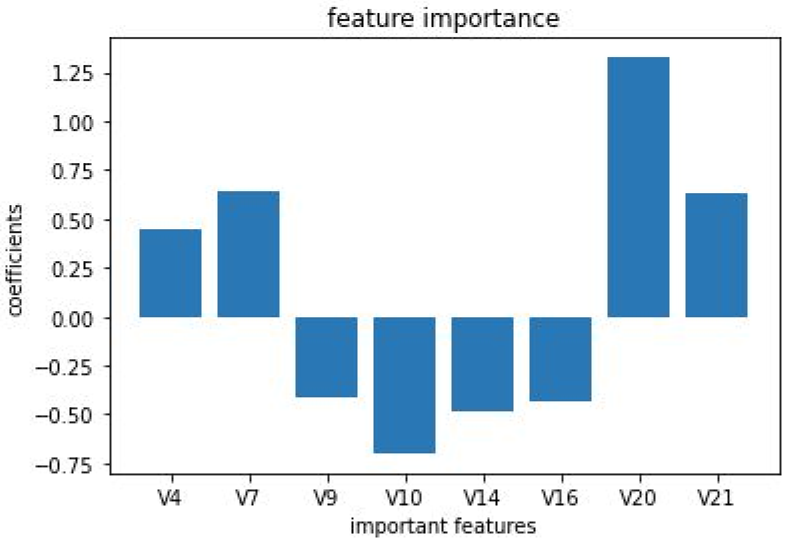
1. RandomForestClassifier
    a. Random forest classifier creates a set of decision trees from randomly selected subset of training set
        i. Vote aggregation
    b. Extremely high accuracy
        i. A lot of weak estimators can form a strong estimator
    c. It’s an ensemble method of decision trees generated on a randomly split database
    d. It has the power to handle a large data set with high dimensionality
    e. Default n_estimators = 10

2. NeuralNetXL
    a. Allows for weights to be determined by the loss gradient
    b. Large amount of data to be processed and analyzed
    c. Multi-dimensional output available
    d. High accuracy
    e. Ease of parameter tuning

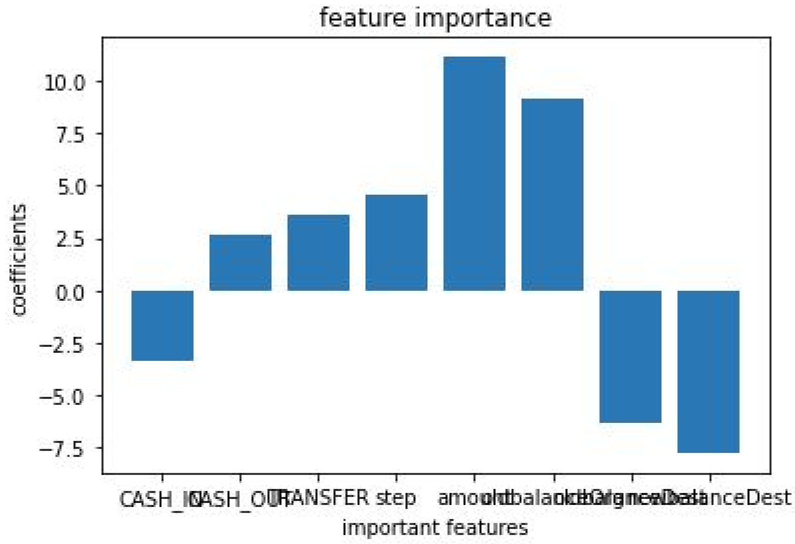
Feature Selection

For logistic regression, SelectFromModel method from sklearn was used. The feature_importances_ method was not available for logistic regression, so coefficients were graphed against the most important features (top 8). Because other models had better F1-score, the important features from the logistic regression were not explicitly reported.

Logistic regression credit card (nonscaled)

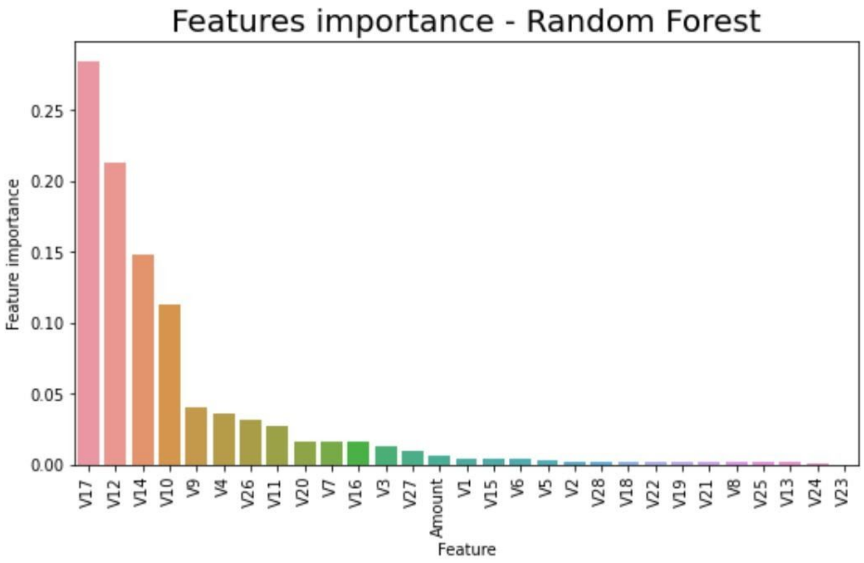


Logistic regression synthetic (scaled)



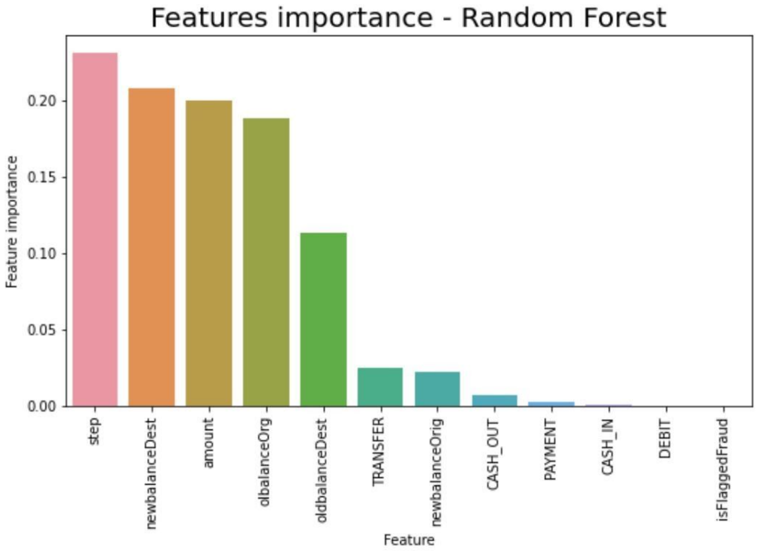
For the random forest model, we used the sklearn method gridSearcgCV, which lets you combine an estimator with a grid search preamble to tune hyper-parameters. This technique helps to find the optimal parameters. We then used the parameters that returned the best score as inputs for our model. We then used the feature_importances_ method from sklearn on the model to determine which features were the most important to the model. We plotted these values on a bar graph so that a user could interpret which features are the most important.

Credit Card



Credit Card Data	
Rank	Feature
1	V17
2	V12
3	V14
4	V10
5	V9

Synthetic



Synthetic Data

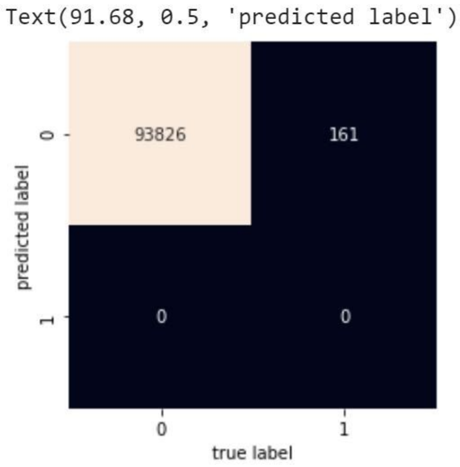
Rank	Feature
1	step
2	newbalanceDest
3	amount
4	oldbalanceOrg
5	oldBalanceDest

Reporting of Results

Null Model:

Credit Card

isFraud	492 / 284,807	.00172
isNotFraud	284,315 / 284,807	.9983



```
Text(91.68, 0.5, 'predicted label')
```

TN = 284,315	FP = 0
FN = 492	TP = 0

Precision = not defined
Recall = 0

Synthetic

isFraud	1,142 / 1,048,575	.00109
isNotFraud	1,047,433 / 1,048,575	.99891

TN = 1,047,433	FP = 0
FN = 1,142	TP = 0

Precision = not defined
Recall = 0

Logistic Regression Model

Credit Card

Nonscaled

TN = 93,807	FP = 56
FN = 19	TP = 105

Scaled

TN = 93,813	FP = 52
FN = 13	TP = 109

Synthetic

Nonscaled

TN = 345,542	FP = 242
FN = 109	TP = 137

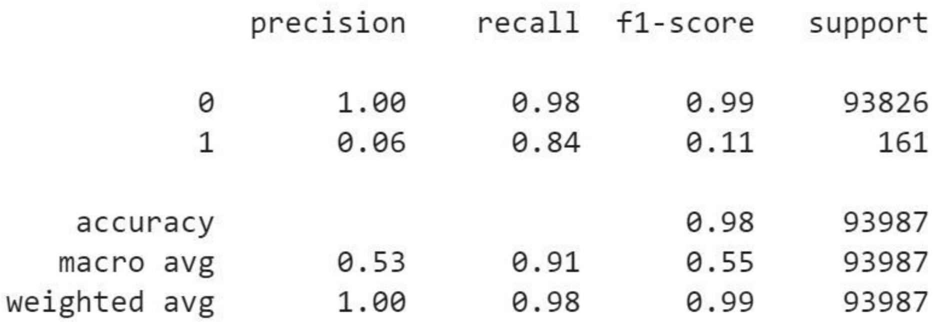
Scaled

TN = 345,650	FP = 345
FN = 0	TP = 35

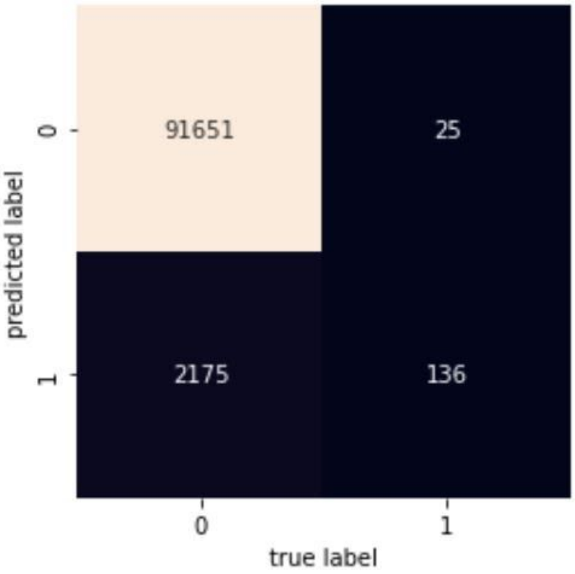
	Credit Card	Synthetic
Accuracy	0.999 (nonscaled) 0.893 (scaled)	0.999 (nonscaled) 0.999 (scaled)
Precision	0.847 (nonscaled) 0.893 (scaled)	0.557 (nonscaled) 1.0 (scaled)
Recall	0.652 (nonscaled) 0.677 (scaled)	0.361 (nonscaled) 0.092 (scaled)
F1	0.737 (nonscaled) 0.770 (scaled)	0.438 (nonscaled) 0.169 (scaled)

NaiveBayes Model:

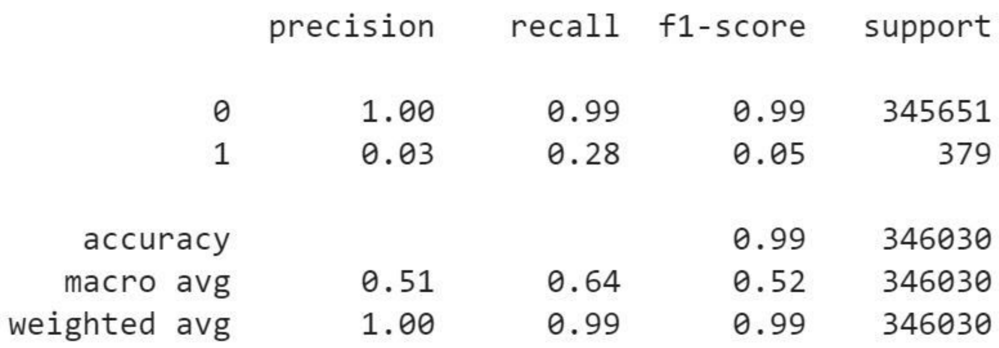
Credit Card



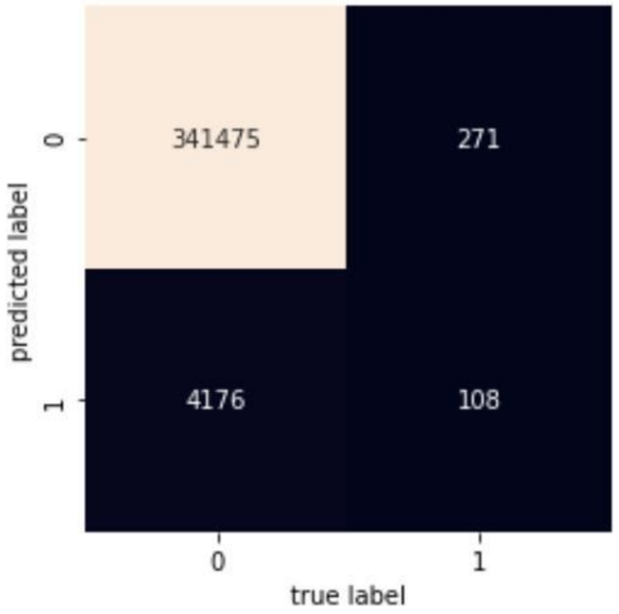
```
              precision    recall  f1-score   support

           0       1.00      0.98      0.99     93826
           1       0.06      0.84      0.11       161

    accuracy                           0.98     93987
   macro avg       0.53      0.91      0.55     93987
weighted avg       1.00      0.98      0.99     93987
```



Synthetic

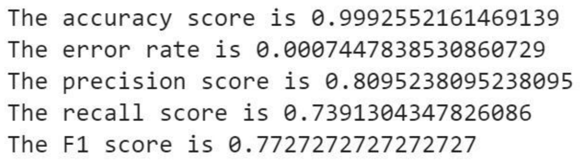


```
              precision    recall  f1-score   support

           0       1.00      0.99      0.99    345651
           1       0.03      0.28      0.05       379

    accuracy                           0.99    346030
   macro avg       0.51      0.64      0.52    346030
weighted avg       1.00      0.99      0.99    346030
```

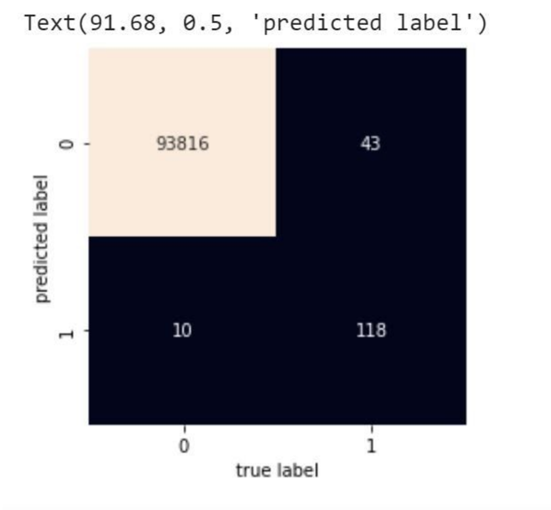


RandomForest Model

Credit Card

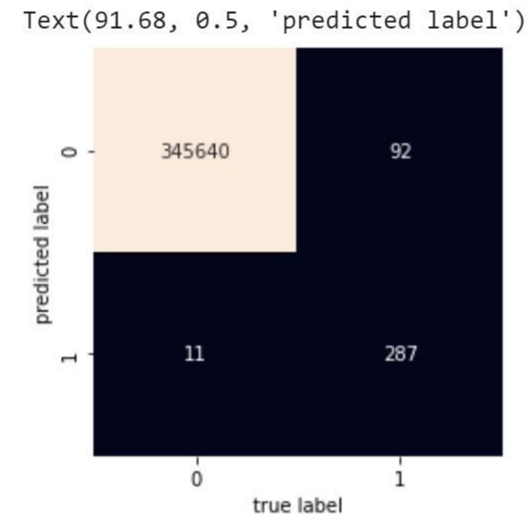


```
The accuracy score is 0.9992552161469139
The error rate is 0.0007447838530860729
The precision score is 0.8095238095238095
The recall score is 0.7391304347826086
The F1 score is 0.7727272727272727
```



```
Text(91.68, 0.5, 'predicted label')
```

Synthetic



```
Text(91.68, 0.5, 'predicted label')
```

```
The error rate is 0.00029766205242320876
The precision score is 0.9630872483221476
The recall score is 0.7572559366754618
The F1 score is 0.8478581979320532
```

NeuralNetXL Model

All of these numbers are based on the testing set and are evaluated based on the best model. Side Note: The first iteration of the for loop selects all columns whose Accuracy is higher than the null model.

Credit Card (Testing set): n* = 13

TN = 31179	FP = 6
FN =12	TP = 48

Metrics:

- Accuracy: 0.999329686164856
- Recall: 0.8
- Precision: 0.888889
- F1 Score: 0.8421052631578948

Synthetic Card (Testing set): n* = 7

TN = 114333	FP = 34
FN = 45	TP = 94

Metrics:

- Accuracy: .9993555545806885
- Recall: 0.734375
- Precision:0.6762589928057554
- F1 Score: 0.7041198501872659

Feature selection was based on the accuracy of each model when a column was added. The forward selection method takes all columns that have accuracy greater than the null model due to the fact that to how long the runtime took. I choose this model to avoid overfitting since Neural Networks are sensitive to extreme values and overfitting. The metrics of the model are not too different when compared to the null model. The results were good overall, and the recall rates were high for Credit cards and relatively high for the Synthetic datasets. Random forests did perform better overall when compared to the Neural Network model for the synthetic dataset so I think that it is the better model for that dataset which is mutually beneficial, since it requires less computing power.

Recommendations of Study

Credit Card

It is recommended to focus on collecting data for attributes: V17, V12, V14, V10, V9, V4, V26, V11 and V20, respectively, because these features were determined to be more important than others. NeuralNetXL was determined to be the best model with an F1-score of 0.842, but compared to the next best model Random Forest, it is only greater by .025. Therefore, it needs to be decided whether such a small increase in F1-score is worth extra computation power and memory.

Synthetic

We recommend focusing on collecting data for attributes: step, newBalanceDest, amount, oldBalanceOrg and oldBalanceDest, respectively, because these features were determined to be more important than others. Random Forest seemed to be the best model with an F1-score of 0.848.

Overall

Recall score should be the main quality of fit since in fraud detection, there is a high cost associated with false negatives. Using recall scores, the customer service department should adapt to quickly handle customers who may have locked cards due to false fraud detection. We recommend using data with less imbalance or fixing the imbalance, and possibly datasets that are more up-to-dated. Many of our codes took a long time to run, so we recommend solutions such as running random Forest gridSearchCV with more combinations if given more RAM. If the company has resources and access to more similar data, comparing features chosen by Synthetic and Credit Card data and examining if they match (for anonymous attributes) would also be helpful in understanding overall credit card transaction data and the patterns of fraudulent transactions. Finding a better way to optimize against false negatives for NeuralNetXL is also recommended.

Credit Card Overall

Model	F1-score
NullModel	Accuracy = 0.9983
NaiveBayes - Gaussian	0.110
Logistic Regression	0.770 (scaled)
Random Forest	0.817
NeuralNetXL	0.8421

Synthetic Overall

Model	F1-score
NullModel	Accuracy = 0.99891
NaiveBayes - Gaussian	0.046320
Logistic Regression	0.438 (nonscaled)
Random Forest	0.848
NeuralNetXL	0.730

Sources

[1] https://www.fbi.gov/scams-and-safety/common-scams-and-crimes/credit-card-fraud

[2] https://www.iii.org/fact-statistic/facts-statistics-identity-theft-and-cybercrime