HHS Public Access

Author manuscript

Proc SIAM Int Conf Data Min. Author manuscript; available in PMC 2017 February 01.

Published in final edited form as:

Proc SIAM Int Conf Data Min. 2016 May ; 2016: 567–575. doi:10.1137/1.9781611974348.64.

DPClass: An Effective but Concise Discriminative Patterns-Based Classification Framework

Jingbo Shang, **Wenzhu Tong**, **Jian Peng**, and **Jiawei Han**

Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Abstract

Pattern-based classification was originally proposed to improve the accuracy using selected frequent patterns, where many efforts were paid to prune a huge number of non-discriminative frequent patterns. On the other hand, tree-based models have shown strong abilities on many classification tasks since they can easily build high-order interactions between different features and also handle both numerical and categorical features as well as high dimensional features. By taking the advantage of both modeling methodologies, we propose a natural and effective way to resolve pattern-based classification by adopting discriminative patterns which are the prefix paths from root to nodes in tree-based models (e.g., random forest). Moreover, we further compress the number of discriminative patterns by selecting the most effective pattern combinations that fit into a generalized linear model. As a result, our discriminative pattern-based classification framework (DPClass) could perform as good as previous state-of-the-art algorithms, provide great interpretability by utilizing only very limited number of discriminative patterns, and predict new data extremely fast. More specifically, in our experiments, DPClass could gain even better accuracy by only using top-20 discriminative patterns. The framework so generated is very concise and highly explanatory to human experts.

1 Introduction

Various algorithms and models have been introduced for classification. Generalized linear classification models, such as support vector machine [25] and logistic regression [12], usually have reasonably good performance but lack the power of modeling complex high-order interactions between features. Tree-based models, such as random forest [1] and gradient boosted trees [11], have been deployed in many practical settings and often achieved high accuracy, because the high model complexity of trees provides the chance of high-order combinations of different features. Neural network is another kind of powerful classifiers, especially in image classification problems [14], which models nonlinear relationship among features and usually performs with high prediction accuracy. However, in real world applications, many people favor generalized linear models instead of complex models, including trees and neural networks, as long as the accuracies are enough in practice, because they are mature, flexible, more efficient when making prediction, and easier to be understood by providing probabilistic interpretation [12]. The low

*Corresponding authors: Jiawei Han and Jian Peng, hanj@illinois.edu, jianpeng@illinois.edu. shang7@illinois.edu, wtong8@illinois.edu

interpretability makes complex models not suitable for many applications, such as classification problems in medical applications and scientific domains, in which feature importance and contribution of feature combinations from the model could be highly useful for obtaining intuitive understanding of the application. The ultimate goal would be to construct accurate models that are also simple enough to interpret.

To address this challenge, one possible solution is to feed constructed high-order features to generalized linear models and enhance accuracy and interpretability. Along this direction, many previous pattern-based models have been established in the last decade and demonstrated powers in several domains, including (1) association rule-based classification on categorical data [22, 18, 30, 4, 29, 27]; (2) frequent pattern-based classification on text [19, 17] and graph [15, 6] data; (3) discriminative pattern-based classification on general data [2, 3], which mine discriminative patterns starting with frequent patterns and have shown their advantages over both tree-based models and generalized linear models. Many efforts are paid to prune a huge number of non-discriminative frequent patterns in those models, however, the number of extracted patterns utilized in later classification models is at the magnitude of thousands, which is still large.

In this paper, we propose a novel discriminative patterns-based classification framework (DPClass) with the goal to generate a very concise high-order classification model. The key component of DPClass is a fast and effective pattern extraction algorithm. Instead of starting with frequent patterns, we first train tree-based models to generate a large set of hypothetical high-order patterns, and then we explore all prefix paths from root nodes to leaf nodes in the tree-based models as our discriminative patterns. Finally, we further compress the number of discriminative patterns by selecting the most effective pattern combinations that fit into a generalized linear model with high classification accuracy. In this way, DPClass generates a set of discriminative high-order patterns with high predictivity and interpretability. From another perspective, we can view DPClass as a way to compress the multi-tree based models by only selecting the most discriminative pattern combinations and fitting them into a generalized linear model. Surprisingly, DPClass achieves comparable or even improved performance over the original tree-based models with only storing dozens of robust discriminative patterns. Such models can also be extremely useful for applications (e.g., mobile apps), where model storage and online computational cost are restricted.

In summary, our main contributions are as following.

- DPClass can learn a very small amount (e.g., 20) of interpretable patterns involving high-order interactions among original features, as verified in our synthetic experiment.
- DPClass can compress multi-tree based models into a low-dimensional generalized linear model, thus making the online prediction extremely fast.
- DPClass has comparable accuracy as the previous state-of-the-art algorithms and sometimes even better in our experiments on various real world datasets.

2 Related Work

Pattern-based classification is a well-studied problem traditionally from the perspective of frequent pattern mining. Association rules are the easiest way to make connections between frequent patterns and classification labels [30, 18]. A more effective solution is to first generate frequent patterns as a large pattern pool and then apply different heuristics, such as information gain, to select the most discriminative patterns based on labels, which will be further used in classical classifiers [2, 3, 8]. However, there are several problems in these methods. The first problem is that the number of frequent patterns in the pool are often very large, which leads to expensive computational cost during pattern selection. The second one is that the number of selected patterns can be as large as thousands for many applications, which limits the interpretability of the classification model and also causes the inefficiency in the classification steps. Another minor issue is that the discretization of continuous variables is a little bit tricky and thus makes the performance unstable.

One kind of the state-of-the-art classification models is tree-based models. Both decision tree and boosted tree models are explainable but quite sensitive to the training data. Traditional ensemble methods using multiple trees, such as random forest [1] and gradient boosting decision trees [10], greatly reduce the risk of overfitting and enhance the performance. As noticed by Ren et al. [24], because the growth and pruning in different trees are all independent, the global refinement could provide chances to get better performance. However, the increased model size of those multi-tree based models sacrifices the interpretability and thus our proposed method is significantly different from this category. Another popular usage of multi-tree based models is utilizing them to induce new feature spaces. The most common way to induce features from trees is to encode each tree as a flat index list and each instance to a binary vector indexed by the trees [24, 13, 7, 23, 20]. Vens et al. [28] further transfers the binary vectors into a inner product kernel space using a support vector machine and shows improved classification accuracy. Furthermore, pairwise interactions have also been introduced to improve the classification and regression [21], which is actually a two-layer trees model. Although some of these models have already applied post-pruning techniques for trees, the dimension of newly created feature space is still high due to a large number of trees constructed. For example, in the most recent work by Ren et al. [24], after many efforts on pruning, the model size of the pruned random forest is still around megabytes, which still keeps the prediction slow for real-time applications. In our experiments, the results show that our proposed framework DPClass could deliver comparable results using only top-20 discriminative patterns, which is substantially reduced even compared to the most efficient model in Ren et al. [24] that is specially designed for visualization tasks.

By utilizing high-order discriminative patterns in a generalized linear model, pattern selection can be performed by existing feature selection algorithms. Simply selecting patterns with highest independent heuristics such as information gain and gini index [16] is limited to very simple tasks, because most powerful patterns selected by such heuristics can introduce redundancy and potentially the overfitting issue. As classification labels are given, LASSO [26] is widely used in feature selection tasks as well as forward selection [5]. Due to the relatively large number of candidate discriminative patterns, backward selection, another

popular feature selection technique, might be not suitable in our settings. Therefore, we only adopt LASSO and forward selection into our framework DPClass to select discriminative patterns and compare their performances in our experiments.

3 Preliminaries

In this section, we will give formal definitions of key concepts in our problems and framework.

3.1 Problem Formulation

For a binary classification task, we assume that the training data is a set of n data points in a d-dimensional feature space together with their labels $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)$, where $\forall i (1 \le i \le n)$, $y_i \in \{+1, -1\}$, $\mathbf{x}_i \in \mathbb{R}^d$. It is worth noting that the values in the data point $\mathbf{x}_i$ can be either continuous (numerical) or discrete (categorical). As categorical features can be transformed into several binary dummy indicators, we assume $\mathbf{x}_i \in \mathbb{R}^d$ without loss of generality. In some previous pattern-based models, such as DDP-Mine [3], patterns are mined based on categorical values and thus they are only able to handle the continuous variables after carefully manual discretizations, which might be tricky and often requires prior knowledge about the data.

Our proposed framework, DPClass, will first generate a discriminative pattern pool within a reasonable size and then select top-k patterns based on their prediction accuracy or a cardinality regularization using a generalized linear classification model on training data. Since the number of selected patterns can be controlled, we hope these patterns provide us informative interpretability with reasonable predictive power. In addition, for new test data, we will only need to evaluate a very small set of patterns and make predictions with a generalized linear model efficiently.

3.2 Definition

First of all, we define the patterns in our classification problem. Traditional frequent patterns are only applied on the categorical data or itemset data. Discretization is needed for continuous variables. Instead of directly comparing the numerical values, we introduce the thresholding boolean function here.

Definition 1. **Condition** is a thresholding boolean function on a specific feature dimension. It is in the form of $(x_{\cdot,j} < v)$ *or* $(x_{\cdot,j} \ge v)$, where j indicates the specific dimension and v is the threshold value. The relational operator in a condition is either < or ≥.

Note that the threshold values in DPClass are not specified by users beforehand. In some previous pattern-based models, such as DDPMine [3], users have to choose a method to discretize values of continuous variables prior to pattern mining. DPClass makes all these values automatically determined based on the training data without any human interventions.

Example 1. *Suppose* $\mathbf{x}_i \in \mathbb{R}^{10}$, *one possible condition is that* $\mathbf{x}_{\cdot,1} < 0.5$. *Another example could be* $\mathbf{x}_{\cdot,2} \ge 0.5$.

Similar to traditional itemset, which are usually defined by a set of items, we define the pattern as a set of conditions. More formally, we use conjunctions to concatenate different conditions.

Definition 2. **Pattern** is a conjunction clause of conditions on specific feature dimensions. More formally, it is defined as following.

$$(x_{\cdot,j_1} < v_1) \wedge (x_{\cdot,j_2} \geq v_2) \wedge \ldots \wedge (x_{\cdot,j_m} \geq v_m)$$

where m is the number of conditions within this pattern. Note that different patterns can have different m values.

Example 2. *Suppose* $\mathbf{x}_i \in \mathbb{R}^{10}$, *one possible pattern is that* $(\mathbf{x}_{\cdot,1} < 18) \wedge (\mathbf{x}_{\cdot,3} \geq 100) \wedge (\mathbf{x}_{\cdot,9} < 0.5)$.

After we have defined patterns, we define discriminative patterns as following.

Definition 3. **Discriminative Patterns** refer to those patterns which have strong signals on the classification task given by the labels of data. For example, a pattern with very high information gain on the training data should be a discriminative pattern.

Example 3. *Suppose* $\mathbf{x}_i \in \mathbb{R}^{10}$ *and the labels are generated as following.*

$$y_i = [(\mathbf{x}_{i,1} \geq 1) \wedge (\mathbf{x}_{i,2} < 0)] \vee [(\mathbf{x}_{i,1} < 18) \wedge (\mathbf{x}_{i,3} \geq 100)]$$

Then, both patterns $(\mathbf{x}_{i,1} \geq 1) \wedge (\mathbf{x}_{i,2} < 0)$ *and* $(\mathbf{x}_{i,1} < 18) \wedge (\mathbf{x}_{i,3} \geq 100)$ should be among the most discriminative patterns. Of course, some similar patterns containing or having overlaps with these two patterns might also be discriminative patterns.

Discriminative patterns may have some overlapped predictive effects. Some discriminative patterns are special cases of others. For example, in the previous example, both patterns $(\mathbf{x}_{i,1} \geq 1) \wedge (\mathbf{x}_{i,2} < 0)$ and $(\mathbf{x}_{i,1} \geq 1) \wedge (\mathbf{x}_{i,2} < 0) \wedge (\mathbf{x}_{i,3} < 0)$ could indicate a positive label. However, the second pattern only encodes a subset of data points encoded by the first pattern and thus does not provide extra information for classification. This common phenomenon makes directly taking the top discriminative patterns based on some independent heuristics waste the budget of the number of patterns, if linear combination of these patterns are not synergistic. Therefore, we propose to select the top-*k* patterns by their predictive performance to make the selected patterns complementary and compact.

Definition 4. **Top-**k **Patterns** is a size-k subset of discriminative patterns, which have the best performance (i.e., the accuracy in classification tasks) based on the training data.

Here we have an assumption that the training and testing data have the same distribution, which is widely assumed in classification problems. In this case, the accuracy on training data is similar to the testing data if our model is not overfitted.

EXAMPLE 4. *In the last example, ideally, top-2 patterns should be* $\{(\mathbf{x}_{i,1} \geq 1) \wedge (\mathbf{x}_{i,2} < 0), (\mathbf{x}_{i,1} < 18) \wedge (\mathbf{x}_{i,3} \geq 100)\}$.

4 Methodology

In our proposed DPClass framework, as shown in Figure 1, a constrained multi-tree based model is trained on the training data. By adopting every prefix path from the root of a tree to any of its non-leaf nodes as a discriminative pattern, a large discriminative pattern pool is ready for further top-k discriminative patterns selection. We propose two different solutions to select top-k discriminative patterns based on the training data, both showing promising performance in our experiments.

4.1 Constrained Multiple Tree-based Model for Discriminative Patterns Generation

The first component in DPClass framework is the generation of high-quality discriminative patterns, as shown in Algorithm 1. We use **tree bag** to refer the set of instances falling into a specific node in the decision tree. The random decision tree [1] introduces the randomness via bootstrapping training data, randomly selecting features and splitting values when dividing a large tree bag into two smaller ones.

In real-world applications, discriminative patterns are usually somehow frequent, and the length of such patterns are not too long. More specifically, we assume that the number of instances satisfying a given discriminative pattern should be at least σ, and the length of discriminative patterns is no more than D. As one of the most famous multi-tree based models, random forest [1] is the best fit addressing both requirements if we treat every prefix path from the root of a tree to its non-leaf node as a discriminative pattern. First of all, distributions of labels of instances in a tree bag always have low entropy. Therefore, the patterns are discriminative on the training data. Secondly, it provides many putative patterns from various random decision trees trained on different bootstrapped datasets. Moreover, the depth threshold D and the minimum tree bag size σ can be naturally added as constraints during the growth of trees.

Algorithm 1: Discriminative Pattern Generation

Require: n training instances $(\mathbf{x}_i, y_i)$, the number of trees T, the depth threshold D, and minimum tree bag size σ
Return: A set of discriminative patterns for further selection.
$\mathcal{P} \leftarrow \emptyset$
for $t = 1$ ***to*** T **do**
  Build a random decision tree [1] with maximum depth D and minimum tree bag size σ.
  for *each non-leaf node* u **do**
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{root \rightarrow u\}$
return $\mathcal{P}$

4.2 Pattern Space

Instances in their original feature space are now mapped into the pattern space using a set of discriminative patterns, as shown in Algorithm 2. For each discriminative pattern, there is

one corresponding binary dimension describing whether the instances satisfy the pattern or not. Because the dimension of the pattern space is equal to the number of discriminative patterns which is a very large number after the generation phase, we need to further select a limited number of patterns and thus make the pattern space small and efficient. It is also worth a mention that this mapping process is able to be fully parallelized for further speedup.

Algorithm 2: Construct Pattern Space

Require: n instances $(\mathbf{x}_i)$, a discriminative patterns set $\mathcal{P}$
Return: n instances in pattern space $(\mathbf{x'}_i)$
for $i = 1$ ***to*** n **do**
  $\mathbf{x'}_i \leftarrow \mathbf{0}$
  for *j-th pattern P_j in $\mathcal{P}$* **do**
    if $\mathbf{x}_i$ *satisfies pattern* P_j **then**
      $\mathbf{x'}_{i,j} \leftarrow 1$
return $(\mathbf{x'}_i)$

Algorithm 3: Top-k Pattern Selection: Forward

Require: n training instances $(\mathbf{x}_i, y_i)$, a set of discriminative patterns $\mathcal{P}$ and k
Return: Top-k discriminative patterns set $\mathcal{P}_k$ and a generalized linear model $f(\cdot)$
$\mathcal{P}_k \leftarrow \emptyset$
for $t = 1$ ***to*** k **do**
  for *each pattern p in $\mathcal{P}$* **do**
    $\mathbf{x'} \leftarrow$ construct pattern space$(\mathbf{x}, \mathcal{P}_k \cup \{p\})$
    $g(\cdot) \leftarrow$ a generalized linear model [25] on $(\mathbf{x'}_i, y_i)$
    $acc_p \leftarrow g(\cdot)$'s training accuracy
  $\mathcal{P}_k \leftarrow \mathcal{P}_k \cup \{\arg\max_p acc_p\}$
$\mathbf{x'} \leftarrow$ construct pattern space$(\mathbf{x}, \mathcal{P}_k)$
$f(\cdot) \leftarrow$ a generalized linear model on $(\mathbf{x'}_i, y_i)$
return $\mathcal{P}_k$, $f(\cdot)$

4.3 Top-k Patterns Selection

After a large pool of discriminative patterns is generated, further top-k selection needs to be done to identify the most informative and interpretable patterns. A naive way is to use heuristic functions, such as information gain and gini index, to evaluate the significance of different patterns on the classification task and choose the top ranked patterns. However, the effects of top ranked patterns based on the simple heuristic scores may have a large portion of overlaps and thus their combination does not work optimally. Therefore, to achieve the best performance and find complementary patterns, we propose two effective solutions: forward selection and LASSO, which make decisions based on the effects of the pattern combinations instead of considering different patterns independently.

4.3.1 Forward Pattern Selection

Instead of exhausted search of all possible combinations of k discriminative patterns, we gradually add the discriminative patterns one by one while each newly added discriminative pattern is the best choice at that time [5], which provides an efficient approximation of the

exhausted search. Since we use a generalized linear classification model to combine the selected discriminative patterns, when we have fixed the first k' discriminative patterns, we empirically add one more discriminative pattern with the biggest improvement of the training classification accuracy based on $k'+1$ discriminative patterns, as shown in Algorithm 3. As we have mentioned before, when assuming training and testing data have the same distribution, using training accuracy is very reasonable.

Algorithm 4: Top-k Pattern Selection: LASSO

Require: n training instances $(\mathbf{x}_i, y_i)$, a set of discriminative patterns $\mathcal{P}$, k, and a small value ϵ
Return: Top-k discriminative patterns P_i and a generalized linear model $f(\cdot)$

```
P_k ← ∅
l ← 0, r ← +∞
x' ← construct pattern space(x, P)
while l + ε < r do
    λ ← (l + r)/2
    w ← arg min_w Equation 4.1
    if non-zero weighted patterns ≤ k then
        P_k ← {p|p's weight is non-zero}
        r ← λ
    else
        l ← λ
x' ← construct pattern space(x, P_k)
f(·) ← a generalized linear model on (x'_i, y_i)
return P_k, f(·)
```

4.3.2 LASSO based Pattern Selection

L1 regularization (i.e., LASSO [26]) is designed to make the weight vector sparse by tuning a nonnegative parameter λ. Since we are actually selecting features in the pattern space, for a given λ, we optimize the following loss function to get a subset of patterns which are most important.

$$\mathscr{L} = \sum_{i}^{n} l\left(\mathbf{x}'^{T}_{i}\mathbf{w}, y_i\right) + \lambda \cdot \| w \|_1 \tag{4.1}$$

where, $\mathbf{x}'_i$ is the mapped binary feature representation in pattern space of i-th instance; $\mathbf{w}$ is the weight vector in the generalized linear model; $l(\cdot, \cdot)$ is a general loss function such as logistic loss. To ensure there are at most k patterns having non-zero weights in the pattern space, we should carefully choose a value for λ. It is important to assume that there exists some hidden importance ranking among features and once the weight of a feature becomes non-zero in a given $\lambda = v$, it will also be nonzero for almost any smaller $\lambda < v$. Therefore, we propose a binary search algorithm as shown in Algorithm 4. In this paper, we adopt the LASSO implementation in GLMNET [9] for binary classification, whose loss function is the cross entropy.

Algorithm 5: Prediction

Require: n testing instances $(\mathbf{x}_i)$, top-k discriminative patterns set $\mathcal{P}_k$, and the generalized linear model $f(\cdot)$
Return: predictions of testing instances $\hat{y}_i$
$\mathbf{x}' \leftarrow$ construct pattern space$(\mathbf{x}, \mathcal{P}_k)$
for $i = 1$ ***to*** n **do**
  $\hat{y}_i \leftarrow f(\mathbf{x}'_i)$
return $\hat{y}$

4.4 Prediction

Once the top-k discriminative patterns are determined, for any upcoming new test instance, we first map it into the learned pattern space and apply the pre-trained generalized linear model, as shown in Algorithm 5. As the number of patterns is limited, both the mapping into the pattern space and the prediction of the generalized linear model will be extremely fast.

4.5 Time Complexity Analysis

To build up a single random decision tree with depth threshold D and minimum tree bag size σ, by assuming both numbers of random features and random partitions are small and fixed constants, the time complexity is $O(nD)$, because the total number of instances on each level of the tree is n. Therefore, to generate T trees in total, the time complexity is $O(TnD)$ in the generation step.

For the selection step, the complexity is mainly determined by the number of discriminative patterns induced by T random decision trees, which is dependent on the total number of non-leaf nodes. As the maximum depth of a single tree is D, there is an upper bound on number of leaf nodes 2^D. Starting from the tree bag size, the number of leaf nodes should be no more than $\lceil \frac{n}{\sigma} \rceil$ Since the trees here are all binary trees, the number of leaf nodes is one more than the number of non-leaf nodes. Therefore, the number of discriminative patterns $|P|$ (i.e., the number of non-leaf nodes) is bounded by $T \cdot \min\left\{2^D, \lceil \frac{n}{\sigma} \rceil\right\} - 1$. If we solve logistic regression and LASSO using (sub-)gradient descent algorithm, and thus the time complexity per gradient step is only linear to the dimension of features and the number of instances. The time complexity is proportional to $O(|P| \cdot n \cdot k^2)$ if forward selection is used, while it is proportional to $O(n \cdot k \cdot |P|)$, if LASSO is used. By assuming the numbers of iterations to converge are similar in LASSO and forward selection, LASSO will be a little more efficient than forward selection.

When predicting new test instances, one can easily figure out the bottleneck is mapping instances into the learned pattern space. Therefore, in the batch mode where instances are considered together, the time complexity is $O(n \cdot k \cdot D)$. In the streaming (or online) mode where instances come one by one, the time complexity is $O(k \cdot D)$, where k is the number of discriminative patterns and D is the maximum number of conditions in a single pattern.

It is worth mentioning that all modules can be fully parallelized, leading to further speedup in practice.

5 Experiments

5.1 Datasets, Baselines, and Settings

To demonstrate the interpretability of DPClass, we generate a synthetic dataset where the features are demographics and lab test results of patients and the label is whether the patient does have a disease. Assuming doctors can diagnose the disease using some rules based on these information, we will check whether the top discriminative patterns selected by DPClass are consistent with the actual diagnosing rules.

DDPMine [3] is the previous state-of-the-art discriminative pattern based algorithm. It first discretizes the continuous variables such that frequent pattern mining algorithm could be applied. Using frequent and discriminative patterns, new feature space is constructed and any classical classifiers could be further utilized. Random Forest (**RF**) [1] is another baseline method using same parameters as those in the random forest used in DPClass, except for D. There is no limit on the depth in RF.

We have selected several binary classification datasets from UCI Machine Learning Repository as shown in Table 1 with statistics of the number of instances and the number of features. To compare with DDPMine, we use the same datasets in the DDPMine paper, including adult, hypo, sick, crx, sonar, chess, waveform, and mushroom. Because both DDPMine and DPClass achieve almost perfect accuracy (very close to 100%) on the datasets waveform and mushroom, we omit them in this paper. In addition, we are very interested in the performance of DPClass on high-dimensional datasets (nomao, musk, and madelon datasets), since DDPMine performs poorly at high-dimensional data. In datasets crx, sonar, chess, nomao, and musk, we cannot find the existing train/test splitting. Therefore, we ourselves divide the data into train/test (2:1) by unbiased sampling.

We also would like to see the impact of different parameters of DPClass, such as k, the number of selected discriminative patterns selected, and T, the number of trees in the random forest.

In DPClass, the default setting is $T = 100$, $D = 6$, $\sigma = 10$, $K = 20$. We will show both results using forward selection (**DPClass-F**) and LASSO (**DPClass-L**) for selecting the top-k discriminative patterns.

5.2 Discovery of Interpretable Patterns

We generate a small medical dataset to demonstrate the interpretability of DPClass. For each patient, we draw several uniformly sampled features as the following rules.

- Age (A). Positive Integers no more than 60.
- Gender (G). Male or Female.
- Lab Test 1 (LT1). Blood Types. Categorical values from {A, B, O, AB}.
- Lab Test 2 (LT2). Continuous values in [0,1].

In total, we have 10^5 random patients for training and $5 \cdot 10^4$ patients for testing.

The positive label of the disease is assigned to a patient if at least one of the following rules holds.

- (age > 18) and (gender = Male) and (LT1 = AB) and (LT2 ≥ 0.6)
- (age > 18) and (gender = Female) and (LT1 = O) and (LT2 ≥ 0.5)
- (age ≤ 18) and (LT2 ≥ 0.9)

To make the classification tasks more challenging, we add 0.1% noise to the training data. That is, 0.1% labels in training will be flipped.

We apply both DPClass-F and DPClass-L on this dataset. Both give the test accuracy 99.99%. The top-3 discriminative patterns found in both DPClass-F and DPClass-L are listed as below. We observe that the found patterns are quite close to the groundtruth rules and thus demonstrate that our selected discriminative patterns can provide high-quality explanation.

- (age > 18) and (gender = Female) and (LT1 = O) and (LT2 ≥ 0.496)
- (age ≤ 18) and (LT2 ≥ 0.900)
- (age > 18) and (gender = Male) and (LT1 = AB) and (LT2 ≥ 0.601)

We also apply DDPMine to this dataset but its accuracy is only 95.64%, because the discretization is difficult to be perfect. The top-3 patterns mined by DDPMine are as following, which are quite different from the expected.

- (LT2 > 0.8)
- (gender = Male) and (LT1 = AB) and (LT2 ≥ 0.6) and (LT2 < 0.8)
- (gender = Female) and (LT1 = O) and (LT2 ≥ 0.6) and (LT2 < 0.8)

5.3 Compare to DDPMine

DDPMine is the previous state-of-the-art pattern-based classification method, which outperforms classical classification models including decision tree and support vector machine [2, 3]. By testing on the same datasets in DDPMine paper, as shown in the first 6 columns of Table 2, DPClass-F and DPClass-L always have higher accuracy over DDPMine. Except for sick dataset, DPClass-F has the highest accuracy, while DPClass-L works best on sick dataset. It seems that DPClass-F works a little better than DPClass-L. However, their results are quite close to each other and are both better than DDPMine's on most datasets.

5.4 Impact of Top-k patterns

The most interesting parameter in DPClass is k, the number of discriminative patterns used in the final generalized linear model. It controls the model size of the generalized linear model used for prediction and thus affects its efficiency. Because the default value of k is 20 and its effectiveness has been proved in previous experiments, we vary k from 1 to 40 to see the trends of both training and testing accuracies on different datasets. In this experiment, we only use those three datasets with existing training and testing partition.

As shown in Figure 2, the test accuracy is always following the trend of training accuracy and the training accuracy is increasing as k grows. It demonstrates that our pattern selection based on training accuracy is reasonable. In real world applications, k could be determined by cross validations.

Although the accuracies are growing almost all the time, it slows down much when k is greater than 20. Therefore, we conclude that a very small k (e.g., $k = 20$) is enough for these comprehensive real-world datasets, which further proves that our proposed DPClass can compress the model into a very tiny size while its accuracy remains comparable.

5.5 Impact of number of trees

Another important parameter in DPClass is the number of trees needed to generate the large pool of discriminative patterns. As we argued before, a single tree is not enough to generate that many patterns, and thus we have a strong motivation to try $T = 1$ as an extreme case. The default value 100 works well in previous experiments, and thus we vary T in {1, 10, 50, 100, 500, 1,000} to see the trends of both training and testing accuracies. In this experiment, we only use those three datasets with existing training and testing partition.

As shown in Figure 3, when $T = 1$, the accuracy is much lower than others, which means only a single decision tree is not enough for a diverse patterns pool. According to the curves, one can easily observe and conclude that a reasonable large T, such as 100, is enough to achieve a reasonable result.

5.6 High Dimensional Data

We are also interested in high-dimensional datasets (i.e., at least 100 dimensions) because DDPMine is not effective in large dimensional data. As the dimension of the original feature space grows, we have to increase the depth threshold D, as well as the number of trees T, to involve higher order interactions and increase the number of candidate discriminative patterns. Therefore, we set $D = 10$ and $T = 200$. Meanwhile, the dimension of mapped pattern space may also need to be increased due to the higher complexity of problems. As a result, we set $k = 50$ in nomao and musk. However, we kept $k = 20$ in madelon because many features are noises.

As shown in last 3 columns of Table 2, DPClass can always outperform DDPMine and generate comparable results to those by the random forest model. More importantly, in madelon dataset, DPClass-F and DPClass-L outperform random forest significantly and thus demonstrate the robustness especially when the features are high dimensional and noisy. It is also worth a mention that the training process of DPClass is at least 10 times faster than DDPMine in high dimensional datasets.

5.7 Scalability

The test running time is linearly proportional to the model complexity, which is related to the number of patterns the model used. In our experiments, DDPMine needs 100 to 1,000 patterns while DPClass only needs 20, which indicates a significant reduction of prediction runtime. Moreover, the random forest without any constraints will contain more than 10,000

nodes (i.e., patterns), which is far more expensive. Although the evaluation of random forest for a single testing instance will traverse only a number of nodes equals to the sum of depths in different trees, it always needs more than 1,000 traverses in our experiments. Therefore, DPClass is the most efficient model for testing new instances, compared to DDPMine and random forest, by achieving about **20 to 50 times speedup** in practice. Furthermore, DPClass could be fully parallelized for further speedup.

6 Conclusion

In this paper, we propose an effective and concise discriminative pattern-based classification framework (*DP-Class*) to address the general classification problem and provide interpretability by incorporating a limited number of discriminative patterns. DPClass first extracts the prefix paths from root nodes to non-leaf nodes in tree-based models as candidate discriminative patterns and then further compress the number of discriminative patterns by selecting the most effective pattern combinations according to their predictive accuracy in a generalized linear model. Comprehensive experiments have demonstrated that DPClass is able to model high-order interactions and present a small amount of interpretable patterns to help human experts understanding the classification tasks. Moreover, it provides comparable or even better accuracy than the previous state-of-the-art pattern-based classification model DDPMine and the uncompressed random forest model.

In future, we plan to extend our DPClass to a uniform machine learning framework DPLearn, which supports multi-classes classification, regression, and ranking along the same discriminative pattern selection direction. Another possible direction is to apply DPClass to labeled textual and sequential data targeting on finding interesting patterns (e.g., language patterns).

Acknowledgements

Research was sponsored in part by the U.S. Army Research Lab. under Cooperative Agreement No. W911NF-09-2-0053 (NSCTA), National Science Foundation IIS-1017362, IIS-1320617, and IIS-1354329, HDTRA1-10-1-0120, and grant 1U54GM114838 awarded by NIGMS through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative (www.bd2k.nih.gov), and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC.

References

[1]. Chen, C., Liaw, A., Breiman, L. Using random forest to learn imbalanced data. University of California, Berkeley: 2004.

[2]. Cheng, H., Yan, X., Han, J., Hsu, C-W. ICDE. IEEE; 2007. Discriminative frequent pattern analysis for effective classification; p. 716-725.

[3]. Cheng, H., Yan, X., Han, J., Yu, PS. ICDE. IEEE; 2008. Direct discriminative pattern mining for effective classification; p. 169-178.

[4]. Cong, G., Tan, K-L., Tung, AK., Xu, X. SIGMOD. ACM; 2005. Mining top-k covering rule groups for gene expression data; p. 670-681.

[5]. Derksen S, Keselman H. Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. British Journal of Mathematical and Statistical Psychology. 1992; 45(2):265–282.

[6]. Deshpande M, Kuramochi M, Wale N, Karypis G. Frequent substructure-based approaches for classifying chemical compounds. TKDE. 2005; 17(8):1036–1050.

[7]. Ebina T, Toh H, Kuroda Y. Drop: an svm domain linker predictor trained with optimal features selected by random forest. Bioinformatics. 2011; 27(4):487–494. [PubMed: 21169376]

[8]. Fan, W., Zhang, K., Cheng, H., Gao, J., Yan, X., Han, J., Yu, P., Verscheure, O. SIGKDD. ACM; 2008. Direct mining of discriminative and essential frequent patterns via model-based search tree; p. 230-238.

[9]. Friedman J, Hastie T, Tibshirani R. glmnet: Lasso and elastic-net regularized generalized linear models. R package version. 2009; 1

[10]. Friedman JH. Greedy function approximation: a gradient boosting machine. Annals of statistics. 2001:1189–1232.

[11]. Ganjisaffar, Y., Caruana, R., Lopes, CV. SIGIR. ACM; 2011. Bagging gradient-boosted trees for high precision, low variance ranking models; p. 85-94.

[12]. Hosmer, DW., Jr, Lemeshow, S. Applied logistic regression. John Wiley & Sons; 2004.

[13]. Kobetski M, Sullivan J. Discriminative tree-based feature mapping. Intelligence. 2011; 34(3)

[14]. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems. 2012:1097–1105.

[15]. Kudo T, Maeda E, Matsumoto Y. An application of boosting to graph classification. Advances in neural information processing systems. 2004:729–736.

[16]. Lee C, Lee GG. Information gain and divergence-based feature selection for machine learning-based text categorization. Information processing & management. 2006; 42(1):155–165.

[17]. Leslie, CS., Eskin, E., Noble, WS. Pacific symposium on biocomputing. Vol. 7. World Scientific; 2002. The spectrum kernel: A string kernel for svm protein classification; p. 566-575.

[18]. Li, W., Han, J., Pei, J. ICDM. IEEE; 2001. Cmar: Accurate and efficient classification based on multiple class-association rules; p. 369-376.

[19]. Lodhi H, Saunders C, Shawe-Taylor J, Cristianini N, Watkins C. Text classification using string kernels. JMLR. 2002; 2:419–444.

[20]. Lou Y, Caruana R, Gehrke J. Intelligible models for classification and regression. SIGKDD. 2012

[21]. Lou Y, Caruana R, Gehrke J, Hooker G. Accurate intelligible models with pairwise interactions. SIGKDD. 2013

[22]. B. L. W. H. Y. Ma. Integrating classification and association rule mining. SIGKDD. 1998

[23]. Moosmann, F., Triggs, B., Jurie, F. NIPS. MIT Press; 2007. Fast discriminative visual codebooks using randomized clustering forests; p. 985-992.

[24]. Ren S, Cao X, Wei Y, Sun J. Global refinement of random forest. CVPR. 2015:723–730.

[25]. Suykens JA, Vandewalle J. Least squares support vector machine classifiers. Neural processing letters. 1999; 9(3):293–300.

[26]. Tibshirani R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological). 1996:267–288.

[27]. Veloso, A., Meira, W., Zaki, MJ. ICDM. IEEE; 2006. Lazy associative classification; p. 645-654.

[28]. Vens, C., Costa, F. ICDM. IEEE; 2011. Random forest based feature induction; p. 744-753.

[29]. Wang, J., Karypis, G. SDM. Vol. 5. SIAM; 2005. Harmony: Efficiently mining the best rules for classification; p. 205-216.

[30]. Yin, X., Han, J. SDM. Vol. 3. SIAM; 2003. Cpar: Classification based on predictive association rules; p. 369-376.

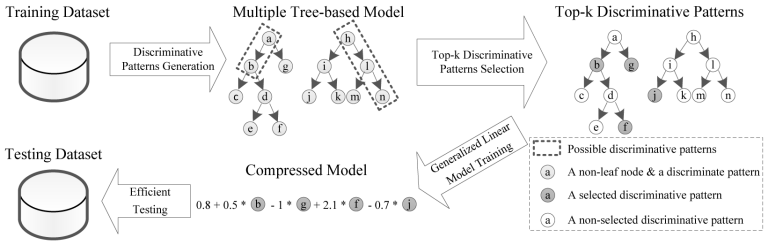

Figure 1.
Overview of DPClass Framework

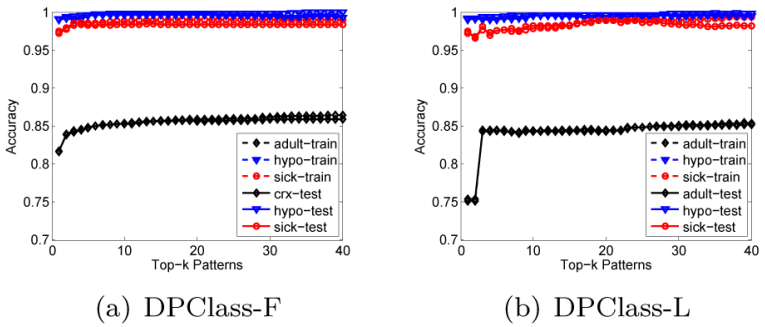

Figure 2.
The impact of top-k patterns. Training and testing accuracies are almost overlapped.

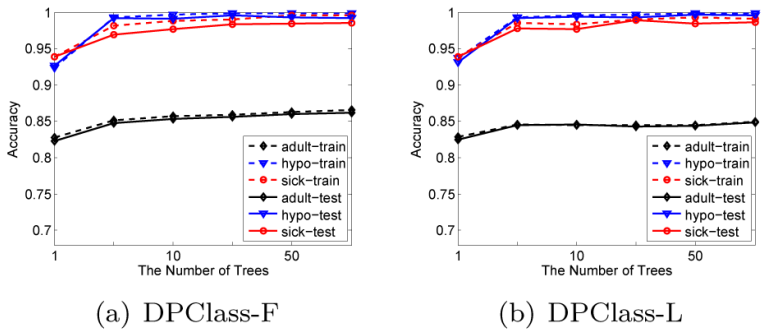

Figure 3.
The impact of the number of trees. Training and testing accuracies are almost overlapped.

Table 1

Datasets statistics.

Dataset	# instances	dimensions	variable types
adult	45,222	14	mixed
hypo	37,72	19	mixed
sick	37,72	19	mixed
chess	28,056	6	mixed
crx	690	15	mixed
sonar	208	60	numeric
nomao	29,104	120	mixed
musk	7,074	166	numeric
madelon	1300	500	numeric

Table 2

Test Accuracy on UCI Machine Learning Datasets tested in DDPMine. DDPMine outperforms decision tree and support vector machine on all these datasets [2, 3]. RF refers the random forest without any constraints.

Dataset	adult	hypo	sick	crx	sonar	chess	namao	musk	madelon
DPClass-F	**85.66%**	**99.58%**	98.35%	**89.35%**	**85.29%**	92.25%	97.17%	95.92%	74.50%
DPClass-L	84.33%	99.28%	**98.87%**	87.96%	83.82%	92.05%	96.94%	95.71%	**76.00%**
RF	85.45%	97.22%	94.03%	**89.35%**	83.82%	**94.22%**	**97.86%**	**96.60%**	56.50%
DDPMine	83.42%	92.69%	93.82%	87.96%	73.53%	90.04%	96.83%	93.29%	59.83%