Profiling - A technical approach

8 Oct 2014
## Contents

1. **Introduction**
2. Web mining
3. Performance metrics for binary classifiers
4. Preprocessing and feature selection
5. Feature scoring and feature scaling
6. Classification algorithms
7. Conclusions
Introduction – User profiling (I)

A profile is a description of someone containing the most important or interesting facts.

In the context of users of software applications, a user profile or user model contains essential information about an individual user.

The motivation for building user profiles is that users differ in their preferences, interests, background and goals.

- Discovering these differences is vital to provide personalized services.

The content of a user profile varies among application domains.

- **Online newspaper**: topics the user like/don’t like to read, preferred newspapers, reading habits and patterns, etc.
- **Advertising**: name, age, job, interests, hobbies, etc.
Our goal in the LAMP project was to implement a profiler to build user profiles composed of preferences and interests from the information present in the Web browsing history in order to send custom advertising.
Introduction – Profiling data sources (I)

During data collection information from different sources are gathered and saved for further processing.

Main Web sources

- Content data: simple text, images, structured information from databases
  → Bag-of-words approach
- Structure data: represents the way content is organized
  → HTML or XML tags help to identify important info: title, bold, hyperlinks, etc.
- Usage data: represents the behavior of the user when browsing the Web
  → Time of access, session time, number of clicks, etc.
  → Data are collected from web servers, proxys or packet sniffers.
- User data: obtained explicitly or implicitly from the user
  → Demographic (age, education, country, status, etc.), preferences and interests.
Introduction – Profiling data sources (II)

- Common user profile contents
  - **Interests**, knowledge, background and skills, goals, behavior, interaction preferences, individual characteristics, context

- How to obtaining user information
  - Explicit information
    → The user provides data via forms or other similar interfaces, such as search terms in searching engines like the Google bar.
    → Generally, this information is demographic
      - age, sex, job, birthday, marital status
    → Interests and preferences
  - Implicit information
    → Problems with explicit info: (i) long forms, (ii) false info
    → Web mining: techniques from Machine Learning, Data Mining and Information Retrieval
Introduction – Web classification (I)

- The LAMP project performs automatic Web classification to derive the interest and preferences of the user
  - Topics of Web sites visited → Ordered list of interests

- Web classification is the process of assigning a Web page to one or more predefined category labels.

- According to the output of the classifier
  - Category ranking (soft classification) → prioritized list of categories
  - Category assignment (hard classification) → “True” or “False”

- According to the number of classes
  - Binary ($m = 2$ mutually exclusive classes)
  - $m$-ary or multiclass ($m > 2$)
    - Single label (1-of-$m$)
    - Multilabel ($n$-of-$m$)
Introduction – Web classification (II)

(a) Binary classification

(b) Multiclass, single-label, hard classification

(c) Multiclass, multilabel, hard classification

(d) Multiclass, soft classification

(a) Flat Classification

(b) Hierarchical Classification
Introduction – Web classification (III)

Multiclass strategies (1 → n)

- One vs. All
  → n binary classifiers

- One vs. One
  → n(n-1)/2 binary classifiers
  → Decision strategies
    • Majority vote (true vs. false classifiers)
    • Cumulative sum of scores
    • Directed Acyclic Graphs
Contens

1| Introduction
2| Web mining
3| Performance metrics for binary classifiers
4| Preprocessing and feature selection
5| Feature scoring and feature scaling
6| Classification algorithms
7| Conclusions
Web mining (I)

- Web mining is the use of data mining techniques to automatically discover and extract information from Web documents

- Web mining stages
  - Resources finding
  - Information selection and preprocessing
  - Generalization
  - Analysis
Web mining (II)

- Web mining categories
  - Web content mining
  - Web structure mining
  - Web usage mining

<table>
<thead>
<tr>
<th>View of Data</th>
<th>IR View</th>
<th>DB View</th>
<th>Web Content Mining</th>
<th>Web Structure Mining</th>
<th>Web Usage Mining</th>
</tr>
</thead>
</table>
| Main Data    | - Unstructured
- Semi structured | - Semi structured
- Web site as DB | - Links structure | - Interactivity |
| Representation | - Bag of words, n-grams
- Terms, phrases
- Concepts or ontology
- Relational | - Hypertext documents
- Hypertext graphs (OEM)
- Relational | - Graph | - Relational table
- Graph |
| Method       | - TFIDF and variants
- Machine learning
- Statistical (including NLP) | - Proprietary algorithms
- ILP
- (Modified) association rules | - Proprietary algorithms | - Machine Learning
- Statistical
- (Modified) association rules |
| Application Categories | - Categorization
- Clustering
- Finding extraction rules
- Finding patterns in text
- User modeling | - Finding frequent substructures
- Web site schema discovery | - Categorization
- Clustering | - Site construction, adaptation, and management
- Marketing
- User modeling |
1 | Introduction
2 | Web mining
3 | Performance metrics for binary classifiers
4 | Preprocessing and feature selection
5 | Feature scoring and feature scaling
6 | Classification algorithms
7 | Conclusions
Performance metrics for binary classifiers (I)

- **Two-way contingency tables**
  
<table>
<thead>
<tr>
<th></th>
<th>YES is correct</th>
<th>No is correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assigned YES</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>Assigned NO</td>
<td>c</td>
<td>d</td>
</tr>
</tbody>
</table>

- Cell a (true positives, TP): counts documents correctly assigned to the positive category
- Cell b (false positives, FP): counts documents incorrectly assigned to the positive category
- Cell c (false negatives, FN): counts documents incorrectly assigned to the negative category (incorrectly rejected from the positive category)
- Cell d (true negatives, TN): counts documents correctly assigned to the negative category (correctly rejected from the positive category)
Performance metrics for binary classifiers (II)

Conventional performance measures

- **Accuracy (Ac):** global percentage of documents correctly assigned both to the positive and the negative categories
  \[ Ac = \frac{TP + TN}{TP + FN + TN + FP} \]

- **Error rate (e):** global percentage of documents incorrectly assigned both to the positive and the negative categories
  \[ e = \frac{FP + FN}{TP + FN + TN + FP} \]

- **Recall (R):** percentage of documents in the positive class correctly classified as positive
  \[ R = \frac{TP}{TP + FN} \]

- **Precision (P):** percentage of documents classified as positive that actually belongs to the positive class
  \[ P = \frac{TP}{TP + FP} \]
Performance metrics for binary classifiers (III)

- Performance measures in the multiclass context using binary classifiers
  - Macro-averaging (per-category average)
    → All categories have equal weight regardless of each frequency
  - Micro-averaging (per-document average)
    → All documents have equal weights

- Limitations of conventional measures
  - Accuracy, error, precision and recall may be misleading
  - Trade-off between precision and recall
    → F-measure

\[
F_\beta(r, p) = \frac{(\beta^2 + 1)pr}{\beta^2 p + r} \quad F_1 = \frac{2rp}{r + p}
\]
<table>
<thead>
<tr>
<th>1</th>
<th>Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Web mining</td>
</tr>
<tr>
<td>3</td>
<td>Performance metrics for binary classifiers</td>
</tr>
<tr>
<td>4</td>
<td>Preprocessing and feature selection</td>
</tr>
<tr>
<td>5</td>
<td>Feature scoring and feature scaling</td>
</tr>
<tr>
<td>6</td>
<td>Classification algorithms</td>
</tr>
<tr>
<td>7</td>
<td>Conclusions</td>
</tr>
</tbody>
</table>
Preprocessing and feature selection (I)

- **Preprocessing**
  - Low rate (rare) words removal
  - Stopwords list
  - Stemming (lemmatizing)

- **Feature selection**
  - Subset of the most representative features (relevant and non-redundant) with significantly lower dimensionality
  - Benefits
    - Visualization and interpretation
    - Reducing measurement, storage, and computational requirements
    - Improving classification performance
Preprocessing and feature selection (II)

- Feature selection approaches
  - Filters (independent of the classifier)
  - Wrappers (guided by the performance of the classifier)
  - Embedded methods (within the training of the classifier)

- Filtering methods in document classification
  - Scoring features using a particular metric and selecting the best \( k \) features
  - Common metrics for filtering
    - Document frequency (DF)
    - Information gain (G)
    - Mutual information (I)
    - \( \chi^2 \) statistic (CHI)

\[
G(t) = -\sum_{i=1}^{m} P_r(c_i) \log P_r(c_i) \\
I(t, c) = \log \frac{P_r(t \wedge c)}{P_r(c)} |t| \log P_r(c | t) \\
\chi^2(t, c) = \frac{N \times (AD - CB)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)} \times \frac{(A + C) \times (A + D)}{(A + C) \times (A + D)}
\]
1 | Introduction
2 | Web mining
3 | Performance metrics for binary classifiers
4 | Preprocessing and feature selection
5 | Feature scoring and feature scaling
6 | Classification algorithms
7 | Conclusions
Feature scoring and feature scaling

- **Bag-of-words approach**
  - Dimension equal to the number of words in the vocabulary
  - An additional engineering choice is the representation of words

- **Feature scoring**
  - Term frequency – inverse document frequency (TF-IDF)
    \[
    TF - IDF = TF(t, c) \times \log\left(\frac{N + 1}{DF(t) + 1}\right)
    \]

- **Feature scaling**
  - TF values are normalized to obtain vector magnitudes equal to 1
Contents

1 | Introduction
2 | Web mining
3 | Performance metrics for binary classifiers
4 | Preprocessing and feature selection
5 | Feature scoring and feature scaling
6 | Classification algorithms
7 | Conclusions
Classification algorithms (I)

- Information Retrieval (IR) and Machine Learning (ML) provide us techniques to automatically perform Web classification.

- Paradigm of machine learning
  - A classifier is built by means of an automatic **inductive process** that learns the characteristics of the categories involved in the classification problem from a set of **pre-classified instances**
    - Inductive process: learning is done across large repetitive iterations.
    - Pre-classified documents: the learning process is a supervised process.

- The generalization ability is the capacity to correctly classify a new unseen instance.

- Train-test approach
  - Training set (learning or optimization algorithm)
  - Test set (validation in unseen documents)
Classification algorithms (II)
Linear classifiers

- Divides the space of documents linearly, i.e. establishes a linear decision threshold between classes under study

\[ \vec{c}_i = \langle w_{1i}, \ldots, w_{|\tau|_i} \rangle \]

\[ w_{ki} = \beta \sum_{d_j \in POS_i} \left| POS_i \right| - \gamma \sum_{d_j \in NEG_i} \left| NEG_i \right| \]

\[ \vec{c}_i \cdot x = \langle w_{1i}, \ldots, w_{|\tau|_i} \rangle \cdot \langle w_{1x}, \ldots, w_{|\tau|_x} \rangle > t \]

- This dot product represents a measure of similarity (or distance) between a document and the vector of weights

- The classification of a new document is done invoking this dot product
The classifier ranks the nearest neighbors among the training documents and uses the categories of the k top ranking ones to predict the category:

- Similarity between each neighbor and the new input document is computed.
- The sum of distances for neighbors belonging to the same category is the category weight, which is used for category ranking.

No inductive training process is carried out.

The position of each document in the search space.
Classification algorithms (IV) 
Probabilistic classifiers

- A probabilistic classifier interprets the classification problem in terms of the probability $P(c_i | \tilde{d}_j)$.
- This probability is computed by means of the Bayes theorem:

$$P(c_i | \tilde{d}_j) = \frac{P(c_i)P(\tilde{d}_j | c_i)}{P(d_j)}$$

- The problem is simplified assuming word independence.

- Naïve Bayes classifiers:

$$P(\tilde{d}_j | c_i) = \prod_{k=1}^{T} P(w_{kj} | c_i)$$
Classification algorithms (V)
Decision trees (DTs)

- DTs are symbolic classifiers (non-numeric)
- A decision tree is organized in “nodes”, “branches” and “leaves”
Classification algorithms (VI)
Neural Networks (NNs)

- Models for expressing knowledge using a connection paradigm based on human brain
- All neurons are arranged in interconnected layers
  - One or more hidden layers and an output layer
- Multilayer Perceptron (MLP) is the simplest and most common NN
Backpropagation is the most common way to train a neural network (optimize its weights)
- The aim is to minimize an error function

New document classification with NNs

\[ y_k = f_k(x, w) = g_i \left\{ \sum_{j=1}^{N_H} w_{jk} g_t \left( \sum_{i=1}^{d} w_{ij} x_i + b_j \right) + b_k \right\} \]
Classification algorithms (VIII)
Support Vector Machines (SVMs)

- Structural risk minimization approach
  - Generalization error = \( f \) (training error, learning technique)
  - Trade-off between the **quality** and the **complexity** of the approximation

- SVMs are **binary classifiers** that search for the **optimum separating hyperplane** that maximizes the distance between the classes under study in a higher dimensional transformed space

\[
\Phi(w, \xi) = \frac{1}{2} \|w\|^2 + C \sum_{n=1}^{N} \xi^n \\
t^n \left( w^T z^n + w_0 \right) \geq 1 - \xi^n \text{ and } \xi^n \geq 0, \quad n = 1, \ldots, N
\]
Classification algorithms (IX)
Support Vector Machines (SVMs)

- Optimum solution is given in terms of Lagrange multipliers
- Training vectors with non-zero Lagrange multiplier are the support vectors

\[ w = \sum_{n=1}^{N} \eta^n t^n \phi(x^n) \]

- Output of a SVM classifier

\[ y = \sum_{n \in S} \eta^n t^n K(x^n, x) + w_0 \]

\[ K(x^n, x^m) = (z^n)^T z^m = \phi^T(x^n) \phi(x^m) = \sum_{i=1}^{d} \phi_i(x^n) \phi_i(x^m) \]

- New patterns are classified according with the output equation
  - \( \text{Sign}(y) = +1 \) → positive class (class 1)
  - \( \text{Sign}(y) = -1 \) → negative class (class 2)
Classification algorithms (X)  
Support Vector Machines (SVMs)

- Implementation of structural risk minimization algorithm
  - QP problem need to be solved
    \[ Q = t^n t^m K(x^n, x^m) \]
  - Size of Q is \( nxn \) → Memory restrictions in high dimensionality problems
  - Implementation proposed by Osuna et al. (1997)
    → Decomposition into smaller tasks
      - Inactive part → Set of Lagrange variables temporarily fixed
      - Active part or \textit{working set} → Set of Lagrange variables to be updated/optimized
    → Problem size is \( q << n \)
Conclusions

- Web sites in the Web browser history can be used to compose a user profile.
- Machine Learning provides useful tools for automatic Web classification.
- “Bag-of-words” is the most widely used approach for document representation in automatic document classification.
- Pre-processing and feature selection stages for dimensionality reduction are essential to achieve high-performance classifiers.
- Filtering and TF-IDF are the most common techniques for feature selection, scoring, and scaling.
- SVMs are high-performance reference classifiers in the context of automatic document classification with high generalization ability.