13 Text classification and Naïve Bayes

Thus far, this book has mainly discussed the process of ad hoc retrieval, where users have transient information needs that they try to address by posing one or more queries to a search engine. However, many users have ongoing information needs. For example, you might need to track developments in multicore computer chips. One way of doing this is to issue the query multi- core AND computer AND chip against an index of recent newswire articles each morning. In this and the following two chapters we examine the question: How can this repetitive task be automated? To this end, many systems sup STANDING QUERY port standing queries. A standing query is like any other query except that it is periodically executed on a collection to which new documents are incrementally added over time.

If your standing query is just multicore AND computer AND chip, you will tend to miss many relevant new articles which use other terms such as multicore processors. To achieve good recall, standing queries thus have to be refined over time and can gradually become quite complex. In this example, using a Boolean search engine with stemming, you might end up with a query like (multicore OR multi-core) AND (chip OR processor OR microprocessor).

To capture the generality and scope of the problem space to which standing queries belong, we now introduce the general notion of a classification problem. Given a set of classes, we seek to determine which class(es) a given object belongs to. In the example, the standing query serves to divide new newswire articles into the two classes: documents about multicore computer chips and documents not about multicore computer chips. We refer to this as two-class ROUTING classification. Classification using standing queries is also called routing or FILTERING filteringand will be discussed further in Section 15.3.1 (page 335).

A class need not be as narrowly focused as the standing query multicore computer chips. Often, a class is a more general subject area like China or coffee. Such more general classes are usually referred to as topics, and the classification task is then called text classification, text categorization, topic classification, or topic spotting. An example for China appears in Figure 13.1. Standing queries and topics differ in their degree of specificity, but the methods for

solving routing, filtering, and text classification are essentially the same. We

therefore include routing and filtering under the rubric of text classification

in this and the following chapters.

The notion of classification is very general and has many applications within

and beyond information retrieval (IR). For instance, in computer vision, a

classifier may be used to divide images into classes such as landscape, por-

trait, and neither. We focus here on examples from information retrieval such

as:

• Several of the preprocessing steps necessary for indexing as discussed in

Chapter 2: detecting a document's encoding (ASCII, Unicode UTF-8 etc; page 20); word segmentation (Is the white space between two letters a word boundary or not? page 24) ; truecasing (page 30); and identifying the language of a document (page 46).

• The automatic detection of spam pages (which then are not included in the search engine index).

• The automatic detection of sexually explicit content (which is included in search results only if the user turns an option such as SafeSearch off).

SENTIMENT DETECTION • Sentiment detection or the automatic classification of a movie or product review as positive or negative. An example application is a user searching for negative reviews before buying a camera to make sure it has no undesirable features or quality problems.

EMAIL SORTING • Personal email sorting. A user may have folders like talk announcements, electronic bills, email from family and friends, and so on, and may want a classifier to classify each incoming email and automatically move it to the appropriate folder. It is easier to find messages in sorted folders than in a very large inbox. The most common case of this application is a spam folder that holds all suspected spam messages.

VERTICAL SEARCH ENGINE • Topic-specific or vertical search. Vertical search engines restrict searches to a particular topic. For example, the query computer science on a vertical search engine for the topic China will return a list of Chinese computer science departments with higher precision and recall than the query computer science China on a general purpose search engine. This is because the vertical search engine does not include web pages in its index that contain the term china in a different sense (e.g., referring to a hard white ceramic), but does include relevant pages even if they do not explicitly mention the

term China.

• Finally, the ranking function in ad hoc information retrieval can also be based on a document classifier as we will explain in Section 15.4 (page 341).

This list shows the general importance of classification in IR. Most retrieval systems today contain multiple components that use some form of classifier.

The classification task we will use as an example in this book is text classification.

A computer is not essential for classification. Many classification tasks have traditionally been solved manually. Books in a library are assigned Library of Congress categories by a librarian. But manual classification is

expensive to scale. The multicore computer chips example illustrates one alternative approach: classification by the use of standing queries – which can

RULES IN TEXT be thought of as rules – most commonly written by hand. As in our exam-

CLASSIFICATION ple (multicore OR multi-core) AND (chip OR processor OR microprocessor), rules are

sometimes equivalent to Boolean expressions.

A rule captures a certain combination of keywords that indicates a class.

Hand-coded rules have good scaling properties, but creating and maintaining them over time is labor intensive. A technically skilled person (e.g., a

domain expert who is good at writing regular expressions) can create rule

sets that will rival or exceed the accuracy of the automatically generated classifiers we will discuss shortly; however, it can be hard to find someone with

this specialized skill.

Apart from manual classification and hand-crafted rules, there is a third

approach to text classification, namely, machine learning-based text classification. It is the approach that we focus on in the next several chapters. In

machine learning, the set of rules or, more generally, the decision criterion of the text classifier, is learned automatically from training data. This approach
STATISTICAL TEXT is also called statistical text classification if the learning method is statistical.

CLASSIFICATION In statistical text classification, we require a number of good example documents (or training documents) for each class. The need for manual classification is not eliminated because the training documents come from a person

LABELING who has labeled them – where labeling refers to the process of annotating each document with its class. But labeling is arguably an easier task than writing rules. Almost anybody can look at a document and decide whether or not it is related to China. Sometimes such labeling is already implicitly part of an existing workflow. For instance, you may go through the news

articles returned by a standing query each morning and give relevance feedback (cf. Chapter 9) by moving the relevant articles to a special folder like

multicore-processors.

We begin this chapter with a general introduction to the text classification problem including a formal definition (Section 13.1); we then cover Naive Bayes, a particularly simple and effective classification method (Sections 13.2–13.4). All of the classification algorithms we study represent documents in high-dimensional spaces. To improve the efficiency of these algorithms, it is generally desirable to reduce the dimensionality of these spaces; to this end, a technique known as feature selection is commonly applied in text classification as discussed in Section 13.5. Section 13.6 covers evaluation of text classification. In the following chapters, Chapters 14 and 15, we look at two other families of classification methods, vector space classifiers and support vector machines.

13.1 The text classification problem

In text classification, we are given a description $d \in X$ of a document, where

X is the document space; and a fixed set of classes $C = \{c1$ DOCUMENT SPACE , c2, . . . , cJ$\}$. Classes

CLASS are also called categories or labels. Typically, the document space X is some

type of high-dimensional space, and the classes are human defined for the

needs of an application, as in the examples China and documents that talk

TRAINING SET about multicore computer chips above. We are given a training set D of labeled

documents hd, ci,where hd, ci $\in X \times C$. For example:

hd, ci = hBeijing joins the World Trade Organization, Chinai

for the one-sentence document Beijing joins the World Trade Organization and

the class (or label) China.

LEARNING METHOD Using a learning method or learning algorithm, we then wish to learn a clas-

CLASSIFIER sifier or classification function γ that maps documents to classes:

(13.1) $\gamma : X \rightarrow C$

SUPERVISED LEARNING This type of learning is called supervised learning because a supervisor (the

human who defines the classes and labels training documents) serves as a

teacher directing the learning process. We denote the supervised learning

method by Γ and write $\Gamma(D) = \gamma$. The learning method Γ takes the training

set D as input and returns the learned classification function γ.

Most names for learning methods Γ are also used for classifiers γ. We

talk about the Naive Bayes (NB) learning method Γ when we say that “Naive

Bayes is robust,” meaning that it can be applied to many different learning

problems and is unlikely to produce classifiers that fail catastrophically. But

when we say that “Naive Bayes had an error rate of 20%,” we are describing

an experiment in which a particular NB classifier γ (which was produced by

the NB learning method) had a 20% error rate in an application.

Figure 13.1 shows an example of text classification from the Reuters-RCV1

collection, introduced in Section 4.2, page 69. There are six classes (UK, China, . . . , sports), each with three training documents. We show a few mnemonic words for each document's content. The training set provides some typical examples for each class, so that we can learn the classification function γ.

TEST SET Once we have learned γ, we can apply it to the test set (or test data), for example, the new document first private Chinese airline whose class is unknown. In Figure 13.1, the classification function assigns the new document to class
γ(d) = China, which is the correct assignment.

The classes in text classification often have some interesting structure such as the hierarchy in Figure 13.1. There are two instances each of region categories, industry categories, and subject area categories. A hierarchy can be an important aid in solving a classification problem; see Section 15.3.2 for further discussion. Until then, we will make the assumption in the text classification chapters that the classes form a set with no subset relationships between them.

Definition (13.1) stipulates that a document is a member of exactly one class. This is not the most appropriate model for the hierarchy in Figure 13.1. For instance, a document about the 2008 Olympics should be a member of two classes: the China class and the sports class. This type of classification problem is referred to as an any-of problem and we will return to it in Section 14.5 (page 306). For the time being, we only consider one-of problems where a document is a member of exactly one class.

Our goal in text classification is high accuracy on test data or new data – for example, the newswire articles that we will encounter tomorrow morning in the multicore chip example. It is easy to achieve high accuracy on the training set (e.g., we can simply memorize the labels). But high accuracy on

the training set in general does not mean that the classifier will work well on new data in an application. When we use the training set to learn a classifier

for test data, we make the assumption that training data and test data are

similar or from the same distribution. We defer a precise definition of this

notion to Section 14.6 (page 308).