Web Mining (Sukree Sinthupinyo : 12 June 2000)

Outline

Overview
Technique and example
Conclusion

Overview

350,000,000 Web pages [1]
Information retrieval and information extraction from the WWW
Try to understand information from WWW

Machine Learning and Web Mining

Naïve Bayes Classifier
First-order Learner
TF/IDF (Term Frequency/Inverse Document Frequency)
Etc.

Naïve Bayes [2]

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_i P(a_i | v_j)$$

Naïve Bayes (*cont.*)

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Naïve Bayes (*cont.*)

$$v_{NB} = \underset{v_j \in \{yes,no\}}{\operatorname{argmax}} P(v_j) \prod_i P(a_i | v_j)$$

$= \underset{v_j \in \{yes,no\}}{\operatorname{argmax}}$ $P(v_j)\, P(Outlook = sunny | v_j)\, P(Temperature = cool | v_j)$
$P(Humidity = high | v_j)\, P(Wind = strong | v_j)$

$= P(PlayTennis = yes) = 9/14 = .64$

$= P(PlayTennis = yes) = 5/14 = .36$

LEARN_NAÏVE_BAYES_TEXT(*Example, V*) [2]

1. collect all words, punctuation, and other tokens that occur in *Examples*
   Vocabulary all distinct words and other tokens in *Examples*
2. calculate the required $P(v_j)$ and $P(w_k|v_j)$
   For each target value v_j in V do
   $docs_j$ the subset of documents from *Examples* for which the target value is v_j
   $P(v_j)$ $|docs_j| / |Examples|$
   $Text_j$ a single document created by concatenating all members of $docs_j$
   n total number of distinct word positions in $Text_j$
   for each word w_k in *Vocabulary*
   n_k number of times word w_k occurs in $Text_j$
   $P(w_k|v_j)$ $(n_k + 1) / (n + |Vocabulary|)$

CLASSIFY_NAÏVE_BAYES_TEXT(*Doc*)

Return the estimated target value for the document *Doc*. a_i denotes the word found in the ith position within *Doc*.

positions all word positions in *Doc* that contain tokens found in *Vocabulary*
Return v_{NB}, where

$$v_{NB} = \underset{v_j \in V}{\text{argmax}} P(v_j) \prod_{i \in positions} P(a_i|v_j)$$

First-Order Text Classification [2]

FOIL [3,4]

Background relations

```
has_word(Page)      (e.g. has_apple, has_study, etc)
link_to(Page,Page)
```

Extracted Rule

```
student(A) :- not(has_data(A)), not(has_comment(A)), link_to(B,A),
              has_jame(B), has_paul(B), not(has_mail(B)).
faculty(A) :- has_professor(A), has_ph(A), link_to(B,A),
              has_faculty(B).
course(A)  :- has_instructor(A), not(has_good(A)), link_to(A,B),
              not(link_to(B,_1)), has_assign(B).
```

First-Order Text Classification (*cont.*)

Background relations

```
class(Page). link_to(Hyperlink,Page,Page). has_word(Hyperlink).
all_words_capitalized(Hyperlink). has_alphanumeric_word(Hyperlink).
has_neighborhood_word(Hyperlink).
```

Extracted Rule

```
instructors_of(A,B) :- course(A), person(B), link_to(C,B,A).
department_of(A,B)  :- person(A), department(B), link_to(C,D,A),
                       link_to(E,F,D), link_to(G,B,F),
                       has_neighborhood_word_graduate(E).
members_of_project(A,B) :- research_project(A), person(B),
                       link_to(C,A,D), link_to(E,D,B),
                       has_neighborhood_word_people(C).
```

References

[1] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam and S. Slattery. Learning to Construct Knowledge Bases from the World Wide Web. *Artificial Intelligence*, 1999.

[2] T. Mitchell. *Machine Learning*. The McGraw-Hill Companies, Inc., 1997.

[3] J.R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239-2666, 1990.

[4] J.R. Quinlan and R.M. Cameron-Jones. FOIL: A midterm repoort. In *Proceedings of the European Conference on Machine Learning*, Page 3-20, Vienna, Austria, 1993.