

# Text Mining with Oracle Text

*An Oracle White Paper  
April 2005*

# Text Mining with Oracle Text

Introduction .....	3
Positioning.....	3
Supervised Classification .....	4
Rule Based Classification.....	4
Classification Training.....	4
Decision Trees .....	5
SVM.....	6
Clustering.....	6
Conclusion.....	7
Further Reading.....	7

## INTRODUCTION

Text mining is loosely defined as “data mining for text”. In our view text mining involves using natural language processing (NLP) and statistical/probabilistic techniques for finding nuggets and patterns in text collections. A nugget is a new, unknown item where a pattern is the occurrence of a known item.

Some applications of text mining are: classification, clustering, information extraction, and questioning/answering. In the rest of the paper we will describe the Oracle Text features in these contexts.

## POSITIONING

One of the biggest problems facing businesses today is “information overload”. Especially now, with almost every company linked to the Internet, we are awash with information.

Billions of emails flow around the world, and even internal company document archives are approaching the terabyte level.

How can we make use of all this information - sorting out the useful from the not so useful?

One way is to use keyword search engines but has its limitations. In many languages - and especially in English - the same word can have multiple, completely separate meanings. An Internet search for “river bank” will return references to the “Hudson River Bank & Trust Company”, the “Great River Bank and Trust” and many other hits which really have nothing to do with what I wanted to find.

The problem here is one of context - reading the above paragraph, most humans will realise the wrong type of “bank” is being referred to. A simple keyword search ignores the context and blindly returns just what was asked for.

An alternative is to have human knowledge engineers sort through the information, and classify the documents into hierarchies. That would allow me, as a searcher, to browse through the hierarchies looking for items we want. Depending on the type of “bank” we were looking for, we might start at a top-level category of “finance” or we might start at “nature”.

This sort of manual hierarchical classification is extremely effective for retrieval - but is enormously expensive in manpower.

What we need to do is to harness the speed of computers, but give them the power to look beyond simple keyword searching, in order to make use of the context within which the words exist.

Oracle Text provides features for supervised and unsupervised classification that can help you solve your categorization problems.

## **SUPERVISED CLASSIFICATION**

The classification task is very well known. The idea is to classify documents into a number of predefined categories. A document can be in one category, multiple categories, or not categories at all.

Here is a classic scenario. You are Head of Information Dissemination at Acme Finance, Inc. You have just arranged with a major newspaper to receive all their newspaper articles (and those of their 200 partners round the world) as an online feed, with new articles dropping in as soon as they're written.

Of course, you can have everyone list their interests and have an expert scan the incoming stories and send them to everyone on the list.

Assuming you can't justify dedicating one or more people to this task, you can insert all the stories into a database table, and use a free text engine (such as Oracle Text) to index them. Users can then perform their searches - or you could run all the searches and send the results to the interested users.

In order to get the results to users as fast as possible, you are going to have to run these searches very frequently, and you are also going to need to filter out all those hits that you have already sent. This is all pretty inefficient.

What you really want to do is to have users tell you what they are interested in, perhaps by registering a collection of queries, and then run each incoming document against the complete set of registered queries.

### **Rule Based Classification**

This is where Oracle Text's CTXRULE index type comes in. A set of queries, or "rules", can be collected in a table - together with additional information such as the name of the user who registered this rule. A special type of index is created on these rules. We can think of CTXRULE as the inverse of CONTEXT index type. Instead of creating an index on documents we create an index on the queries.

Then, for each document that is processed (such as the news stories in the scenario above), a search is made for rules that match this particular document.

### **Classification Training**

In the previous section we describe the classification process: For each category there are rules which define the category, and using those rules the application classifies documents into the appropriate categories.

The process that defines the rules can be manual or automatic. Manual rule generation means that a human expert will create the necessary rules. This is very accurate, but also time consuming and therefore doesn't scale for large collections and categories. Automatic rule generation makes use of Oracle Text's training features, which will statistically analyze document groups and generate CTXRULE-compatible rules. The user must supply a training set consisting of documents already known to belong to one or more categories.

The benefits of automatic rule generation are:

- No manual creation of rules is required.
- As the document set grows, new groups of documents can be used for further training.
- You are making use of existing knowledge to classify future documents.

Let's assume a routing application where the system receives incoming news articles that should be distributed based on certain categories. Users could spend some time perfecting their ideal queries, and then leave those queries to be run automatically on each incoming document.

However, this does require that users spend a significant amount of time perfecting these queries - and of course assumes that the users understand the syntax of CTXRULE queries. For many non-technical users, this may be asking too much.

What might be more useful is for the users to provide a set of example documents from the subject - or "category" - in which they are interested. A useful classification system should then be able to analyze this set of documents, and automatically generate a rule (or set of rules) that would identify future incoming documents in the same subject area.

Oracle Text can do this for you. It uses two methods to do this, named after their underlying algorithms, which are known as "Decision Trees" and "Support Vector Machine" (SVM).

### **Decision Trees**

A decision tree, in general, is a method of deciding between two (or more, but usually two) choices. In our case, the choices are "the document matches the training set" or "the document does not match the training set".

A decision tree has a set of attributes that may be tested. In our case, the attributes can be: words from the document; stems of words from the document (e.g. the stem of "running" is "run"); and themes from the document - if themes are supported for the language in use.

The learning algorithm in Oracle Text builds one or more decision trees for each category provided in the training set. These decision trees are then coded into queries suitable for use by a CTRRULE index (see the previous section).

The decision trees include the concept of confidence. Each rule that is generated is allocated a percentage value that represents the accuracy of the rule *given the current training set*. In trivial examples, this accuracy is almost always 100%, but this merely represents the limitations of the training set. Similarly, the rules generated from a trivial training set may seem to be less than what you might expect, but these are sufficient to distinguish the different categories given the current training set.

The advantage of the Decision Trees algorithm is that it can generate rules that are easily inspected and modified by a human. So this method makes sense when you want the computer to generate the bulk of the rules, but you want to “fine tune” them afterwards by editing the rule sets.

### **SVM**

The second method we can use for training purposes is known as Support Vector Machine (SVM) classification. SVM is a type of machine learning algorithm derived from statistical learning theory. A strong benefit of SVM is the ability to learn from a very small sample set.

The SVM algorithm is not easy to explain in non-technical terms, but is much used in the area of classification and pattern recognition. An excerpt from an academic paper explains it thus:

“Support Vector Machines non-linearly map their n-dimensional input space into a high dimensional feature space. In this high dimensional feature space a linear classifier is constructed”

### **CLUSTERING**

Clustering - as opposed to classification, is the unsupervised division of patterns into groups.

We can distinguish two main clustering techniques: hierarchical and partitional. The first produces a nested series of partitions and the second produces only one. *K*-Mean is an example of a partitional clustering algorithm that produces a single partition of the data set (sometimes called flat output). *K*-Mean is a trade-off between quality and scalability.

The *K*-Mean algorithm works in “feature space”, where each dimension is represented by one of the features (such as a word or theme) from the document set. It assigns *k* centers, or “prototypes”, one for each cluster. It then calculates the distance from each prototype in feature space for each document, and assigns the document to the cluster to which it is closest. It then calculates a new prototype for

each cluster based on the mean of each document assigned to it, and repeats the process.

The clusters can be used for building features like showing similar documents in the collection.

The benefits are:

- The automatic discovery of patterns in the collection.
- It is useful for identifying categories from the collection.
- It is useful for building abstractions (for example: More like this).
- It provides a statistical snapshot of the collection.

## CONCLUSION

Oracle Text offers a rich set of features for text mining. The ability to find, classify, cluster, and visualize documents - based on their textual, content metadata, or attributes - makes the Oracle database the single point of integration for all data management.

## FURTHER READING

1. Oracle Text Reference Guide. Oracle Corp., Redwood Shores, CA (2004).
2. Oracle Text Application Developer's Guide. Oracle Corp., Redwood Shores, CA (2004).
3. S. Alpha *et al.* "TREC-10 Statistical classification (adaptive and batch) and question-answering"  
(<http://trec.nist.gov/pubs/trec10/papers/orcltrec10.pdf>)
4. C. Liao *et al.* "Feature Preparation in Text Categorization"  
([http://www.oracle.com/technology/products/text/pdf/feature\\_preparation.pdf](http://www.oracle.com/technology/products/text/pdf/feature_preparation.pdf))
5. Oracle Text Home Page (<http://otn.oracle.com/products/text/>)



White Paper Title

January 2005

Authors: Omar Alonso and Roger Ford

Contributing Authors:

Oracle Corporation

World Headquarters

500 Oracle Parkway

Redwood Shores, CA 94065

U.S.A.

Worldwide Inquiries:

Phone: +1.650.506.7000

Fax: +1.650.506.7200

[www.oracle.com](http://www.oracle.com)

Oracle is a registered trademark of Oracle Corporation. Various product and service names referenced herein may be trademarks of Oracle Corporation. All other product and service names mentioned may be trademarks of their respective owners.

Copyright © 2004 Oracle Corporation

All rights reserved.