Received: 8 June 2022 | Revised: 14 April 2023 | Accepted: 18 April 2023

DOI: 10.1111/stan.12297

ORIGINAL ARTICLE

WILEY

The optimal input-independent baseline for binary classification: The Dutch Draw

Joris Pries[1] | **Etienne van de Bijl**[1] | **Jan Klein**[1] | **Sandjai Bhulai**[2] | **Rob van der Mei**[1,2]

[1]Department of Stochastics, Centrum Wiskunde & Informatica, Amsterdam, The Netherlands

[2]Department of Mathematics, Vrije Universiteit, Amsterdam, The Netherlands

Correspondence
Joris Pries, Department of Stochastics, Centrum Wiskunde & Informatica, Amsterdam, North Holland 1098 XG, The Netherlands.
Email: jorispries@gmail.com

Before any binary classification model is taken into practice, it is important to validate its performance on a proper test set. Without a frame of reference given by a baseline method, it is impossible to determine if a score is “good” or “bad.” The goal of this paper is to examine all baseline methods that are independent of feature values and determine which model is the “best” and why. By identifying which baseline models are optimal, a crucial selection decision in the evaluation process is simplified. We prove that the recently proposed *Dutch Draw baseline* is the best *input-independent* classifier (independent of feature values) for all *order-invariant* measures (independent of sequence order) assuming that the samples are randomly shuffled. This means that the *Dutch Draw baseline* is the optimal baseline under these intuitive requirements and should therefore be used in practice.

KEYWORDS
baseline, benchmark, binary classification, evaluation, supervised learning

1 | INTRODUCTION

A *binary classification* model is trying to answer the following question: Should the instance be labeled as zero or one? This question might seem simple, but there are many practical applications for binary classification, ranging from predicting confirmed COVID-19 cases (Pirouz, Shaffiee Haghshenas, Shaffiee Haghshenas, & Piro, 2020), detecting malicious intrusions (Li, Yu, Bai, Hou, & Chen, 2018) to determining if a runner is fatigued or not (Buckley et al., 2017). Whenever a classification model is developed for a practical application, it is important to validate the performance on a test set. However, a baseline is necessary to put the achieved performance in perspective. Without this frame of reference, only partial conclusions can be drawn from the results. An accuracy of 0.9 indicates that 90% of all predictions are correct. But it could be that the model actually did not learn anything and such a high accuracy can already be achieved by predicting only zeros. To put the performance in perspective, it should therefore be compared with some meaningful benchmark method, preferably with a state-of-the-art model.

Nevertheless, many state-of-the-art methods are instance-specific. They can rapidly change and often involve many fine-tuned parameters. Thus, as a necessary additional check in the development process, van de Bijl et al. (2022) plead for a supplementary baseline that is *general*, *simple*, and *informative*. This is used to test if the new model truly performs better than a simple model. It should be considered a major warning sign when a model is outperformed by, for example, a weighted coin flip. The model can use information about the feature values of a sample, yet it is outperformed by a model that does not even consider these values. Is the model then actually learning something productive?

A theoretical approach for binary classification is proposed in van de Bijl et al. (2022) based on *Dutch Draw classifiers*. Such a classifier draws uniformly at random (u.a.r.) a subset out of all samples, and labels these 1, and the rest 0. The size of the drawn subset is optimized to obtain the optimal expected performance, which is the *Dutch Draw baseline*. For most commonly used performance measures, a closed-form expression is given (van de Bijl et al., 2022).

However, there are infinitely many ways to devise a baseline method. We only investigate prediction models that do not take any information from the features into account, as this will result in a more general and simple baseline. We call these models *input-independent*. Irrespective of the input, the way that such a model predicts remains the same. Any newly developed model should at least beat the performance of these kinds of models, as an *input-independent* model cannot exploit patterns in the data to predict the labels more accurately. However, sometimes a model can get lucky by accidentally predicting the labels perfectly for a specific order of the labels. The order of the samples should not influence the "optimality" of a model. This is why we introduce the notion of *average-permutation-optimality*. Furthermore, the order of the samples should not change the outcome of the performance measure (*order-invariant*). This is not a strict condition, as most commonly used measures have this property. Under these restrictions, we prove that the *Dutch Draw baseline* is *average-permutation-optimal* out of all *input-independent* classifiers for any *order-invariant* measure.

To summarize, in this paper we:

- determine natural requirements for a general, informative and simple baseline;
- prove that the Dutch Draw baseline is the optimal baseline under these requirements.

These contributions improve the evaluation process of any new binary classification method.

The remainder of this paper is organized as follows. First, the necessary preliminaries and notations are discussed in Section 2. Next, in Section 3 we determine requirements for a general, simple and informative baseline. Furthermore, we formally define what optimality entails under these requirements. In Section 4, an alternative definition for the *Dutch Draw classifiers* is given, which is necessary for the main proof. In Section 5, we prove that the Dutch Draw baseline is optimal. Finally, Section 6 summarizes the general findings and discusses possible future research opportunities.

2 | PRELIMINARIES

Next, we introduce some concepts and notations to lay the foundation for the main proof. *Binary classifiers* (Section 2.1) and *performance measures* (Section 2.2) for binary classification are discussed, which will play a crucial role in the proof of the main result.

2.1 | Binary classifiers

To find a good baseline for a *binary classification model*, we first have to discuss what a *binary classifier* actually is. To this end, let $\mathcal{X}$ be the feature space (think e.g., $\mathbb{R}^d$). Normally, a binary classifier is defined as a function $h : \mathcal{X} \times \mathbb{R} \rightarrow \{0, 1\}$ that maps feature values to zero or one, where the second input is used to model stochasticity. However, this classifier only classifies one sample at a time. Instead, we are interested in classifiers that classify *multiple* samples *simultaneously*: $h_M : \mathcal{X}^M \times \mathbb{R} \rightarrow \{0, 1\}^M$, where $M \in \mathbb{N}_{>0}$ denotes the number of samples that are classified. This gives classifiers the ability to precisely predict k out of M samples positive. Note that a *single sample* classifier h can simply be extended to classify M samples *simultaneously* by applying the classifier for each sample individually using $h_M : ((x_1, \dots, x_M), r) \mapsto (h(x_1, r), \dots, h(x_M, r))$. Note that $r \in \mathbb{R}$ can be viewed as a random seed. Let $\mathcal{H}_M$ be the set of *all* binary classifiers that classify M samples at the same time.

2.2 | Performance measures for binary classification

To assess the effectiveness of a binary classification model, it is necessary to choose a *performance measure*, which quantifies how much the *predicted* labels agree with the *actual* labels. Namely, each sample indexed by i has feature values $\mathbf{x}_i \in \mathcal{X}$ and a corresponding label $y_i \in \{0, 1\}$. Let $\mathbf{X} := (\mathbf{x}_1 \dots \mathbf{x}_M) \in \mathcal{X}^M$ be the combined feature values of M samples. Furthermore, let $\mathbf{Y} = (y_1, \dots, y_M)$ denote the corresponding labels. A performance measure for binary classification is then defined as $\mu : \{0, 1\}^M \times \{0, 1\}^M \rightarrow \mathbb{R}$, where the first entry of μ is the predictions made by the classifier and the second entry is the corresponding labels. The performance of classifier h_M can now be written as: $\mu(h_M(\mathbf{X}, r), \mathbf{Y})$.

2.2.1 | Undefined cases

Some measures are undefined for specific combinations of $h_M(\mathbf{X}, r)$ and $\mathbf{Y}$. Take for example the *true positive rate* (Tharwat, 2021), which is the number of correctly *predicted* positives divided by the total number of *actual* positives. When there are no actual positives, the measure is ill-defined, as it divides by zero. Less obvious, the measure *negative predictive value* (Tharwat, 2021) is undefined when no negatives are predicted, as it is defined as the number of correctly predicted negatives divided by the total number of predicted negatives. Assigning a constant value C to undefined cases solves many issues. However, this can make it desirable for a classifier to always predict labels that lead to a previously undefined measure in order to minimize the measure. Therefore, we redefine μ from now on for every $\hat{\mathbf{Y}}, \mathbf{Y} \in \{0,1\}^M$ to be equal to a specific constant C_{undef}, when $\mu(\hat{\mathbf{Y}}, \mathbf{Y})$ was undefined. We make a distinction for each objective (maximizing/minimizing). Let

$$C_{\text{undef}} := \begin{cases} \max\limits_{\hat{\mathbf{Y}}, \mathbf{Y} \in \{0,1\}^M} \left\{\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right\} & \text{if minimizing,} \\ \min\limits_{\hat{\mathbf{Y}}, \mathbf{Y} \in \{0,1\}^M} \left\{\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right\} & \text{if maximizing.} \end{cases}$$

It is therefore always disadvantageous for a classifier to predict a previously undefined case. By defining C_{undef} in this way, we do not have to omit such classifiers from our analysis.

3 | ESSENTIAL CONDITIONS

To prove that the optimal Dutch Draw classifier yields the "optimal" baseline, we first have to define "optimality." When is a baseline considered to be optimal? To determine this, the following two questions must be answered: (1) which methods do we compare and (2) how do we compare them? To this end, we define the notion of *input-independent* classifiers, *order-invariant* measures, and *average-permutation-optimality*.

3.1 | Input-independent classifier

Any binary classifier can be used as a baseline. However, any good standardized baseline should be *general*, *simple*, and *informative* (van de Bijl et al., 2022). Thus, it needs to be applicable to any domain, quick to train and clearly still beatable. To this end, we investigate all models that do not take any feature values into account, as they meet these three requirements. Without considering feature values, they can be applied to any domain. Furthermore, they do not require any training, because they cannot learn the relationship between the feature values and the corresponding labels. This makes them also clearly still beatable, as any newly developed model should leverage the information from the feature values to make better predictions.

A binary classifier $h_M \in \mathcal{H}_M$ is called *input-independent* if for all feature values $\mathbf{X}_i, \mathbf{X}_j \in \mathcal{X}^M$ and $r \in \mathbb{R}$ it holds that $h_M(\mathbf{X}_i, r) = h_M(\mathbf{X}_j, r) =: h_M(r)$, where the notation of $h_M(r)$ is chosen to visualize that the classifier h_M is not dependent on the input. An example of an input-independent classifier is a *coin flip*, as the feature values have no influence on the probability distribution of the coin. Let $\mathcal{H}_M^{i.i.} = \{h_M \in \mathcal{H}_M : h_M \text{ is input-independent}\}$ be the set of all input-independent binary classifiers. A newly developed model, that was optimized using the same performance measure, should always beat the performance of an *input-independent* model, as it gains information from the feature values. Otherwise, the model was not able to exploit this extra information to make better predictions.

3.2 | Order-invariant measure

To assess the performance of a method, a measure needs to be chosen. Reasonably, the order of the samples should not change the outcome of this measure. If a measure has this property, we call it *order-invariant*. To define this formally, we have to introduce the following notions. Let S_M denote the set of all permutations of a set of size M by $S_M := \left\{\pi : \{1, \dots, M\} \to \{1, \dots, M\} \text{ s.t. } \{\pi(i)\}_{i=1}^{M} = \{1, \dots, M\}\right\}$. To apply permutations to a matrix, we consider *sample-wise permutations*. For every $M \times K$ dimensional matrix $X = (\mathbf{x}_1 \dots \mathbf{x}_M)$, let X_π denote the sample-wise permutation under π. Thus, $X_\pi := \left(\mathbf{x}_{\pi(1)} \dots \mathbf{x}_{\pi(M)}\right)$, with $K \in \mathbb{N}_{>0}$ the number of features. This means that the matrix X is reordered by row.

A measure μ is *order-invariant* if for every permutation $\pi \in S_M$ and for all $h_M(\mathbf{X}, r), \mathbf{Y} \in \{0, 1\}^M$ it holds that:

$$\mu\left(h_M(\mathbf{X}, r), \mathbf{Y}\right) = \mu(h_M(\mathbf{X}, r)_\pi, \mathbf{Y}_\pi). \tag{1}$$

This means that any reordering of the coupled *predicted* and *actual* labels does not affect the performance score. This is not a hard restriction, as most measures have this property. Note for example that the number of *true positives* (TP), *true negatives* (TN), *false positives* (FP), and *false negatives* (FN) are all *order-invariant*. Most commonly used measures are a function of these four measures (Sokolova & Lapalme, 2009), making them also *order-invariant*.

3.3 | Defining optimality

To find the "optimal" baseline, it is first essential to specify what "optimality" entails. There are three important factors to discuss. (1) A binary classifier can be *stochastic*, which is why we examine the *expected* performance. (2) The "optimal" classifier is the best classifier with respect to the group of classifiers that is considered (denoted by $\tilde{\mathcal{H}}_M$). (3) It is the "best" for a *specific* dataset, but we consider all permutations of the dataset. Otherwise, the "optimal" classifier would simply be the deterministic classifier that produces the exact labels accidentally. This phenomenon is similar to a broken clock that gives the correct time twice a day, but should not be used to determine the time.

With this in mind, we introduce the notion of *average-permutation-optimality*. A classifier is *average-permutation-optimal* if it minimizes/maximizes the expected performance for a random permutation of the test set out of all considered binary classifiers ($\tilde{\mathcal{H}}_M$). Thus,

$$h_M^{\min} \in \underset{h_M \in \tilde{\mathcal{H}}_M}{\arg\min} \left\{ \mathbb{E}_{\pi \sim \mathcal{U}(S_M)} \left[\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(\mathbf{X}_\pi, r), \mathbf{Y}_\pi) \right] \right] \right\}, \tag{2}$$

$$h_M^{\max} \in \underset{h_M \in \tilde{\mathcal{H}}_M}{\arg\max} \left\{ \mathbb{E}_{\pi \sim \mathcal{U}(S_M)} \left[\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(\mathbf{X}_\pi, r), \mathbf{Y}_\pi) \right] \right] \right\}. \tag{3}$$

4 | THE DUTCH DRAW

A Dutch Draw classifier is defined in van de Bijl et al. (2022) for $\theta \in [0, 1]$, as

$$\sigma_\theta(\mathbf{X}, \cdot) := (\mathbf{1}_E(i))_{i \in \{1, \dots M\}} \quad \text{with} \quad E \subseteq \{1, \dots M\} \text{drawn u.a.r. such that } |E| = \lfloor M \cdot \theta \rceil. \tag{4}$$

In other words, the classifier draws u.a.r. a subset E of size $\lfloor M \cdot \theta \rceil$ out of all samples, which it then labels as 1, while the rest is labeled 0. In this section, we introduce an alternative definition, that is used in the main proof, and show that all Dutch Draw classifiers are *input-independent*.

4.1 | Alternative definition

Instead of the definition in Equation (4), we introduce an alternative definition for the Dutch Draw classifiers to simplify the proof of the main result. Given a binary vector $(y_1, \dots, y_M) \in \{0,1\}^M$ of length M, note that the number of ones it contains can be counted by taking the sum $\sum_{i=1}^{M} y_i$. Next, we define sets of binary vectors (of the same length) that contain the same number of ones. For all $j \in \{0, \dots, M\}$, define

$$\mathcal{Y}_j := \left\{ \hat{\mathbf{Y}} = (y_1, \dots, y_M) \in \{0,1\}^M \text{ s.t. } \sum_{i=1}^{M} y_i = j \right\}. \tag{5}$$

In other words, $\mathcal{Y}_j$ contains all binary vectors of length M with exactly j ones and M-j zeros.

A Dutch Draw classifier selects u.a.r. E out of M samples and labels these as one, and the rest zero. Note that this is the same as taking u.a.r. a vector from $\mathcal{Y}_E$. To simplify notation, let $\mathcal{U}(A)$ denote the uniform distribution over a finite set A. Thus, when $X \sim \mathcal{U}(A)$ it must hold that $\mathbb{P}(X = a) = \frac{1}{|A|}$ for each $a \in A$. Now, a Dutch Draw classifier σ_θ can be rewritten as

$$\sigma_\theta(\mathbf{X}, \cdot) := \hat{\mathbf{Y}} \text{ with } \hat{\mathbf{Y}} \sim \mathcal{U}\left(\mathcal{Y}_{\lfloor M \cdot \theta \rceil}\right). \tag{6}$$

Put differently, a Dutch Draw classifier σ_θ chooses u.a.r. a vector with exactly $\lfloor M \cdot \theta \rceil$ ones as prediction out of all vectors with $\lfloor M \cdot \theta \rceil$ ones ($\mathcal{Y}_{\lfloor M \cdot \theta \rceil}$). This alternative definition simplifies the proof of the main result.

4.2 | Input-independence

Next, we discuss why all Dutch Draw classifiers are *input-independent* (see Section 3.1). Note that a Dutch Draw classifier σ_θ is independent of feature values, as it is only dependent on θ and M, see Equation (6). In other words, any Dutch Draw classifier is by definition *input-independent*. Instead of $\sigma_\theta(\mathbf{X}, r)$, we can therefore write $\sigma_\theta(r)$. To conclude, for every $\theta \in [0,1]$ it holds that $\sigma_\theta \in \mathcal{H}_M^{i.i.}$, which is the set of all input-independent binary classifiers.

4.3 | Optimal Dutch Draw classifier

The optimal Dutch Draw classifier $\sigma_{\theta_{\text{opt}}}$ is determined by minimizing/maximizing the expected performance for the parameter θ out of all allowed parameter values Θ (van de Bijl et al., 2022). Note that some measures are undefined for certain predictions, thus Θ is not always equal to $[0,1]$. Take, for example, the measure *precision* (Tharwat, 2021), which is defined as the number of true positives divided by the total number of predicted positives. Therefore, if no positives are predicted, the measure becomes undefined (division by zero). By adapting each measure according to Section 2.2.1, all undefined cases are resolved and $\Theta = [0,1]$ always holds.

Using the alternative definition of the Dutch Draw classifier (see Equation 6), we obtain:

$$\theta^*_{\min} \in \underset{\theta \in [0,1]}{\arg\min} \left\{ \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_{\lfloor M \cdot \theta \rceil})} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y}) \right] \right\} \text{ and } \theta^*_{\max} \in \underset{\theta \in [0,1]}{\arg\max} \left\{ \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_{\lfloor M \cdot \theta \rceil})} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y}) \right] \right\}. \tag{7}$$

Depending on the objective, either $\sigma_{\theta^*_{\min}}$ or $\sigma_{\theta^*_{\max}}$ is an optimal Dutch Draw classifier. Observe that the optimal Dutch Draw classifier depends on Y, thus a different dataset could lead to a different optimal Dutch Draw classifier.

5 | THEOREM AND PROOF

After defining *input-independence* (Section 3.1), *order-invariance* (Section 3.2), *average-permutation-optimality* (Section 3.3), and introducing an alternative formulation for the Dutch Draw classifier, all ingredients for the following theorem are present.

Theorem 1 (Main result). *The optimal Dutch Draw classifier $\sigma_{\theta_{opt}}$ is average-permutation-optimal out of all input-independent classifiers ($\mathcal{H}_M^{i.i.}$), for any order-invariant measure μ. In other words:*

$$\sigma_{\theta^*_{\min}} \in \underset{h_M \in \mathcal{H}_M^{i.i.}}{\arg\min} \left\{ \mathbb{E}_{\pi \sim \mathcal{U}(S_M)} \left[\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(\mathbf{X}_\pi, r), \mathbf{Y}_\pi) \right] \right] \right\}, \tag{8}$$

$$\sigma_{\theta^*_{\max}} \in \underset{h_M \in \mathcal{H}_M^{i.i.}}{\arg\max} \left\{ \mathbb{E}_{\pi \sim \mathcal{U}(S_M)} \left[\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(\mathbf{X}_\pi, r), \mathbf{Y}_\pi) \right] \right] \right\}. \tag{9}$$

This means that the optimal Dutch Draw classifier is the best general, simple, and informative baseline.

5.1 | Intuition behind proof

An *input-independent* classifier cannot learn the actual label from feature values. The predictions are therefore arbitrary. By averaging the performance over all permutations of the dataset (*average-permutation-optimal*), it is only relevant how many labels are predicted to be zero (or one) by the classifier, as the performance measure is not dependent on the order (*order-invariance*). The optimal Dutch Draw classifier is determined by optimizing the number of predicted zeros and ones, which makes this baseline *average-permutation-optimal*.

Proof. Let $h_M \in \mathcal{H}_M^{i.i.}$ be an *input-independent* classifier and let μ be a *order-invariant* measure, where we assume that μ needs to be maximized. The classifier is *average-permutation-optimal* if it maximizes the expected performance under a random permutation of the test set out of all *input-independent* classifiers (see Equation 3).

For any *input-independent* classifier h_M, it holds that

$$\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(\mathbf{X}_\pi, r), \mathbf{Y}_\pi) \right] = \mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(r), \mathbf{Y}_\pi) \right]. \tag{10}$$

The input $\mathbf{X}_\pi$ is not relevant for the classification, and can thus be omitted. In total, there are 2^M unique possible predictions in $\{0,1\}^M$. Denote these distinct vectors by $\hat{\mathbf{Y}}_1, \dots, \hat{\mathbf{Y}}_{2^M}$ such that $\bigcup_{i=1}^{2^M} \hat{\mathbf{Y}}_i = \{0,1\}^M$. Next, the expectation in Equation (10) can be written out by:

$$\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(r), \mathbf{Y}_\pi) \right] = \sum_{i=1}^{2^M} \mathbb{P}(h_M(\cdot) = \hat{\mathbf{Y}}_i) \cdot \mu(\hat{\mathbf{Y}}_i, \mathbf{Y}_\pi). \tag{11}$$

As we need to proof *average-permutation-optimality*, we have to take the expectation of Equation (11) over all permutations. Using linearity of expectation gives:

$$\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}\left[\sum_{i=1}^{2^M}\mathbb{P}(h_M(\cdot)=\hat{\mathbf{Y}}_i)\cdot\mu(\hat{\mathbf{Y}}_i,\mathbf{Y}_\pi)\right]=\sum_{i=1}^{2^M}\mathbb{P}(h_M(\cdot)=\hat{\mathbf{Y}}_i)\cdot\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}\left[\mu(\hat{\mathbf{Y}}_i,\mathbf{Y}_\pi)\right]. \quad (12)$$

Instead of taking the expectation of a sum, we now take the sum of expectations.

The measure μ is *order-invariant*, thus using Equation (1) gives

$$\mu(\hat{\mathbf{Y}}_i,\mathbf{Y}_\pi)=\mu((\hat{\mathbf{Y}}_i)_{\pi^{-1}},(\mathbf{Y}_\pi)_{\pi^{-1}})=\mu((\hat{\mathbf{Y}}_i)_{\pi^{-1}},\mathbf{Y}). \quad (13)$$

Applying a permutation does not change a *order-invariant* measure μ. In this case, we apply the inverse permutation π^{-1} to retrieve $\mathbf{Y}$.

Because of Equation (13), it therefore also holds that

$$\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}\left[\mu(\hat{\mathbf{Y}}_i,\mathbf{Y}_\pi)\right]=\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}\left[\mu((\hat{\mathbf{Y}}_i)_{\pi^{-1}},\mathbf{Y})\right]. \quad (14)$$

S_M is a *group*, which is why the set of all *inverse permutations* is the same as the set of all *permutations* (Artin, 2011; Dixon & Mortimer, 1996). Given that the permutations are drawn u.a.r., taking the expectation over all the *inverse permutations* is the same as taking the expectation over all *permutations*. When permutation π is drawn u.a.r., it namely holds that $\mathbb{P}(\pi=s)=\mathbb{P}(\pi=s^{-1})=\frac{1}{|S_M|}$ for all $s\in S_M$. Therefore,

$$\begin{aligned}\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}\left[\mu((\hat{\mathbf{Y}}_i)_{\pi^{-1}},\mathbf{Y})\right]&=\sum_{s\in S_M}\left(\mu((\hat{\mathbf{Y}}_i)_{s^{-1}},\mathbf{Y})\cdot\mathbb{P}(\pi=s)\right)=\sum_{s\in S_M}\left(\mu((\hat{\mathbf{Y}}_i)_{s^{-1}},\mathbf{Y})\cdot\mathbb{P}(\pi=s^{-1})\right)\\&=\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}\left[\mu((\hat{\mathbf{Y}}_i)_{\pi},\mathbf{Y})\right]. \quad (15)\end{aligned}$$

Thus, π^{-1} can be replaced with π in Equation (14).

Recall that $\mathcal{Y}_j$ is the set of all binary vectors of length M with j ones (see Equation 5). Furthermore, note that applying a u.a.r. chosen permutation $\pi\in S_M$ on $\hat{\mathbf{Y}}_i\in\mathcal{Y}_j$ is the same as selecting u.a.r. $\hat{\mathbf{Y}}\in\mathcal{Y}_j$ as outcome, because for every $\hat{\mathbf{Y}}_\star\in\mathcal{Y}_j$ it holds that

$$\mathbb{P}\left(\hat{\mathbf{Y}}=\hat{\mathbf{Y}}_\star\right)=\frac{1}{|\mathcal{Y}_j|}\text{with }\hat{\mathbf{Y}}\sim\mathcal{U}\left(\mathcal{Y}_j\right)\ \text{ and }\ \mathbb{P}\left((\hat{\mathbf{Y}}_i)_\pi=\hat{\mathbf{Y}}_\star\right)=\frac{1}{|\mathcal{Y}_j|}\text{with }\pi\sim\mathcal{U}\left(S_M\right).$$

The latter follows, as for each $\hat{\mathbf{Y}}_\star,\hat{\mathbf{Y}}_\triangle\in\mathcal{Y}_j$ there are exactly as many permutations to go from $\hat{\mathbf{Y}}_i$ to $\hat{\mathbf{Y}}_\star$ as permutations to go from $\hat{\mathbf{Y}}_i$ to $\hat{\mathbf{Y}}_\triangle$.

Let $|\hat{\mathbf{Y}}_i|$ denote the number of ones in $\hat{\mathbf{Y}}_i$. Now, we can rewrite the expectation $\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}[\cdot]$ over all permutations into an expectation over a u.a.r. drawn vector with the same number of ones, by

$$\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}\left[\mu((\hat{\mathbf{Y}}_i)_\pi,\mathbf{Y})\right]=\mathbb{E}_{\hat{\mathbf{Y}}\sim\mathcal{U}\left(\mathcal{Y}_{|\hat{\mathbf{Y}}_i|}\right)}\left[\mu(\hat{\mathbf{Y}},\mathbf{Y})\right]. \quad (16)$$

Using Equations (14), (15), and (16) in combination with Equation (12) gives

$$\sum_{i=1}^{2^M}\mathbb{P}(h_M(\cdot)=\hat{\mathbf{Y}}_i)\cdot\mathbb{E}_{\pi\sim\mathcal{U}(S_M)}\left[\mu(\hat{\mathbf{Y}}_i,\mathbf{Y}_\pi)\right]=\sum_{i=1}^{2^M}\mathbb{P}(h_M(\cdot)=\hat{\mathbf{Y}}_i)\cdot\mathbb{E}_{\hat{\mathbf{Y}}\sim\mathcal{U}\left(\mathcal{Y}_{|\hat{\mathbf{Y}}_i|}\right)}\left[\mu(\hat{\mathbf{Y}},\mathbf{Y})\right].$$

We have now eliminated all permutations from the equation. Note that the expectation in the right-hand side is the same for each $\hat{\mathbf{Y}}_i \in \mathcal{Y}_j$. In other words, the expectation is the same for two vectors, when they have the same number of ones. Grouping the vectors with the same number of ones, gives

$$\sum_{i=1}^{2^M} \mathbb{P}(h_M(\cdot) = \hat{\mathbf{Y}}_i) \cdot \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}\left(\mathcal{Y}_{|\hat{\mathbf{Y}}_i|}\right)} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right] = \sum_{j=0}^{M} \mathbb{P}(h_M(\cdot) \in \mathcal{Y}_j) \cdot \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right].$$

Instead of summing over all possible binary vectors $\hat{\mathbf{Y}}_i \in \{0,1\}^M$, all vectors with the same number of ones are grouped together, as they have the same expectation. All probability mass of the grouped vectors is also added up. Note, that it is thus only relevant for a classifier in which group $\mathcal{Y}_j$ the prediction $h_M(\cdot)$ belongs.

For any $j \in \{0, \dots, M\}$ it holds that $\mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right]$ is bounded by maximizing over all possible values of j. Thus,

$$\mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right] \leq \max_{j' \in \{0, \dots, M\}} \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_{j'})} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right]. \tag{17}$$

Observe that $\sum_{j=0}^{M} \mathbb{P}(h_M(\cdot) \in \mathcal{Y}_j) = 1$ and $\mathbb{P}(h_M(\cdot) \in \mathcal{Y}_j) \geq 0$ hold for each j, therefore it follows using Equation (17) that

$$\sum_{j=0}^{M} \mathbb{P}(h_M(\cdot) \in \mathcal{Y}_j) \cdot \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right] \leq \max_{j' \in \{0, \dots, M\}} \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_{j'})} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right].$$

Consequently, we have found an upper bound for Equation (9). Namely,

$$\max_{h_M \in \mathcal{H}_M^{i.i.}} \left\{ \mathbb{E}_{\pi \sim \mathcal{U}(S_M)} \left[\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(\mathbf{X}_\pi, r), \mathbf{Y}_\pi)\right]\right] \right\} \leq \max_{j \in \{0, \dots, M\}} \left\{ \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \mu(\hat{\mathbf{Y}}, \mathbf{Y}) \right\}. \tag{18}$$

Equality only holds for any classifier $h_M \in \mathcal{H}_M^{i.i.}$, when all probability mass is given to $\arg\max_{j \in \{0, \dots, M\}} \left\{ \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \mu(\hat{\mathbf{Y}}, \mathbf{Y}) \right\}$. In other words, the maximum can only be attained if

$$\sum_{j_{\max} \in \arg\max_{j \in \{0, \dots, M\}} \left\{ \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \mu(\hat{\mathbf{Y}}, \mathbf{Y}) \right\}} \mathbb{P}(h_M(\cdot) \in \mathcal{Y}_{j_{\max}}) = 1. \tag{19}$$

A classifier $h_M \in \mathcal{H}_M^{i.i.}$ can therefore only attain the maximum if all predictions belong to a group $\mathcal{Y}_j$ or possibly multiple groups that all maximize the expectation.

Remember that the Dutch Draw selects the optimal classifier based on Equation (7), which leads to

$$\lfloor M \cdot \theta^*_{\max} \rceil \in \arg\max_{j \in \{0, \dots, M\}} \left\{ \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \left[\mu(\hat{\mathbf{Y}}, \mathbf{Y})\right] \right\}.$$

Combining this with the alternative definition of the Dutch Draw (Equation 6) directly gives that

$$\sum_{j_{\max} \in \arg\max_{j \in \{0, \dots, M\}} \left\{ \mathbb{E}_{\hat{\mathbf{Y}} \sim \mathcal{U}(\mathcal{Y}_j)} \mu(\hat{\mathbf{Y}}, \mathbf{Y}) \right\}} \mathbb{P}(\sigma_{\theta^*_{\max}}(\cdot) \in \mathcal{Y}_{j_{\max}}) = 1.$$

This shows in combination with Equation (19) that the optimal Dutch Draw classifier *actually* attains the bound given in Equation (18). It now follows that,

$$\sigma_{\theta^*_{\max}} \in \underset{h_M \in \mathcal{H}_M^{i.i.}}{\arg\max} \left\{ \mathbb{E}_{\pi \sim \mathcal{U}(S_M)} \left[\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(\mathbf{X}_\pi, r), \mathbf{Y}_\pi) \right] \right] \right\}.$$

Similarly, we also find that when μ needs to be *minimized*, it follows that

$$\sigma_{\theta^*_{\min}} \in \underset{h_M \in \mathcal{H}_M^{i.i.}}{\arg\min} \left\{ \mathbb{E}_{\pi \sim \mathcal{U}(S_M)} \left[\mathbb{E}_{r \in \mathbb{R}} \left[\mu(h_M(\mathbf{X}_\pi, r), \mathbf{Y}_\pi) \right] \right] \right\}.$$

Thus, we can conclude that the optimal Dutch Draw classifier attains the minimum/maximum expected performance and is therefore *average-permutation-optimal* for all *input-independent* classifiers with a *order-invariant* measure. ■

6 | DISCUSSION AND CONCLUSION

A baseline is crucial to assess the performance of a prediction model. However, there are infinitely many ways to construct a baseline. As a necessary check in the development process, van de Bijl et al. (2022) plead for a supplementary baseline that is *general*, *simple*, and *informative*. In this paper, we have therefore examined all baselines that are independent of feature values, which makes them general and relatively simple. Additionally, these baselines are also informative, as it should be considered a major warning sign when a newly developed model is outperformed by a model that does not take any feature values into account. In this paper, we have shown that, out of all *input-independent* binary classifiers, the Dutch Draw baseline is *average-permutation-optimal* for any *order-invariant* measure. Our findings improve the evaluation process of any new binary classification method, as we have proven that the Dutch Draw baseline is ideal to gauge the performance score of a newly developed model.

Next, we discuss two points that could be considered an "unfair" advantage for the Dutch Draw baseline. Firstly, we have considered in this paper classifiers that predict M labels *simultaneously*. This gives classifiers a potential advantage over classifying each sample *sequentially*, as for example, exactly k out of M samples can be labeled positive. This can only be done sequentially when a classifier is allowed to track previous predictions or to change based on the number of classifications it has made. Even with this advantage, we believe that all *input-independent* models still remain clearly beatable by a newly developed model.

Secondly, the Dutch Draw baseline can be derived for most commonly used measures without any additional knowledge about the number of positive labels P. Nonetheless, it was shown in van de Bijl et al. (2022) that the Dutch Draw baseline can only be calculated for the measure *accuracy* when it is known if $P \geq M/2$ holds. If the distribution of the training set is the same as the test set, the training set can be used to determine whether $P \geq M/2$ is likely to hold. Furthermore, a domain expert could estimate whether it is likely that a dataset contains more positives than negatives. Take for example a cybersecurity dataset, where there are often significantly less harmful instances and more normal instances (Wheelus, Bou-Harb, & Zhu, 2018). There are thus many ways to estimate if $P \geq M/2$ holds. Nevertheless, even if the Dutch Draw baseline uses this information (only for the *accuracy*), we believe that any newly developed model should still beat the Dutch Draw baseline, as it does not use any feature values to improve prediction.

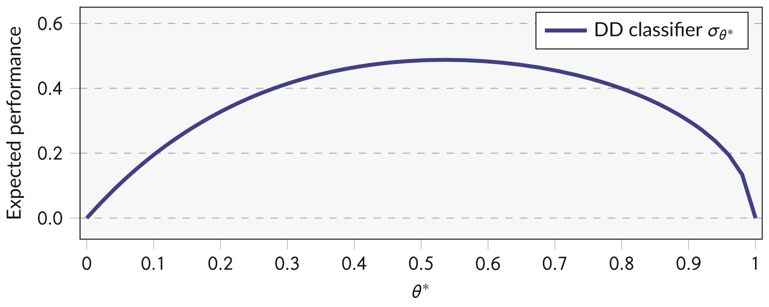

FIGURE 1 Nontrivial Dutch Draw (DD) baseline. For every $\theta^* \in \Theta^*$, the expected G-mean 2 of the DD classifier σ_{θ^*} is determined for a dataset with $P = 5$ and $N = 45$. The maximum (≈ 0.4877) is attained when $\theta^* = 0.54$.

Finally, we address future research opportunities. In this paper, we have only considered *binary* classification. A natural extension would be to also consider *multiclass* classification (Grandini, Bagli, & Visani, 2020), probabilistic classification, or regression (between $[0, 1]$). Is a strategy similar to the Dutch Draw optimal in these cases? Can a closed-form expression of the optimal baseline be derived? We believe that the three introduced properties (namely, input-independent, order-invariant, and average-permutation-optimal) are still relevant for these problems. This could help identify what kind of classifier is considered to be optimal. van de Bijl et al. (2022) stated that the Dutch Draw baseline could be used to scale existing measures. This paper provides more motivation to scale measures with the Dutch Draw baseline and not by using any other *input-independent* classifier. Yet, it could still be investigated how each measure should be scaled in order to maximize the explainability behind a performance score. Most performance measures reduce to a linear function of TP, which makes a Dutch Draw classifier that predicts as many positives/negatives (without making the performance measure undefined) optimal. However, there are nonlinear cases, such as the *G-mean 2*, that lead to a non-trivial baseline. Take, for example, $P = 5$ and $N = 45$, which leads to a Dutch Draw baseline of approximately 0.4877 with $\theta^*_{\max} = 0.54$ (see Figure 1).

Nonlinear cases could perhaps also lead to situations where multiple ($\underline{\text{not all}}$) $\mathcal{Y}_j$ groups are optimal (see Equation 19). We have not encountered this behavior for common performance measures, but it would be interesting to identify performance measures that do have this property.

ACKNOWLEDGEMENTS

The authors have no relevant financial or non-financial interests to disclose.

DATA AVAILABILITY STATEMENT

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

ORCID

Joris Pries https://orcid.org/0000-0002-4429-6531

REFERENCES

Artin, M. (2011). *Algebra* (2nd ed.). Pearson Education, London, UK.

Buckley, C., O'Reilly, M., Whelan, D., Farrell, A. V., Clark, L., Longo, V., … Caulfield, B. (2017). *Binary classification of running fatigue using a single inertial measurement unit*. Paper presented at the 2017 IEEE 14th International Conference on Wearable and Implantable Body Sensor Networks (BSN), Eindhoven, Netherlands, 197–201.

Dixon, J. D., & Mortimer, B. (1996). *Permutation groups* (Vol. *163*). Springer Science & Business Media, New York, NY.

Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for multi-class classification: An overview. *arXiv:2008.05756*.

Li, L., Yu, Y., Bai, S., Hou, Y., & Chen, X. (2018). An effective two-step intrusion detection approach based on binary classification and k -nn. *IEEE Access*, *6*, 12060–12073.

Pirouz, B., Shaffiee Haghshenas, S., Shaffiee Haghshenas, S., & Piro, P. (2020). Investigating a serious challenge in the sustainable development process: Analysis of confirmed cases of covid-19 (new type of coronavirus) through a binary classification using artificial intelligence and regression analysis. *Sustainability*, *12*, 2427.

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, *45*, 427–437.

Tharwat, A. (2021). Classification assessment methods. *Applied Computing and Informatics*, *17*, 168–192. https://doi.org/10.1016/j.aci.2018.08.003

van de Bijl, E., Klein, J., Pries, J., Bhulai, S., Hoogendoorn, M., & van der Mei, R. (2022). *The dutch draw: Constructing a universal baseline for binary prediction models*. Retrieved from. https://arxiv.org/abs/2203.13084

Wheelus, C., Bou-Harb, E., & Zhu, X. (2018). *Tackling class imbalance in cyber security datasets*. Paper presented at the 2018 IEEE International Conference on Information Reuse and Integration (IRI), Salt Lake City, UT, 229–232.

How to cite this article: Pries, J., Bijl, E., Klein, J., Bhulai, S., & Mei, R. (2023). The optimal input-independent baseline for binary classification: The Dutch Draw. *Statistica Neerlandica*, 1–12. https://doi.org/10.1111/stan.12297