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

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

Tsinghua University[1], *Brandeis University*[2], *Harvard University*[3]

1. Introduction

Feature selection is one of the most fundamental problems in machine learning [Fukunaga, 2013]. Due to the simplicity and effectiveness, the Relief algorithm by Kira and Rendell [1992] has been proven to be one of the most successful feature selection algorithms. Following the large margin principle, it is interpreted to be an online learning algorithm that solves a convex optimization problem with a margin-based cost function. Compared with exhaustive or heuristic combinatorial searches, Relief decomposes a complex, global and nonlinear classification task into a simple and local one. Relief is the first algorithm that uses hypothesis-margin to calculate feature weights for the purpose of classification. See Gilad-Bachrach et al. [2004] for a formal definition of the hypothesis-margin. Besides, the same idea are shared among early Relief-based algorithms, namely a margin is defined by the fixed 1-nearest-neighbor. Kononenko [1994] extended Relief to ReliefF, which uses multiple nearest neighbours to adjust feature weights. Gilad-Bachrach et al. [2004] proposed Simba, which updates the nearest neighbours every time when the feature weights are updated. Sun and Wu [2008] extended Relief from feature selection to feature extraction using local information. Some competitive feature selection methods have also been proposed based on the large hypothesis-margin principle [Crammer et al., 2003; Yang et al., 2008]. In particular, Sun and Li [2006] developed a Relief-based framework to include global information. Based on this framework, IM4E was proposed by Bei and Hong [2015] to balance margin-quantity maximization and margin-quality maximization. Besides Relief-based algorithms, the large margin method has been widely discussed in machine learning [Weinberger and Saul, 2009; Hariharan et al., 2010; Zhu et al., 2016].

Another appealing property of the Relief-based algorithms is that they allow the examination of the knowledge associated with the nearest neighbors, which may not be reflected by the individual features. However, although feature interactions are indirectly considered in the Relief-based algorithms by normalizing the feature weights, natural effects of association cannot be reflected by feature weights of Relief. For example, Relief and many of its extensions do not tell us whether the cause of a high feature weight is from its linear effect or its interaction with other features [Urbanowicz et al., 2018]. In addition, these methods cannot clearly reveal the influence of interaction terms on the generation capabilities of Relief-based classifiers and in particular, the degree of such influence, which is the motivation of our work.

To this end, ***I****terative* ***M****ax-****MI****n entropy mar****G****in-maximization with inte****RA****ction* ***TE****rms* algorithm (IMMIGRATE, henceforth) is proposed in this paper. It has the following novelties. (1) Taking the stability of margin contributors' distribution into account, a generalized quadratic form distance ($d(\vec{x}_i, \vec{x}_j) = \left|\vec{x}_i - \vec{x}_j\right|^T \mathbf{W} \left|\vec{x}_i - \vec{x}_j\right|$, $\mathbf{W} \geq 0$, $\|\mathbf{W}\|_F^2 = 1$) based framework is proposed to capture interaction terms. (2) An iterative optimization method is designed for minimizing the cost function efficiently with a closed-form solution for matrix update. (3) A novel prediction method is proposed to apply the weight matrix by expected generalized distance. (4) A new classifier with stronger generation capability is formed by applying Boosting algorithm. Experimental results show that IMMIGRATE achieves state-of-the-art results compared with most classifiers. Meanwhile, Boosted IMMIGRATE outperforms other Boosting classifiers significantly. Finally, to efficiently implement IMMIGRATE on high-dimensional dataset, IM4E-IMMIGRATE algorithm is designed for effective selection of interaction terms on them. Gene expression datasets are used to demonstrate the effectiveness of the IM4E-IMMIGRATE algorithm. What's more, the computation time of IMMIGRATE is comparable to other popular feature selection method with interaction terms. Moreover, proposed IMMIGRATE and distance metric learning [Xing et al., 2003; Weinberger and Saul, 2009] share the similar distance metric form. However, our method are quite different. Besides different frameworks, in particular, we use generalized quadratic form distance instead of distance metric, where $\mathbf{W}$ does not have to be positive defined. And the normalized $\mathbf{W}$ represents weights for corresponding features or interactions, where more important terms have larger weights.

The rest of the paper is organized as follows. Section 2 explains the mathematical foundation of Relief algorithms. IMMIGRATE algorithm is proposed in Section 3. Section 4 summarizes and discusses the experiments on synthetic dataset, gene expression datasets and UCI datasets. The paper is concluded and discussed in Section 5.

2. Review: Relief Algorithm

The Relief algorithm [Kira and Rendell, 1992] provides a framework for calculating feature weights using a fixed number of instances. Feature weights are usually referred to as feature "scores" and range from -1 to 1. After calculating the feature weights, a certain threshold is set to select relevant features and discard irrelevant features. Relief can be viewed as a convex optimization problem with a cost function whose mathematical expression is shown in Eq. 2.1 if the threshold is set to be 0. Relief minimizes

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

where $NH(x)$ is the nearest "hit" (from the same class) of x; $NM(x)$ is the nearest "miss" (from a different class) of x; $\big|\vec{x}_n - NH(\vec{x}_n)\big|$ calculates the absolute element-wise differences; $\vec{w}^T\big|\vec{x}_n - NH(\vec{x}_n)\big|$ is the weighted Manhattan distance. If we denote $\vec{u} = \sum_{n=1}^{M} \big(\big|\vec{x}_n - NH(\vec{x}_n)\big| - \big|\vec{x}_n - NM(\vec{x}_n)\big|\big)$, Eq. 2.1 can be simplified as $C = \vec{w}^T\vec{u}$. Minimizing C can be solved using the Lagrange multiplier method with the Lagrangian function $L = \vec{w}^T\vec{z} + \lambda(\|\vec{w}\|_2^2 - 1) + (-\vec{w}^T)\vec{\theta}$, where $\lambda \geq 0$ and $\vec{\theta} \geq 0$. Using the Karush-Kuhn-Tucker condition by Kuhn and Tucker [2014], the solution is straightforward: $\vec{w} = (-\vec{u})^+/\|(-\vec{u})^+\|_2$, where $(\vec{a})^+ = [max(a_1, 0), max(a_2, 0), \cdots, max(a_A, 0)]$. This solution to the original Relief algorithm is fundamental to understanding the Relief-based algorithms. The pseudo code of the Relief algorithm is listed in Algorithm 1.

Algorithm 1 Original Relief Algorithm

```
State : Feature selection for binary classification
$N \leftarrow$ number of training instances
$A \leftarrow$ number of features(i.e. attributes)
$M \leftarrow$ number of randomly chosen instances out of $M$ to update weight $\vec{w}$
initial all features weights to 0: $\vec{w} := 0$
  for $i := 1$ to $M$ do
    randomly select a "target" instance $x_i$
    find its $NH(x_i)$ and $NM(x_i)$ in Eq. 2.1
    for $a := 1$ to A do
      $w[a] := w[a] - (x_i[a] - NH(x_i)[a])^2/M + (x_i[a] - NM(x_i)[a])^2/M$
    end for
  end for
  return the vector $W$ of feature scores that estimate the quality of features
```

3. IMMIGRATE Algorithm

The framework of IMMIGRATE algorithm is established in this section.

Let $\mathcal{D} = \{z_n | z_n = (\vec{x}_n, y_n) \in \mathcal{R}^{A+1}, \vec{x}_n \in \mathcal{R}^A, y_n \in \{-1, 1\}\}_{n=1}^N$, where N is the number of instances; A is the number of features; $\vec{x}_n$ represents feature vector and y_n is class. Only binary classification is considered in this formulation. Since binary classification is the basic classification task in machine learning, it is often used to test the performance of feature selection algorithms. The definitions in our margin-based framework (e.g., hits and misses) make it easy to extend a binary classification formulation to a multiple classification problem. Our IMMIGRATE implementation in R is applicable for multiple classification tasks. Here, we define the following notations $\mathcal{H}_n = \{j \in \{1, 2, \cdots, N\} | z_j \in \mathcal{D}, y_j = y_n \; and \; j \neq n\}$, $\mathcal{M}_n = \{j \in \{1, 2, \cdots, N\} | z_j \in \mathcal{D}, y_j \neq y_n\}$, where $\mathcal{H}_n$ and $\mathcal{M}_n$ represent the index sets of hits and misses of the instance z_n, respectively.

3.1. Max-Min Entropy Principle

Given a generalized distance metric $d(\vec{x}_i, \vec{x}_j)$ between two instances $\vec{x}_i$ and $\vec{x}_j$, a hypothesis-margin [Gilad-Bachrach et al., 2004] is defined as

$$\rho_{n,h,m} = d(\vec{x}_n, \vec{x}_m) - d(\vec{x}_n, \vec{x}_h), \tag{3.2}$$

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 $\vec{x}_n$ and $\mathcal{H}_n$, $\mathcal{M}_n$ are the index sets for hits and misses separately. Since the generalized distance metric is unknown due to the unknown feature weights, the nearest hit and nearest miss are undetermined under such a framework. Hence, the method proposed in Sun and Li [2006], Sun and Wu [2008], Sun et al. [2010] and Bei and Hong [2015] is adopted, where the margin is defined as follows.

$$\rho_n = \sum_{m \in \mathcal{M}_n} \beta_{n,m} d(\vec{x}_n, \vec{x}_m) - \sum_{h \in \mathcal{H}_n} \alpha_{n,h} d(\vec{x}_n, \vec{x}_h) \tag{3.3}$$

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, hidden random variable $\alpha_{n,h}$ represents the probability that $\vec{x}_h$ is the nearest hit of instance $\vec{x}_n$, while hidden variable $\beta_{n,m}$ indicates the probability that $\vec{x}_m$ is the nearest miss of instance $\vec{x}_n$. The derivations of $\alpha_{n,h}$ and $\beta_{n,m}$ are closely related to the generalized distance metric.

As proposed, the probabilities $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$ represent some distribution of hits and misses. The stability of margin contributors' distribution of $\vec{x}_n$ can be defined using its hit probabilities $\{\alpha_{n,h}\}$ and miss probabilities $\{\beta_{n,m}\}$. Thus, the hit entropy and miss entropy are respectively defined as $E_{hit} = -\sum_{h \in \mathcal{H}_n} \alpha_{n,h} \log \alpha_{n,h}$ and $E_{miss} = -\sum_{m \in \mathcal{M}_n} \beta_{n,m} \log \beta_{n,m}$.

The following two scenarios help to explain the intuition of using the hit entropy and miss entropy. In scenario A, all neighbours are distributed evenly around the target instance. In scenario B, the neighbour distribution is highly uneven. In particular, 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 margin contributors' distribution is to discard one instance from the system and then the change degree is checked out. Firstly, in scenario A, if the 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 nearest hit can largely reduce the margin since its hit probabilities are concentrated at a few hits. Thus, hit probabilities prefer large entropy, such as scenario A. Meanwhile. The miss probabilities prefer small entropy (e.g., scenario B) because the disappearance of the nearest miss can largely increase the margin. In such a max-min entropy framework, the hit entropy should be maximized and the miss entropy should be minimized. This max-min entropy principle [Bei and Hong, 2015] is an extension of the large margin principle and the hit entropy and miss entropy are optimized to be consistent with the large margin principle.

3.2. IMMIGRATE Algorithm

To capture feature interactions, the IMMIGRATE algorithm is developed by choosing the distance metric in framework Eq. 3.3 as a generalized quadratic form distance (the Mahalanobis distance). The "IMMIGRATE" stands for ***I**terative **M**ax-**MI**n entropy mar**G**in-maximization with inte**RA**ction **TE**rms algorithm*.

The quadratic form distance for instances z_i and z_j with a weight matrix $\mathbf{W}$ is defined as

$$d(\vec{x}_i, \vec{x}_j) = \left|\vec{x}_i - \vec{x}_j\right|^T \mathbf{W} \left|\vec{x}_i - \vec{x}_j\right|, \tag{3.4}$$

where $\mathbf{W} \geq 0$ and $\|\mathbf{W}\|_F^2 = 1$. The weight matrix is a natural extension of the weight vector since a weight vector can be represented as a diagonal matrix. The cost function Eq. 3.5 is designed to maximize the generalized margin under max-min entropy principle.

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

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, λ are both tune parameters.$\|\mathbf{W}\|_F^2$ is the Frobenius norm of $\mathbf{W}$.

$\|\mathbf{W}\|_F^2 = \sum_{i,j} w_{i,j}^2 = \sum_i \lambda_i^2$, with λ_i's are eigenvalues of matrix $\mathbf{W}$.

Now an iterative optimization framework is proposed and three steps are included to iteratively minimize the cost function. The framework starts from a randomly generated weight matrix as the prior one and ends up until the change of cost reaches a preset limit, and then the posterior one is used as the final output.

Step 1: The optimization of cost function Eq. 3.5 starts from a randomly initialized $\mathbf{W}$ (satisfying $\mathbf{W} \geq 0$ and $\|\mathbf{W}\|_F^2 = 1$). Then 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 $\mathbf{W}$.

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

Fixing $\mathbf{W}$, $\alpha_{n,h}$ and $\beta_{n,m}$ can be obtained from Eq. 3.6, where $d(\vec{x}_i,\vec{x}_j) = \left|\vec{x}_i - \vec{x}_j\right|^T \mathbf{W}\left|\vec{x}_i - \vec{x}_j\right|$ is the newly defined distance metric.

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

The derivative of the cost function with respect to $(\alpha_{n,h}, \beta_{n,m})$ is

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

$$\left|\frac{\partial^2 C}{\partial(\alpha_{n,h},\beta_{n,m})}\right| = -\frac{\sigma^2}{(\alpha_{n,h}\beta_{n,m})} - (\frac{\partial^2 C}{\partial\beta_{n,m}\alpha_{n,h}})^2 < 0. \tag{3.8}$$

Therefore, when fixing $\mathbf{W}$, a saddle point in $(\alpha_{n,h}, \beta_{n,m})$ space can be found.

3.2.2. Fix $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$, Update $\boldsymbol{W}$

Fixing $\alpha_{n,h}$ and $\beta_{n,m}$, the derivation of $\mathbf{W}$ is a little computationally arduous. However, a closed form solution for $\mathbf{W}$ is derived in Theorem 1.

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

$$\mathbf{\Sigma} = \sum_{n=1}^{N} \Sigma_{n,H} - \Sigma_{n,M}, \; \mathbf{\Sigma}\ \psi_i = \mu_i\ \psi_i, \tag{3.9}$$

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

$$\mathbf{W} = \Phi\, \Phi^T, \tag{3.10}$$

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

The proof of Theorem 1 is in the supplementary material.

Under the iterative optimization framework, start with Step 1, the optimization algorithm iteratively executes Steps 2 and 3 until convergence. The hyperparameter σ is tuned by cross-validation. In application, when the change of objective cost function is less than the preset limit, the iterative optimization will stop. This procedure is shown computationally efficient for IMMIGRATE algorithm.

3.2.3. Remove Small Weights

When there are too many features and interaction terms, overfitting is easily caused in the operation. Thus, we add an option in IMMIGRATE algorithm to remove small weights. The following added Step 4 is optional for implement. Step 4: Set weights in $\mathbf{W}$ which are smaller than a preset threshold to be 0, and normalize the $\mathbf{W}$. Notice that with randomly initialized matrix $\mathbf{W}$, the weights of some important features or interaction terms might be

Algorithm 2 IMMIGRATE Algorithm

```
State : Feature selection for binary classification
 $N \leftarrow$ number of training instances
 $A \leftarrow$ number of features(i.e. attributes)
Input : a training dataset $\{z_n = (\vec{x}_n, y_n)\}_{n=1,\cdots,N}$
Initialize: Let $t = 0$, randomly initialize $\mathbf{W}^{(0)}$ satisfying nonnegative $\mathbf{W}^{(0)} \geq 0$, and $\|\mathbf{W}^{(0)}\|_F^2 = 1$

1: repeat
2:     Calculate $\{\alpha_{n,h}^{(t+1)}\}$ and $\{\beta_{n,m}^{(t+1)}\}$ using Eq. 3.6 with $d(\vec{x}_i, \vec{x}_j) = \left|\vec{x}_i - \vec{x}_j\right|^T \mathbf{W}^{(t)} \left|\vec{x}_i - \vec{x}_j\right|$
3:     Calculate $\mathbf{W}^{(t+1)}$ using Theorem 1 and Eq. 5.17.
4: until the change of C in Eq. 3.5 is small enough or the iteration indicator $t$ reaches a preset limit
Return $\mathbf{W}^{(t)}$
```

small at the first several step of convergence. Thus, Step 4 will be added to the iterative optimization procedure after several iteration. Here, we set that Step 2, 3, 4 are iterated to minimize the cost function after half of the maximum iteration number(preset limit).

The original Relief-based algorithms use a preset threshold to select features, and the corresponding weights which are lower than the threshold will be discarded. The remaining features are selected ones and K-nearest neighbor classifier is used for classifying new samples.

In IMMIGRATE algorithm, the small weights are removed as the feature selection procedure of original Relief-based algorithms. What's more, since the weight matrix is normalized where the Frobenius norm of $\mathbf{W}$ is equal to 1, the larger weight indicates more important feature or interaction terms. In other words, important features and interaction terms are not only selected, but also ranked according to its relevance to the outcome.

3.2.4. New Prediction Approach

The prediction method is improved by the learned weight matrix **W** as expected generalized distance.

$$y' = \arg\min_{c} \sum_{y_n=c} \alpha_n^c(\vec{x}')d(\vec{x}', \vec{x}_n), \tag{3.11}$$

where c denotes the class, and

$$\alpha_n^c(\vec{x}') = \frac{exp\big(-d(\vec{x}', \vec{x}_n)/\sigma\big)}{\sum_{y_k=c} exp\big(-d(\vec{x}', \vec{x}_k)/\sigma\big)}, \tag{3.12}$$

For a new sample $z' = (\vec{x}', y')$, we use the learned weight matrix **W** to select a class for z' using Eq. 3.11 and Eq. 3.12, where $d(\vec{x}', \vec{x}_n) = \big|\vec{x}' - \vec{x}_n\big|^T \mathbf{W} \big|\vec{x}' - \vec{x}_n\big|$.

Here, the new samples are divided in the class of the shortest expected generalized distance. Compared with original Relief-based algorithm, the obtained weights for features and interaction terms are exploited by the expected generalized distance in Eq. 3.11 and Eq. 3.12.

Actually, this new prediction method is to maximize the margin contribution of $\vec{x}'$ (Eq. 3.3):

$$\Bigg(\sum_{m\in\mathcal{M}_{\vec{x}'}} \beta_{n,m} d(\vec{x}', \vec{x}_m) - \sum_{h\in\mathcal{H}_{\vec{x}'}} \alpha_{n,h} d(\vec{x}', \vec{x}_h)\Bigg), \tag{3.13}$$

where $\mathcal{H}_{\vec{x}'}$ and $\mathcal{M}_{\vec{x}'}$ are the index sets of hits and misses of the new sample $\vec{x}'$ separately. In Eq. 3.11, we choose a class c whose corresponding index set of hits is $\mathcal{H}_{\vec{x}'}$. For other classes except c, the corresponding samples are included in the misses. To minimize Eq. 3.11, a class is selected to minimize the second term and maximize the first term in Eq. 3.13, in which the margin contribution of $\vec{x}'$ is maximized.

3.3. Boosting for IMMIGRATE Algorithm

Boosting methods [Schapire, 1990] have been widely applied in machine learning by using a set of weak learners to create a strong learner. And they have proved very effective in improving the generalization capabilities of many classification algorithms. For example, as three widely used Boosting based algorithms, Freund et al. [1996]; Freund and Mason [1999], Random Forest Liaw et al. [2002], and XgBoost Chen and Guestrin [2016] are shown competitive in many applications in terms of prediction accuracy.

IMMIGRATE can also be used as the base classifier in Boosting algorithm. Here, using IMMIGRATE as the base classifier, AdaBoost algorithm in Freund et al. [1996] is applied to propose Boosting algorithm for IMMIGRATE. Since sample weights are required for AdaBoost classifier, the cost function of IMMIGRATE is changed from Eq. 3.5 to Eq. 3.14, where $D(x)$ is the corresponding sample weight for sample x.

Notice that in Algorithm 2, the IMMIGRATE classifier gets stronger in the process of convergence. The random initialized distance matrix $\mathbf{W}$ can provide different generalization capabilities for IMMIGRATE classifiers. Here, we propose an appropriate application of Boosting algorithm to IMMIGRATE classifier, which is called BIM (***B***_OOSTED_ ***IM***_MIGRATE Algorithm_) for short. In BIM, the maximum iteration number is limited for each IMMIGRATE classifier to provide different generalization capabilities and create relatively weak learners. Also, different hyperparameter σ can provide different capabilities for IMMIGRATE classifiers. Algorithm 3 shows the procedure for BIM algorithm.

$$\begin{aligned}
C = &\sum_{n=1}^{N} D(\vec{x}_n)\Big(\sum_{h\in\mathcal{H}_n}\alpha_{n,h}\left|\vec{x}_n-\vec{x}_h\right|^T\mathbf{W}\left|\vec{x}_n-\vec{x}_h\right| - \sum_{m\in\mathcal{M}_n}\beta_{n,m}\left|\vec{x}_n-\vec{x}_m\right|^T\mathbf{W}\left|\vec{x}_n-\vec{x}_m\right|\Big)\\
&+\sigma\sum_{n=1}^{N} D(\vec{x}_n)[E_{miss}(z_n)-E_{hit}(z_n)],\\
&\textit{subject to}:\ \mathbf{W}\geq 0 \textit{ and } \|\mathbf{W}\|_F^2=1,\\
&\sum_{h\in\mathcal{H}_n}\alpha_{n,h}=1\ ,\ \sum_{m\in\mathcal{M}_n}\beta_{n,m}=1,\ \ \forall\ n,\\
&\quad\alpha_{n,h}\geq\ 0\ \beta_{n,m}\geq\ 0,\ \ \forall\ n,\\
&\sum_{n=1}^{N} D(\vec{x}_n)=1,\ \ D(\vec{x}_n)\geq 0,\ \ \forall\ n,
\end{aligned} \tag{3.14}$$

Algorithm 3 BIM Algorithm

```
State : Boosted IMMIGRATE for binary classification
  $N \leftarrow$ number of training instances
  $A \leftarrow$ number of features(i.e. attributes)
  $T \leftarrow$ number of classifiers for BIM
Input : a training dataset $\{z_n = (\vec{x}_n, y_n)\}_{n=1,\cdots,N}$
Initialize : for each $\vec{x}_n$, set $D_1(\vec{x}_n) = 1/N$

 1: for $t := 1$ to $T$ do
 2:     Limit Max Iteration of IMMIGRATE less than preset
 3:     Train weak IMMIGRATE classifier $h_t(x)$ using a chosen $\sigma_t$ and weights
        $D_t(x)$ by Eq. 3.14
 4:     Compute the error rate $\epsilon_t$ as
            $$\epsilon_t = \sum_{i=1}^{N} D_t(x_i) I[y_i \neq h_t(x_i)]$$
 5:     if $\epsilon_t \geq 1/2$ or $\epsilon_t = 0$ then
 6:         Discard $h_t$, $T = T - 1$ and Continue
 7:     end if
 8:     Set $\alpha_t = 0.5 \times \log((1 - \epsilon_t)/\epsilon_t)$
 9:     Update $D(x_i)$: For each $x_i$,
          $D_{t+1}(x_i) = D_t(x_i) \exp(\alpha_t I[y_i \neq h_t(x_i)])$
10:     Normalize $D_{t+1}(x_i)$, so that $\sum_{i=1}^{N} D_{t+1}(x_i) = 1$
11: end for
Output:
            $$h_{final}(x) = \arg\max_{y \in \{0,1\}} \sum_{t: h_t(x) = y} \alpha_t$$
```

3.4. IMMIGRATE on high-dimensional data

High-dimensional data are attractive for feature selection tasks Fan and Li [2006]. Since the interactions between each two features are considered in IMMIGRATE, IM4E-IMMIGRATE algorithm is designed to be computationally efficient to facilitate the implementation on high-dimensional data.

Firstly, apply IM4E to obtain a weight vector. Then, select a certain number of features based on the weight vector learned by IM4E(choose features whose corresponding weights are above a preset threshold). Use the selected features and initialize the weight matrix of IMMIGRATE to be consistent with the weight vector learned by IM4E (diagonal one). Finally, IMMIGRATE is implemented.

IM4E-IMMIGRATE is a sub-optimal solution to facilitate IMMIGRATE on high dimensional data. But it is effective and efficient to be implemented. Moreover, it can also be boosted to make a stronger version.

4. Experiments

All the algorithms used in our experiments either have implementations in R [Team et al., 2013] or have been implemented by us in R(compiled R package can be found in the supplementary material), such as Simba [Gilad-Bachrach et al., 2004], LFE [Sun and Wu, 2008], IM4E [Bei and Hong, 2015], IMMIGRATE and BIM.

4.1. Results on Synthetic Dataset

In this experiment, the robustness of IMMIGRATE algorithm is tested using synthesized dataset where two features are purposely designed for interaction term selection. A feature selection algorithm is called robust if the results obtained from original dataset are consistent with the ones from datasets with noises. We generate dataset1 with 200 samples as follows. 100 samples with class 0 and 100 samples with class 1 are randomly generated from Gaussian distributions with mean [4,2], variance diag[1,1] and with mean[6,0], variance diag[1,1] separately. Noises with class 0 and class 1 are randomly generated from a Gaussian distribution with mean [8,-2], variance diag[8,8] and with mean[2,4], variance diag[8,8], respectively. The scatter plot of dataset with 10% noise is shown in Fig. 1. The noises are designed to disturb the detection of the interaction term. The level of noises starts from 0.05, and gradually increases 0.05 each time until greater than 0.5. IMMIGRATE and LFE are run on the synthesized dataset and the weights corresponding to the interaction term between features 1 and 2 are collected. Interaction weights learned by IMMIGRATE and LFE are plotted in Fig. 2, which clearly shows that IMMIGRATE is more robust than LFE.

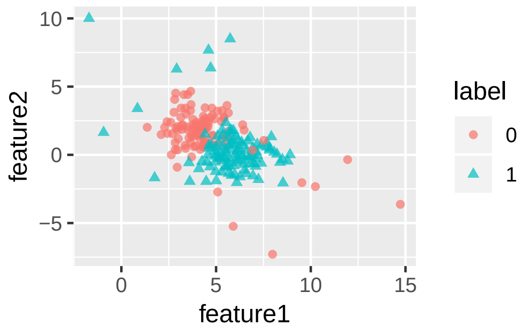


Figure 1: Synthesized Dataset1 with 10% noise.

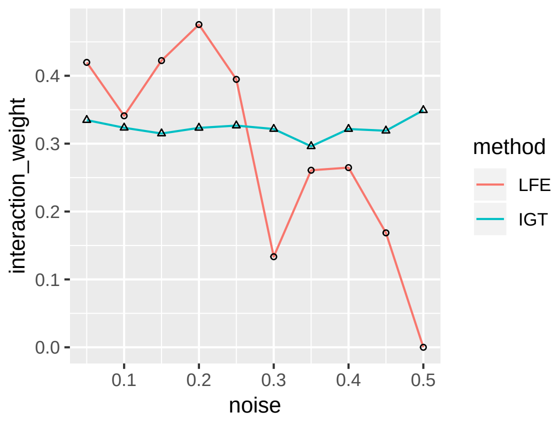


Figure 2: IMMIGRATE is highly robust in learning the weights, which indicate the interaction between two features, from noisy datasets.

4.2. Results on Real Datasets

IMMIGRATE is also compared with several existing popular methods using real datasets from UCI database. Cross validation is used to test the performance. The following existing algorithms are used in this experiment: Support Vector Machine [Soentpiet et al., 1999] with Sigmoid Kernel (SV1), with Radial basis function Kernel (SV2), LASSO [Tibshirani, 1996] (LAS),

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]	0,0,$\underline{\mathbf{5}}$	1,0,$\underline{\mathbf{4}}$	0,1,$\underline{\mathbf{4}}$	0,0,$\underline{\mathbf{5}}$	0,0,$\underline{\mathbf{5}}$	0,0,$\underline{\mathbf{5}}$	1,0,$\underline{\mathbf{4}}$	0,0,$\underline{\mathbf{5}}$	1,1,$\underline{\mathbf{3}}$	1,0,$\underline{\mathbf{4}}$	1,1,$\underline{\mathbf{3}}$	-,-,-	-,-,-

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

Decision Tree [Freund and Mason, 1999] (DT), Naive Bayes Classifier [John and Langley, 1995] (NBC), Radial basis function Network [Haykin, 1994] (RBF), 1-Nearest Neighbor [Aha et al., 1991] (1NN), 3-Nearest Neighbor (3NN), Relief [Kira and Rendell, 1992] (REL), ReliefF [Kononenko, 1994; Robnik-Šikonja and Kononenko, 2003] (RFF), Simba [Gilad-Bachrach et al., 2004] (SIM), Linear Discriminant Analysis [Fisher, 1936] (LDA). In addition, several methods designed for detecting interaction terms are included: LFE [Sun and Wu, 2008], Stepwise conditional likelihood variable selection for Discriminant Analysis [Li and Liu, 2018] (SOD), hierNet [Bien et al., 2013] (HIN). What's more, for the comparison among Boosting methods, three most widely used and competitive ones are used: Adaptive Boosting Freund et al. [1996]; Freund and Mason [1999] (ADB), Random Forest Liaw et al. [2002] (RF), and XgBoost Chen and Guestrin [2016] (XGB). In the discussion of the experimental results, IM4E is abbreviated as IM4, IMMIGRATE as IGT, BIM as BIM and Boosted IM4E-IMMIGRATE as B4G.

These methods are set in the same way suggested in original papers: Relief and Simba use Euclidean distance and 1-NN classifier; ReliefF use 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, where we fix λ as 1 since it has no actual contribution and tune σ as suggested in Bei and Hong [2015]. The IMMIGRATE algorithm has one hyperparameter σ. When tuning σ, σ starts from $\sigma_0 = 4$ and gradually decreases to half each time until it is not larger than 0.2. For large-scale datasets, we choose σ which gives us the best results. The BIM algorithm uses σ_t, $\sigma_{max} = 4$, $\sigma_{min} = 0.2$ are chosen, the number of classifiers: $T = 100$, σ_t starts from σ_{max} and gradually decreases to $\sigma_t \times (\sigma_{min}/\sigma_{max})^{1/T}$ each time until it is not greater than σ_{min}. The preset threshold in IM4E-IMMIGRATE is $1/A$, where A is the number of features.

4.2.1. Results on Gene Expression Datasets

Gene expression datasets typically have thousands of features. Comparison on five publicly available gene expression datasets are carried out: GLI[Freije et al., 2004], Colon[Alon et al., 1999](COL), Myeloma[Tian et al., 2003](ELO), Breast[Van't Veer et al., 2002](BRE), Prostate[Singh et al., 2002](PRO). All five datasets have more than ten thousand features. Feature selection methods are widely tested in these high-dimensional datasets.

10-fold cross-validation is performed for ten times, namely 100 trials are carried out. The average accuracy is reported on the corresponding datasets in Table 1. The last row "(W,T,L)" indicates the number of times each algorithm W,T,L (win,tie,loss) when compared with Boosted IM4E-IMMIGRATE (B4G) by the paired Student's t-test with the significance level of $\alpha = 0.05$. Algorithm A is significantly better than (i.e. win) another algorithm B on a dataset C if the p-value of the paired Student's t-test with corresponding null hypothesis is less than $\alpha = 0.05$.

As shown in Table 1, although Boosted IM4E-IMMIGRATE (B4G) is not always the best, it outperforms other methods in most cases. In particular, when comparing IM4E-IMMIGRATE(EGT) with other methods, it also outperforms in most cases.

4.2.2. Results on UCI Datasets

UCI datasets from Frank and Asuncion [2010] are used to compare the performance of a cohort of algorithms. The used datasets include Breast Cancer Wisconsin (Prognostic) (BCW), Breast Cancer (BRC), Cryotherapy (CRY), Wholesale customers (CUS), Ecoli (ECO), Glass Identification (GLA), Haberman's Survival (HMS), Immunotherapy (IMM), Ionosphere (ION), Lymphograph (LYM), MONK's Problems (MON), Parkinsons (PAR), Pima-Indians-Diabetes (PID), Connectionist Bench (Sonar, Mines vs. Rocks)

(SMR), Statlog (Heart) (STA) Urban Land Cover (URB), User Knowledge Modeling (USE) and Wine (WIN). For datasets with more than two classes, the largest two classes are used in this experiment. In addition, we use three large-scale data with respect to the sample size: Waveform Database Generator (WAV, 3304 instances), Crowdsourced Mapping (CRO, 9903) and Electrical Grid Stability Simulated (ELE, 10000).

10-fold cross-validation is also performed for ten times. Tables 2 and 3 show the average accuracy on the corresponding datasets. The last row "(W,T,L)" indicates the number of times each algorithm W,T,L (win,tie,loss) when compared with IMMIGRATE(IGT) in Table 2 and BIM in Table 3 separately by using the paired Student's t-test with the significance level of $\alpha = 0.05$.

Although IMMIGRATE or BIM is not always the best, it outperforms other methods significantly in most cases. Based on Table 2, IMMIGRATE and BIM achieve state-of-the-art performance as base classifier and booster version separately.

Moreover, to show the results of feature weights, the heat maps of four datasets: GLA, LYM, SMR and STA are supplemented.

Table 2: Summarizes the accuracies on UCI datasets.

Data	SV1	SV2	LAS	DT	NBC	RBF	1NN	3NN	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	66.4	67.1	67.7	67.1	73.9	65.2	71.8	66.4	**74.5**
BRC	65.1	75.5	72.9	68.9	70.3	64.0	68.1	69.0	56.0	58.6	62.2	65.0	72.0	79.7	77.5	75.8	**75.2**
CRY	72.9	90.6	87.4	85.3	84.4	89.7	89.1	85.4	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	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	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	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	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	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	85.8	86.2	84.2	91.0	83.3	90.3	92.6	88.3	**92.6**
LYM	83.6	81.5	85.2	75.2	83.6	71.1	77.2	82.8	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	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	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	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	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	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	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	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	87.2	95.0	95.0	93.8	99.7	92.9	98.9	98.2	**99.0**
W,T,L[1]	0,0,**18**	2,2,**14**	1,3,**14**	1,3,**14**	0,1,**17**	0,0,**18**	1,1,**16**	0,1,**17**	0,0,**18**	0,0,**18**	0,1,**17**	0,1,**17**	1,4,**13**	3,2,**13**	1,6,**11**	1,0,**17**	-,-,-

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

Table 3: Summarizes the accuracies on UCI datasets.

Data	ADB	RF	XGB	**BIM**
BCW	78.2	78.6	78.6	**78.3**
BRC	73.2	71.8	73.8	**75.9**
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	**92.9**
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	**76.7**
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**
W,T,L[1]	0,3,<u>**15**</u>	2,6,<u>**10**</u>	1,4,<u>**13**</u>	-,-,-

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

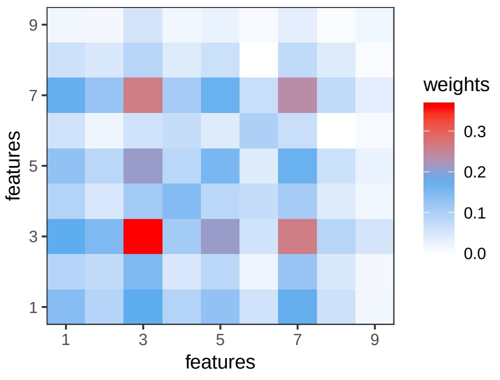
5. Conclusion & Discussion

In this paper, a novel feature selection algorithm IMMIGRATE is proposed for detecting and ranking interaction terms, including the extended version of Boosted IMMIGRATE(BIM) and IM4E-IMMIGRATE. Large margin and max-min entropy principle are used to present a generalized quadratic form distance based framework for feature learning. Non-linear margin-based cost function is proposed. To minimize the cost function, an iterative optimization framework is designed for implementing the IMMIGRATE algorithm and the close-form of matrix update is derived in Theorem 1. IMMIGRATE outperforms most methods in tasks and achieve state-of-the-art results. And BIM outperforms other Boosting algorithms. 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) state-of-the-art generalization capabilities; (5) easily boosted. What's more, the computation time of IMMIGRATE is comparable to other feature selection methods with interaction terms, such as SODA, hierNet. The well compiled codes for IMMIGRATE and BIM are matrix-based, where parallel computing can be implemented to accelerate.

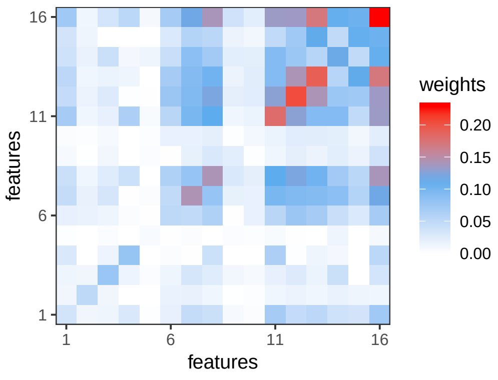
Several points are required to be further explored in this work. First, since the margin-based framework is not restricted to binary classification scenarios, multiple class classification can be achieved easily based on this work. Second, IMMIGRATE only considers pair-wise interactions between features. Interaction terms among multiple features also play an important role in many cases. Our work provides a basis for exploring new algorithms for detecting interactions among multiple features. Third, thresholds are set artificially in IMMIGRATE if we want to remove small weights to choose relevant features or interactions. $l1$-regularization has been shown by Ng [2004] to have advantages on finding sparse solutions, which is quite valuable for high-dimensional data. Thus, if $l1$-regularization can be imposed into the current cost function, the performance of IMMIGRATE on high-dimensional data might be improved. The encountered overfitting problem in some cases can be avoided. Fourth, the strategy of selecting σ is remaining open to solve. Based on the experimental results, to achieve our best results, different σ's are chosen for different datasets.

Supplementary Material

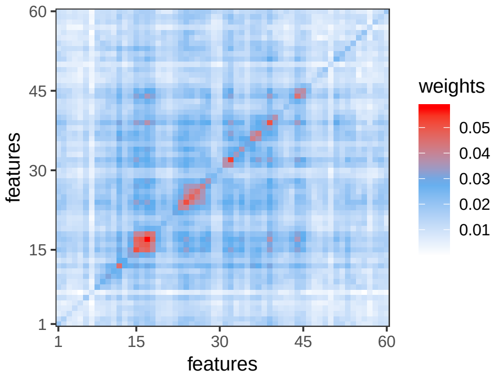
Heat Maps



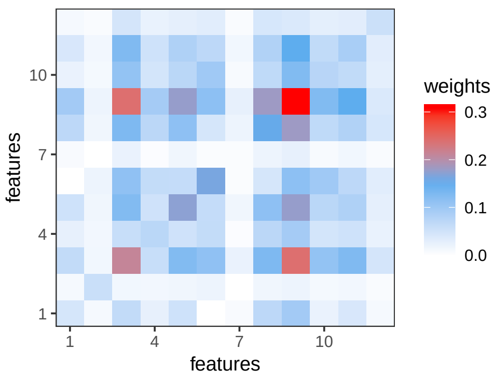
GLA



LYM



SMR



STA

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

Proof of Theorem 3.1

$$\min_{\vec{w}^T} C = \sum_{n=1}^{N}\Big(\sum_{h\in\mathcal{H}_n}\alpha_{n,h}\big|\vec{x}_n-\vec{x}_h\big|^T\mathbf{W}\big|\vec{x}_n-\vec{x}_h\big| - \sum_{m\in\mathcal{M}_n}\beta_{n,m}\big|\vec{x}_n-\vec{x}_m\big|^T\mathbf{W}\big|\vec{x}_n-\vec{x}_m\big|\Big),$$
$$+\,\sigma\sum_{n=1}^{N}[E_{miss}(z_n)-E_{hit}(z_n)],$$
$$\textit{subject to} \;:\; \mathbf{W}\geq 0 \textit{ and } \|\mathbf{W}\|_F^2 = 1,$$
$$\sum_{h\in\mathcal{H}_n}\alpha_{n,h}=1\ ,\ \sum_{m\in\mathcal{M}_n}\beta_{n,m}=1\ \forall\ n, \tag{5.15}$$

where $\|\mathbf{W}\|_F^2$ is the Frobenius norm of $\mathbf{W}$. $\|\mathbf{W}\|_F^2 = \sum_{i,j} w_{i,j}^2 = \sum_i \lambda_i^2$, with λ_is are eigenvalues of matrix $\mathbf{W}$.

Theorem 1. *Fixing $\{\alpha_{n,h}\}$ and $\{\beta_{n,m}\}$, the cost function eq.3.5 has a closed-form solution for updating* $\boldsymbol{W}$.

$$\boldsymbol{\Sigma} = \sum_{n=1}^{N}\Sigma_{n,H}-\Sigma_{n,M},\ \boldsymbol{\Sigma}\ \psi_i = \mu_i\ \psi_i, \tag{5.16}$$

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

$$\mathbf{W} = \Phi\,\Phi^T, \tag{5.17}$$

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 of Thm. 3.1

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

$$\mathbf{W} = P\Lambda P^T = P\Lambda^{1/2}\Lambda^{1/2}P^T,$$
$$= [\sqrt{\lambda_1}\ p_1,\cdots,\sqrt{\lambda_A}\ p_A][\sqrt{\lambda_1}\ p_1,\cdots,\sqrt{\lambda_A}\ p_A]^T, \tag{5.18}$$

where P is 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 $\lambda_1 > \lambda_2 > \cdots > \lambda_A$.

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

$$\|\mathbf{W}\|_F^2 = \sum_{i,j} w_{i,j}^2 = \sum_i (\phi_i^T \phi_i)^2 = 1 \tag{5.19}$$

3. Let us rearrange the Eq. 5.15.

$$\begin{aligned} &\sum_{h \in \mathcal{H}_n} \alpha_{n,h} \left|\vec{x}_n - \vec{x}_h\right|^T \mathbf{W} \left|\vec{x}_n - \vec{x}_h\right| = tr(\mathbf{W} \sum_{h \in \mathcal{H}_n} \alpha_{n,h} \left|\vec{x}_n - \vec{x}_h\right| \left|\vec{x}_n - \vec{x}_h\right|^T), \\ &tr(\mathbf{W}\Sigma_{n,H}) = tr(\Sigma_{n,H} \sum_{i=1}^{A} \phi_i \phi_i^T) = \sum_{i=1}^{A} \phi_i^T \Sigma_{n,H} \ \phi_i, \end{aligned} \tag{5.20}$$

Then, Eq. 5.15 can be simplified as follows.

$$\begin{aligned} \min_{\vec{w}^T} C &= \sum_{i=1}^{A} \phi_i^T \Sigma \ \phi_i, \\ subject\ to\ &: \|\mathbf{W}\|_F^2 = \sum_i (\phi_i^T \phi_i)^2 = 1, \langle \phi_i, \phi_j \rangle = 0, \end{aligned} \tag{5.21}$$

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

The orthogonal condition will be ignored by us when deriving the closed form solution because as we can notice at the last step, this condition has already been satisfied.

4. The Lagrangian of Eq. 5.21 is easy to obtain.

$$L = \sum_{i=1}^{A} \phi_i^T \Sigma \ \phi_i + \lambda (\sum_{i=1}^{A} (\phi_i^T \phi_i)^2 - 1), \tag{5.22}$$

Derive L with respect to ϕ_i,

$$\partial L / \partial \phi_i = 2\Sigma\phi_i + 4\lambda\phi_i^T\phi_i\phi_i = 0, \tag{5.23}$$

Denote $\phi_i/\|\phi_i\|_2 := \psi_i$. From Eq. 5.23,

$$\Sigma\ \psi_i = \mu_i\ \psi_i, \tag{5.24}$$

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

5. 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. 5.21 can be simplified to be

$$\min_{\vec{w}^T} C = \sum_{i=1}^{A}\eta_i\mu_i,\ \textit{subject to}\ \sum_{i=1}^{A}(\eta_i)^2 = 1, \eta_i \geq 0 \tag{5.25}$$

6. It is excited to notice Eq. 5.25 is exactly the same as the original Relief Algorithm.

$$\vec{\eta} = (-\vec{\mu})^+/\|(-\vec{\mu})^+\|_2, \tag{5.26}$$

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

$$\mathbf{W} = \Phi\Phi^T \tag{5.27}$$

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

Aha, D. W., Kibler, D., and Albert, M. K. (1991). Instance-based learning algorithms. *Machine learning*, 6(1):37–66.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., and Levine, A. J. (1999). 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*, 96(12):6745–6750.

Bei, Y. and Hong, P. (2015). Maximizing margin quality and quantity. In *Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on*, pages 1–6. IEEE.

Bien, J., Taylor, J., and Tibshirani, R. (2013). A lasso for hierarchical interactions. *Annals of statistics*, 41(3):1111.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM.

Crammer, K., Gilad-Bachrach, R., Navot, A., and Tishby, N. (2003). Margin analysis of the lvq algorithm. In *Advances in neural information processing systems*, pages 479–486.

Fan, J. and Li, R. (2006). Statistical challenges with high dimensionality: Feature selection in knowledge discovery. *arXiv preprint math/0602133*.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

Frank, A. and Asuncion, A. (2010). Uci machine learning repository [http://archive. ics. uci. edu/ml]. irvine, ca: University of california. *School of information and computer science*, 213:2–2.

Freije, W. A., Castro-Vargas, F. E., Fang, Z., Horvath, S., Cloughesy, T., Liau, L. M., Mischel, P. S., and Nelson, S. F. (2004). Gene expression profiling of gliomas strongly predicts survival. *Cancer research*, 64(18):6503–6510.

Freund, Y. and Mason, L. (1999). The alternating decision tree learning algorithm. In *icml*, volume 99, pages 124–133.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *Icml*, volume 96, pages 148–156. Citeseer.

Fukunaga, K. (2013). *Introduction to statistical pattern recognition*. Elsevier.

Garcia, S., Derrac, J., Cano, J., and Herrera, F. (2012). Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE transactions on pattern analysis and machine intelligence*, 34(3):417–435.

Gilad-Bachrach, R., Navot, A., and Tishby, N. (2004). Margin based feature selection-theory and algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, page 43. ACM.

Hariharan, B., Zelnik-Manor, L., Varma, M., and Vishwanathan, S. (2010). Large scale max-margin multi-label classification with priors. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 423–430. Citeseer.

Haykin, S. (1994). *Neural networks: a comprehensive foundation*. Prentice Hall PTR.

John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 338–345. Morgan Kaufmann Publishers Inc.

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

Kononenko, I. (1994). Estimating attributes: analysis and extensions of relief. In *European conference on machine learning*, pages 171–182. Springer.

Kuhn, H. W. and Tucker, A. W. (2014). Nonlinear programming. In *Traces and emergence of nonlinear programming*, pages 247–258. Springer.

Li, Y. and Liu, J. S. (2018). Robust variable and interaction selection for logistic regression and general index models. *Journal of the American Statistical Association*, pages 1–16.

Liaw, A., Wiener, M., et al. (2002). Classification and regression by randomforest. *R news*, 2(3):18–22.

Ng, A. Y. (2004). Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM.

Robnik-Šikonja, M. and Kononenko, I. (2003). Theoretical and empirical analysis of relieff and rrelieff. *Machine learning*, 53(1-2):23–69.

Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, 5(2):197–227.

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., et al. (2002). Gene expression correlates of clinical prostate cancer behavior. *Cancer cell*, 1(2):203–209.

Soentpiet, R. et al. (1999). *Advances in kernel methods: support vector learning*. MIT press.

Sun, Y. and Li, J. (2006). Iterative relief for feature weighting. In *Proceedings of the 23rd international conference on Machine learning*, pages 913–920. ACM.

Sun, Y., Todorovic, S., and Goodison, S. (2010). Local-learning-based feature selection for high-dimensional data analysis. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1610–1626.

Sun, Y. and Wu, D. (2008). A relief based feature extraction algorithm. In *Proceedings of the 2008 SIAM International Conference on Data Mining*, pages 188–195. SIAM.

Team, R. C. et al. (2013). R: A language and environment for statistical computing.

Tian, E., Zhan, F., Walker, R., Rasmussen, E., Ma, Y., Barlogie, B., and Shaughnessy Jr, J. D. (2003). The role of the wnt-signaling antagonist dkk1 in the development of osteolytic lesions in multiple myeloma. *New England Journal of Medicine*, 349(26):2483–2494.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S., and Moore, J. H. (2018). Relief-based feature selection: introduction and review. *Journal of biomedical informatics*.

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., et al. (2002). Gene expression profiling predicts clinical outcome of breast cancer. *nature*, 415(6871):530.

Weinberger, K. Q. and Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244.

Xing, E. P., Jordan, M. I., Russell, S. J., and Ng, A. Y. (2003). Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 521–528.

Yang, M., Wang, F., and Yang, P. (2008). A novel feature selection algorithm based on hypothesis-margin. *JCP*, 3(12):27–34.

Zhu, J., Song, J., and Chen, B. (2016). Max-margin nonparametric latent feature models for link prediction. *arXiv preprint arXiv:1602.07428*.