MachineLearningNanodegree Shanmukha Mudigonda CapstoneProposal Date:August 04,2019

Github Link:

<https://github.com/mudigosa/Fraud-Detection>

DomainBackground

AsreportedbyBusinessInsider[1],lastyear,CreditCards.comfoundthatcreditcardfraud wasontherise.Bothnumberoffraudsandtypesofcreditcardscamsaremoreandmore. Thesenumbersseemunfortunatelydestinedtogrowevenmoreinthefuture.Forthisreason, itisimportanttofindawaytoautomaticallyrecogniseanomalies.Alotofresearchhasbeen doneinordertofindasolutiontothisproblem.References[6]and[7]proposesolutionsusing ArtificialNeuralNetworks.

**ProblemStatement**

Theaimofthefrauddetectionsystemistodetectfraudaccuratelyandbeforefraudiscom- mitted.Thegoalistodetectleastandaccuratefalsefrauddetection.Themostcommonly techniquesusedfrauddetectionmethodsareRandom Forest and XGB Classifier machine learningalgorithms.

Eachtransactionhasasetofuniquefeatures,suchasthevalueofthetransaction,thetimeatwhichitoccured,issuerandrecipient,anidandothersensitivedata.Theproblemwillbestructuredasaclassificationproblem,whereeachtransactioncouldbeclassifiedas”Normal”or”Fraud”.

**DatasetsandInputs**

Inordertoreproducethiskindofproblem,IfoundausefuldatasetavailableonKaggle[4].The datasetscontainstransactionsmadebycreditcardsinSeptember2013byeuropeancardholders. Thisdatasetpresentstransactionsthatoccurredintwodays,wherewehave492fraudsoutof 284,807transactions.Thedatasetishighlyunbalancedandthepositiveclass(frauds)account for0.172%ofalltransactions.

ItcontainsonlynumericalinputvariableswhicharetheresultofaPCAtransformation.Due toconfidentialityissues,theauthorscannotprovidetheoriginalfeaturesandmorebackground informationaboutthedata.FeaturesV1,V2,...V28aretheprincipalcomponentsobtained withPCA,theonlyfeatureswhichhavenotbeentransformedwithPCAare”Time”and ”Amount”.Feature”Time”containsthesecondselapsedbetweeneachtransactionandthe firsttransactioninthedataset.Thefeature”Amount”isthetransactionAmount,thisfeature canbeusedforexample-dependantcost-senstivelearning.Feature”Class”istheresponse variableandittakesvalue1incaseoffraudand0otherwise.

Data Exploration

The dataset to be used in solving this problem is an anonymized set of credit card   
transactions labelled as fraudulent or genuine. Due to confidentiality issues, the original   
features and more background information about the data were not provided.

The dataset presents transactions that occurred in two days, where there are 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

The dataset contains 31 numerical features. The first 28 features are labelled V1, V2….V28   
and these are principal components obtained with Principal Components Analysis of the   
raw/original data.

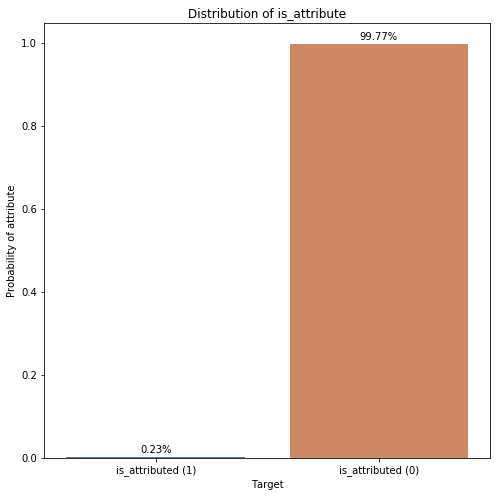
The only features which have not been transformed with PCA are:

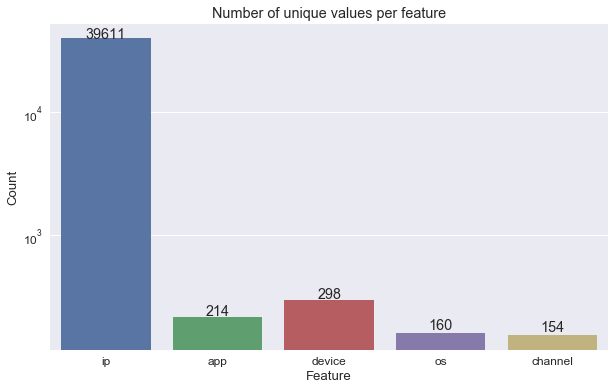
I. Time - contains the seconds elapsed between each transaction and the first

transaction in the dataset

II. Amount - transaction amount

The last feature to be discussed is ‘Class’ which is the response variable and it takes value 1 in case of fraud and 0 otherwise.





**Data Preprocessing**

The first step taken in pre-processing the dataset was **Normalization.** The Normalization   
procedure was applied only on the **Amount** Feature since it wasn’t on the same scale as the other features.

.

**Implementation:**

The implementation stage involved creating a training and predicting pipeline .This stage involved testing four algorithms to see which best suits the problem. The following   
algorithms were used;

I. XGB Classifier

II. Random Forest

The following metrics were used to measure performance as the data set scaled;

I. Time taken to train the model.

II. Time taken to make predictions on the train and cross validation set.

NoticethatinthedatasetthatIhavechosen,”Time”isthesecondselapsedbetweeneach transactionandthefirsttransactioninthedataset.Itisunknownduringwhattimeofthe daythetransactionsactuallybegan.Then,thecolumncanatmostinformushowclosethe transactionsweremadeinbetween2fraudulentones.SinceIwanttopredictfraudsregardless oftransactiontime,thisfeaturewillbedroppedearlier.Noticethattimeandamounthave verydifferentmagnitudesinthedataset,whichwilllikelyresultinthelargemagnitudevalue ”washingout”thesmallmagnitudevalue.Itwouldbebettertoscalethedatatosimilarmag- nitudes.MostofthedataresultfromtheproductofaPCAanalysis.Iwilldothesametothe ”amount”column(rememberthattimewillbedropped!).

Ineedtofaceanotherproblemalso.Fraudulenttransactionsaresignificantlylowerthan normalhealthytransactionsi.e.accountingittoaround1or2%ofthetotalnumberofobser- vations.WiththeautoencoderIwillgeneratedataforaugmentingthedataset

**Metrics**

In a binary classification problem such as this, a model classifies examples as either positive (fraudulent) or negative (genuine). The decision made by the model, either positive or   
negative can be represented in a structure known as **confusion matrix**. This confusion   
matrix has four elements that define it, contextually they are:

 True Positive (TP) – An example where a transaction is fraudulent and is classified   
correctly as fraudulent.

 False Positive (FP) – An example where a transaction is genuine and is classified as   
incorrectly as fraudulent.

 True Negative (TN) – An example where a transaction is fraudulent but is classified   
incorrectly as genuine.

 False Negative (FN) – An example that is genuine and is classified correctly as fraudulent.

In most binary classification problems, accuracy is used as a metric for evaluating models   
and is given as follows:

Accuracy = 𝑇𝑟𝑢𝑒 𝑃𝑜𝑠𝑖𝑡𝑖𝑣𝑒 + 𝑇𝑟𝑢𝑒 𝑁𝑒𝑔𝑎𝑡𝑖𝑣𝑒   
𝑇𝑜𝑡𝑎𝑙 𝑂𝑏𝑠𝑒𝑟𝑣𝑎𝑡𝑖𝑜𝑛𝑠

Accuracy is a measure of how well a model accurately predicts true examples as positive   
and false examples as negative.

Now in situations like this where there is a class imbalance ratio (negative examples   
outnumber positive examples by a very wide margin), the performance of a model is not   
reflected by accuracy because there are not enough training points for the positive class   
(fraudulent transactions).

Given the class imbalance ratio in the dataset, there are other evaluation metrics that take such class imbalance into consideration:

I. Precision   
II. Recall

Recall measures the fraction of fraudulent transactions that are correctly predicted while Precision measures that fraction of cases predicted to be fraudulent that are truly   
fraudulent.

Precision and Recall are defined as follows:

**SolutionStatement**

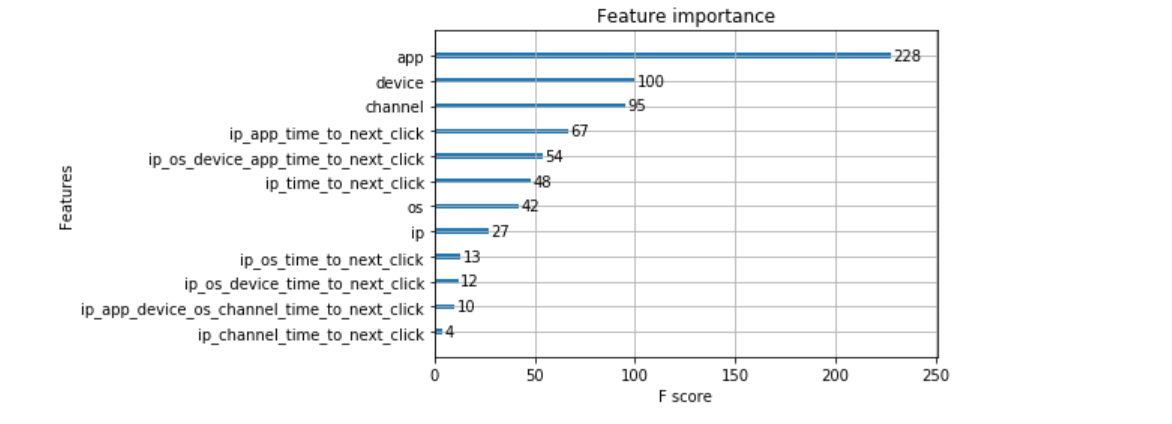
Autoencodersareaparticularclassofneuralnetworksthattakenaninput,”compress”that inputdowntocorefeaturesandtriestoreconstructtheoriginalinputfromthissqueezed representation.Thiskindofapproach,isusedforapplicationsinwhichwewanttorecognise anomaliesor”distortions”ondata.

WiththisprojectIwanttoexperimentwithautoencodersinordertodetectfraudsand measureitsresults.Theaimofthisprojectistomeasurethequalityofthistechniquetomake acomparisonwithothers.ThereasonwhyIchoosethismodelisthatIwanttoplayalittle bitwithan”alternative”and(maybe)wrongmodel,tomeasurehowmuchdoesitdifferfrom

mostusedimplementations.Thehugeclassimbalanceislikelytocausehugeproblemsfor anautoencoderifIdon’tusesomeverycleverdataaugmentation.Perhapsbyintegratingan adversarialfactorintoautoencoders(ontheencoderend),resultscouldbegoodenoughtobe comparedwith”standars”solutions.

BenchmarkModel (Justification)

Inordertomeasurethequalityofthisexperimentandtoevaluatetheresults,weneedsome referencemodel.Itseemsreallydifficulttofindothersthatconductedandsharedtheresult ofthiskindofanalysis.Aftersomeresearch,Ifoundtwoothersolutionsontheexactsame datasetthatIselected.Reference[5]recognisefraudsusingmachine learning algorithm XGB with93.5% ofaccuracywhilereference[6]



**EvaluationMetrics**

IwilltrytooptimisetheparametersoftheAutoencoderinsuchwaythatthereconstruction errorisminimised.ROCandPrecision-Recallcurveswillbeusedinordertoevaluatethe results

**Improvement**

The following techniques can be used to improve accuracy:

Logistic Regression

The logistic regression algorithm is a popular algorithm used in classification problems. This   
algorithm works by learning a numerical weight for each feature and a constant term from our   
training dataset. Given a new point from the test dataset, it multiplies each feature by the   
corresponding learnt weight and adds all the product together with the learnt constant term.

The result of this operation is then fit into a logit function which scales it down to a number between 0 & 1. The final number gotten can be interpreted as probabilities with results lesser than 0.5   
indicating a prediction of false and results greater than or equals to 0.5 indicating a

prediction of true.

The following parameters can be tuned to optimize the logistic regression classifier:   
I. Penalty - specifies the kind of regularization option: L1 OR L2

II. C – Inverse of regularization strength, smaller values specify stronger

regularization.

The choice for picking the logistic regression algorithm was due its good probabilistic

framework, which is useful when you want to adjust classification threshold. Also it’s   
relatively fast handling >100,000 data points easily. Also given the nature of the problem, as new transactions are processed there would be a need to easily update the model to take in new data which logistic regression supports.

The only problem that might arise from using logistic regression is that it typically creates a linear decision surface which might not be appropriate if the problem requires a more   
complex decision surface.

Decision Tree

Decision tree is a machine learning algorithm with the goal of predicting a target variable by learning simple decision rules inferred from the data features.

Here the max\_depth parameter which specifies the maximum depth of the tree is the major parameter for tuning.

The decision tree algorithm predicts labels by decision rules which seems logical given the   
problem of predicting fraudulent transactions. It is not prone to noise and outliers.

The decision tree is notorious for overfitting and might be a problem if the problem requires a linear decision surface.

Gaussian Naïve Bayes

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems.

Naïve Bayes stems from the Bayes theorem which involves computing the probability of an event based on the probabilities of certain related events. While the Bayes theorem fails at using more than feature, Naïve Bayes allows for multiple features and the “naïve” in Naïve Bayes implies these features are independent of each other.

Gaussian Naïve Bayes is simply an extension of the Naïve Bayes algorithm to   
numerical/continuous features, most commonly assuming a Gaussian distribution. The   
Gaussian Naïve Bayes algorithm works well out of the box and tuning its parameter is rarely ever necessary.

The Gaussian Naïve Bayes algorithm works well out of the box and more often than not   
requires little or no parameter tuning to increase performance. It also performs well even   
with the presence of irrelevant features and its relatively unaffected by them, now this is   
important because the features are anonymous and there is no effective way of having   
intuitions about features that might explain the target variable and weed out irrelevant   
features.

Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) is a neural network, which consists of at least three layers of   
nodes. Except for the input nodes, each node is a neuron that uses a nonlinear [activation](https://en.wikipedia.org/wiki/Activation_function)   
[function.](https://en.wikipedia.org/wiki/Activation_function)

Its multiple layers and non-linear activation makes it suitable for data that is non-linearly   
separable.

For this project the MLP would have just 1 hidden layer consisting of 21 nodes with a relu   
activation function. The output layer would consist of just one node with a sigmoid   
activation function that outputs numbers between 0 & 1 to represent the probability of   
predicting a value for fraudulent transactions.

The Multi-Layer Perceptron deals with a complex decision boundary very well but might also be a problem if the decision surface in the dataset is linear and might risk overfitting. The   
choice of the MLP classifier was to introduce a neural network algorithm to already existing options and to deal with non-linearity if it exists in the dataset. III. Methodology

Data Pre-processing

The first step taken in pre-processing the dataset was **Normalization.** The Normalization   
procedure was applied only on the **Amount** Feature since it wasn’t on the same scale as the other features.

Next, the **SMOTE** algorithm was used to balance out the class ratio. It works by constructing new points from the minority class until it evens out the deficit in the class ratio.

**Fig 2.** Distribution of the two classes before and after the SMOTE algorithm was used the   
balance the dataset.

**ProjectDesign**

Theprojectwillbedevelopedfollowingthosesteps:

1.Iwillconductatfirstanexploratoryanalysistohaveabetterunderstandingofthedata;

2.theneuralnetworkwillbetrainedandfixedwiththerightparameters;

3.attheendIwillevaluatetheresultsandIwillcomparethemtothebenchmarkreference

**Conclusion:**

The machine learning algorithm XGB used was most accurate model to recognize frauds

with93.5% ofaccuracy.

References

[1]CreditCardFraudDetection-Kaggle.Availableat:<https://www.kaggle.com/mlg->   
ulb/creditcardfraud.

[5]CaseStudy:HowtoImplementCreditCardFraudDetectionUsingMachine Learning.   
Availableat: https://www.altexsoft.com/whitepapers/fraud-detection-how-machine-learning-systems-help-reveal-scams-in-fintech-healthcare-and-ecommerce/

[[6]](https://www.romexsoft.com/blog/implement-credit-card-fraud-detection/%5b6%5d)CreditCardFraudDetectionviaXGB Classifer and Random Forest.Availableat:   
<https://www.kaggle.com/yuridias/credit-card-fraud-detection-RandomForest>