POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Credit card fraud detection

NUMERICAL ANALYSIS FOR MACHINE LEARNING

Computer Science and Engineering (T2I - ARTIFICIAL INTELLIGENCE)

Davide GESUALDI

Student Number: 101761

Personal Code: 10885255

Prof. Edie MIGLIO

OBJECTIVE

Credit card fraud detection has become a critical challenge in modern financial transactions.

Machine learning algorithms have revolutionized the field of fraud detection, by learning from historical transaction data and identifying the anomalies.

The problem of credit card fraud detection is particularly complex from a machine learning perspective due to:

- Imbalanced data: legitimate transactions vastly outnumbering fraudulent ones.
- Concept drift: the concept of fraud evolves over time as consumer habits and fraudulent tactics undergo changes.

The proposed HybridIG-CSO model utilizes an automatic feature selection mechanism that identifies significant features through Information Gain (IG) and Competitive Swarm Optimization (CSO) techniques, with a Random Weight Network (RWN) serving as the foundational classifier.

Table of contents

The models were evaluated using four distinct datasets:

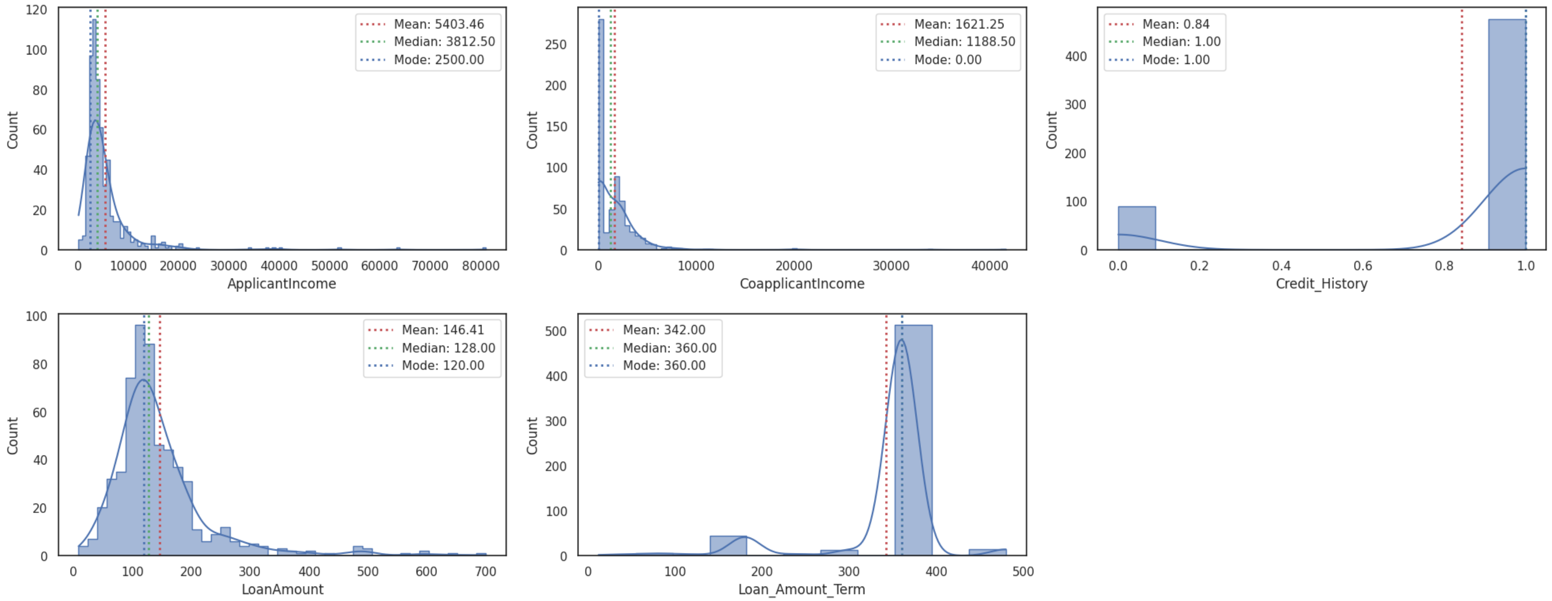
Dataset	Abb.	#Samples	#Features	#PS	Link
Loan Prediction	D_1	614	11	192 (31%)	https://github.com/Paliking/ML_examples/blob/master/LoanPrediction/train_u6lujuX_CVtuZ9i.csv
Creditcardcsvpresent	D_2	3075	10	448 (14%)	https://github.com/gksj7/creditcardcsvpresent/blob/main/creditcardcsvpresent.csv
Default ofCreditCardClients	D_3	30000	23	6636 (22%)	https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients
European cardholders	D_4	284807	30	482 (0.17%)	https://kaggle.com/mlg-ulb/creditcardfraud

"Abb." denotes the assigned dataset code and "#PS" signifies the positive samples in each dataset.

SKEWNESS

Quantifies the asymmetry of the distribution.

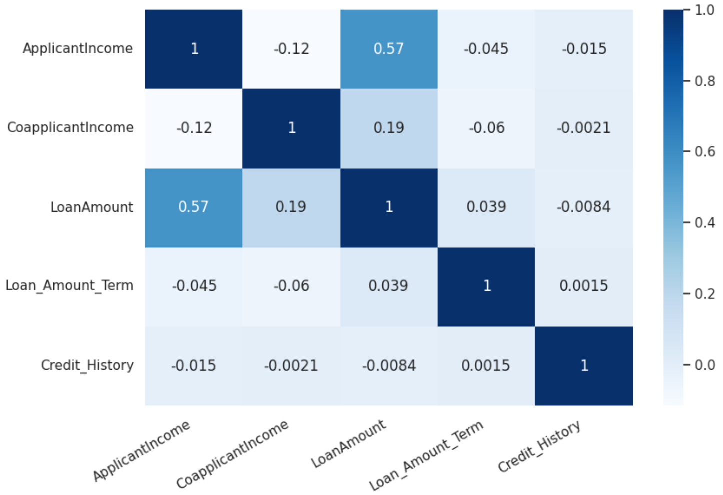
- Positive Skewness: Mean > Median > Mode
- Negative Skewness: Mean < Median < Mode
- Zero Skewness: symmetrical distribution



CORRELATION ANALYSIS

Computation of pairwise attribute correlations using the Pearson correlation coefficient.

The attributes are mainly weakly linear correlated, either positively or negatively.



DATASET SPLITTING

Dataset shuffled and split into train – valid- test sets, with proportions of 70% - 15% - 15%.

Performance assessed both on a separate test set and using a 10-fold cross-validation.

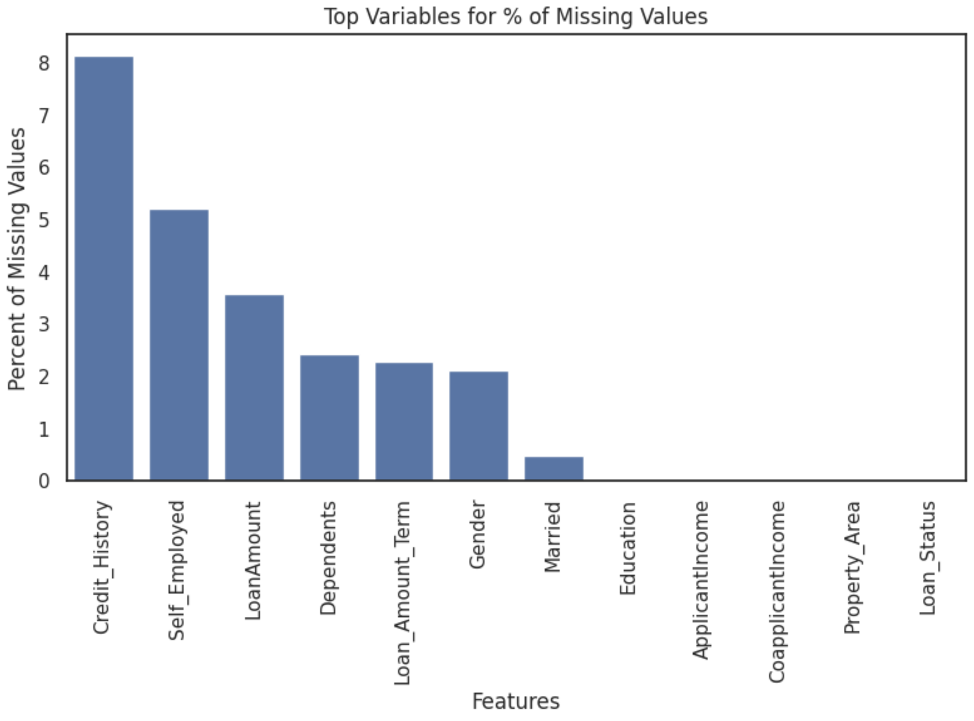
HANDLING MISSING VALUES

Exclusively present in the first dataset.

Categorical features underwent label encoding prior to handling missing data.

K-Nearest Neighbors (KNN) imputation approach was employed:

- KNN Mean Imputation for numerical features.
- KNN Majority Vote for categorical features
- Removal of samples with missing values in multiple categorical features simultaneously (only 13 samples ~2%).

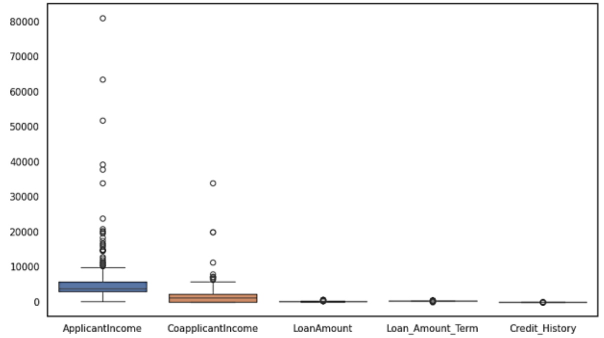


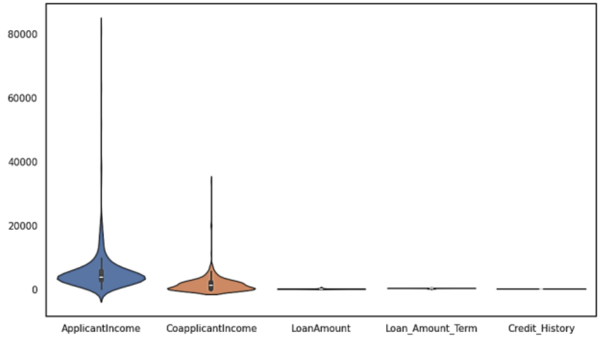
IMBALANCED DATA

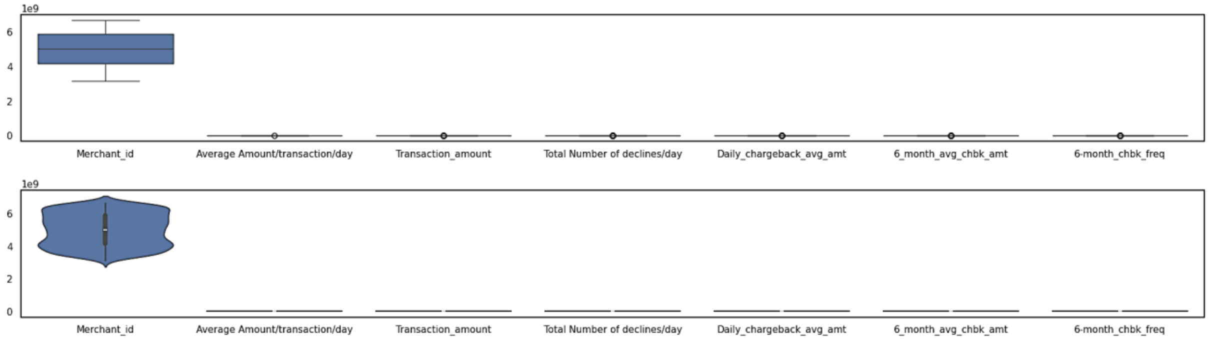
Synthetic Minority Over-sampling Technique (SMOTE), which generates synthetic instances in the neighborhood of existing minority class samples.

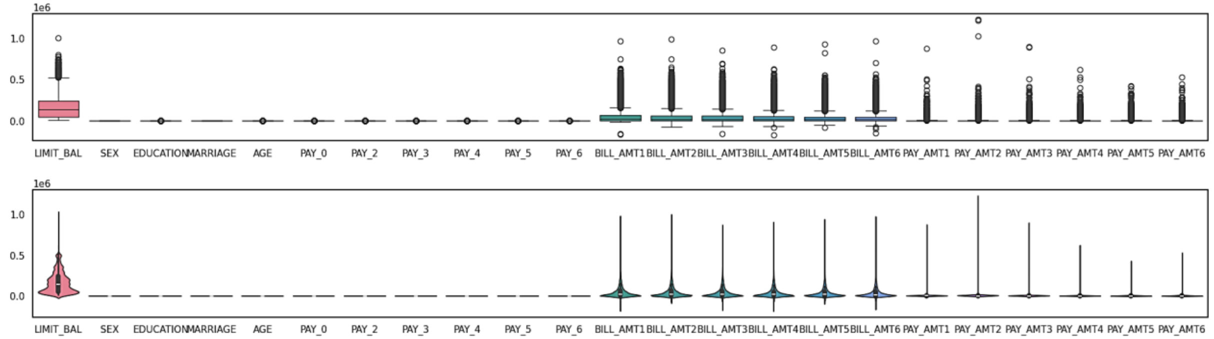
DATA NORMALIZATION

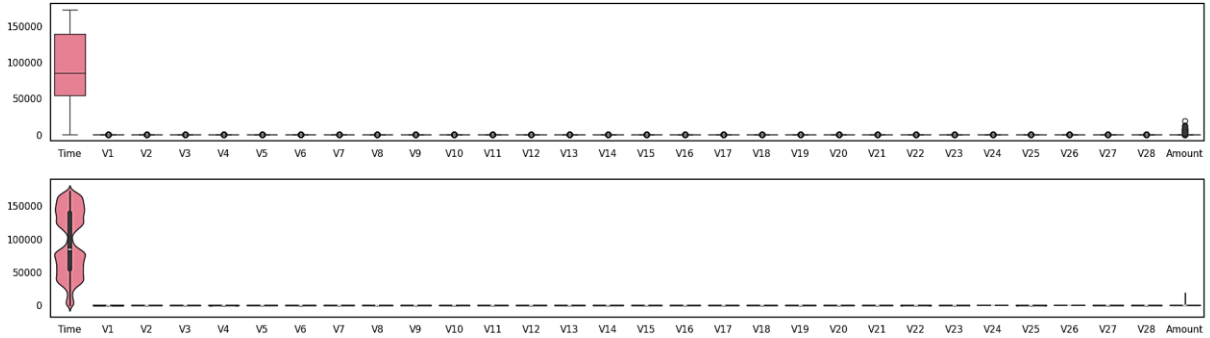
Box and violin plots revealed the need for data normalization, due to the massive presence of outliers and the disparate scales across variables.







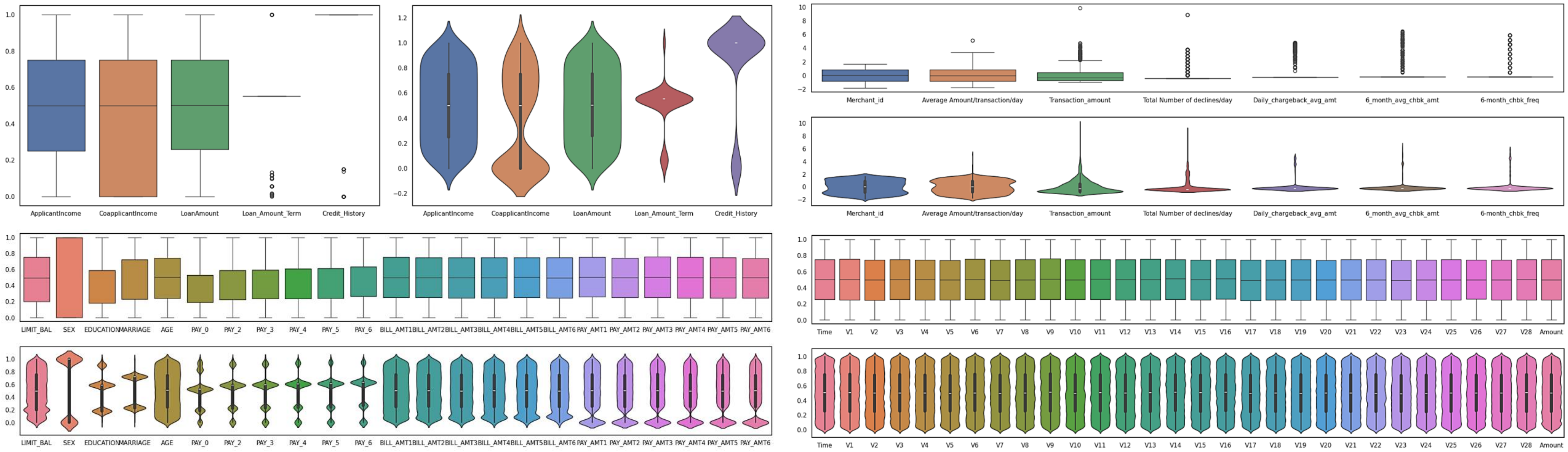




QUANTILE TRANSFORMER

Mitigates the impact of outliers by using quantiles information.

Applies a non-linear transformation such that the probability density function of each feature will be mapped to a uniform distribution within the range [0, 1], making outliers indistinguishable from inliers.



INFORMATION GAIN

FILTER APPROACH: Information Gain (IG) technique measures the reduction in uncertainty, or entropy, about a target variable when a specific feature is known.

IG for a feature X with respect to a target variable Y can be calculated as:

$$IG(Y|X) = H(Y) - H(Y|X)$$

where H(Y) represents the entropy of the target variable Y before considering feature X, and H(Y|X) represents the conditional entropy of Y given the values of feature X.

Distribution entropy $$H(Y) = -\sum_{y \in Y} P(y) log_2(P(y))$$

Attribute entropy $$H(Y|X) = -\sum_{x \in X} P(x) \sum_{y \in Y} P(y|x) log_2(P(y|x))$$

where P(x) is the proportion of instances with value x for feature X.

COMPETITIVE SWARM OPTIMIZATION

WRAPPER APPROACH: CSO is an algorithm rooted in the original PSO technique, devised to tackle the issue of premature convergence that often arises when applying PSO to complex search spaces containing numerous local optima.

CSO relies on pairwise competition between randomly selected particles (potential solutions) within the swarm (population).

Each particle can be considered as a point represented by a position X and a velocity V.

During each iteration:

- The swarm is divided into two equal and randomly selected groups.

- CSO selects two particles, one from each group, and initiates a competition between them.

- The winning particle is directly transferred to the next generation.

- The losing particle is updated based on the information derived from the winner and then included in the next generation.

$$V_{li}(t+1) = R_1(i,t)V_{li}(t) + R_2(i,t)(X_{wi}(t) - X_{li}(t)) + \varphi R_3(i,t)(\overline{X}_i(t) - X_{li}(t))$$

$$X_{li}(t+1) = X_{li}(t) + V_{li}(t+1)$$

RANDOM WEIGHT NETWORK

The fundamental architecture of the RWN architecture follows a fully connected architecture with a single hidden layer.

Unlike conventional gradient descent methods that necessitate the adjustment of multiple parameters, RWN simplifies the training process by focusing solely on the number of hidden neurons.

Algorithm 1: Pseudo-code of RWN

Input: *Training dataset* $N = \{ (x_j, t_j) \mid x_j \epsilon R^n (1 \leq j \leq N)\}$;
Activation function g();
Number of hidden neurons L;
Output: *Output weights* β;
for (i= 1 to L) do
Initialize weights w_i *and biases* b_i *randomly;*
Calculate the hidden layer output matrix H;
Return output weights β

$$H = \begin{Bmatrix} g(w_1 \cdot x_1 + b_1) & \cdots & g(w_L \cdot x_1 + b_L) \\ \vdots & \cdots & \vdots \\ g(w_1 \cdot x_N + b_1) & \cdots & g(w_L \cdot x_N + b_L) \end{Bmatrix}_{N \times L} \qquad \beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m} \qquad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m}$$

$$H\beta = T$$

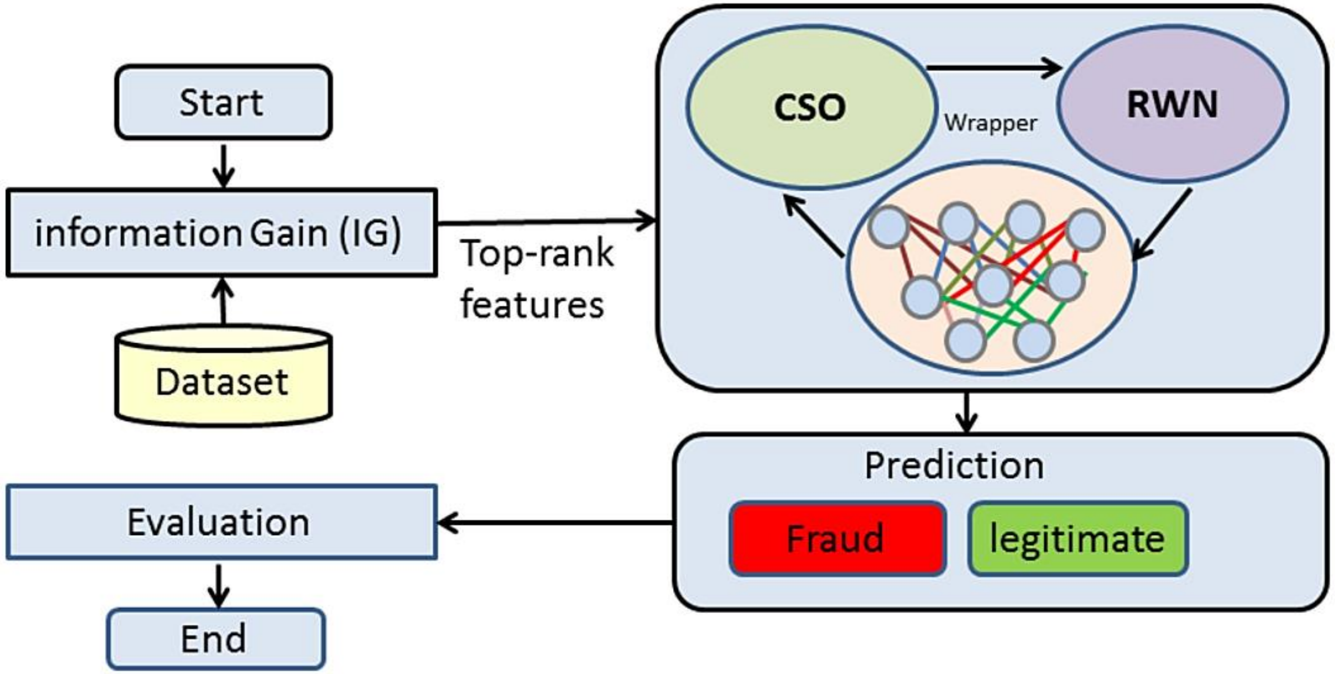
The output weights are determined through the application of the Moore-Penrose (MP) generalized inverse.

HYBRID IG-CSO

The initial phase employs IG as a filter-based method to rank the features within the credit card dataset.

Only the top-ranked features are retained and subsequently passed to the wrapper-based algorithm, CSO.

The RWN algorithm serves as the learning model within this hybrid framework.



The CSO particle is encoded as a real vector encompassing the subsequent components:

- A set of binary flags indicating the inclusion or exclusion of corresponding features.
- A set of binary flags dictating the number of neurons in the hidden layer of the RWN.
- The RWN parameters, which encapsulate the values of input weights and hidden biases.

The CSO algorithm's concepts of position and velocity were implemented by defining three positions and velocities for each particle: one for feature subset selection and two for the RWN configuration (one for weights and one for biases).

The fitness of each particle, which is used to determine the winner in the competition, is calculated as follows:

$$Fit = \alpha CLErr + \beta \frac{ft}{FT} + \gamma \frac{hd}{HD}$$

where:

- CLErr: error rate in classifying the RWN network
- ft: number of features selected
- FT: overall count of features in the dataset
- hd: count of hidden neurons set by the optimizer
- HD: maximum allowable number of neurons in the RWN

The parameters α, β and γ manage the impact of weights.

PERFORMANCE METRICS

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$G - mean = \sqrt{Sensitivity \times Specificity}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Finally, the AUC (Area Under the Curve), assessing the models' differentiation capability via the ROC curve.

Before presenting the obtained results, it is important to acknowledge that accuracy, while commonly used metrics in classification tasks, is not a reliable index of the quality of the trained model in case of imbalanced datasets or when the importance of wrongly predicting positive-class samples is different from wrongly predicting negative-class samples.

It can give a misleading picture of the model’s performance, often favoring the majority class.

More suitable metrics in such scenarios are the AUC, F1 score and G-Mean, which provide a more balanced assessment by considering both precision and recall or by emphasizing the balance between sensitivity and specificity, thereby offering a more comprehensive evaluation of the model's performance on imbalanced data.

The performance achieved in these more robust metrics closely aligns with the results reported in the referenced paper, except for the Classic-RWN model in the first dataset in Experiment I, which exhibited notably worse performance.

EXPERIMENT I

Comparisons performance between HybridIG-CSO, RWN with filter approach, and RWN with CSO

separate test set

Dataset	Algorithm	Accuracy	Precision	Recall	AUC	F1	G-Mean
D_1	Classic-RWN	0.5393	0.5094	0.6429	0.5464	0.5684	0.5451
	IG-RWN	0.7528	0.9623	0.7183	0.7034	0.8226	0.6540
	CSO-RWN	0.6854	0.8302	0.6984	0.6512	0.7586	0.6261
	HybridIG-CSO	**0.7528**	**0.9623**	**0.7183**	**0.7034**	**0.8226**	**0.6540**
D_2	Classic-RWN	0.9567	0.8765	0.8765	0.9251	0.8765	0.9239
	IG-RWN	0.9567	0.9383	0.8352	0.9495	0.8837	0.9494
	CSO-RWN	**0.9827**	0.9259	**0.9740**	**0.9603**	**0.9494**	**0.9597**
	HybridIG-CSO	0.9610	**0.9506**	0.8462	0.9569	0.8953	0.9569
D_3	IG-RWN	**0.7831**	0.5494	**0.5126**	0.6998	0.5303	0.6834
	HybridIG-CSO	0.7793	**0.5753**	0.5044	**0.7066**	**0.5375**	**0.6943**
D_4	IG-RWN	**0.9784**	**0.8941**	**0.7525**	**0.9387**	**0.8172**	**0.9376**
	HybridIG-CSO	0.9701	0.8824	0.6696	0.9288	0.7614	0.9276

10-fold cross-validation

Dataset	Algorithm	Accuracy	Precision	Recall	AUC	F1	G-Mean
D_1	Classic-RWN	0.4904	0.4984	0.4933	0.4912	0.4915	0.4857
	IG-RWN	0.7332	**0.9540**	0.6633	0.7350	0.7809	0.7004
	CSO-RWN	**0.7790**	0.8398	**0.7513**	**0.7792**	**0.7913**	**0.7754**
	HybridIG-CSO	0.7018	0.8960	0.6450	0.7023	0.7474	0.6713
D_2	Classic-RWN	0.9720	0.9722	0.9721	0.9722	0.9721	0.9722
	IG-RWN	0.9730	0.9781	0.9683	0.9731	0.9732	0.9730
	CSO-RWN	**0.9863**	**0.9892**	**0.9837**	**0.9865**	**0.9864**	**0.9865**
	HybridIG-CSO	0.9707	0.9755	0.9665	0.9708	0.9709	0.9707
D_3	IG-RWN	0.6980	**0.5582**	**0.7749**	**0.6980**	**0.6490**	**0.6839**
	HybridIG-CSO	**0.6897**	0.5490	0.7640	0.6897	0.6389	0.6752
D_4	IG-RWN	**0.9651**	**0.9502**	**0.9794**	**0.9651**	**0.9645**	**0.9649**
	HybridIG-CSO	0.9212	0.9253	0.9339	0.9211	0.9268	0.9170

Due to the limitations of the free usage plans on Colab and Kaggle, it was not feasible to implement the classical RWN and the manually tuned CSO-RWN for the last two datasets.

HybridIG-CSO model outperforms the other models on a separate test set, except for the second and fourth datasets.

Using 10-fold cross-validation, the CSO-RWN outperforms the HybridIG-CSO across all metrics. In contrast, for the last two datasets, IG-RWN emerges as the best-performing model among the two tested.

EXPERIMENT II

Comparison with other classifiers.

separate test set

Dataset	Classifier	Accuracy	Precision	Recall	AUC	F1	G-Mean
D_1	NB	0.7528	0.9623	0.7183	0.7034	0.8226	0.6540
	RF	0.7528	0.9057	**0.7385**	**0.7167**	0.8136	**0.6914**
	SVM	0.7528	0.9623	0.7183	0.7034	0.8226	0.6540
	HybridIG-CSO	**0.7528**	**0.9623**	0.7183	0.7034	**0.8226**	0.6540
D_2	NB	0.9113	0.7284	0.7564	0.8393	0.7421	0.8319
	RF	**0.9762**	0.9753	**0.8977**	**0.9758**	**0.9349**	**0.9758**
	SVM	0.9524	**0.9877**	0.7921	0.9663	0.8791	0.9660
	HybridIG-CSO	0.9610	0.9506	0.8462	0.9569	0.8953	0.9569
D_3	NB	0.6769	**0.6630**	0.3734	0.6719	0.4777	0.6719
	RF	0.8047	0.4576	**0.5781**	0.6809	0.5109	0.6433
	SVM	0.7556	0.6022	0.4628	0.7009	0.5234	0.6939
	HybridIG-CSO	**0.7793**	0.5753	0.5044	**0.7066**	**0.5375**	**0.6943**
D_4	NB	0.9886	0.8353	**0.9467**	0.9163	0.8875	0.9127
	RF	**0.9886**	0.8706	0.9136	0.9329	0.8916	0.9309
	SVM	0.9746	**0.8941**	0.7103	**0.9366**	0.7917	**0.9357**
	HybridIG-CSO	0.9701	0.8824	0.6696	0.9288	0.7614	0.9276

10-fold cross-validation

Dataset	Classifier	Accuracy	Precision	Recall	AUC	F1	G-Mean
D_1	NB	0.7248	0.9492	0.6566	0.7267	0.7746	0.6907
	RF	**0.8044**	0.8334	**0.7908**	**0.8050**	**0.8106**	**0.8040**
	SVM	0.7248	**0.9590**	0.6550	0.7269	0.7765	0.6871
	HybridIG-CSO	0.7018	0.8960	0.6450	0.7023	0.7474	0.6713
D_2	NB	0.8413	0.7125	0.9588	0.8411	0.8173	0.8311
	RF	**0.9897**	**0.9955**	**0.9841**	**0.9898**	**0.9898**	**0.9898**
	SVM	0.9673	0.9724	0.9628	0.9673	0.9675	0.9672
	HybridIG-CSO	0.9707	0.9755	0.9665	0.9708	0.9709	0.9707
D_3	NB	0.6724	0.6657	0.6748	0.6724	0.6702	0.6723
	RF	**0.8768**	**0.8525**	**0.8960**	**0.8768**	**0.8737**	**0.8764**
	SVM	0.7017	0.5990	0.7538	0.7017	0.6675	0.6941
	HybridIG-CSO	0.6876	0.5455	0.7621	0.6876	0.6358	0.6727
D_4	NB	0.9160	0.8358	0.9955	0.9160	0.9086	0.9124
	RF	**0.9938**	**0.9906**	**0.9969**	**0.9938**	**0.9937**	**0.9937**
	SVM	0.9619	0.9458	0.9773	0.9619	0.9613	0.9618
	HybridIG-CSO	0.9212	0.9253	0.9339	0.9211	0.9268	0.9170

HybridIG-CSO outperforms the other classifiers in almost all metrics on the first and third datasets on a separate test set.

In all other cases, including using 10-fold cross-validation, Random Forest emerges as the best classifier.

The hyperparameters α, β, γ and ϕ were optimized through a 10-fold cross-validation approach, by determining the combination that yields the highest average accuracy score over the 10 folds.

Obviously, the performance of HybridIG-CSO could be further optimized by expanding the search space for hyperparameters or optimizing with respect to other metrics such as F1, G-mean or AUC, increasing the number of iterations, testing different thresholds for filter-based feature selection using IG, or testing a higher prediction threshold for the sigmoid function.

Ensemble learning with HybridIG-CSO models, using different configurations, could further enhance classification performance.

However, the most noteworthy aspect of this project is the efficiency of the method proposed in the paper. With a significantly simplified training process compared to conventional gradient descent methods, it achieves performance, on a complex machine learning task as credit card fraud detection, that is almost comparable to an ensemble learning method like Random Forest, and in some cases, on a separate test set, even surpasses it.

Particularly remarkable is the use of the hybrid approach for feature selection, combining IG and CSO, which enabled the model to be tested on large datasets, overcoming resource limitations, and outperforming the other models on a smaller dataset like the first one.