A Fourier Spectral Pattern Analysis to Design Credit Scoring Models

Roberto Saia and Salvatore Carta
Dipartimento di Matematica e Informatica - Università di Cagliari
Via Ospedale 72, 09124 Cagliari - Italy
{roberto.saia,salvatore}@unica.it

ABSTRACT

The increase of consumer credit has made it necessary to research more and more effective models for the credit scoring. Such models are usually trained by using the past loan applications, evaluating the new ones on the basis of certain criteria. Although the state of the art offers several different approaches for their definition, this process represents a hard challenge due to several reasons. The most important ones are the data unbalance between the default and the non-default cases that reduces the effectiveness of almost all techniques, and the data heterogeneity, which makes it difficult the definition of a model able to effectively evaluate all the new loan applications. The approach proposed in this paper faces the aforementioned problems by moving the evaluation process from the canonical time domain to a frequency one, using a model based on the past non-default loan applications. It allows us to overcome the data unbalance problem by exploiting only a class of data, also defining a model that is less influenced by the data heterogeneity. The performed experiments show interesting results, since the proposed approach achieves performance closer or better than that of one of the best state-of-the-art approaches of credit scoring, such as random forests, although it operates in a proactive way, only by exploiting the past non-default cases.

CCS CONCEPTS

•Information systems → Data stream mining; Clustering and classification; Business intelligence; •Theory of computation → Pattern matching; •General and reference → *Metrics;*

KEYWORDS

Business intelligence, credit scoring, imbalanced datasets, classification, metrics

ACM Reference format:
Roberto Saia and Salvatore Carta. 2017. A Fourier Spectral Pattern Analysis to Design Credit Scoring Models. In *Proceedings of International Conference on Internet of Things and Machine Learning, Liverpool city, United Kingdom, October 17-18, 2017 (IML'17),* 11 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

IML'17, Liverpool city, United Kingdom

DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

A *credit scoring* process is aimed to evaluate, in terms of reliability, a new loan application (from now on simply named as *instance*), and from its result depends the acceptation or non acceptation of it. It is clear how the effectiveness of such models is strongly related to the gains and losses of the involved financial operators [24], especially in these last years characterized by an increasing use of the consumer credit, which is obviously correlated with that of the defaulted cases (i.e., loans that have been fully or partially not repaid). An ideal approach of *credit scoring* should be able to correctly classify the new instances into two classes, *accepted* or *rejected*), on the basis of the information given by the past instances.

In other words, the *credit scoring* techniques can be considered a set of statistical tools able to calculate the probability that an instance leads toward a default case [26, 35], allowing the financial operators to evaluate the *credit risk* [20] and to monitor the credit activities [8].

The definition of effective *credit scoring* models represents a hard challenge due to several problems, the most important of which is the imbalance in the data used during the model training process [4]. It means that these data sources are composed by a small number of default cases and a big number of non-default ones, an unbalanced configuration that reduces the effectiveness of almost all the machine learning approaches [28].

The idea behind this paper is to *move the process of evaluation of the new instances from the canonical time domain to the frequency one*, performing the spectral analysis through the *Fourier transformation* [19]. It is performed by using the *Fast Fourier Transform* (FFT) algorithm, which allows us to move a *time series* (i.e. in our case, a sequence of discrete-time data represented by the feature values of an instance) from its original time domain to a frequency one, where we can study the data from a different point of view.

Considering that the model used in the process of evaluation of the new instances is defined only by using a class of data (the non-default cases), such approach offers a threefold advantage: it allows us to operate proactively, it faces the problems related to the *cold-start* issue (i.e., the scarcity or absence of default cases), and it reduces the issues related to the data heterogeneity, because their new representation in the frequency domain is less influenced by the data variation. We compare our approach to the *Random Forests* one, since in most cases reported in literature [5, 9, 32] it outperforms the other *credit scoring* approaches.

The main key contributions of this paper are listed below:

(i) definition of the *time series* to be used in the *Fast Fourier Transform* (FFT) process, made on the basis of the data that compose each instance in the considered datasets;
(ii) formalization of the Fourier spectral analysis comparison process, performed in terms of frequency magnitude difference

measured between the *time series* in the set of the past non-default instances and that in an unevaluated one;

(iii) definition of the *Dynamic Feature Selection* (*DFS*) process, aimed to assign a different weight to each frequency component of the previously extracted instance spectrum, on the basis of their relevance (in terms of entropy) in the evaluation process;

(iv) formalization of the *Fourier Spectrum Pattern* (*FSP*) algorithm able to classify the new instances as *accepted* or *rejected* by exploiting the previous spectral analysis, the *DFS* process, and the *tolerance range* ρ.

The remainder of the paper is organized as follows: Section 2 discusses the background and related work; Section 3 provides a formal notation, makes some premises, and defines the faced problem; Section 4 describes the implementation of the proposed approach; Section 5 provides details on the experimental environment, on the used datasets and metrics, as well as on the adopted strategy and the chosen competitor, reporting the experimental results at the end; some concluding remarks and future work are given in the last Section 6.

2 BACKGROUND AND RELATED WORK

Recent literature proposes numerous classification techniques able to operate in the *credit scoring* context [18], as well as a big number of studies aimed to evaluate their performance [32], also by taking into account the optimal configuration of the involved parameters [2] and the metrics to be used in order to evaluate the performance [22].

The two most important advantages derived from the adoption of *credit scoring* techniques [40] are the capability to infer when it is reasonable (in terms of potential risks) to grant a loan to someone, and the capability to define models able to infer the customer behavior, information that can be used to propose targeted financial services. In the context of this paper we take into account only the first one.

2.1 Credit Scoring Models

In order to perform a *credit scoring* process it is possible to exploit many state-of-the-art techniques usually used in the statistic and data mining fields [1, 10]. Some significant examples are the linear discriminant models [38], the logistic regression models [26], the neural network models [6, 16], the genetic programming models [11, 36], the k-nearest neighbor models [25], and the decision tree models [14, 46].

It should be observed that in many cases these techniques can be combined in order to define hybrid approaches of *credit scoring*. Some examples are the techniques that exploit the neural networks and the clustering methods, presented in [27], and the two-stage hybrid modeling procedure with artificial neural networks and multivariate adaptive regression splines, proposed in [31, 45].

2.2 Imbalanced Class Distribution

One of the most important problems that makes it difficult the definition of effective models for the *credit scoring* is the imbalanced class distribution of data [23, 28]. This issue is given by the fact that the data used in order to train the models are characterized by a small number of default cases and a big number of non-default ones, a distribution of data that limits the performance of the classification techniques [9, 28].

This problem leads toward misclassification costs, as reported in [44], which proposes to preprocess the training data through an *over-sampling* or *under-sampling* of the classes, as a possible solution.

The effect of such preprocessing activity on the performance has been studied in [12, 34].

2.3 Cold Start

The *cold start* issue [17, 47] arises when there is not enough information to train a reliable model about a domain. In the context of the *credit scoring*, such scenario appears when the data used to train the model are not representative of all classes of data [3, 43] (i.e., default and non-default cases).

This kind of issue affects many areas, e.g., those related to the recommender systems [21, 33, 42], since they are usually based on models defined on the basis of the previous choices of the users (user profiles), similarly to the *credit scoring* context, where the past loan applications are taken into account.

In the approach proposed in this paper the *cold start* issue can be reduced/overcome by using only a class of data during the model definition process (i.e., only the non-default cases in the training dataset).

2.4 Random Forests

Since its formalization [7], *Random Forests* represents one of the most common techniques for data analysis, thanks to its better performance compared to the other state-of-the-art techniques.

This technique represents an ensemble learning method for classification and regression that is based on the construction of a number of randomized decision trees during the training phase and it infers conclusions by averaging the results.

It is able to face a wide range of prediction problems, without performing any complex configuration, since it only requires the tuning of two parameters: the number of trees and the number of attributes used to grow each tree.

2.5 Fourier Transform and Spectral Analysis

The basic idea behind the approach proposed in this paper is to move the process of evaluation of the new instances (*time series*) from their canonical time domain to the frequency one, in order to obtain a representative pattern composed by their frequency components, as shown in Figure 1.

This operation is performed by recurring to the *Discrete Fourier Transform* (*DFT*), whose formalization is shown in Equation 1, where i is the imaginary unit.

$$F_n \stackrel{\text{def}}{=} \sum_{k=0}^{N-1} f_k \cdot e^{-2\pi i n k/N}, \quad n \in \mathbb{Z} \tag{1}$$

The result of the Equation 1 is a set of sinusoidal functions, each corresponding to a particular frequency component (i.e., the *spectrum*[1]).

[1]The *spectrum* of the *frequency components* is the frequency domain representation of a signal.

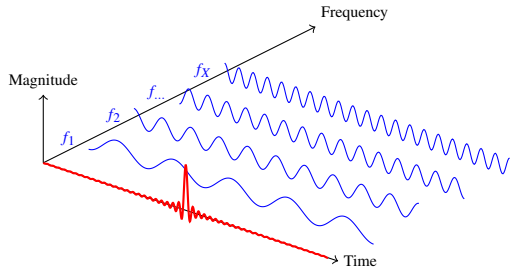

Figure 1: *Time and Frequency Domains*

If it is necessary, we can use the *inverse Fourier transform* shown in Equation 2 to return to the original time domain.

$$f_k = \frac{1}{N}\sum_{n=0}^{N-1} F_n \cdot e^{2\pi i k n/N}, \quad n \in \mathbb{Z} \tag{2}$$

The *Fast Fourier Transform* (FFT) algorithm, used in the context of this paper to perform the Fourier transformations, rapidly computes the DFT, or its *Inverse Fast Fourier Transform* ($IDFT$), by factorizing the input matrix into a product of sparse (mostly zero) factors. It is largely used because it reduces the computational complexity of the process from $O(n^2)$ to $O(n \log n)$, where n denotes the data size.

3 PRELIMINARIES

Formal notation, premises, and problem statement related to this paper are stated in the following:

3.1 Notation

Given a set of classified instances $I = \{i_1, i_2, \ldots, i_N\}$, and a set of features $V = \{v_1, v_2, \ldots, v_M\}$ that compose each $i \in I$, we denote as $I_+ \subseteq I$ the subset of non-default instances, and as $I_- \subseteq I$ the subset of default ones.

We also denote as $\hat{I} = \{\hat{i}_1, \hat{i}_2, \ldots, \hat{i}_U\}$ a set of unclassified instances and as $O = \{o_1, o_2, \ldots, o_U\}$ these instances after the classification process, thus $|\hat{I}| = |O|$.

It should be observed that an instance can belong only to one class $c \in C$, where $C = \{accepted, rejected\}$.

Finally, we denote as $F = \{f_1, f_2, \ldots, f_X\}$ the frequency components of each instance (spectrum), obtained at the end of the DFT process.

3.2 Premises

Considering that the periodic waves are characterized by a frequency f and a wavelength λ (i.e., the distance in the medium between the beginning and end of a cycle $\lambda = \frac{w}{f_0}$, where w stands for the wave velocity), which are defined by the repeating pattern, the non-periodic waves taken into account in the *Discrete Fourier Transform* process do not have a frequency or wavelength. Their fundamental period T is the period where the wave values were taken and sr denotes their number over this time (i.e., the acquisition frequency).

Assuming that the time interval between the acquisitions is equal, on the basis of the previous definitions applied in the context of this paper, the considered non-periodic wave is given by the sequence of values $v_1, v_2, \ldots, v_M$ with $v \in V$, which compose each instance $i \in I_+$ (i.e., the past non-default instances) and $\hat{i} \in \hat{I}$ (i.e., the unevaluated instances), and that representing the *time series* taken into account.

Their fundamental period T starts with v_1 and it ends with v_M, thus we have that $sr = |V|$; the sample interval si is instead given by the fundamental period T divided by the number of acquisition, i.e., $si = \frac{T}{|V|}$.

Through the FFT algorithm we compute the *Discrete Fourier Transform* of each *time series* $i \in I_+$ and $\hat{i} \in \hat{I}$, by converting their representation from the time domain to the frequency one. The obtained frequency-domain representation provides information about the signal's magnitude and phase at each frequency. For this reason, the output (denoted as x) of the FFT computation is a series of complex numbers composed by a real part x_r and an imaginary part x_i, thus $x = (x_r + ix_i)$.

We can obtain the x magnitude by using $|x| = \sqrt{(x_r^2 + x_i^2)}$ and the x phase by using $\varphi(x) = \arctan\left(\frac{x_i}{x_r}\right)$, although in the context of this paper we will take into account only the magnitude at each frequency.

3.3 Problem Statement

On the basis of the comparison of the spectral analysis λ performed by the FFT algorithm on the *time series* in $i \in I_+$ and $\hat{i} \in \hat{I}$, our FSP approach classifies each instance $\hat{i} \in \hat{I}$ as *accepted* or *rejected*.

Given a function $eval(\hat{i}, \lambda)$ created to evaluate the correctness of the $\hat{i}$ classification, which returns a boolean value σ (*0=misclassification*, *1=correct classification*), we formalize our objective as the maximization of the results sum, as shown in Equation 3.

$$\max_{0 \le \sigma \le |\hat{I}|} \sigma = \sum_{u=1}^{|\hat{I}|} eval(\hat{i}_u, \lambda) \tag{3}$$

4 PROPOSED APPROACH

The implementation of our approach is carried out through the following steps:

(1) **Time Series Definition** (TSD): definition of the *time series* to use in the FFT algorithm, in terms of sequence of instance feature values;
(2) **Time Series Analysis** (TSA): comparison of the Fourier spectral patterns of two instances, performed by processing their *time series*, defined in the previous step, through the FFT algorithm;
(3) **Time Series Dynamic Feature Selection** (DFS): determination of the weight of each frequency component in the instance spectrum, on the basis of the Shannon entropy metric;
(4) **Time Series Classification** (TSC): formalization of the FSP algorithm able to classify a new instance as *accepted* or *rejected*, on the basis of the TSA comparison, the DFS process, and the *tolerance range* ρ.

In the following, we provide a detailed description of each of these steps, since we have introduced the high-level architecture of the proposed FSP approach.

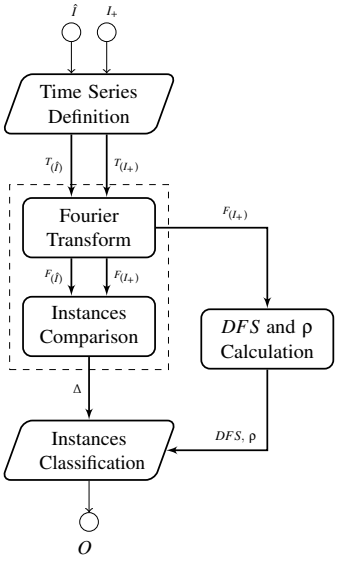

Figure 2: *FSP Architecture*

The high-level description shown in Figure 2 wants shortly introduces the processes involved in our approach, which are however explained in detail in the following.

According to the notation given in Section 3.1, I_+ and $\hat{I}$ denote, respectively, the set of non-default instances and the set of instances to evaluate, while the set O denotes the instances in $\hat{I}$ after their classification. We indicates with $T_{(I_+)}$ and $T_{(\hat{I})}$ the *time series* related, respectively, with the instances in I_+ and $\hat{I}$, and with $F_{(I_+)}$ and $F_{(\hat{I})}$ the set of frequency components obtained by processing these *time series* through the FFT algorithm.

At the beginning, the time series related to the sets of the unevaluated instances and the previous non-default instances are extracted. They are used as input in the Fourier Transform process, obtaining as result the spectral pattern of each instance. The classification of the instances to evaluate is based on a comparison process performed between their spectral patterns and those of the previous non-default ones, taking into account the importance of each frequency component (in terms of entropy) and a tolerance range ρ experimentally defined.

4.1 Time Series Definition

In the first step of our approach we define the *time series* to use in the *Discrete Fourier Transform* process.

Formally, a *time series* represents a series of data points stored by following the time order and usually it is a sequence captured at successive equally spaced points in time, thus it can be considered a sequence of discrete-time data.

In the context of the proposed approach, the *time series* taken into account are defined by using the set of features V that compose each instance in the I_+ and $\hat{I}$ sets, as shown in Equation 4, by following the criterion reported in Equation 5.

$$I_+ = \begin{vmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,M} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ v_{N,1} & v_{N,2} & \cdots & v_{N,M} \end{vmatrix} \quad \hat{I} = \begin{vmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,M} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ v_{U,1} & v_{U,2} & \cdots & v_{U,M} \end{vmatrix} \tag{4}$$

$$\begin{matrix} (v_{1,1}, v_{1,2}, \ldots, v_{1,M}), (v_{2,1}, v_{2,2}, \ldots, v_{2,M}), \cdots, (v_{N,1}, v_{N,2}, \ldots, v_{N,M}) \\ (v_{1,1}, v_{1,2}, \ldots, v_{1,M}), (v_{2,1}, v_{2,2}, \ldots, v_{2,M}), \cdots, (v_{U,1}, v_{U,2}, \ldots, v_{U,M}) \end{matrix} \tag{5}$$

The *time series* related to an item $\hat{i} \in \hat{I}$ will be compared to the *time series* related to all the items $i \in I_+$, by following the criteria explained in the next steps.

4.2 Time Series Analysis

Before we describe the process of analysis based on the Fourier transformation, it is useful to observe the spectral pattern of an instance randomly taken from a dataset, with $|V| = 20$ (Figure 3), beside its canonical representation in the time domain.

The frequency domain representation allows us to perform a data (represented by the sequence of values assumed by the instance features, as described in Section 4.1) analysis in terms of peaks (magnitudes) of the spectral frequencies that compose it. This allows us to detect some patterns in the features, which are not discoverable in the time domain.

Comparing the two different domains, we can observe some interesting properties for the context taken into account in this paper. The most significant are the following:

- The *phase invariance property* shown in Figure 4 proves that also in case of translation[2] between instances, a specific pattern still exists in the frequency spectrum. More formally, it is one of the *phase properties* of the Fourier transform [41], i.e., a shift of a *time series* in the time domain leaves the magnitude unchanged in the frequency domain. This property allows us to detect a particular pattern in the user behavior, regardless to the involved instances that compose it. A concrete example is represented by the values in the features from *6* to *11*, from *12* to *17*, and from *18* to *23*, of the *DC* dataset (described in Table 2 of Section 5.2). They report a sequence of values that belong to three different types of information, related to the loan applicant (i.e., *past repayments*, *bill statement*, and *amount paid*), and by exploiting the spectrum pattern analysis we can detect a specific pattern (behavior), also when it shifts along the features that compose one of these subsets of values;
- Another interesting aspect of the frequency domain is given by the *amplitude correlation property* shown in Figure 5. It proves the existence of a direct correlation between the values assumed by the features in the temporal domain and the magnitudes of the spectral components in the frequency domain. More formally, it is the *homogeneity property* of the Fourier transform [41], i.e., when the amplitude is altered in one domain, it is altered by the same entity in the other domain[3]. This property assures us of the capability of

[2] In terms of signal it represents a change of phase, considering that a translation in time domain corresponds to a change in phase in the frequency domain.
[3] Scaling in one domain corresponds to scaling in the other domain

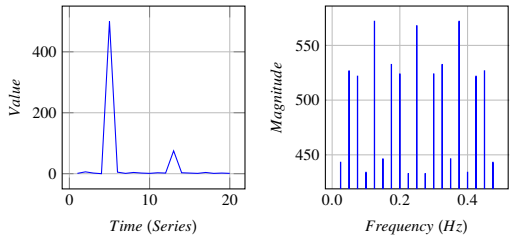

Figure 3: *Time and Frequency Domains*

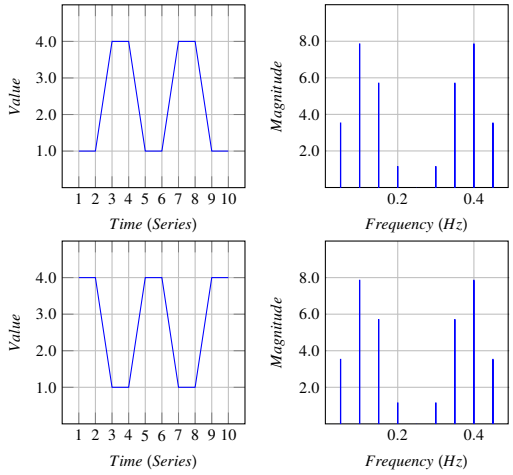

Figure 4: *Phase Invariance Property*

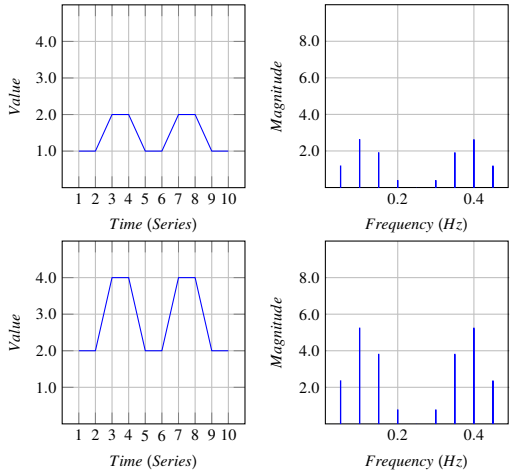

Figure 5: *Amplitude Correlation Property*

the frequency representation to differentiate the instances on the basis of the size of the values in their features.

Practically, the process of analysis is performed by moving the *time series* of the instances to compare from their time domain to the frequency one, by recurring to the FFT approach introduced in Section 2.5.

In this context, although there are many algorithms able to calculate the FFT, the most used are those based on the *Cooley-Tukey* recursive algorithm. It grants us a decimation in time on the basis of the following considerations: when the number N of input data is even, it is possible to express it as $N = 2 \cdot M$, allowing us to split the N element summation of the DFT formula into two M element

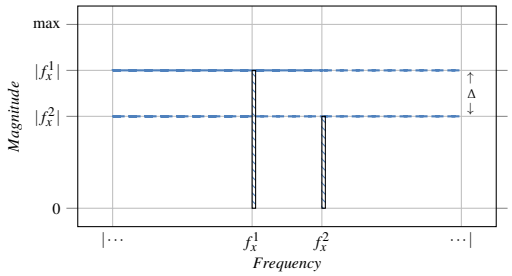

Figure 6: *Delta Difference*

ones, one over $n = 2 \cdot m$, another over $n = 2 \cdot (m+1)$, as shown in the Equation 6.

$$X_k = \sum_{m=0}^{M-1} x_{2m} e^{-2\pi i \frac{mk}{M}} + e^{-2\pi i \frac{k}{N}} \sum_{m=0}^{M-1} x_{2m} + 1 e^{-2\pi i \frac{mk}{M}} \tag{6}$$

We implement the FFT approach by using the *JTransforms* Java library, according to what reported in Section 5.1.

The process of comparison between an instance $\hat{i} \in \hat{I}$ to evaluate and a past non-default instance $i \in I_+$ is performed by measuring the difference Δ between the magnitude $|f|$ of each component $f \in F$ in the frequency spectrum of the involved instances.

It is shown in the Equation 7, where f_x^1 and f_x^2 denote, respectively, the same frequency component of an item $i \in I_+$ and an item $\hat{i} \in \hat{I}$. Such process of comparison between the same frequency component of two instances is also graphically shown in Figure 6.

$$\Delta = \left(|f_x^1| - |f_x^2| \right), \ \ with \ \ |f_x^1| \geq |f_x^2| \tag{7}$$

It should be noted that, as described in Section 4.4, for each instance $\hat{i} \in \hat{I}$ to evaluate, the aforementioned process is repeated by comparing it to each instance $i \in I_+$. This allows us to evaluate the variation Δ in the context of all the non-default past cases.

4.3 Time Series Dynamic Feature Selection

In the context of machine learning and statistics, the *feature selection* process is aimed to detect a subset of relevant features to use during the model definition. It represents an important preprocessing step, since it reduces the complexity of the final model, decreasing the training times, and increasing the generalization of the model. It also reduces the problem related with the *overfitting*, a problem that occurs when a statistical model describes random error or noise instead of the underlying relationship, and this frequently happens during the definition of excessively complex models, since many parameters, with respect to the number of training data, are involved.

In the context of our approach, we perform the *feature selection* task in a dynamic way, by measuring the Shannon entropy (i.e., a metric described in Section 5.3.1) in each feature of the training datasets (Figure 7).

Since the entropy gives us a measure of the uncertainty of a random variable, the larger it is, the less a-priori information one has on the value of it, then the entropy increases as the data becomes equally probable and decreases when their chances are unbalanced.

The adoption of different weights during the evaluation process, performed by the *Dynamic Feature Selection* (DFS) approach, is aimed to differentiate the instance features on the basis of their predictive power.

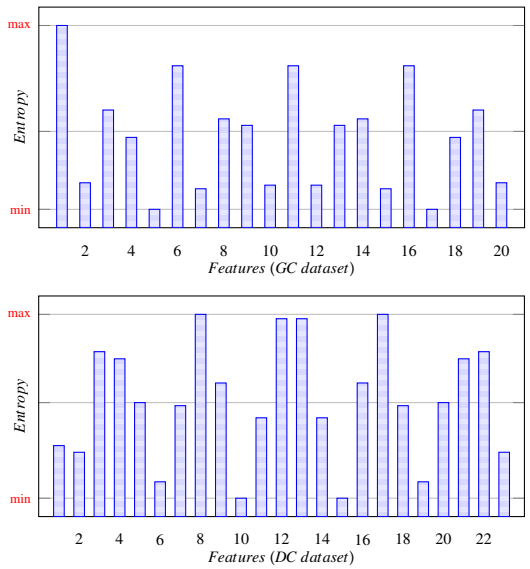

Figure 7: *Instance Features Entropy*

For the needs of the Algorithm 1, we formalize the *DFS* in terms of inverse Shannon entropy (in order to have a high value when the feature is important, and a low value otherwise). Such formalization is shown in Equation 8, where $P(f)$ indicates the probability that the frequency component f is present in the set F.

The obtained result represents the weight of the f frequency component (in terms of entropy) to use in the evaluation process described in the Section 4.4.

$$DFS(f) = 1 - \left(- \sum_{f \in F} P(f) log_2 [P(f)] \right) \quad (8)$$

4.4 Time Series Classification

This section formalizes the *FSP* algorithm used to perform the classification of new instances, together with the analysis of its asymptotic time complexity.

4.4.1 Algorithm. The proposed *FSP* approach is based on the Algorithm 1. It takes as input the set I_+ of non-default instances occurred in the past, the set $\hat{I}$ of unevaluated instances, and the *tolerance range* ρ (determined as described in Section 5.4.2). It returns as output a set O that contains all the instances in $\hat{I}$, classified as *accepted* or *rejected*.

From *step 2* to *step 23* we process all unevaluated instances $\hat{i} \in \hat{I}$, by starting with the extraction of the *time series* of each instance (*step 3*), which is processed at *step 4* in order to obtain the frequency spectrum. From the *step 5* to *step 15* we instead process each non-default instance $i \in I_+$, by performing the extraction of the *time series* of each instance (*step 6*) and by obtaining its frequency spectrum (*step 7*).

The *steps* from *8* to *14* verify if the difference between the magnitude of each frequency components $f \in F$ of the non-default instances and the correspondent component of the current instance, is within the ρ range.

On the basis of the result of this operation, in the *steps* from *9* to *13*, the weight (in terms of entropy) of the current frequency component is used in order to increase the *reliable* value (when the difference is within the ρ range) or the *unreliable* one (otherwise) (steps *10* and *12*).

On the basis of these two values the instance under evaluation is classified as *accepted* or *rejected* in the *steps* from *16* to *20*, and the result of the classification process is returned by the algorithm at the *step 24*, when all instances $\hat{i} \in \hat{I}$ have been processed.

Algorithm 1 *FSP Instances classification*

Input: I_+=Non-default instances, $\hat{I}$=Unevaluated instances, ρ=Tolerance range
Output: O=Set of classified instances

```
1: procedure InstancesClassification(I+,Î,ρ)
2:     for each î in Î do
3:         ts1 = getTimeseries(î)
4:         F1 = getFFT(ts1)
5:         for each i in I+ do
6:             ts2 = getTimeseries(i)
7:             F2 = getFFT(ts2)
8:             for each f in F do
9:                 if (|F2(f)| − |F1(f)| ∈ ρ) then
10:                    reliable += DFS(f)
11:                else
12:                    unreliable += DFS(f)
13:                end if
14:            end for
15:        end for
16:        if reliable > unreliable then
17:            O ← (î,accepted)
18:        else
19:            O ← (î,rejected)
20:        end if
21:        reliable = 0
22:        unreliable = 0
23:    end for
24:    return O
25: end procedure
```

4.4.2 Asymptotic Time Complexity. Although the evaluation of the time needed to perform the classification of a single instance is quite unnecessary, the possible implementation of the proposed *FSP* approach in a *real-time scoring system* [37], where the *response-time* represents a crucial factor, suggests us to analyze the theoretical complexity of the classification Algorithm 1.

Denoted as N the dimension of the training set I_+ (i.e., $N = |I_+|$), we define the asymptotic time complexity of the evaluation of a single instance (according to the *Big O notation*) by observing what follows:

(i) the Algorithm 1 presents three nested loops given by the outer loop that starts at *step 2*, which executes N times the other two inner loops (the first that starts at *step 5* and the second that starts at *step 8*), plus other operations (*getTimeseries*, *getFFT*, *comparisons*, and *assignations*), respectively with complexity of $O(n)$, $O(n \log n)$, $O(1)$, and $O(1)$;

(ii) the first inner loop executes one time the same aforementioned operations, plus its inner loop that executes operations with complexity $O(1)$ (*comparisons* and *assignations*) for a number of times lesser than N (i.e., for $|F|$ times);

On the basis of the previous considerations, we can conclude that the asymptotic time complexity of the algorithm is $O(N^2)$.

It should also be noted that the computational time can be adequately reduced by distributing the process over different machines, by employing large scale distributed computing models like *MapReduce* [15].

5 EXPERIMENTS

This section reports information about the experimental environment, the used datasets and metrics, the adopted strategy, the chosen competitor, as well as the results of the performed experiments.

5.1 Environment

The proposed *FSP* approach was developed in Java, where we use the *JTransforms*[4] library to operate the Fourier transformations.

The state-of-the-art approach used to evaluate its performance was made in *R*[5], using the *randomForest* and *ROCR* packages, as detailed in Section 5.5.

The experiments have been conducted by using two real-world datasets, both characterized by a strong unbalanced distribution of data.

It should be further added that we verified the existence of a statistical difference between the results, by using the independent-samples *two-tailed Student's t-tests* ($p < 0.05$).

5.2 Datasets

The two real-world datasets used in the experiments (i.e., *German Credit* and *Default of Credit Card Clients* datasets, both available at the *UCI Repository of Machine Learning Databases*[6]) represent two benchmarks in this research field. In the following we provide a brief description of their characteristics:

5.2.1 German Credit (GC). It contains *1,000* instances: *700* of them are non-default instances (70.00%) and *300* are default instances (30.00%). Each instance is composed by 20 features (whose type is described in Table 1) and a binary class variable (*accepted* or *rejected*).

Table 1: *Dataset GC Fields*

Feature	Description	Feature	Description
01	*Status of checking account*	*11*	*Present residence since*
02	*Duration*	*12*	*Property*
03	*Credit history*	*13*	*Age*
04	*Purpose*	*14*	*Other installment plans*
05	*Credit amount*	*15*	*Housing*
06	*Savings account/bonds*	*16*	*Existing credits*
07	*Present employment since*	*17*	*Job*
08	*Installment rate*	*18*	*Maintained people*
09	*Personal status and sex*	*19*	*Telephone*
10	*Other debtors/guarantors*	*20*	*Foreign worker*

5.2.2 Default of Credit Card Clients (DC). It contains *30,000* instances: *23,364* of them are non-default instances (77.88%) and *6,636* are default instances (22.12%). Each instance is composed by 23 features (whose type is described in Table 2) and a binary class variable (*accepted* or *rejected*).

5.3 Metrics

This section introduces the metrics used in the context of this paper.

[4]https://sites.google.com/site/piotrwendykier/software/jtransforms
[5]https://www.r-project.org/
[6]ftp://ftp.ics.uci.edu/pub/machine-learning-databases/statlog/

Table 2: *Dataset DC Fields*

Feature	Description	Feature	Description
01	*Credit amount*	*13*	*Bill statement in Aug-2005*
02	*Gender*	*14*	*Bill statement in Jul-2005*
03	*Education*	*15*	*Bill statement in Jun-2005*
04	*Marital status*	*16*	*Bill statement in May-2005*
05	*Age*	*17*	*Bill statement in Apr-2005*
06	*Past repayments in Sep-2005*	*18*	*Amount paid in Sep-2005*
07	*Past repayments in Aug-2005*	*19*	*Amount paid in Aug-2005*
08	*Past repayments in Jul-2005*	*20*	*Amount paid in Jul-2005*
09	*Past repayments in Jun-2005*	*21*	*Amount paid in Jun-2005*
10	*Past repayments in May-2005*	*22*	*Amount paid in May-2005*
11	*Past repayments in Apr-2005*	*23*	*Amount paid in Apr-2005*
12	*Bill statement in Sep-2005*		

5.3.1 Shannon Entropy. The Shannon entropy, formalized by *Claude E. Shannon* in [39], is one of the most important metrics used in information theory. It reports the uncertainty associated with a random variable, allowing us to evaluate the average minimum number of bits needed to encode a string of symbols, based on their frequency.

More formally, given a set of values $v \in V$, the entropy $H(V)$ is defined as shown in the Equation 9, where $P(v)$ is the probability that the element v is present in the set V.

In the context of the classification methods, the entropy-based metrics are frequently used during the *feature selection* [13, 29, 30] process, which is aimed to detect a subset of relevant features (variables, predictors) to use during the definition of the classification model. We use it for this task, dynamically, as described in Section 4.3.

$$H(V) = -\sum_{v \in V} P(v) log_2[P(v)] \tag{9}$$

5.3.2 Accuracy. The *Accuracy* metric reports the number of instances correctly classified, compared to the total number of them.

More formally, given a set of instances X to be classified, it is calculated as shown in Equation 10, where $|X|$ stands for the total number of instances, and $|X^{(+)}|$ for the number of those correctly classified.

$$Accuracy(X) = \frac{|X^{(+)}|}{|X|} \tag{10}$$

5.3.3 Sensitivity. Differently from the *accuracy* metric previously described, which takes into account all kind of classifications, through the *Sensitivity* we only obtain information about the number of instances correctly classified as *reliable*. It gives us an important information, since it evaluates the predictive power of our *FSP* approach in terms of capability to detect the reliable loan applications, offering a crucial decision support in real-world contexts.

More formally, given a set of instances X to be classified, the *Sensitivity* is calculated as shown in Equation 11, where $|X^{(TP)}|$ stands for the number of instances correctly classified as *reliable* and $|X^{(FN)}|$ for the number of *reliable* instances wrongly classified as *unreliable*.

$$Sensitivity(X) = \frac{|X^{(TP)}|}{|X^{(TP)}| + |X^{(FN)}|} \tag{11}$$

5.3.4 F-measure. The *F-measure* is the weighted average of the *precision* and *recall* metrics. It is a largely used metric in the

statistical analysis of binary classification, returning a value in a range $[0,1]$, where 0 is the worst value and 1 the best one.

More formally, given two sets X and Y, where X denotes the set of performed classifications of instances, and Y the set that contains the actual classifications of them, this metric is defined as shown in Equation 12.

$$\begin{aligned} &F\text{-}measure(X,Y) = 2 \cdot \frac{(precision(X,Y) \cdot recall(X,Y))}{(precision(X,Y) + recall(X,Y))} \\ &\text{with} \\ &precision(X,Y) = \frac{|Y \cap X|}{|X|}, \;\; recall(X,Y) = \frac{|Y \cap X|}{|Y|} \end{aligned} \tag{12}$$

5.4 Strategy

This section reports information about the strategy adopted during the execution of the experiments.

5.4.1 Cross-validation. In order to reduce the impact of data dependency, improving the reliability of the obtained results, all the experiments have been performed by using the *k-fold cross-validation* criterion, with *k=10*.

Each dataset is randomly shuffled, then it is divided in k subsets, and each k subset is used as test set, while the other *k-1* subsets are used as training set. The final result is given by the average of all results.

5.4.2 Tolerance Range. Considering that we have introduced a *tolerance range* ρ in the evaluation process performed by the Algorithm 1 (Section 4.4.1), we need to define its upper and lower bounds in the context of each dataset.

This range is used in the spectrum comparison process in order to determine when a Δ value, i.e., the difference between the magnitude of the same frequency component of two instances (one of them that belongs to the non-default cases and the other one that represents the instance to evaluate), as shown in Figure 6, must be considered acceptable or not (the classification of an instance as *accepted* or *rejected* depends on the results of these evaluations).

For each frequency component $f \in F$, measured in the set of past non-default instances I_+, we calculate the difference in terms of magnitude between each possible pair $(f, \hat{f})$, with $f, \hat{f} \in F$.

Denoting as $|f - \hat{f}|_{I_+}$ the aforementioned process of calculation of the differences between the magnitudes assumed by the same frequency component f in the dataset I_+, we define the *tolerance range* ρ of each $f \in F$ as shown in the Equation 13.

$$\begin{aligned} &\rho = [\rho_{min}, \rho_{max}] \\ &\text{with} \\ &\rho_{min} = min(|f - \hat{f}|_{I_+}), \;\; \rho_{max} = max(|f - \hat{f}|_{I_+}) \end{aligned} \tag{13}$$

Differently from our competitor approach (i.e., *Random Forests*), which allows us to determine the parameters value that leads toward its best performance, our approach adopts a dynamic method in order to determine the optimal range (minimum and maximum value) for each frequency component, instead to use a single range for all of them. For this reason, we can not determine these ranges of values a priori, since they are strictly related to the dataset taken into account, according to the Equation 13.

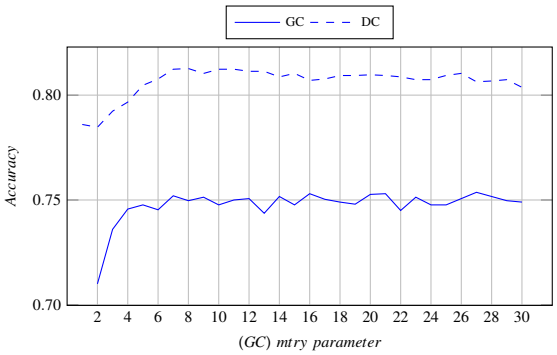

Figure 8: *Random Forests Tuning*

5.5 Competitor

Here, we describe the state-of-the-art approach chosen as competitor in order to evaluate the performance of our approach, beside the parameter tuning process aimed to optimize its performance.

5.5.1 Description. As mentioned previously, the implementation of the state-of-the-art approach to which we compare our approach was made in *R*, by using the *randomForest* and *ROCR* packages. For reproducibility reasons, we fix the seed of the random number generator by calling the *R* function *set.seed()*.

5.5.2 Parameters Tuning. In order to get the best performance from the *RF* approach, we need to perform a tuning process aimed to detect the optimal value of its configuration parameters.

The *caret* package in *R* provides an excellent functionality to perform this type of operation. Considering that *caret* supports only those algorithm parameters that have a crucial role in the tuning process, such as the *mtry* in the *RF* (number of variables randomly sampled as candidates at each split), we use *caret* in order to tune this parameter. The operation was performed by following the *grid search* approach, where each axis of the grid is an algorithm parameter and the points in the grid are specific parameters combinations.

The tests were stopped as soon as the measured accuracy did not improve further. Although the differences are minimal beyond certain values, as can be seen in the Figure 8, the experiments indicate as optimal value for *mtry 27* for the *GC* dataset and *8* for the *DC* dataset, since these values lead toward the maximum value of *Accuracy* (i.e., respectively, 75.36% and 81.26%).

5.6 Results

This section reports, presents and discusses the results of the performed experiments.

5.6.1 Overview. A first analysis of the experimental results (reported in Figure 9, Figure 10, and Figure 11) shows that:

(i) the performance of our *FSP* approach is very similar to the *RF* one, in terms of *Accuracy*, with both the *GC* and *DC* datasets;
(ii) the *FSP* approach gets better performance than *RF* one, in terms of *F-measure*, by using the *DC* dataset, and very close to it by using the *GC* dataset;
(iii) the *FSP* approach outperforms the *RF* one, in terms of *Sensitivity*, with both the *GC* and *DC* datasets.

The above aspects will be more deeply discussed in the next Section 5.6.2 and Section 5.6.3.

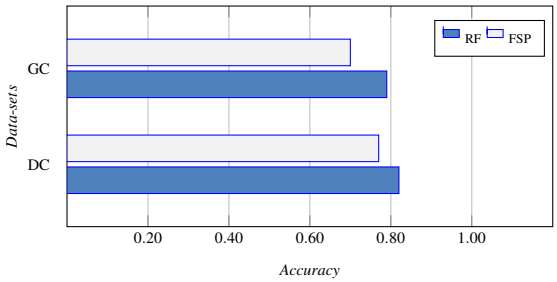

Figure 9: *Accuracy Performance*

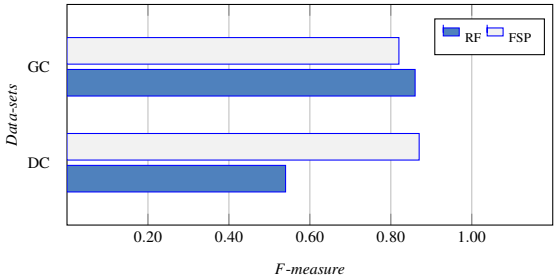

Figure 10: *F-measure Performance*

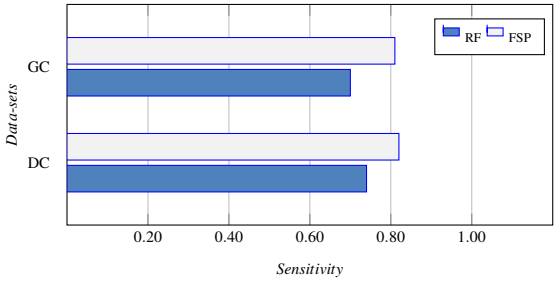

Figure 11: *Sensitivity Performance*

5.6.2 Discussion. The first observation that rises by examining in more detail the experimental results is related to the fact that our *FSP* approach gets performance very close (or better) to those of *RF* one, although it does not exploit the past default cases during the training process.

Another observation is instead related to the *F-measure* results, which show that the effectiveness of the *FSP* approach increases with the number of past non-default instances involved in the training process (*DC* dataset), differently from the *RF* approach, where this does not happen, although its training process involves both the default and the non-default past cases.

It should be noted that, in the light of the obtained results, the proactivity that characterizes our approach can reduce/overcome the *cold-start* problem described in Section 2.3, allowing a real-world system to operate even in the absence of previous cases of default instances, with all the advantages that derive from it.

The last but not less important observation is related to the results in terms of *sensitivity*, which show significant improvements, compared to the state-of-the-art *RF* approach taken into account. It means that the number of correct true positive classifications of instances is higher than that obtained by the *RF* approach, and this provides a clear benefit in a real-world context.

5.6.3 Benefits and Limitation. The experimental results presented and discussed before show that our approach performs similarly to one of the best performing state-of-the-art approaches such as *Random Forests*, although it operates in a proactive manner.

In addition, it is able to outperform *Random Forests* when the training process involves a large number of previous non-default cases, proving to be more effective than its competitor in the identification of the reliable instances.

On the basis of these results, a benefit related to the adoption of our *credit scoring* approach is its ability to face the data unbalance problem that reduces the effectiveness of the canonical approaches, since it exploits only a class of data in the model definition process (i.e., the previous non-default instances). Such proactive strategy also reduces/overcomes the well-known *cold-start* problem.

Another benefit is instead related to the fact that the model used during the evaluation process, based on the spectral pattern of the instances, is more stable than the canonical one, because the frequency components are less influenced by the data heterogeneity.

For the aforementioned reasons, our approach can be used in order to create hybrid approaches able to operate in all contexts, by combining its capability to operate proactively with the advantages offered by the non-proactive state-of-the-art approaches.

We can also identify as main limitation of our approach its few benefits in those cases where it exists a balanced data distribution with enough default and non-default cases to use during the model training. However, it should be underlined that this represents an uncommon real-world scenario.

6 CONCLUSIONS AND FUTURE WORK

The *credit scoring* techniques cover a crucial role in many financial contexts (i.e., bank loans, mortgage lending, insurance policies, etc.). They are adopted by the financial operators in order to assess the potential risks related to the customer applications, allowing them to reduce the losses due to default.

In this paper we proposed a novel approach of *credit scoring* able to classify the new instances as *accepted* or *rejected* by evaluating them in terms of frequency spectral pattern. This operation is performed by moving the evaluation process from the canonical domain to a frequency one, where the evaluation model is defined by using only the past non-default loan applications.

Such strategy presents two main advantages, the first of them is related to its ability to face the data unbalance issue, facing at the same time the *cold-start* problem, and the second one is related to its capability to define a model only by exploiting the non-default previous instances, allowing a system to operate proactively.

Future work would explore the effect, in terms of performance, of the inclusion of the default past instances in the model definition process, evaluating the advantages and disadvantages of the adoption of such non-proactive strategy.

Another interesting study would be to experiment the exploitation of other characteristics of the instances represented in the frequency domain, with the objective to improve the effectiveness of the classification algorithm.

A secondary but also interesting future work would be the evaluation of our approach in the context of heterogeneous environments, where numerous types of financial data are involved (e.g., the electronic commerce environment).

Acknowledgments

This research is partially funded by Regione Sardegna under project "Next generation Open Mobile Apps Development" (NOMAD), "Pacchetti Integrati di Agevolazione" (PIA) - Industria Artigianato e Servizi - Annualità 2013.

REFERENCES

[1] Mahmood Alborzi and Mohammad Khanbabaei. 2016. Using data mining and neural networks techniques to propose a new hybrid customer behaviour analysis and credit scoring model in banking services based on a developed RFM analysis method. *IJBIS* 23, 1 (2016), 1–22. DOI:http://dx.doi.org/10.1504/IJBIS.2016.078020

[2] Shawkat Ali and Kate A. Smith. 2006. On learning algorithm selection for classification. *Appl. Soft Comput.* 6, 2 (2006), 119–138. DOI:http://dx.doi.org/10.1016/j.asoc.2004.12.002

[3] Josh Attenberg and Foster J. Provost. 2010. Inactive learning?: difficulties employing active learning in practice. *SIGKDD Explorations* 12, 2 (2010), 36–41. DOI:http://dx.doi.org/10.1145/1964897.1964906

[4] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. 2004. A study of the behavior of several methods for balancing machine learning training data. *ACM Sigkdd Explorations Newsletter* 6, 1 (2004), 20–29.

[5] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian K. Tharakunnel, and J. Christopher Westland. 2011. Data mining for credit card fraud: A comparative study. *Decision Support Systems* 50, 3 (2011), 602–613. DOI:http://dx.doi.org/10.1016/j.dss.2010.08.008

[6] Antonio Blanco-Oliver, Rafael Pino-Mejías, Juan Lara-Rubio, and Salvador Rayo. 2013. Credit scoring models for the microfinance industry using neural networks: Evidence from Peru. *Expert Syst. Appl.* 40, 1 (2013), 356–364. DOI:http://dx.doi.org/10.1016/j.eswa.2012.07.051

[7] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32. DOI:http://dx.doi.org/10.1023/A:1010933404324

[8] Jeff Brill. 1998. The importance of credit scoring models in improving cash flow and collections. *Business Credit* 100, 1 (1998), 16–17.

[9] Iain Brown and Christophe Mues. 2012. An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Syst. Appl.* 39, 3 (2012), 3446–3453. DOI:http://dx.doi.org/10.1016/j.eswa.2011.09.033

[10] Sherry Y Chen and Xiaohui Liu. 2004. The contribution of data mining to information science. *Journal of Information Science* 30, 6 (2004), 550–558.

[11] Bo-Wen Chi and Chiun-Chieh Hsu. 2012. A hybrid approach to integrate genetic algorithm into dual scoring model in enhancing the performance of credit scoring model. *Expert Syst. Appl.* 39, 3 (2012), 2650–2661. DOI:http://dx.doi.org/10.1016/j.eswa.2011.08.120

[12] Sven F Crone and Steven Finlay. 2012. Instance sampling in credit scoring: An empirical study of sample size and balancing. *International Journal of Forecasting* 28, 1 (2012), 224–238.

[13] Manoranjan Dash and Huan Liu. 1997. Feature Selection for Classification. *Intell. Data Anal.* 1, 1-4 (1997), 131–156. DOI:http://dx.doi.org/10.1016/S1088-467X(97)00008-5

[14] RH Davis, DB Edelman, and AJ Gammerman. 1992. Machine-learning algorithms for credit-card applications. *IMA Journal of Management Mathematics* 4, 1 (1992), 43–51.

[15] Jeffrey Dean and Sanjay Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Commun. ACM* 51, 1 (2008), 107–113. DOI:http://dx.doi.org/10.1145/1327452.1327492

[16] Vijay S Desai, Jonathan N Crook, and George A Overstreet. 1996. A comparison of neural networks and linear scoring models in the credit union environment. *European Journal of Operational Research* 95, 1 (1996), 24–37.

[17] Pinar Donmez, Jaime G. Carbonell, and Paul N. Bennett. 2007. Dual Strategy Active Learning. In *ECML (Lecture Notes in Computer Science)*, Vol. 4701. Springer, 116–127.

[18] Michael Doumpos and Constantin Zopounidis. 2014. Credit Scoring. In *Multicriteria Analysis in Finance*. Springer, 43–59.

[19] Pierre Duhamel and Martin Vetterli. 1990. Fast Fourier transforms: a tutorial review and a state of the art. *Signal processing* 19, 4 (1990), 259–299.

[20] Albert Fensterstock. 2005. Credit scoring and the next step. *Business Credit* 107, 3 (2005), 46–49.

[21] Ignacio Fernández-Tobías, Paolo Tomeo, Iván Cantador, Tommaso Di Noia, and Eugenio Di Sciascio. 2016. Accuracy and Diversity in Cross-domain Recommendations for Cold-start Users with Positive-only Feedback. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, Shilad Sen, Werner Geyer, Jill Freyne, and Pablo Castells (Eds.). ACM, 119–122. DOI:http://dx.doi.org/10.1145/2959100.2959175

[22] David J. Hand. 2009. Measuring classifier performance: a coherent alternative to the area under the ROC curve. *Machine Learning* 77, 1 (2009), 103–123. DOI:http://dx.doi.org/10.1007/s10994-009-5119-5

[23] Haibo He and Edwardo A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Trans. Knowl. Data Eng.* 21, 9 (2009), 1263–1284. DOI:http://dx.doi.org/10.1109/TKDE.2008.239

[24] WE Henley and others. 1997. Construction of a k-nearest-neighbour credit-scoring system. *IMA Journal of Management Mathematics* 8, 4 (1997), 305–321.

[25] WE Henley and David J Hand. 1996. A k-nearest-neighbour classifier for assessing consumer credit risk. *The Statistician* (1996), 77–95.

[26] William Edward Henley. 1994. *Statistical aspects of credit scoring.* Ph.D. Dissertation. Open University.

[27] Nan-Chen Hsieh. 2005. Hybrid mining approach in the design of credit scoring models. *Expert Systems with Applications* 28, 4 (2005), 655–665.

[28] Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intell. Data Anal.* 6, 5 (2002), 429–449.

[29] Feng Jiang, Yuefei Sui, and Lin Zhou. 2015. A relative decision entropy-based feature selection approach. *Pattern Recognition* 48, 7 (2015), 2151–2163. DOI:http://dx.doi.org/10.1016/j.patcog.2015.01.023

[30] Nojun Kwak and Chong-Ho Choi. 2002. Input feature selection for classification problems. *IEEE Trans. Neural Networks* 13, 1 (2002), 143–159. DOI:http://dx.doi.org/10.1109/72.977291

[31] Tian-Shyug Lee and I-Fei Chen. 2005. A two-stage hybrid credit scoring model using artificial neural networks and multivariate adaptive regression splines. *Expert Systems with Applications* 28, 4 (2005), 743–752.

[32] Stefan Lessmann, Bart Baesens, Hsin-Vonn Seow, and Lyn C. Thomas. 2015. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research* 247, 1 (2015), 124–136. DOI:http://dx.doi.org/10.1016/j.ejor.2015.05.030

[33] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. 2014. Facing the cold start problem in recommender systems. *Expert Syst. Appl.* 41, 4 (2014), 2065–2073. DOI:http://dx.doi.org/10.1016/j.eswa.2013.09.005

[34] Ana Isabel Marqués, Vicente García, and José Salvador Sánchez. 2013. On the suitability of resampling techniques for the class imbalance problem in credit scoring. *JORS* 64, 7 (2013), 1060–1070. DOI:http://dx.doi.org/10.1057/jors.2012.120

[35] Loretta J Mester and others. 1997. Whats the point of credit scoring? *Business review* 3 (1997), 3–16.

[36] Chorng-Shyong Ong, Jih-Jeng Huang, and Gwo-Hshiung Tzeng. 2005. Building credit scoring models using genetic programming. *Expert Systems with Applications* 29, 1 (2005), 41–47.

[37] Jon T. S. Quah and M. Sriganesh. 2008. Real-time credit card fraud detection using computational intelligence. *Expert Syst. Appl.* 35, 4 (2008), 1721–1732. DOI:http://dx.doi.org/10.1016/j.eswa.2007.08.093

[38] Alan K Reichert, Chien-Ching Cho, and George M Wagner. 1983. An examination of the conceptual issues involved in developing credit-scoring models. *Journal of Business & Economic Statistics* 1, 2 (1983), 101–114.

[39] Claude E. Shannon. 2001. A mathematical theory of communication. *Mobile Computing and Communications Review* 5, 1 (2001), 3–55.

[40] Mohammad Siami, Zeynab Hajimohammadi, and others. 2013. Credit scoring in banks and financial institutions via data mining techniques: A literature review. *Journal of AI and Data Mining* 1, 2 (2013), 119–129.

[41] Steven W Smith and others. 1997. The scientist and engineer's guide to digital signal processing. (1997).

[42] Le Hoang Son. 2016. Dealing with the new user cold-start problem in recommender systems: A comparative review. *Inf. Syst.* 58 (2016), 87–104. DOI:http://dx.doi.org/10.1016/j.is.2014.10.001

[43] V Thanuja, B Venkateswarlu, and GSGN Anjaneyulu. 2011. Applications of Data Mining in Customer Relationship Management. *Journal of Computer and Mathematical Sciences Vol* 2, 3 (2011), 399–580.

[44] Veronica Vinciotti and David J Hand. 2003. Scorecard construction with unbalanced class sizes. *Journal of Iranian Statistical Society* 2, 2 (2003), 189–205.

[45] Gang Wang, Jinxing Hao, Jian Ma, and Hongbing Jiang. 2011. A comparative assessment of ensemble learning for credit scoring. *Expert Syst. Appl.* 38, 1 (2011), 223–230. DOI:http://dx.doi.org/10.1016/j.eswa.2010.06.048

[46] Gang Wang, Jian Ma, Lihua Huang, and Kaiquan Xu. 2012. Two credit scoring models based on dual strategy ensemble trees. *Knowl.-Based Syst.* 26 (2012), 61–68. DOI:http://dx.doi.org/10.1016/j.knosys.2011.06.020

[47] Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K. Tsou. 2008. Active Learning with Sampling by Uncertainty and Density for Word Sense Disambiguation and Text Classification. In *COLING 2008, 22nd International Conference on Computational Linguistics, Proceedings of the Conference, 18-22 August 2008, Manchester, UK*, Donia Scott and Hans Uszkoreit (Eds.). 1137–1144.