



BornFS: Feature Selection with Balanced Relevance and Nuisance and Its Application to Very Large Datasets

Kilho Shin¹^a, Chris Liu², Katsuyuki Maeda¹ and Hiroaki Ohshima³^b

¹Computer Centre, Gakushuin University, Mejiro, Tokyo, Japan

²Deloitte Tohmatsu Cyber LLC., Marunouchi, Tokyo, Japan

³Graduate School of Information Science, University of Hyogo, Kobe, Hyogo, Japan

Keywords: Feature Selection, Categorical Data.

Abstract: In feature selection, we grapple with two primary challenges: devising effective evaluative indices for selected feature subsets and crafting scalable algorithms rooted in these indices. Our study addresses both. Beyond assessing the size and class relevance of selected features, we introduce a groundbreaking index, *nuisance*. It captures class-uncorrelated information, which can muddy subsequent processes. Our experiments confirm that a harmonious balance between class relevance and nuisance augments classification accuracy. To this end, we present the *Balance-Optimized Relevance and Nuisance Feature Selection* (BornFS) algorithm. It not only exhibits scalability to handle large datasets but also outperforms traditional methods by achieving better balance among the introduced indices. Notably, when applied to a dataset of 800,000 Windows executables, using LCC as a preprocessing filter, BornFS slashes the feature count from 10 million to under 200, maintaining a high accuracy in malware detection. Our findings shine a light on feature selection's complexities and pave the way forward.

1 INTRODUCTION

Feature selection is pivotal in machine learning and gains prominence with increasing data. With big data introducing myriad features, efficient feature selection is paramount. For example, in bioinformatics, datasets may feature thousands of genes, making identifying disorder-linked genes a feature selection challenge. As machine learning progresses, refined feature selection becomes critical.

While dimensionality reduction is often equated with feature selection, they are distinct concepts. Dimensionality reduction is generally applied to continuous features, crafting new dimensions that encapsulate the essence of the original dataset. In contrast, feature selection, which has undergone extensive study particularly for categorical features, involves choosing from the existing features.

Deep neural networks are also gaining attention for their ability to effectively extract features from data. However, these extracted features are typically not human-interpretable, contrasting with feature se-


lection, which directly identifies important features.


This paper delves into categorical feature selection, crucial for tasks like pinpointing disease-causing genes. Categorical feature selection has seen thorough exploration in literature, as demonstrated by (Almuallim and Dietterich, 1994; Hall, 2000; Yu and Liu, 2003; Peng et al., 2005; Zhao and Liu, 2007; Shin et al., 2015).

This paper addresses two primary challenges in modern categorical feature selection:

1. Feature Evaluation Indices: We've introduced robust indices for assessing feature subset quality. Apart from the conventional *high class relevance* and *few feature count* indices, a novel *low nuisance* metric is presented. This quantifies non-class-related information in the feature subset.
2. Algorithmic Innovation with Scalability: We present BORNFS, an algorithm tailored to balance the three indices and adeptly process large datasets.

This paper begins with an overview of categorical feature selection algorithms, introducing the new *nuisance* index and an aggregate index for class relevance balance. We conduct two experiments: first,

^a <https://orcid.org/0000-0002-0425-8485>

^b <https://orcid.org/0000-0002-9492-2246>

analyzing the balance in compact datasets and then in 16 larger benchmark datasets. Results show a correlation between balance and predictive accuracy, leading to our new mechanism, BORNFS. We compare its performance with mRMR and LCC, concluding with an experiment using BORNFS on a dataset with over 10 million features and 800,000 instances.

2 MATHEMATICAL NOTATIONS

A dataset \mathcal{D} is represented as a matrix with n_I rows corresponding to instances and $n_F + 1$ columns corresponding to features. The rightmost column represents the class labels of the instances, while the other columns are denoted as F_1, \dots, F_{n_F} . We focus on datasets where all features are categorical, thus, the value set $R(F)$ for a feature F , including F_i or the class feature C , is finite.

In this context, the dataset defines a probability measure over the sample space $\Omega_{\mathcal{D}} = R(F_1) \times \dots \times R(F_{n_F}) \times R(C)$. The empirical probability for a vector $\mathbf{v} \in \Omega_{\mathcal{D}}$ is given by $P_{\mathcal{D}}[F_1, \dots, F_{n_F}, C = \mathbf{v}] = \frac{n(\mathbf{v})}{n_I}$, where $n(\mathbf{v})$ denotes the number of occurrences of \mathbf{v} . Here, both features and the class feature are considered random variables, allowing the application of probability-based metrics such as entropy and mutual information.

3 RELATED WORK

3.1 Relevance, Redundancy, Interaction

Many studies posit the ultimate goal of feature selection as the identification of the smallest subset of features that exhibits the highest class relevance. This class relevance pertains to the collective correlation of a feature subset \mathcal{F} with class labels. Various measures, such as mutual information $I(\mathcal{F}; C)$, are used to quantify this relevance.

To achieve this goal, several algorithms in the literature, including CFS (Hall, 2000), RELIEF-F (Kira and Rendell, 1992), and mRMR (Zhao et al., 2019), capitalize on the concept of internal redundancy. This concept refers to the shared information among features, where reducing redundancy helps decrease the feature count. For instance, when considering the inclusion of a feature F in a feature set \mathcal{F} , these algorithms assess the balance between redundancy gain and relevance gain. These gains can be estimated as $I(\mathcal{F}; F)$ and $I(\mathcal{F}, F; C) - I(\mathcal{F}; C)$, respectively.

In evaluating $I(\mathcal{F}; F)$, the approximation

$I(\mathcal{F}; F) \approx \sum_{F' \in \mathcal{F}} I(F'; F)$ is common, enhancing time efficiency but potentially missing critical feature interactions. Advanced methods like INTERACT by (Zhao and Liu, 2007) and its successors like LCC (Shin et al., 2011; Shin et al., 2017) have refined feature selection, considering these interactions for improved relevance and feature count.

3.2 Advances in Time Efficiency

To select k features that maximize class relevance, exhaustive search typically requires $O(n_F^k)$ time. Algorithms like mRMR, CFS, and RELIEF-F improve this to $O(k^2 n_F n_I)$, effectively balancing relevance and redundancy. Even more efficient, LCC further reduces the time complexity to $O(n_F n_I \log n_F)$, despite incorporating feature interaction into the selection process.

In practical terms, LCC is currently the only advanced feature selection algorithm known to scale effectively to big data, as evidenced by its performance on the DOROTHEA dataset of 100,000 features and 800 instances, where it operates significantly faster than mRMR in Weka, completing the task in under 300 milliseconds versus more than 3,500 seconds.

3.3 The Algorithm of LCC

To achieve both theoretical and practical time efficiency while grounding on the indices introduced, we develop BORNFS, building upon the algorithmic efficiency and foundation established by LCC.

<p>Input: A dataset \mathcal{D} described by $\mathcal{F}_{\mathcal{D}} \cup \{C\}$; and a threshold $t \in [0, 1]$.</p> <p>Output: A minimal feature subset $\mathcal{F} \subseteq \mathcal{F}_{\mathcal{D}}$ with $1 - \text{Br}(\mathcal{F}; C) \geq t(1 - \text{Br}(\mathcal{F}_{\mathcal{D}}; C))$.</p> <ol style="list-style-type: none"> 1 Number the features of $\mathcal{F}_{\mathcal{D}}$ so that F_1, \dots, F_{n_F} are in an increasing order of $\text{SU}(F_i; C)$; 2 $\mathcal{F} = \emptyset$ and $i = 1$; 3 while $i \leq n_F$ do 4 Let $j = \min\{j \in [i, n_F] \mid 1 - \text{Br}(\mathcal{F} \cup \mathcal{F}_{\mathcal{D}}[j + 1, n_F]; C) < t(1 - \text{Br}(\mathcal{F}_{\mathcal{D}}; C))\}$; 5 $\mathcal{F} = \mathcal{F}_{\mathcal{D}} \cup \{j\}$ and $i = j + 1$; 6 end 7 return \mathcal{F}.
--

Algorithm 1: LCC.

Algorithm 1 outlines the LCC algorithm. To determine the class relevance, it utilizes the complement of the Bayesian risk:

$$1 - \text{Br}(\mathcal{F}; C) = \sum_{\mathbf{v} \in R(\mathcal{F})} \max_{c \in R(C)} \Pr[C = c \mid \mathcal{F} = \mathbf{v}].$$

Exploiting the relevance measure's monotonicity, that is, $1 - \text{Br}(\mathcal{F}; C) \geq 1 - \text{Br}(\mathcal{G}; C)$ for $\mathcal{F} \supset \mathcal{G}$, LCC employs binary search to implement Line 4.

The sorting step (Line 1) utilizes the normalized mutual information metric $SU(F;C) = \frac{2 \cdot I(F;C)}{H(F)+H(C)}$. Introduced empirically to boost classification accuracy, as suggested in (Zhao and Liu, 2007), this sorting method has proven effective. Features ranked earlier by this metric are more likely to be eliminated in Line 4, thus optimizing the selection process and enhancing the overall accuracy of the algorithm.

4 THE THIRD INDEX – NUISANCE

4.1 Definition

In addition to established indices such as class relevance and feature count, we introduce an additional criterion: *nuisance*.

The term *nuisance* denotes the portion of information within a feature subset unrelated to class labels. As the primary goal of feature selection is to enhance the representational ability of features for class labels, any data not contributing to this objective is considered redundant, potentially leading to inaccurate predictions. Nuisance misleads classifiers by introducing irrelevant information. It can be quantified in various ways, including:

- Conditional Entropy: $H(\mathcal{F} | C) = H(\mathcal{F}) - I(\mathcal{F}; C)$
- Inverted Bayesian Risk $Br(C; \mathcal{F})$:
- Ratio: $\frac{H(\mathcal{F})}{I(\mathcal{F}; C)}$.

4.2 Balancing Relevance and Nuisance

To effectively balance class relevance and nuisance, a robust measure is necessary. Our study utilizes the harmonic mean of $\frac{I(\mathcal{F};C)}{I(\mathcal{F}_D;C)}$ for normalized class relevance and $\frac{I(\mathcal{F};C)}{H(\mathcal{F})}$ as the reciprocal of nuisance’s ratio representation as the primary metric. This metric is particularly chosen for its sensitivity to both relevance and nuisance, ensuring that it approaches zero in cases of low relevance or high nuisance. It is formulated as:

$$\mu_H(\mathcal{F};C) = \frac{2 \cdot I(\mathcal{F};C)}{I(\mathcal{F}_D;C) + H(\mathcal{F})}. \quad (1)$$

However, our methodology is flexible and not limited to this specific metric alone. Alternative methods to quantify nuisance and various functions to evaluate the balance between relevance and nuisance are also viable and can be integrated into our approach, allowing for adaptability and optimization according to different dataset characteristics and analytical needs.

The selected metric, $\mu_H(\mathcal{F};C)$, has the following properties:

1. $\mu_H(\mathcal{F};C) = 0$ is equivalent to $I(\mathcal{F};C) = 0$;
2. $\mu_H(\mathcal{F};C) = 1$ implies $I(\mathcal{F};C) = I(\mathcal{F}_D;C) = H(\mathcal{F})$;
3. When $I(\mathcal{F}_D;C) = H(C)$, $\mu_H(\mathcal{F};C)$ coincides with $SU(\mathcal{F};C)$.

5 EFFICACY VALIDATION OF THE PROPOSED INDICES

We evaluated three indices, particularly the balance index μ_H , across two experiments linking μ_H scores to predictive accuracy. The first explored all feature subsets in compact datasets, while the second assessed sampled subsets in larger benchmark datasets.

5.1 Data Preparation

Before experiments, datasets underwent discretization into ten equal parts and binarization, resulting in binary features, except for class variables, using one-hot encoding.

5.2 Evaluation of Predictive Accuracy

To evaluate the classification power of each feature subset, we perform five-fold cross-validation on the narrowed dataset using LightGBM (LGBM) and Multiple Layer Perceptron (MLP) classifiers.

5.2.1 Experiment 1: Exhaustive Investigation with Small Datasets

For this experiment, we work with the three MONKS datasets. While each of these datasets contains the same 432 instances and is described by six features, their annotations differ, as outlined in (Blake and Merz, 1998). After binarization, these datasets are represented by 17 binary features alongside a class variable.

For each binarized MONKS dataset, we examine all $2^{17} - 1$ non-empty feature subsets. These subsets are evaluated based on their μ_H scores and AUC-ROC values using the LGBM and MLP classifiers.

Figure 1 illustrates the correlation between the $\mu_H(\mathcal{F};C)$ scores (represented on the x-axis) and the AUC-ROC scores (on the y-axis) for feature subsets. Given that $2^{17} - 1$ represents a considerably large number, the plot is limited to those \mathcal{F} with the highest $I(\mathcal{F};C)$. The figure demonstrates that as the $\mu_H(\mathcal{F};C)$ values increase, the AUC-ROC scores increasingly

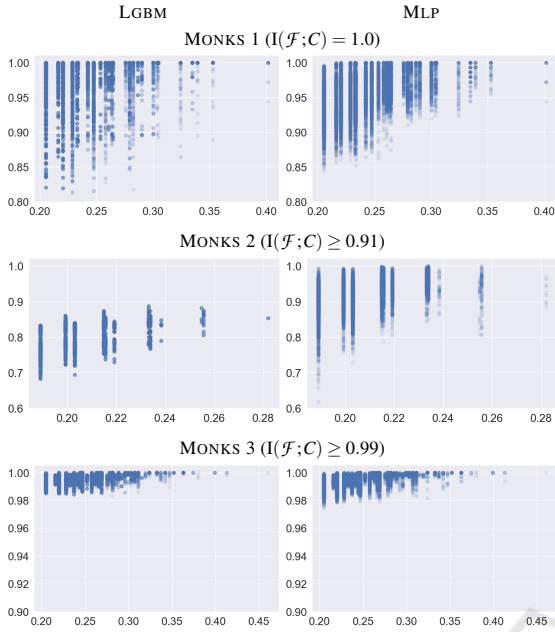


Figure 1: Relation between μ_H (x axis) and AUC-ROC (y axis) for the MONKS datasets.

converge within narrower and elevated ranges. This observation implies a substantial probability of attaining feature subsets with enhanced predictive performance by using high $\mu_H(\mathcal{F}; C)$ scores as a criterion for selection.

5.2.2 Experiment 2: Investigation with Larger Datasets

In our second experiment, we examine 15 larger datasets listed in Table 1. Ten were part of feature selection challenges at NIPS 2003 (NIPS, 2003) and WCCI 2006 (WCCI, 2006). One comes from the KDD 1999 network intrusion detection challenge (KDD, 1999), while others are from the UCI repository (Blake and Merz, 1998).

Given the larger number of features in these datasets, an exhaustive evaluation similar to our first experiment isn't viable. To pinpoint succinct feature subsets, \mathcal{F} , with prominent μ_H scores, we turn to a hill-climbing sampling method, as outlined in Algorithm 2. This approach yields subsets $\mathcal{F}_1 \subset \dots \subset \mathcal{F}_n$ for each dataset, where the size of subset \mathcal{F}_i is i and it meets the condition $\mu_H(\mathcal{F}_i; C) < \mu_H(\mathcal{F}_{i+1}; C)$.

Figure 2 showcases the relationship between the AUC-ROC scores (on the y -axis) and the μ_H scores (plotted on the x -axis) for each feature subset \mathcal{F}_i produced by Algorithm 2. Excluding two exceptions—DEXTER with LGBM and DOROTHEA with MLP—there is a discernible positive correlation between the predictive performance of the classifiers

Table 1: Attributes of datasets.

Dataset	#F	#BF	#I	Ref.
AD	1,559	3,137	3,279	(Blake and Merz, 1998)
ADA	49	134	4,147	(WCCI, 2006)
ARCENE	10,001	83,950	100	(NIPS, 2003)
DEXTER	20,001	35,924	300	(NIPS, 2003)
DOROTHEA	100,001	100,001	800	(NIPS, 2003)
GINA	971	9,683	3,153	(WCCI, 2006)
GISSETTE	5,001	39,809	6,000	(NIPS, 2003)
HIVA	1,618	3,235	3,845	(WCCI, 2006)
KDD	42	346	25,192	(KDD, 1999)
KR	37	74	3,196	(Blake and Merz, 1998)
MADOLON	501	4,972	2,000	(NIPS, 2003)
MUSHROOM	23	114	8,124	(Blake and Merz, 1998)
NOVA	16,970	29,368	1,754	(WCCI, 2006)
SPAMBASE	58	134	4,601	(Blake and Merz, 1998)
SYLVA	217	746	13,086	(WCCI, 2006)

Input: A dataset described by a feature set \mathcal{F}_D and C .

Output: Feature subsets $\mathcal{F}_1 \subset \dots \subset \mathcal{F}_n$ with $|\mathcal{F}_i| = i$.

```

1 Let  $n = 0$  and  $\mathcal{F}_0 = \emptyset$ ;
2 while true do
3   Let  $F \in \arg \max \mu_H(\mathcal{F}_n, F; C)$ ;
4   if  $\mu_H(\mathcal{F}_n, F; C) > \mu_H(\mathcal{F}_n; C)$  then
5     Let  $n = n + 1$ ;
6      $\mathcal{F}_n = \mathcal{F}_{n-1} \cup \{F\}$ ;
7   else
8     return  $\mathcal{F}_1, \dots, \mathcal{F}_n$ 
9   end
10 end
    
```

Algorithm 2: A hill-climbing sampling.

and the μ_H scores of their corresponding feature subsets. Especially with MLP, this positive correlation is strongly evident across all the datasets.

5.3 Conclusions from the Experiments

Both experiments affirm the effectiveness of nuisance in feature selection and the metric μ_H for capturing feature set nature.

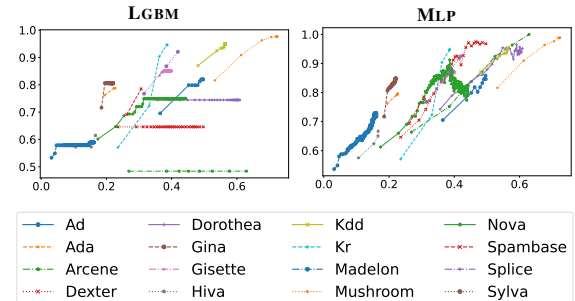


Figure 2: Relation between μ_H (x axis) and AUC-ROC (y axis) for larger datasets.

6 A NEW ALGORITHM – BORNFS

We introduce the first categorical feature selection algorithm assessing nuisance, class relevance, and feature count, named *Balance-Optimized Relevance and Nuisance Feature Selection* (BORNFS). It's highly time-efficient, scalable to large datasets.

6.1 Design Policies of BORNFS

Class relevance ought to be given the highest priority among the three indices. This prioritization ensures that the most significant factor influencing the algorithm's effectiveness is appropriately emphasized.

Specifically, $I(\mathcal{F}; C)$ gauges the information crucial for accurate understanding and effective learning. A feature set's significance is compromised if its class relevance is low, regardless of its size or nuisance level.

To manage class relevance, we introduce *t-abundance* and *t-minimality*, guided by a threshold $t \in (0, 1]$. Our algorithm seeks to yield feature subsets adhering to these criteria.

Definition 1. Given a dataset \mathcal{D} described by a feature set $\mathcal{F}_{\mathcal{D}}$ and a threshold $t \in (0, 1]$, the criteria of *t-abundance* and *t-minimality* for a feature subset $\mathcal{F} \subseteq \mathcal{F}_{\mathcal{D}}$ are defined as:

1. **t-abundance:** $\frac{I(\mathcal{F}; C)}{I(\mathcal{F}_{\mathcal{D}}; C)} \geq t$;
2. **t-minimality:** $\frac{I(\mathcal{G}; C)}{I(\mathcal{F}_{\mathcal{D}}; C)} < t$ for any $\mathcal{G} \subsetneq \mathcal{F}$.

The fulfillment of *t-abundance* ensures that the feature subset possesses adequate relevance, while *t-minimality* reduces the feature count. Therefore, our revised objective is to develop a fast algorithm that selects feature subsets which are both *t-abundant* and *t-minimal*, and that also exhibit low nuisance.

6.2 The Key Idea

In developing BORNFS, we adapted the LCC framework for its significant speed and ability to scale to very large datasets. The key feature contributing to the efficiency of LCC is its iterative binary search routine, which selects a single feature per iteration. For BORNFS, we customized it to be guided by the *t-abundance* principle.

Moreover, before executing each binary search, BORNFS arranges the features within the search range based on an estimation of potential gains in relevance and nuisance. To optimize the balance, BORNFS leverages the property of the LCC search framework where features positioned later are more likely to be selected.

When using \mathcal{F} to denote the set of features selected prior to the next search, we express the potential gain in relevance as $\Delta_r(F; \mathcal{F})$ and the potential gain in nuisance as $\Delta_n(F; \mathcal{F})$ as follows:

$$\Delta_r(F; \mathcal{F}) = I(\mathcal{F}, F; C) - I(\mathcal{F}; C); \quad (2)$$

$$\Delta_n(F; \mathcal{F}) = H(F) - I(F; \mathcal{F}, C). \quad (3)$$

The algorithm is designed to swiftly update these values following the addition of a new feature to \mathcal{F} .

While the introduction of this sorting procedure causes computational overhead, it does not affect the overall asymptotic time complexity, as detailed in (Section 7.1).

6.3 Algorithm Description

Algorithm 3 describes BORNFS.

6.3.1 Fundamental Structure

The algorithm iteratively selects features, choosing one from a search range during each cycle. Let's define:

- \mathcal{F} as the set of previously chosen features;
- The current search range as F_s, \dots, F_{n_F} .

When the algorithm calculates $r_i = \frac{I(\mathcal{F}, F_{i+1}, \dots, F_{n_F}; C)}{I(\mathcal{F}_{\mathcal{D}}; C)}$ for $i \in [s, n_F]$, if $r_i \geq t$, it considers features F_s, \dots, F_i as insignificant according to *t-abundance*.

As the ratio decreases with increasing i , a binary search efficiently find the value:

$$s^* = \min \left\{ i \in [s, n_F] \mid \frac{I(\mathcal{F}, F_{i+1}, \dots, F_{n_F}; C)}{I(\mathcal{F}_{\mathcal{D}}; C)} < t \right\}.$$

Upon finding s^* , BORNFS includes F_{s^*} in \mathcal{F} and shifts the search range to F_{s^*+1}, \dots, n_F .

6.3.2 Sorting of Features in a Search Range

In the iterative process of identifying s^* , features positioned earlier in the search range are more likely to be omitted, as indicated by previous studies (Zhao and Liu, 2007; Shin et al., 2017). Leveraging this insight, BORNFS sorts features based on their potential impact on the balance between relevance and nuisance. To assess the impact, BORNFS employs one of the following:

Ratio: $\Gamma_R(F; \mathcal{F}) = \frac{\Delta_r(F; \mathcal{F})}{\Delta_n(F; \mathcal{F})};$

Harmonic mean: $\Gamma_H(F; \mathcal{F}) =$

$$\frac{2(I(\mathcal{F}; C) + \Delta_r(F; \mathcal{F}))}{I(\mathcal{F}_{\mathcal{D}}; C) + H(\mathcal{F}) + \Delta_r(F; \mathcal{F}) + \Delta_n(F; \mathcal{F})}.$$

Furthermore, the execution speed of BORNFS is impacted by the frequent sorting process using either Γ_R or Γ_H . To mitigate this, BORNFS introduces a *hop* parameter, denoted as h , which dictates that sorting occurs every h iterations. Essentially, LCC can be considered a variant of BORNFS with an infinite hop value ($h = \infty$).

```

Input: A dataset described by  $\mathcal{F}_D \cup \{C\}$ ;  $t \in (0, 1]$ ;
          one of  $\Gamma_R$  and  $\Gamma_H$ ; and a hop  $h \in \mathbb{N}$ .
Output: A  $t$ -abundant and  $t$ -minimal  $\mathcal{F} \subseteq \mathcal{F}_D$ .
1  Let  $\mathcal{F} = \emptyset$ . Let  $s = 1$  and  $c = 0$ . while  $s \leq n_F$  do
2  |   if  $c \bmod h \equiv 0$  then
3  |   |   Renumber  $F_s, \dots, F_{n_F}$  in an increasing order
3  |   |   of  $\Gamma(F; \mathcal{F})$ ;
4  |   else
5  |   end
6  |   if  $\exists s^* = \min \left\{ i \in [s, n_F] \mid \frac{1(\mathcal{F}, F_{i+1}, \dots, F_{n_F}; C)}{1(\mathcal{F}_D; C)} < t \right\}$ 
6  |   |   then
7  |   |   |   Let  $\mathcal{F} = \mathcal{F} \cup \{F_{s^*}\}$ ;
8  |   |   |   Let  $s = s^* + 1$ .;
9  |   |   else
10 |   |   |   return  $\mathcal{F}$ .
11 |   |   end
12 |   |   Increment  $c$  by one.;
13 end
14 return  $\mathcal{F}$ .
    
```

Algorithm 3: Born Feature Selection (BORNFS).

7 COMPARING BORNFS WITH LCC AND mRMR

In this section, we juxtapose LCC, mRMR, and BORNFS concerning asymptotic computational complexity, real-time efficiency, and the quality of their outputs.

Algorithm	Source
BORNFS	Java executable codes available at (Shin and Maeda, 2023)
mRMR	C++ codes crafted by its creators, sourced from (Peng, 2007).
LCC	Java executable codes by its creators, sourced from (Shin et al., 2015)

7.1 Asymptotic Time Complexity

The average time complexity of mRMR is $O(k^2 n_F)$, where k denotes the number of features to select, and that of LCC is $O(n_F \log n_F)$. In this section, we demonstrate that the time complexity of BORNFS is also $O(n_F \log n_F)$.

To analyze, for the i th iteration of selecting a feature, \mathcal{F}_{i-1} denotes the previously selected feature set,

and ℓ_i is a random variable representing the size of the search range in the current iteration. For ease of analysis, we consider ℓ_i as a continuous variable spanning the range $[0, n_F]$. When $f_i(x)$ is the probability density function of ℓ_i , $f_i(x | \ell_{i-1} = t) = \frac{1}{t}$ holds for $x \in (n_F - t, n_F]$. Thus, the expected size of the search range, ℓ_i , is:

$$\mathbb{E}(\ell_i) = \int_0^{n_F} x f_i(x) dx = \frac{n_F}{2^{i-1}}.$$

Therefore, during the binary search of the i th iteration, mutual information values – requiring $O((i + 2^{-i} n_F) n_F)$ time for each – are computed $\log \frac{n_F}{2^{i-1}}$ times on average. The following equation offers an approximation of the average time complexity:

$$\sum_{i=1}^{\lceil \log_2 n_F \rceil} \left(i + \frac{n_F}{2^i} \right) n_F \log \frac{n_F}{2^{i-1}} \approx \frac{1}{6} n_F \log_2^3 n_F + n_F n_F \log_2 n_F.$$

On the other hand, during the sorting procedure, BORNFS computes $\Delta_r(F; \mathcal{F}_{i-1})$ and $\Delta_n(F; \mathcal{F}_{i-1})$ for $\mathbb{E}(\ell_i)$ features F , resulting in a time complexity of $O(i \cdot 2^{1-i} n_F n_F)$. Additionally, the time complexity of sorting $\mathbb{E}(\ell_i)$ features is $O(2^{1-i} n_F \log(2^{1-i} n_F))$. They accumulate to $O(n_F n_F)$ and $O(n_F \log n_F)$ across $i = 1, 2, \dots, \lceil \log_2 n_F \rceil$ respectively.

Finally, it is concluded that the overall average time complexity of BORNFS is $O(n_F n_F \log n_F)$.

7.2 Run-Time Efficiency

We assessed runtimes using datasets presented in Table 1, as shown in Figure 3. The left chart shows the actual runtimes, while the right charts them relative to BORNFS with Γ_R . Notably, the drop in efficiency from LCC is limited to merely fivefold.

7.3 Quality of Outputs

We ran BORNFS and LCC for $t = 0.5, 0.6, 0.7, 0.8, 0.9, 1.0$ to obtain six feature subsets for each algorithm and each dataset. We configured mRMR to output the top 100 features in each run, resulting in 100 feature subsets for each dataset.

From the obtained feature subsets, we identified those with the highest scores of μ_H . Figure 4 displays the attributes of the identified feature subsets \mathcal{F} , including: (a) $\mu_H(\mathcal{F}; C)$; (b) $I(\mathcal{F}; C)$; and (c) $|\mathcal{F}|$. These attributes are further analyzed in Table 2 and Figure 5, which present a comparison of the algorithms based on the averages across the datasets relative to BORNFS with Γ_R . Specifically, Figure 5 illustrates the balance among relevance, nuisance and feature count using a radar chart.

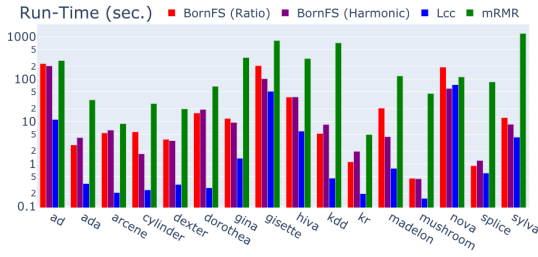
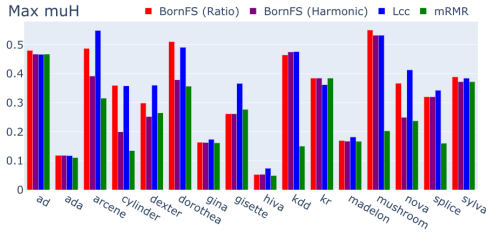
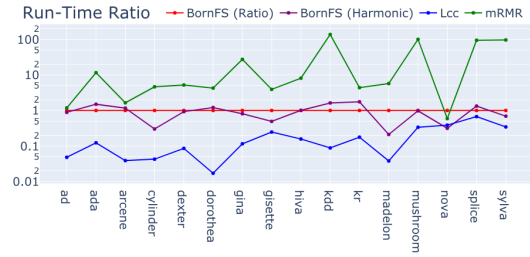
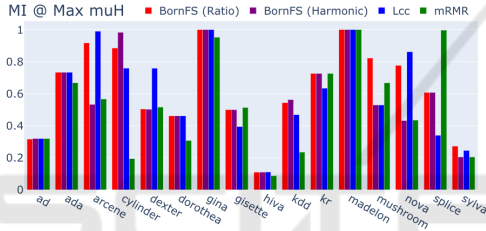
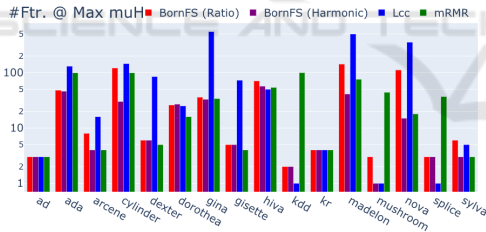


Figure 3: Run-time in logarithmic scale (iMac Pro with 2.5GHz 14-core Xeon W, averages of ten trials).

(a) Maximum $\mu_H(\mathcal{F}; C)$.(b) Class relevance: $I(\mathcal{F}; C)$.(c) Feature count: $|\mathcal{F}|$.Figure 4: Results when $\mu_H(\mathcal{F}; C)$ is maximized.Table 2: Averaged ratios when $\mu_H(\mathcal{F}; C)$ is maximized.

	BORNFS Γ_R	BORNFS Γ_H	LCC	mRMR
$\mu_H(\mathcal{F}; C)$	1.00	0.90	1.06	0.76
$I(\mathcal{F}; C)$	1.00	0.92	0.91	0.87
$H(\mathcal{F} C)$	1.00	1.02	0.68	1.95
$ \mathcal{F} $	1.00	0.50	3.12	3.84

The observation is that LCC records low nuisance scores across datasets without considering nuisance in feature selection. Despite this, LCC selects more features than others; for example, in the GINA dataset, BORNFS selects around 30 features while LCC chooses over 500.

The difference arises from the feature selection

methodologies. LCC uses a pre-established feature order, leading to redundant feature selection. In contrast, BORNFS updates its feature order based on previously selected features, reducing redundancy and creating a more streamlined feature set.

8 APPLICATION TO A HUGE REAL DATASET

To assess BORNFS's performance on large datasets, experiments were conducted using the EMBER repository, intended to foster malware detection research. The dataset includes over 10 million features and comprises 300,000 benign and 300,000 malicious samples for training, along with 100,000 of each for testing.

Due to the massive feature count over 10 million, a two-step approach was adopted for BORNFS to efficiently handle the dataset:

- Initial Reduction:** BORNFS was deployed with $h = \infty$ and $t = 1.0$ on the complete dataset to significantly reduce the feature count. This step likely leaves a substantial number of redundant features.
- Redundancy Elimination:** To remove redundant features from the initial reduction, BORNFS was reapplied with $h = 10$ to the reduced feature set. The threshold parameter t was varied among 1.0, 0.95, and 0.9 for this purpose.

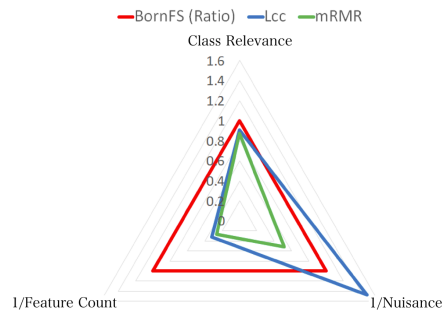


Figure 5: Radar chart.

Table 3: Two-step application of BORNFS changing the hop parameter values.

	Initial	$h = \infty$	$h = 10$		
		$t = 1.0$	$t = 1.0$	0.95	0.9
Feature count	10,868,073	541	540	155	50
AUC-ROC LGBM		0.97	0.97	0.96	0.86
AUC-ROC MLP		0.98	0.98	0.98	0.88
Runtime (min.)		24.33	515.92	69.09	9.57

To determine the predictive capability of the feature subsets from the two-step approach, LGBM and MLP classifiers were used on the refined dataset, and AUC-ROC scores were calculated. Remarkably, with the threshold parameter t at 0.95, AUC-ROC scores were high, reaching 0.96 for LGBM and 0.98 for MLP, despite reducing the feature count from over 10 million to just 155. These results are detailed in Table 3.

The experiment was conducted on an AWS r5.4xlarge instance, which has 16 vCPUs and 128GB of memory. The total time taken for the experiment with $t = 0.95$ was 93 minutes, comprising 24 minutes for the first step and an additional 69 minutes for the second step.

9 CONCLUSION

Beyond the established metrics for evaluating the goodness of feature selection, that is class relevance and feature count, we introduced *nuisance* as a third metric. This metric measures the amount of irrelevant data that can distort understanding. Our μ_H score evaluates the balance between relevance and nuisance and has shown a positive correlation with classifier performance. Our method, BORNFS, harmonizes these metrics and has outperformed others, including LCC and mRMR, on large datasets. It maintains accuracy while reducing the number of features.

ACKNOWLEDGEMENTS

This work was supported by JSPS KAKENHI Grant Number 21H05052, and Number 21H03775.

REFERENCES

Almuallim, H. and Dietterich, T. G. (1994). Learning boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69(1 - 2).

Anderson, H. S. and Roth, P. (2018). EMBER: an open dataset for training static PE malware machine learning models. *CoRR*, abs/1804.04637.

Blake, C. S. and Merz, C. J. (1998). UCI repository of machine learning databases. Technical report, University of California, Irvine.

Hall, M. A. (2000). Correlation-based feature selection for discrete and numeric class machine learning. In *ICML2000*, pages 359–366.

KDD (1999). *KDD Cup 1999: Computer network intrusion detection*.

Kira, K. and Rendell, L. (1992). A practical approach to feature selection. In *ICML1992*, pages 249–256.

NIPS (2003). *Neural Information Processing Systems Conference 2003: Feature selection challenge*.

Peng, H. (2007). mRMR (minimum Redundancy Maximum Relevance Feature Selection). <http://home.penglab.com/proj/mRMR/>. Online; accessed 29-February-2020.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information: Criteria of max-dependency, max-relevance and min-redundancy. *IEEE TPAMI*, 27(8).

Shin K. and Maeda K. (2023). Temporarily open at <https://00m.in/Xud17>.

Shin, K., Fernandes, D., and Miyazaki, S. (2011). Consistency measures for feature selection: A formal definition, relative sensitivity comparison, and a fast algorithm. In *IJCAI2011*, pages 1491–1497.

Shin, K., Kuboyama, T., Hashimoto, T., and Shepard, D. (2015). Super-CWC and super-LCC: Super fast feature selection algorithms. In *IEEE BigData 2015*, pages 61–67.

Shin, K., Kuboyama, T., Hashimoto, T., and Shepard, D. (2017). sCWC/sLCC: Highly scalable feature selection algorithms. *Information*, 8(4).

WCCI (2006). *IEEE World Congress on Computational Intelligence 2006: Performance prediction challenge*.

Yu, L. and Liu, H. (2003). Feature selection for high-dimensional data: a fast correlation-based filter solution. In *ICML2003*.

Zhao, Z., Anand, R., and Wang, M. (2019). Maximum Relevance and Minimum Redundancy Feature Selection Methods for a Marketing Machine Learning Platform. *IEEE DSAA2019*.

Zhao, Z. and Liu, H. (2007). Searching for interacting features. In *IJCAI2007*, pages 1156 – 1161.