algorithms

Article

A Model-Agnostic Algorithm for Bayes Error Determination in Binary Classification

Umberto Michelucci [1,*], Michela Sperti [2], Dario Piga [3], Francesca Venturini [1,4] and Marco A. Deriu [2]

1 TOELT LLC, Machine Learning Research and Development, Birchlenstr. 25, 8600 Dübendorf, Switzerland; vent@zhaw.ch
2 PolitoBIOMed Lab, Department of Mechanical and Aerospace Engineering, Politecnico di Torino, 10129 Turin, Italy; michela.sperti@polito.it (M.S.); marco.deriu@polito.it (M.A.D.)
3 IDSIA—Dalle Molle Institute for Artificial Intelligence, USI-SUPSI, Via la Santa 1, 6962 Lugano, Switzerland; dario.piga@supsi.ch
4 Institute of Applied Mathematics and Physics, Zurich University of Applied Sciences, Technikumstrasse 9, 8401 Winterthur, Switzerland
* Correspondence: umberto.michelucci@toelt.ai

Abstract: This paper presents the intrinsic limit determination algorithm (ILD Algorithm), a novel technique to determine the best possible performance, measured in terms of the AUC (area under the ROC curve) and accuracy, that can be obtained from a specific dataset in a binary classification problem with categorical features regardless of the model used. This limit, namely, the Bayes error, is completely independent of any model used and describes an intrinsic property of the dataset. The ILD algorithm thus provides important information regarding the prediction limits of any binary classification algorithm when applied to the considered dataset. In this paper, the algorithm is described in detail, its entire mathematical framework is presented and the pseudocode is given to facilitate its implementation. Finally, an example with a real dataset is given.

Keywords: machine learning; intrinsic limits; ROC curve; binary classification; area under the curve; Naïve Bayes classifier

check for updates

Citation: Michelucci, U.; Sperti, M.; Piga, D.; Venturini, F.; Deriu, M.A. A Model-Agnostic Algorithm for Bayes Error Determination in Binary Classification. *Algorithms* **2021**, *14*, 301. https://doi.org/10.3390/a14110301

Academic Editor: Frank Werner

Received: 29 September 2021
Accepted: 15 October 2021
Published: 20 October 2021

1. Introduction

The majority of machine learning projects tend to follow the same pattern, namely, many different machine learning model types (such as decision trees, logistic regression, random forest, neural network, etc.) are first trained from data to predict specific outcomes, and then tested and compared to find the one that gives the best prediction performance on validation data. Many techniques to compare models have been developed and are commonly used in several settings [1–3]. For some specific model types, such as neural networks, it is difficult to know when to stop the search [4]. There is always the hope that a different set of hyperparameters, as the number of layers, or a better optimizer, will give a better performance [4,5]. This makes the model comparison laborious and time-consuming.

Many reasons may lead to a bad accuracy: deficiencies in the classifier, overlapping class densities [6,7], noise affecting the data [8–10], and limitations in the training data being the most important [11]. Classifier deficiencies could be addressed by building better models, of course, but other types of errors linked with the data (for example, mislabeled patterns or missing relevant features) will lead to an error that cannot be reduced by any optimization in the model training, regardless of the effort invested. This error is also known in the literature as Bayes error (BE) [4,12]. The BE can, in theory, be obtained from the Bayes theorem if one would know all density probabilities exactly. However, this is impossible in all real-life scenarios, and thus the BE cannot be computed directly from the data in all non-trivial cases. The Naïve Bayes classifier [12] tries to approximately address this problem, but it is based on the assumption of the conditional independence of the features, rarely satisfied in practice [11–13]. The methods to estimate the BE developed in the past decade tend to follow the same strategy: reduce the error linked to the classifier as much as possible, thus being left with only the BE. Ensemble strategies and meta learners [11,13,14] have been widely used to address this problem. The idea is to exploit the multiple predictions to provide an indication of the limits to the performance for a given dataset [11]. This approach has been widely used with neural networks, given their universal approximator nature [15,16].

In any supervised learning task, knowing the BE linked to a given dataset would be of extreme importance. Such a value would help practitioners decide whether or not it is worthwhile to spend time and computing resources in improving the developed classifiers or acquiring additional training data. Even more importantly, knowing the BE would let practitioners assess if the available features in a dataset are useful for a specific goal. Suppose for example that a set of clinical exams are available for a large number of patients. If such a feature set gives a BE of 30% (so an accuracy of 70%) in predicting an outcome but the desired BE is smaller than 20%, it is useless to spend time in developing models. Therefore, time would be better spent in acquiring additional features. The problem of determining the BE intrinsic of a given dataset is addressed and solved in this work from a theoretical point of view. The problem of determining the BE of a given dataset is addressed in this paper.

The contribution of this paper is twofold. First, a new algorithm, called Intrinsic Limit Determination algorithm (ILD algorithm), is presented. The ILD algorithm allows computing the maximum performance in a binary classification problem, expressed both as the largest area under the ROC curve (AUC) and as the accuracy that can be achieved with any given dataset with categorical features. This is by far the most significant contribution of this paper, as the ILD algorithm for the first time allows evaluating the BE for a given dataset exactly. This paper demonstrates how the BE is a limit not dependent on any chosen model but is an inherent property of the dataset itself. Thus, the ILD algorithm gives the upper limit of the prediction possibilities of any possible model when applied to a given dataset, under the conditions that the features are categorical and that the target variable is binary. Second, the mathematical framework on which the ILD algorithm is based is discussed and a mathematical proof of the algorithm validity is given. The algorithm's computational complexity is also discussed.

Although the applicability conditions may seem greatly limiting, there are a large number of cases where the ILD can be applied. Consider for example medical datasets, that typically have a large portion of features that are categorical (i.e., presence of absence of certain conditions) [17–21] Many of the continuous features (like age for example) are usually transformed into categorical (age ranges for example). Therefore, in all these cases, the ILD algorithm will give very important information to doctors on the datasets themselves.

The paper is organized as follows. The necessary notation and dataset restructuring for the ILD algorithm are discussed in Section 2. In Section 3, the complete mathematical formalism necessary for the ILD algorithm is explained in detail and in Section 4 the fundamental ILD Theorem is given and proof is presented. Application of the ILD algorithm to a real dataset is provided in Section 5. Finally, in Section 6 conclusions and promising research developments are discussed.

2. Mathematical Notation and Dataset Aggregation

Let us consider a dataset with N categorical features, i.e., each feature can only assume a finite set of values. Let us suppose that the ith feature, denoted as F_i, takes n_i possible values. For notational simplicity, it is assumed that the categorical feature is encoded in such a way that its possible values are integers from 1 to n_i, with $i = 1, \dots, N$ (note that each n_i can assume different integer values). Each possible combination of the features is called here a *bucket*. The idea is that the observations will be aggregated in buckets depending on their features. The number of observations present in the dataset are indicated with M. All the observations with the same set of features are said to be in the same bucket.

The problem is a binary classification one, aiming at predicting an event that can have only two possible outcomes, indicated here with class 0 and class 1. In general, in the jth bucket, there will be a certain number of observations (that we will indicate with $m_0^{[j]}$) with a class of 0, and a certain number of observations (that we will indicate with $m_1^{[j]}$) with a class of 1.

The feature vector for each observation, denoted as x_k (with $k = 1, \ldots, M$), is thus defined by an N-dimensional vector $(F_{1,k}, F_{2,k}, \ldots, F_{N,k}) \in \mathbb{N}^N$, where $F_{i,k}$ denotes the value of the i-th feature of the k-th sample.

The following useful definitions are now introduced.

Definition 1. *A feature bucket $B^{[j]}$ is a possible combination of the values of the N features, i.e.,*

$$B^{[j]} = \{(F_1^{[j]}, F_2^{[j]}, \ldots, F_N^{[j]}) \in \mathbb{N}^N \,|\, F_i^{[j]} \in \{0, 1, \ldots, n_i - 1\}, i = 1, \ldots, N\} \tag{1}$$

In the rest of the paper, the feature bucket is indicated as

$$B^{[j]} = (F_1^{[j]}, F_2^{[j]}, \ldots, F_N^{[j]})$$

to explicitly mention the feature values characterizing the bucket $B^{[j]}$. The total number N_B of feature buckets is thus equal to $N_B = \prod_{i=1}^{N} n_i$.

As an example, in the case of two binary features F_1 and F_2, four possible feature buckets can be constructed, namely, $B^{[1]} = (0, 0)$, $B^{[2]} = (0, 1)$, $B^{[3]} = (1, 0)$ and $B^{[4]} = (1, 1)$.

Definition 2. *The set $M^{[j]}$ of the indices of observations belonging to the j-th feature bucket $B^{[j]}$ is defined as*

$$M^{[j]} = \{k \in [1,\ M] \,|\, F_{0,k} = F_0^{[j]} \textit{ and } F_{1,k} = F_1^{[j]} \textit{ and} \ldots\ F_{N,k} = F_N^{[j]}\} \tag{2}$$

The cardinality of the set $M^{[j]}$ will be denoted as $|M^{[j]}|$. In a binary classification problem, the observations belonging to the feature bucket $B^{[j]}$, denoted as

$$O^{[j]} = \{x_k \,|\, k \in M^{[j]}\} \tag{3}$$

will contain $m_0^{[j]} \in \mathbb{N}_0$ observations with a target variable equal to 0 and $m_1^{j]} \in \mathbb{N}_0$ observations with a target variable equal to 1. Note that by definition

$$|M^{[j]}| = m_0^{[j]} + m_1^{[j]} \tag{4}$$

and

$$M \equiv \sum_{j=1}^{N_B} |M^{[j]}| = \sum_{j=1}^{N_B} (m_0^{[j]} + m_1^{[j]}) \tag{5}$$

Based on the definitions above, the original dataset can be equivalently represented by a new dataset of buckets, an *aggregated dataset B*, each of which contains a certain number of samples $|M^{[j]}| = m_0^{[j]} + m_1^{[j]}$. A visual representation of the previously described rearrangement of the original dataset is reported in Figure 1 for a dataset with two binary features.

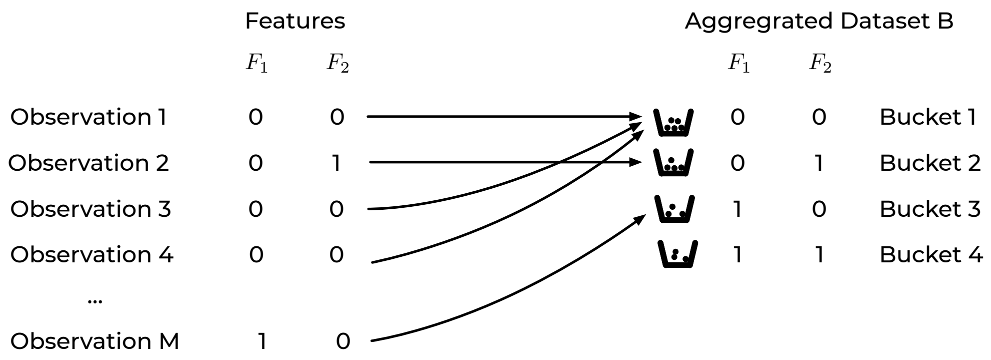


Figure 1. An intuitive representation of the dataset aggregation step for a dataset with two binary features F_1 and F_2. Observations with, for example, $F_1 = 0$ and $F_2 = 0$ will be in bucket 1 in the aggregated dataset B. Features with $F_1 = 0$ and $F_2 = 1$ in bucket 2 and so on.

In the aggregated dataset B, each record is thus a feature bucket $B^{[j]}$, characterized by the number of observations of class 0 ($m_0^{[j]}$) and the number of observations of class 1 ($m_1^{[j]}$). In the previous example of a dataset with only two binary features, B would look like the one in Table 1. In this easy example a dataset with any number of observations M would be reduced to one with only 4 records, i.e., the number of possible buckets.

Table 1. An example of an aggregated dataset B with only two binary features.

Bucket	**Feature 1**	**Feature 2**	**Class 0**	**Class 1**
1	$F_1^{[1]} = 0$	$F_2^{[1]} = 0$	$m_0^{[1]}$	$m_1^{[1]}$
2	$F_1^{[2]} = 0$	$F_2^{[2]} = 1$	$m_0^{[2]}$	$m_1^{[2]}$
3	$F_1^{[3]} = 1$	$F_2^{[3]} = 0$	$m_0^{[3]}$	$m_1^{[3]}$
4	$F_1^{[4]} = 1$	$F_2^{[4]} = 1$	$m_0^{[4]}$	$m_1^{[4]}$

With this new representation of the dataset, generated by aggregating all observations sharing the same feature values in buckets, the proposed ILD algorithm allows computing the best possible ROC curve considering *all possible* predictions.

3. ILD Algorithm Mathematical Framework

As the output of any machine learning model is a function of the feature values, and as a bucket is a collection of observations all with the same feature values, any possible deterministic model will associate to the jth bucket of features only one possible class prediction $p^{[j]}$ that can be either 0 or 1. More in general, to each model can be associated a prediction vector $\mathbf{p}= (p^{[1]}, p^{[2]}, \dots, p^{[N_B]})$. In the next sections important quantities (as TPR and FPR) evaluated for the aggregated dataset B as functions of $m_0^{[i]}$, $m_1^{[i]}$ and $\mathbf{p}$ are derived.

3.1. True Positive, True Negative, False Positive, False Negative

In the feature bucket i, if $p^{[i]} = 0$, then only $m_0^{[i]}$ observations would be correctly classified. On the other side, if $p^{[i]} = 1$, only $m_1^{[i]}$ observations would be correctly classified. For each bucket i the true positive $TP^{[i]}$ can be written as

$$TP^{[i]} = m_1^{[i]} p^{[i]} \tag{6}$$

In fact, if $p^{[i]} = 0$, then $TP^{[i]} = 0$, and if $p^{[i]} = 1$, then $TP^{[i]} = m_1^{[i]}$. Considering the entire dataset, the true positive, true negative (TN), false positive (FP), false negative (FN) are given by

$$\begin{aligned} TP &= \sum_{i=1}^{N_B} TP^{[i]} = \sum_{i=1}^{N_B} m_1^{[i]} p^{[i]} && (7) \\ TN &= \sum_{i=1}^{N_B} TN^{[i]} = \sum_{i=1}^{N_B} m_0^{[i]} (1 - p^{[i]}) && (8) \\ FP &= \sum_{i=1}^{N_B} FP^{[i]} = \sum_{i=1}^{N_B} m_0^{[i]} p^{[i]} && (9) \\ FN &= \sum_{i=1}^{N_B} FN^{[i]} = \sum_{i=1}^{N_B} m_1^{[i]} (1 - p^{[i]}) && (10) \end{aligned}$$

where the sums are performed over all the N_B buckets.

3.2. Accuracy

In a binary classification problem, the accuracy is defined [22] as

$$a = \frac{TP + TN}{M}. \tag{11}$$

Using Equations (7) and (8), the accuracy can be rewritten as

$$a = \frac{1}{M} \sum_{i=1}^{N_B} \left[p^{[i]} (m_1^{[i]} - m_0^{[i]}) + m_0^{[i]} \right] \tag{12}$$

The maximum value of the accuracy is obtained if the model predicts $p^{[i]} = 1$ as soon as $m_1^{[i]} > m_0^{[i]}$. This can be stated as

Theorem 1. *The accuracy for an aggregated categorical dataset B, expressed as Equation (12), is maximized by choosing* $p^{[i]} = 1$ *when* $m_1^{[i]} > m_0^{[i]}$ *and* $p^{[i]} = 0$ *when* $m_1^{[i]} \leq m_0^{[i]}$.

Proof. The proof is given by considering each bucket separately. Let's consider a bucket i that has $m_1^{[i]} > m_0^{[i]}$. In this case, there are two possibilities:

$$\begin{aligned} p^{[i]} = 1 \rightarrow \quad & p^{[i]} (m_1^{[i]} - m_0^{[i]}) + m_0^{[i]} = m_1^{[i]} \\ p^{[i]} = 0 \rightarrow \quad & p^{[i]} (m_1^{[i]} - m_0^{[i]}) + m_0^{[i]} = m_0^{[i]} \end{aligned} \tag{13}$$

Therefore, the ith contribution to the accuracy in Equation (12) is maximized by choosing $p^{[i]} = 1$ for those buckets where $m_1^{[i]} \geq m_0^{[i]}$. With a similar reasoning, the contribution to the accuracy for those buckets where $m_1^{[i]} < m_0^{[i]}$ is maximized by choosing $p^{[i]} = 0$. This concludes the proof. □

3.3. Sensitivity and Specificity

The sensitivity or true positive rate (TPR) is the ability to correctly predict the positive cases [22]. Considering the entire dataset, the TPR can be expressed using Equations (7) and (10) as

$$TPR = \frac{TP}{TP + FN} = \frac{\sum_{i=1}^{N_B} m_1^{[i]} p^{[i]}}{\sum_{i=1}^{N_B} \left[m_1^{[i]} p^{[i]} + m_1^{[i]} (1 - p^{[i]}) \right]} = \frac{\sum_{i=1}^{N_B} m_1^{[i]} p^{[i]}}{\sum_{i=1}^{N_B} m_1^{[i]}} \tag{14}$$

Analogously, the specificity or true negative rate (TNR), which is the ability to correctly reject the negative cases [22], can be written using Equations (8) and (9) as

$$TNR = \frac{TN}{TN + FP} = \frac{\sum_{i=1}^{N_B} m_0^{[i]}(1 - p^{[i]})}{\sum_{i=1}^{N_B} \left[m_0^{[i]}(1 - p^{[i]}) + m_0^{[i]} p^{[i]} \right]} = \frac{\sum_{i=1}^{N_B} m_0^{[i]}(1 - p^{[i]})}{\sum_{i=1}^{N_B} m_0^{[i]}} \tag{15}$$

3.4. ROC Curve

The receiver operating characteristic (ROC) curve is built by plotting the true positive rate TPR on the y-axis, and the false positive rate (FPR) on the x-axis. For completeness, the $FPR = 1 - TNR$ is

$$FPR = 1 - TNR = \frac{\sum_{i=1}^{N_B} \left[m_0^{[i]} - m_0^{[i]}(1 - p^{[i]}) \right]}{\sum_{i=1}^{N_B} m_0^{[i]}} = \frac{\sum_{i=1}^{N_B} m_0^{[i]} p^{[i]}}{\sum_{i=1}^{N_B} m_0^{[i]}} \tag{16}$$

3.5. Perfect Bucket and Perfect Dataset

Sometimes a bucket may contain only observations that are all in class 0 or 1. Such a bucket is called in this paper *perfect bucket* and is defined as follows.

Definition 3. *A feature bucket j is called* perfect *if one of the following is satisfied:*

$$\begin{cases} m_0^{[j]} = 0 \\ m_1^{[j]} > 0 \end{cases} \tag{17}$$

or

$$\begin{cases} m_0^{[j]} > 0 \\ m_1^{[j]} = 0 \end{cases} \tag{18}$$

It is also useful to define the set of all perfect buckets P.

Definition 4. *The set of all perfect buckets, indicated with P is defined by*

$$P \equiv \{B_j \; j = 1, \ldots, N_B \, | (m_0^{[j]} = 0 \, and \, m_1^{[j]} > 0) \, or \, (m_0^{[j]} > 0 \, and \, m_1^{[j]} = 0)\} \tag{19}$$

Note that by definition, the set B/P contains only *imperfect* buckets, namely, buckets where $m_0^{[j]} > 0$ and $m_1^{[j]} > 0$.

An important special case is that of a *perfect dataset*, one in which all buckets are perfect. Indicating with B the set containing all buckets, we have $B = P$. It is easy to see that we can create a prediction vector that will predict all cases perfectly, by simply choosing, for feature bucket j

$$p^{[j]} = \begin{cases} 0 \text{ if } m_0^{[j]} > 0 \\ 1 \text{ if } m_1^{[j]} > 0. \end{cases} \tag{20}$$

Remember that all feature buckets are perfect, and in a perfect feature bucket $m_0^{[j]}$ and $m_1^{[j]}$ cannot be greater than zero at the same time. To summaries our definitions we can define the following.

Definition 5. *A dataset B (where B is the dataset containing all feature buckets) is called perfect if $B = P$.*

Definition 6. *A dataset B (where B is the dataset containing all feature buckets) is called imperfect if $P \subset B$.*

4. The Intrinsic Limit Determination Algorithm

Let us introduce the predictor vector $\mathbf{p}_S \equiv (1, 1, \ldots, 1)$, for which it clearly holds

$$\begin{cases} TPR|_{\mathbf{p}_S} = & 1 \\ FPR|_{\mathbf{p}_S} = & 1 \end{cases} \tag{21}$$

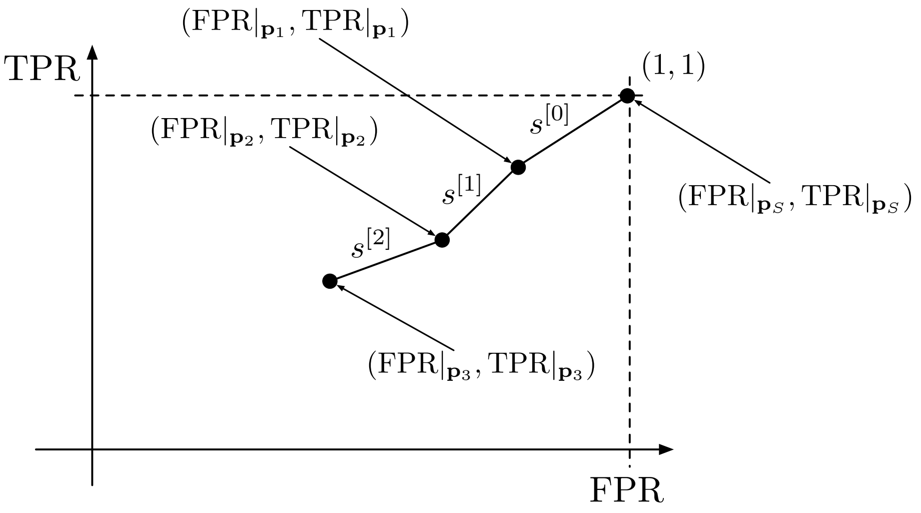
$TPR|_{\mathbf{p}_S}$ and $FPR|_{\mathbf{p}_S}$ indicate TPR and FPR evaluated for $\mathbf{p}_S$, respectively.

Let us indicate as a *flip* the change of a component of a prediction vector from the value of 1 to the value of 0. Any possible prediction vector can thus be obtained by a finite series of flips starting from $\mathbf{p}_S$, where a flip is done only on components with a value equal to 1. Let us denote with $\mathbf{p}_1$ the prediction vector obtained after the first flip, $\mathbf{p}_2$ after the second, and so on. After N_B flips, the prediction vector will be $\mathbf{p}_{N_B} \equiv (0, 0, \ldots, 0)$. The TPR and FPR evaluated for a prediction vector $\mathbf{p}_i$ (with $i = 1, \ldots, N_B$) are indicated as $TPR|_{\mathbf{p}_i}$ and $FPR|_{\mathbf{p}_i}$. The set of tuples of points' coordinates is indicated with $\mathcal{P}$:

$$\mathcal{P} = \{(FPR|_{\mathbf{p}_S}, TPR|_{\mathbf{p}_S}), \ldots, (FPR|_{\mathbf{p}_{N_B}}, TPR|_{\mathbf{p}_{N_B}})\}. \tag{22}$$

A curve can be constructed by joining the points contained in $\mathcal{P}$ in *ordered segments*, where ordered segments means that the point $(FPR|_{\mathbf{p}_S}, TPR|_{\mathbf{p}_S})$ will be connected to $(FPR|_{\mathbf{p}_1}, TPR|_{\mathbf{p}_1})$; $(FPR|_{\mathbf{p}_1}, TPR|_{\mathbf{p}_1})$ with $(FPR|_{\mathbf{p}_2}, TPR|_{\mathbf{p}_2})$; and so on. The segment that joins the point $(FPR|_{\mathbf{p}_S}, TPR|_{\mathbf{p}_S})$ with $(FPR|_{\mathbf{p}_1}, TPR|_{\mathbf{p}_1})$ is denoted as $s^{[0]}$; the one that joins the points $(FPR|_{\mathbf{p}_1}, TPR|_{\mathbf{p}_1})$ and $(FPR|_{\mathbf{p}_2}, TPR|_{\mathbf{p}_2})$ is denoted as $s^{[1]}$, and so on. In Figure 2, a curve obtained by joining the tuples given by the respective prediction vectors obtained with three flips is visualized to give an intuitive understanding of the process.

The ILD algorithm provides a constructive process to select the order of the components to be flipped to obtain the curve $\tilde{C}$ characterized by the theoretical maximum AUC that can be obtained from the considered dataset, regardless of the predictive model used.



$(\mathrm{FPR}|_{\mathbf{p}_1}, \mathrm{TPR}|_{\mathbf{p}_1})$
TPR
$(1, 1)$
$s^{[0]}$
$(\mathrm{FPR}|_{\mathbf{p}_2}, \mathrm{TPR}|_{\mathbf{p}_2})$
$s^{[1]}$
$(\mathrm{FPR}|_{\mathbf{p}_S}, \mathrm{TPR}|_{\mathbf{p}_S})$
$s^{[2]}$
$(\mathrm{FPR}|_{\mathbf{p}_3}, \mathrm{TPR}|_{\mathbf{p}_3})$
FPR

Figure 2. Example of a curve constructed by joining 3 segments obtained after 3 flips.

To be able to describe the ILD algorithm effectively and prove its validity, some additional concepts are needed and described in the following paragraphs.

4.1. Effect of One Single Flip

Let us consider what happens if one single component, say the jth component, of $\mathbf{p}_S$ is changed from 1 to 0. The TPR and FPR values clearly change. By denoting with $\mathbf{p}_1$ the prediction vector in which the jth component was changed, the following equations hold:

$$\begin{cases} TPR|_{\mathbf{p}_1} = & 1 - \dfrac{m_1^{[j]}}{\sum\limits_{i=1}^{N_B} m_1^{[i]}} \\ FPR|_{\mathbf{p}_1} = & 1 - \dfrac{m_0^{[j]}}{\sum\limits_{i=1}^{N_B} m_0^{[i]}}. \end{cases} \tag{23}$$

Therefore, TPR and FPR will be reduced by an amount equal to the ratio $m_1^{[j]} / \sum_{i=1}^{N_B} m_1^{[i]}$ and $m_0^{[j]} / \sum_{i=1}^{N_B} m_0^{[i]}$, respectively. As an example, the effect of multiple single flips on TPR and FPR is illustrated in Figure 3. Here are shown the ROC curves resulting from a random flipping starting from $\mathbf{p}_S$ for a real-life dataset, namely, the Framingham dataset [23] (see Section 5). As expected, flipping components randomly results in a curve that lies close to the diagonal. As the diagonal corresponds to randomly assigning classes to the observations, randomly flipping does not bring to the best prediction possible with the given dataset.

By ordering the points in $\mathcal{P}$ in ascending order based on the ratio $TPR|_{\mathbf{p}_j} / FPR|_{\mathbf{p}_j}$, a new set of points $\tilde{\mathcal{P}}$ is constructed. It can happen that in a given dataset, multiple points have $FPR|_{\mathbf{p}_j} = 0$. In this case, this ratio cannot be calculated. If this happens, all the points with $FPR|_{\mathbf{p}_j} = 0$ can be placed at the end of the list of points. The order between those points is irrelevant. It is interesting to note that a flip for perfect buckets for which $m_0^{[j]} = 0$ will have $TPR|_{\mathbf{p}_j} = 0$ and for all perfect buckets for which $m_1^{[j]} = 0$ will have $FPR|_{\mathbf{p}_j} = 0$.

With the set of ordered points $\tilde{\mathcal{P}}$, a curve $\tilde{\mathcal{C}}$ can be constructed by joining the points in $\tilde{\mathcal{P}}$ as described in the previous paragraph. Note that the relative order of all points with $TPR|_{\mathbf{p}_j} = 0$ is also irrelevant, in the sense that this order does not affect the AUC of $\tilde{\mathcal{C}}$.

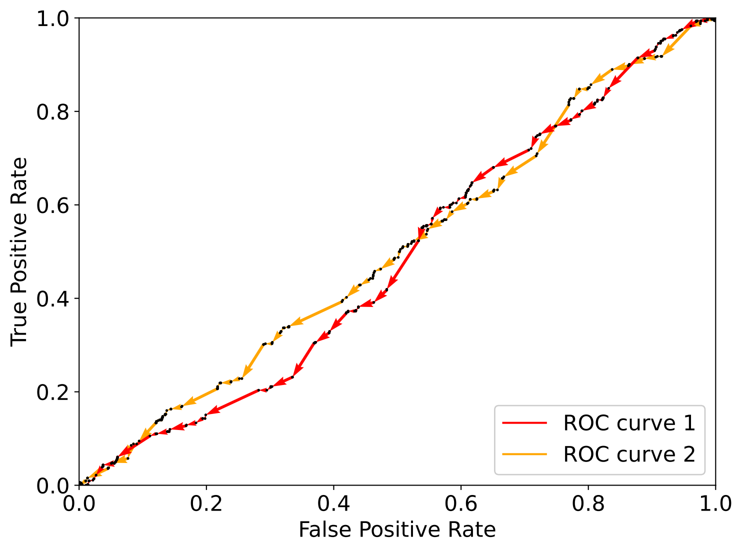


Figure 3. Two examples of ROC curves obtained from random flipping using the dataset as described in the text.

4.2. ILD Theorem

The ILD theorem can now be formulated.

Theorem 2 (ILD Theorem). *Among all possible curves that can be constructed by generating a set of points $\mathcal{P}$ by flipping all components of $\mathbf{p}_S$ in any order one at a time, the curve $\tilde{C}$ has the maximum AUC.*

Proof. The Theorem is proven by giving a construction algorithm. It starts with one set of points $\mathcal{P}_0$ generated by flipping components of $\mathbf{p}_S$ in a random order. Let us consider two adjacent segments generated with $\mathcal{P}_0$: $s^{[j]}$ and $s^{[j+1]}$. In Figure 4A, the two segments are plotted in the case where the angle between them $\beta^{[j,j+1]} < \pi$. The angles $\alpha^{[j]}$ and $\alpha^{[j+1]}$ indicate the angles of the segments with the horizontal direction and $\beta^{[j,j+1]}$ the angle between the two segments j and $j+1$. The area under the two segments and any horizontal line that lies below the segments can be increased by simply switching the order of the two flips, as it is depicted in Figure 4B. Switching the order simply means flipping first the $j+1$ component and then the j component.

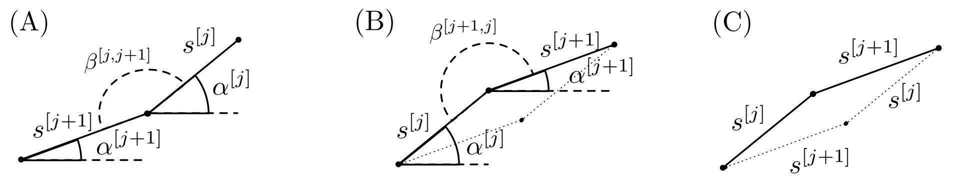


Figure 4. Visual explanation for the ILD Theorem. Panel (**A**): two consecutive segments after flipping components j and then $j+1$; Panel (**B**): two consecutive segments after flipping components $j+1$ and then j; Panel (**C**): parallelogram representing the difference between the area under the segments in panel (**A**) and the area under the segments in panel (**B**).

It is important to note that in Figure 4A $\beta^{[j,j+1]} < \pi$, while in panel (B) $\beta^{[j+1,j]} > \pi$. It is evident that the area under the two segments in panel (B) is greater than the one in panel (A). The parallelogram in Figure 4C depicts the area difference.

The proof is based on repeating the previous step until all angles $\beta^{[j,j+1]} > \pi$. This is described in pseudocode in Algorithm 1.

Algorithm 1: Algorithm to construct the curve $\tilde{C}$

```
Generate a set of points $\mathcal{P}_0$ by flipping the components of $\mathbf{p}_S$ in a random
  sequence;
while true do
    $c = 0$;
    for $i = 1, \ldots, N_B$ do
        if $\beta^{[i,i+1]} < \pi$ then
            Switch points $i$ and $(i+1)$ in $\mathcal{P}_0$;
            c = c + 1;
        end
    end
    if $c = 0$ then
        Exit While loop and end the Algorithm
    end
    The final set of points $\mathcal{P}_0$ obtained in the loop above will generate the curve $\tilde{C}$.
end
```

The area under the curve obtained with Algorithm 1 cannot be made larger with any further switch of points in $\mathcal{P}$. □

Note that Algorithm 1 will end after a finite number of steps. This can be shown by noting that the described algorithm is nothing else than the bubble sort algorithm [24] applied to the set of angles $\alpha^{[1]}, \alpha^{[2]}, \ldots, \alpha^{[N_B]}$. Therefore, this algorithm has a worst-case and average complexity of $\mathcal{O}(N_B^2)$.

4.3. Handling Missing Values

Missing values can be handled by imputing them with a value that does not appear in any feature. All observations that have missing values in a specific feature will be assigned to the same bucket and considered similar, as we have no way of knowing better.

5. Application of the ILD Algorithm to the Framingham Heart Study Dataset

In this section, a practical application of the ILD algorithm is illustrated. The application to a real dataset has the purpose of giving an intuitive understanding of the method and highlight its power. Here, the medical dataset named Framingham [23] were chosen, which is publicly available on the Kaggle website [25]. This dataset comes from an ongoing cardiovascular risk study made on residents of the town of Framingham (Framingham, MA, USA). Different cardiovascular risk score versions were developed during the years [26], the most current of whom is the 2008 study in [27], to which the ILD algorithm results are also referred for comparison of performances. The reader interested in descriptive statistics on the dataset's features (averages, standard deviations, counts, description, etc.) can find all information in [23].

The classification goal is to predict, given a series of risk factors, the 10-years risk of a patient of future coronary heart disease. This is a high impact task, as 17.9 million deaths occur worldwide every year due to heart diseases [28] and their early prognosis may be of crucial importance for a correct and successful treatment. Note that the dataset available in [25] has 16 features. To make the comparison as meaningful as possible, seven of the eight features used in the original study [23] were selected. The high-density lipoprotein (HDL) cholesterol variable is unfortunately missing with respect to the original Framingham dataset employed in [27] and therefore could not be used. Thus, the dataset used in this study contains 4238 patients and 7 features: gender (0: female, 1: male); smoker (0: no, 1: yes); diabetes (0: no, 1: yes); hypertension treatment (0: no, 1: yes); age; total cholesterol; and systolic blood pressure (SBP). The last three features are continuous variables and were discretized as follows:

- age: 0 if age < 40 years, 1 if age ≥ 40 years and age < 60 years,
  2 if age ≥ 60 years;
- total cholesterol: 0 if total cholesterol < 200 mg/dL,
  1 if total cholesterol ≥ 200 mg/dL and total cholesterol < 240 mg/dL,
  2 if total cholesterol ≥ 240 mg/dL;
- SBP: 0 if SBP < 120 mmHg, 1 if SBP ≥ 120 mmHg.

The outcome variable is binary (0: no coronary disease, 1: coronary disease). Finally, to create the buckets all missing values are substituted by a feature value of -1.

The application of the algorithm starts with the population of the buckets as described in the previous sections. A total number of 177 buckets are populated. The dataset is split into two parts: a training set S_T (80% of the data) and a validation one S_V (20% of the data). For comparison, the Naïve Bayes classifier is also trained and validated respectively on S_T and S_V. The performance of the ILD algorithm and the Naïve Bayes classifier are shown through the ROC curve in Figure 5. The AUC for the ILD algorithm is 0.78, clearly higher than that for the Naïve Bayes classifier, namely, 0.68.

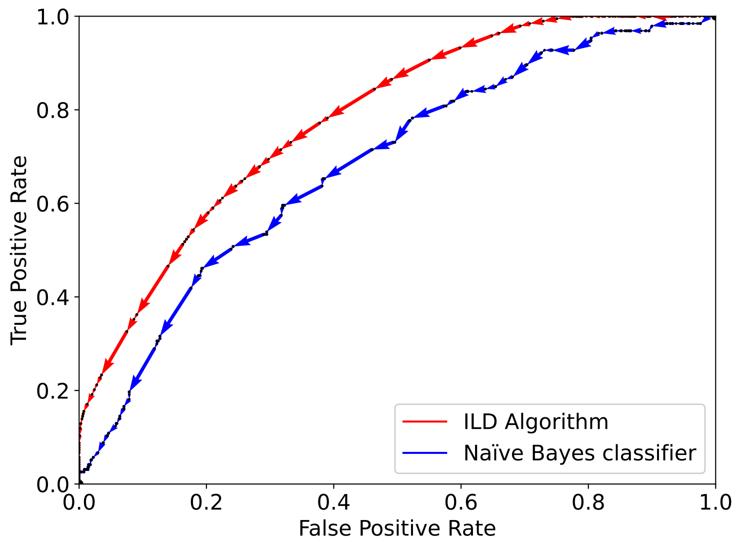


Figure 5. Comparison of the performance of the ILD algorithm (red) and Naïve Bayes classifier (sblue) implemented on categorical features based on one single training and validation split using the dataset as described in the text.

To further test the performance, the split and training is repeated for 100 different dataset splits. Each time both the ILD algorithm and the Naïve Bayes classifier are applied to the validation set S_V and the resulting AUCs are plotted in Figure 6 (top panel). For clarity, the difference between the AUC provided by the two algorithms is shown in Figure 6 (bottom panel).

This example shows that the application of the ILD algorithm allows the comparison of the prediction performance of a model, here the Naïve Bayes classifier, with the maximum obtainable for a given dataset. The maximum accuracy over the validation set S_V is 85% for the Naïve Bayes classifier (calculated for a specific threshold, i.e., 61%, which optimizes the accuracy over the training set S_T) and 86% for the ILD algorithm (calculated applying Theorem 1). The reported accuracies are the ones that refer to the ROC curves shown in Figure 5. The two values are similar and have been reported for completeness, even if this result may by misleading. The reason lies in the strong dataset unbalance, as only 15% of the patients experienced a cardiovascular event. In particular, both the Naïve Bayes classifier and ILD algorithm obtains a true positive rate and a false positive rate near zero in the point of the ROC curve which maximises the accuracy over the validation set, with a high misclassification in positive patients, which however cannot be noticed from the accuracy result. As it is known, the accuracy is not a good metric for unbalanced datasets, and the AUC is a much widely used metric that does not suffer from the problem described above.

The authors are aware that this is an unbalanced dataset and that balanced datasets are easier to deal with when studying classification problems. The goal was to give an example that reflects real use cases. For example, in medical problems, datasets are almost always extremely unbalanced since the event to be predicted is thankfully rare (for example death or the onset of diseases). It is not uncommon to find datasets with an unbalance of 99% vs. 1%. This was also the reason why the ILD algorithm concentrates on the AUC, as this is the typical metric used when dealing with such use-cases.

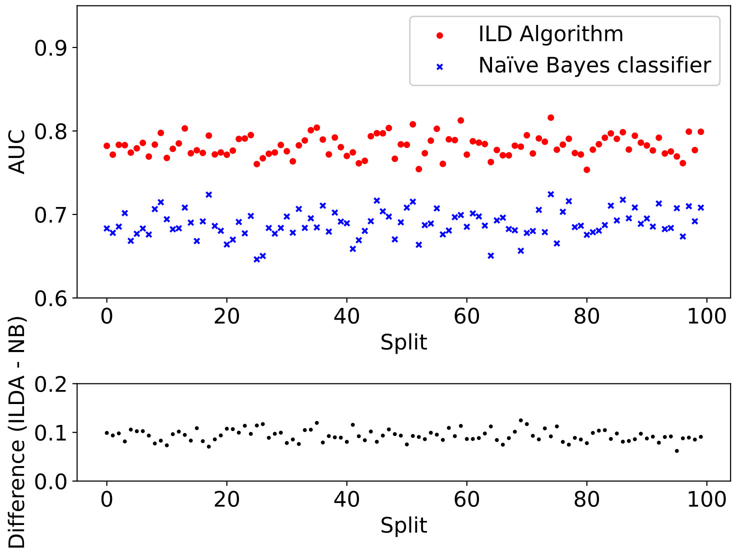


Figure 6. Comparison between the performance of the ILD algorithm (red) and Naïve Bayes classifier (blue) implemented on categorical features based on 100 different training and validation splits. (Top panel): AUC; (bottom panel): difference between the AUC provided by the ILD algorithm and the Naïve Bayes classifier.

6. Conclusions

The work presents a new algorithm, the ILD algorithm, which determines the best possible ROC curve that can be obtained from a dataset with categorical features and binary outcome, regardless of the predictive model.

The ILD algorithm is of fundamental importance to practitioners because it allows:

- to determine the prediction power (namely, the BE) of a specific set of categorical features,
- to decide when to stop searching for better models, and,
- to decide if it is necessary to enrich the dataset.

The ILD algorithm thus has the potential to revolutionize how binary prediction problems will be solved in the future, allowing practitioners to save an enormous amount of efforts, time, and money (considering that, for example, computing time is expensive especially in cloud environments).

The limitations of the ILD algorithm are the two restrictions for its applicability. First, the features must be categorical. The generalization of this approach to continuous features is the natural next step and will open new ways of understanding datasets with continuous features. Second, the ILD algorithm works well when the different buckets are populated with enough observations. The ILD algorithm would not give any useful information on a dataset with just one observation in each bucket (as it would be a perfect dataset). Consider the example of gray levels images. Even if pixel values could be considered categorical (the gray level of a pixel is an integer that can assume values from 0 to 255), two major problems would arise if the ILD algorithm would be applied to such a case: the number of buckets would be extremely large and each bucket would contain only one image therefore making the ILD algorithm completely useless, as only perfect buckets will be constructed.

An important further research direction is the expansion of the ILD algorithm to detect the best performing models that do not overfit the data. In the example of images, it is clear that being a perfect dataset one could theoretically construct a perfect predictor, therefore giving a maximum accuracy of 1. The interesting question is how to determine the maximum accuracy or the best AUC only in cases in which no overfitting is occurring. This is a nontrivial problem that is currently under investigation by the authors. To address, at least partially, this problem, the authors have defined a perfection index (IP) that can help in this regard. IP is discussed in Appendix A.

It is useful to note that this method differs significantly from other works as, for example, that in [29], where the authors describes RankOpt, a linear binary classifier which optimizes the area under the ROC curve (the AUC) or that in [30], which presents a Support Vector Method for optimizing multivariate nonlinear performance measures like the F1-score. This method is completely model-agnostic (in other words it does not depend on any model type) and determines the absolute maximum of the AUC with the given datasets. It does not try to maximize it, but it finds its absolute maximum.

To conclude, although more research is needed to generalize the ILD algorithm, it is, to the best knowledge of the authors, the first algorithm that is able to determine the exact BE from a generic dataset with categorical features, regardless of the predictive models.

Author Contributions: Conceptualization: Methodology, Investigation, Software, Writing—Original Draft, Writing—Review and Editing, Supervision: U.M.; Data curation, Software, Visualization: M.S.; Writing—Review and Editing, Investigation, Validation: D.P.; Writing—Review and Editing, Investigation, Validation: F.V.; Writing—Review and Editing, Project administration: M.A.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been partly funded by the European Union's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie-RISE Grant Agreement No 872181.

Data Availability Statement: The dataset used in the article is available for download at [25].

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

It is very useful to give a measure of *how* perfect a dataset is with a single number. To achieve this, a perfection index (PI) I_P can be defined.

Definition A1. *The Perfection Index I_P is defined as*

$$I_P \equiv \frac{1}{M}\sum_{i=1}^{N_B} |m_1^{[i]} - m_0^{[i]}|. \tag{A1}$$

Note that if a bucket i is perfect then either $m_1^{[i]}$ or $m_0^{[i]}$ are zero. In a perfect dataset $B/P = \varnothing$. In this case, from Equation (A1) it is easy to see that $I_P = 1$ as

$$I_P = \frac{\sum_{i\in B_1} m_1^{[i]} + \sum_{i\in B_0} m_0^{[i]}}{M} = 1 \tag{A2}$$

where

$$B_1 \equiv \{j \in [1, N_B] | m_0^{[j]} = 0\} \tag{A3}$$

and

$$B_0 \equiv \{j \in [1, N_B] | m_1^{[j]} = 0\} \tag{A4}$$

For an imperfect dataset, it is to see that I_P will be less than 1. In fact,

$$I_P = \frac{\sum_{i\in B_1} m_1^{[i]} + \sum_{i\in B_0} m_0^{[i]} + \sum_{i\in B_{01}} |m_1^{[i]} - m_0^{[i]}|}{M} \tag{A5}$$

where

$$B_{01} \equiv \{j \in [1, N_B] | m_1^{[j]} > 0 \text{ and } m_0^{[j]} > 0\} \tag{A6}$$

that cannot be one as long as B_{01} is not empty. There is a special interesting case when the dataset B is completely imperfect, meaning $B_0 = B_1 = \varnothing$, and for every bucket i with

$i = 1, \ldots, N_B$ is true that $m_1^{[i]} = m_0^{[i]}$. In this case, regardless of the predictions a model may make, the accuracy will always be 0.5. In this case, $I_P = 0$. In facts, one can see that in this case

$$a = \frac{1}{M} \sum_{i \in \tilde{B}} [p^{[i]}(m_1^{[i]} - m_0^{[i]}) + m_0^{[i]}] = \frac{1}{2} \tag{A7}$$

since $m_1^{[i]} = m_0^{[i]}$ and that

$$\sum_{i \in \tilde{B}} m_0^{[i]} = \sum_{i \in \tilde{B}} m_1^{[i]} = \frac{M}{2} \tag{A8}$$

This index is particularly useful since the following theorem can be proved.

Theorem A1. *The perfection index satisfy the relationship*

$$I_P = \max_{S_P} a - \min_{S_P} a \tag{A9}$$

where S_P is the set of all possible prediction vectors for the bucket set B. The perfection index measures that the ranges of possible values that the accuracy (a) can have.

Proof. To start the proof the formula for the maximum ($\max_{S_P} a$) and minimum ($\min_{S_P} a$) possible accuracy must be derived. Starting from Equation (12) and choosing (to get the maximum accuracy) $p^{[i]} = 1$ for $m_1^{[i]} \geq m_0^{[i]}$ and $p^{[i]} = 0$ for $m_1^{[i]} \leq m_0^{[i]}$ the result is

$$\max_{S_P} a = \frac{1}{M} \left[\sum_{m_1 \geq m_0} m_1^{[i]} + \sum_{m_1 \leq m_0} m_0^{[i]} \right] \tag{A10}$$

at the same time, by choosing $p^{[i]} = 0$ for $m_1^{[i]} \geq m_0^{[i]}$ and $p^{[i]} = 1$ for $m_1^{[i]} \leq m_0^{[i]}$ $\min_{S_P} a$ can be written as

$$\min_{S_P} a = \frac{1}{M} \left[\sum_{m_1 \geq m_0} m_0^{[i]} + \sum_{m_1 \leq m_0} m_1^{[i]} \right] \tag{A11}$$

To prove the theorem, let us rewrite the maximum accuracy from Equation (A10) as

$$\max_{S_P} a = \frac{1}{M} \sum_{m_1 \geq m_0} [(m_1^{[i]} - m_0^{[i]}) + \sum_{m_1 \leq m_0} [(m_1^{[i]} - m_0^{[i]}) + \sum_{m_1 \geq m_0} m_0^{[i]} + \sum_{m_1 \leq m_0} m_1^{[i]}] \tag{A12}$$

and using Equations (A2) and (A11) previous equation can be re-written as

$$\max_{S_P} a = I_P + \min_{S_P} a. \tag{A13}$$

This concludes the proof. □

Interpretation of the Perfection Index I_p

In the two extreme cases, if I_P is small then the ILD algorithm will give very useful information. The dataset is "imperfect" enough that an analysis as described is very useful. The more the value of I_P gets close to one, the more the analysis, as it is formulated here, is less helpful. This is because, in the case of a perfect dataset, a perfect prediction model can be easily built, and therefore the question of what is the best possible model loses significance. Note that there is a big assumption made, namely, that a feature bucket with a few observations contains the same information as one with one thousand observations in it. In real life, one feature bucket with just one (or few) observation will probably be due to the lack of observations collected with the given set of features and therefore should be doubted in its importance.

References

1. Raschka, S. Model evaluation, model selection, and algorithm selection in machine learning. *arXiv* **2018**, arXiv:1811.12808.
2. Arlot, S.; Celisse, A. A survey of cross-validation procedures for model selection. *Stat. Surv.* **2010**, *4*, 40–79. [CrossRef]
3. Michelucci, U.; Venturini, F. Estimating neural network's performance with bootstrap: A tutorial. *Mach. Learn. Knowl. Extr.* **2021**, *3*, 357–373. [CrossRef]
4. Michelucci, U. *Applied Deep Learning—A Case-Based Approach to Understanding Deep Neural Networks*; APRESS Media, LLC: New York, NY, USA, 2018.
5. Yu, T.; Zhu, H. Hyper-parameter optimization: A review of algorithms and applications. *arXiv* **2020**, arXiv:2003.05689.
6. García, V.; Mollineda, R.A.; Sánchez, J.S. On the k-NN performance in a challenging scenario of imbalance and overlapping. *Pattern Anal. Appl.* **2008**, *11*, 269–280. [CrossRef]
7. Yuan, B.W.; Luo, X.G.; Zhang, Z.L.; Yu, Y.; Huo, H.W.; Johannes, T.; Zou, X.D. A novel density-based adaptive k nearest neighbor method for dealing with overlapping problem in imbalanced datasets. *Neural Comput. Appl.* **2021**, *33*, 4457–4481. [CrossRef]
8. Schlimmer, J.C.; Granger, R.H. Incremental learning from noisy data. *Mach. Learn.* **1986**, *1*, 317–354. [CrossRef]
9. Angluin, D.; Laird, P. Learning from noisy examples. *Mach. Learn.* **1988**, *2*, 343–370. [CrossRef]
10. Raychev, V.; Bielik, P.; Vechev, M.; Krause, A. Learning programs from noisy data. *ACM Sigplan Not.* **2016**, *51*, 761–774. [CrossRef]
11. Tumer, K.; Ghosh, J. Bayes error rate estimation using classifier ensembles. *Int. J. Smart Eng. Syst. Des.* **2003**, *5*, 95–109. [CrossRef]
12. Gareth, J.; Daniela, W.; Trevor, H.; Robert, T. *An Introduction to Statistical Learning: With Applications in R*; Spinger: Berlin/Heidelberg, Germany, 2013.
13. Tumer, K.; Bollacker, K.; Ghosh, J. A mutual information based ensemble method to estimate bayes error. In *Intelligent Engineering Systems through Artificial Neural Networks*; ASME Press: New York, NY, USA, 1998; Volume 8.
14. Ghosh, J. Multiclassifier systems: Back to the future. In *International Workshop on Multiple Classifier Systems*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 1–15.
15. Richard, M.D.; Lippmann, R.P. Neural network classifiers estimate Bayesian a posteriori probabilities. *Neural Comput.* **1991**, *3*, 461–483. [CrossRef] [PubMed]
16. Shoemaker, P.; Carlin, M.; Shimabukuro, R.; Priebe, C. *Least-Squares Learning and Approximation of Posterior Probabilities on Classification Problems by Neural Network Models*; Technical Report; Naval Ocean Systems Center: San Diego, CA, USA, 1991.
17. Gibson, W.J.; Nafee, T.; Travis, R.; Yee, M.; Kerneis, M.; Ohman, M.; Gibson, C.M. Machine learning versus traditional risk stratification methods in acute coronary syndrome: A pooled randomized clinical trial analysis. *J. Thromb. Thrombolysis* **2020**, *49*, 1–9. [CrossRef] [PubMed]
18. Sherazi, S.W.A.; Jeong, Y.J.; Jae, M.H.; Bae, J.W.; Lee, J.Y. A machine learning–based 1-year mortality prediction model after hospital discharge for clinical patients with acute coronary syndrome. *Health Inform. J.* **2020**, *26*, 1289–1304. [CrossRef] [PubMed]
19. Vaid, A.; Somani, S.; Russak, A.J.; De Freitas, J.K.; Chaudhry, F.F.; Paranjpe, I.; Johnson, K.W.; Lee, S.J.; Miotto, R.; Richter, F.; others. Machine learning to predict mortality and critical events in a cohort of patients with COVID-19 in New York City: Model development and validation. *J. Med. Internet Res.* **2020**, *22*, e24018. [CrossRef] [PubMed]
20. Kim, H.J.; Han, D.; Kim, J.H.; Kim, D.; Ha, B.; Seog, W.; Lee, Y.K.; Lim, D.; Hong, S.O.; Park, M.J.; et al. An Easy-to-Use Machine Learning Model to Predict the Prognosis of Patients with COVID-19: Retrospective Cohort Study. *J. Med. Internet Res.* **2020**, *22*, e24225. [CrossRef] [PubMed]
21. Wang, S.; Pathak, J.; Zhang, Y. Using electronic health records and machine learning to predict postpartum depression. In *MEDINFO 2019: Health and Wellbeing e-Networks for All*; IOS Press, 1013 BG: Amsterdam, The Netherlands, 2019; pp. 888–892.
22. Hogg, R.V.; Tanis, E.A.; Zimmerman, D.L. *Probability and Statistical Inference*; Pearson/Prentice Hall: Upper Saddle River, NJ, USA, 2010.
23. Mahmood, S.S.; Levy, D.; Vasan, R.S.; Wang, T.J. The Framingham Heart Study and the epidemiology of cardiovascular disease: A historical perspective. *Lancet* **2014**, *383*, 999–1008. [CrossRef]
24. Nocedal, J.; Wright, S. *Numerical Optimization*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2006.
25. Framingham Dataset Download, Kaggle Website. 2021. Available online: https://www.kaggle.com/eeshanpaul/framingham (accessed on 29 June 2021).
26. Wilson, P.W.; D'Agostino, R.B.; Levy, D.; Belanger, A.M.; Silbershatz, H.; Kannel, W.B. Prediction of coronary heart disease using risk factor categories. *Circulation* **1998**, *97*, 1837–1847. [CrossRef] [PubMed]
27. D'Agostino, R.B.; Vasan, R.S.; Pencina, M.J.; Wolf, P.A.; Cobain, M.; Massaro, J.M.; Kannel, W.B. General cardiovascular risk profile for use in primary care. *Circulation* **2008**, *117*, 743–753. [CrossRef] [PubMed]
28. World Health Organisation. Cardiovascular Diseases (CVDs). 2021. Available online: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds) (accessed on 28 June 2021).
29. Herschtal, A.; Raskutti, B. Optimising area under the ROC curve using gradient descent. In Proceedings of the Twenty-First International Conference on Machine Learning, Banff, AB, Canada, 4–8 July 2004; p. 49.
30. Joachims, T. A support vector method for multivariate performance measures. In Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany, 7–11 August 2005; pp. 377–384.