Statistical stability indices for LIME: obtaining reliable explanations for Machine Learning models

Giorgio Visani[a,b], Enrico Bagli[b], Federico Chesani[a], Alessandro Poluzzi[b] and Davide Capuzzo[b]

[a]Universit degli Studi di Bologna, Dipartimento di Ingegneria e Scienze Informatiche, viale Risorgimento 2, 40136 Bologna (BO), Italy;
[b]CRIF S.p.A., via Mario Fantin 1-3, 40131 Bologna (BO), Italy

ARTICLE HISTORY
Compiled February 3, 2020

ABSTRACT
Nowadays we are witnessing a transformation of the business processes towards a more computation driven approach. The ever increasing usage of Machine Learning techniques is the clearest example of such trend.

This sort of revolution is often providing advantages, such as an increase in prediction accuracy and a reduced time to obtain the results. However, these methods present a major drawback: it is very difficult to understand on what grounds the algorithm took the decision.

To address this issue we consider the LIME method.
We give a general background on LIME then, we focus on the stability issue: employing the method repeated times, under the same conditions, may yield to different explanations.

Two complementary indices are proposed, to measure LIME stability. It is important for the practitioner to be aware of the issue, as well as to have a tool for spotting it. Stability guarantees LIME explanations to be reliable, therefore a stability assessment, made through the proposed indices, is crucial.

As a case study, we apply both Machine Learning and classical statistical techniques to Credit Risk data. We test LIME on the Machine Learning algorithm and check its stability. Eventually, we examine the goodness of the explanations returned.

KEYWORDS
Credit Scoring; Machine Learning; Explainability; LIME; Stability

1. Introduction

Nowadays, more and more interest is devoted to the concept of "learning from the data". This is due to the huge availability of datasets, as well as to the increased computational power which allows complex algorithms to run and deliver a result in a relatively short time.

"Learning from data" usually concerns an outcome measurement, both quantitative or categorical. The best guess about the outcome is carried out, based on a set of features. Using the data collected about the process, a prediction model is built,

CONTACT Giorgio Visani. Email: giorgio.visani2@unibo.it

with the aim of predicting the outcome for new unseen objects (Hastie, Tibshirani, & Friedman, 2009).

The concept of making predictions about the future, empowered by the learning, has been a widely studied topic in statistics. Simple algorithms and methods have been developed to address this topic, among them the most famous are Linear Regression and Generalised Linear Models (Greene, 2003).

Recently, with the advent of powerful computing tools, more sophisticated techniques have been developed. In particular, Machine Learning models are able to perform intelligent tasks usually done by humans, supporting the automation of data driven processes. The most popular among such models are Neural Networks.

Despite the enhanced accuracy, Machine Learning models display weakness especially when it comes to interpretability, i.e. the ability to explain or to present in understandable terms to a human" (Hall & Gill, 2018). They usually adopt large model structures and refine the result using a huge number of iterations. The logic underlying the model ends up hidden under potentially many strata of mathematical calculations, as well as scattered across a too vast architecture, preventing humans from grasping it.

The interpretability issue has been tackled from various perspectives. The two main approaches are to strive for transparent models, namely models whose logic is understandable out of the box, or to use a black-box model and apply techniques enabling the understanding of the machine learning reasoning. Into the latter class, a subsequent partition of the methods divides them into Global and Local Explainability techniques (Guidotti et al., 2018). Global methods aim to give an understanding of the model as a whole: the explanation should apply to all the records in the dataset. Local methods instead, attempt to provide very good understanding just for a small portion of records.

Herein, we focus on LIME (Local Interpretable Model-agnostic Explanations), a local interpretability technique, developed by Ribeiro, Singh, & Guestrin, 2016.

The technique may suffer from a lack of stability, namely repeated applications of the method under the same conditions may obtain different results. It is desirable to obtain unambiguous explanations from LIME, in this way the practitioner is ensured to discover always the same motivation for an individual to take on its prediction.
Unfortunately, this problem is rarely taken into consideration and even worse, many times the issue is not spotted at all, e.g. when just a single call to the method is done and the result is considered to be okay without further checks.

The leading aim of this work is to find a reliable way to measure LIME stability. To this end, we derive a pair of complementary stability indices, calculated on repeated calls of the method, to evaluate the similarity of the results. We consider them to be the best way for performing the stability assessment. They may be applied on every trained LIME method and will allow the practitioner to be aware about potential instability of the results, otherwise to ensure that the trained method is consistent.

In this contribution, a brief introduction of Machine Learning models is presented in Chapter 2. The interpretability issue is dealt with in Chapter 3, where LIME is presented as a technique to solve it. A thorough discussion about LIME stability and how to test it, using the newly designed stability indices, can be found in Chapter 4. Eventually, a practical application of the method in the Credit Risk Modelling field is shown in Chapter 5. Chapter 6 is dedicated to Discussion and Conclusions.

2. Machine Learning Models

The classical definition of Machine Learning dates back to 1997 on behalf of Tom Mitchell (1997):

> A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.

By this train of thought, almost any kind of prediction algorithm may fall into the class of Machine Learning models. As an example, consider the parameter tuning phase of the Logistic Regression: it relies on an iterative algorithm, usually Newton-Raphson, which improves the estimated models performance at each iteration, measured by the increase in the Likelihood value. According to Mitchell's definition, Logistic Regression belongs to the Machine Learning class of algorithms.

Because of the extent of such a general framework, in this contribution we consider only non-parametric Machine Learning models. Non-parametric models estimate the relationship between the target variable and the predictor variables, without constraining it to have a precise functional form. This peculiarity allows to model non-linear relations, making the technique more flexible compared to parametric methods. Machine Learning models of this kind usually outperform classical methods in non-linear settings and achieve the same results when the "nature of true relations" is linear.

Machine Learning models are also capable of discovering interactions among features and to account for that into the learning process, without being explicitly programmed to do so. The same relationships between features can be retrieved also using a classical statistical model, but we have to encode them into the algorithm. It is not possible to test all the combinations of features because such models cannot handle a high number of variables. Since we have to choose which interactions to code, classical frameworks require a deep understanding of the dataset and the underlying process that generated the data. Machine Learning models, on the other hand, produce a good result even without requiring to have background domain knowledge for the training dataset.

3. Background on LIME

LIME is a method developed by Ribeiro et al., 2016 for explaining black-box models, i.e. models whose inner logic is hidden and not clearly understandable.
It provides a number of explainable models which closely resemble the original model behaviour. Each model is specific for an input point x: only in its neighbourhood the explainable model's predictions are guaranteed to be very close to the black-box ones. This peculiarity places LIME among the Local Explainability tools.

3.1. General Idea

LIME aims to approximate the black-box model f with a simple function g around the point of interest x. g is required to lie into the class of explainable models G.

$$f : \mathbb{R}^P \to \mathbb{R}, \quad \text{black-box model}$$
$$g : \mathbb{R}^p \to \mathbb{R}, \quad \text{explainable model}$$

where P is the number of features employed by the black-box model, to make predictions about the response variable. The explainable model g uses only p of the original P variables, in order to reduce the complexity.
Solving the following optimisation problem, we obtain the function g most similar to f in the neighbourhood of x.

$$arg \min_{g \in G} L(f, g, \pi_x) + \Omega(g)$$
$$\Omega(g) : \text{complexity of g}$$
$$L : \text{loss function}$$
$$\pi_x : \text{weight assigned according to } x \text{ proximity}$$

Chosen a given individual x, LIME returns a local explainable model g, which in turn provides the most important variables to predict the points in the x neighbourhood (see Figure 1).

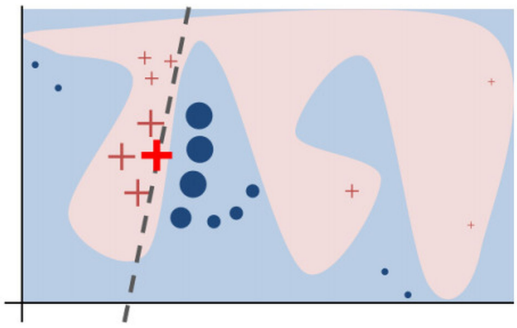


Figure 1. LIMEs modus operandi.Courtesy of Ribeiro et al., 2016

3.2. LIME Algorithm in detail

LIME relies on producing new points, generated from a multivariate distribution of the features in the dataset. The features are considered to follow a Normal distribution, whose parameters are inferred from the dataset. For the purpose of data generation, each feature is assumed to be independent from the others.
The points are generated all over the space of the dataset variables.

In order to account for locality, LIME weights each new point using a Gaussian Kernel. Its purpose is to assign a weight π_x to each point, based on its distance from the individual to be explained.

Next step is to query the black-box model and obtain the predicted values for the new points. Doing so, we end up with a brand new dataset.

Such dataset undergoes a rescaling process, which standardises each X feature, leaving untouched the response variable. This allows to compare the contribution of each feature on a similar scale, namely the standard deviation of each variable.

In order to obtain a human-understandable linear model, it is mandatory to use only a bunch of features. Therefore LIME performs also a feature selection step, done usually with Lasso technique. This allows the explanations to be compact and human readable. The number of variables to be retained, namely p, is decided by the practitioner.

On the standardised p-dimensional dataset, LIME performs Ridge Regression, i.e. a Linear Regression combined with a penalty related to the $\ell 2$ norm of the coefficients (Hoerl & Kennard, 1970), used to prevent overfitting. The model training is done in a weighted fashion: each point contributes to the model according to its weight π_x.

The result is a linear model, which provides understanding of the process through its coefficients: the higher the coefficient, the bigger the variation in the value of the response variable when the feature is changed. The sign of the coefficient tells us the direction of the variation, namely if we will face a decrease or an increase of the output value.

3.3. LIME Drawbacks

LIME is sensitive to the dataset dimensionality: when it is employed to interpret a Machine Learning model built using a huge number of variables, the local explanation is unable to discriminate among relevant and irrelevant features.

This phenomenon is due to the weighting kernel. Generally speaking, it can be considered as a similarity (or distance) function, thus it inherits the drawbacks of this class. As thoroughly described by Beyer, Goldstein, Ramakrishnan, & Shaft, 1999, in high dimensional datasets, chosen a fixed point, the distance to its nearest data point approaches the distance to the farthest one, as dimensionality increases.
LIME applies the kernel function before variable reduction, thus for high dimensional datasets, the kernel is not able to distinguish between near and distant points, considering all of them approximately at the same distance. This results in a loss of the locality concept and consequently in a bad performance of the algorithm.

Such occurrence is intuitively shown in Figure 2. In it, the Credit Scoring dataset used in Section 5 has been employed to train a Gradient Boosting Tree model using 100 variables. Although the Gradient Boosting has shown good performance in such a setting, LIME applied to the model has not been able to discriminate among important and irrelevant regressors. In particular, many features exhibit low values and almost all of them are equally important.

This weakness curbs LIME's employment on black-box models handling high dimensional datasets. To date, it is a practitioner duty to ensure the dataset dimensionality is low enough for LIME to work well. This usually requires feature selection, upstream of data modelling.

4. Evaluating the LIME Stability

Consider choosing a specific individual and performing LIME on it, several times. Indeed, it is desirable to obtain the same explanations from each call.

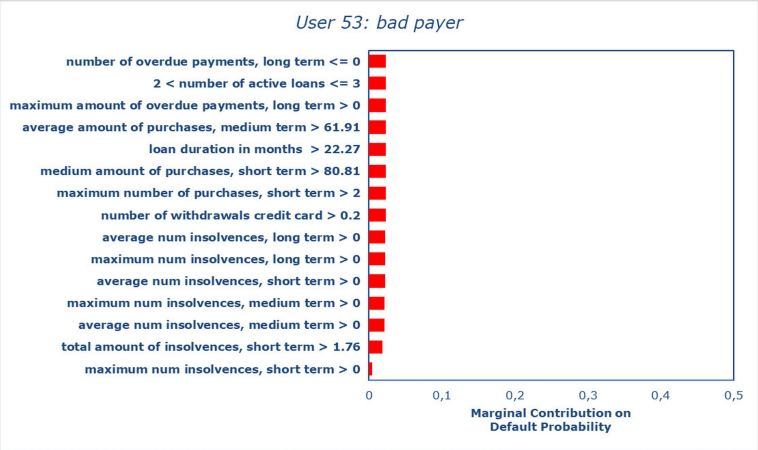


Figure 2. LIME explanations are not informative when applied to Machine Learning models with many independent variables, in this case Gradient Boosting model using 100 features.

Every time LIME is employed, it generates new data points, which follow the same distribution (law) but are different among distinct applications. This is due to the random nature of the sampling. Using different points it may happen to obtain divergent explainable models g, thus different explanations, for the chosen individual.
Under those circumstances, LIME stability cannot be taken for granted.

This is a LIME's well-known issue, although it is rarely addressed. As far as we are aware of, only in Shankaranarayana and Runje (2019) we found mention of a stability assessment. The authors propose to compare standard deviations of the coefficients, as well as the ratio of coefficients standard deviations to means, among 10 explainable models g, generated by different LIME calls on the same individual.

Although we consider Shankaranarayana's work headed in the right direction, we feel more work has to be done in order to provide solid grounds and mathematical rigour to the metrics evaluating LIME stability.

Hereafter, a formal description of the framework considered, in order to evaluate LIME stability.
Consider the black-box model f composed by P variables, a chosen point to be explained x and a fixed number of variables p, used in LIME's explainable models. The Weighted Ridge Regression model g (LIME's output) can be viewed as a mapping function between the set of variables and the respective coefficients.

$$g : \mathcal{F} \rightarrow \mathbb{R} \tag{1}$$

where $\mathcal{F}$ is the set of variables, of cardinality P. p out of P features will be associated with a value different from 0, the others $P-p$ variables will have 0 coefficient, meaning they are irrelevant to the model. The formulation $g(\texttt{feat})$ indicates the coefficient value of the feature named `feat`, in the model g.

We perform m different calls to LIME on the model f and the individual x, obtaining m different explainable models $g_1 ... g_m$.

We want to: (i) check whether different g are composed by the same variables, (ii) compare the coefficients of the same variable among $g_1 ... g_m$ and test whether they can

be considered equal.

To this purpose, we devise two complementary indices: the Variables Stability Index (VSI) and Coefficients Stability Index (CSI).

4.1. Variables Stability Index: VSI

The Variables Stability Index (VSI), whose steps are explained in Algorithm 1, addresses the first point, namely it compares the variables composition of the $g_1 ... g_m$ models.

We consider the set $C_2^m(g_1 ... g_m)$ of all possible combinations of the m explainable models, two by two. The generic element of $C_2^m(g_1 ... g_m)$ is the pair (g_α, g_β).

We define a measure of concordance among the two explainable models in each pair:

$$\begin{aligned} &\texttt{pair} = (g_\alpha, g_\beta) \\ &\mathcal{F}_\alpha = \{\texttt{feat} \in \mathcal{F} : g_\alpha(\texttt{feat}) \neq 0\} \\ &\mathcal{F}_\beta = \{\texttt{feat} \in \mathcal{F} : g_\beta(\texttt{feat}) \neq 0\} \\ &\textsc{concordance}(\texttt{pair}) = |\mathcal{F}_\alpha \cap \mathcal{F}_\beta| \end{aligned} \tag{2}$$

where $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$ represent respectively the variables used in the explainable models g_α and g_β. The CONCORDANCE function returns an integer value, namely the cardinality of the intersection between $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$, ranging from 0 to p. It represents the number of variables used by both g_α and g_β.

Algorithm 1: Variables Stability Index (VSI)

Input: $g_1 ... g_m$
$n = 0$
for *pair* *in* $C_2^m(g_1 ... g_m)$ **do**
  $n = n + \frac{\textsc{concordance}(\texttt{pair})}{p}$
end
VSI $= \frac{n}{|C_2^m(g_1 ... g_m)|}$
Output: VSI

We evaluate the CONCORDANCE over all the pairs in $C_2^m(g_1 ... g_m)$ and we average them, obtaining the VSI index, ranging from 0 to 1. We express the index as a percentage: it now spans from 0 to 100, the more it approaches 100 the more the variables found in different applications are the same.

4.2. Coefficients Stability Index: CSI

The equality between coefficients of the different $g_1 ... g_m$ models is now under investigation. In the following, we derive the statistical distribution of the coefficients and we rely on it, to create confidence intervals and possibly statistical tests.

It is a well-known result (Greene, 2003), that under the classic assumptions of Linear Regression, the coefficients are guaranteed to follow a Gaussian distribution. This is not sufficient, since we deal with Weighted Ridge Regression.

In van Wieringen (2015), the distribution of the Ridge Regression estimator is given by the formula:

$$\hat{\beta}(\lambda) \sim \mathcal{N}\Big((X^TX + \lambda I_p)^{-1}X^TX\beta, \sigma^2(X^TX + \lambda I_p)^{-1}X^TX[(X^TX + \lambda I_p)^{-1}]^T\Big) \quad (3)$$

where X is the matrix of observations. In our setting, X is composed by the points randomly sampled inside LIME. The matrix I_p stands for the identity matrix (dimensions $p \times p$). σ^2 is the variance of the $\mathcal{E}_i$ random variables describing the errors per each sampled point. Under the Regression assumptions the errors $\mathcal{E}_i$ are independent and identically distributed (IID) following a Gaussian law: $\mathcal{E}_i \sim \mathcal{N}(0, \sigma^2)$. λ stands for the Ridge regularisation coefficient. The vector β represents the true values of the coefficients in population, whereas $\hat{\beta}$ consists in the estimates of the true values, using the X dataset.

In our setting, we may consider the β values as the unknown coefficients of the best linear approximation of f in the neighbourhood of x. LIME aims to provide $\hat{\beta}$ closest as much as possible to the unknown β values.

Concerning Weighted Regression, it is usually estimated via Generalised Least Squares (GLS) which guarantee the distribution of its estimators to be the following (see Johnston & DiNardo, 1972) :

$$\hat{\beta} \sim \mathcal{N}\Big((X^TWX)^{-1}X^TWX\beta, \sigma^2(X^TWX)^{-1}\Big) \quad (4)$$

In the formula, W is the $n \times n$ diagonal matrix of weights per each unit. In our setting, the W matrix is populated by the kernel weights calculated on the distance of each sampled point from x.

It is important to recall that σ^2 is an unknown value and we are requested to obtain an unbiased estimator inferred from the data. Such estimator takes the form:

$$\hat{\sigma}^2 = \frac{\mathcal{E}W\mathcal{E}^T}{n-p}$$

for the Weighted Regression, as stated in Johnston and DiNardo (1972). $\mathcal{E}$ stands for the vector of the errors per each sampled point: $\mathcal{E} = (\mathcal{E}_1 ... \mathcal{E}_n)$. As far as Ridge Regression is concerned, the variance estimator remains unchanged from the Linear Regression's one (van Wieringen, 2015).

Using the building blocks stated before, we derive the distribution of the Weighted Ridge Regression estimator. Starting from the Ridge Regression law (Equation 3), we know (Billingsley, 2008) that the Gaussian distribution is invariant whenever we employ a matrix of known weights. This guarantee the Weighted Ridge law of the coefficients to be Gaussian. Its distribution is [1]:

$$\begin{aligned}\hat{\beta}(\lambda) \sim \mathcal{N}\Big((X^TWX + \lambda I_p)^{-1}X^TWX\beta, \\ \sigma^2(X^TWX + \lambda I_p)^{-1}X^TWX[(X^TWX + \lambda I_p)^{-1}]^T\Big)\end{aligned} \quad (5)$$

[1] We do not state all the derivations of the expectation and variance formulae, for the sake of readability.

We provide also the formula for the variance estimator of the Weighted Ridge Regression [2]:

$$\hat{\sigma}^2 = \frac{\mathcal{E} W \mathcal{E}^T}{n - p} \tag{6}$$

where n is the number of data points sampled inside LIME, p denotes the number of variables considered in the explainable model.

Knowing the distribution of the coefficients, we might derive a test statistic to assess a null hypothesis of equality. This comparison can be carried out also among coefficients of two different regression models, as long as they were estimated on two independent samples drawn from the same law, as derived by Brame, Paternoster, Mazerolle, & Piquero, 1998 for the coefficients of two distinct Linear Regressions.

This assumption holds true in our experimental design, since the data are sampled from the features' distribution inferred from the original data. It means that the true generating distribution of $X_1..X_m$, i.e. the datasets sampled in repeated LIME calls, is identical, while the differences among them are attributable only to the sampling variance.

Unfortunately, the simplifications carried out in Brame et al., 1998 and Greene, 2003 in order to derive the t-test statistic, rely on the equality of the expected value of the two coefficients taken into consideration. This is true in Linear Regression, but the framework brakes down using a regularisation technique such as Ridge: the regulariser trades off the unbiasedness of the estimator in exchange for a possibly strong reduction of the variance.

Since the estimator is not unbiased anymore, the expected value is now depending on the design matrix X. As stated before, different LIME calls give rise to different design matrices, this implies the expected values of a specific variable, taken from two different explainable models g_α and g_β, to be different: $\mathrm{E}[g_\alpha(\texttt{feat}) - g_\beta(\texttt{feat})] \neq 0$. This result causes the derivation of the t-test statistic to break down.

Testing the null hypothesis of equality has proven tricky and not easily solvable, hence we rely on the Gaussian distribution of the coefficients to construct 95% confidence intervals. To do that, we design the function ConfInt, taking as input a g's coefficient and giving back its confidence interval:

$$\begin{aligned} \textsc{ConfInt}(g(\texttt{feat})) = [&g(\texttt{feat}) - 1.96 \sqrt{\mathrm{Var}(g(\texttt{feat}))}\,, \\ &g(\texttt{feat}) + 1.96 \sqrt{\mathrm{Var}(g(\texttt{feat}))}] \end{aligned} \tag{7}$$

where $\mathrm{Var}(g(\texttt{feat}))$ is calculated based on the distribution given in Equation 5.

We may consider the parameters to be different, within a 5% error rate, when the confidence intervals are not overlapped at all. Instead, we consider them to be stable whenever the confidence intervals overlap to some extent.

[2] In this formulation, we consider $\mathcal{E}$, as the errors of the Linear Regression model. In other words, the Weighted Ridge $\hat{\sigma}^2$ estimator is the same of Weighted Regression.
We may not use the errors of any Ridge model to calculate an unbiased estimator of the error variance σ^2, because Ridge regularisation term decreases the variance. Using such errors would cause the estimator to be biased towards 0.

To this purpose, we devise the binary function OVERLAP, which takes as input a generic pair of confidence intervals $\texttt{CIpair} = (\texttt{CI}_\alpha, \texttt{CI}_\beta)$ and returns either value 1 or 0, based on the overlap presence.

$$\text{OVERLAP}(\texttt{CIpair}) = \begin{cases} 0 & \text{if } \texttt{CI}_\alpha \cup \texttt{CI}_\beta = \varnothing \\ 1 & \text{otherwise} \end{cases} \tag{8}$$

The comparison among confidence intervals is carried out separately for each variable. Chosen a certain feature, we check through the $g_1 ... g_m$ explainable models if the feature is relevant (coefficient different from 0). Whenever this happens, we build the confidence interval for the coefficient, using the function CONFINT, and we consider the set of all confidence intervals, namely $\mathcal{M}$, for the chosen variable.
We create all the possible combinations of the $\mathcal{M}$ items, two by two. This results in the set $C_2^{|\mathcal{M}|}$, whose generic element is $\texttt{CIpair} = (\texttt{CI}_\alpha, \texttt{CI}_\beta)$. We calculate the overlap between the two intervals, using the OVERLAP function, for all the pairs in $C_2^{|\mathcal{M}|}$.

The outcome is a count variable, which we normalise dividing by the cardinality of the set $C_2^{|\mathcal{M}|}$. The value obtained ranges from 0 to 1 and it is called the Partial Index (PAR) for the variable considered. It represents a measure of concordance of the specific variable's coefficients among different LIME calls.
To achieve a general concordance metric, we average the Partial Indices of all the features and obtain the Coefficients Stability Index (CSI), ranging from 0 to 1. Consider now the index as a percentage, rescaling it from 0 to 100: the more CSI approaches 100, the more LIME coefficients may be considered stable in the neighbourhood of the chosen individual.

CSI steps are detailed in Algorithm 2.

4.3. Interpretation of the indices

The previously defined indices constitute a useful tool for assessing LIME stability in practical scenarios. By construction, VSI measures the concordance of the variables retrieved, whereas CSI tests the similarity among coefficients for the same variable, in repeated LIME calls.

Both of them range from 0 to 100.
High VSI values guarantee the variables retrieved in different LIME are almost always the same. On the contrary, low values testify explanations are not trustworthy: we may retrieve completely different variables explaining the same Machine Learning decision, according to different LIME calls.

As far as CSI is concerned, high values ensure LIME coefficient for each feature is reliable. Low values, instead, induce the practitioner to be very cautious: given a feature, the first LIME call will give back a certain value of the coefficient, but the one after is likely to retrieve a different value. Since the coefficient represents the impact of the feature on the Machine Learning decision, obtaining different values correspond to very different explanations.

Each index has a proper meaning and checks for a particular stability instance. Achieving high values for both of them ensures stability, however low values for only

Algorithm 2: Coefficients Stability Index (CSI)

Input: $g_1 ... g_m$
for feat *in* $\mathcal{F}$ **do**
  $\mathcal{M} = \{\}$
  for i *in* $1...m$ **do**
    if $g_i(\mathit{feat}) \neq 0$ **then**
      $\texttt{CI} = \textsc{ConfInt}\ (g_i(\texttt{feat}))$
      $\mathcal{M} = \mathcal{M} \cup \texttt{CI}$
    end
  end
  $n = 0$
  for CIpair *in* $C_2^{|\mathcal{M}|}$ **do**
    if OVERLAP *(CIpair)* **then**
      $n++$
    end
  end
  $\textsc{Par}_{\texttt{feat}} = \frac{n}{|C_2^{|\mathcal{M}|}|}$
end
CSI = mean(Par)
Output: CSI

one metric are still possible. Keeping the measurements separated allows for understanding which one of the two complementary definitions of stability has been violated by the trained LIME method.

5. Practical Application to Credit Risk Data

Credit Risk Modelling (CRM) consists in estimating the probability that a debtor will not repay the due amount. This task is regarded as a fully-fledged prediction task and as such, a variety of different learning techniques have been applied over the years in order to solve it.

Since long, statistical approaches have been exploited, forming the core techniques in this field. Recently also Machine Learning models have been given a chance. They have usually shown an increase in the prediction power, although they do not provide reliable explanations for the scores they come up with.

This is a particularly delicate issue in CRM, since it is a highly regulated field: GDPR (Kingston, 2017), as well as the "Ethical Guidelines for trustworthy AI" (HLEG, 2019) and the Report from the "European Banking Authority" (European Banking Authority (EBA), 2020) testify the care dedicated to such topics by the European Community.

In order to exploit the Machine Learning potential in the Credit Scoring field, it is mandatory to address the interpretability issue. To do that, our proposal concerns applying LIME on top of a well performing black-box algorithm. By doing so, we wish to retain the increased predictive power of Machine Learning, while providing meaningful explanations to the applicants involved, as well as to the regulator.

To validate the above mentioned approach, we use a real-life dataset representative of a loan application process. It comes from an anonymised statistical sample, obtained by pooling data from several Italian financial institutions.
In it, there are several demographic, economic and financial variables used as predictors, whereas the response variable consists of only two categories (bad payer, good payer), framing the problem as classification.
The dataset composition is shown in the Table 1.

Table 1. Dataset Composition

Data set name	Population	%Bad
Train set	39.418	2,9%
Test set	16.893	3,1%
Total	56.311	3%

Before using learning techniques of any kind, we take care of selecting the important features among the many variables available in the dataset. This is done for two reasons: (*i*) classical models are not performing well in high dimensional settings and (*ii*) LIME applied to high dimensional Machine Learning models would cause the method to fail. To this end, we retained only the most important 20 features, to be employed for learning purposes.

5.1. Logistic Regression model

On the dataset described above, we employ Logistic Regression along with a Machine Learning model, for the sake of comparison among the two techniques.
We consider Logistic Regression as the benchmark model, being explainable and widely used in Credit Scoring. The technique is well described in Agresti, 2015 and it usually obtains satisfactorily results on CRM datasets. Logistic Regression provides also interpretability out-of-the-box: the parameters derived from the best curves estimation, can be regarded as odds ratio, i.e. the ratio between the probability of default and non-default, namely $\frac{P(Y=1|X=x)}{P(Y=0|X=x)}$.

5.2. Machine Learning Model

About the choice of Machine Learning algorithms, we use tree-based Machine Learning models, specifically Gradient Boosting Trees (see Figure 3). They retain the enhanced predictive power of Machine Learning models, while having the additional advantage of requiring almost none pre-processing. Because of their structure, they are able to cope with outliers and extreme values easily.

5.3. Model comparison

The dataset has been divided into Train and Test set. Both the Logistic Regression model and the Gradient Boosting have been trained on the Training data, and their performances are evaluated on the Test set. The comparison is done by means of the Gini Index, considered the most reliable figure of merit of the model performances in CRM field (Hand, 2001). In Figure 4, the Lorentz Curve of the two models is displayed, along with the Gini values.

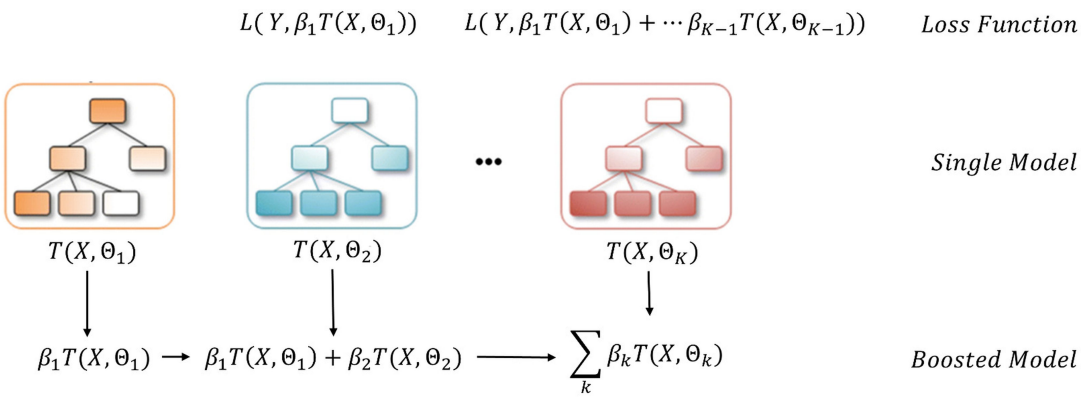


Figure 3. Gradient Boosting Tree model.

$T(X, \Theta_k)$ is the best Tree built at step k, its parameters Θ_k are chosen in order to minimise the Loss Function between the target variable Y and the Boosted model of the previous step.
The β_k parameter is the weight of the Tree, when added in the Boosted Ensemble, this is also chosen with respect to the Loss Function.

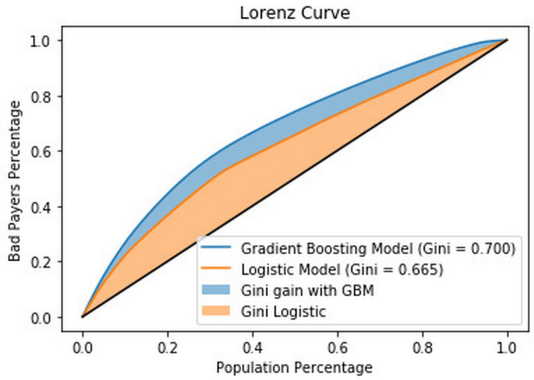


Figure 4. Lorenz Curve comparison: Gradient Boosting vs Logistic Regression

Using Gradient Boosting instead of Logistic Regression, it is possible to recognise an improvement in performance, testified by a Gini increase of more than 3 points. This result is consistent with the recent comparison between Statistical and Machine Learning models applied to the CRM field, carried out by (Moscatelli, Narizzano, Parlapiano, & Viggiano, 2019) at Banca d'Italia.

5.4. LIME applied to Credit Risk models

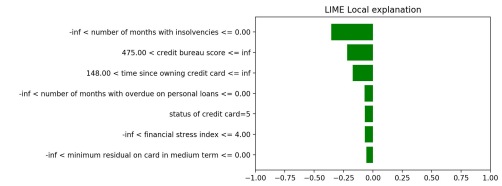
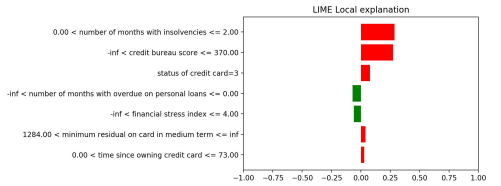
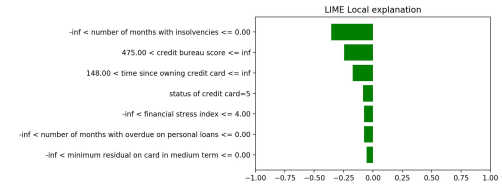
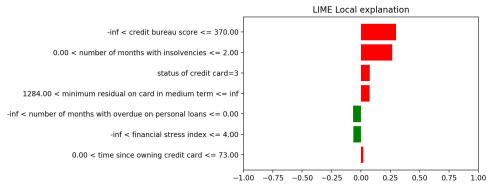
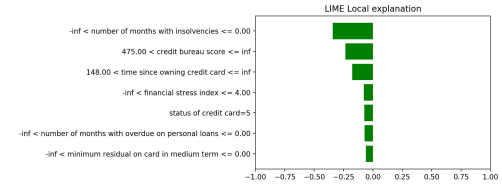
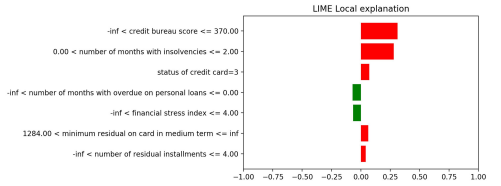
We test LIME on several data points, with the purpose of understanding the logic hidden into the Gradient Boosting model employed. We report its explanations in Table 2, for one good user (on the left) and one bad (on the right).

In order to check its stability we employ the technique several times on the same unit and compare the results, using the above mentioned stability indices.
In this particular application, we apply LIME 10 times, giving rise to 10 distinct Local Ridge Regression models. Only the 7 most important features are considered in the explanation.

In Table 2 the VSI and CSI indices are shown for the two different individuals

Table 2. LIME applied to Gradient Boosting model.

The sum of the bars' values, along with the intercept, produces the Local Ridge model prediction (denoted as LIME Prediction). The bars' length highlight the specific contribution of each variable: the green ones push the model towards "good payer" prediction, whereas the red ones to "bad payer". The stability indices for LIME, evaluated on repeated calls, are satisfactory.

Unit Number: 1 GBM Prediction: 0.059 LIME Prediction: 0.054 Intercept: 1.104 VSI: 100% CSI: 99.13%	Unit Number: 53 GBM Prediction: 0.956 LIME Prediction: 1.031 Intercept: 0.424 VSI: 97.14% CSI: 99.49%
	
	
	

explained by LIME. Since we achieve high values of the indices, we consider LIME to be stable among different calls, for the chosen individuals.

Moreover, LIME explanations make sense from an economic and financial standpoint: in Table 2 the key regressors are the Credit Bureau Score (CBS), namely a comprehensive value developed using information provided by the Italian Credit Bureau, and the number of months where unpaid instalments occurred, within the last year.

On the left part, the user exhibits 0 months with unpaid instalments and falls inside a good class of CBS index. Such circumstances are the major ones leading Gradient Boosting model to classify him as a good payer.

On the contrary, the user shown on the right displays at least one month with insolvencies and falls inside a bad CBS class, these conditions drive the model to classify him as bad payer.

6. Discussion and conclusions

Often Machine Learning models produce more accurate predictions compared to classical models: there was evidence of such trend also in the use case presented above, regarding the CRM field. Credit Risk would therefore benefit from the employment of such more powerful techniques.

Regrettably, to date there is no methodology allowing unambiguous explanations of Machine Learning models: in recent years, a number of methods have been proposed, but their consistency and reliability is still a discussion topic.

We focus on the LIME technique and apply it successfully to Credit Risk data. Digging further into the method, we try to establish whether LIME is stable, namely if repeated calls of the method, on the same individual, result in very close explanations.

To this end, we derive the distribution of the local model coefficients, used by LIME under the hood. Building on top of the coefficients' distribution, we create a stability index, which evaluates whether the coefficients for the same variable among different LIME calls are similar (CSI). Meanwhile, we monitor whether the variables returned by different LIME calls are the same. This is done using another index: VSI. The two complementary indices both range from 0 to 100, where higher values correspond to an higher degree of stability.

When used together, they provide useful insights to the practitioner about the consistency of the trained LIME method: they help understand whether LIME is likely to modify its output at the next call.

We consider it an important step: it improves the trust in LIME as a reliable explanation method and it goes towards meeting the regulator's requests in the CRM field.

However, such result just ensures LIME is concordant among different applications: the model may still return explanations not really close to the Machine Learning model. More research still needs to be done in this direction.

Acknowledgements

We would like to thank professor Giuliano Galimberti who provided insight and expertise that greatly assisted the research, although he may not agree with all of the interpretations/conclusions of this paper.

Funding

We acknowledge financial support by CRIF S.p.A. and Universit degli Studi di Bologna.

References

Agresti, A. (2015). *Foundations of linear and generalized linear models.* John Wiley & Sons.

Beyer, K., Goldstein, J., Ramakrishnan, R., & Shaft, U. (1999). When is nearest neighbor meaningful? In (pp. 217–235). Springer.

Billingsley, P. (2008). *Probability and measure*. John Wiley & Sons.

Brame, R., Paternoster, R., Mazerolle, P., & Piquero, A. (1998). Testing for the equality of maximum-likelihood regression coefficients between two independent equations. *Journal of Quantitative Criminology*, *14*(3), 245–261.

European Banking Authority (EBA). (2020). *Report on big data and advanced analytics.* Retrieved from `https://eba.europa.eu/eba-report-identifies-key-challenges-roll-out-big-data-and-advanced-analytics`

Greene, W. H. (2003). *Econometric analysis*. Pearson Education India.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, *51*(5), 93.

Hall, P., & Gill, N. (2018). *An introduction to machine learning interpretability-dataiku version*. O'Reilly Media, Incorporated.

Hand, D. J. (2001). Modelling consumer credit risk. *IMA Journal of Management mathematics*, *12*(2), 139–155.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

HLEG, A. (2019). *Ethics guidelines for trustworthy AI.* Retrieved from `https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai`

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67.

Johnston, J., & DiNardo, J. (1972). *Econometric methods* (Vol. 2). New York.

Kingston, J. (2017). Using artificial intelligence to support compliance with the general data protection regulation. *Artificial Intelligence and Law*, *25*(4), 429–443.

Mitchell, T. M. (1997). Machine learning.

Moscatelli, M., Narizzano, S., Parlapiano, F., & Viggiano, G. (2019). Corporate default forecasting with machine learning.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). Why should i trust you?: Explaining the predictions of any classifier. In (pp. 1135–1144). ACM.

Shankaranarayana, S. M., & Runje, D. (2019). ALIME: Autoencoder based approach for local interpretability. In *International conference on intelligent data engineering and automated learning* (pp. 454–463). Springer.

van Wieringen, W. N. (2015). Lecture notes on ridge regression. *arXiv preprint arXiv:1509.09169*.