**Abstract**

The credit cards frauds is a big problem in many parts of the world. When talking about frauds in credit cards, most people think about the values defrauded, However the total cost of the frauds has to be understood in its whole extension:

1. There is the loss due the fraud itself (what most people see)
2. There is the cost associated to managing the losses – cancelling orders and refunding charges (exceeds the cost of the frauds 300%)
3. There is the cost of mistakenly rejecting orders
4. There is the cost of developing and applying mechanisms to avoid frauds.

Therefore, the problem is not only what you lose due the frauds themselves, but the cost to manage this process (Update security measures, block cards, reimburse customers etc), the cost of building and maintaining mechanisms to prevent the frauds and of course the cost of losing money when you unduly blocks a sales assuming wrongly it is a fraud.

Therefore, the challenge is to build a model which is ease to maintain, highly effective in spotting frauds and at the same time doesn´t have a too high false positive rate. AI/ML seems to be well placed to address all these issues. It automatize the process of adjusting for identifying new types of frauds (the big problem with the current processes) and it can archive an effective identification rate without too many false negative.

**Problem statement /Business problem:**

Our objective is to spot possible frauds in credit card operations. This identification will be based on the client´s profile, the seller´s profile and in the data of the transaction itself. The objective is to identify the transactions to be red flaged as high possibility of fraud.

Today frauds spin around 4% of all sales made with credit cards in Brazil. Today the credit card company already has mechanisms in place to try to spot potential problems, however these mechanisms have two sorts of problems:

1. The fact that they deploy semi-static rules generates a scenario where people in the background have to keep looking for new types of frauds and adjusting the defences.
2. The process suffers a paradoxical problem: if it is too stringent it creates problems for the clients blocking legit sales, if it is too loose it allows a too high level of frauds. A middle term is difficult to archive – even more so when you have to keep adjusting the rules.

To address these two issues, the idea would be to create an AI/ML algorithm which would not only identify (or at least red flag) the suspect transactions but also would identify changes in the patterns and adapt automatically to new fraud patterns.

In order to do that we managed to get a database merging the three instances of data (client, seller and transaction) and the idea is to develop a AI/Machine learning model which not only assertive (Assertive meaning identify a high percentage of the actual frauds without blocking too many legit ones) but adaptable.

We were verbally informed that the current mechanism spots around 50% (2% of the total) of potential frauds and red-flags around 2% of legit ones. If that is true (We have no way to verify this information), that means the current process gets it right in 2% of the cases and wrong in 2%. In summary it is not able to stop 50% of the frauds and has a false positive rate of 2%.

It is interesting to notice that although frauds respond for just 4% of the transactions they answer for 8% of the value of the transactions. That means each 1% of fraud eliminated responds for approximately R$ 16.000.000,00 month (CAD 5.330.000,00)

**Datasets/Getting the data**

We used a real sanitized dataset representing a subset of the transactions occurred during one day of September of 2019 in Brazil, totalizing 290.398 transactions.

**Data exploration/Evaluating the data/ Cleaning the data**

Accuracy and completeness: A visual inspection showed that the data was basically correct. No missing data was identified.

**Stan Insert here**

**Selecting the training data and the test data**

The idea is to use 95% of the measurements as our training data and 5% as our test data. We had a database with 290.398 clients and we separated it into training data 275.878 and test data 14.520.

**The meaning of the columns (data dictionary)**

This database is a crossing of three sources:

* Clients data
* Sellers data
* And the transactions proper.

Information regards the clients:

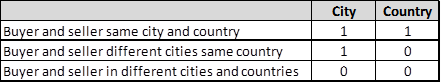
* Age
* Sex
* Income
* Average expenditure  in recurrent  payments  per transaction
* Average expenditure  buying goods (Using the card directly) per transaction
* Average expenditure buying services (using the card directly) per transaction
* Average expenditure buying goods on-line per transaction
* Average expenditure buying services on-line per transaction

Information about the seller:

* Score of the seller (Number 0 to 100) indicating the ranking of the seller regards frequency of frauds.

Information about the transaction:

* Type: which one of the five categories it belongs (0-recurrent, 1- goods, 2-services, 3-on-line good or 4-on-line services).
* Value of the transaction
* If the addresses of the seller and the buyer  are in the same city
* If the addresses of the seller and the buyer  are in the same country
* If the transaction was fraudulent or not (Y or N)



**Approach /Analytical problem**

Initially we realized that the variable value only has a meaning as long it is compared with the average expenditure with this particular type of transaction. Given this fact we realized that it would be necessary to create a synthetic variable called “Dispersion”. Dispersion is created comparing the value of the transaction and its type with the average expenditure with this particular type. Example: If value is equal R$ 20,00 and type equal 0. That means this is a recurrent transaction. The average expenditure with recurrent transactions is R$ 25,00. In that case “Dispersion” will be: absolute (25-20)/25 -> 0.20. This variable gives us a view of how far from the average the value of transaction is.

In sequence we evaluate if we could deploy the method decision-tree directly to identify the frauds. The results were a bit disappointing, although the accuracy seems good it happens just because the data is unbalanced and the frauds are  4% of the total.

The objective is to actually say if the transaction is a fraud or not (not just calculate the chance it to be). Note it identified correctly only 26% of the frauds (157 out of 588), although with a very low false positive (0,2%).

Confusion Matrix and Statistics

          Reference

Prediction     N     Y

         N 13898   431

         Y    34   157

               Accuracy : 0.968

                 95% CI : (0.965, 0.9708)

    No Information Rate : 0.9595

    P-Value [Acc > NIR] : 4.669e-08

                  Kappa : 0.391

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9976

            Specificity : 0.2670

         Pos Pred Value : 0.9699

         Neg Pred Value : 0.8220

             Prevalence : 0.9595

         Detection Rate : 0.9572

   Detection Prevalence : 0.9868

      Balanced Accuracy : 0.6323

However, the method indicated how many clusters would be ideal (74) and also gave us an idea about how to procced. The tree also showed an interesting thing: The personal data has almost nothing to do with the results. The relevant factors identified were:

* Score of the seller – Indicates the seller is often target for frauds (probably due type of service or lack of internal controls).
* The type of transaction – Internet services have a way more frauds than everybody else
* City – Transactions where the buyer and the seller are in different cities have much more frauds (they tend to overlap with internet services sales).
* Dispersion (How far from the average spent with this kind of item the value of the operation is)

Then, we decided to create a second synthetic variable which we called V11. To get to the V11 we grouped the transactions using k-means and checked the % of the frauds in each cluster. In sequence we verified to which cluster the transaction belongs and load this % into V11.

In the sequence when, we grouped the transactions by clusters we noticed that there were direct and inverse correlations between some of the variables and the frequency of occurrence of frauds. Analyzing it further we realized that we could measure this correlation in a variable. Therefore we created a third synthetic variable which we called “Pontos”.

The correlations identified were as follows:



That understanding allows us to create an algorithm which gives the transaction a score between 1 and 8. That variable becomes another of our indicators. Therefore we created three synthetic variables “dispersion”, “V11” and “pontos”.

Understanding the correlation between the variables

fraud total percentage pontos incomemean city count seller disper type V11 V12

1 0 0 0.00000000 0 0.0000000 0.00000000 0.0000000 0.0000000 0.00000000 0.0000000 0.00000000 0.0000000

2 11129 257284 0.04325570 1 0.2224684 0.78694773 0.8936676 0.4836477 0.03171155 0.5155814 0.04148007 0.9585199

3 10665 176858 0.06030262 2 0.2193214 0.69504541 0.8478000 0.3839804 0.04185849 0.6262858 0.05724616 0.9427538

4 9028 84225 0.10718908 3 0.2374615 0.38884540 0.6939401 0.3978975 0.07532368 0.8855479 0.09684283 0.9031572

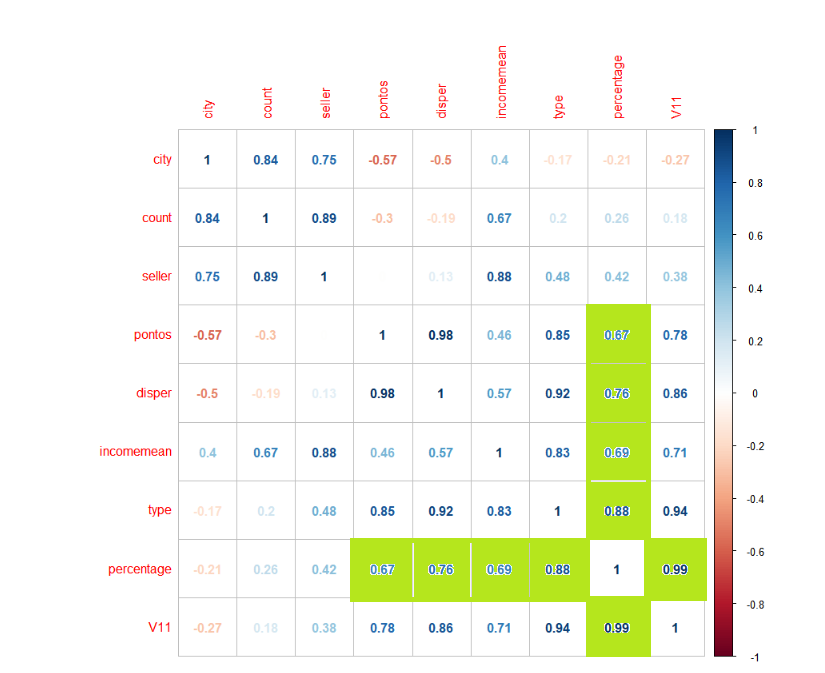
5 8539 68613 0.12445163 4 0.2299642 0.28858617 0.6347340 0.3975054 0.08782032 0.9205853 0.11061024 0.8893898

6 7797 55553 0.14035246 5 0.2170008 0.26569850 0.6040410 0.3427865 0.09242719 0.9258642 0.12386626 0.8761337

7 4907 31702 0.15478519 6 0.2181868 0.04520746 0.5522413 0.3000573 0.11571376 0.9813297 0.13721372 0.8627863

8 2056 22233 0.09247515 7 0.2097575 0.00000000 0.3972168 0.2496612 0.12394320 0.9885648 0.09891327 0.9010867

9 787 8812 0.08931003 8 0.2000000 0.00000000 0.0000000 0.2306427 0.13803058 0.9929159 0.09794667 0.9020533



As we can see, the percentage of frauds has strong correlation with five variables:

**V11** => Synthetic variable V11 (come from the clustering process - % of frauds associated with the profile) is also very effective predictor. (0.99)

**Type** => type of the transaction (Recurrent, in person or on-line). (0.88)

**Disper**=> Synthetic variable which indicates how far from the average expenditures of this specific client with that specific kind of item the transaction is. (0.76)

**Income** => Income of the buyer (0.69)

**Pontos** => synthetic variable merging eight parameters (0.67)

The conclusion of our analysis showed that the factors affecting the chance of a fraud have much more to do with the type of the transaction, how far from the average the value of the transaction is and the income of the buyer (wealth people tend to have less frauds – better financial education?). And only after that we have the score of the seller.

Note that the only personal parameter which seems to affect the level of frauds is the income (Age and sex doesn´t seem to have any significant impact). Now with or database adjusted we tested three possible strategies:

* Decision-tree
* Random-Forest
* Random Forest + Bagging

**Summary of findings /Evaluating the results**

**Using Decision-Tree model**

The composition of these variables applied in a Decision tree-method generated the following result:

In a test set composed by 5% of the total database (5% of 290.398 -> 14.520)

* The model identified as fraud  and they were in fact frauds 365 ( 2,50%)
* The model identified as fraud but they were not 78 ( 0,53%)
* The model identified as not being frauds but they were 223 ( 1,53%)
* The model identified as not being frauds and they in fact were not 13.854 (95,41%)

**It spotted 62% of the frauds with a very low false positive rate (78 ->0,5%)**

Confusion Matrix and Statistics

Reference

Prediction N Y

N 13854 223

Y 78 365

Accuracy : 0.9793

95% CI : (0.9768, 0.9815)

No Information Rate : 0.9595

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.6975

Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9944

Specificity : 0.6207

Pos Pred Value : 0.9842

Neg Pred Value : 0.8239

Prevalence : 0.9595

Detection Rate : 0.9541

Detection Prevalence : 0.9695

Balanced Accuracy : 0.8076

'Positive' Class : N

This represents an improvement of 12% in the current model. That means in practical terms every day you would be able to spot something 7.300 frauds, letting just 4.470 scape.

This model adds value of 1.400 additional frauds spotted every day with the additional advantage that it “learns” new patterns if they appear and therefore will demand less effort to maintain.

Of course the actual implementation of the model need to be analyzed with care, tests need to be made and eventual performance issues evaluated. (We need to remember that the whole verification has to occur in few seconds just after the user made the transaction – although in the on-line services we may have the option of letting an order “pending approval”)

**Using Random Forest model**

If instead using decision tree we use random forest the results would be like:

In a test set composed by 5% of the total database (5% of 290.398 -> 14.794)

* The model identified as fraud  and they were in fact frauds 351 ( 2,50%)
* The model identified as fraud but they were not 74 ( 0,5%)
* The model identified as not being frauds but they were 207 ( 1,5%)
* The model identified as not being frauds and they in fact were not 13.162 (95,41%)

**It spotted 63% of the frauds with false positive rate (False positive ->0,5%)**

**Therefore using random forest would be more effective in identifying frauds but would triple the percentage of false positive.**

|  |
| --- |
| Confusion Matrix and Statistics  Reference  Prediction N Y  N 13162 207  Y 74 351    Accuracy : 0.9796  95% CI : (0.9771, 0.9819)  No Information Rate : 0.9595  P-Value [Acc > NIR] : < 2.2e-16    Kappa : 0.7038    Mcnemar's Test P-Value : 3.422e-15    Sensitivity : 0.9944  Specificity : 0.6290  Pos Pred Value : 0.9845  Neg Pred Value : 0.8259  Prevalence : 0.9595  Detection Rate : 0.9542  Detection Prevalence : 0.9692  Balanced Accuracy : 0.8117    'Positive' Class : N      Note the relative importance of the variables in this method   1. Pontos – V12 – Synthetic variable 2. V11 - Kmeans % - Synthetic variable 3. Seller\_scor – Original variable (seller info) 4. Type of transaction - Original variable 5. Dispersion – synthetic variable   Note that this ranking doesn´t match exactly the ranking identified when using correlation. |
|  |
| |  | | --- | |  | |

**Using the** **Random Forest + Bagging model**

In a test set composed by 5% of the total database (5% of 290.398 -> 14.284)

* The model identified as fraud  and they were in fact frauds 427 ( 2,98%)
* The model identified as fraud but they were not 652 ( 4,5%)
* The model identified as not being frauds but they were 146 (0,10%)
* The model identified as not being frauds and they in fact were not 13.059 (91,40%)

**It spotted 75% of the frauds with false positive rate (False positive ->4,5%)**

Reference

Prediction 0 1

0 13059 146

1 652 427

Accuracy : 0.9441

95% CI : (0.9402, 0.9478)

No Information Rate : 0.9599

P-Value [Acc > NIR] : 1

Kappa : 0.4902

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.9524

Specificity : 0.7452

Pos Pred Value : 0.9889

Neg Pred Value : 0.3957

Prevalence : 0.9599

Detection Rate : 0.9142

Detection Prevalence : 0.9245

Balanced Accuracy : 0.8488

'Positive' Class : 0

**Using the Treebag Model using SMOT model**

In a test set composed by 5% of the total database (5% of 290.398 -> 14.520)

* The model identified as fraud  and they were in fact frauds 569 ( 3,91%)
* The model identified as fraud but they were not 1.351 ( 9,30%)
* The model identified as not being frauds but they were 19 (0,13%)
* The model identified as not being frauds and they in fact were not 12.581 (86,64%)

**It spotted 96,76% of the frauds with false positive rate (False positive ->9,30%)**

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 12581 19

1 1351 569

Accuracy : 0.9056

95% CI : (0.9008, 0.9104)

No Information Rate : 0.9595

P-Value [Acc > NIR] : 1

Kappa : 0.4176

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.9030

Specificity : 0.9677

Pos Pred Value : 0.9985

Neg Pred Value : 0.2964

Prevalence : 0.9595

Detection Rate : 0.8665

Detection Prevalence : 0.8678

Balanced Accuracy : 0.9354

'Positive' Class : 0

**Using the Support Vector Machine Classifier model**

In a test set composed by 5% of the total database (5% of 290.398 -> 14.520)

* The model identified as fraud  and they were in fact frauds 334 ( 2,30%)
* The model identified as fraud but they were not 2.304 ( 15,86%)
* The model identified as not being frauds but they were 254 (1,17%)
* The model identified as not being frauds and they in fact were not 11.628 (80,08%)

**It spotted 56,80% of the frauds with false positive rate (False positive ->15,86%)**

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 11628 254

1 2304 334

Accuracy : 0.8238

95% CI : (0.8175, 0.83)

No Information Rate : 0.9595

P-Value [Acc > NIR] : 1

Kappa : 0.1508

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.8346

Specificity : 0.5680

Pos Pred Value : 0.9786

Neg Pred Value : 0.1266

Prevalence : 0.9595

Detection Rate : 0.8008

Detection Prevalence : 0.8183

Balanced Accuracy : 0.7013

'Positive' Class : 0

**Using the XGBoost model**

In a test set composed by 5% of the total database (5% of 290.398 -> 14.520)

* The model identified as fraud  and they were in fact frauds 571 ( 3,93%)
* The model identified as fraud but they were not 1.034 ( 7,12%)
* The model identified as not being frauds but they were 17 (0,11%)
* The model identified as not being frauds and they in fact were not 12.898 (88,82%)

**It spotted 97,10% of the frauds with false positive rate (False positive ->7,12%)**

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 12898 17

1 1034 571

Accuracy : 0.9276

95% CI : (0.9233, 0.9318)

No Information Rate : 0.9595

P-Value [Acc > NIR] : 1

Kappa : 0.4905

Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.9258

Specificity : 0.9711

Pos Pred Value : 0.9987

Neg Pred Value : 0.3558

Prevalence : 0.9595

Detection Rate : 0.8883

Detection Prevalence : 0.8895

Balanced Accuracy : 0.9484

'Positive' Class : 0

Summarizing the methods



**Model/Analytical model**



**Normalization criteria**

* Income (1-5)
* Seller score (0-100)
* City (0-1)
* Country (0-1)
* Type (0-4)
* Dispersion (0-1)
* V11 (0-1)
* Pontos (0-8)

Values divided by the maximum value