Co-Selection of Features and Instances for Unsupervised Rare Category Analysis
Jingrui He∗ Jaime Carbonell∗

Abstract
Rare category analysis is of key importance both in theory and in practice. Previous research work focuses on supervised rare category analysis, such as rare category detection and rare category classification. In this paper, for the first time, we address the challenge of unsupervised rare category analysis, including feature selection and rare category selection. We propose to jointly deal with the two correlated tasks so that they can benefit from each other. To this end, we design an optimization framework which is able to co-select the relevant features and the examples from the rare category (a.k.a. the minority class). It is well justified theoretically. Furthermore, we develop the Partial Augmented Lagrangian Method (PALM) to solve the optimization problem. Experimental results on both synthetic and real data sets show the effectiveness of the proposed method.

1 Introduction
Rare category analysis refers to the problem of detecting and characterizing the minority classes in an unlabeled data set. It is of key importance both in theory and in practice. Take financial fraud detection as an example. Most financial transactions are legitimate, which constitute the majority class; whereas the fraudulent transactions of the same type are similar to each other, and they correspond to one minority class. For example, small-amount-probing type of fraudulent transactions are similar to each other in terms of the small amount stolen in each transaction and the recurring nature. They may not share the same location, time, etc. Therefore, these features can be seen as being irrelevant to this class.

The key observation is that the above two tasks are correlated with each other. On one hand, the analysis of the minority class examples helps us identify the relevant features; on the other hand, the identification of the relevant features is crucial to the selection of the minority class examples. Therefore, we propose to jointly deal with the two tasks so that they can benefit from each other. To this end, we formulate the problem as a well justified optimization framework, which co-selects the relevant features and the examples from the minority class. Furthermore, we design an effective search procedure based on augmented Lagrangian method. The basic idea is to alternatively find the relevant features and the minority class examples. Finally, we demonstrate the performance of the proposed method by extensive experimental results.

The main contributions of this paper can be summarized as follows.

Problem Definition. To the best of our knowledge, we are the first to address the two important tasks in unsupervised rare category analysis; and we propose to jointly deal with them;

Problem Formulation. We design an optimization framework for the co-selection of features and instances, which is well justified theoretically;

Search Procedure. We develop an effective algorithm to solve the optimization problem which is based on augmented Lagrangian.

The rest of the paper is organized as follows: in Section 2, we review related work; then in Section 3, we present the optimization framework with theoretical justification; Section 4 introduces the algorithm for solving the optimization problem; experimental results are given in Section 5; following some discussion in Section 6, finally, we conclude in Section 7.
2 Related Work

In this section, we review related work on supervised rare category analysis, outlier detection and unsupervised feature selection. Supervised rare category analysis can be further divided into two major groups, rare category detection and rare category classification.

Rare Category Detection. Here, the goal is to find at least one example from each minority class with the help of a labeling oracle, minimizing the number of label requests. Assuming the relevance of all the features, researchers have developed several methods for rare category detection. For example, in [25], the authors assumed a mixture model to fit the data, and experimented with different hint selection methods, of which Interleaving performs the best; in [12], the authors studied functions with multiple output values, and used active sampling to identify an example for each of the possible output values; in [13], the authors developed a new method for detecting an instance of each minority class via an unsupervised local-density-differential sampling strategy; and in [8], the authors presented an active learning scheme that exploits cluster structure in the data, which was proven to be effective in rare category detection. Different from these methods, in our paper, no labeling oracle is available for querying, and the goal is to select a set of examples which are likely to come from the minority class. Furthermore, we assume only some of the features are relevant to the minority classes, and hope to identify those features.

Rare Category Classification (Imbalanced Classification). Here, the goal is to construct an accurate classifier for the minority classes given labeled examples from all the classes. Existing methods can be roughly categorized into 3 groups [5], i.e. sampling based methods [21][19][6], adapting learning algorithms by modifying objective functions or changing decision thresholds [28][16], and ensemble based methods [27][7]. Furthermore, some researchers have worked on feature selection for imbalanced data to improve the performance of the classifier, such as in [30]. The major difference between these methods and our method is that we work in an unsupervised fashion, i.e. no labeled data is available.

Outlier Detection. Outlier detection refers to the problem of finding patterns in data that do not conform to expected behavior [4]. According to [4], the majority of outlier detection techniques can be categorized into classification based [3], nearest neighbor based [26], clustering based [29], information theoretic [15], spectral [10], and statistical techniques [1]. Compared with our method, outlier detection finds individual and isolated instances that differ from a given class and from each other. Typically these are in low-density regions. This is a very different process than discovering a new compact class, where we are looking for a local density spike and the minority class instances are strongly self-similar. Furthermore, in outlier detection, the discovery of one outlier can not help discover other outliers since the outliers are different from each other; whereas in rare category analysis, the discovery of a minority class instance can help us understand the properties of this class, and thus help us discover instances from the same class. For example, in rare disease diagnosis, by studying a patient with a particular kind of rare disease, the doctors may gain more insights about this disease, and design better techniques for diagnosing and treating such disease.

Unsupervised Feature Selection. Generally speaking, existing methods can be categorized as wrapper models and filter models. The wrapper models evaluate feature subsets based on the clustering results, such as the FSSEM algorithm [11], the mixture-based approach which extends to the unsupervised context the mutual-information based criterion [20], and the ELSA algorithm [17]. The filter models are independent of the clustering algorithm, such as the feature selection algorithm based on maximum information compression index [23], the feature selection method using distance-based entropy [9], and the feature selection method based on Laplacian score [14]. Similar to unsupervised feature selection, in our paper, we also assume that the class labels are unknown. However, in our settings, the class proportions are extremely skewed, and we are only interested in the features relevant to the minority classes. In this case, both wrapper and filter methods select the features primarily relevant to the majority classes. Therefore, we need new methods that are tailored for our problem.

3 Optimization Framework

In this paper, we focus on the binary case, i.e. one majority class and one minority class, and our goal is to (1) select a set of examples which are likely to come from the minority class, and (2) identify the features relevant to this minority class. In this section, we formulate this problem as an optimization framework, and provide some theoretical justification.

3.1 Notation Let $D = \{x_1, \ldots, x_n\}$, $x_i \in \mathbb{R}^d$ denote a set of $n$ unlabeled examples, which come from 2 classes, i.e. the class labels $y_i \in \{1, 2\}$, $i = 1, \ldots, n$. $y_i = 1$ corresponds to the majority class with prior $1 - p$, and $y_i = 2$ corresponds to the minority class with prior $p$. $p \ll 1$. Furthermore, of the $d$ features, only $d_f$ features are relevant to the minority class. In other words, the examples from the minority class have very similar values on those features, and their values on the other features may be quite diverse. For the sake of simplicity, assume that the $d_f$ features are independent to each other. Therefore, the examples from the minority class are tightly clustered in the $d_f$-dimensional subspace spanned by the relevant features, which we call the relevant subspace.
Let $S_{d_r}$ denote the set of all $d_r$-dimensional subspaces of $\mathbb{R}^d$, and let $S_{\text{min}}$ denote the relevant subspace, $S_{\text{min}} \in S_{d_r}$. Let $f(x)$ denote the probability density function (pdf) of the data in $\mathbb{R}^d$, i.e. $f(x) = (1 - p)f_{\text{maj}}(x) + pf_{\text{min}}(x)$, where $f_{\text{maj}}(x)$ and $f_{\text{min}}(x)$ are the pdfs of the majority and minority classes in $\mathbb{R}^d$ respectively. Given feature subspace $S \in S_{d_r}$ and $x \in \mathbb{R}^d$, let $x(S)$ denote the projection of $x$ on $S$, and $f(S)(x(S))$, $f_{\text{maj}}(x(S))$ and $f_{\text{min}}(x(S))$ denote the projection of $f(x)$, $f_{\text{maj}}(x)$ and $f_{\text{min}}(x)$ on $S$ respectively.

To co-select the minority class examples and the relevant features, we define two vectors: $a \in \mathbb{R}^n$ and $b \in \mathbb{R}^d$. Let $a_i$ and $b_j$ denote the $i$th and $j$th elements of $a$ and $b$ respectively. $a_i = 1$ if $x_i$ is from the minority class, and 0 otherwise; $b_j = 1$ if the $j$th feature is relevant to the minority class, and 0 otherwise.

3.2 Objective Function Given the prior $p$ of the minority class and the number of relevant features $d_r$, we hope to find $np$ data points whose corresponding $a_i = 1$, and $d_r$ features whose corresponding $b_j = 1$. Intuitively, the $np$ points should form a compact cluster in the relevant subspace, and due to the characteristic of the minority class, this cluster should be more compact than any other $np$ data points in any $d_r$-dimensional subspace. To be more strict, we have the following optimization problem.

**Problem 1**

$$
\min_{a, b} F(a, b) = \frac{1}{np} \sum_{i=1}^{n} \sum_{k=1}^{n} a_i a_k \left( \sum_{j=1}^{d} b_j (x_{ij} - x_{kj})^2 \right)
$$

s.t. $\sum_{i=1}^{n} a_i = np$, $a_i = 0, 1$

$$\sum_{j=1}^{d} b_j = d_r$$

In the objective function $F(a, b)$, $\sum_{j=1}^{d} b_j (x_{ij} - x_{kj})^2$ is the squared distance between $x_i$ and $x_k$ in the subspace $S_b$ spanned by the features with non-zero $b_j$. This squared distance contributes to $F(a, b)$ if and only if both $a_i$ and $a_k$ are equal to 1. Given a set of $np$ points, define the set distance of every data point to be the sum of the squared distances between this point and all the points within this set. Therefore, by solving this optimization problem, we aim to find a set of $np$ points and $d_r$ features such that the average set distance of these points to this set in the corresponding subspace $S_b$ is the minimum.

Problem 1 can be easily applied to the case where either $a$ or $b$ is known, and we want to solve for the other vector. To be specific, if $a$ is known, i.e. we know the examples that belong to the minority class, and we want to find the $d_r$-dimensional subspace where the minority class can be best characterized, we can use the same objective function $F(a, b)$, and solve for $b$ using the minority class examples. Similarly, if $b$ is known, i.e. we know which features are relevant to the minority class, and we want to find the examples from the minority class, we can also use $F(a, b)$, and solve for $a$ in the subspace $S_b$ spanned by the relevant features.

3.3 Justification The optimization problem we introduced in the last subsection is reasonable intuitively. Next, we look at it from a theoretical perspective.

For any $d_r$-dimensional subspace, $\forall S \in S_{d_r}$, and for any data point, $x \in \mathbb{R}^d$, let function $ψ^S(x(S))$ denote the average squared distance between $x(S)$ and its $np$ nearest neighbors. More precisely, $ψ^S(x(S)) = \min_{\rho_{np} \in D_r} \{ \frac{1}{np} \sum_{y \in D_{np}} \| x(S) - y(S) \|^2 \}$, where $y(S)$ denotes the $i$th nearest neighbor of $x(S)$ within $x_1(S), \ldots, x_n(S)$. Furthermore, define function $φ^S$ to be the expected average squared distance between $x(S)$ and its $np$ nearest neighbors. More precisely, $φ^S(x(S)) = E(ψ^S(x(S)))$. Here, the expectation is with respect to $x(S), i = 1, \ldots, np$.

Based on the above definitions, we have the following theorem.

**Theorem 3.1.**

1. In $S_{\text{min}}$, the support region of the minority class is within hyper-ball $B$ of radius $r$;

2. The support region of $f$ in any $d_r$-dimensional subspace is bounded, i.e. $\max_{S \in S_{d_r}} \max_{x, y \in \mathbb{R}^d} f(S)(x(S)) > 0, f(S)(y(S)) > 0, \| x(S) - y(S) \|^2 = \alpha < +\infty$;

3. The density of the majority class in hyper-ball $B$ is non-zero, i.e. $\min_{y \in \mathbb{R}^d, y(S_{\text{min}}) \in B} (1 - p) f_{\text{maj}}(y(S_{\text{min}})) = f_0 > 0$;

4. For a certain data point, if its projection in the $d_r$-dimensional subspace $S$ is not within hyper-ball $B$, then the function value of $φ^S$ evaluated at this point is big enough, i.e. $\min_{S \in S_{d_r}, x \in \mathbb{R}^d, x(S) \notin B} φ^S(x(S)) - 4r^2 > \beta > 0$;

5. The number of examples is sufficiently large, i.e. $n \geq \max \left\{ \frac{\log \frac{1}{\delta}}{\log \frac{1}{\rho_{np}}}, \log \frac{1}{\epsilon}, \log \frac{2V_B}{\lambda} \right\}$, where $V_B$ is the volume of hyper-ball $B$, and $C_a \epsilon d_r$ equals $d$ choose $d_r$; then with probability at least $1 - \delta$, in the solution to Problem 1, the subspace $S_b$ spanned by the features with $b_j = 1$ is the relevant subspace $S_{\text{min}}$, and the data points with $a_i = 1$ are within hyper-ball $B$.

**Proof** Please refer to the appendix.
The conditions of Theorem 3.1 are straightforward except conditions (3) and (4). The purpose of condition (3) is to limit our attention to the problems where the support regions of the majority and the minority classes overlap. According to condition (4), for any \( d_r \)-dimensional subspace, \( \forall S \in S_{d_r} \), if the projection of \( x \) on \( S \) and the projection of \( y \) on the relevant subspace \( S_{min} \) are not within hyper-ball \( B \), i.e. \( x(S) \notin B \) and \( y(S_{min}) \in B \), \( \mathcal{P}(x(S)) \) is bigger than \( \mathcal{P}(S_{min}) \) by at least \( \beta \) when there are at least \( n \) points within hyper-ball \( B \) in the relevant subspace \( S_{min} \). Therefore, this condition can be roughly interpreted as follows. The density around \( y \) neighbors is much larger than that between \( x \) nearest neighbors. In this way, assuming the other conditions in Theorem 3.1 are also satisfied, with high probability, we can identify the relevant subspace and pick the examples within \( B \) according to \( a \).

It should be pointed out that if we want to select \( n \) points from the minority class, picking them from hyper-ball \( B \) is the best we can hope for. In this way, each selected example has a certain probability of coming from the minority class. On the other hand, if some selected points are outside \( B \), their probability of coming from the minority class is 0.

4 Partial Augmented Lagrangian Method

In this section, we introduce the Partial Augmented Lagrangian Method (PALM) to effectively solve Problem 1. In our method, we alternate the optimization of \( a \) and \( b \), i.e. given the current estimate of \( a \), we solve for \( b \) that leads to the minimum value of \( F(a, b) \); given the current estimate of \( b \), we solve for \( a \) that decreases the value of \( F(a, b) \) as much as possible.

To be specific, \( F(a, b) \) can be rewritten as \( F(a, b) = \sum_{j=1}^{d} b_j \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{1}{np} a_k a_j (x_{ij} - x_{kj})^2 \). Therefore, given \( a \), we can solve for \( b \) as follows. For each feature \( j \), calculate its score \( s_j = \sum_{i=1}^{n} \sum_{k=1}^{n} a_k a_j (x_{ij} - x_{kj})^2 \). Then find the \( d \) features with the smallest scores, and set their corresponding \( b_j = 1 \). It is easy to show that this vector \( b \) minimizes \( F(a, b) \) given \( a \). On the other hand, given \( b \), solving for \( a \) is not straightforward, since \( F(a, b) \) is not a convex function of \( a \). Therefore, this problem cannot be solved by general binary integer programming (BIP) algorithms. Even though BIP algorithms can be combined with heuristics, the performance largely depends on the heuristics employed. In this paper, we first relax the constraints on \( a \): instead of requiring that \( a_i \) be binary, we require that \( a_i \in [0, 1] \), i.e. we solve the following optimization problem of \( a \):

**Problem 2**

\[
\begin{align*}
\min & \quad F_b(a) = \frac{1}{np} \sum_{i=1}^{n} \sum_{k=1}^{n} a_i a_k (\sum_{j=1}^{d} b_j (x_{ij} - x_{kj})^2) \\
\text{s.t.} & \quad \sum_{i=1}^{n} a_i = np, a_i \in [0, 1]
\end{align*}
\]

Next we use augmented Lagrangian method [24] to solve Problem 2 in an iterative way. The reason for using augmented Lagrangian method is the following: it is a combination of Lagrangian and quadratic penalty methods; compared with the Lagrangian method, the addition of the penalty terms to the Lagrangian function does not alter the stationary point of the Lagrangian function, and can help damp oscillations and improve convergence. Furthermore, the penalty parameter does not have to go to infinity in order to get the optimal solution [22]. Here, we define the following augmented Lagrangian function

\[
L_A(a, \lambda, \sigma) = \frac{1}{np} \sum_{i=1}^{n} \sum_{k=1}^{n} a_i a_k (\sum_{j=1}^{d} b_j (x_{ij} - x_{kj})^2)
\]

\[
- \sum_{i=1}^{2n+1} \lambda_i d_i(a) + \frac{\sigma}{2} \sum_{i=1}^{2n+1} d_i^2(a)
\]

where \( \lambda_i, i = 1, \ldots, 2n+1 \) are the Lagrange multipliers, \( \sigma \) is a positive parameter, and \( d_i(a), i = 1, \ldots, 2n+1 \) are a set of functions defined as follows.

\[
d_i(a) = \begin{cases} 
  c_i(a) & \text{if } i \leq 1 \text{ or } c_i(a) \leq \frac{\lambda_i}{\sigma} \\
  \frac{\lambda_i}{\sigma} & \text{otherwise}
\end{cases}
\]

\[
c_1(a) = \sum_{i=1}^{n} a_i - np = 0
\]

\[
c_{j+1}(a) = a_j \geq 0, 1 \leq j \leq n
\]

\[
c_{k+n+1}(a) = 1 - a_k \geq 0, 0 \leq k \leq n
\]

Here \( c_i(a), i = 1, \ldots, 2n+1 \) denote the original constraints on \( a \), both equality and inequality, and \( d_i(a) \) are truncated versions of \( c_i(a) \), i.e. \( d_i(a) \) is equal to \( c_i(a) \) if and only if the corresponding constraint is active or near-active; it is fixed at \( \frac{\lambda_i}{\sigma} \) otherwise.

We minimize \( L_A(a, \lambda, \sigma) \) based on Algorithm 4.20 in [22]. Putting together the optimization of \( a \) and \( b \), we have the Partial Augmented Lagrangian Method, which is presented in Algorithm 1.

The algorithm works as follows. Given the initial values \( \lambda_0 \) and \( \sigma_0 \) of \( \lambda \) and \( \sigma \), and the maximum number of iteration steps \( \text{step}_{\text{max}} \), it will output vectors \( a \) and \( b \) that correspond to a local minimum of \( F(a, b) \). In Step 1, we initialize \( a \) and \( b \). Next, in Step 2, we assign \( \lambda \) and \( \sigma \) to their initial values, and calculate \( \tilde{R}_{\text{prev}} \), which is the maximum absolute value of all the \( d_i(a) \) functions, \( i = 1, \ldots, 2n+1 \). Then
Step 4 to Step 16 are repeated \(step_{max}\) times. In Step 4, we minimize the augmented Lagrangian function with respect to \(a\), given the current estimates of \(\lambda\), \(\sigma\), and \(b\). To be specific, we use gradient descent to update \(a\), and gradually decrease the step size until convergence. Once we have obtained an updated estimate of \(a\), calculate \(K\), which is the maximum absolute value of the current \(d_i(a)\) functions. If the value of \(K\) is less than a half of \(K_{prev}\), then we update the Lagrange multipliers using the formula in Step 7, which is called the steepest ascent formula in [22]. Furthermore, we update \(K_{prev}\) using the smaller value of \(K\) and \(K_{prev}\). Otherwise, if the value \(K\) is bigger than a half of \(K_{prev}\), we double the value of \(\sigma\). Next, we update the value of \(b\) based on the current estimate of \(a\). To be specific, for each feature, we calculate its score based on the formula in Step 14. Then in Step 16, we pick \(d_i\) features with the smallest scores, and set the corresponding \(b_j\) to 1, which minimizes \(F(a, b)\) given \(a\).

In our experiments, the algorithm always converges around 20 iteration steps, so we set \(step_{max} = 30\).

**Algorithm 1 Partial Augmented Lagrangian Method (PALM)**

**Input:** Initial values of \(\lambda\) and \(\sigma\): \(\lambda_0\) and \(\sigma_0\), \(step_{max}\)

**Output:** \(a\) and \(b\)

1. Initialize \(a\) and \(b\)
2. \(\lambda = \lambda_0, \sigma = \sigma_0, K_{prev} = \|d(a)\|_\infty\)
3. for \(step = 1\) to \(step_{max}\) do
4. \(a := \text{argmin}_a L_1(a, \lambda, \sigma), K := \|d(a)\|_\infty\)
5. if \(K \leq \frac{K_{prev}}{2}\) then
6. for \(i = 1\) to \(2n + 1\) do
7. \(\lambda_i := \lambda_i - \sigma d_i(a)\)
8. end for
9. \(K_{prev} := \min(K, K_{prev})\)
10. else
11. \(\sigma := 2 \times \sigma\)
12. end if
13. for \(j = 1\) to \(d\) do
14. Calculate the score for the \(j^{th}\) feature \(s_j = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{n} a_i a_k (x_{ij} - x_{kj})^2\)
15. end for
16. Pick \(d_i\) features with the smallest scores, and set their corresponding \(b_j\) to 1
17. end for

The computational complexity of PALM can be analyzed as follows. The complexity of Step 1 and Step 2 is \(O(n)\). Inside the outer loop, the complexity of Step 4 is \(O(n^2d_i)\), the complexity of the first inner loop is \(O(n)\), the complexity of the second inner loop is \(O(n^2d)\), and the complexity of Step 16 is \(O(d,d)\). Putting everything together, the overall complexity of PALM is \(O(step_{max} \cdot \text{max}(n^2d, d, d))\).

Notice that the vectors \(a\) and \(b\) generated by PALM correspond to a local minimum of \(F(a, b)\). To improve its performance, we can run PALM with different initializations of \(a\) and \(b\) in Step 1 of Algorithm 1, and pick the best values of \(a\) and \(b\) that correspond to the smallest \(F(a, b)\).

The vectors \(a\) and \(b\) can be interpreted as follows. For \(b\), its \(d_i\) non-zero elements correspond to the relevant features. For \(a\), ideally the minority class examples should correspond to \(a_i = 1\). However, this may not be the case in practice. Therefore, we rank the elements of \(a\) from large to small, and hope to find all the minority class examples from the top of the ranked list. In other words, the elements of \(a\) that correspond to the top \(np\) examples of the ranked list are converted to 1; whereas the elements of \(a\) that correspond to the remaining examples are converted to 0.

### 5 Experimental Results

In this section, we demonstrate the performance of PALM from the following perspectives: (1) the quality of rare category selection; (2) the quality of feature selection; (3) the benefit of co-selecting features and instances simultaneously. In addition, we also want to (1) test the sensitivity of the proposed PALM to small perturbations in \(p\) and \(d_i\); and (2) compare the performance of PALM with binary integer programming (BIP).

In our experiments, we retrieve the minority class examples from the ranked list generated by different methods, and use the following performance measures: (1) the precision-scope curve, i.e. the percentage of the minority class examples when a certain number of examples are retrieved, such as \(10\% \times np, \ldots, 100\% \times np\); (2) the recall-scope curve, i.e. the percentage of the minority class examples when a certain number of MINORITY class examples are retrieved, such as \(10\% \times np, \ldots, 100\% \times np\).

#### 5.1 Synthetic Data Sets

**An illustrative example.** To demonstrate the performance of PALM, we first use a simple synthetic data set shown in Figure 1. In this figure, there are 1000 examples from the majority class, denoted as black dots, which are uniformly distributed in the feature space, and only 10 examples from the minority class, denoted as red circles, whose features on \(Z\) are uniformly distributed. Of the 3 features, only 2 features (\(X\) and \(Y\)) are relevant to the minority class, i.e. the minority class examples have very similar values on these features; and 1 feature (\(Z\)) is irrelevant to the minority class, i.e. the minority class examples spread out on this feature. Using PALM, given the number of minority class examples and the number of relevant features, we are able to identify the relevant features, with the corresponding \(b_j = 1\). Of the 10 examples with the largest \(a_i\) values, 9 examples are from the minority class, and the remaining minority class example has the 11th largest \(a_i\) value.

**Accuracy of feature selection.** Next we test the precision of the selected features of PALM using synthetic data
sets with different prior $p$. Figure 2 shows the comparison results of PALM with Laplacian score method [14], feature variance method (selecting the features with the largest variance), CRO [18], and random sampling. The x-axis is the proportion of irrelevant features, and the y-axis is the precision of the selected features. From these results, we can see that PALM is much better than the other 4 methods especially when the prior $p$ is small. As for Laplacian score method, although it is comparable with PALM for large $p$, its performance quickly deteriorates as $p$ decreases (e.g. Figure 2a and b), which is the case we are interested in for rare category analysis.

5.2 Real Data Sets

Methods for comparison and data sets. In this subsection, we test the performance of PALM on rare category selection. To the best of our knowledge, there are no existing methods for this task. For example, existing clustering algorithms (such as K-means) mainly work in the balanced settings, i.e., the proportions of different classes do not vary a lot; and they assume that all the features are relevant to each class. Therefore, we have designed the following methods for the sake of comparison.

1. Random: randomly rank all the examples, and select the first $np$ points from the ranked list as the minority class examples.

2. NNDB-based: calculate the score of each example using NNDB [13]. Note that no feedback from the labeling oracle is available, so the scores are not updated.

3. Interleave-based: calculate the score of each example using the Interleave principle [25]. Similar as the NNDB-based method, the scores of the examples are not updated in this method.

4. PALM-full: assume that all the features are relevant to the minority class, i.e. $b_j = 1, \ j = 1, \ldots, d$, and run PALM with $d_r = d$.

Note that NNDB-based method and Interleave-based method are both derived from rare category detection methods. For PALM, we tune the number of relevant features $d_r$ without any label information.

Here we use 4 real data sets, which are summarized in Table 1. In this paper, we focus on binary problems, i.e. there is only one majority class and one minority class in the data set. Therefore, for each data set, we construct several subproblems as follows. We combine the examples from two different classes into a smaller binary data set, using the larger class as the majority class, the smaller class as the minority class, and test the different methods on these binary subproblems. For each data set, we present the results on 2 binary subproblems. On the other subproblems, similar results are observed and therefore omitted for brevity.
Table 1: Properties of the data sets [2] used.

<table>
<thead>
<tr>
<th>DATA SET</th>
<th>n</th>
<th>d</th>
<th>LARGEST CLASS</th>
<th>SMALLEST CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECOLI</td>
<td>336</td>
<td>7</td>
<td>42.56%</td>
<td>2.68%</td>
</tr>
<tr>
<td>GLASS</td>
<td>214</td>
<td>9</td>
<td>35.51%</td>
<td>4.21%</td>
</tr>
<tr>
<td>ABALONE</td>
<td>4177</td>
<td>7</td>
<td>16.50%</td>
<td>0.34%</td>
</tr>
<tr>
<td>YEAST</td>
<td>1484</td>
<td>8</td>
<td>31.20%</td>
<td>1.68%</td>
</tr>
</tbody>
</table>

Accuracy of rare category selection. Figure 3 to Figure 10 compare the performance of different methods on the 4 real data sets. In these figures, the left figure shows precision vs. scope, and the right figure shows recall vs. scope. On all the data sets, PALM performs the best: the precision and recall sometimes reach 100%, such as Figure 8 and Figure 9. As for the other methods (Interleave-based, NNDB-based, and PALM-full), their performance depends on different data sets, and none of them is consistently better than Random. Comparing with Random, Interleave-based, and NNDB-based, we can see that PALM does a better job at selecting the minority class examples; comparing with PALM-full, we can see that the features selected by PALM indeed help improve the performance of rare category selection.

Notice that in some figures (Figure 3b, Figure 4b, Figure 5b, Figure 7b, and Figure 8b), at the end of the recall curves, the different methods seem to overlap with each other. This is because with no supervision, it is sometimes difficult to retrieve all the examples from the minority class, and the last example from the minority class tends to appear towards the end of the ranked list. Therefore, the recall value at 100%np is often close to the prior of the minority class in the data set.

Comparison with BIP. Next, in Figure 11 and Figure 12, we compare the performance of PALM and Binary where the vector \( a \) is obtained by a BIP algorithm combined with heuristics on Abalone data set. To be specific, in Binary, we randomly initialize a binary vector \( a \) which satisfies all the constraints in Problem 1. Then we pick each pair of elements in \( a \) with different values, and swap their values if this leads to a smaller value of the objective function. The vector \( b \) is obtained in the same way as PALM. From these figures, we can see that the performance of Binary is consistently worse than PALM in terms of both precision and recall, showing the effectiveness of PALM in obtaining the vector \( a \).

Sensitivity of PALM. Finally, we test the performance of PALM when there are small perturbations in the prior of the minority class and the number of relevant features. To this end, we first run PALM with \( p \) increased by 5% (PALM+5%) and decreased by 5% (PALM-5%) respectively, and compare their performance with PALM in Figure 13. From this figure, we can see that PALM is quite robust against small perturbations in \( p \). Then we run PALM with \( d_r \) increased by 1 (PALM+1) and decreased by 1 (PALM-1) respectively, and compare their performance with PALM and PALM-full in Figure 14. From this figure, we can see that PALM is also robust against small perturbations in \( d_r \) in most cases (Abalone, Ecoli, and Glass), and in all the cases, the performance of PALM+1 and PALM-1 is better than PALM-full (i.e. PALM without feature selection).

We tested different heuristics, and only the best performance is reported here.
Figure 3: Abalone data set: class 1 vs. class 7, $p = 0.362$, 4 features selected by PALM.

Figure 5: Ecoli data set: class 1 vs. class 4, $p = 0.197$, 3 features selected by PALM.

Figure 4: Abalone data set: class 2 vs. class 7, $p = 0.381$, 4 features selected by PALM.

Figure 6: Ecoli data set: class 2 vs. class 4, $p = 0.313$, 4 features selected by PALM.
<table>
<thead>
<tr>
<th>Scope</th>
<th>Precision</th>
<th>PALM</th>
<th>PALM-full</th>
<th>Random</th>
<th>NNDB-based</th>
<th>Interleave-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.2</td>
<td>0.3</td>
<td>0.1</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
</tr>
<tr>
<td>40</td>
<td>0.4</td>
<td>0.5</td>
<td>0.3</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
</tr>
<tr>
<td>60</td>
<td>0.6</td>
<td>0.7</td>
<td>0.5</td>
<td>0.9</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>80</td>
<td>0.8</td>
<td>0.9</td>
<td>0.7</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>100</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Figure 7**: Glass data set: class 1 vs. class 3, $p = 0.195$, 2 features selected by PALM.

<table>
<thead>
<tr>
<th>Scope</th>
<th>Recall</th>
<th>PALM</th>
<th>PALM-full</th>
<th>Random</th>
<th>NNDB-based</th>
<th>Interleave-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.3</td>
<td>0.4</td>
<td>0.2</td>
<td>0.6</td>
<td>0.8</td>
<td>0.9</td>
</tr>
<tr>
<td>40</td>
<td>0.5</td>
<td>0.6</td>
<td>0.4</td>
<td>0.8</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>60</td>
<td>0.7</td>
<td>0.8</td>
<td>0.6</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>80</td>
<td>0.9</td>
<td>1.0</td>
<td>0.8</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>100</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Figure 8**: Glass data set: class 2 vs. class 3, $p = 0.183$, 3 features selected by PALM.

<table>
<thead>
<tr>
<th>Scope</th>
<th>Precision</th>
<th>PALM</th>
<th>PALM-full</th>
<th>Random</th>
<th>NNDB-based</th>
<th>Interleave-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.1</td>
<td>0.2</td>
<td>0.1</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
</tr>
<tr>
<td>40</td>
<td>0.3</td>
<td>0.4</td>
<td>0.2</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
</tr>
<tr>
<td>60</td>
<td>0.5</td>
<td>0.6</td>
<td>0.3</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
</tr>
<tr>
<td>80</td>
<td>0.7</td>
<td>0.8</td>
<td>0.5</td>
<td>0.9</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>100</td>
<td>0.9</td>
<td>1.0</td>
<td>0.7</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Figure 9**: Yeast data set: class 2 vs. class 6, $p = 0.093$, 2 features selected by PALM.

<table>
<thead>
<tr>
<th>Scope</th>
<th>Recall</th>
<th>PALM</th>
<th>PALM-full</th>
<th>Random</th>
<th>NNDB-based</th>
<th>Interleave-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.1</td>
<td>0.2</td>
<td>0.1</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
</tr>
<tr>
<td>40</td>
<td>0.3</td>
<td>0.4</td>
<td>0.2</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
</tr>
<tr>
<td>60</td>
<td>0.5</td>
<td>0.6</td>
<td>0.3</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
</tr>
<tr>
<td>80</td>
<td>0.7</td>
<td>0.8</td>
<td>0.5</td>
<td>0.9</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>100</td>
<td>0.9</td>
<td>1.0</td>
<td>0.7</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Figure 10**: Yeast data set: class 2 vs. class 9, $p = 0.055$, 3 features selected by PALM.
6 Discussion

In this section, we discuss about the extension of our work in the following two aspects: (1) multiple minority classes; (2) no prior information.

6.1 Multiple Minority Classes

The current version of PALM can only be applied to the binary settings, i.e. there is one majority class and one minority class in the data set. PALM can be generalized to multiple minority classes in the data set by targeting the minority classes one at a time. To be specific, we may repeatedly form Problem 1 for each minority class in order to find the examples from that class as well as the relevant features. Notice that examples identified as coming from a certain minority class will be removed from the data set. Therefore, the key problem here is how to design a good ordering for the minority classes to be processed by PALM. The simplest answer is to randomly permute the minority classes. However, if minority class A has more examples and more relevant features than minority class B, and we happen to process class B first, we may accidentally select some examples from class A and assign them to class B. Since these examples will be removed from the data set, when we process class A, we will not be able to correctly identify the examples from this class as well as the relevant features. Therefore, we propose to order the minority classes according to their priors, and run PALM from the largest minority class to the smallest one.

On the other hand, in many real applications, we may have multiple majority classes. However, since we are not interested in modeling the majority classes, as long as they satisfy the conditions in Theorem 3.1, we can still use PALM to co-select the relevant features and examples from the minority classes.

6.2 No Prior Information

As shown in Section 5, PALM is robust to small perturbations in the prior of the minority class and the number of relevant features. In some applications, we may not have even a rough estimate of such information. One possible solution is to evaluate different values of $p$ and $d_r$ based on the objective function. However, notice that the objective function in Problem 1 favors smaller values of $p$ and $d_r$. In other words, smaller values or $p$ and $d_r$ will produce smaller values of the objective function. To address this problem, we may plot the value of the objective function vs. $p$ and $d_r$, and use the elbow point to find a good estimate for $p$ and $d_r$. 

Figure 11: Abalone data set: class 1 vs. class 7, $p = 0.362$.

Figure 12: Abalone data set: class 2 vs. class 7, $p = 0.381$. 
7 Conclusion
In this paper, we address the problem of unsupervised rare category analysis. To be specific, our goal is to co-select the relevant features and the examples from the minority class. To this end, we proposed an optimization framework, which is well justified theoretically. To solve this optimization problem, we designed the Partial Augmented Lagrangian Method (PALM), which alternatively finds the relevant features and the minority class examples. The effectiveness of PALM is demonstrated by extensive experimental results. Future research work includes: (1) extending the optimization framework to multiple classes, which may be addressed by running PALM with respect to the prior of each minority class, from large to small; (2) generalizing PALM to the cases where the prior information (i.e. the prior of the minority class $p$ and the number of relevant features $d_r$) is not available, which may be addressed by introducing objective functions to evaluate different values of $p$ and $d_r$.

References

Appendix

Proof of Theorem 3.1 The basic idea of the proof is to show that if the selected feature subspace $S_h$ is NOT $S_{min}$, or at least one point in the set of $np$ points is outside $B$ in $S_{min}$, we can always use $S_{min}$, and find another set of $np$ points such that all the points are within $B$, and its objective function is smaller than the original set. To do this, first, notice that according to condition (3), the expected proportion of data points falling inside $B$, $E(\frac{np}{n}) \geq p + V_B f_0$, where $n_B$ denotes the number of points within $B$. Second, according to condition (2), $\forall x \in D$, $\Pr[0 \leq \|x(S) - z(x(S))\|^2 \leq \alpha^2] = 1, i = 1, \ldots, np$. Therefore,

$$\Pr[\frac{n_B}{n} < p] < \exists x \in D, \exists S \in D_S, \text{s.t. } \psi(S(x(S))) < \phi(S(x(S)) - \beta]$$

Furthermore, according to condition (2),

$$\Pr[\frac{n_B}{n} < E(\frac{n_B}{n}) < -V_B f_0] + \int_{\frac{n_B}{n} < E(\frac{n_B}{n}) < -V_B f_0} \exp(-2n(V_B f_0)^2)$$

$$\leq \exp(-2n(V_B f_0)^2) + nC_{d^r} d_{d^r} \exp(-2n(V_B f_0)^2)$$

where $C_{d^r}$ is an upper bound on the number of subspaces in $D_S$, and Inequality (a) is based on Hoeffding’s inequality and condition (2).

Let $\exp(-2n(V_B f_0)^2) \leq c_{d^r} \exp(-2n(V_B f_0)^2) \leq \frac{\delta}{2}$, we get $n \geq \frac{1}{2(V_B f_0)^2} \log \frac{2}{\delta}$, and $n \geq \frac{\alpha^2}{4\delta^2} \log \frac{2C_{d^r}}{\delta}$. In other words, if the number of examples $n$ is sufficiently large, i.e. $n \geq \max\{\frac{1}{2(V_B f_0)^2} \log \frac{2}{\delta}, \frac{\alpha^2}{4\delta^2} \log \frac{2C_{d^r}}{\delta}\}$, then with probability at least $1 - \delta$, there are at least $np$ points within hyper-ball $B$, and $\forall x \in D, \forall S \in D_S$, $\psi(S(x(S)) \geq \phi(S(x(S)) - \beta$. Furthermore, according to condition (4), $\forall x \in D, \forall S \in D_S$, if $x(S) \notin B$, $\psi(S(x(S)) > 4r^2$. Notice that $\forall a, \forall b, F(a, b) \geq \sum_{i=1}^{\alpha_i} \psi(S_h(x(S)))$. On the other hand, if $S_h = S_{min}$, and the points with $\alpha_i = 1$ are

\[\text{Note that given } \sum_{i=1}^{\alpha_i = 1} \psi(S_h(x(S))) \text{ can be seen as the average of } np \text{ independent items.}\]
within $B$ in $S_{\text{min}}$, then $F(a, b) < 4npr^2$. This is because the squared distance between any two points within $B$ in $S_{\text{min}}$ is no bigger than $4r^2$.

Given $a$ and $b$, if $S_b$ is not $S_{\text{min}}$, we can design $a'$ and $b'$ in such a way that $S_b$ is $S_{\text{min}}$, and the points that correspond to $a'_i = 1$ are within $B$ in $S_{\text{min}}$. We can always find such a vector $a'$ since we have shown that there are at least $np$ points within $B$. Therefore, $F(a, b) \geq \sum_{i=1}^{np} \psi_{S_b}(x_{i}) > 4np > F(a', b')$. On the other hand, if $S_b$ is $S_{\text{min}}$, but at least one point with $a_i = 1$ is outside $B$, we can design $a'$ and $b'$ in such a way that $b' = b$, and $a'$ replaces the points with $a_i = 1$ that are outside $B$ with some points within $B$ that are different from existing points in $a$. For the sake of simplicity, assume that only $x_t$ is outside $B$. Therefore, $F(a, b) = \frac{1}{np} \sum_{i \neq t} \sum_{k \neq t} a_i a_k \|x_i^{(S_{\text{min}})} - x_k^{(S_{\text{min}})}\|^2 + \frac{2}{np} \sum_{i=1}^{n} a_i \|x_i^{(S_{\text{min}})} - x_i^{(S_{\text{min}})}\|^2 \geq \frac{1}{np} \sum_{i \neq t} \sum_{k \neq t} a_i a_k \|x_i^{(S_{\text{min}})} - x_k^{(S_{\text{min}})}\|^2 + 2\psi_{S_{\text{min}}}(x_t^{(S_{\text{min}})}) > \frac{1}{np} \sum_{i \neq t} \sum_{k \neq t} a_i a_k \|x_i^{(S_{\text{min}})} - x_k^{(S_{\text{min}})}\|^2 + 8r^2 \geq F(a', b')$. The above derivation can be easily generalized to the case where more than one point with $a_i = 1$ are outside $B$. Therefore, in the solution to Problem 1, $S_b$ is the relevant subspace $S_{\text{min}}$, and the data points with $a_i = 1$ are within $B$. ■