Study of category score algorithms for k-NN classifier

H. Kou¹, G. Gardarin¹ ²
¹PRiSM Laboratory, University of Versailles Saint-Quentin, France
²e-xmlmedia Inc., France

Abstract

In the context of document categorization, this paper analyzes category score algorithms for k-NN classifier found in the literature, including majority voting algorithm (MVA), simple sum algorithm (SSA). MVA and SSA are the two mainly used algorithms to estimate the score for candidate categories in k-NN classifier systems. Based on the hypothesis that utilization of relation between documents and categories could improve system performance, we propose two new weighting calculation algorithm of category score: a concept-based weighting (CBW) score algorithm and a term independence-based weighting (IBW) score algorithm. Our experimental results confirm our hypothesis and show that in terms of precision average IBW and CBW are better than the other score algorithms while SSA is higher than MVA. According to macro-average F1 CBW performs best. The Rocchio-based algorithm (RBA) always performs worst.

1 Introduction

Document categorization is the procedure of assigning one or multiple predefined category labels to a free text document. It is a useful component of various language processing applications. A primary application of text categorization is to assign subject categories to documents to support information retrieval. It can ease the organization of increasing textual information, in particular Web pages and other electronic form documents. Many of the document categorization algorithms have been widely investigated, including k-nearest neighbor (k-NN) algorithms (Yang [1], [13], [3]), naïve Bayes algorithms [6], Rocchio algorithms [14]. Among them k-NN is one of the top-performing classifiers [12][13][2] and is comparable to the most effective support vector machine algorithm reported in [2]. The k-NN algorithm is based on the assumption that the classification of an instance is most similar to the classification of other nearby instances. The main ideas of k-NN is to find some top nearest neighbors and then estimate the score for every category by using the category membership
of these neighbors instances. The calculation approaches to category score have important impact on a k-NN classifier. In the literature, one can find different algorithms of category score calculation, for example, majority voting and similarity summing [10][12].

This paper investigates the existing algorithms for calculating category scores for a k-NN classifier and proposes two new calculation models. The paper is organized as follows: Section 2 presents related work; Section 3 presents three score models found in the literature. Section 4 introduces the independence-based weighting (IBW) calculation model and the concept-based weighting (CBW) calculation model for category score; Section 5 analyzes the results of our experiments while Section 6 concludes this paper.

2 Related work

This section will present three machine-learning algorithms that are pertinent to our works.

2.1 Rocchio Algorithm

Rocchio proposed a classic retrieval algorithm for query expansion using relevance judgment on documents [5]. The basis of this algorithm is Rocchio formula defined by (2.1).

\[
\tilde{q}'(\tilde{q}, D, \alpha, \beta, \gamma) = \alpha \cdot \tilde{q} + \beta \cdot \frac{1}{|R|} \sum_{\tilde{y} \in R} \tilde{y} + \gamma \cdot \frac{1}{|S|} \sum_{\tilde{z} \in S} \tilde{z}
\]  

(2.1)

where \(\tilde{q}\) is the original query, \(\tilde{q}'\) is the extended query, D is a training set, \(R \subset D\) consists of the training documents relevant to the query, \(S = D - R\) consists of the training documents irrelevant to the query, and \(\alpha, \beta, \gamma\) are the weights for the three components in the summation, and \(|*|\) denotes the number of documents. [14] has used a modified form (2.2) of Rocchio formula to document categorization.

\[
\tilde{c}(D, \gamma) = \frac{1}{|R|} \sum_{\tilde{y} \in R} \tilde{y} + \gamma \cdot \frac{1}{|S^*|} \sum_{\tilde{z} \in S^*} \tilde{z}
\]  

(2.2)

In (2.2) \(\tilde{c}(D, \gamma)\) is the centroid of a category and represents a category model. D and R are as defined above; \(S^* \subset D - R\) consists of the n most-similar negative instances to the positive centroid (the first part in (2.2)). Using the category centroid, a new document \(x\) can be scored by computing the cosine similarity \(r(\tilde{x}, \tilde{c}(D, \gamma)) = \cos(\tilde{x}, \tilde{c}(D, \gamma))\), where \(\tilde{x}\) is document vector of \(x\). A binary decision is obtained by thresholding on this score [14].

2.2 Naïve Bayes Algorithm

Given a document \(d\), naïve Bayes classifiers try to estimate the conditional probability \(P(c_i | d)\) that the category \(c_i\) should be assigned to the document \(d\). We take \(P(c_i | d)\) as category score. Then it assigns to the document \(d\) the category (ies)

\[
P(c_i | d) = \frac{P(d | c_i) P(c_i)}{P(d)}
\]  

(2.3)
with category score higher than a score threshold. It can be computed by (2.3). Because the $P(d)$ does not influence the order of category score, so the denominator of $P(c_i|d)$ can be ignored. The equation above can be rewritten as (2.4).

$$P(c_i|d) = P(d|c_i)P(c_i)$$  \hspace{1cm} (2.4)

where $P(c_i)$ is the prior probability that a document is in category $c_i$, and $P(d|c_i)$ is the probability of observing document $d$ in category $c_i$. To simplify the calculation of $P(d|c_i)$, one assumes that words or terms occur in the collection documents independently each other. This is well-known Bayes independence assumption. The independence assumption between terms induces (2.5).

$$P(d|c_i) = P(c_i)\prod_{j=1}^{T} P(t_j|c_i)$$  \hspace{1cm} (2.5)

So the estimation of $P(d|c_i)$ is reduced to estimating each $P(t_j|c_i)$ independently, where $P(t_j|c_i)$ is the prior probability that a term $t_j$ is in category $c_i$. $P(c_i)$ and $P(t_j|c_i)$ can be estimated by (2.6).

$$P(c_i) = \frac{|c_i|}{N}, \quad P(t_j|c_i) = \frac{1 + \sum_{d_j \in c_i} f_{jl}}{T + \sum_{r=1}^{T} \sum_{d_r \in c_i} f_{rl}}$$  \hspace{1cm} (2.6)

where $|c_i|$ is the number of training document in the category $c_i$, $N$ is the number of total training documents, $T$ is the number of total terms in an optimal feature term subset and $f_{jl}$ is the term frequency of $j$th term $t_j$ in the $l$th training document $d_l$.

The naïve Bayes classifier we present here is used in [2]. It takes into account the frequency of terms in document, and is different from one in [6] which only considers whether a term occurs in document.

### 2.3 k-NN Algorithm

According to k-NN, given a new document to be categorized, the system ranks its nearest neighbors among all training documents by calculating document similarity, and we call the categories of the $k$ top-ranking neighbors the candidate categories of the new document. The category score will be calculated for each candidate category by using the similarity of nearest neighbor documents to the new document. Then one or more categories can be assigned to the new document by a suitable thresholding strategy [15].

The calculation of category score is very crucial for k-NN. In practice, a majority voting algorithm and similarity summing algorithm are often used for category score calculation [10]. But these two algorithms do not take into account the different contribution of neighbor documents to category score (see Section 3.3). [10] has shown that in terms of accuracy the similarity sum algorithm is better than majority voting algorithm in the case of even category distribution while they are almost same in the case of uneven category distribution.
3 Category score algorithms

This section analyzes three category score algorithms found in the literature.

3.1 Majority Voting Algorithm (MVA)

For each candidate category $c_i$ majority voting algorithm (MVA) takes the number of positive documents in $D_k$ as its category score, and is defined by (3.1).

$$\text{Score.MVA}(c_i,d) = |c_i \cap D_k|$$

(3.1)

It only considers the quantity distribution of a category in local zone $D_k$ and ignores the quality distribution. Simply considering occurrences of category will mislead to categorization. For example, in the case of $k=40$, given a new document, if its first 19 nearest neighbor documents belong to one category A and the other 21 nearest neighbor documents belong to another category B, then according to MVA algorithm the new document should belong to category B. Apparently the result is less reliable. The majority-voting algorithm has been used in [10].

3.2 Simple Sum Algorithm (SSA)

Given a category, it takes as category score the simple sum of similarity between a new document and the positive documents in the nearest neighbors. A lot of researchers have adopted this algorithm [11][12][10]. The computation formula is defined by (3.2).

$$\text{Score.SSA}(c,d) = \sum_{d' \in D'_c} \text{sim}(d,d')$$

(3.2)

where $D'_c = D_k \cap c$ consists of the positive neighbor documents of category $c$. In contrast to the MVA, SSA makes use of the values of similarity between new document and positive documents in $D'_c$. According to SSA, the result of the example in Section 3.2 is odd that the new document belongs to category A.

3.3 Rocchio formula-Based Algorithm (RBA)

[14] presented different approaches to category score computation for the event tracking$^2$ in (3.3) and (3.4), where $\tilde{x}$ is a test document; $\tilde{y}(\tilde{z})$ is a positive (negative) training document; D is the training set of documents; k is the number of nearest neighbors of $\tilde{x}$ in D; $P_k(Q_k)$ is the set of the positive (negative) instances among the k nearest neighbors of $\tilde{x}$ in D. (3.4) uses the score averages instead of the score sums in (3.3). They concluded that (3.4) could significantly improve the performance of (3.3). By extending the formula (3.4), we have

$$r(\tilde{x},k,D) = \sum_{\tilde{y} \in P_k} \cos (\tilde{x},\tilde{y}) - \sum_{\tilde{z} \in Q_k} \cos (\tilde{x},\tilde{z})$$

(3.3)

$$r'(\tilde{x},k,D) = \frac{1}{|P_k|} \sum_{\tilde{y} \in P_k} \cos (\tilde{x},\tilde{y}) - \frac{1}{|Q_k|} \sum_{\tilde{z} \in Q_k} \cos (\tilde{x},\tilde{z})$$

(3.4)

\footnote{For a category, positive documents are its member documents.}
\footnote{Given an event described by a few sample instances of stories, event tracking is the procedure of identifying any and all subsequent stories describing the same event.}
Rocchio formula-based algorithm (RBA) of category score. For each candidate category \( c \), first one uses the modified Rocchio formula (2.2) to create a local centroid vector in the \( D_k \) as follows:

\[
 c(D_k, d, c, \gamma) = \frac{1}{|D_{kc}^+|} \sum_{d' \in D_{kc}^+} d' + \gamma \cdot \frac{1}{|D_{kc}^-|} \sum_{d' \in D_{kc}^-} d' 
\]  

(3.5)

where \( D_k = D_k^+ \cup D_k^- \), \( D_k^+ \) consists of the nearest documents belonging to the category \( c \) and \( D_k^- \) the nearest documents not belonging to the category \( c \). Note that \( D_k^- = \emptyset \) (empty) may hold. If \( |D_{kc}^-| = 0 \), the local centroid vector is defined by

\[
 c(D_k, d, c, \gamma) = \frac{1}{|D_k^+|} \sum_{d' \in D_k^+} d' + \frac{1}{|D_k^-|} \sum_{d' \in D_k^-} d' 
\]  

(3.6)

The category score is calculated by the formula (3.7). Obviously, if \( \gamma = -1 \), (3.7) is same as (3.3). Given a candidate category, in order to calculate its category score, RBA not only considers the positive instances but also takes into account the impact of the negative ones. If \( |D_{kc}^-| = 0 \), the second part in (3.7) is defined only by the first part.

\[
\text{Score}_{RBA}(c, d) = \cos(c, c(D_k, d, c, \gamma)) 
\]  

\[
= \frac{1}{|D_{kc}^+|} \sum_{d' \in D_{kc}^+} \cos(c, d') + \gamma \cdot \frac{1}{|D_{kc}^-|} \sum_{d' \in D_{kc}^-} \cos(c, d') 
\]  

(3.7)

4 Weighting category score algorithms

MVA simply counts the number of positive documents as category score while both SSA and RBA directly use the similarity values between neighbor instances and new document. Such estimation of category score is not fine. They will bring about wrong assignments in the case of multiple-categories.

Figure 4.1: Documents labeled by triangle belong to three categories: subtraction, plus and circle

Figure 4.1 shows an example, where the new document labeled by cross has 6 nearest neighbors labeled by triangle, and all these 6 neighbor documents belong to three categories labeled by subtraction, plus and circle. All the precedent algorithms will produce the same category score for the three categories, for example 6 by MVA, they are not capable of showing the hidden relationship: the 6 documents play more important role in the plus category than in the other two. To refine category score, it is reasonable to give higher score to the plus category. To

© 2002 WIT Press, Ashurst Lodge, Southampton, SO40 7AA, UK. All rights reserved. 
Web: www.witpress.com Email witpress@witpress.com 
Paper from: Data Mining III, A Zanasi, CA Brebbia, NFF Ebecken & P Melli (Editors). 
ISBN 1-85312-925-9
answer the problem, we propose two algorithms of category score calculation in this section: independence-based weighting (IBW) score algorithm and concept-based weighting (CBW) score algorithm. The idea is of taking into account different roles that individual sample instance document plays in each category when calculating category score.

**Hypothesis**: we believe that a better use of the internal relations between documents and categories can improve the categorization performance. More precisely, the internal relations between a document \(d\) and a category \(c\) is the level at which the document belongs to the category. We use the notion affinity to measure such level, noted as \(\text{Aff}(d, c)\).

### 4.1 Independence-based weighting (IBW) score algorithm

Supposed that the independence assumption of Bayes (Section 2.2) holds, the conditional probability \(P(c|d)\) of category presence can be simply calculated for every category \(c\) given a training document \(d\) by combining the formulas (2.3)-(2.6). We take the value of such conditional probability as affinity between training document and category:

\[
\text{Aff}(d, c) = P(c|d)
\]  

(4.1)

Obviously, a member document of category produces the high value of affinity while a non-member document do low one. Based on the formula (4.1), given a training document \(d\) and a category \(c\), the category score can be calculated by (4.2). We call (4.2) independence-based weighting (IBW) algorithm.

\[
\text{Score}_{IBW}(c, d) = \sum_{d' \in D^+_k} \text{simil}(d, d') \cdot \text{Aff}(d', c)
\]

(4.2)

where \(D^+_k\) consists of positive documents of category \(c\) in the \(k\) nearest neighbor documents. We introduce \(\text{Aff}(d', c)\) in (4.2) to weight the different contributions of training document to category score.

### 4.2 Concept-based weighting (CBW) score algorithm

Some experiences have shown that centroid vector of category can reflect the category theme and summarize the characteristics of the category [3]. So, centroid vectors can represent categories at the level of concepts or domain topics [4]. Due to this fact, centroid is also called concept vector of category. The similarity between training document and centroid of category can measure the level of membership of document to category. The following table shows this fact by using some documents and categories in Reuters-21578 collection [8]:

<table>
<thead>
<tr>
<th>DocNo</th>
<th>Category</th>
<th>Similarity</th>
<th>Member</th>
</tr>
</thead>
<tbody>
<tr>
<td>2938</td>
<td>earn</td>
<td>0.678</td>
<td>yes</td>
</tr>
<tr>
<td>2938</td>
<td>soybean</td>
<td>0.01</td>
<td>no</td>
</tr>
<tr>
<td>2964</td>
<td>soybean</td>
<td>0.29</td>
<td>yes</td>
</tr>
<tr>
<td>2964</td>
<td>earn</td>
<td>0.023</td>
<td>no</td>
</tr>
<tr>
<td>5702</td>
<td>soybean</td>
<td>0.721</td>
<td>yes</td>
</tr>
<tr>
<td>5702</td>
<td>earn</td>
<td>0.029</td>
<td>no</td>
</tr>
</tbody>
</table>

From above table we can see that the member documents of category produce higher similarity values than non-member documents. So it is reasonable to
consider the value of cosine similarity between document vector \( d \) and concept vector \( v_c \) of category \( c \) as affinity (4.3).

\[
A(f(d,c)) = \text{simil}(d,v_c)
\] (4.3)

With the formula (4.3), a concept-based weighting (CBW) algorithm of category score calculation can be defined by (4.4).

\[
\text{Score.CBW}(c,d) = \sum_{d' \in D^c} \text{simil}(d,d') \cdot A(f(d',c))
\] (4.4)

5 Experiments and analysis

Evaluation of binary classification. Recall and precision are two commonly used measures of performance. Given a category, recall (\( r \)) is the proportion of correctly assigned documents to all documents belonging to the category and precision (\( p \)) one of correctly assigned documents to all assigned documents. There is a trade-off between recall and precision, hence \( F_1(r,p) \) measure is often used by combing recall and precision. \( F_1(r,p) \) is defined as \((2rp/(r+p))\). When recall and precision are same, \( F_1(r,p) \) takes its highest value that is equal to recall [13].

Macro- and Micro-Average. There are two ways to measure the average performance of a binary classifier over multiple categories, namely, the macro-average and the micro-average [13]. Macro-average gives an equal weight to the performance on every category, regardless how rare or how common a category is. Micro-average, however, gives an equal weight to the performance on every document, thus favoring the performance on common categories.

11-points Recall-Precision Graph. The classifiers often produce a score for category-document pair. In this case, given a category, we can rank documents in the descending order of their scores. Then by going down the document list, we calculate both recall and precision when encountering a positive document. We call all calculated recall-precision pairs observed points. Finally, we use these observed points to produce 11 new points with prefixed recall values and furthermore draw recall-precision curve for a category. See [9] for detail.

Experiment design. We use Reuter-21578 collections as experiment benchmark. ApteMod split strategy is taken but we only chose 9805 news stores with news body and eliminate the news stories without news body and category labels. As result, 7063 training documents and 2742 test documents are included in our experiment. Furthermore, we chose 21 largest categories to test. After removing 319 stop-words, Porter stemming algorithm [7] is performed to convert words into their word stems, and 14,743 unique terms are obtained. \( \chi^2 \) approach [11] is used to measure the predictive power of terms for categories. In our experiments, 1100 feature terms with higher \( \chi^2 \) value are selected as optimal subset of feature terms, and vector model is adapted to represent documents by using tf-idf term weighting model [9]. k-NN is conducted by taking the value of \( k \) as 10,20,30,40,50,60,70 respectively, and RCut thresholding strategy [15] is used with its value 1. The category score algorithms MVA, SSA, RBA (\( \gamma = -1 \)), IBW and CBW are tested.

Due to the limit of space, we only present some parts of our experimental results. Figure 5.1 shows the macro average F1 performance of k-NN classifier by
these 5 score algorithms with different values of k: 10, 20, 30, 40, 50, 60, 70. CBW performance is the highest for seven k values. SSA achieves the same performs as CBW at 40, 50, 60 and 70. MVA is less than IBW and SSA. For the small k value (10, 20, 30) IBW is higher than SSA while it is lower than SSA for the big k value (40, 50, 50, 70). RBA performs worst, which means that negative documents decrease performance.

![MacroF1](image1)

**Figure 5.1:** macro-average F1 by 5 score models

![Recall-Precision Curve(k=20)](image2)

**Figure 5.2:** recall-precision curves of different score models with k=20
According to the average of macro-average F1 over these seven experiments, the 5 score algorithms can be ordered in (5.1). The inequality $\text{CBW} > \text{SSA}$ confirms that introduction of internal relationship between training documents and categories can improve the system performance.

$$\text{CBW} > \text{SSA} > \text{IBW} > \text{MVA} >> \text{RBA} \quad (5.1)$$

Figure 5.2 is their recall-precision curves for $k=20$. IBW outperforms the others. CBW is higher than SSA over the high recall level range $[0.6, 1]$ while the performances of SSA and CBW are mixed over the low recall level range $[0, 0.56]$. Again MVA is less than SSA with the exception of the lowest recall level range $[0,0.26]$. Obviously, RBA is inferior to the other 3 models.

$$\text{IBW} > \text{CBW} > \text{SSA} > \text{MVA} >> \text{RBA} \quad (5.2)$$

In order to get more insight into the performance of different score algorithms, we display the average precision-k curves over the recall level range $[0.3, 0.7]$ in Figure 5.3. We can see that IBW is better than CBW and both of them outperform the others while MVA is lower than SSA. The curve of RBA is very low and is not shown in Figure 5.3. Then they can be ranked as follows (5.3):

$$\text{IBW} > \text{CBW} > \text{SSA} > \text{MVA} >> \text{RBA} \quad (5.3)$$

![Figure 5.3: Precision averages of score models over recall range [0, 0.7]](image)

From the order relation expressions (5.1) and (5.3), the following results can be achieved:

In any case, one of IBW and CBW is the best among 5 score algorithms. According to (5.2) and (5.3) they are better than the others. One explanation to this fact is the utilization of internal relations between documents and categories for calculating category score. This proves our hypothesis made in Section 4.

SSA is superior to MVA. It is reasonable because SSA uses the values of similarity between a new document and category instance documents while MVA only uses the number of category instance documents in k top neighbors.

The performance of RBA is always much less than the others in our experiments. This implies that introduction of negative documents in RBA cannot improve performance but decreases performance by contraries.
6 Conclusion

Due to the increasing volume of text information across WWW, document categorization becomes an effective approach to organizing such information. k-NN is a classical machine-learning algorithm and has been applied to document categorization. This paper discussed existing calculation algorithms (SSA, MVA, RBA) of category score used in k-NN classifiers. Then we proposed two weighting algorithms (CBW and IBW) to calculate category score. The empirical results confirm that subtle utilization of internal relation between documents and categories for calculating category score can benefit performance of k-NN classifier. In the term of precision average IBW and CBW are better than the other score models, while SSA is higher than MVA. According to macro-average F1 CBW performs best. Rocchio-based algorithm (RBA) always performs worst.

References