

Chapter 5

Classifying Unstructured Textual Data Using the Product Score Model: An Alternative Text Mining Algorithm

Qiwei He and Bernard P. Veldkamp

Abstract Unstructured textual data such as students' essays and life narratives can provide helpful information in educational and psychological measurement, but often contain irregularities and ambiguities, which creates difficulties in analysis. Text mining techniques that seek to extract useful information from textual data sources through identifying interesting patterns are promising. This chapter describes the general procedures of text classification using text mining and presents an alternative machine learning algorithm for text classification, named the product score model (PSM). Using the bag-of-words representation (single words), we conducted a comparative study between PSM and two commonly used classification models, decision tree and naïve Bayes. An application of these three models is illustrated for real textual data. The results showed the PSM performed the most efficiently and stably in classifying text. Implications of these results for the PSM are further discussed and recommendations about its use are given.

Keywords: text classification, text mining, product score model, unstructured data

Introduction

Language is magic that diversifies our lives. The way individuals talk and write provides a window into their emotional and cognitive worlds. Yet despite the interesting attributes of textual data, analyzing them is not easy. One of the major reasons is that textual data are generally more diverse than numerical data and are often unstructured, neither having a predefined data model nor fitting well into relational patterns. The irregularities and ambiguities make it even harder to classify textual data compared with structured data stored in field form in databases. Thus, to address the challenge of exploiting textual information, new methods need to be developed.

The development of information technology demonstrated breakthroughs in handling unstructured textual data during the past decade. A promising technique is text mining, which exploits information retrieval, information extraction, and corpus-based computational linguistics. Analogous to data mining, text mining seeks to extract useful information from textual data sources by identifying interesting patterns.

However, a preprocessing step is required to add transforming unstructured data stored in texts into a more explicitly structured intermediate format (Feldman & Sanger, 2007).

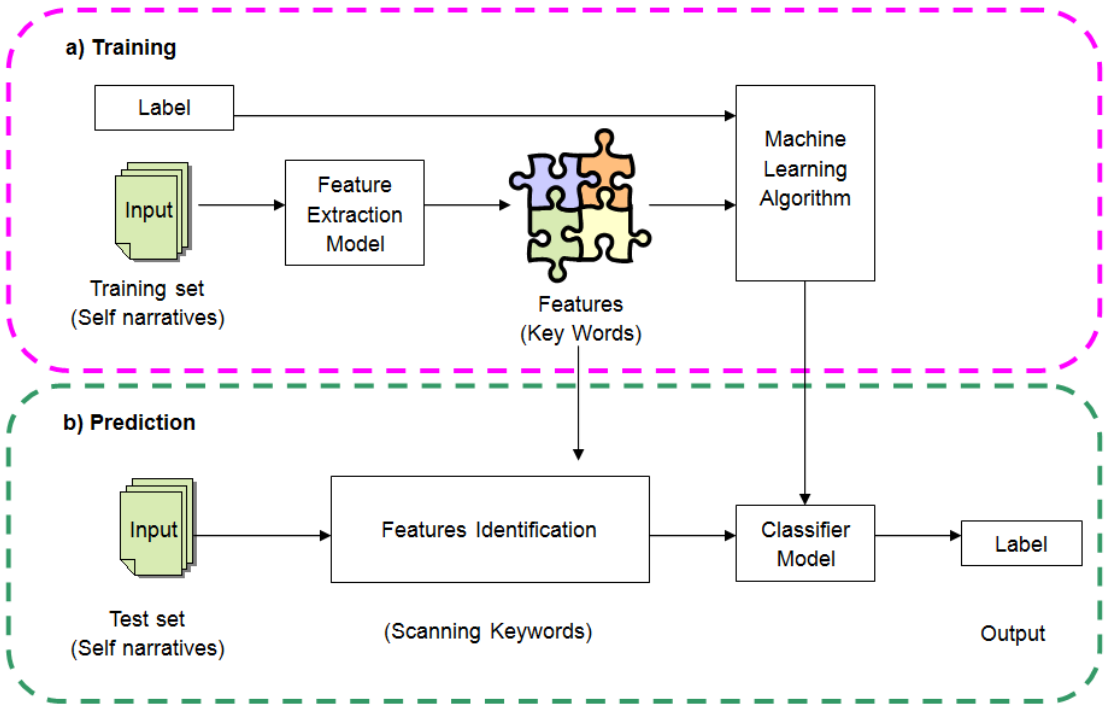
Text mining techniques are used, for example, for text classification, where textual objects from a universe are assigned to two or more classes (Manning & Schütze, 1999). Common applications in educational measurement classify students' essays into different grade levels with automated scoring algorithms, e.g., Project Essay Grade (PEG; Page, 2003) and automated scoring of open answer questions, e.g., E-raters (Burstein, 2003). Feature extraction and machine learning are the two essential sections in text classification, playing influential roles in classification efficiency. During feature extraction, textual components are transformed into structured data and labeled with one or more classes. Based on these encoded data, the most discriminative lexical features are extracted by using computational statistic models, such as the chi-square selection algorithm (Oakes, Gaizauskas, Fowkes, Jonsson, & Beaulieu, 2001) and likelihood ratio functions (Dunning, 1993). In the machine learning section, documents are allocated into the most likely classes by applying machine learning algorithms such as decision trees (DTs), naïve Bayes (NB), support vector machines (SVM), and the K nearest neighbor model (KNN). Although many machine learning classifiers have been tested efficiently in text classification, new alternative models are still being explored to further improve text classification performance and accelerate the speed of word processing (see more in Duda, Hart, & Stork, 2001; Vapnik, 1998).

This chapter briefly describes the general procedure for supervised text classification where the actual status (label) of the training data has been identified ("supervised"), introduces an effective and much used feature extraction model, i.e., the chi-square selection algorithm, and presents an alternative machine learning algorithm for text classification, named the product score model (PSM). To evaluate the PSM performance, a comparative study was conducted between PSM and two standard classification models, DTs and NB, based on an example application for real textual data. The research questions focus on (a) whether the PSM performs more efficiently in classifying text compared to the standard models, and (b) whether the PSM maintains stable and reliable agreement with the human raters' assessment.

Supervised Text Classification

Supervised text classification is a commonly used approach for textual categorization, which generally involves two phases, a training phase and a prediction phase (Jurafsky & Martin, 2009; see Figure 1).

During training, the most discriminative keywords for determining the class label are extracted. The input for the machine learning algorithm consists of a set of prespecified keywords that may potentially be present in a document and labels classifying each document. The objective of the training phase is to “learn” the relationship between the keywords and the class labels. The prediction phase plays an important role in checking how well the trained classifier model performs on a new dataset. The test set should consist of data that were not used during training. In the testing procedure, the keywords extracted from the training are scanned in each new input. Thus, the words that were systematically recognized are fed into the “trained” classifier model, which predicts the most likely label for each new self-narrative. To ensure proper generalization capabilities for the text classification models, a cross-validation procedure is generally applied.



Note: Supervised text classification generally involves two phases, training and prediction. The objective of the training phase is to model (i.e., to learn) the relationship between the keywords and labels. The prediction is used to check how well the trained classifier model performs on a new dataset.

Figure 1 The framework of supervised text classification

To improve the efficiency of the training and prediction procedure, a preprocessing routine is often implemented. This involves screening digital numbers, deducting noninformative “stop words” (e.g., “I”, “to”), common punctuation marks (e.g., “.”, “:”), and frequently used abbreviations (e.g., “isnt”, “Im”), and “stemming” the rest of words, for instance, with the Porter algorithm (Porter, 1980) to remove common morphological endings. For example, the terms “nightmares,” “nightmaring,” and “nightmared,” though in variant lexical forms, are normalized in an identical stem “nightmar”¹ by removing the suffixes and linguistic rule-based indicators.

Chi-Square Feature Selection Algorithm

A classifier extraction can be designed to capture salient words or concepts from texts using a feature selection algorithm that compares the frequency of each word type in the text corpus² of interest to the frequency of that word type in the whole text corpora (Conway, 2010). Forman (2003) reviewed many feature selection methods for text classification, in which the chi-square selection algorithm (Oakes et al., 2001) was recommended for use due to its high effectiveness in finding robust keywords and testing for the similarity between different corpora. Thus, we briefly introduce this algorithm here and then apply it in the example data.

To apply the chi-square algorithm for feature selection, the N word types in the training set are compiled into an N -by-2 table, schematically shown in Table 1. The two columns correspond to the two corpora, C_1 and C_2 . Each row corresponds to a particular word i . The number of word occurrences in C_1 and C_2 is indicated by n_i and m_i , respectively. The sum of the word occurrences in each corpus is defined as the corpus length,

$$\text{len}(C_1) = \sum_{i=1}^k n_i, \quad \text{len}(C_2) = \sum_{i=1}^k m_i \quad (1)$$

Table 1 Structuralizing Textual Data in a Binary Classification

	C_1	C_2
Word 1	45	1
Word 2	23	0
\vdots	\vdots	\vdots
Word i	n_i	m_i
\vdots	\vdots	\vdots
Word k	n_k	m_k
Total	$len(C_1)$	$len(C_2)$

Note: C represents the class label of text corpus, and n_i and m_i represent the number of occurrences of a word i in two corpora, respectively

Table 2 Confusion Matrix for Word i in the 2-by-2 Chi-Square Score Calculation

	C_1	C_2
Word i	n_i	m_i
\neg Word i	$len(C_1) - n_i$	$len(C_2) - m_i$

Note: C represents the class label of text corpus, and n_i and m_i represent the number of occurrences of a word i in two corpora, respectively

Each word is then compiled into its own 2-by-2 contingency table as shown in Table 2. The values in each cell are called the observed frequencies (O_{ij}). Using the assumption of independence, the expected frequencies (E_{ij}) are computed from the marginal probabilities. The chi-square statistic sums the differences between the observed and the expected values in all squares of the table, scaled by the magnitude of the expected values, as the following formula:

$$X^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}. \quad (2)$$

To ensure the reliability of the calculation, as Manning and Schütze (1999) suggested, in practice features or words that occur fewer than five times are usually eliminated. However, for a small sample, the number of word occurrences could be even lower, perhaps three times. Based on the chi-square scores, all words are ranked in descending order, and those standing at the top are extracted as robust classifiers.³ Further, if the ratio n_i / m_i is larger than the ratio $len(C_1) / len(C_2)$, the word is regarded as more typical of corpus C_1 (as a “positive indicator”); otherwise, it is more typical of corpus C_2 (as a “negative indicator”) (Oakes et al., 2001).

Text Classification Models

Training text classifiers is the procedure where machines “learn” to automatically recognize complex patterns, to distinguish between exemplars based on their different patterns, and to make intelligent predictions on their class. Among various machine learning algorithms, decision trees (C4.5; Quinlan, 1993) and naïve Bayes are two of the most widely used text classification models (see more algorithms in Kotsiantis, 2007).

Decision Trees

A decision tree is a well-known machine learning approach to automatically induce classification trees based on training data sets. In the tree structures, leaves represent class labels, and branches represent conjunctions of features that lead to those class labels. The feature that best divides the training data is the root node of the tree. There are numerous methods for finding the feature that best divides the training data such as information gain (Hunt, Marin, & Stone, 1966) and the Gini index (Breiman, 1984). The objects at each node are split into piles in a way that gives maximum information gain and stopped until they are categorized into a terminate class.

Naive Bayes

Naive Bayes is a probabilistic classifier applying Bayes’s theorem with strong (naive) independence assumptions (Lewis, 1998). It is simple but effective in practice (Hand & Yu, 2001). In text classification, the basic idea behind NB is to estimate probabilities of categories given a text document by using the joint probabilities of words and categories with the assumption of word independence. Namely,

$$P(C, \mathbf{w}) = \frac{p(C)p(w_1|C)p(w_2|C)\dots p(w_k|C)}{p(w_1, \dots, w_k)} = \frac{p(C)\prod_{i=1}^k p(w_i|C)}{p(\mathbf{w})}, \quad (3)$$

where C represents a specific class and \mathbf{w} represents the keyword vectors. $p(C)$ is the prior probability of a certain class, and $p(w_i|C)$ is the conditional probability of a word occurs in a certain class. In the binary classification, the two probabilities from categories C_1 and C_2 could be simply compared in a ratio R . That is,

$$R = \frac{P(C_1, \mathbf{w})}{P(C_2, \mathbf{w})} = \frac{p(C_1)\prod_{i=1}^k p(w_i|C_1)}{p(C_2)\prod_{i=1}^k p(w_i|C_2)}. \quad (4)$$

If $R > 1$, the object is classified in category C_1 ; else it is classified in category C_2 .

Product Score Model

The product score model (He, Veldkamp, & de Vries, 2012) is an alternative machine learning algorithm, which features in assigning two weights for each keyword (in binary classification)—the probability of the word i occurs in the two separate corpora, U_i and V_i —to indicate to how much of a degree the word can represent the two classes. The weights are calculated by

$$\begin{cases} U_i = (n_i + a) / \text{len}(C_1) \\ V_i = (m_i + a) / \text{len}(C_2) \end{cases}. \quad (5)$$

Note that a smoothing constant a (we use $a = 0.5$ in this study) is added to the word occurrence in Formula (5) to account for words that do not occur in the training set, but might occur in new texts. (For more on smoothing rules, see Manning & Schütze, 1999; Jurafsky & Martin, 2009.)

The name *product score* comes from a product operation to compute scores for each class, i.e., S_1 and S_2 , for each input text based on the term weights. That is,

$$\begin{cases} S_1 = P(C_1) \cdot \prod_{i=1}^k U_i = P(C_1) \cdot \prod_{i=1}^k [(n_i + a) / \text{len}(C_1)] \\ S_2 = P(C_2) \cdot \prod_{i=1}^k V_i = P(C_2) \cdot \prod_{i=1}^k [(m_i + a) / \text{len}(C_2)] \end{cases} \quad (6)$$

where a is a constant, and $P(C)$ is the prior probability for each category given the total corpora. The classification rule is defined as:

$$\text{choose} \begin{cases} C = 1 & \text{if } \log(S_1 / S_2) > b \\ C = 2 & \text{else} \end{cases}, \quad (7)$$

where b is a constant.⁴

To avoid mismatches caused by randomness, unclassification rules are also taken into account. As mentioned above, based on the chi-square selection algorithm, the keywords are labeled as two categories, positive indicator and negative indicator. Thus, we define a text as “unclassified” when either one of the following conditions is met: (a) no keywords are found in the text; (b) only one keyword is found in the text; (c) only two keywords are found in the text, and one is labeled as a positive indicator while the other as a negative indicator.

Example Application

Data

As part of a larger study exploring the relationship between life narratives and students’ personality adaption, 656 life stories were collected from 271 undergraduate students at Northwestern University, in the United States. The classification target was to label the life stories into four categories: redemption (RED), contamination (CON), redemption and contamination (BOTH), and neither redemption nor contamination (NEITHER). In the narrative research in the discipline of personality psychology, redemption and contamination are the two most important sequences for revealing the “change” tendency in people’s emotional well-being through writing (McAdams, 2008). In a redemption sequence, a demonstrably “bad” or emotionally negative event or circumstance leads to a happy outcome, whereas in a contamination scene, a good or positive event or state becomes bad or negative. Three experienced experts were invited to label each story based on McAdams’s manual coding system (McAdams, 2008). The Kappa agreement among the three human raters was 0.67.

The label for each story was defined as the decision made by at least two human raters, and was identified as the “standard” for the training process. According to the human raters’ assessment, 231 stories were labeled “change” (i.e., redemption or contamination or both), and 425 stories were labeled “no change” (i.e., neither redemption nor contamination).

Method

Given concerns about the common feature—“the change” tendency—in the redemption and contamination sequences, a two-stage classification framework was constructed. On the first stage, all the input was divided into two groups, “change” and “no change.” A further detailed classification was conducted at the second stage to categorize the preliminary results as redemption and contamination. To illustrate the application of the text classification models, we focused only on the first stage in the present study. The dataset was randomly split into a training set and a testing set, 70% and 30%, respectively. The “stop word list” and the Porter algorithm were used in the preprocessing to deduct the noninformative words and normalize the words into their common lexical forms. The robust classifiers were extracted by using the chi-square selection algorithm. Three machine learning models, DC, NB, and PSM, were applied for a comparative study.

Six performance metrics, accuracy, sensitivity (recall), specificity, positive predictive value (precision) (PPV), negative predict value (NPV), and F1 measure, were used to evaluate the efficiency of the three employed machine learning algorithms. A contingency table was used to perform calculations (see Table 3). All six indicators are defined in definitions (1) through (6), respectively. Accuracy, the main metric used in classification, is the percentage of correctly defined texts. Sensitivity and specificity measure the proportion of actual positives and actual negatives that are correctly identified, respectively. These two indicators do not depend on the prevalence (i.e., proportion of “change” and “no change” texts of the total) in the corpus, and hence are more indicative of real-world performance. The predictive values, PPV and NPV, are estimators of the confidence in predicting correct classification; that is, the higher predictive values, the more reliable the prediction would be. The F1 measure combines the precision and recall in one metric, which is often used in information retrieval to show classification efficiency. This measurement can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst value at 0.

Further, to check the stability of the three classification models, we explored all the metrics with an increasing number of word classifiers by adding 10 keywords, five from

positive classifiers (i.e., “change”) and five from negative classifiers (i.e., “no change”), each time. The number of keywords included in the textual assessment ranged from 10 to 2,600.

Table 3 Contingency Table for Calculating Classification Metrics

	True Standard	
	C_1	C_2
Assigned C_1	a	b
Assigned C_2	c	d

Note: a is a true positive value (TP), b is a false positive value (FP), c is a false negative value (FN), and d is a true negative value (TN)

$$\text{Accuracy} = \frac{a + d}{a + b + c + d} \quad 1$$

$$\text{Sensitivity} = \frac{a}{a + c} \quad 2$$

$$\text{Specificity} = \frac{d}{b + d} \quad 3$$

$$\text{Positive Predictive Value (PPV)} = \frac{a}{a + b}$$

$$\text{Negative Predictive Value (NPV)} = \frac{d}{c + d}$$

$$\text{F1-score} = \frac{2 \times \text{Sensitivity} \times \text{PPV}}{\text{Sensitivity} + \text{PPV}} \quad 6$$

Notes:

¹ The stemming algorithm is used to normalize lexical forms of words, which may generate stems without an authentic word meaning, such as “nightmar.”

² A body of texts is usually called a text corpus. The frequency of words within the text corpus can be interpreted in two ways: word token and word type. *Word token* is defined as individual occurrence of words, i.e., the repetition of words is considered, whereas *word type* is defined as the occurrence of different words, i.e., excluding repetition of words.

³ Since we are interested only in ranking the chi-square score for each word to find the optimal classifier, assessing the significance of the chi-square test is not important in this way.

⁴ In principle, the scope of threshold b could be set to be infinite. However, in practice, $(-5, +5)$ is recommended as *a priori* for b .

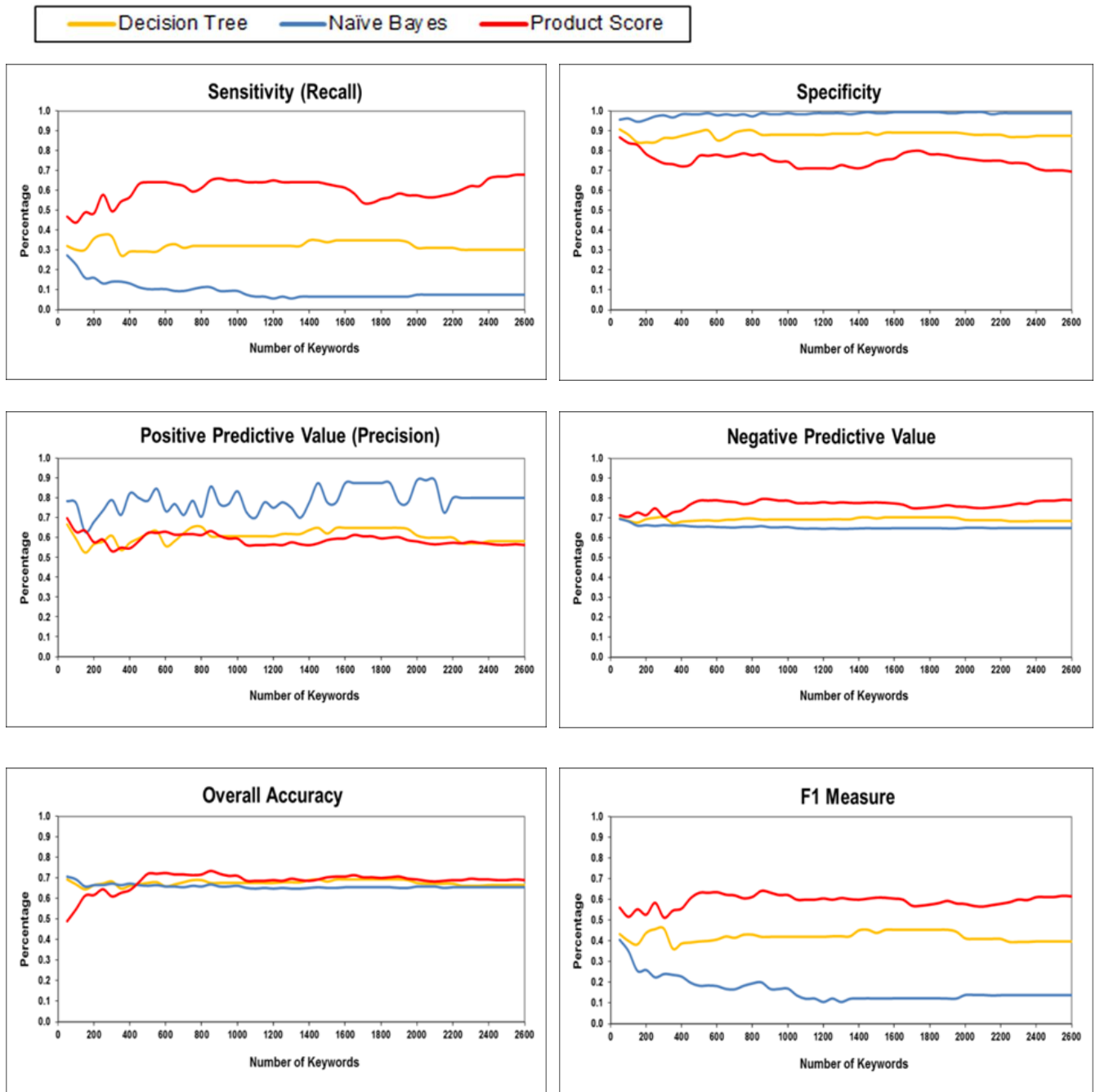
Results

Among the top 20 robust positive classifiers (i.e., keywords representing a “change” tendency), the expressions with negative semantics, e.g., “death,” “depress,” “scare,” “lost,” “anger,” “die,” “stop,” took a one-third proportion; whereas among the top 20 robust negative classifiers (i.e., keywords representing “no change” tendency), expressions with positive semantics, e.g., “peak,” “dance,” “high,” “promo,” “best,” “excite,” “senior,” accounted for the most, around 35%.

This result implies that people generally describe life in a happy way. The words with negative semantics would be informative for detecting the “change” tendency in the life stories.

The performances of three classification models are shown in Figure 2 with six metrics. Note that the three models resulted in a similar overall accuracy rate of around 70%, although the PSM was a bit superior to the other two, yet not robust. Further, the PSM ranked the highest in the F1 measure, which suggested that this model performed more efficiently than the DT and the NB in the text classification. In the sensitivity analysis, the NB yielded the highest specificity (more than 90%) but sacrificed too much in sensitivity (around 10%). The PSM performed worst on specificity (around 75%) but yielded the best result in sensitivity (around 60%). The PSM was more sensitive in detecting “change” life stories but a bit less capable of finding “no-change” stories than the other two models. However, among the three models, the PSM was the most balanced between sensitivity and specificity; that is, this model showed relatively satisfactory sensitivity without losing too much specificity. Another noticeable point was that the PSM showed the highest value in the NPV. This implies that we could have the most reliable prediction to deduct “no-change” life stories from the further stage by using the PSM rather than the DT and the NB. In the PPV plot, the NB curve ranked highest but it waved substantially with the increasing number of keywords, whereas the DT and the PSM remained stable throughout the whole processing.

The PSM and DT showed relatively low PPV values (around 60%), suggesting that the confidence for reliable prediction of “change” life stories was not that strong. However, since at this preliminary stage we targeted discarding the “no-change” life stories from further classification, PPV is less important than NPV in this sense.



Note: The horizontal axis indicates the number of keywords included in the textual assessment. The text analysis started with 10 keywords with the highest chi-square scores, i.e., five keywords labeled as positive classifiers and five keywords labeled as negative classifiers, and ended with 2600 keywords, i.e., 1300 keywords from either classifier label

Figure 2 Comparisons of text classification models, DT, NB and PSM based on the example application

Discussion

The example study demonstrated that the PSM is a promising machine learning algorithm for text (binary) classification. Although the three classification models showed a similar overall accuracy rate, the PSM performed the best in the F1 measure and remained stable as the number of keywords increased, implying better efficiency in text classification and more reliable agreement with the human raters' assessment than the other two standard models. Similar results were found in a recent study by He et al. (2012), where the PSM was validated in text classification for posttraumatic stress disorder (PTSD) patients' self-narratives regarding their stressful events and physical and mental symptoms. Analogous to the example application, the PSM successfully classified the self-narratives written by individuals with PTSD and non-PTSD in high agreement (82%) with the psychiatrists' diagnoses and presented stable results as the number of keywords increased.

Further, to help practitioners select an optimal algorithm for their own problems, the following pros and cons of each model can be considered and compared. The DT model is one of the most comprehensive models for visually tracking the path in classification. It is easily understood why a decision tree classifies an instance as belonging to a specific class. However, this model may result in low accuracy, especially for a small sample dataset. The DT uses splits based on a single feature at each internal node. Thus, many features are necessary to extract from the training set. Another frequent problem that may occur in applying DT algorithms is the overfitting. The most straightforward way of using them is to prune the tree by not allowing it to its full size (Kotsiantis, 2007) or establish a nontrivial termination criterion such as a threshold test for the feature quality metric (see more in Elomaa, 1999; Bruha, 2000).

The major advantages of NB are its short computational time for training and its simple form of a product with the assumption of independence among the features. Unfortunately, the assumption of independence among words is not always correct, and thus, the NB is usually less accurate than other more sophisticated learning algorithms. However, the NB is still a very effective model in classification. Domingos and Pazzani (1997) performed a large-scale comparison of the NB with state-of-the-art algorithms, e.g., DT, instance-based learning, and rule induction, on standard benchmark datasets, and found it to be sometimes superior to the other learning schemes, even on datasets with substantial feature dependencies.

Despite adopting the same assumption of word independence in the NB, the PSM has more flexibility in the model decision threshold. As shown in Formula (7), the decision threshold b could be set as an unfixed constant in practice. For instance, in a clinical setting such as the PTSD screening process, on one hand, psychiatrists may want to exclude people without PTSD from further tests, which needs a relatively higher specificity value.

On the other hand, when psychiatrists focus on treatment for patients with PTSD, a more sensitive result from the text analysis is probably required to detect potential patients as precisely as possible. With the example data in the current study, to yield satisfactory sensitivity in finding the “change” elements in life stories without sacrificing too much specificity, an optimal threshold of PSM log ratio score could be set at $b = -4$. However, since the PSM allocates a set of term weights for each key feature, more time and more storage space are expected in the training and validation process, which might reduce the PSM’s effectiveness in a large sample.

In addition to the applications of text classification within the field of psychology and psychiatry, the PSM is also expected to extend its usage in educational measurement. For instance, this model might be used as an alternative approach to classify students’ essays into different grade levels, to retrieve information about students’ noncognitive skills by analyzing their writing components, e.g., diaries, posts, blogs, and short messages, and further to extract patterns among students’ noncognitive skills and their academic grades.

In conclusion, the present study introduced the general procedure of text classification within the framework of text mining techniques and presented an alternative machine learning algorithm, the PSM, for text classification. In the comparative study with two standard models, DT and NB, the PSM was shown to be very promising in text (binary) classification. It might be interesting to extend the PSM into a generalized multiple classification algorithm in future work, and to find out whether and how educational measurement could benefit from this new procedure.

References

- Breiman, L. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth.
- Bruha, I. (2000). From machine learning to knowledge discovery: Survey of preprocessing and postprocessing. *Intelligent Data Analysis*, 4, 363–374.

- Burstein, J. (2003). The E-rater scoring engine: Automated essay scoring. In M. D. Shermis & J. C. Burstein (Eds.), *Automated essay scoring: A cross-disciplinary perspective* (pp. 113–121). Mahwah, NJ: Erlbaum.
- Conway, M. (2010). Mining a corpus of biographical texts using keywords. *Literary and Linguistic Computing*, 25(1), 23–35.
- Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2–3), 103–130.
- Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern classification*. New York, NY: Wiley.
- Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19, 61–74.
- Elomaa, T. (1999). The biases of decision tree pruning strategies. *Advances in Intelligent Data Analysis Proceedings*, 1642, 63–74.
- Feldman, R., & Sanger, J. (2007). *The text mining handbook: Advanced approaches in analyzing unstructured data*. Cambridge, England: Cambridge University Press.
- Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3, 1289–1305.
- Hand, D. J., & Yu, K. M. (2001). Idiot's Bayes - Not so stupid after all? *International Statistical Review*, 69(3), 385–398.
- He, Q., Veldkamp, B. P., & de Vries, T. (2012). Screening for posttraumatic stress disorder using verbal features in self-narratives: A text mining approach. *Psychiatry Research*. doi: 10.1016/j.psychres.2012.01.032
- Hunt, E. B., Marin, J., & Stone, P. J. (1966). *Experiments in induction*. New York, NY: Academic Press.
- Jurafsky, D., & Martin, J. H. (2009). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, NJ: Pearson Prentice Hall.
- Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. *Informatica*, 31, 249–268.
- Lewis, D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In C. Nedellec & C. Rouveirol (Eds.), *Machine learning: ECML-98, Proceedings from the 10th European Conference on Machine Learning, Chemnitz, Germany* (pp. 4–15). New York, NY: Springer.
- Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing*. Cambridge, MA: MIT Press.

- McAdams, D. P. (2008). Personal narratives and the life story. In O. P. John, R. W. Robins, & L. A. Pervin (Eds.), *Handbook of personality: Theory and research* (pp. 242–264). New York, NY: Guilford.
- Oakes, M., Gaizauskas, R., Fowkes, H., Jonsson, W. A. V., & Beaulieu, M. (2001). A method based on chi-square test for document classification. In: *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 440–441). New York, NY: ACM.
- Page, E. B. (2003). Project Essay Grade: PEG. In M. D. Shermis & J. C. Burstein (Eds.), *Automated essay scoring: A cross-disciplinary perspective* (pp. 43–54). Mahwah, NJ: Erlbaum.
- Porter, M. F. (1980). An Algorithm for Suffix Stripping. *Program-Automated Library and Information Systems*, 14(3), 130–137.
- Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco, CA: Morgan Kaufmann.
- Vapnik, V. N. (1998). *Statistical learning theory*. New York, NY: Wiley.