Credit Card Fraud Detection Dataset Analysis

The following analysis refers to the Credit Card Fraud Detection Dataset, a public dataset that contains information about normal and fraudulent transactions made by European credit cards in September 2013. Download made on the site https://www.kaggle.com/mlg-ulb/creditcardfraud. The machine learning method – *logistic regression made in the* python programming language for dataset analysis was used.

The *dataset* contains more than 280,000 transactions, of which only 492 are fraudulent. This represents 0.172% of all transactions, that is, we have an unbalanced *dataset,* a possible problem for the model. The dataset has 30 *features,*V1 to V28 (for confidentiality reasons, no more information about *the features, Time and Amount,* the transaction value could be obtained. The Time *feature* contains the elapsed time between the first transaction and the others. It's not relevant information for analysis, so it was over considered. *Amount* is how much was spent on the transaction, so it is a possibly important *feature* for the analysis. V1 to V28 are the rest of the features *that* have been analyzed. The last column of *the dataset* (*Class*) refers to whether or not the transaction was fraudulent. That's the *target of* the study.

The goal is to make a logistic regression model to classify whether an operation is fraudulent or not and verify which features are the most and least relevant to the model.

Table 1 – Data uploded into jupyter notebook

V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	149.62	0
0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	2.69	0
-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	378.66	0
-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	123.50	0
-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	69.99	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
-5.364473	-2.606837	-4.918215	7.305334	1.914428	...	0.213454	0.111864	1.014480	-0.509348	1.436807	0.250034	0.943651	0.823731	0.77	0
0.868229	1.058415	0.024330	0.294869	0.584800	...	0.214205	0.924384	0.012463	-1.016226	-0.606624	-0.395255	0.068472	-0.053527	24.79	0
2.630515	3.031260	-0.296827	0.708417	0.432454	...	0.232045	0.578229	-0.037501	0.640134	0.265745	-0.087371	0.004455	-0.026561	67.88	0
-0.377961	0.623708	-0.686180	0.679145	0.392087	...	0.265245	0.800049	-0.163298	0.123205	-0.569159	0.546668	0.108821	0.104533	10.00	0
-0.012546	-0.649617	1.577006	-0.414650	0.486180	...	0.261057	0.643078	0.376777	0.008797	-0.473649	-0.818267	-0.002415	0.013649	217.00	0

With the data loaded into the *notebook jupyter*, as seen in table 1 (V1 to V5 did not appear in table 1, as there are many data), the inputs and target were separated into 2 *variables.* I*nputs* were the features V1 to V28 and Amount. The target was Class.

For better analysis, the data were randomly shuffled, since they were collected and stored in the same time window. At the end of the study, a model was also made without shuffling to compare the results.

Then, the dataset was balanced, *because the number* of fraudulent transactions against the total is very low and this can harm the algorithm of the model, leading to misinterpretations and results.

The next step was to *separate the dataset* into 80% training data and 20% test data that was used to check the final accuracy of the model with new data.

Table 2 - Values of coefficients and intercept - randomized data

Feature Name	***Coefficient***
V22	1,1664
V4	0,7353
V28	0,7039
V23	0,5223
V18	0,5008
V26	0,2715
V17	0,2032
V7	0,2017
V1	0,1029
V11	0,0952
V16	0,0666
V13	0,0567
V5	0,0131
Amount	0,0003
V9	-0,0996
V19	-0,1653
V21	-0,2211
V2	-0,2831
V14	-0,3245
V10	-0,4452
V6	-0,5153
V8	-0,5528
V15	-0,6523
V20	-0,6758
V24	-0,7697
V25	-0,8224
V27	-0,8461
V3	-0,8540
V12	-1,1498
Intercept	-2,3413

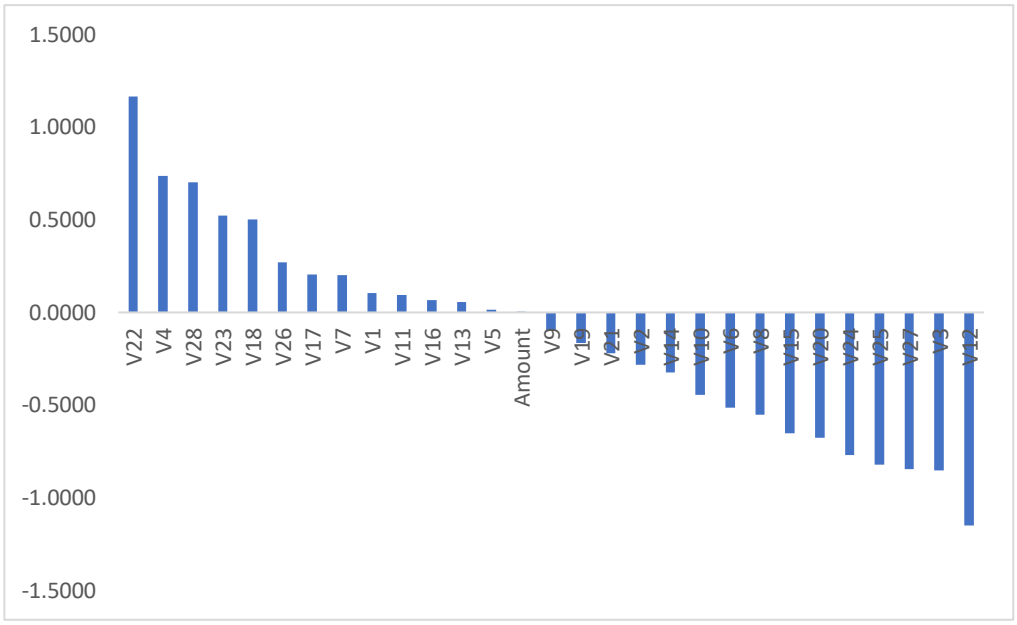

Figure 1 - Coefficient values - randomized data

In table 2, we have the values of the coefficients for the *logistic regression model* and in Figure 1 the visual representation of the values. It can be seen that the Amount *value* is close to zero and is not impacting on the model, that is, in practice, the value of transactions is not a relevant factor for classifying whether the transaction is fraudulent or not. It can be a transaction of R$5.00 or R$10,000.00, it is not relevant. Other features that are not relevant are V9, V5. You cannot verify the reason for this behavior because information about what each feature represents was not *provided.* V22 and V12 are the most important features for the model, but at least the reason presented earlier, it is not possible to comment on the reason for its impact.

Table 3 - Model accuracy values - randomized data

Accuracy	
Dados Treino	96,57 %
Dados Teste	95,43 %

The results obtained from *the accuracy* of the model can be seen in table 3. The difference obtained with the training and test data is expected and, as the difference between them is small, there is no need for an *initial concern with overfitting,* since the model behaved well in the presence of new data. Then, a good model was obtained and without the presence *of overfitting, being* able to classify fraudulent transactions with an *accuracy* of 95 %, which is an optimal result for a vital and sensitive operation in the current economic models.