08: Classification Evaluation and Practical Issues

Instructor: Yizhou Sun
yzsun@cs.ucla.edu

October 31, 2018
# Learnt Prediction and Classification Methods

<table>
<thead>
<tr>
<th></th>
<th>Vector Data</th>
<th>Set Data</th>
<th>Sequence Data</th>
<th>Text Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classification</strong></td>
<td>Logistic Regression; Decision Tree; KNN SVM; NN</td>
<td></td>
<td></td>
<td>Naïve Bayes for Text</td>
</tr>
<tr>
<td><strong>Clustering</strong></td>
<td>K-means; hierarchical clustering; DBSCAN; DBSCAN; Mixture Models</td>
<td></td>
<td></td>
<td>PLSA</td>
</tr>
<tr>
<td><strong>Prediction</strong></td>
<td>Linear Regression GLM*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Frequent Pattern Mining</strong></td>
<td>Apriori; FP growth</td>
<td></td>
<td>GSP; PrefixSpan</td>
<td></td>
</tr>
<tr>
<td><strong>Similarity Search</strong></td>
<td></td>
<td></td>
<td></td>
<td>DTW</td>
</tr>
</tbody>
</table>
Evaluation and Other Practical Issues

- Model Evaluation and Selection
- Other issues
- Summary
Model Evaluation and Selection

• Evaluation metrics: How can we measure accuracy? Other metrics to consider?
• Use validation test set of class-labeled tuples instead of training set when assessing accuracy
• Methods for estimating a classifier’s accuracy:
  • Holdout method, random subsampling
  • Cross-validation
Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

- **Holdout method**
  - Given data is randomly partitioned into two independent sets
  - Training set (e.g., 2/3) for model construction
  - Test set (e.g., 1/3) for accuracy estimation
  - **Random sampling**: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained

- **Cross-validation** (*k*-fold, where *k* = 10 is most popular)
  - Randomly partition the data into *k* mutually exclusive subsets, each approximately equal size
  - At *i*-th iteration, use $D_i$ as test set and others as training set
  - **Leave-one-out**: *k* folds where *k* = # of tuples, for small sized data
  - **Stratified cross-validation**: folds are stratified so that class dist. in each fold is approx. the same as that in the whole data
Classifier Evaluation Metrics: Confusion Matrix

**Confusion Matrix:**

<table>
<thead>
<tr>
<th>Actual class\Predicted class</th>
<th>$C_1$</th>
<th>$\neg C_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1$</td>
<td>True Positives (TP)</td>
<td>False Negatives (FN)</td>
</tr>
<tr>
<td>$\neg C_1$</td>
<td>False Positives (FP)</td>
<td>True Negatives (TN)</td>
</tr>
</tbody>
</table>

**Example of Confusion Matrix:**

<table>
<thead>
<tr>
<th>Actual class\Predicted class</th>
<th>buy_computer = yes</th>
<th>buy_computer = no</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>buy_computer = yes</td>
<td>6954</td>
<td>46</td>
<td>7000</td>
</tr>
<tr>
<td>buy_computer = no</td>
<td>412</td>
<td>2588</td>
<td>3000</td>
</tr>
<tr>
<td>Total</td>
<td>7366</td>
<td>2634</td>
<td>10000</td>
</tr>
</tbody>
</table>

- Given $m$ classes, an entry, $CM_{i,j}$ in a confusion matrix indicates # of tuples in class $i$ that were labeled by the classifier as class $j$
- May have extra rows/columns to provide totals
Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

- Classifier Accuracy, or recognition rate: percentage of test set tuples that are correctly classified
  \[ \text{Accuracy} = \frac{(TP + TN)}{\text{All}} \]

- Error rate: \( 1 - \text{accuracy} \), or
  \[ \text{Error rate} = \frac{(FP + FN)}{\text{All}} \]

- Class Imbalance Problem:
  - One class may be rare, e.g. fraud, or HIV-positive
  - Significant majority of the negative class and minority of the positive class

- Sensitivity: True Positive recognition rate
  \[ \text{Sensitivity} = \frac{TP}{P} \]

- Specificity: True Negative recognition rate
  \[ \text{Specificity} = \frac{TN}{N} \]
Classifier Evaluation Metrics: Precision and Recall, and F-measures

- **Precision**: exactness – what % of tuples that the classifier labeled as positive are actually positive

\[
\text{precision} = \frac{TP}{TP + FP}
\]

- **Recall**: completeness – what % of positive tuples did the classifier label as positive?

\[
\text{recall} = \frac{TP}{TP + FN}
\]

- Perfect score is 1.0

- Inverse relationship between precision & recall

- **F measure** (*F*$_1$ or **F-score**): harmonic mean of precision and recall,

\[
F = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}
\]

- **$F_\beta$**: weighted measure of precision and recall
  - assigns $\beta$ times as much weight to recall as to precision

\[
F_\beta = \frac{(1 + \beta^2) \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}}
\]
# Classifier Evaluation Metrics: Example

<table>
<thead>
<tr>
<th>Actual Class \ Predicted class</th>
<th>cancer = yes</th>
<th>cancer = no</th>
<th>Total</th>
<th>Recognition(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cancer = yes</td>
<td>90</td>
<td>210</td>
<td>300</td>
<td>30.00 (sensitivity)</td>
</tr>
<tr>
<td>cancer = no</td>
<td>140</td>
<td>9560</td>
<td>9700</td>
<td>98.56 (specificity)</td>
</tr>
<tr>
<td>Total</td>
<td>230</td>
<td>9770</td>
<td>10000</td>
<td>96.50 (accuracy)</td>
</tr>
</tbody>
</table>

- **Precision** = \( \frac{90}{230} = 39.13\% \)  
- **Recall** = \( \frac{90}{300} = 30.00\% \)
Classifier Evaluation Metrics: ROC Curves

- **ROC** (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between the **true positive rate** and the **false positive rate**
- The area under the ROC curve is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- Area under the curve: the closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model

- Vertical axis represents the true positive rate
- Horizontal axis represents the false positive rate
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0
Plotting an ROC Curve

• True positive rate: $TPR = TP / P$ (sensitivity)
• False positive rate: $FPR = FP / N$ (1-specificity)

• Rank tuples according to how likely they will be a positive tuple
  • Idea: when we include more tuples in, we are more likely to make mistakes, that is the trade-off!
  • Nice property: not threshold (cut-off) need to be specified, only rank matters
Example:

- \#P = 5
- \#N = 5

<table>
<thead>
<tr>
<th>Tuple #</th>
<th>Class</th>
<th>Prob.</th>
<th>TP</th>
<th>FP</th>
<th>TN</th>
<th>FN</th>
<th>TPR</th>
<th>FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>p</td>
<td>0.9</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>4</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>p</td>
<td>0.8</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>0.4</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>n</td>
<td>0.7</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>4</td>
<td>p</td>
<td>0.6</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>0.6</td>
<td>0.2</td>
</tr>
<tr>
<td>5</td>
<td>p</td>
<td>0.55</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0.8</td>
<td>0.2</td>
</tr>
<tr>
<td>6</td>
<td>n</td>
<td>0.54</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0.8</td>
<td>0.4</td>
</tr>
<tr>
<td>7</td>
<td>n</td>
<td>0.53</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>8</td>
<td>n</td>
<td>0.51</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>9</td>
<td>p</td>
<td>0.50</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>1.0</td>
<td>0.8</td>
</tr>
<tr>
<td>10</td>
<td>n</td>
<td>0.4</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Evaluation and Other Practical Issues

• Model Evaluation and Selection

• Other issues

• Summary
Multiclass Classification

• Multiclass classification
  • Classification involving more than two classes (i.e., > 2 Classes)
  • Each data point can only belong to one class

• Multilabel classification
  • Classification involving more than two classes (i.e., > 2 Classes)
  • Each data point can belong to multiple classes
  • Can be considered as a set of binary classification problem
Solutions

• Method 1. **One-vs.-all** (OVA): Learn a classifier one at a time
  • Given m classes, train m classifiers: one for each class
  • Classifier j: treat tuples in class j as *positive* & all others as *negative*
  • To classify a tuple $X$, choose the classifier with maximum value

• Method 2. **All-vs.-all** (AVA): Learn a classifier for each pair of classes
  • Given m classes, construct $m(m-1)/2$ binary classifiers
  • A classifier is trained using tuples of the two classes
  • To classify a tuple $X$, each classifier votes. $X$ is assigned to the class with maximal vote

• Comparison
  • All-vs.-all tends to be superior to one-vs.-all
Illustration of One-vs-All

One-vs-all (one-vs-rest):

Class 1: ▲
Class 2: □
Class 3: ✗

Classify x according to: \( f(x) = \arg\max_i f_i(x) \)
Illustration of All-vs-All

Classify x according to majority voting
Extending to Multiclass Classification

Directly

• Very straightforward for
  • Logistic Regression
  • Decision Tree
  • Neural Network
  • KNN
Classification of Class-Imbalanced Data Sets

- Class-imbalance problem
  - Rare positive example but numerous negative ones, e.g., medical diagnosis, fraud, oil-spill, fault, etc.

- Traditional methods
  - Assume a balanced distribution of classes and equal error costs: not suitable for class-imbalanced data

How about predicting every data point as blue class?
Solutions

• Pick the right evaluation metric
  • E.g., ROC is better than accuracy
• Typical methods for imbalance data in 2-class classification (training data):
  • **Oversampling**: re-sampling of data from positive class
  • **Under-sampling**: randomly eliminate tuples from negative class
  • **Synthesizing new data points** for minority class
• Still difficult for class imbalance problem on multiclass tasks

https://svds.com/learning-imbalanced-classes/
Illustration of Oversampling and Undersampling

Oversampling

Original dataset

Final dataset

Undersampling

Original dataset

Final dataset
Illustration of Synthesizing New Data Points

- SMOTE: Synthetic Minority Oversampling Technique (Chawla et. al)

  1. The first step is to ignore the majority class examples:

  2. For every minority instance, choose its $k$ nearest neighbors.

  (For 300% replication, 3 neighbors are chosen):

  3. Create new instances halfway between the first instance and its neighbors.

  Result:
Evaluation and Other Practical Issues

• Model Evaluation and Selection
• Other issues
• Summary
Summary

- Model evaluation and selection
  - Evaluation metric and cross-validation

- Other issues
  - Multi-class classification
  - Imbalanced classes