Using Text for Prediction

Biju Francis

10/17/08

Overview

What is prediction ?

Document Patterns and Classification

Predictive Methods

- Similarity and Nearest Neighbor Methods
- Logic Methods
- Probabilistic Methods
- Weighted Scoring Methods

Performance Evaluation

Conclusion

What does it mean?

- Predicting the future based on past examples
- Pattern must be found in the past that will hold
- Text Categorization - assignment of natural language documents to predefined categories according to their content
- e.g. Sorting newswire articles into a set of predefined categories

Document Patterns

word1	word2	word3	word4	word5	word6	…..	label
0	1	1	1	0	1		1
1	0	1	0	0	0		0
1	1	0	1	1	1		1
0	1	0	0	1	0		0
1	1	1	1	0	1		1
0	1	0	1	1	1		1
1	0	1	1	0	1		0
0	1	1	1	1	1		1

Perdictive Patterns in spreadsheet

Documents in digital form – books,manuals,newswire articles

Convert unstructured data into structured data and create dictionary

Encode document as vector of numbers(e.g. ones and zeros) representing absence or presence of individual words

Pattern is formed when combination of words occurs for the class of interest

Prediction accuracy depends on predictive quality of attributes

Document Classification

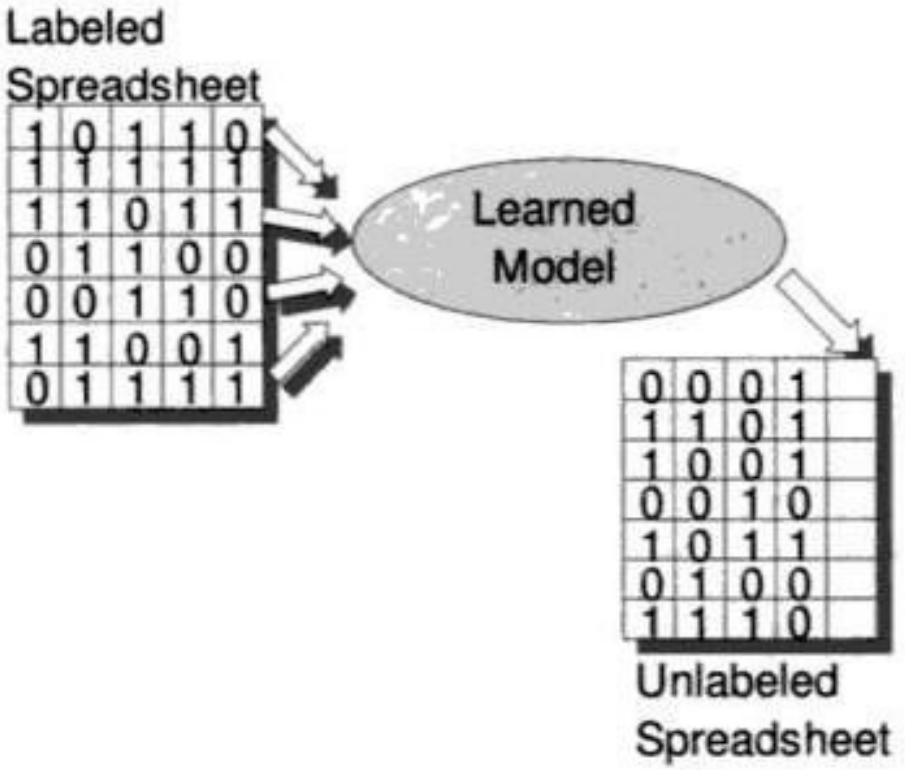

Figure 3.6. Learning and Applying Models

- Initial Document Sample needs classified/labeled – likely manually – for building predictors
- Unclassified spreadsheet has identical format
- Objective of predictive method is to learn from sample data some model that can predict values for the unclassified spreadsheet
- Documents will remain relatively stable for some time period. Over longer periods, documents in the training sample may be discarded and new ones added.

Learning to predict from Text

Given: a collection of labeled records (training set). Each record contains a set of features (attributes), and the true class (label)

Find: a learning method for the class as a function of the values of the features

Goal: previously unseen records should be assigned a class as accurately as possible

A test set is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it

Learning to predict from text

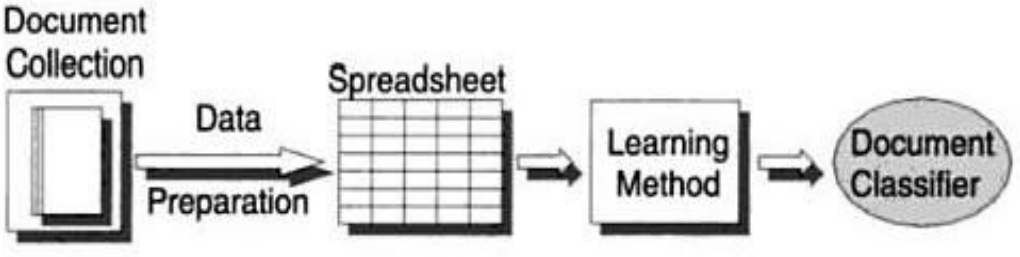

Figure 3.7. From Text to Classifiers

Cover four of the most widely used and effective methods

- Similarity and Nearest Neighbor Methods
- Decision Rules
- Scoring by Probabilities
- Linear scoring methods

Similarity and Nearest Neighbor Methods

Basic Nearest -Neighbor Algorithm for documents

Compute the similarity of newDoc to all documents in collection {D(I)}

Select the *k* documents that are most similar to newDoc.

The answer is the label that occurs most frequently in the *k* selected documents

Value of k can be estimated by experimental procedures

Generally a single value of k would be used for almost all categories

Document Similarity

Calculating similarity

- Count number of words two documents have in common
- Similarity is the number of positive words found in both stored and new document

Several other ways to measure similarity

Cosine similarity is widely used and gives better results. Its the Distance between vectors d_1 and d_2 *captured* by the cosine of the angle *x* between them.

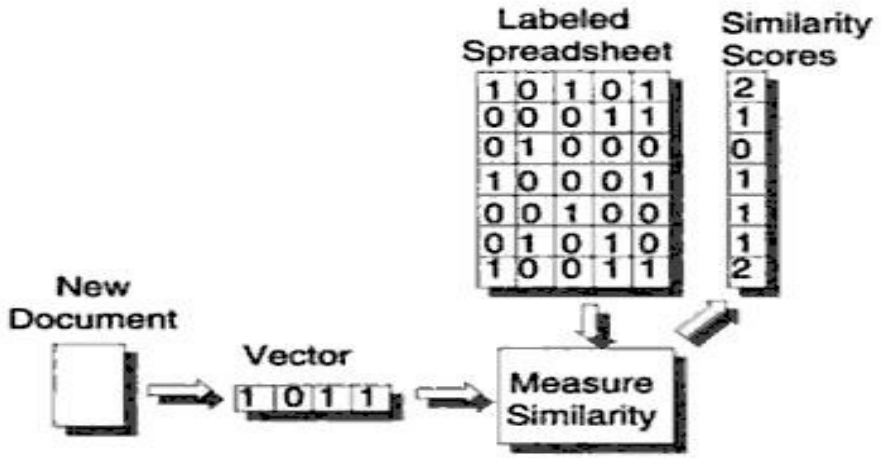

Figure 3.10. Computing Similarity Scores for Documents

Summary

Nearest Neighbor method can be viewed as a special invocation of a search engine – Instead of retrieved documents, their labels are important.

Requires no special effort to learn from the data and provides no specialized values in finding generalized patterns in the data

May need more computation time than most other methods since sequentially comparing new documents to stored ones is inefficient

Virtually zero training effort – Just collect the data and store them

Since Data is sparse, efficiency can be improved by creation of additional data structures that point to the positive entries.

Decision rules

Goal is to find one or more patterns that would produce positive documents from some hypothetical search string – These patterns are the rules for the group of positive examples

A new unlabeled document is tested against these rules – if all words in any rule are found the document's label is positive.

Example rules – For classifying documents into their country categories

“Washington DC or George Bush or Colin Powell” - “United States”

“Egypt or Kenya or Nigeria” - “Africa”

“China or India or Beijing or Japan” - “Asia”

Rules can be insightful since they are composed of meaningful words

Rules can be less predictive if the underlying concept is complex

Generating Rules

Procedures for finding decision rules are more complicated than other methods

The primary steps are the following:

1. Find a covering rule set that completely separates the two classes.
2. Prune the rule set into a sequence of smaller rule sets.
   - Optimize each rule set and repeat the pruning process until a single rule remains.
3. Evaluate the rule sets and pick the best one as the final solution.

One way to generate rules is to keep adding words to a phrase until no errors are made – as shown in the algorithm below

1. Grow conjunctive phrase T (until false positive errors are zero) by greedily adding words that minimize error.
2. Record T as the next rule R. Remove documents covered by T, and continue with step 1 until all documents are covered.

Figure 3.12. Greedy Rule Induction for Obtaining a Covering Rule Set

Generating rules

Covering rules may appear perfect because they separate the two classes.

If the learning method cannot find short phrases to cover lots of documents it will substitute longer phrases to cover fewer documents

However, they are often too specific covering few cases and overfitting the training collection of documents

Rules should make sense to the reader – if we see arbitrary rules we may question the validity of the phrase

It might be a better idea to use simpler phrases – they will make mistakes on the training documents but the more compact rules may be more accurate for predictions on new documents

Simplified phrases can be obtained by pruning the covering set of rules

Pruning Decision rules

1. Compute err/word for each single deletion operation (word or phrase).
2. Select the operation with the minimum err/word.
3. If more words remain, go back to step 1. Otherwise, the selected set of phrases is the one where err/word is minimum over all cumulative deletion operations.
4. Store the selected set of phrases and repeat the whole process by pruning that set, starting at step 1.

Figure 3.13. Pruning Decision Rules

The original covering rule set will be the most verbose set of words and phrases.

Rule set is simplified by repeatedly applying pruning operator – delete a word or phrase

Compute err/word – prune a rule set at the point where the number of errors introduced per number of discarded components is minimum

Repeat the procedure on the new smaller rule set

Pruning Rules Example

Table 3.1. Example of Rule Induction Error Summary

RSet	Rules	Words	TrainErr	TestErr	TestSD	AvgWords	Err/Word
1	9	10	.0000	.1236	.0349	9.9	0.00
2*	6	7	.0337	.1011	.0320	7.0	1.00
3	5	5	.0787	.1236	.0349	5.0	2.00
4	4	4	.0899	.1236	.0349	4.0	4.00
5**	3	3	.1011	.1124	.0335	3.0	1.00
6	2	2	.1910	.1910	.0417	2.0	8.00
7	1	1	.3820	.3820	.0515	1.0	17.00

Sample data using this process – has seven rule sets and shows the errors of each set

Covering rule set had 9 rules and 10 words and error is estimated at .1236

Rule set #2 is the minimum error rule set

Rule set #5 is the smallest rule set within 1 Standard error of the minimum error (also known as 1SE rule set)

Performance difference between minimum error and 1 SE rule set is not of much significance (1 SE Rule set generally has lower complexity)

Pruning rules Optimization

Phrases are not mutually exclusive

Deleting a word can increase overlap

Deleting a phrase may cause some positive documents to have no occurrences

Backfitting

```
for every word w in a set of phrases do
  for every dictionary word d_j do
    Compute error of set of phrases with w replaced by d_j
    If this is lower than the original error, replace w by d_j
  endfor
endfor
```

Figure 3.14. Optimizing Decision Rules

Advantage : fixes problems in rule set without changing its size

Decision rule summary

Most always choose the most compact and reasonable set of phrases (within 1SE of minimum error as explained before) to find a good decision rule

Could be a tradeoff between number of rules and overall size of rule set

- Large number of short phrases
- Small number of long phrases

Data model that employs binary (or ternary) word values is effective and maintains the interpretablility of the answers

Decision rule induction is a relatively complex procedure but the interpretability of the result is worth the effort

The results can be intuitive, informative and insightful

Scoring by Probabilities

Direct lookup of probability of words in a document

C = Class label and x = feature vector - we have to estimate probability oc class given the presence or absence of words from a dictionary

i.e. estimate Pr(C|x) = Probability score

Single labels = Choose class that has the largest probability score

Multiple labels = To maximize accuracy choose class for which probability score > 0.5

Divide document into 2 classes – C & Not C

This becomes a binary classification problem

Simplified approach to probability estimation - *Bayes with independence* or *naïve Bayes*

Scoring by Probabilities

Bayes Rule:

Pr(C|x) = Pr(x|C) * Pr(C)/Pr(x)

For 2 or more classes Pr(x) need not be computed since it won't change the ranking of Pr(C|x)

However, it needs to be computed to get a probability estimate

For two classes C1 and C2 :

Pr(x) = Pr(x|C1)Pr(C1) + Pr(x|C2)Pr(C2)

Probability of x can be computed by looking up the Probability of the presence or absence of each word and multiplying them together

$$\Pr(x|C) = \prod_j \Pr(x_j|C), \quad \Pr(x) = \sum_C \Pr(C) \prod_j \Pr(x_j|C).$$

Scoring by Probability - Example

w1	w2	w3	w4	Class
1	0	0	1	1
0	0	0	1	0
1	1	0	1	0
1	0	1	1	1
0	1	1	0	0
1	0	0	0	0
1	0	1	0	1
0	1	0	0	1
0	1	0	1	0
1	1	1	0	0

	Class=1	Class=0
Pr(Class)	0.40	0.60
Pr(w1|Class)	0.75	0.50
Pr(w2|Class)	0.25	0.67
Pr(w3|Class)	0.50	0.33
Pr(w4|Class)	0.50	0.50

Figure 3.16. Scoring by Probabilities

Pr(C) = frequency of 1's in the last column divide by number of examples

Pr(xj = 1|C) = frequency of 1's for the jth component of x where class is labeled C

Pr(x) for each word and Pr(C) is computed for the above example

Suppose we have a new document D = {w2,w3,w4}

Pr(class=1|D) = ((1-.75)*.25*.5*.5)*.4/Pr(D)) = 0.00625/Pr(D)

Pr(class=0|D) = ((1-.5)*.67*.33*.5)*.6/Pr(D)) = 0.03333/Pr(D)

As a result document D will be labeled as Class = 0

Probability scoring

The Naïve Bayes method can be expressed as a linear structure as shown below. Probability of Class C given a binary feature vector x is

$$\Pr(C|x) = \frac{\Pr(C)\prod_j \Pr(x_j = 0|C)}{\Pr(x)} \prod_j \left(\frac{\Pr(x_j = 1|C)}{\Pr(x_j = 0|C)}\right)^{x_j},$$

which can be rewritten as

$$\Pr(C|x) = \frac{1}{\Pr(x)} \exp\left(\sum_j w_j x_j + b\right),$$

where

$$w_j = \ln \frac{\Pr(x_j = 1|C)}{\Pr(x_j = 0|C)}, \quad b = \ln \Pr(C) + \sum_j \ln \Pr(x_j = 0|C).$$

Linear scoring methods

Linear Model

Word	Weight
dividend	0.8741
earnings	0.4988
eight	−0.0866
extraordinary	−0.0267
months	−0.1801
payout	0.6141
rose	−0.0253
split	0.9050
york	−0.1908
...	...

New Document

Words	Score
dividend, payout, rose	1.4629

Figure 3.17. Computing the Weighted Score of a Document

$$\text{Score}(D) = \sum_j w_j x_j + b = w \cdot x + b.$$

- Feature vector of high dimension = better prediction performance
- Linear scoring algorithm has the ability to take a large set of features and then select only useful features
- Assign score to a document based on weight of each word in the document
- Key problem is that of learning to assign weights
- Linear scoring methods can efficiently handle sparse data

Linear scoring method

Uses a complex mathematical formulation to derive scoring

Assume vector x of input variables that determines a label y {-1,1}

Given a continuous model p(x) : predict y =1 if p(x) >/0 and y=-1 otherwise

Classification error is

$I(p(x),y) = 1$ if $p(x)y <= 0$ and 0 if $p(x)y > 0$

One method to solve this is using linear predictors

$p(x) = w*x + b$ where w = weight and b is the bias – (w,b) is the weight vector

Let (xi,yi) be the i-th row of a spreadsheet – xi = vector , yi = label (1 if it belongs to category C otherwise -1)

We can compute a linear classifier by finding weight (w',b') that minimizes the average classification error in the training set

$(w',b') = \arg\min (1/n)\sum I(w*xi+b,yi)$

To make it computationally desirable we replace the classification error loss I(p,y) with the hinge loss

$\min(1/n)\sum g(w*xi+b,yi)$ where $g(p,y) = 1-py$ if $py <= 1$ else its 0

Linear scoring method

There is another method the minimizes the following loss function – known as robust classification loss

h(p,y) = -2py if py< -1,(py-1)^2/2 if py (-1,1) and 0 if py > 1

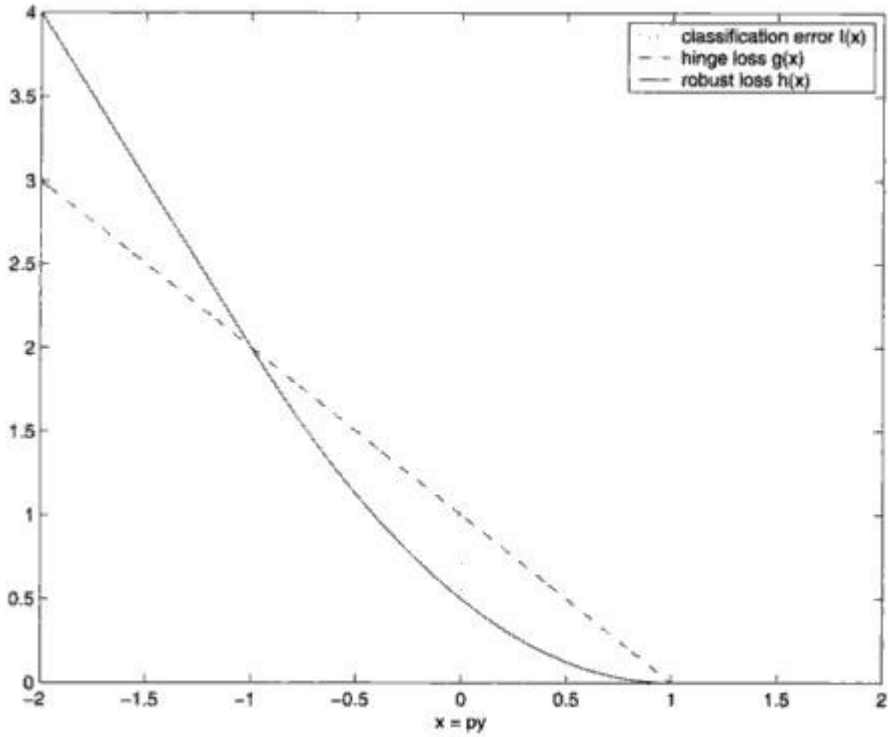

Figure 3.18. Plots of Some Loss Functions

Linear scoring methods

Input: training data $(x^1, y^1), \ldots, (x^n, y^n)$
Parameters: $K, c, \eta_1, \ldots, \eta_n$
Output: weight vector w_j $(j = 1 \ldots m)$, b
Initialize: $\alpha_i = 0$ $(i = 1 \ldots n)$; $w_j = 0$ $(j = 1 \ldots m)$; $b = 0$
for $k = 1$ to K **do**
  for $i = 1$ to n **do**
    $p = (w \cdot x^i + b)y^i$
    $d_i = \max(\min(2c - \alpha_i, \eta_i((c - \alpha_i)/c - p)), -\alpha_i)$
    $w = w + d_i x^i y^i$
    $b = b + d_i y^i$
    $\alpha_i = \alpha_i + d_i$
  endfor
endfor

Figure 3.19. Learning the Weights for the Linear Categorizer

- Goes through each data point and updating the weight accordingly
- Since it examines data sequentially it handles large amount of data without potential memory issues
- Algorithm can be terminated when a certain stopping criteria is met
- One way to do is to have a fixed number of iterations

Evaluation of Performance

Two sets of sample data – either divided randomly or by time

The two sets are similar but come from different population

Performance is evaluated by training on one sample and testing on the other

Learning takes place exclusively on the training set

Standard measure for classification is the error rate and its standard error is given below

$$\text{Error rate}(erate) = \frac{\text{number of errors}}{\text{number of documents}},$$

$$\text{Standard Error}(SE) = \sqrt{\frac{erate * (1 - erate)}{\text{number of documents}}}.$$

Evaluation of Performance

For Text Categorization a more detailed analysis of error is desired

Three ratios are used – precision,recall & F-measure

Precision :

number of correct positive predictions/number of positive predictions

Recall :

number of correct positive predictions/number of positive class documents

F-measure:

2 / (1/precision) + (1/recall) **OR** (2*precision*recall)/(precision+recall)

Evaluation of Performance

Example:

- Assume a database of labeled documents
- Also assume a label - sports
- Consider a classifier that labels documents as sports or not sports
- Performance of classifier can be assessed by computing the 3 measures
  - Recall is the percentage of all sports documents retrieved
  - Precision is the percentage of documents it correctly labels as sports
  - F-measure is the harmonic mean of precision and recall

A precision of 1.0 means every document labeled as sports document was indeed a sports document

A recall of 1.0 means that every sports document was labeled as a sports document

Precision & Recall

High precision is often valued -

If a program identifies spam email with high precision and low recall , it may leave spam in your inbox (low recall) but when it puts a spam email in trash its usually correct(high precision)

There is an inverse relationship between precision and recall

Precision – recall tradeoff: Increasing precision lowers recall (and vice versa)

Classifiers make this tradeoff by varying some constant

Precision & Recall

Nearest neighbor – Instead of simple majority set threshold to some other value. Lower value for threshold would boost recall while higher value will boost precision

Decision rules: Cost of different errors can be altered. If false negative errors are made twice as costly as false positive errors then recall would be boosted

Probabilistic scoring: threshold can be altered from 0.5 to another value Lower value would boost recall and higher value would boost precision

Linear model: Threshold can be changed from 0 to a different value. Lower value would boost recall and higher value would boost precision

Applications of Text mining

Prototypical text mining application is Text Categorization

Newswires are automatically assigned topics such as sports,finance, politics,etc..

An application we use every day:email.

In its simplest form filtering spam is an instance of binary classification – whether an email is spam or not spam

Precision takes priority over recall – Its dangerous to move a good message to trash than not to detect spam email

More applications covered in chapter 7

THANKS !!