Text Categorization

• Introduction to the task of text categorization
  – Manual vs. automatic text categorization
• Text categorization applications
• Evaluation of text categorization
• K nearest neighbor text categorization method
Text Categorization

- **Tasks**
  - Assign predefined categories to text documents / objects
- **Motivation**
  - Provide an organizational view of the data
- **Large cost of manual text categorization**
  - Millions of dollars spent for manual categorization in companies, governments, public libraries, hospitals
  - Manual categorization is almost impossible for some large scale application (Classification or Web pages)

Text Categorization

- **Automatic text categorization**
  - Learn algorithm to automatically assign predefined categories to text documents / objects
  - automatic or semi-automatic
- **Procedures**
  - **Training**: Given a set of categories and labeled document examples; learn a method to map a document to correct category (categories)
  - **Testing**: Predict the category (categories) of a new document
- **Automatic or semi-automatic categorization can significantly reduce manual effort**
Text Categorization: Examples

Example: 1990 US Census

- Included 22 million responses
- Needed to be classified into industry categories (200+) and occupation categories (500+)
- Estimate $15 million if done by hand
- Two alternative automatic text categorization methods evaluated
  - Knowledge-Engineering (Expert System)
  - Machine Learning (k-nearest neighbor method)
Example: 1990 US Census

- Knowledge-Engineering Approach
  - Expert System (Designed by domain expert)
  - Hand-Coded rule
    (e.g., “Professor” and “Lecturer” → “Education”)
  - Development cost: 2 experts, 8 years (192 Person-months)
  - Accuracy = 47%
- Machine Learning Approach
  - k-Nearest Neighbor (KNN) classification: details later; find your language by what language your neighbors speak
  - Fully automatic
  - Development cost: 4 Person-months
  - Accuracy = 60%

Many Applications!

- Web page classification (Yahoo-like category taxonomies)
- News article classification (more formal than most Web pages)
- Automatic email sorting (spam detection; into different folders)
- Word sense disambiguation (Java programming vs. Java in Indonesia)
- Gene function classification (find the functions of a gene from the articles talking about the gene)
- What is your favorite application?...
Techniques Explored in Text Categorization

- Rule-based Expert system (Hayes, 1990)
- Nearest Neighbor methods (Creecy’92; Yang’94)
- Decision symbolic rule induction (Apte’94)
- Naïve Bayes (Language Model) (Lewis’94; McCallum’98)
- Regression method (Furh’92; Yang’92)
- Support Vector Machines (Joachims’98)
- Boosting or Bagging (Schapier’98)
- Neural networks (Wiener’95)
- …..
### Contingency Table Per Category (for all docs)

<table>
<thead>
<tr>
<th></th>
<th>Truth: True</th>
<th>Truth: False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicted Positive</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>Predicted Negative</td>
<td>c</td>
<td>d</td>
</tr>
<tr>
<td>a+c</td>
<td>b+d</td>
<td>n=a+b+c+d</td>
</tr>
</tbody>
</table>

- **a**: number of truly positive docs
- **b**: number of false-positive docs
- **c**: number of false negative docs
- **d**: number of truly-negative docs
- **n**: total number of test documents

### Text Categorization: Evaluation

**Sensitivity**: \( \frac{a}{a+c} \)  
truly-positive rate, the larger the better

**Specificity**: \( \frac{d}{b+d} \)  
truly-negative rate, the larger the better

Depends on decision threshold, trade off between the values
Text Categorization: Evaluation

Recall: $r = \frac{a}{a+c}$  percentage of positive docs detected

Precision: $p = \frac{a}{a+b}$  how accurate are the predicted positive docs

Accuracy: $\frac{a+d}{n}$  how accurate are all the predicted docs

F-measure: $F_{\beta} = \frac{\left(\beta^2 + 1\right)pr}{\beta^2p + r}$  $F_1 = \frac{2pr}{p + r}$

Harmonic average: $\frac{1}{\frac{1}{x_1} + \frac{1}{x_2}}$

Error: $\frac{b+c}{n}$  error rate of predicted docs

Accuracy+Error=1

---

Text Categorization: Evaluation

- **Micro F1-Measure**
  - Calculate a single contingency table for all categories and calculate F1 measure
  - Treat each prediction with equal weight; better for algorithms that work well on large categories

- **Macro F1-Measure**
  - Calculate a single contingency table for every category calculate F1 measure separately and average the values
  - Treat each category with equal weight; better for algorithms that work well on many small categories
K-Nearest Neighbor Classifier

- Also called “Instance-based learning” or “lazy learning”
  - low/no cost in “training”, high cost in online prediction
- Commonly used in pattern recognition (5 decades)
- Theoretical error bound analyzed by Duda & Hart (1957)
- Applied to text categorization in 1990’s
- Among top-performing text categorization methods

From all training examples:

- Find k examples that are most similar to the new document
  - “neighbor” documents
- Assign the category that is most common in these neighbor documents
  - neighbors vote for the category
- Can also consider the distance of a neighbor
  - a closer neighbor has more weight/influence
K-Nearest Neighbor Classifier

- Idea: find your language by what language your neighbors speak

- Use K nearest neighbors to vote
  1-NN: Red; 5-NN: Brown; 10-NN: ?; Weighted 10-NN: Brown

K Nearest Neighbor: Technical Elements

- Document representation
- Document distance measure: closer documents should have similar labels; neighbors speak the same language
- Number of nearest neighbors (value of K)
- Decision threshold
K Nearest Neighbor: Framework

Training data  \( D = \{(x_i, y_i)\}, \quad x_i \in \mathbb{R}^M, \text{docs}, \quad y_i \in \{0, 1\} \)

Test data  \( x \in \mathbb{R}^M \quad \text{The neighborhood is} \quad D_k \subset D \)

Scoring Function  \( \hat{y}(x) = \frac{1}{k} \sum_{x_i \in D_k(x)} \text{sim}(x, x_i)y_i \)

Classification:  \[
\begin{cases}
1 & \text{if } \hat{y}(x) - t > 0 \\
0 & \text{otherwise}
\end{cases}
\]

Document Representation:  \( X \) uses tf.idf weighting for each dimension

Choices of Similarity Functions

- Euclidean distance  \( d(x_1, x_2) = \sqrt{\sum_v (x_{1v} - x_{2v})^2} \)
- Kullback Leibler distance  \( d(x_1, x_2) = \sum_v x_{1v} \log \frac{x_{1v}}{x_{2v}} \)
- Dot product  \( x_1 \ast x_2 = \sum_v x_{1v} \ast x_{2v} \)
- Cosine Similarity  \( \cos(x_1, x_2) = \frac{\sum_v x_{1v} \ast x_{2v}}{\sqrt{\sum_v x_{1v}^2} \sqrt{\sum_v x_{2v}^2}} \)
- Kernel functions  \( k(x_1, x_2) = e^{-d(x_1, x_2)/2\sigma^2} \) (Gaussian Kernel)

Automatic learning of the metrics
Choices of Number of Neighbors (K)

Trade off between small number of neighbors and large number of neighbors

• Find desired number of neighbors by cross validation
  – Choose a subset of available data as training data, the rest as validation data
  – Find the desired number of neighbors on the validation data
  – The procedure can be repeated for different splits; find the consistent good number for the splits
Characteristics of KNN

Pros
• Simple and intuitive, based on local-continuity assumption
• Widely used and provide strong baseline in TC Evaluation
• No training needed, low training cost
• Easy to implement; can use standard IR techniques (e.g., tf.idf)

Cons
• Heuristic approach, no explicit objective function
• Difficult to determine the number of neighbors
• High online cost in testing; find nearest neighbors has high time complexity

Text Categorization (I)

• Introduction to the task of text categorization
  – Manual vs. automatic text categorization
• Text categorization applications
• Evaluation of text categorization
• K nearest neighbor text categorization method
  – Lazy learning: no training
  – Local-continuity assumption: find your language by what language your neighbors speak