Email Classification using Co-Training

Harshit Chopra (02005001)
Kartik Desikan (02005103)
Motivation

- Internet has grown vastly
- Widespread usage of emails
- Several emails received per day (~100)
- Need to organize emails according to user convenience
Classification

- Given training data \( \{(x_1, y_1), \ldots, (x_n, y_n)\} \)
- Produce classifier \( h : X \rightarrow Y \)
- \( h \) maps object \( x_i \) in \( X \) to its classification label \( y_i \) in \( Y \)
Features

- Measurement of some aspect of given data
- Generally represented as binary functions
- Eg: Presence of hyperlinks is a feature
- Presence indicated by 1, and absence by 0.
Email Classification

- Email can be classified as:
  - Spam and non-spam, or
  - Important or non-important, etc

- Text represented as bag of words:
  1. words from headers
  2. words from bodies
Supervised learning

- Main tool for email management: text classification
- Text classification uses supervised learning
- Examples belonging to different classes is given as training set
- Eg: Emails classified as interesting and uninteresting
- Learning systems induce general description of these classes
Supervised Classifiers

- Naïve Bayes Classifier
- Support Vector Machines (SVM)
Naïve Bayes

- According to Bayes Theorem,

\[ P(\text{Class} \mid \text{Doc}) = \frac{P(\text{Class})P(\text{Doc} \mid \text{Class})}{P(\text{Doc})} \]

- Under naïve conditional independence assumption,

\[ P(\text{Doc} \mid \text{Class}) = P(a_1, \ldots, a_n \mid \text{Class}) = \prod_{i=1}^{n} P(a_i \mid \text{Class}) \]
Naïve Bayes

- Classify text document according to

\[
\max_j P(Class_j) \prod_{i=1}^{n} P(a_i | Class_j)
\]

- E.g. \(Class = \{Interesting, Junk\}\)
Linear SVM

- Creates *Hyperplane* separating data in 2 classes with max separation
Problems with Supervised Learning

- Requires lot of training data for accurate classification
- Eg: Microsoft Outlook Mobile Manager requires ~600 emails as labeled examples for best performance
- Very tedious job for average user
One solution

- Look at user’s behavior to determine important emails
- E.g.: Deleting mails without reading might be an indication of junk
- Not reliable
- Requires time to gather information
Another solution

- Use semi-supervised learning
- Few labeled data coupled with large unlabeled data
- Possible algorithms
  - Transductive SVM
  - Expectation Maximization (EM)
  - Co-Training
Co-Training Algorithm

- Based on idea that some features are redundant – “redundantly sufficient”
  1. Split features into two sets, $F_1$ and $F_2$
  2. Train two independent classifiers, $C_1$ and $C_2$, one for each feature set
  3. Produce two initial weak classifiers using minimum labeled data
Co-Training Algorithm

4. Each classifier $C_i$ examines unlabeled examples and labels them

5. Add most confident +ve and −ve examples to set of labeled examples

6. Loop back to Step 2
Intuition behind Co-Training

- One classifier confidently predicts class of an unlabeled example
- In turn, provides one more training example to other classifier
- E.g.: Two messages with subject
  1. “Company meeting at 3 pm”
  2. “Meeting today at 5 pm?”
Given: 1\textsuperscript{st} message classified as “meeting”

Then, 2\textsuperscript{nd} message is also classified as “meeting”

Messages likely to have different body content

Hence, classifier based on words in body provided with extra training example
Co-Training on Email Classification

- Assumption: presence of redundantly sufficient features describing data
- Two sets of features:
  1. words from subject lines (header)
  2. words from email bodies
Experiment Setup

- 1500 emails and 3 folders
- Division as 250, 500 and 750 emails
- Division into 3 classification problems where ratio of +ve : -ve examples are
  - 1:5 (highly imbalanced problem)
  - 1:2 (moderately imbalanced problem)
  - 1:1 (balanced problem)
- Expected: Larger the imbalance, worse the learning results
Experiment Setup

- For each task 25% of examples left as test set
- Rest is training set with labeled and unlabeled data
- Random selection of labeled data
- Stop words removed and Stemmer used
- These pre-processed words form the feature set
- For each feature/word, frequency of the word is the feature value
Experiment Run

- 50 co-training iterations
- Appropriate training examples supplied in each iteration
- Results represent average of 10 runs of the training
- Learning algorithms used:
  - Naïve Bayes
  - SVM
Results 1: Co-training with Naïve Bayes

<table>
<thead>
<tr>
<th></th>
<th>Absolute difference between accuracy in 1\textsuperscript{st} and 50\textsuperscript{th} iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Subject-based classifier</td>
</tr>
<tr>
<td>(1:5) Highly imbalanced problem</td>
<td>-17.11%</td>
</tr>
<tr>
<td>(1:2) Moderately imbalanced problem</td>
<td>-9.41%</td>
</tr>
<tr>
<td>(1:1) Balanced problem</td>
<td>0.78%</td>
</tr>
</tbody>
</table>
Inference from Result 1

- Naïve Bayes performs badly
- Possible reasons:
  - Violation of conditional independence of feature sets.
  - Subject and bodies not redundant. Need to be used together
  - Great sparseness among feature values
Result 2: Co-training with Naïve Bayes and feature selection

<table>
<thead>
<tr>
<th>Relative Imbalance</th>
<th>Absolute difference between accuracy in 1&lt;sup&gt;st&lt;/sup&gt; and 50&lt;sup&gt;th&lt;/sup&gt; iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Subject-based classifier</td>
</tr>
<tr>
<td>(1:5) Highly imbalanced problem</td>
<td>-1.09%</td>
</tr>
<tr>
<td>(1:2) Moderately imbalanced problem</td>
<td>1.62%</td>
</tr>
<tr>
<td>(1:1) Balanced problem</td>
<td>1.54%</td>
</tr>
</tbody>
</table>
Inference from Result 2

- Naïve Bayes with feature selection works lot better
- Feature sparseness likely cause of poor result of co-training Naïve Bayes
## Result 3: Co-training with SVM

<table>
<thead>
<tr>
<th>Imbalance Level</th>
<th>Subject-based classifier</th>
<th>Body-based classifier</th>
<th>Combined classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1:5) Highly imbalanced problem</td>
<td>0.28%</td>
<td>1.80%</td>
<td>1.80%</td>
</tr>
<tr>
<td>(1:2) Moderately imbalanced problem</td>
<td>14.89%</td>
<td>22.47%</td>
<td>17.42%</td>
</tr>
<tr>
<td>(1:1) Balanced problem</td>
<td>12.36%</td>
<td>15.84%</td>
<td>18.15%</td>
</tr>
</tbody>
</table>
Inference from Result 3

- SVM clearly outperforms Naïve Bayes
- Works well for very large feature sets
Conclusion

- Co-training can be applied to email classification
- Depends on learning method used
- SVM performs quite well as a learning method for email classification
References

- S Kiritchenko, S Matwin: *Email classification with co-training*, In Proc. of CASCON, 2001