Comparison of Four Multi-Label Classification Methods

Abstract

Multi-label classification methods are closely related to modern real-world applications, including diseases diagnostics and genre classifications. Extending from the single-label classification techniques we learned in class, this project looks into four different methods (Binary Relevance One-vs-all, Binary Relevance One-vs-one, Label Powerset, ML-kNN) for multi-label classification by comparing the different methods theoretically as well as applying the methods to a real dataset on music and emotions. Through analyzing the results from real data analysis using four evaluation metrics (Accuracy, Precision, Hamming Loss, and Recall), we conclude that Binary Relevance One-vs-All method outperforms the other three methods whereas the ML-kNN method yields the worst performance.
**Background and Significance**

Traditional single-label classification methods associate each observation with only one class. In our statistics class, we focused on various techniques for single-label classification, including logistic regression, multivariate discriminant analysis, tree-based method, and support vector machines, etc. In real-world applications, however, an observation may be associated with more than one classes. For example, a patient may be diagnosed with multiple diseases at the same time; a song may belong to more than one genre; and a newspaper article may be classified into multiple categories. Thus, in this project, we focus on explaining and comparing four fundamental methods in multi-label classification through reading existing literatures and applying the methods in real data analysis.

**Concept and Methodology**

Multi-label classification techniques fall into two major categories -- *problem transformation* and *algorithm adaptation*. *Problem transformation* methods transform a multi-label classification dataset into one or more single-label classification datasets so that the classification problem will be further solved by single-label classifiers. *Algorithm adaptation* methods are extended from specific learning algorithms in order to handle multi-label data directly. Here, we introduce three problem transformation methods (*Binary Relevance One-vs-all, Binary Relevance One-vs-one, and Label Powerset*) and one algorithm adaptation method (*Multi-label k-Nearest Neighbor*).

<table>
<thead>
<tr>
<th>Example</th>
<th>x₁</th>
<th>...</th>
<th>x₇</th>
<th>Adventure</th>
<th>Drama</th>
<th>Comedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Game of Thrones</td>
<td>x₁₁</td>
<td></td>
<td>x₇₇</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>The Big Bang Theory</td>
<td>x₁₂</td>
<td></td>
<td>x₇₇</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Rick and Morty</td>
<td>x₁₃</td>
<td></td>
<td>x₇₇</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>College Romance</td>
<td>x₁₄</td>
<td></td>
<td>x₇₇</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Table 1**

To exemplify these methods we will use the data set of Table 1. It consists of four TV shows that belong to one or more of the three classes: *adventure, drama, and comedy*. Having value 1 means the TV show is labeled with that class, having 0 means the TV is not labeled with that class. In addition, associated with each TV show are 7 variables (X₁ to X₇) that are used as predictor variables for fitting classification models.

**1 Binary Relevance**

Binary relevance methods convert a multi-label dataset into multiple single-label binary datasets. One technique under binary relevance is called One-vs-All (*BR-OvA*). The *BR-OvA* method transforms the dataset with k labels into k single-label datasets and fits a binary classifier for each label. In our example, *BR-OvA* method transforms the original dataset in Table 1 to what is shown in Table 2. Solution for each of the three dataset in Table 2 will be created according to single-label binary classification.

Another technique under binary relevance is called One-vs-One (*BR-OvO*). *BR-OvO* converts a multi-label dataset into several binary datasets, where each dataset contains two different labels. In our example, dataset transformation is shown in Table 3. Note that in a dataset, instances with both labels or none of the

<table>
<thead>
<tr>
<th>Example</th>
<th>x₁</th>
<th>...</th>
<th>x₇</th>
<th>Adventure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Game of Thrones</td>
<td>x₁₁</td>
<td></td>
<td>x₇₇</td>
<td>1</td>
</tr>
<tr>
<td>The Big Bang Theory</td>
<td>x₁₂</td>
<td></td>
<td>x₇₇</td>
<td>0</td>
</tr>
<tr>
<td>Rick and Morty</td>
<td>x₁₃</td>
<td></td>
<td>x₇₇</td>
<td>1</td>
</tr>
<tr>
<td>College Romance</td>
<td>x₁₄</td>
<td></td>
<td>x₇₇</td>
<td>0</td>
</tr>
</tbody>
</table>

**Table 2**

**Table 3**
labels are removed, and the binary classifier solution is obtained from the rest of the dataset. When making prediction for a new instance, we use each binary classifier to choose one out of the two labels in a transformed dataset. Then we rank all the labels assigned to the instance according to the votes from the individual binary classifiers. Finally, we achieve multi-label classification through choosing the labels with higher ranking given a threshold.

2 Label Powerset

The label powerset method converts a multi-label dataset to a single multi-class dataset by considering each label combination as a unique class. It achieves multi-label classification by assigning an instance to a class that consists of a set of labels. In our example, we give each unique label set a class (C110, C001, C101, C011). Then a multi-class classifier is trained to assign an instance to one of the above classes.

3 Multi-Label k-Nearest Neighbor (ML-kNN)

ML-kNN is an algorithm adaptation method that predicts if an instance should be labeled with label $y_k$ based on whether a sufficient number of its k nearest neighbors are labeled with $y_k$. In particular, for an unknown instance $x_t$, the method predicts whether $x_t$ has label $y_k$ by comparing which of the following probabilities is higher: $P(x_t \text{ has label } y_k \mid \text{ the number of k-nearest neighbors labeled } y_k)$ and $P(x_t \text{ does not have label } y_k \mid \text{ the number of its k-nearest neighbors labeled } y_k)$. By applying Bayes’ Theorem, the objective turns into a comparison between $P(x_t \text{ has label } y_k) \times P(\text{the number of k-nearest neighbors labeled } y_k \mid x_t \text{ has label } y_k)$ and $P(x_t \text{ does not have label } y_k) \times P(\text{the number of k-nearest neighbors labeled } y_k \mid x_t \text{ does not label } y_k)$, both of which could be easily calculated from the given data.

Real Data Analysis and Result

We used the dataset emotions that contains 593 instances, 72 independent variables and 6 labels. Each instance is a song clip, with 72 quantitative variables describing various music features, labeled with either one or more of the 6 labels indicating emotions evoked by the music, such as “amazed-surprised” and “happy-pleased”. Initial data exploration suggests that there exists a high level of correlation among the 6 labels, as can be seen from the large number of label co-concurrences in Figure 2. We divided the original dataset into a train set containing 475 instances, and a test set containing 118 instances. Using the train set, we fit the four multi-label classifiers One-vs-All, One-vs-One, Label Powerset, and ML-kNN, and then compare the performance of these four models when applied on the test set. For the problem transformation methods, we used the default SVM classifier after the dataset is transformed into
single-label data, so the comparison across methods is fair. At the prediction stage, we used two different threshold methods: SCut threshold (Yang, 2001), which adjusts the threshold for each label to minimize MSE for the train set, and MCut threshold (Largeron, Moulin, & Gery, 2012), which determines a unique threshold for each instance based on the largest difference in ranked probabilities of the labels.

Here we define the four evaluation metrics for multi-label classification that we used when comparing the performance of the four models. Let D be a multi-label data set with |D| instances, each having a set of labels Y_i; let H be a classifier, and Z_i be the set of predicted labels for an instance x_i. Then

\[ \text{Accuracy}(H, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|}, \]

\[ \text{Precision}(H, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|Y_i \cap Z_i|}{|Z_i|}, \]

\[ \text{HammingLoss}(H, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|Y_i \Delta Z_i|}{|Y_i|}, \]

\[ \text{Recall}(H, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|Y_i \cap Z_i|}{|Y_i|}. \]

The \( \Delta \) in hamming loss stands for XOR operation in boolean logic (Tsoumakas, 2007).

The results are summarized in the following table:

<table>
<thead>
<tr>
<th></th>
<th>Using SCut Threshold</th>
<th></th>
<th>Using MCut Threshold</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One vs All</td>
<td>One vs One</td>
<td>Label Powerset</td>
<td>ML-KNN</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0.707</td>
<td>0.640</td>
<td>0.666</td>
<td>0.424</td>
</tr>
<tr>
<td>Precision</td>
<td>0.785</td>
<td>0.732</td>
<td>0.739</td>
<td>0.638</td>
</tr>
<tr>
<td>Recall</td>
<td>0.771</td>
<td>0.763</td>
<td>0.717</td>
<td>0.486</td>
</tr>
<tr>
<td>HammingLoss</td>
<td>0.122</td>
<td>0.163</td>
<td>0.149</td>
<td>0.265</td>
</tr>
</tbody>
</table>

When SCut threshold is used, the One-vs-All classifier consistently performs the best in accuracy, precision, recall and hamming loss. One-vs-One classifier and Label Powerset yield similar results, with Label Powerset slightly outperforming in terms of accuracy, precision and hamming loss. ML-kNN method yields the worst result among the four classifiers, with the lowest accuracy, precision, recall and the highest hamming loss. When MCut threshold is used, the One-vs-All classifier still performs well with respect to all four metrics, yielding the highest precision and the lowest hamming loss. Label Powerset performs the best in accuracy and recall. ML-kNN method still yields the worst result compared to the other classifiers. However, it is noteworthy that the above comparison may be data dependent. If we use other multi-label datasets or other threshold criteria then the results may differ.

Conclusions and Future Work

In this paper we summarized four methods for multi-label classification (Binary Relevance One-vs-all, Binary Relevance One-vs-one, Label Powerset, ML-kNN) and compared their performance on a multi-label data set on emotions evoked by music. Binary Relevance method, specifically One-vs-all, yields the best result in the real data analysis, but has the drawback of neglecting correlation among labels. Label Powerset considers correlation between labels indirectly, but has the drawback of not including all possible label combinations in the model-fitting process, which leads to overfitting of the train set. ML-kNN performs the worst for the given dataset, but it has the advantage of producing a ranking of the labels. In the future we intend to fit the classifiers on more multi-label data sets and investigate how the classifiers perform differently on data sets with different features.
References


Appendix

Figure 1: Number of instances for each label

Figure 2: Label Co-Concurrence

Figure 3: Performance of the four classifiers using Scut threshold

Figure 4: Performance of the four classifiers using Mcut threshold