Article

IMMIGRATE: A Margin-based Feature Selection Method with Interaction Terms

Ruzhang Zhao[1], Pengyu Hong [2,*] and Jun S. Liu[3,*]

[1] Department of Biostatistics, Bloomberg School of Public Health, Johns Hopkins University; rzhao@jhu.edu
[2] Department of Computer Science, Brandeis University; hongpeng@brandeis.edu
[3] Department of Statistics, Harvard University; jliu@stat.harvard.edu
* Correspondence: hongpeng@brandeis.edu, Tel.: +1-781-736-2729; jliu@stat.harvard.edu, Tel.: +1-617-495-1600

Version February 19, 2020 submitted to Entropy

Abstract: Traditional hypothesis-margin researches focus on obtaining large margins and feature selection. In this work, we show that the robustness of margins is also critical and can be measured using entropy. In addition, our approach provides clear mathematical formulations and explanations to uncover feature interactions, which is often lack in large hypothesis-margin based approaches. We design an algorithm, termed IMMIGRATE, for training the weights associated with the interaction terms. IMMIGRATE simultaneously utilizes both local and global information and can be used as a base learner in Boosting. We evaluate IMMIGRATE in a wide range of tasks, in which it demonstrates exceptional robustness and achieves the state-of-the-art results with high interpretability.

Keywords: hypothesis-margin; feature selection; entropy; IMMIGRATE

---

1. Introduction

Feature selection is one of the most fundamental problems in machine learning and pattern recognition [1]. The Relief algorithm by Kira and Rendell [2] is one of the most successful feature selection algorithms. It can be interpreted as an online learning algorithm that solves a convex optimization problem with a hypothesis-margin-based cost function. Instead of deploying exhaustive or heuristic combinatorial searches, Relief decomposes a complex, global and nonlinear classification task into a simple and local one. Following the large hypothesis-margin principle for classification, Relief calculates the weights of features, which can be used for feature selection. The hypothesis-margin of an instance x with respect to (w.r.t.) a set of samples $\mathcal{P}$ is later formerly defined in Gilad-Bachrach *et al.* [3] as $\frac{1}{2}(\|\vec{x} - \mathrm{NM}(\vec{x})\| - \|\vec{x} - \mathrm{NH}(\vec{x})\|)$, where $\mathrm{NH}(\vec{x})$ and $\mathrm{NM}(\vec{x})$ denote the nearest samples to $\vec{x}$ in $\mathcal{P}$ with the same and different labels, respectively. The large hypothesis-margin principle has motivated several successful extensions of the Relief algorithm. For example, ReliefF [4] uses multiple nearest neighbors. Simba [3] recalculates the nearest neighbors every time the feature weights are updated. Yang *et al.* [5] consider global information to improve Simba. I-RELIEF [6] identifies the nearest hits and misses in a probabilistic manner, which forms a variation of hypothesis-margin. LFE [7] extends Relief from feature selection to feature extraction using local information. IM4E is proposed by Bei and Hong [8] to balance margin-quantity maximization and margin-quality maximization. Both approaches in Sun and Wu [7], Bei and Hong [8] use a variation of hypothesis-margin proposed in Sun and Li [6].

The Relief-based algorithms indirectly consider feature interactions by normalizing the feature weights [9], which, however, cannot directly reflect the natural effects of associations and hence results in poor interpretability on the effects of feature interactions. For example, Relief and many of its extensions cannot tell whether a high weight of a certain feature is caused by its linear effect or its interaction with other features [9]. Similarly, these methods cannot clearly reveal the influence of interaction terms on classification. In particular, the degree of such influence cannot be measured.

To this end, we propose the ***I**terative **M**ax-**MI**n entropy mar**G**in-maximization with inte**RA**ction **TE**rms* algorithm (IMMIGRATE, henceforth). IMMIGRATE directly measures the influence of feature interactions and delivers the following novelties. First, when defining our hypothesis-margin, we introduce a new trainable quadratic-Manhattan measurement to capture interaction terms, which interprets interaction importance directly. Second, we take advantage of the margin stability by measuring the underlying entropy based on the distributions of instances. Third, we derive an iterative optimization algorithm to efficiently minimize the cost function. Fourth, we design a novel classification method that utilizes the learned quadratic-Manhattan measurement to predict the class of a new instance. Fifth, we design a more powerful approach (i.e., Boosted IMMIGRATE) by using IMMIGRATE as the base learner of Boosting [10]. Sixth, to make IMMIGRATE efficient for analyzing high-dimensional datasets, we take advantage of IM4E [8] to provide an effective initialization.

The rest of the paper is organized as follows. Section 2 explains the foundation of the Relief algorithm. The IMMIGRATE algorithm is explained in Section 3. Section 4 summarizes and discusses the experiments on different datasets. Experimental results show that our approach achieves the state-of-the-art results. Boosted IMMIGRATE outperforms other Boosting classifiers significantly. The computation time of IMMIGRATE is comparable to other popular feature selection methods that consider interaction terms. The paper is concluded and discussed in Section 5 including comparisons with related works.

2. Review: the Relief Algorithm

We first explain a few notations used in the rest of the paper: $\vec{x}_i$ as the i-th instance in the training set $\mathcal{P}$; y_i as the class label of $\vec{x}_i$; N as the size of $\mathcal{P}$; A as the number of features(i.e. attributes); $\vec{w}$ as the feature weight vector; and $|\vec{x}_i|$ as a vector where absolute value operation is element-wise. Relief [2] iteratively calculates the feature weights in $\vec{w}$ (Algorithm 1). The higher a feature weight is, the more relevant the corresponding feature is. After the calculation of feature weights, a threshold is chosen to select relevant features. Relief can be viewed as a convex optimization problem that minimizes the cost function:

$$\begin{aligned} C = \sum_{n=1}^{M} \left(\vec{w}^{\,T}\left|\vec{x}_n - \mathrm{NH}(\vec{x}_n)\right| - \vec{w}^{\,T}\left|\vec{x}_n - \mathrm{NM}(\vec{x}_n)\right|\right), \\ \text{subject to}: \vec{w} \geq 0,\ \|\vec{w}\|_2^2 = 1, \end{aligned} \tag{2.1}$$

where $M(\ll N)$ is a user defined number of randomly chosen training samples, $\mathrm{NH}(\vec{x})$ is the nearest "hit" (from the same class) of $\vec{x}$; $\mathrm{NM}(\vec{x})$ is the nearest "miss" (from a different class) of $\vec{x}$; and $\vec{w}^{\,T}\left|\vec{x}_n - \mathrm{NH}(\vec{x}_n)\right|$ is the weighted Manhattan distance. Denote $\vec{u} = \sum_{n=1}^{M}\left(\left|\vec{x}_n - \mathrm{NH}(\vec{x}_n)\right| - \left|\vec{x}_n - \mathrm{NM}(\vec{x}_n)\right|\right)$. Minimizing the cost function of Relief 2.1 can be solved using the Lagrange multiplier method and the Karush-Kuhn-Tucker conditions [11] to get a close form solution: $\vec{w} = (-\vec{u})^{+}/\|(-\vec{u})^{+}\|_2$, where $(\vec{a})^{+}$ truncates the negative elements to 0. This solution to the original Relief algorithm is important for understanding the Relief-based algorithms.

Algorithm 1 The Original Relief Algorithm

N: the number of training instances.
A: the number of features(i.e. attributes).
M: the number of randomly chosen training samples to update feature weight $\vec{w}$.
Input: a training dataset $\{z_n = (\vec{x}_n, y_n)\}_{n=1,\cdots,N}$.
Initialization: Initialize all feature weights to 0: $\vec{w} = 0$.

for $i = 1$ **to** M **do**
  Randomly select an instance $\vec{x}_i$ and find its $\mathrm{NH}(\vec{x}_i)$ and $\mathrm{NM}(\vec{x}_i)$.
  Update the feature weights by $\vec{w} = \vec{w} - (\vec{x}_i - \mathrm{NH}(\vec{x}_i))^2/M + (\vec{x}_i - \mathrm{NM}(\vec{x}_i))^2/M$,
  where the square operation is element-wise.
Return: $\vec{w}$.

3. IMMIGRATE Algorithm

IMMIGRATE stands for **I**terative **M**ax-**MI**n entropy mar**G**in-maximization with inte**RA**ction **TE**rms algorithm (IMMIGRATE, henceforth). Without loss of generality, we establish the IMMIGRATE algorithm in a general binary classification setting. This formulation can be easily extended to handle multiple class classification problems. Our implementation of IMMIGRATE is applicable to multiple classification tasks. Let the whole data set be $\mathcal{P} = \{z_n | z_n = (\vec{x}_n, y_n), \vec{x}_n \in \mathbb{R}^A, y_n = \pm 1\}_{n=1}^N$; the hit index set of $\vec{x}_n$ be $\mathcal{H}_n = \{j | z_j \in \mathcal{P}, y_j = y_n \,\&\, j \neq n\}$, and the miss index set of $\vec{x}_n$ be $\mathcal{M}_n = \{j | z_j \in \mathcal{P}, y_j \neq y_n\}$.

3.1. Hypothesis-Margin

Given a distance $d(\vec{x}_i, \vec{x}_j)$ between two instances $\vec{x}_i$ and $\vec{x}_j$, a hypothesis-margin [3] is defined as $\rho_{n,h,m} = d(\vec{x}_n, \vec{x}_m) - d(\vec{x}_n, \vec{x}_h)$, where $\vec{x}_h, h \in \mathcal{H}_n$ and $\vec{x}_m, m \in \mathcal{M}_n$ represent the nearest hit and nearest miss for instance $\vec{x}_n$, respectively. We adopt the probabilistic hypothesis-margin defined by Sun and Li [6] as

$$\rho_n = \sum_{m \in \mathcal{M}_n} \beta_{n,m} d(\vec{x}_n, \vec{x}_m) - \sum_{h \in \mathcal{H}_n} \alpha_{n,h} d(\vec{x}_n, \vec{x}_h), \tag{3.2}$$

where $\alpha_{n,h} \geq 0$, $\beta_{n,m} \geq 0$, $\sum_{h \in \mathcal{H}_n} \alpha_{n,h} = 1$, $\sum_{m \in \mathcal{M}_n} \beta_{n,m} = 1$, for $\forall\, n \in \{1, \cdots, N\}$. As in the above design, the hidden random variable $\alpha_{n,h}$ represents the probability that $\vec{x}_h$ is the nearest hit of instance $\vec{x}_n$, while $\beta_{n,m}$ indicates the probability that $\vec{x}_m$ is the nearest miss of instance $\vec{x}_n$.

3.2. Entropy to Measure Hypothesis-Margin Stability

Here, we consider how the distributions of the hits and misses contribute to the stability of the hypothesis-margin(hypothesis-margin quality). That is to say, how the distributions of instances with the same or different labels w.r.t. target instance can get more stable margins.

The probabilities $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$ in Eq. 3.2 represent the distributions of hits and misses. The stability of an instance $\vec{x}_n$'s hypothesis-margin can be defined using its hit probabilities $\{\alpha_{n,h}\}$ and miss probabilities $\{\beta_{n,m}\}$. Let's check the hit entropy and miss entropy, which are defined as $E_{hit}(\vec{x}_n) = -\sum_{h \in \mathcal{H}_n} \alpha_{n,h} \log \alpha_{n,h}$ and $E_{miss}(\vec{x}_n) = -\sum_{m \in \mathcal{M}_n} \beta_{n,m} \log \beta_{n,m}$, respectively. The following two scenarios help to explain the intuition of using the hit entropy and miss entropy. Scenario A(stability): all neighbors are distributed evenly around the target instance; scenario B(instability): the neighbor distribution is highly uneven. An extreme example for scenario B is that one instance is quite close to the target and the rest are quite far away from the target. An easy experiment to test the stability of the distributions of hits and misses is to discard one instance from the system and to check the change degree of hypothesis-margin. In scenario A, if the true nearest hit is discarded, the margin changes slightly since there are many other hits evenly distributed around target. However, in scenario B, the disappearance of the true nearest miss can increase the margin significantly. In details, the disappearance of the true nearest miss makes the other misses have larger probabilities to be the nearest miss($\{\beta_{n,m}\}$), which results in the increase of margin in Eq. 3.2. However, if scenario B works for hits, the margin will decrease accordingly when the true nearest hit disappears. Similarly, if scenario A works for misses, the even distribution will not contribute to the margin. In conclusion, hits prefer scenario A(stability) and misses scenario B(instability).

Since scenario A and B are corresponding to hit and low entropies, respectively, the margin can benefit from a large hit entropy E_{hit} (e.g., scenario A) and a low miss entropy E_{miss} (e.g., scenario B). We can set up a framework to maximize the hit entropy and minimize the miss entropy, which is equivalent to make the hypothesis-margin in Eq. 3.2 the most stable. We call the level of stability of hypothesis-margin as hypothesis-margin quality. And Bei and Hong [8] use the term max-min entropy principle to describe the process that we maximize the hit entropy and minimize the loss entropy to maximize the hypothesis-margin quality. Actually, the process of maximizing stable hypothesis-margin is an extension of the large hypothesis-margin principle.

3.3. Quadratic-Manhattan Measurement

We extend the margin in Eq. 3.2 by using a new quadratic-Manhattan measurement defined in Eq. 3.3:

$$q(\vec{x}_i, \vec{x}_j) = \left|\vec{x}_i - \vec{x}_j\right|^T \mathbf{W} \left|\vec{x}_i - \vec{x}_j\right|, \tag{3.3}$$

where $\mathbf{W}$ is a non-negative symmetric matrix (element-wise non-negative) and its Frobenius norm $\|\mathbf{W}\|_F = 1$. The quadratic-Manhattan measurement is a natural extension of the weight vector. The off-diagonal elements in $\mathbf{W}$ capture the feature interactions and the diagonal elements in $\mathbf{W}$ capture the features. Here, we explain the motivation why quadratic-Manhattan measurement can capture the influence of interactions. For example, $w_{a,b}(a \neq b)$, the element in the a-th row and b-th column of $\mathbf{W}$, reflects the influence of the interactions between two features a and b. In details, according to the extension of quadratic form, $w_{a,b}(a \neq b)$ is the coefficient for the combination of the a-th and b-th elements in vector $\left|\vec{x}_i - \vec{x}_j\right|$. The quadratic-Manhattan measurement is a natural extension of the weighted Manhanttan distance in Eq. 2.1. In Relief-based algorithms, the motivation of weighted Manhattan distance Eq. 2.1 can be equivalently captured by the feature weight update equation in Algorithm 1. Similarly, $w_{a,b}$ can be updated using the combination of the a-th and b-th features based on a randomly given instance, which is a straightforward way to understand the process of capturing interactions.

We define our new hypothesis-margin using the quadratic-Manhattan measurement as

$$\sum_{m \in \mathcal{M}_n} \beta_{n,m} q(\vec{x}_n, \vec{x}_m) - \sum_{h \in \mathcal{H}_n} \alpha_{n,h} q(\vec{x}_n, \vec{x}_h). \tag{3.4}$$

3.4. IMMIGRATE

We design the following cost function Eq. 3.5 to maximize our new hypothesis-margin (quantity) and the hypothesis-margin quality simultaneously:

$$\begin{aligned} C = &\sum_{n=1}^{N} \left(\sum_{h \in \mathcal{H}_n} \alpha_{n,h} \left|\vec{x}_n - \vec{x}_h\right|^T \mathbf{W} \left|\vec{x}_n - \vec{x}_h\right| - \sum_{m \in \mathcal{M}_n} \beta_{n,m} \left|\vec{x}_n - \vec{x}_m\right|^T \mathbf{W} \left|\vec{x}_n - \vec{x}_m\right| \right) \\ &+ \sigma \sum_{n=1}^{N} [E_{miss}(z_n) - E_{hit}(z_n)], \\ &\text{subject to}: \mathbf{W} \geq 0,\ \mathbf{W}^T = \mathbf{W},\ \|\mathbf{W}\|_F^2 = 1, \\ &\forall n,\ \sum_{h \in \mathcal{H}_n} \alpha_{n,h} = 1,\ \sum_{m \in \mathcal{M}_n} \beta_{n,m} = 1, \text{and } \alpha_{n,h} \geq 0, \beta_{n,m} \geq 0, \end{aligned} \tag{3.5}$$

where $E_{miss}(z_n) = -\sum_{m\in\mathcal{M}_n} \beta_{n,m} \log \beta_{n,m}$, $E_{hit}(z_n) = -\sum_{h\in\mathcal{H}_n} \alpha_{n,h} \log \alpha_{n,h}$, and σ is a hyperparameter that can be tuned via internal cross-validation.

We also design the following optimization procedure containing two iterative steps to find $\mathbf{W}$ that minimizes the cost function. The framework starts from a randomly initialized $\mathbf{W}$ and stops when the change of cost function is less than a preset limit or the iteration number reaches a preset threshold. In practice, we find that it typically takes ten times to stop and obtain good results. And based on our experiments, the different initialization of $\mathbf{W}$ will not influence the results of the iterative optimization. Our iterative optimization strategy is efficient to achieve reasonably good results. The computation time of IMMIGRATE is comparable to other interaction related methods such as SODA [12], hierNet [13].

The visualization of optimization procedure is in Figure. 1, where ΔC is the change of cost function Eq. 3.5 in one iteration and ϵ is a pre-set limit.

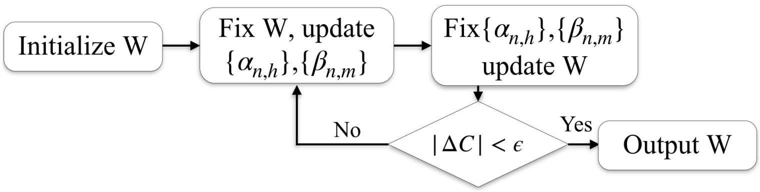


Figure 1. Flow chart of IMMIGRATE

Step 1: The optimization of cost function Eq. 3.5 starts from a randomly initialized $\mathbf{W}$ (satisfying $\mathbf{W} \geq 0$, $\mathbf{W}^T = \mathbf{W}$ and $\|\mathbf{W}\|_F^2 = 1$). Then the following two steps are iterated to minimize the cost function. Step 2: Fix $\mathbf{W}$, update $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$. Step 3: Fix $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$, update $\mathbf{W}$.

3.4.1. Fix $\mathbf{W}$, Update $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$

Fixing $\mathbf{W}$ and setting $\frac{\partial C}{\partial \alpha_{n,h}} = 0$ and $\frac{\partial C}{\partial \beta_{n,m}} = 0$, we can obtain the closed-form updates of $\alpha_{n,h}$ and $\beta_{n,m}$ as

$$\begin{aligned} \alpha_{n,h} &= \frac{exp(-q(\vec{x}_n, \vec{x}_h)/\sigma)}{\sum_{h\in\mathcal{H}_n} exp(-q(\vec{x}_n, \vec{x}_h)/\sigma)}, \\ \beta_{n,m} &= \frac{exp(-q(\vec{x}_n, \vec{x}_m)/\sigma)}{\sum_{k\in\mathcal{M}_n} exp(-q(\vec{x}_n, \vec{x}_k)/\sigma)}. \end{aligned} \tag{3.6}$$

The Hessian matrix of C w.r.t. probability pair $(\alpha_{n,h}, \beta_{n,m})$ is:

$$\frac{\partial^2 C}{\partial(\alpha_{n,h}, \beta_{n,m})} = \begin{pmatrix} \sigma/\alpha_{n,h} & \partial^2 C/\partial\beta_{n,m}\alpha_{n,h} \\ \partial^2 C/\partial\beta_{n,m}\alpha_{n,h} & -\sigma/\beta_{n,m} \end{pmatrix}. \tag{3.7}$$

Since $\alpha_{n,h}, \beta_{n,m} > 0$, the determinant of the Hessian matrix is negative, where a saddle point is found in $(\alpha_{n,h}, \beta_{n,m})$ space. Therefore, the cost function C achieves its local minimum and local maximum w.r.t. $\alpha_{n,h}$ and $\beta_{n,m}$, respectively.

3.4.2. Fix $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$, Update $\mathbf{W}$

Fixing $\alpha_{n,h}$ and $\beta_{n,m}$, the minimization w.r.t. $\mathbf{W}$ is convex. In Eq. 3.5, $\mathbf{W}$ satisfies $\mathbf{W} \geq 0$, $\mathbf{W}^T = \mathbf{W}$, $\|\mathbf{W}\|_F^2 = 1$. In our iterative optimization strategy, we impose $\mathbf{W}$ to be a distance metric for computation. Then, a closed-form solution to $\mathbf{W}$ can be derived (see Eq. 3.9).

Theorem 3.1. *With $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$ fixed, the cost function Eq. 3.5 has a closed-form solution to updating W.*

$$\mathbf{\Sigma} = \sum_{n=1}^{N} \Sigma_{n,H} - \Sigma_{n,M}, \ \mathbf{\Sigma}\ \psi_i = \mu_i\ \psi_i, \tag{3.8}$$

where $\Sigma_{n,H} = \sum_{h\in\mathcal{H}_n} \alpha_{n,h} |\vec{x}_n - \vec{x}_h| |\vec{x}_n - \vec{x}_h|^T$, $\Sigma_{n,M} = \sum_{m\in\mathcal{M}_n} \beta_{n,m} |\vec{x}_n - \vec{x}_m| |\vec{x}_n - \vec{x}_m|^T$, *and* $\|\psi_i\|_2^2 = 1$, $\mu_1 \le \mu_2 \le \cdots \le \mu_A$. *$\psi_i$'s and μ_i's are the eigenvectors and eigenvalues of* $\mathbf{\Sigma}$ *separately.*

$$\mathbf{W} = \Phi\Phi^T, \tag{3.9}$$

where $\Phi = (\sqrt{\eta_1}\psi_1, \sqrt{\eta_2}\psi_2, \cdots, \sqrt{\eta_A}\psi_A)$, $\sqrt{\eta_i} = \sqrt{(-\mu_i)^+ / \sqrt{\sum_{i=1}^A ((-\mu_i)^+)^2}}$.

Proof. Since $\mathbf{W}$ is a distance metric matrix, it is symmetric and positive-semidefinite. Eigenvalue decomposition of $\mathbf{W}$ is

$$\begin{aligned} \mathbf{W} &= P\Lambda P^T = P\Lambda^{1/2}\Lambda^{1/2}P^T, \\ &= [\sqrt{\lambda_1}\ p_1, \cdots, \sqrt{\lambda_A}\ p_A][\sqrt{\lambda_1}\ p_1, \cdots, \sqrt{\lambda_A}\ p_A]^T, \end{aligned} \tag{3.10}$$

where P is an orthogonal matrix. Thus, $\langle p_i, p_j \rangle = 0$.

Let $\Phi = [\phi_1, \cdots, \phi_A] = [\sqrt{\lambda_1}p_1, \cdots, \sqrt{\lambda_A}p_A]$, where $\langle \phi_i, \phi_j \rangle = 0$ and $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_A$.

The constraint $\|\mathbf{W}\|_F^2 = 1$ can be simplified since $\mathbf{W}$ can be decomposed to be some orthogonal vectors,

$$\|\mathbf{W}\|_F^2 = \sum_{i,j} w_{i,j}^2 = \sum_i (\phi_i^T \phi_i)^2 = 1. \tag{3.11}$$

Let us rearrange the Eq. 3.5:

$$\begin{aligned} &\sum_{h\in\mathcal{H}_n} \alpha_{n,h} |\vec{x}_n - \vec{x}_h|^T \mathbf{W} |\vec{x}_n - \vec{x}_h| \mathrm{tr}(\mathbf{W} \sum_{h\in\mathcal{H}_n} \alpha_{n,h} |\vec{x}_n - \vec{x}_h| |\vec{x}_n - \vec{x}_h|^T), \\ &\mathrm{tr}(\mathbf{W}\Sigma_{n,H}) = \mathrm{tr}(\Sigma_{n,H} \sum_{i=1}^A \phi_i \phi_i^T) = \sum_{i=1}^A \phi_i^T \Sigma_{n,H}\ \phi_i. \end{aligned} \tag{3.12}$$

Then, Eq. 3.5 can be simplified as follows:

$$\begin{aligned} &C = \sum_{i=1}^A \phi_i^T \Sigma\ \phi_i, \\ &\text{subject to} : \|\mathbf{W}\|_F^2 = \sum_i (\phi_i^T \phi_i)^2 = 1, \langle \phi_i, \phi_j \rangle = 0, \end{aligned} \tag{3.13}$$

where $\Sigma = \sum_{n=1}^N \Sigma_{n,H} - \Sigma_{n,M}$ and $\Sigma_{n,H} = \sum_{h\in\mathcal{H}_n} \alpha_{n,h} |\vec{x}_n - \vec{x}_h| |\vec{x}_n - \vec{x}_h|^T$, $\Sigma_{n,M} = \sum_{m\in\mathcal{M}_n} \beta_{n,m} |\vec{x}_n - \vec{x}_m| |\vec{x}_n - \vec{x}_m|^T$.

The orthogonal condition will be ignored when we derive the closed-form solution because this condition has already been satisfied at the last step.

The Lagrangian of Eq. 3.13 is easy to obtain:

$$L = \sum_{i=1}^A \phi_i^T \Sigma\ \phi_i + \lambda(\sum_{i=1}^A (\phi_i^T \phi_i)^2 - 1). \tag{3.14}$$

And derive L with respect to ϕ_i:

$$\partial L / \partial \phi_i = 2\Sigma\phi_i + 4\lambda\phi_i^T \phi_i \phi_i = 0. \tag{3.15}$$

Denote $\phi_i / \|\phi_i\|_2 := \psi_i$. From Eq. 3.15,

$$\Sigma\ \psi_i = \mu_i\ \psi_i, \tag{3.16}$$

where $\mu_i = -2\lambda\|\phi_i\|_2^2$. ψ_i and μ_i are the eigenvector and eigenvalue of Σ, separately.

Let $\phi_i = \sqrt{\eta_i}\psi_i$, $\eta_i \geq 0$. Thus, $C = \sum_{i=1}^{A} \sqrt{\eta_i}{\psi_i}^T \Sigma \sqrt{\eta_i}\psi_i = \sum_{i=1}^{A} \eta_i \mu_i {\psi_i}^T \psi_i = \sum_{i=1}^{A} \eta_i \mu_i$, and $\|\mathbf{W}\|_F^2 = \sum_i (\sqrt{\eta_i}\psi_i^T \sqrt{\eta_i}\psi_i)^2 = \sum_i (\eta_i)^2 = 1$.

Then, Eq. 3.13 can be simplified to be

$$C = \sum_{i=1}^{A} \eta_i \mu_i, \text{ subject to}: \sum_{i=1}^{A} (\eta_i)^2 = 1, \eta_i \geq 0. \tag{3.17}$$

It is excited to notice Eq. 3.17 is exactly the same as the original Relief Algorithm (Algorithm 1):

$$\vec{\eta} = (-\vec{\mu})^+ / \|(-\vec{\mu})^+\|_2, \tag{3.18}$$

where $(\vec{a})^+ = [max(a_1, 0), max(a_2, 0), \cdots, max(a_I, 0)]$, and $\phi_i = \sqrt{\eta_i}\psi_i$.

Using $\Phi = [\phi_1, \cdots, \phi_A] = [\sqrt{\lambda_1}p_1, \cdots, \sqrt{\lambda_A}p_A]$,

$$\mathbf{W} = \Phi\Phi^T. \tag{3.19}$$

The orthogonal condition is achieved, because $\|\mathbf{W}\|_F^2 = \sum_i (\phi_i^T \phi_i)^2 = 1$.

In addition, since $\mathbf{W} = \Phi\Phi^T$, updated $\mathbf{W}$ is also a distance metric. □

Algorithm 2 IMMIGRATE Algorithm

Input: a training dataset $\{z_n = (\vec{x}_n, y_n)\}_{n=1,\cdots,N}$.

Initialization: Let $t = 0$, randomly initialize $\mathbf{W}^{(0)}$ satisfying $\mathbf{W}^{(0)} \geq 0$, $\mathbf{W}^T = \mathbf{W}$, $\|\mathbf{W}^{(0)}\|_F^2 = 1$.

repeat

Calculate $\{\alpha_{n,h}^{(t+1)}\}$, $\{\beta_{n,m}^{(t+1)}\}$ with Eq. 3.6.

Calculate $\mathbf{W}^{(t+1)}$ with Theorem 3.1, Eq. 3.9.

t = t+1.

until the change of C in Eq. 3.5 is small enough or the iteration indicator t reaches a preset limit.

Output: $\mathbf{W}^{(t)}$.

3.4.3. Weight Pruning

The previous Relief-based algorithms offer options to remove weights lower than a preset threshold. IMMIGRATE offers a similar option to prune small weights: set small elements in $\mathbf{W}$ to 0. By default, we use a threshold to prune small weights to 0, where $\mathbf{W}$ should be normalized w.r.t. Frobenius norm after the pruning.

3.4.4. Predict New Samples

We design a new prediction method using the learned weight matrix $\mathbf{W}$:

$$\begin{aligned} \hat{y}' &= \arg\min_c \sum_{y_n=c} \alpha_n^c(\vec{x}')q(\vec{x}', \vec{x}_n), \\ \alpha_n^c(\vec{x}') &= \frac{exp\big(-q(\vec{x}', \vec{x}_n)/\sigma\big)}{\sum_{y_k=c} exp\big(-q(\vec{x}', \vec{x}_k)/\sigma\big)}, \end{aligned} \tag{3.20}$$

where $z' = (\vec{x}', y')$ is a new instance, c denotes the class and $\hat{y}'$ is the predicted label. This prediction method assigns a new instance to a class that maximizes its hypothesis-margin using the learned weight matrix $\mathbf{W}$, which makes it more reasonable than the k-NN method used in the traditional Relief-based algorithms.

3.5. IMMIGRATE in Ensemble Learning

Boosting [10,14,15] has been widely used to create ensemble learners that produce the state-of-the-art results in many tasks. Boosting combines a set of relatively weak base learners to create a much stronger learner. To use IMMIGRATE as the base classifier in the AdaBoost algorithm [14], we modify the cost function Eq. 3.5 to include sample weights and use the modified version in the boosting iterations. We name the algorithm BIM, standing for **B**oosted **IM**MIGRATE (Refer to Eq. 3.21 and Algorithm 3 for the details of BIM. BIM schedules the adjustment of the hyperparameter σ in its boosting iterations. It starts with letting σ be a predefined σ_{max} and gradually decreases σ_{max} by $(\sigma_{min}/\sigma_{max})^{1/T}$ in each interaction until reaching σ_{min}, where T is a predefined maximum number of boosting iterations.

$$
\begin{aligned}
C = & \sum_{n=1}^{N} D(\vec{x}_n)\Big(\sum_{h\in\mathcal{H}_n} \alpha_{n,h} |\vec{x}_n - \vec{x}_h|^T \mathbf{W} |\vec{x}_n - \vec{x}_h| - \sum_{m\in\mathcal{M}_n} \beta_{n,m} |\vec{x}_n - \vec{x}_m|^T \mathbf{W} |\vec{x}_n - \vec{x}_m| \Big) \\
& + \sigma \sum_{n=1}^{N} D(\vec{x}_n)[E_{miss}(z_n) - E_{hit}(z_n)], \\
& \text{subject to}: \mathbf{W} \geq 0,\ \mathbf{W}^T = \mathbf{W},\ \|\mathbf{W}\|_F^2 = 1, \\
& \forall n,\ \sum_{h\in\mathcal{H}_n} \alpha_{n,h} = 1,\ \sum_{m\in\mathcal{M}_n} \beta_{n,m} = 1, \text{and } \alpha_{n,h} \geq 0, \beta_{n,m} \geq 0,
\end{aligned} \tag{3.21}
$$

where $E_{miss}(z_n) = -\sum_{m\in\mathcal{M}_n} \beta_{n,m} \log \beta_{n,m}$, $E_{hit}(z_n) = -\sum_{h\in\mathcal{H}_n} \alpha_{n,h} \log \alpha_{n,h}$, $\sum_{n=1}^{N} D(\vec{x}_n) = 1$, and $D(\vec{x}_n) \geq 0,\ \forall\ n$

Algorithm 3 BIM Algorithm

T: the number of classifiers for BIM.
Input : a training dataset $\{z_n = (\vec{x}_n, y_n)\}_{n=1,\cdots,N}$.
Initialization : for each $\vec{x}_n$, set $D_1(\vec{x}_n) = 1/N$.

for $t := 1$ **to** T **do**
  Limit max number of iteration of IMMIGRATE less than preset.
  Train weak IMMIGRATE classifier $h_t(x)$ using a chosen σ_t and weights $D_t(x)$ by Eq. 3.21.
  Compute the error rate ϵ_t as $\epsilon_t = \sum_{i=1}^{N} D_t(x_i) I[y_i \neq h_t(x_i)]$.
  if $\epsilon_t \geq 1/2$ or $\epsilon_t = 0$ **then**
    Discard h_t, $T = T - 1$ and continue .
  Set $\alpha_t = 0.5 \times \log[(1-\epsilon_t)/\epsilon_t]$.
  Update $D(x_i)$: For each x_i,
    $D_{t+1}(x_i) = D_t(x_i) \exp(\alpha_t I[y_i \neq h_t(x_i)])$.
  Normalize $D_{t+1}(x_i)$, so that $\sum_{i=1}^{N} D_{t+1}(x_i) = 1$.
Output: $h_{final}(x) = \arg\max_{y\in\{0,1\}} \sum_{t:h_t(x)=y} \alpha_t$.

3.6. IMMIGRATE for High-Dimensional Data Space

When applied to high-dimensional data, IMMIGRATE can incur a high computational cost because it considers the interactions between every feature pair. To reduce the computational cost, we first use IM4E [8] to learn a feature weight vector, which is used to initialize the diagonal elements of $\mathbf{W}$ in the proposed quadratic-Manhattan measurement. We also use the learned feature weight vector to choose the features with weights above a preset limit. In the rest computation, we only model the interactions between those chosen features. The remaining features are empirically decided and can be adjusted accordingly to the need of a specific application. We term this procedure IM4E-IMMIGRATE, which is a sub-optimal solution but effective and efficient. It can also be boosted (Boosted IM4E-IMMIGRATE) to be stronger.

4. Experiments

In our experiments, all continuous features are normalized with mean zero and unit variance. And cross-validation is used here to compare the performances of various approaches. We have implemented IMMIGRATE in R and MATLAB. The R package is available at https://CRAN.R-project.org/package=Immigrate, and the MATLAB version is available at https://github.com/RuzhangZhao/Immigrate-MATLAB- is also available. Both IMMIGRATE and BIM can be accelerated by parallel computing as their computations are matrix-based.

4.1. Synthetic Dataset

We first test the robustness of the IMMIGRATE algorithm using a synthesized dataset where we have two interacting features following Gaussian distributions in a binary classification setting. The simulated dataset contains 100 samples from one class governed by a Gaussian distribution with mean $(4, 2)^T$ and variance $\begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$ and another 100 samples from the other class governed by a Gaussian distribution with mean $(6, 0)^T$ and the same variance. In addition, we add noise following a Gaussian distribution with mean $(8, -2)^T$ and variance $\begin{pmatrix} 8 & 4 \\ 4 & 8 \end{pmatrix}$ to the fist class, and add noises following a Gaussian distribution with mean $(2, 4)^T$ and the same variance to the second class. Fig. 2 shows a scatter plot of the synthesized dataset containing 10% samples from the noise distributions. The slope of the orange dotted line in Fig. 2 is 1, which separates data with different labels.

The noises are included to disturb the detection of the interaction term. The noise level starts from 5%, and gradually increases by 5% to 50%. As the baseline, we apply logistic regression and see that the *t*-test *p*-values of the interaction coefficients increase from 3×10^{-11}, 7×10^{-5}, to 0.7 when the noise levels increase from 0, 10%, to 50%. Local Feature Extraction (LFE, Sun and Wu [7]) is a Relief-based algorithm which considers interaction terms indirectly, though the interaction information is only used for feature extraction. We run IMMIGRATE and LFE on the synthesized datasets and compare the weights of the interaction term between features 1 and 2 in Fig. 3, which shows IMMIGRATE is more robust than LFE.

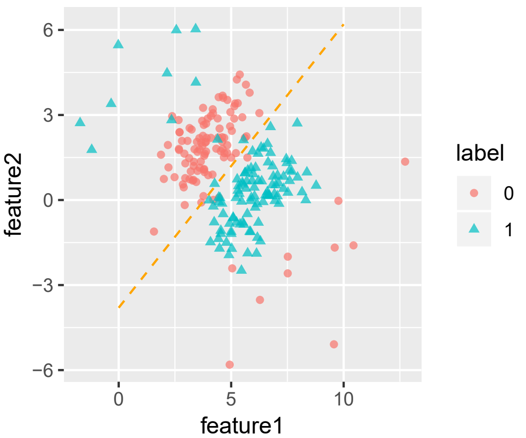


Figure 2. The synthetic dataset with 10% noise.

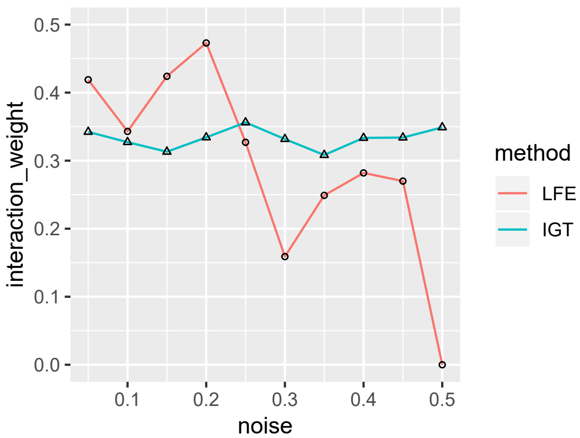


Figure 3. IMMIGRATE (IGT) is more robust than LFE.

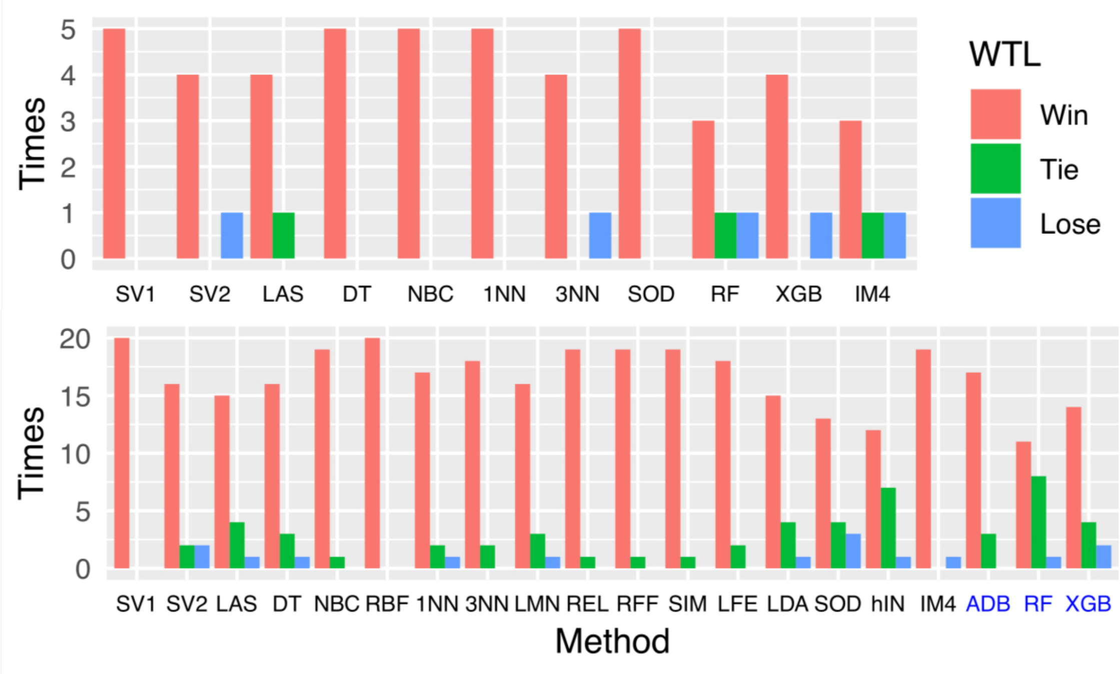


Figure 4. Results of paired t-test on gene expression datasets (top subplot) and UCI datasets (bottom subplot). The top plot shows how well (i.e., "Win" (red bars), "Tie" (green bars), and "Lose" (blue bars)) our Boosted IM4E-IMMIGRATE performs compared with other approaches. In the bottom plot, the results of methods labeled in black are the comparisons with our IMMIGRATE, and the results of methods (ABD, RF, and XGB) labeled in blue are the comparisons with our BIM.

4.2. Real Datasets

We compare IMMIGRATE with several existing popular methods using real datasets from the UCI database. The following existing algorithms are used in the comparison: Support Vector Machine [16] with Sigmoid Kernel (SV1), Support Vector Machine with Radial basis function Kernel (SV2), LASSO (LAS) [17], Decision Tree (DT) [15], Naive Bayes Classifier (NBC) [18], Radial basis function Network (RBF) [19], 1-Nearest Neighbor (1NN) [20], 3-Nearest Neighbor (3NN), Large Margin Nearest Neighbor (LMN) [21], Relief (REL) [2], ReliefF (RFF) [4,22], Simba (SIM) [3], and Linear Discriminant Analysis (LDA) [23]. In addition, several methods designed for detecting interaction terms are included: LFE [7], Stepwise conditional likelihood variable selection for Discriminant Analysis (SOD) [12], and hierNet (HIN) [13]. We also include three most widely used and competitive ensemble learners: Adaptive Boosting (ADB) [14,15], Random Forest (RF) [24], and XgBoost (XGB) [25]. We use the following abbreviations when presenting the results: IM4 for IM4E, IGT for IMMIGRATE, and B4G for the boosted IM4E-IMMIGRATE.

Whenever possible, we use the settings of the above methods reported in their original papers: LMNN uses 3-NN classifier; Relief and Simba use Euclidean distance and 1-NN classifier; ReliefF uses Manhattan distance and k-NN classifier (k=1,3,5 is decided by internal cross-validation); in SODA, gam (=0,0.5,1) is determined by internal cross-validation and logistic regression is used for prediction. The IM4E algorithm owns two hyperparameters λ and σ. We fix $\lambda = 1$ as it has no actual contribution and tune σ as suggested by Bei and Hong [8]. Hence, the IMMIGRATE algorithm only has one hyperparameter σ. When tuning σ, we gradually decrease σ from $\sigma_0 = 4$ by half each time until it is not larger than 0.2. The preset limit for weight pruning is $1/A$, where A is the number of features. Also, the preset iteration number is chosen to be 10. For each dataset, σ and whether weight pruning is applied are determined by the best internal cross-validation results. For BIM, we use $\sigma_{max} = 4$, $\sigma_{min} = 0.2$, and the maximal number of boosting iterations T is 100. The preset threshold in IM4E-IMMIGRATE is $2/A$.

We repeat ten-fold cross-validation ten times for each algorithm on each dataset, i.e., 100 trials are carried out. When comparing two algorithms (i.e., A vs B), we calculate the paired Student's t-test using the results of 100 trials. First, the null hypothesis is there is no difference between the performances of A and those of B. When the p-value is larger than the significant level cutoff 0.05, we say A "Tie" B, which means there is no significant difference between their performances. When the p-value is smaller than the significant level cutoff 0.05, the second null hypothesis is the performances of B are no worse than those of A. When the new p-value is smaller than the significant level cutoff 0.05, we say A "Win" B which means A on average performs better than B on this dataset (i.e., A performs significantly better than B), and vice versa.

4.2.1. Gene Expression Datasets

Gene expression datasets typically have thousands of features. We use the following five gene expression datasets for feature selections: GLI [26], Colon [27](COL), Myeloma (ELO) [28], Breast (BRE) [29], Prostate (PRO) [30]. All datasets have more than 10,000 features. Refer to Table A1 in Appendix A for details of all datasets.

We perform ten-fold cross-validation ten times, i.e., 100 trials in total. The results are summarized in Table 1. The last row "(W,T,L)" indicates the number of times that the Boosted IM4E-IMMIGRATE (B4G) W,T,L (win,tie,loss) compared with each algorithm by the paired Student's t-test with the significance level of $\alpha = 0.05$. The comparison results are also summarized in Figure 4 (top plot) for easy comparison. Although our B4G is not always the best, it outperforms other methods in most cases. In particular, when IM4E-IMMIGRATE (EGT) is compared with other methods, it also outperforms in most cases.

Table 1. Summarizes the accuracies on five high-dimensional gene expression datasets.

Data	SV1	SV2	LAS	DT	NBC	1NN	3NN	SOD	RF	XGB	IM4	**EGT**	**B4G**
GLI	85.1	86.0	85.2	83.8	83.0	88.7	87.7	88.7	87.6	86.3	87.5	89.1	89.9
COL	73.7	82.0	80.6	69.2	71.1	72.1	77.9	78.1	82.6	79.5	84.3	78.6	82.5
ELO	72.9	90.2	74.6	77.3	76.3	85.6	91.3	86.9	79.2	77.9	88.9	88.6	88.4
BRE	76.0	88.7	91.4	76.4	69.4	83.0	73.6	82.6	86.3	87.3	88.1	90.2	91.5
PRO	71.3	69.9	87.9	86.4	68.0	83.2	82.7	83.2	91.8	90.5	88.0	89.5	89.7
W,T,L[1]	**5**,0,0	**4**,0,1	**4**,1,0	**5**,0,0	**5**,0,0	**5**,0,0	**4**,0,1	**5**,0,0	**3**,1,1	**4**,0,1	**3**,1,1	-,-,-	-,-,-

[1] The last row shows the number of times Boosted IM4E-IMMIGRATE(**B4G**) W,T,L (win,tie,loss) compared with each algorithm by paired t-test

** Ten-fold cross-validation is performed for ten times, namely 100 trials are carried out for each dataset. The average accuracies are reported on the corresponding datasets in Table 1,2,3. Here, with 100 trials and two algorithms A and B, paired Student's t-test is carried out between the results of these two algorithms. Under the significance level of $\alpha = 0.05$, algorithm A is significantly better than (i.e. win) another algorithm B on a dataset if the p-value of the paired Student's t-test with corresponding null hypothesis is less than $\alpha = 0.05$. (The rule also applies to experiments on UCI datasets) .

4.2.2. UCI Datasets

We also carry out an extensive comparison using many UCI datasets [31]: BCW, CRY, CUS, ECO, GLA, HMS, IMM, ION, LYM, MON, PAR, PID, SMR, STA, URB, USE and WIN. Refer to Appendix A Table A1 for the full names and links for those datasets. If a dataset has more than two classes, we use two classes with the largest sample size. In addition, we use three large-scale datasets: CRO*, ELE*, WAV*.

We perform ten-fold cross-validation ten times. Tables 2 for IMMIGRATE and Table 3 for BIM show the average accuracies on the corresponding datasets. In Table 2, the last row "(W,T,L)" indicates the number of times IMMIGRATE (IGT) and BIM W,T,L (win,tie,loss) when compared with each algorithm separately by using the paired Student's t-test with the significance level of $\alpha = 0.05$. The comparison results are also summarized in Figure 4 (bottom subplot), where the first 17 items (black) indicate the results for IMMIGRATE while the last three items (blue) indicate the results for BIM.

Although IMMIGRATE or BIM is not always the best, they outperform other methods significantly in one-to-one comparisons in terms of cross-validation results. Figure 4 (bottom subplot, black part) and Table 2 show that IMMIGRATE achieves the state-of-the-art performance as the base classifier while Figure 4 (bottom subplot, blue part) and Table 3 show BIM achieves the state-of-the-art performance as the boosted version. To visualize the feature selection results of our approaches, we plot the feature weight heat maps of four datasets (GLA, LYM, SMR and STA) in Appendix B Figure A5.

Table 2. Summarizes the accuracies on UCI datasets.

Data	SV1	SV2	LAS	DT	NBC	RBF	1NN	3NN	LMN	REL	RFF	SIM	LFE	LDA	SOD	hIN	IM4	**IGT**
BCW	61.4	66.6	71.4	70.5	62.4	56.9	68.2	72.2	69.5	66.4	67.1	67.7	67.1	73.9	65.2	71.8	66.4	74.5
CRY	72.9	90.6	87.4	85.3	84.4	89.7	89.1	85.4	87.8	73.8	77.2	79.7	86.0	88.6	86.0	87.9	86.2	89.8
CUS	86.5	88.9	89.6	89.6	89.5	86.8	86.5	88.7	88.8	82.1	84.7	84.3	86.4	90.3	90.8	90.3	87.5	90.1
ECO	92.9	96.9	98.6	98.6	97.8	94.6	96.0	97.8	97.8	89.0	90.7	91.2	93.1	99.0	97.9	98.7	97.5	98.2
GLA	64.2	76.7	72.3	79.4	69.5	73.0	81.1	78.1	79.4	64.1	63.5	67.1	81.2	72.0	75.3	75.0	78.0	87.5
HMS	63.8	64.5	67.7	72.5	67.2	66.8	66.0	69.3	71.2	65.3	66.0	65.7	64.9	69.0	67.4	69.4	66.6	69.2
IMM	74.3	70.6	74.4	84.1	77.9	67.3	69.4	77.9	76.7	69.9	71.8	69.0	75.0	75.2	72.3	70.2	80.7	83.8
ION	80.5	93.5	83.6	87.4	89.4	79.9	86.7	84.1	84.5	85.8	86.2	84.2	91.0	83.3	90.3	92.6	88.3	92.9
LYM	83.6	81.5	85.2	75.2	83.6	71.1	77.2	82.8	86.6	64.9	71.0	70.4	79.6	85.2	79.3	84.8	83.3	87.2
MON	74.4	91.7	75.0	86.4	74.0	68.2	75.1	84.4	84.9	61.4	61.8	65.0	64.8	74.4	91.9	97.2	75.6	99.5
PAR	72.7	72.5	77.1	84.8	74.1	71.5	94.6	91.4	91.8	87.3	90.3	84.6	94.0	85.6	88.2	89.5	83.2	93.8
PID	65.6	73.1	74.7	74.3	71.2	70.3	70.3	73.5	74.0	64.8	68.0	67.0	67.8	74.5	75.7	74.1	72.1	74.7
SMR	73.5	83.9	73.6	72.3	70.3	67.1	86.9	84.7	86.1	69.5	78.3	81.0	84.3	73.1	70.5	83.0	76.4	86.5
STA	69.8	71.6	70.8	68.9	71.0	69.5	67.8	70.8	71.3	59.7	64.0	63.0	66.7	71.3	71.8	69.2	70.8	75.9
URB	85.2	87.9	88.1	82.6	85.8	75.3	87.2	87.5	87.9	81.9	83.2	73.0	87.9	73.0	87.9	88.3	87.4	89.9
USE	95.7	95.2	97.2	93.2	90.6	84.9	90.5	91.5	92.0	54.5	63.7	69.5	85.8	96.9	96.2	96.5	94.1	96.4
WIN	98.3	99.3	98.6	93.1	97.3	97.2	96.4	96.6	96.5	87.2	95.0	95.0	93.8	99.7	92.9	98.9	98.2	99.0
CRO*	75.4	97.5	89.9	91.0	88.8	75.4	98.4	98.5	98.6	98.5	98.7	95.1	98.6	89.1	95.2	95.5	81.9	98.2
ELE*	72.3	95.7	79.9	80.0	82.5	70.8	81.1	83.9	89.7	64.6	75.4	76.2	79.8	79.9	93.7	93.6	83.2	93.7
WAV*	90.0	91.9	92.2	86.2	91.4	84.0	86.5	88.3	88.8	77.6	80.0	83.6	84.7	91.8	92.0	92.1	91.1	92.4
W,T,L[1]	**20**,0,0	**16**,2,2	**15**,4,1	**16**,3,1	**19**,1,0	**20**,0,0	**17**,2,1	**18**,2,0	**16**,3,1	**19**,1,0	**19**,1,0	**19**,1,0	**18**,2,0	**15**,4,1	**13**,4,3	**12**,7,1	**19**,0,1	-,-,-

[1] The last row (W,T,L) shows the number of times that IMMIGRATE (**IGT**) wins/ties/losses an existing algorithm according to the paired *t*-test on the cross-validation results.

Table 3. Summarizes the accuracies on UCI datasets.

Data	ADB	RF	XGB	**BIM**
BCW	78.2	78.6	78.6	78.3
CRY	90.4	92.9	89.9	91.5
CUS	90.8	91.1	91.4	91.0
ECO	98.0	98.9	98.2	98.6
GLA	85.0	87.0	87.9	86.8
HMS	65.8	72.1	70.0	72.0
IMM	77.2	84.2	81.7	86.1
ION	92.1	93.5	92.5	93.1
LYM	84.8	87.0	87.4	88.1
MON	98.4	95.8	99.1	99.7
PAR	90.5	91.0	91.9	93.2
PID	73.5	76.0	75.1	76.2
SMR	81.4	82.8	83.3	86.6
STA	69.0	71.3	69.5	74.1
URB	87.9	88.6	88.8	91.4
USE	96.0	95.3	94.9	96.1
WIN	97.5	99.1	98.2	99.1
CRO*	97.3	97.4	98.5	98.6
ELE*	91.1	92.3	95.2	94.1
WAV*	89.5	91.2	90.8	93.3
W,T,L[1]	**17**,3,0	**11**,8,1	**14**,4,2	-,-,-

[1] The last row (W,T,L) shows the number of times that the Boosted IMMIGRATE (**BIM**) wins/ties/losses an existing algorithm according to the paired *t*-test on the cross-validation results.

5. Related Works

In many recent researches, Relief-based algorithms and feature selection with interaction terms have been well explored. Some methods are reviewed here to show the connection and differences with our approach. The hypothesis-margin definition in Eq. 3.2 adopted in this work is also used in previous studies, such as Bei and Hong [8]. However, Bei and Hong [8] do not consider the interactions between features. Our work provides a measurable way to show the influence of each feature interaction.

Sun and Wu [7] propose local feature extraction (LFE) method which learns linear combination of features for feature extraction. LFE explores the information of feature interaction terms indirectly, which is partly our aim. However, LFE does not consider global information or margin stability, which results in significant differences in the cost function and the optimization procedures.

Our quadratic-Manhattan measurement Eq. 3.3 is related to the Mahalanobis metric used in previous works on metric learning, such as Large Margin Nearest Neighbor (LMNN) [21]. Weinberger and Saul [21] use semi-definite programming for learning distance metric in LMNN. LMNN and our approach are both based on K-Nearest Neighbor. A major difference is that our quadratic-Manhattan measurement has matrix $\mathbf{W}$ be a non-negative symmetric matrix (element-wise non-negative) and its Frobenius norm $\|\mathbf{W}\|_F = 1$. While in metric learning, metric learning imposes the matrix to be symmetric semi-positive definite. Actually, non-negative requirement provides IMMIGRATE high intepretability, where items in matrix serve as interaction importance. Quadratic-Manhattan measurement serves well in the classification task and offers a great explanation about how features, in particular, feature interaction terms, contribute to the classification results.

6. Conclusion & Discussion

In this paper, a novel feature selection algorithm IMMIGRATE is proposed for detecting and weighting interaction terms. We also develop its extended versions, such as, Boosted IMMIGRATE (BIM) and IM4E-IMMIGRATE. A new quadratic-Manhattan measurement is proposed to extend the hypothesis-margin. IMMIGRATE and its variants follow the principle of maximizing stable hypothesis-margin. An iterative optimization framework is designed for implementing the IMMIGRATE algorithm and the closed-form update of parameters is derived in Theorem 3.1. Extensive experiments show that IMMIGRATE outperforms existing methods and improves the state-of-the-art. BIM outperforms other boosting-based approaches. Its robustness is clearly demonstrated on synthetic dataset where we know the ground truth. In conclusion, compared with other Relief-based algorithms, IMMIGRATE mainly has the following advantages: (1) both local and global information are considered; (2) interaction terms are used; (3) robust and less prone to noise; (4) easily boosted. The computation time of IMMIGRATE variants is comparable to other methods able to detect interaction terms.

There are several directions for improving IMMIGRATE. First, in section 3.4.3, small weights are removed to obtain sparse solutions. We can explore using l_0 or l_1 to cut insignificant weights. Second, to further improve the computational efficiency of IMMIGRATE for large-scale datasets, we can improve training by using well selected prototypes [32]. Third, IMMIGRATE only considers pair-wise interactions between features. Interactions among multiple features can play important roles in real applications. Our work provides a basis for developing new algorithms to detect multi-feature interactions. For example, people can use tensor form to consider weights for multi-feature interactions. Fourth, although our iterative optimization procedure is efficient, it achieves sub-optimal solutions. In particular, procedure 3.4.1 and 3.4.2 are both sub-optimal. It remains an open challenge to develop better optimization algorithms. Finally, the selection of an appropriate σ currently relies on internal cross-validation. A better strategy may be developed by rigorously investigating the theoretical contributions of σ.

Author Contributions: methodology, R.Z. and P.H.; software, R.Z.; validation, R.Z., P.H. and J.S.L.; investigation, R.Z., P.H. and J.S.L.; resources, R.Z., P.H. and J.S.L.; data curation, R.Z. and P.H.; writing–original draft preparation, R.Z.; writing–review and editing, R.Z., P.H. and J.S.L.; supervision, P.H. and J.S.L.; funding acquisition, P.H. and J.S.L.

Funding: This research was supported partially by the the National Science Foundation grants DMS-1613035, DMS-1712714, and OAC-1920147.

Acknowledgments: The authors thank Dr. Xin Xing for the valuable suggestions to improve the work. And the authors thank Dr. Yang Li for the helpful suggestions about R codes.

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

Abbreviations

The following abbreviations are used in this manuscript:

NH	Nearest Hit
NM	Nearest Miss
IM4E	Iterative Margin-Maximization under Max-Min Entropy algorithm
IMMIGRATE	*__I__terative __M__ax-__MI__n entropy mar__G__in-maximization with inte__RA__ction __TE__rms* algorithm

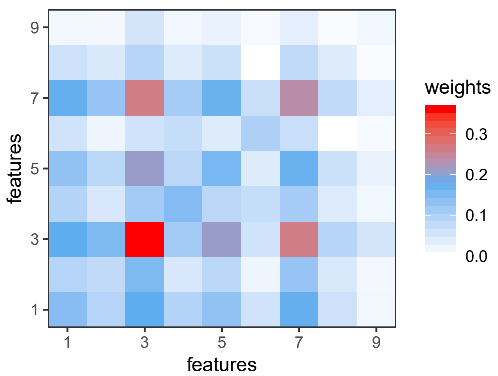
339 Appendix A Summarizes the information of Real Datasets

Table A1. Summarizes the information of UCI datasets and gene expression datasets.

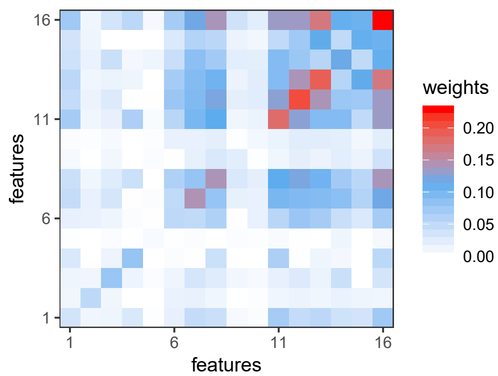
Data	No.F[1]	No.I[2]	Full Name
BCW	9	116	Breast Cancer Wisconsin (Prognostic)
CRY	6	90	Cryotherapy
CUS	7	440	Wholesale customers
ECO	5	220	Ecoli
GLA	9	146	Glass Identification
HMS	3	306	Haberman's Survival
IMM	7	90	Immunotherapy
ION	32	351	Ionosphere
LYM	16	142	Lymphograph
MON	6	432	MONK's Problems
PAR	22	194	Parkinsons
PID	8	768	Pima-Indians-Diabetes
SMR	60	208	Connectionist Bench (Sonar, Mines vs. Rocks)
STA	12	256	Statlog (Heart)
URB	147	238	Urban Land Cover
USE	5	251	User Knowledge Modeling
WIN	13	130	Wine
CRO*	28	9003	Crowdsourced Mapping
ELE*	12	10000	Electrical Grid Stability Simulated
WAV*	21	3304	Waveform Database Generator
GLI	22283	85	Gliomas Strongly Predicts Survival[26]
COL	2000	62	Tumor and Normal Colon Tissues[27]
ELO	12625	173	Myeloma[28]
BRE	24481	78	Breast Cancer[29]
PRO	12600	136	Clinical Prostate Cancer Behavior[30]

[1] No.F: Number of Features.
[2] No.I: Number of Instances.

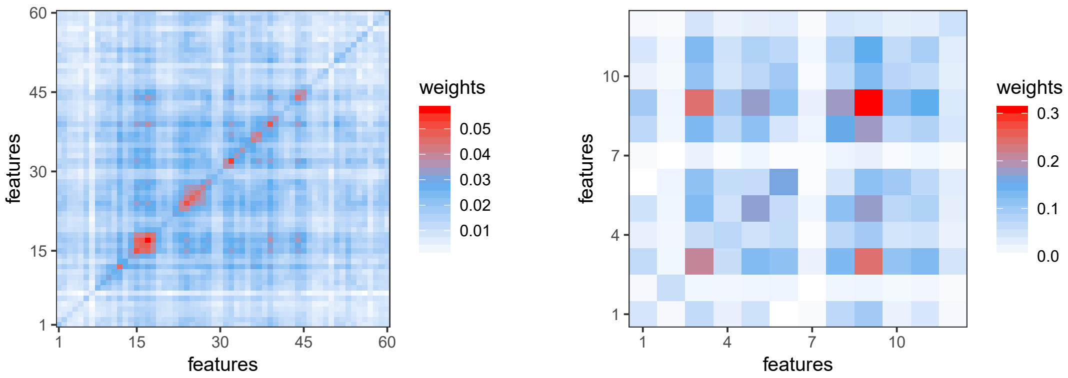
340 Appendix B Heat Maps



GLA



LYM



SMR STA

Figure: Heat Maps of Feature Weights Learned by IMMIGRATE.
The color bar shows the value of corresponding colors.

References

1. Fukunaga, K. *Introduction to statistical pattern recognition*; Elsevier, 2013.
2. Kira, K.; Rendell, L.A. A practical approach to feature selection. In *Machine Learning Proceedings 1992*; Elsevier, 1992; pp. 249–256.
3. Gilad-Bachrach, R.; Navot, A.; Tishby, N. Margin based feature selection-theory and algorithms. Proceedings of the twenty-first international conference on Machine learning. ACM, 2004, p. 43.
4. Kononenko, I. Estimating attributes: analysis and extensions of RELIEF. European conference on machine learning. Springer, 1994, pp. 171–182.
5. Yang, M.; Wang, F.; Yang, P. A Novel Feature Selection Algorithm Based on Hypothesis-Margin. *JCP* **2008**, *3*, 27–34.
6. Sun, Y.; Li, J. Iterative RELIEF for feature weighting. Proceedings of the 23rd international conference on Machine learning. ACM, 2006, pp. 913–920.
7. Sun, Y.; Wu, D. A relief based feature extraction algorithm. Proceedings of the 2008 SIAM International Conference on Data Mining. SIAM, 2008, pp. 188–195.
8. Bei, Y.; Hong, P. Maximizing margin quality and quantity. Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on. IEEE, 2015, pp. 1–6.
9. Urbanowicz, R.J.; Meeker, M.; La Cava, W.; Olson, R.S.; Moore, J.H. Relief-based feature selection: introduction and review. *Journal of biomedical informatics* **2018**.
10. Schapire, R.E. The strength of weak learnability. *Machine learning* **1990**, *5*, 197–227.
11. Kuhn, H.W.; Tucker, A.W. Nonlinear programming. In *Traces and emergence of nonlinear programming*; Springer, 2014; pp. 247–258.
12. Li, Y.; Liu, J.S. Robust variable and interaction selection for logistic regression and general index models. *Journal of the American Statistical Association* **2018**, pp. 1–16.
13. Bien, J.; Taylor, J.; Tibshirani, R. A lasso for hierarchical interactions. *Annals of statistics* **2013**, *41*, 1111.
14. Freund, Y.; Schapire, R.E.; others. Experiments with a new boosting algorithm. Icml. Citeseer, 1996, Vol. 96, pp. 148–156.
15. Freund, Y.; Mason, L. The alternating decision tree learning algorithm. icml, 1999, Vol. 99, pp. 124–133.
16. Soentpiet, R.; others. *Advances in kernel methods: support vector learning*; MIT press, 1999.
17. Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* **1996**, pp. 267–288.
18. John, G.H.; Langley, P. Estimating continuous distributions in Bayesian classifiers. Proceedings of the Eleventh conference on Uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc., 1995, pp. 338–345.
19. Haykin, S. *Neural networks: a comprehensive foundation*; Prentice Hall PTR, 1994.
20. Aha, D.W.; Kibler, D.; Albert, M.K. Instance-based learning algorithms. *Machine learning* **1991**, *6*, 37–66.

21. Weinberger, K.Q.; Saul, L.K. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research* **2009**, *10*, 207–244.
22. Robnik-Šikonja, M.; Kononenko, I. Theoretical and empirical analysis of ReliefF and RReliefF. *Machine learning* **2003**, *53*, 23–69.
23. Fisher, R.A. The use of multiple measurements in taxonomic problems. *Annals of eugenics* **1936**, *7*, 179–188.
24. Breiman, L. Random forests. *Machine learning* **2001**, *45*, 5–32.
25. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016, pp. 785–794.
26. Freije, W.A.; Castro-Vargas, F.E.; Fang, Z.; Horvath, S.; Cloughesy, T.; Liau, L.M.; Mischel, P.S.; Nelson, S.F. Gene expression profiling of gliomas strongly predicts survival. *Cancer research* **2004**, *64*, 6503–6510.
27. Alon, U.; Barkai, N.; Notterman, D.A.; Gish, K.; Ybarra, S.; Mack, D.; Levine, A.J. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences* **1999**, *96*, 6745–6750.
28. Tian, E.; Zhan, F.; Walker, R.; Rasmussen, E.; Ma, Y.; Barlogie, B.; Shaughnessy Jr, J.D. The role of the Wnt-signaling antagonist DKK1 in the development of osteolytic lesions in multiple myeloma. *New England Journal of Medicine* **2003**, *349*, 2483–2494.
29. Van't Veer, L.J.; Dai, H.; Van De Vijver, M.J.; He, Y.D.; Hart, A.A.; Mao, M.; Peterse, H.L.; Van Der Kooy, K.; Marton, M.J.; Witteveen, A.T.; others. Gene expression profiling predicts clinical outcome of breast cancer. *nature* **2002**, *415*, 530.
30. Singh, D.; Febbo, P.G.; Ross, K.; Jackson, D.G.; Manola, J.; Ladd, C.; Tamayo, P.; Renshaw, A.A.; D'Amico, A.V.; Richie, J.P.; others. Gene expression correlates of clinical prostate cancer behavior. *Cancer cell* **2002**, *1*, 203–209.
31. Frank, A.; Asuncion, A. UCI Machine Learning Repository [http://archive. ics. uci. edu/ml]. Irvine, CA: University of California. *School of information and computer science* **2010**, *213*, 2–2.
32. Garcia, S.; Derrac, J.; Cano, J.; Herrera, F. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE transactions on pattern analysis and machine intelligence* **2012**, *34*, 417–435.

Sample Availability: Samples of the compounds ...... are available from the authors.