Credit Card Fraud Detection

Anshupriya Srivastava

12/10/2019

Summary

Credit card companies must recognize fraudulent transactions to ensure that their customers are not charged for items that were not purchased by them. To avoid this, a model is being created that helps these companies identify fraudulent transactions. Since the data available is highly imbalanced, various techniques are used to adjust the class distribution of the data set before running classification models on the data. Recall is our most important indicator since we would rather detect regular transactions as fraudulent, instead of charging a customer for something they did not purchase. The most effective model has a recall of 98.03% on the test set (obtained after oversampling the data and splitting and train-test data) and 93.90% when tested on the entire data set.

Introduction

The development of contactless payment systems, the growth of mobile technology, and the creation of Open Banking are slowing down the use of cash. As purchases migrate online, due to the higher volume of cashless transactions and more access points for the average consumer, there is an increased risk of crimes such as identity theft, account takeover, fraudulent transactions, and data breaches. Identity theft is one of the most common consequences of data breaches. According to Javelin, 31.7% of breach victims eventually suffered identity fraud in 2016 compared to just 2.8% of unaffected victims in 2016. The most common form of identity theft was credit card fraud (133,015 reports). This project aims to identify fraudulent transactions so that credit card users do not have to pay for items they did not purchase.

Data

The data-set comprises transactions made in September 2013 by European credit cardholders. The data set has been collected and analysed during a research collaboration of Worldline and the Machine Learning Group (http://mlg.ulb.ac.be) of ULB (Université Libre de Bruxelles) on big data mining and fraud detection. This data-set contains two-day transaction details where 492 of 284,807 transactions were fraudulent. The data-set is highly imbalanced, with the positive category (fraud) accounting for 0.172% of all transactions.

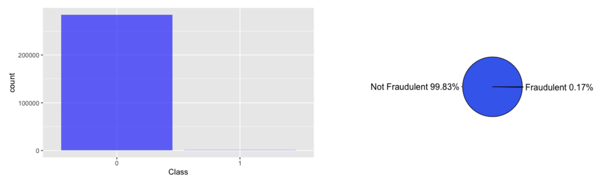


Fig 1. Class Distribution

Numerical variables resulting from PCA (Principal Component Analysis) transformation are in the data-set. However, due to data privacy issues, the original features and more background information are not available. V1, V2...V28 are the key PCA components collected. The only features not transformed with PCA are 'Time' and 'Amount.' The 'Time' feature includes the seconds from the first transaction in the data-set. The 'Amount' feature is the payment amount used for cost-sensitive learning. 'Class' is the response parameter, and in case of fraud, it takes value 1 and 0 otherwise. It is slightly difficult evaluate if there are any missing or erroneous values in this data-set since all variables have PCA transformed. A plot describing the distributions and correlation of all variables in the data-set is available in the appendix. The correlation matrix is not reliable due to the imbalance in the data set. The next pair of graphs explore the relationship of Amount with Class and Time.

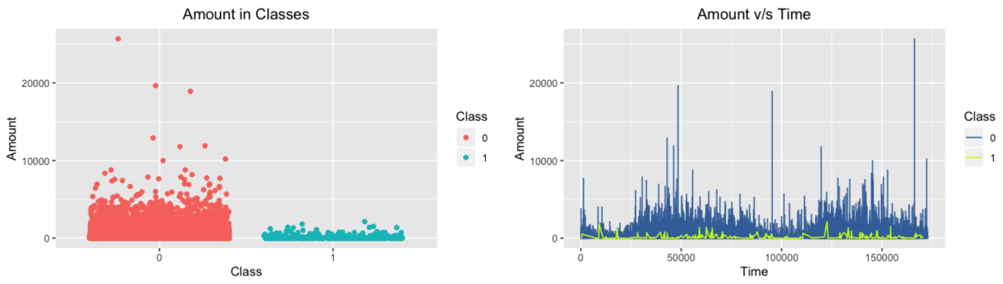


Fig 2. Distribution of Amount with Class and Time

The first plot shows that high-value transactions are not fraudulent. Fraudulent transactions are of lower value. The fraudulent transactions amount have a mean of 122.21 ,whereas non-fraudulent transactions have a mean of 88.29. Despite having a higher mean value, the maximum amount spent during a fraudulent transaction is 2125.87 as compared to non-fraudulent transactions where the highest amount spent is 25691.16. This implies that fraudulent transactions are not very evident. A consumer may not even realize that their card is being used by someone else for a prolonged period due to the low value of transactions. The second plot demonstrates the relationship between Time and Amount classified on Class. Once again, it's quite evident that the amount value of fraudulent transactions is fairly low as compared to non-fraudulent transactions. Also, there doesn't seem to be a pattern based on time. A fraudulent transaction can occur anytime. So, the variable time is being excluded from this analysis.

The graph below describes the distribution of one of the known variables, Amount. It is evident that amount is heavily skewed. Performing a log transformation on this variable creates a more normal distribution, which will be used throughout the analysis.

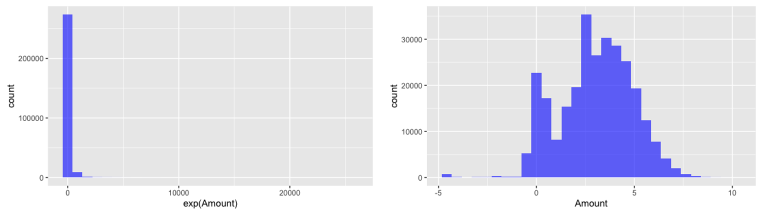


Fig 3. Amount Distribution

The following analysis is based on the two techniques used to balance the data set. The first method used is Undersampling. Undersampling works by sampling the dominant class to reduce the number of samples. The method used here is selecting a few samples from the Fraudulent class randomly and using them with the non-fraudulent transactions in a new data set. A correlation matrix is available in the appendix. Class shows high positive correlation with V2, V4, V11 and high negative correlation with V14, V12, V10. The graph below shows an equal distribution in both classes along with a t-SNE plot. t-SNE (t-Distributed Stochastic Neighbor Embedding) is a non-linear dimension reduction algorithm used to study high-dimensional data. It maps two or more dimensions of multi-dimensional data for analysis.

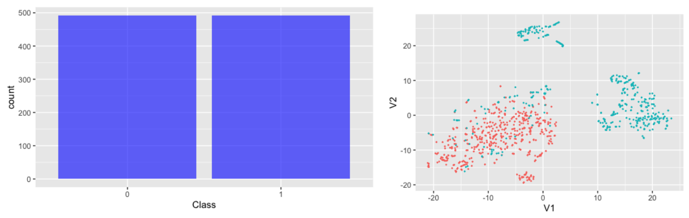


Fig 4. Class Distribution after Undersampling

There are equal instances of fraudulent and non-fraudulent transactions. Also, clear clusters of fraudulent and non-fraudulent transactions are visible in the second graph. This suggests that the classification algorithms that will be used on this data will be able to segregate between the classes. A correlation matrix is available in the appendix. Class shows high positive correlation with V2, V4, V11 and high negative correlation with V14, V12, V10. A similar pair of graphs are visualized for the second method used to balance the data-set - Oversampling. Oversampling artificially creates observations in the data set belonging to the class that is underrepresented. The technique used to oversample is SMOTE (Synthetic Minority Over-sampling Technique). SMOTE creates synthetic observations of the minority class (in this case, fraudulent transactions). SMOTE finds the k-nearest-neighbors for minority class observations and randomly chooses one of the k-nearest-neighbors and using it to creates a similar, but randomly tweaked, new observations.

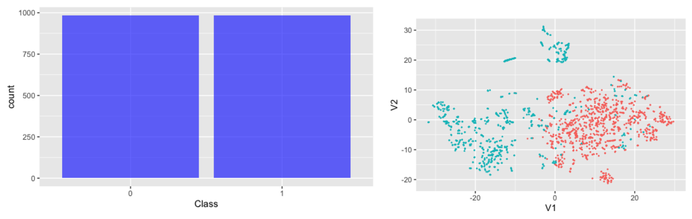


Fig 5. Class Distribution after Oversampling

There are equal instances of fraudulent and non-fraudulent transactions since the algorithm is tuned to created equal counts of both cases. There are a total of 1968 observations equally divided between fraudulent and non-fraudulent cases. Also, clear clusters of fraudulent and non-fraudulent transactions are visible in the second graph. This suggests that the classification algorithms that will be used on the data will be able to segregate between the two classes.

Model

The undersampled and oversampled data-sets are split into training and testing data-sets using a 3:1 ratio that is 75 percent of data is used as training data and 25 percent as testing data. All models have been tested on the testing data-set as well as the entire data-set (original data-set). The test results are compared and a model is selected based on the highest **Recall** value. Recall is the most important evaluation metric in this case. This is because the requirement is to minimize false negatives. Recall is a model's ability to find all the data points of interest in a data-set. The formula for recall is given below.

$$recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

In this case the requirement is to identify fraudulent transactions (data points of interest) from the perspective of a credit card company. The cost of False Negatives (or a fraudulent transaction being categorized as a non-fraudulent transaction) is much higher than False Positives (categorizing a non-fraudulent transaction as a fraudulent one).

$$recall = \frac{fraudulent\ transctions}{fradulent\ transactions + transactions\ incorrectly\ labeled\ as\ fraudulent}$$

Even though Recall is used to evaluate these models, it is also interesting to observe Precision and F1-Score. There is a recall-precision trade-off. Precision expresses the proportion of positive identifications that are actually correct. F1 score is a combination of Precision and Recall.

Three classification algorithms (**Logistic Regression**, **K-nearest-neighbor** and **Decision Trees**) are used on the training sets obtained. The reasons behind picking these three modeling techniques are explained below:

Logistic Regression : The primary reason behind using this method is that it has been taught in class. Apart from that, Logistic Regression is incredibly easy to implement and very efficient to train. Because of its simplicity Logistic Regression is also a good baseline that can be use to measure the performance of other more complicated Algorithms.

K-nearest-neighbor : KNN is a popular clustering algorithm because it is simple to implement, robust to noisy training data, and effective if training data is large.

Decision Trees : Decision trees can learn non-linear relationships, and are reasonably robust to outliers. Decision Tree is simple to visualize and requires little data preparation.

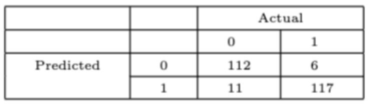
Undersampling

The undersampled data-set consists of 984 observations. This is split into training and testing such that there are 738 observations in the training data set and 246 observations in the testing data set.

All independent variables are used in the logistic model [apart from time as mentioned above] to predict the class. A table with the logistic regression output is given in the appendix. Since this model can help understand the impact of each variable on the final output it is a good idea to study it's results. The model suggests that V2, V3, V13, V21 - V26, V28 and Amount as not significant. However, this is not an accurate representation in this case for two reasons. First of all, there is no description available for the given independent variables. Secondly, these variables are PCA transformed. It is unreasonable to drop them from the model. So, the final model involves all independent variables.

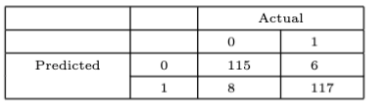
The tables below show the confusion matrix of the test data set with 246 observations.

Table 1: Logisistic Regression



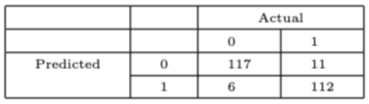
		Actual	
		0	1
Predicted	0	112	6
	1	11	117

Table 2: k-nearest neighbors



		Actual	
		0	1
Predicted	0	115	6
	1	8	117

Table 3: Decision Trees



		Actual	
		0	1
Predicted	0	117	11
	1	6	112

All three models seem to be able to classify fraudulent and non-fraudulent transactions effectively. The table below describes the evaluation metric used to validate the model.

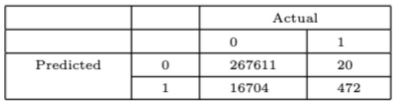
Table 13: Comparison of Evaluation metrics on Testing Data (Undersampling)

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	93.09%	91.41%	95.10%	93.23%
KNN	94.31%	93.60%	95.12%	94.35%
Decision Trees	93.09%	94.91%	91.06%	92.94%

KNN seems to perform best is this situation with the highest recall value of 95.12 percent.

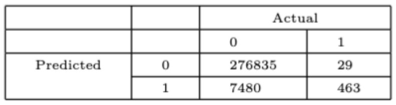
The next set of observations are made by testing the same models on the entire data-set that has imbalanced data points.

Table 4: Logisistic Regression



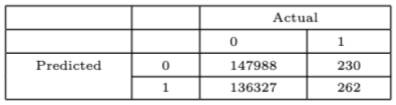
		Actual	
		0	1
Predicted	0	267611	20
	1	16704	472

Table 5: k-nearest neighbors



		Actual	
		0	1
Predicted	0	276835	29
	1	7480	463

Table 6: Decision Trees



		Actual	
		0	1
Predicted	0	147988	230
	1	136327	262

Decision tree performs poorly when tested on the entire data-set. Decision tree is a Greedy algorithm and hence cannot guarantee to return the globally optimal decision tree. Also, while working with continuous numerical variables, decision tree loses information, when it categorizes variables in different categories. It is possible that this model over-fits the training data and that wasn't clearly visible in the test set due to lesser number of observations. However, on a larger data-set it's evident that this model is not the best fit. A comparative study of the evaluation metrics will corroborate the results shown in the confusion matrix.

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	94.13%	2.75%	95.93%	5.34%
KNN	97.36%	5.83%	94.11%	10.97%
Decision Trees	52.05%	0.20%	53.25%	0.38%

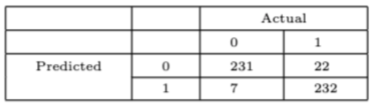
Logistic Regression and KNN have high Recall values which means that they are categorizing Fraudulent transactions quite effectively. Logistic Regression has lesser number of false negatives.

Oversampling

The oversampled data-set consists of 1968 observations with equal instances of fraudulent and non-fraudulent transactions. This data-set is also split into training and testing such that there are 1476 observations in the training data set and 492 observations in testing data set.

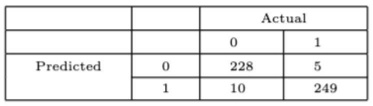
The tables below show the confusion matrix of the test data set with 492 observations.

Table 7: Logisistic Regression



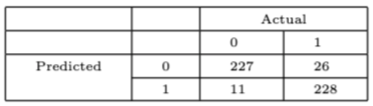
		Actual	
		0	1
Predicted	0	231	22
	1	7	232

Table 8: k-nearest neighbors



		Actual	
		0	1
Predicted	0	228	5
	1	10	249

Table 9: Decision Trees



		Actual	
		0	1
Predicted	0	227	26
	1	11	228

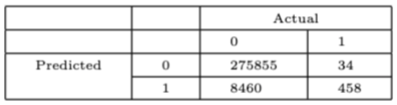
All three models seem to be able to classify fraudulent and non-fraudulent transactions quite effectively. KNN has the lowest number of false negatives. The table below describes the evaluation metric used to validate the model.

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	94.11%	97.07%	91.34%	94.12%
KNN	96.95%	96.14%	98.03%	97.07%
Decision Trees	92.48%	95.39%	89.76%	92.49%

KNN seems to perform best is this situation with the highest recall value of 98.03 percent.

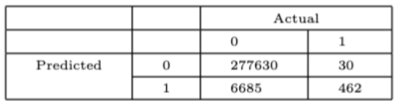
The next set of observations are made by testing the same model on the entire data-set that has imbalanced data point.

Table 10: Logisistic Regression



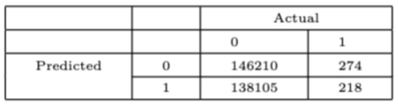
		Actual	
		0	1
Predicted	0	275855	34
	1	8460	458

Table 11: k-nearest neighbors



		Actual	
		0	1
Predicted	0	277630	30
	1	6685	462

Table 12: Decision Trees



		Actual	
		0	1
Predicted	0	146210	274
	1	138105	218

Decision tree performs poorly even when trained on the oversampled data-set.

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	97.02%	5.13%	93.09%	9.734%
KNN	97.64%	6.46%	93.90%	12.09%
Decision Trees	51.41	0.16%	44.31%	0.31%

Logistic Regression and KNN have high Recall values which means that they are categorizing Fraudulent transactions quite effectively. However, the false positives are quite high.

Conclusion

In conclusion, it is evident that KNN outperforms the other algorithms. Using an oversample data set and training it using KNN will produce the most desirable results. Although, KNN provides with a very high recall value and less number of false negatives it does not have a high precision value. A low precision value indicates high number of false positives. False Positives can cause dissatisfaction for a customer who tends to use their cards often and the card is blocked by the credit card company whilst being falsely categorized in the incorrect category. Another, drawback is using sampled data. Undersampling may cause loss of important information and oversampling tends to over-fit. For future work, it would be interesting to find a way to have a high precision value along with a high recall value (high F1-score).

The GIT link for the project: CREDIT CARD FRAUD DETECTION

Appendix

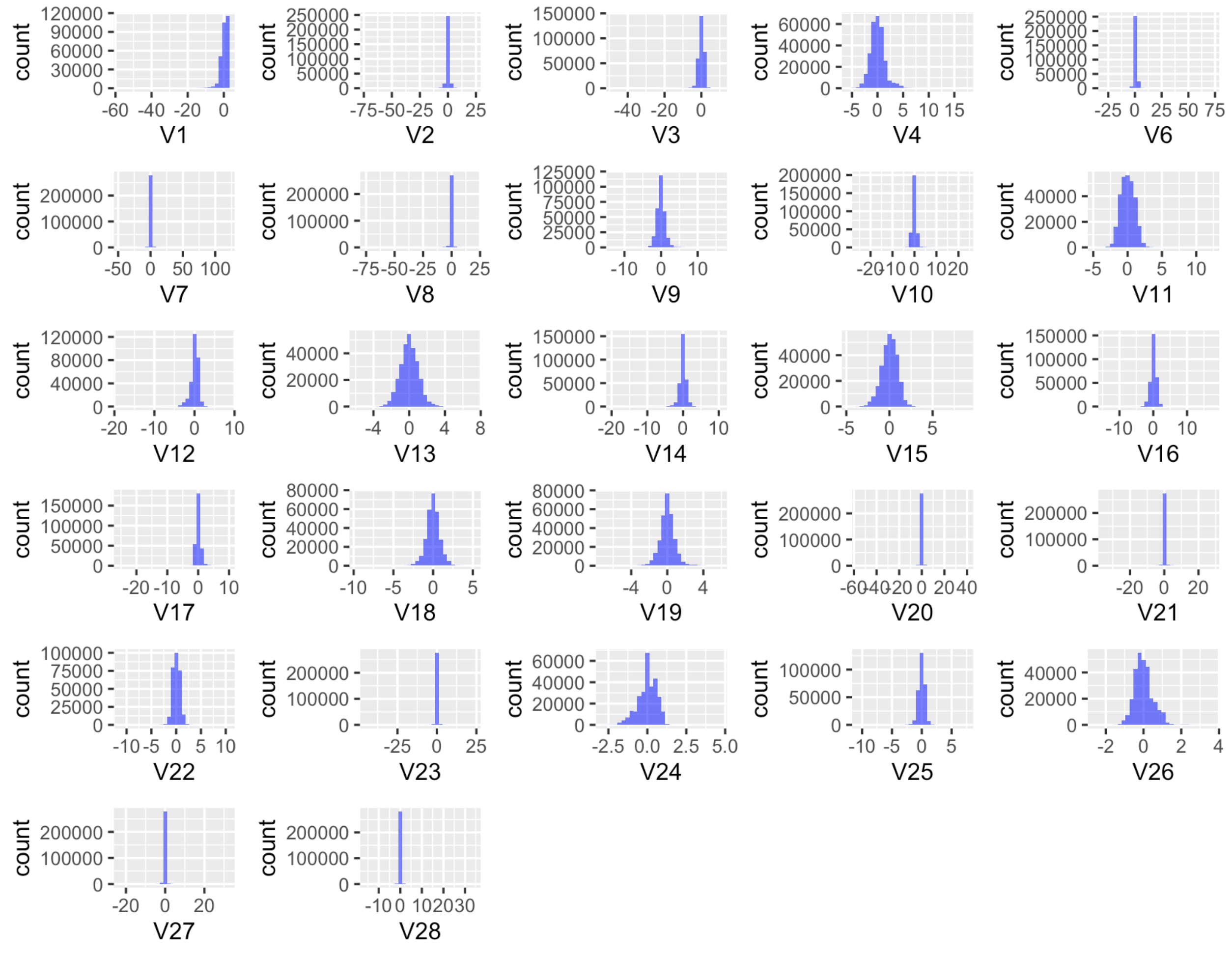


Fig 6. Distribution of all Independent Variables

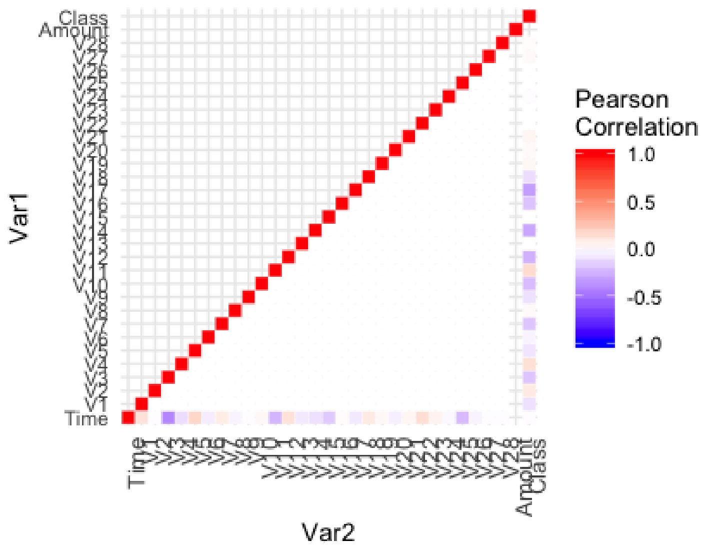


Fig 7. Correlation of all Independent Variables in the entire data-set

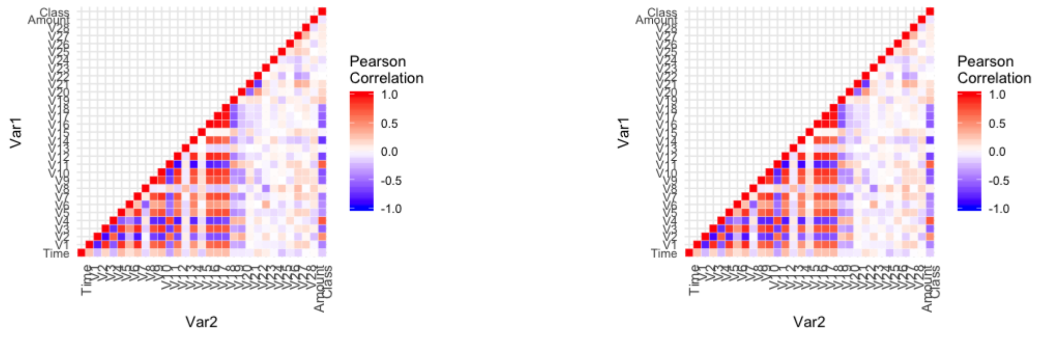


Fig 7. Correlation of all Independent Variables in - (left) Under-sampled; (right) Over-sampled

Logistic Regression output for undersampled data:

NA	NA	NA	NA	NA
	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.62538529011047	1.21037307750368	2.16907112270307	0.0300772846704192
V1	-15.558559300926	3.21287327313321	-4.84256862261898	1.28171348909436E-06
V2	14.8148944091952	3.14697936551844	4.70765540172348	2.50582237099048E-06
V3	-36.2515620133052	7.40958336627415	-4.89252367121068	9.95511383169843E-07
V4	23.1103469243814	4.52924728266634	5.10246967809108	3.35249473860172E-07
V5	-25.05198863399	5.11168805198485	-4.90092282220984	9.53875261606946E-07
V6	-8.75518647795621	1.77511798589298	-4.93217157819056	8.13204353791349E-07
V7	-47.0323221456843	9.68763745812304	-4.85488049578568	1.20459365169688E-06
V8	9.90310806452409	2.15109881751362	4.6037439023702	4.14962697729417E-06
V9	-23.9858811471045	4.84983840914208	-4.94570728416239	7.58678906907615E-07
V10	-56.6041647018544	11.3246689477862	-4.99830634898336	5.78360492618572E-07
V11	39.5593041657036	7.97656435895983	4.95944148200445	7.06961391133952E-07
V12	-70.9522842367013	14.2742612829348	-4.97064491326961	6.67305638865892E-07
V13	0.48454403753607	0.215896234056379	2.2443376080823	0.0248106875533102
V14	-73.9926366263533	14.8166466102481	-4.99388549735508	5.91764903649896E-07
V15	-1.41452464148272	0.34942545937634	-4.04814418504991	5.16253416424805E-05
V16	-65.8752229281909	13.3646574072734	-4.92906184728238	8.26254172112015E-07
V17	-119.977687422474	24.2881536050238	-4.93976155510058	7.82181602099686E-07
V18	-44.5800852829159	9.06433616403033	-4.91818534487076	8.73501865081205E-07
V19	14.619707893311	2.98535045590605	4.89714963427095	9.72367944190319E-07
V20	7.25490016696931	1.64280391385848	4.41616927362292	1.00465421523867E-05
V21	9.71573676407097	1.97542688251671	4.91829733110295	8.73002395736128E-07
V22	1.11415450475189	0.278409527025099	4.00185480955705	0.000062847861914012
V23	-0.193781305235393	0.370459540100493	-0.52308358743529	0.600916086403356
V24	-0.847682022799446	0.396925505618182	-2.1356199357339	0.0327104051313589
V25	2.88466608243964	0.709906385170322	4.06344574819896	4.83535647641454E-05
V26	0.509378643796477	0.459595630344121	1.10831916181423	0.267724002850681
V27	12.4828822227599	2.60875789255205	4.78499068786654	1.7099526046043E-06
V28	10.3349090884551	2.33956267526373	4.41745339747743	9.98705942641716E-06
Amount	-0.201570977279637	0.099917210017247	-2.0173799613184	0.0436558768138331

Logistic Regression output for Over-sampled Data:

NA	NA	NA	NA	NA
	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-4.41003141370458	0.669767481095107	-6.58442151669402	4.56659869675183E-11
V1	-0.439571965867951	0.285320310745426	-1.54062626919033	0.12340777073209
V2	-1.26869466941313	0.680333604262932	-1.86481258821196	0.0622076781908763
V3	2.40029487983137	0.662341106650122	3.62395577706354	0.000290131233875001
V4	-0.446041339794489	0.533466329980742	-0.836119010192434	0.403087954799071
V5	-0.773335000107841	0.318150325121022	-2.43072201737864	0.0150687707895872
V6	-1.58527694378026	0.952388691564071	-1.66452726478391	0.0960071071303294
V7	-1.24703634474932	0.29099642350022	-4.2854009329375	1.82409920257289E-05
V8	-1.86093574026057	0.764100438217423	-2.43545959036741	0.0148728857265102
V9	-4.29700622556828	1.63726347568248	-2.62450502890325	0.0086775011164145
V10	2.89710261288583	1.15325839326709	2.51210191037809	0.0120014425171559
V11	-5.25498570367189	2.00786736132637	-2.61719763211874	0.00886549947446013
V12	-0.193079259588372	0.223097492567974	-0.865447914120072	0.386792992297462
V13	-5.63243232980325	2.07620549181741	-2.71284916257152	0.006670746324855
V14	-0.076400336204726	0.223298154101178	-0.342144951946663	0.732241809795153
V15	-3.97372817354396	1.81937566305515	-2.1841163725754	0.0289536978963472
V16	-7.33752628849986	3.26207531076034	-2.24934300697969	0.0244906816069468
V17	-2.50717671295161	1.19949088587917	-2.09020071971115	0.0365997729246007
V18	1.47645511023848	0.541715314169115	2.72551849951496	0.00642005785931307
V19	-0.7145912028532	0.415222583315504	-1.72098347143663	0.0852538232312078
V20	0.47393034168979	0.392209232989865	1.20836100179732	0.226908433546072
V21	0.198690654491482	0.318984726144621	0.622884540250369	0.53336039796626
V22	-0.43332878789697	0.274210185187157	-1.58027969530457	0.114042827717377
V23	-0.345817750734661	0.461623153339051	-0.749134327932346	0.453776246503917
V24	-0.11530418687914	0.473786696338163	-0.243367295389067	0.80772087678322
V25	0.150417745177193	0.488988091105667	0.307610242280295	0.758378925674218
V26	2.33852465039963	1.09046181029905	2.1445268677115	0.0319906950076686
V27	1.27740441678435	1.23169722736713	1.03710911123419	0.299685009430377
V28	0.0962214429313794	0.142112232642614	0.677080650568333	0.498354784915316
Amount	-0.201570977	0.09991721	-2.017379961	0.043655877

Acknowledgements

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

Dal Pozzolo, Andrea; Caelen, Olivier; Le Borgne, Yann-Ael; Waterschoot, Serge; Bontempi, Gianluca. Learned lessons in credit card fraud detection from a practitioner perspective, Expert systems with applications,41,10,4915-4928,2014, Pergamon

Dal Pozzolo, Andrea; Boracchi, Giacomo; Caelen, Olivier; Alippi, Cesare; Bontempi, Gianluca. Credit card fraud detection: a realistic modeling and a novel learning strategy, IEEE transactions on neural networks and learning systems,29,8,3784-3797,2018,IEEE

Dal Pozzolo, Andrea Adaptive Machine learning for credit card fraud detection ULB MLG PhD thesis (supervised by G. Bontempi)

Carcillo, Fabrizio; Dal Pozzolo, Andrea; Le Borgne, Yann-Aël; Caelen, Olivier; Mazzer, Yannis; Bontempi, Gianluca. Scarff: a scalable framework for streaming credit card fraud detection with Spark, Information fusion,41, 182-194,2018,Elsevier

Carcillo, Fabrizio; Le Borgne, Yann-Aël; Caelen, Olivier; Bontempi, Gianluca. Streaming active learning

strategies for real-life credit card fraud detection: assessment and visualization, International Journal of Data Science and Analytics, 5,4,285-300,2018,Springer International Publishing

Bertrand Lebichot, Yann-Aël Le Borgne, Liyun He, Frederic Oblé, Gianluca Bontempi Deep-Learning Domain Adaptation Techniques for Credit Cards Fraud Detection, INNSBDDL 2019: Recent Advances in Big Data and Deep Learning, pp 78-88, 2019

Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, Frederic Oblé, Gianluca Bontempi Combining Unsupervised and Supervised Learning in Credit Card Fraud Detection Information Sciences, 2019