
# Automatic Categorization of Web Sites

by

*Lida Zhu*

Supervisors:
Morten Goodwin Olsen, Agata Sawicka and Mikael Snaprud

Master Thesis in
Information and Communication Technology

University of Agder

Grimstad, 26. May. 2008

Version 1.0

**Abstract:**

In this thesis we have presented a solution to classify websites into geographical attribute code (NUTS) and economical activities attribute codes(NACE). We propose a solution for web site classification with high accuracy. We use keyword-based document classification methods which had shown good performance. After classification, each document is assigned a class label from a set of predefined categories, which is based on a pool of pre-classified sample documents.

Our solution includes to remove stop words and skip html tags, which identify the informative term, remove the non-informative or redundant terms to improve the classification accuracy; use mutual information for feature selection to reduce the dimensional feature space and produce vectors for classification; finally, use Naïve Bayes and Decision Tree algorithm to perform the classification and also provide the performance comparison.

The system has shown great performance in the experiment. It classifies web sites into NACE categories with maximum accuracy of 97% performed on 46 web pages, while NUTS classification has best accuracy of 93% performed on 223 web pages.

# Table of Contents

## Illustration Index

## Equation Index



## Index of Tables

# 1 Introduction

Nowadays, life becomes much more convenient with the rapid development of Internet, all kinds of information can be found through the Internet. However, how to get the most relevant information in a faster manner is becoming a significant problem with the explosively growth of World Wide Web. There have been a lot of studies and research on the effective web information retrieval techniques, that includes data mining, clustering, and classification etc.

Traditional data mining is used to deal with structured data in database or data warehouses, that is for users to get information which suits their needs. It uses several effective classification algorithms like Naïve Bayes, Decision Tree and k-Nearest Neighbour, they have shown good quality and performance for classification work. However, the following characters of the world wide web are great challenges for data mining technologies. The size of the web is too huge; and the complexity of web pages is too difficult for traditional data mining technologies; also the web keeps constantly updating. Therefore, the traditional data mining techniques become inadequate. Users need a tool to reach the most relevant information and also to do “mining” through Internet. Thus, web mining has become increasingly popular domain in data mining.

In this project, we focus on the automatic classification of web documents as a application domain in web mining field. The purpose of this project, is to develop a application that perform automatic web sites classification for EIAO machinery. EIAO is European Internet Accessibility Observatory, it is established for large-scale assessment of websites accessibility. [1] In order to provide quality background material,

Classification of Economic Activities in the European Community (NACE) and The Nomenclature of Territorial Units for Statistics (NUTS) become two important specific subjects to describe specific web sites in EIAO's data warehouse. NACE refers to statistical classification of economic activities in European Union, while NUTS is the statistical classification of the geographical location in the regional level for EU countries. We mainly focus on NACE code in our work because NACE code is more complicated than NUTS, although they are built in the same structure. So we assume if the classification scheme can work effectively on NACE, then it would be applicable to NUTS. To achieve the goal, the main idea of classification is to pre-classify some web sites in the URL repository manually in NACE and NUTS, and learn information from classified data as training data to build a model using a classification scheme. Subsequently the system uses the model to classify new web sites and assign each one a NACE and NUTS code automatically.

Our work mainly focused on the classification scheme. [2] showed that keyword-based documents classification method can be used for Web document classification and has shown good results in Web page classification. Since keyword-based classification basically searches for “a set of associated, frequently occurring text pattern”, they suggested to use information retrieval to extract the keywords or terms first, and apply simple association analysis techniques such as feature selection.

With this idea, in this thesis we propose a strategy model that implementation in classifier. We also propose some information retrieval techniques and classification methods for the model and made a comparison matrix of performance based on our experiment data.

The thesis outline is given in the next section.

## 1.1 Report outline

Chapter 1 is the overview of this project, it also covers the outline of the whole report.

Chapter 2 describes the main problem of the project, we narrow the main problem down into several sub-problems. The best strategy is to develop a high accuracy classifier to automatic categorisation web sites into NUTS/NACE.

Chapter 3 provides the theory and literature background information, the advantages and disadvantages of methods, and identify the suitable algorithm of classification.

Chapter 4 describes the proposed model of the project, illustrates implementation and shows the result of experiment.

Chapter 5 evaluates the project. We analyse and compare the result of experiment and discuss their performances with recall and precise value.

Chapter 6 outlines the conclusion of the project. We also provide a possible further work for the future.

In the end, we attached the list of reference for this thesis.

# 2 Problem description

In order to classify in NACE and NUTS, this project developed a solution strategy of automatic web sites categorisation for EIAO machinery. European Internet Accessibility Observatory (EIAO) is established for large-scale assessment of web site accessibility. It mainly consists of three elements: Internet robot, Web accessibility Metrics and Data Warehouse. When it starts crawling, the crawlers begin to download the web pages from the Internet. Then, evaluates web pages using the web accessibility metrics. In the end, store the underlying detailed results in the data warehouse and display the result on the Web interface.

In order to provide quantitative background material, NUTS/NACE becomes the official criterion choice for accessibility benchmarking on the European level to identify web sites. The NUTS code corresponds to the geographical location of the organisation behind a website in the European Union, and the NACE code describes the business field of the organisation behind the website for European Community .

Currently, EIAO uses manual classification for only a small number of web sites in NACE and NUTS code. Besides, the EIAO project plans to evaluate 10,000 web sites. So it is very desirable to develop classifier which can automatic categorisation web sites.

In order to make evaluation and comparison of results valuable and informative of web-site-level, it essentially requires the results to be of high accuracy. Since only a small number of web sites are currently categorised, it also demands the classifier with ability of predictions. Therefore, the

classifier should be able to classify a large number of web sites with high accuracy in NACE and NUTS.

## 2.1 Sub-problems with strategies

1. **Classify web sites into NACE.**
   Classify the web sites into NACE classes according to the manual classification list from URL repository of EIAO.
2. **Classify web sites into NUTS.**
   Organize the web sites into directories based on the manual NUTS classification index from EIAO.
3. **How to deal with large, complex form, data set.**
   Preprocessing the website. Identify the informative and non-informative attributes in the web page documents and remove those the non-informative and even redundant attributes such as <html> and so on.
4. **How to lower down the high-dimensional vector space.**
   Extracting features. Select features only with valuable information, in order to reduce the high-dimensional features space.
5. **Determine the most appropriate classifier for classification task.**
   According to the requirements of classifier, we made a comparison and evaluation on classifiers based on accuracy, robustness, scalability and speed.

# 3 Background

In this chapter, we provide background information from literatures to describe the procedures to perform classification.

In Section 3.1 we introduce the standard we use for classification NACE and NUTS.

In Section 3.2 we describes pre-processing steps to prepare the data for classification.

In Section 3.3 we present the classification methods to use to complete classification task.

## 3.1 NACE & NUTS categories for classification of URLs for the European Internet Accessibility Observatory

In process of classification, we need to pre-classify data to build a model, then use the model to perform classification. Therefore, we need to find a suitable statistical classification standard to pre-classify data. Since EIAO wants to provide a quality information about each web site, they use NACE and NUTS as two subjects for describing. Hence, we follow the definition by NACE and NUTS on classification. In this section, we introduce the NACE and NUTS classification about their usages and structures.

### 3.1.1 NACE

NACE is a statistical classification of economic activities used within European Community. It was established as a industry classification standard in order to ensure the comparability between national and community. It specifies a large range of economic activities in statistical economic application domains with a 6-digit code. The following figure 1 shows the structure of the statistical classification system.

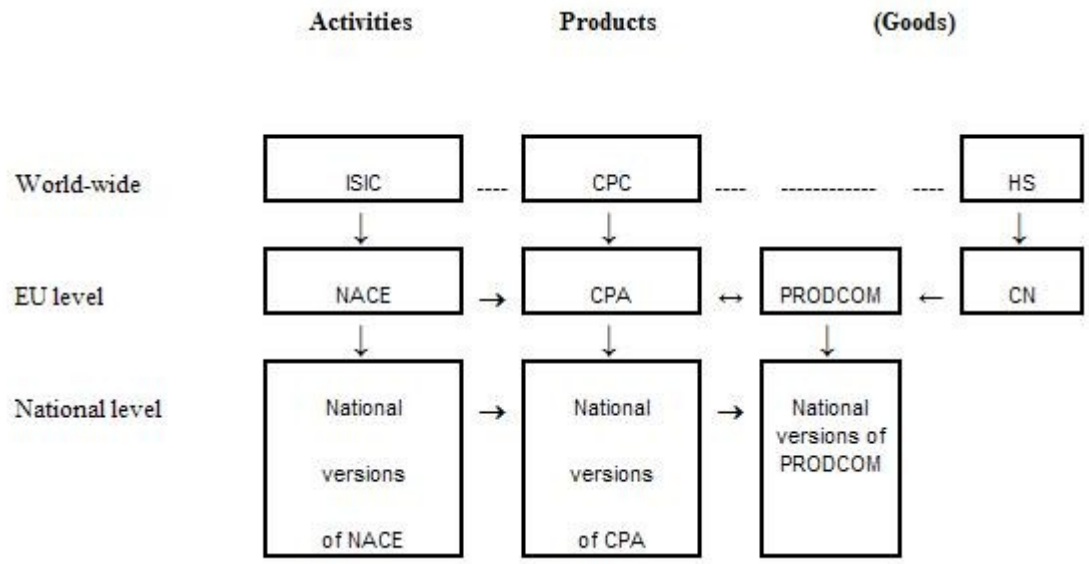


*Figure 1: Integrated system of statistical classification [4]*

As we can see in the figure 1, NACE refers to ISIC which is International Standard Industrial Classification used world wide including in EU and National level. It shares the index of highest level with ISIC and National versions of NACE which also ensure international comparability. Comparing to ISIC, it contains much more details in the lower level.

In structure, it is designed in hierarchical classification, which makes it possible to map the dataset in directories into NACE, and perform classification and prediction. It contains four levels. The first level have 21 sections identified by an alphabetical code from A to Q in different economic fields but doesn’t appears in NACE code because it is not specific in activities. The lower levels describes detailed activities identified in digital to consist NACE code. For more information, refer to [3], [4]

### 3.1.2 NUTS

The Nomenclature of Territorial Units for Statistics (NUTS) is a statistical standard classification at a regional level for EU members and EFTA countries in geography. It is used to provide a specific classification of territorial units for statistical purpose in European Union.

NUTS is also hierarchical structured classification. It consists of three levels and begins with two-letter code represented the EU countries. The first level is identified each EU state regions. And the lower level is the divisional regions of upper level that identified in digital code and so on. Note that when the number of regions is more than 9 in each level, NUTS uses capital letters for numeration. In addition, there are two levels of local administrative units (LAUs) defined below the three levels in NUTS for big size countries. Whereas not all countries need to use every level, it depends on the size.

| Code | Country | Level 1 | Level 2 | Level 3 |
|---|---|---|---|---|
| UK | UNITED KINGDOM | | | |
| ... | ... | ... | ... | ... |
| UKI | | LONDON | | |
| UKI1 | | | Inner London | |
| UKI11 | | | | Inner London – West |
| UKI12 | | | | Inner London – East |
| UKI2 | | | Outer London | |
| UKI21 | | | | Outer London - East and North East |
| UKI22 | | | | Outer London – South |
| UKI23 | | | | Outer London - West and North West |
| ... | ... | ... | ... | ... |

*Figure 2: Example of NUTS categories in UK [0]*

As shown in the Figure 3.2, United Kingdom is divided in regions as London (UKI) in the NUTS 1, and London is divided in Inner London (UKI1) and Outer London (UKI2) in level 2 (NUT2). It provides NUTS code for each level.

Considering the desirable qualities of NACE and NUTS that make them widely used in the world, we decide to use them both as the criteria of classification to form the web sites in data collection. In this project, since NACE and NUTS are similar, the classification task would also be similar. However, according to our data collection, we have more directories in NACE than NUTS, since we use English as the keywords of natural language processing when classification. That is, there are a lot more categories in

NACE in English websites than only two classes (UK and Ireland) in NUTS. Hence, we decided to focus more to solution on NACE in our work.

## 3.2 Features selection

### 3.2.1 Introduction

Nowadays, there are various forms data, and huge sizes of text noise exists in the Internet which makes classification very difficult. It brings great trouble to analysis data. When we analysis electronic documents, like web page documents, incomplete and complex form are common properties of data. The attributes in the data could be irrelevant or redundant for the classification. Therefore, it is necessary to pre-process data which attempt to identify attributes that contain irrelevant information to classification process and to exclude the non-informative attributes before we perform classification. The preprocessing is used to improve the qualities of data in order to improve the accuracy, scalability and efficiency of classification by removing the irrelevant and redundant data without decrease the accuracy of classification. “Ideally, the time spent on relevance analysis as preprocessing, when added to the time spent on learning from the resulting ‘reduced’ feature subset, should be less than the time that would have been spent on learning from the original set of features.” [2] Hence, let us introduce the steps in preprocessing.

In order to analysis the unstructured and incomplete data in the documents, let’s introduce here a text indexing methods, “Information Retrieval techniques, which had been developed to handle unstructured textual documents”[2]. A typical techniques to handle documents of Information Retrieval is to search relevant documents based on keywords. That is, a classification of dividing the set of documents in based on some certain terms. It is desirable to use information retrieval techniques to analysing and extracting useful information from the data.

When it comes to deal with the “reduced feature subset”, it is a very big problem with the high dimensionality of the feature space. “The native feature space consists of the unique terms (words or phrases) that occur in documents, which can be tens or hundreds of thousands of terms for even a moderate-size test collection”.[5] As we know, attributes is the way the terms be analysed. The high dimensionality would be the main cause of extremely expensive computation. Therefore, it is desirable to “select relevant features from feature space for building robust learning models”[6], which called feature selection in machine learning, to reduce the dimensionality and improve the efficiency of classification. We will introduce several automatic feature selection methods such as no manual definition involved.

In this section, we will describes the methods of information retrieval in section 3.2.2. Section 3.2.3 introduces some basic measurement and concept structure in information retrieval. Section 3.2.4 provides several the methods of feature selection. Section 3.2.5 gives an evaluation and comparison of feature selection methods.

### 3.2.2 The methods of Information Retrieval

#### Stop word

Stop words are set of words that are non-informative terms although may appear frequently, such as “a, the, of, for, with”, and so on. “Stop words may vary when the set of documents are vary”. For example, “database system” could be a important word in newspaper while it could be a stop word in the set of research paper about a database system conference.[2]. That's the main reason why stop words are language

dependent. We should notice that in implementation, especially when NUTS classification is performed. Hence, Removing stop words will save spaces for storing document contents and improve efficiency and accuracy of classification.

**Stemming**

A group of different words may share the same root such as past tense and plural and singular usage. Stemming is an algorithm developed to reduce words to its stem. For example, words “connection”, “connecting”, “connected” and “connections” can be viewed as different occurrences of the same root word “connect”. So that to reduce the unnecessary space for storing and increase the keywords frequency. Therefore, we could expect a higher accuracy by stemming. However, we didn't implement it in our solution due to the application we chose doesn't include this method. But we could use it to improve the classification in the future work. [2]

### 3.2.3 Basic measures of Information Retrieval

There are two basic measures for the effectiveness of processing information retrieval:

- *Precision is the percentage of retrieved documents that are in fact relevant to the query.*
- *Recall is the percentage of documents that are relevant to the query and were retrieved.*
- $$Recall = \frac{Relevant \wedge Retrieved}{Relevant}$$

*Equation 1: Formula of Recall*

•

$$Precision = \frac{Relevant \wedge Retrieved}{Retrieved}$$

*Equation 2: Formula of Precision*

[2]

In Figure 3 below the four possible query results available.

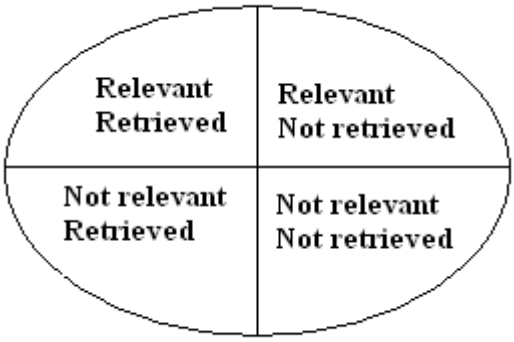


*Figure 3: four possible query results applied classification.[7]*

Example:

*"Suppose 100 students are to be classified based on height. Actuality, there are 30 tall students and 70 not tall. A classification technique classifies 65 students as tall and 35 are not tall."[7]*

The following figure 4 illustrate the classification result mapped into classification figure 3 above. According to the formula, the precision of tall students is tall students that are classified as tall divided by all the students that are classified as tall. That is, 20/(20+45). While the recall of tall students is tall students that are classified as tall divided by all the tall students. That is, 20/(20+10). So as the students are not tall. The precision is 25/(10+25) and the recall is 25/(25+45).

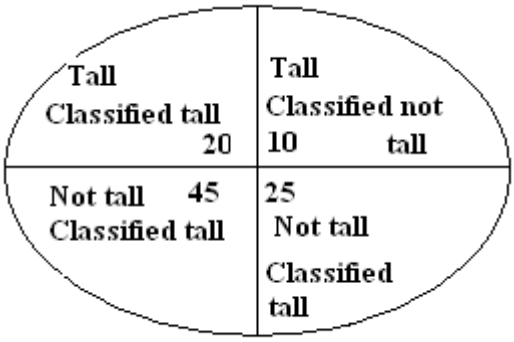


*Figure 4: Precision and Recall applied to classification[7]*

**Concept Hierarchies**

"Concept hierarchies are often used to show the relationship between related keywords to documents."[7]. Figure 5 below illustrates a concept hierarchy that shows the relationship among the document about cat categories.

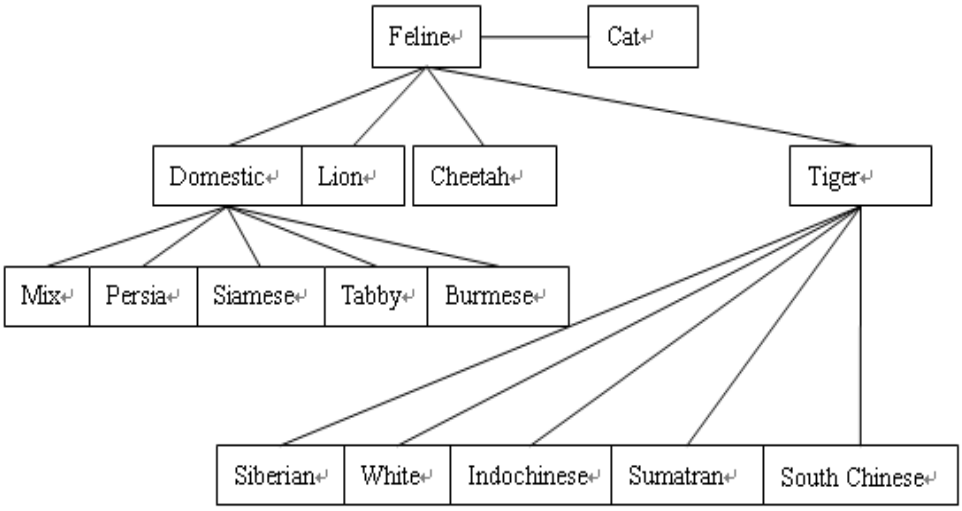


*Figure 5: Example of concept hierarchical structure for cat category[7]*

The figure 5 shows a hierarchical structure of the documents about the cat. Typically, web sites have similar structure. For example, some web sites consist of web pages with similar subject, while other web pages have similar topic and connect with each other but belongs to different web sites. It is desirable to map the web page documents into hierarchical structure based on their relationship such as Hyperlinks. Note that NACE and NUTS are also hierarchical structured, it is possible to map the web pages in NACE and NUTS then perform classification task. It a typical associated-base classification due to it considers mainly about the relationship among the terms and documents, unlike keyword-based classification. It is more desirable to perform the classification task in the hierarchical structure using keyword-base methods than pure keyword classification. However, since it is still a new topic in resent research, we couldn't be able to implement it in our experiment yet. But it will be a very interesting area in the future work.

### 3.2.4 Methods for Feature Selection

**Document frequency thresholding (DF)**

Document frequencies are the number of documents in which vectors appear in the class. DF threshold removes the documents which have less value than threshold.[5]

It is the simplest method in keyword algorithms, and does not lead to expensive or complex computation. However, DF terms with value less than threshold may be relatively informative and should not be removed. While, the terms with high DF value may be non informative, like “a”. As we introduced in last section, stop word and stemming could theoretically improve the efficiency of this method. Its advantage is small computation. And it shows a good result in practise.

**Information gain (IG)**

“Information Gain measures the number of bits of information obtained for category prediction by knowing the presence or absence of a term in a document” [5] That is, it measures how much additional information gained from each feature and remove those attributes whose information gain is less than a certain threshold. The definition of information gain for term t is the following formula:

$$IG(t) = P(t)\sum_{i=1}^{M} P(C_i \mid t)\log\frac{P(C_i \mid t)}{P(C_i)} + P(\bar{t})\sum_{i=1}^{M} P(C_i \mid \bar{t})\log\frac{P(C_i \mid \bar{t})}{P(C_i)}$$

*Equation 3: Formula of Information Gain*

where P(t) is the probability of t, P(Ci) is the probability of Ci; P(Ci|t) means the probability of Ci in the condition of term t appears in the document. The computation involves the conditional probabilities and the determination of suitable threshold. Hence, we could expect a higher accuracy and lower speed than Document Frequency in classification.

**Mutual Information (MI)**

Mutual Information measures the associativity between terms and categories. The mutual information criterion is estimated as :

$$I(t,c) \approx \log\left(\frac{A \times N}{(A+C) \times (A+B)}\right)$$

*Equation 4: Formula of Mutual Information[5]*

where t represent a term and c is a category. A is the number of times when t and c both appear. C is the frequency of c occur without t, and B is the frequency of t occur without c. N is the total number of documents in the category.

In our experiment, Mallet is implemented with MI which shows a quite good result in classification.

### 3.2.5 Comparison

Several feature selection techniques have been tried in recent years. However, thorough evaluations are rarely carried out for large text

categorization problems. It is partly due to the fact that many learning algorithms do not scale to a high-dimensional feature space.

[5] shows that IG has most effective in term removal without sacrificing accuracy. The performance of DF is found comparable to IG. And mutual information is found has poor performance with KNN. Note that for other algorithms than KNN, the impact is not known. The following table 1 provides us a comparison matrix for feature selection methods.

| Method | Document Frequency (DF) | Information Gain (IG) | Mutual Information (MI) |
|---|---|---|---|
| favouring common terms | Y | Y | N |
| using categories | N | Y | Y |
| Accuracy in KNN | excellent | excellent | poor |

*Table 1: A comparison for feature selection methods [5]*

From the table 1, we notice that the DF and IG all have excellent performance in favouring common terms. It indicates that common terms are informative for text classification even up to 90%. This further indicates that removing stopwords and stemming may only improve the classification task about 10%. When we deal with very large dataset, DF could dedicate a great help to handle with problems which are intractable in real-world life. A weakness of MI is that it is very sensitive to “marginal probabilities of terms” which leads to its poor performance.[5] Theoretically, [5] also shows that IG is the average mutual information, which makes it possible to change the feature selection methods MI used in Mallet for IG in the future.

## 3.3 Web classification algorithms

### 3.3.1 Introduction

After described the prepare work of dataset, we introduce the core part of classification system, the classifiers. Many classification methods have been proposed in machine learning, statistics and so on. In resent years, several classifiers were developed with scalable classification method in data mining which are capable of handling large set of data.

In classifier, the data was processed in two steps. In the first step, a model is built in the form of classification rules based on the available class-labelled data, known as training data. The classifiers learn from the training data by analysing in the measurement of attributes in order to construct a model.

In the second step, the model is used for classifying the testing dataset. Since the training data may lead the estimation to over-fitting, so the accuracy of the model is estimated based on the training data set. If the accuracy is acceptable, then we could trust the classifiers with new data.

The following figure 6 shows the procedures of the classification scheme in detail. They involves the steps we discussed before such as preprocessing and feature selection.

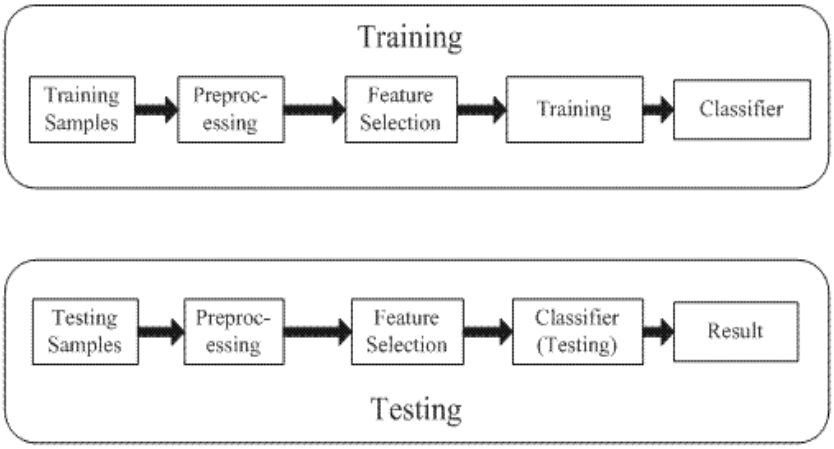


*Figure 6: Model of Classification*

In the figure 6, it clearly illustrates the classification scheme procedure combined with preprocessing and feature selection techniques. They are used to extract features from both training and testing data and provide vectors for classifiers to build model or to be classified. With the help of these steps, the performance of classifiers could be more reliable.

Evaluations and comparisons for classification method usually basic on the following issues: accuracy, speed, robustness, over-fitting and scalability and so on. Speed involved in training time and testing time during the classification. Robustness is the ability of the classifier method to make the correct decision basic on the data without enough information or noisy data such as Over-fitting. Over-fitting means the training data sometimes may contain noise which does not fit the model. Scalability refers to the ability of classifier to build the model effective when it applied to large database. [2]

In this section, we introduce the basic techniques for data classification such as Naive Bayes in section 3.3.2, Decision Tree in section 3.3.3. And other approaches to classification like k-Nearest Neighbor and Support Vector Machine are also introduced in section 3.3.4 and 3.3.5. Not only we

will describe the basic idea of how the classifiers works, but also include the advantages and disadvantages estimation will be given as characteristics. An evaluation and comparison of classification methods with multiple measures is also shown in section 3.3.6.

### 3.3.2 Naïve Bayes Classifier(NB Classifier)

Naïve Bayes is a typical statistical classifier. They have “exhibited high accuracy and speed when applied to large database”[2]. Before we introduce its algorithm, some definitions need to be explained first. Its algorithm is based on Bayes theorem.

**Bayes theorem**

*“Let X and Y be a pair of random variables.*

- *The **joint probability** refers to the probability that variable X will take on the value x and variable Y will take on the value y as P(X=x, Y=y)*
- *A **conditional probability** is the probability that a random variable will take on a particular value given that the outcome for another random variable is known. P(Y=y|X=x)*
- *This conditional probability P(Y|X) is also known as the **posterior probability** for Y, as opposed to its **prior probability**, P(Y)*
- *The relationship between joint and conditional probabilities for X and Y shows in the following way:*

$$P(X,Y)=P(Y|X)\times P(X)=P(X|Y)\times R(Y)$$ .

- *Bayes theorem should be:*

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}.$$

*Equation 5: Bayes theorem " [8]*

**Class conditional independence**

Naive Bayes classifier assumes that "the effect of an attribute value on a given class is independent of the value s of the other attributes".[2] That is used to simplify the computation. And it is also the main reason of inaccuracy when Naive Bayes classifier is used in real world.

Let's use a example to illustrate how it works:

*Suppose that we have a database of customers on the mailing list. The database describes attributes of the customers, such as their name, age, income, occupation, and credit rating. The question is how to find out whether or not they are likely to purchase a new computer, when new customers are added. The unknown sample we wish to classify is*

*X = (age = "<=30", income = "medium", student = "yes", credit_rating = "fair").[2]*

| RID | age | income | student | credit_rating | Class: buys_computer |
|---|---|---|---|---|---|
| 1 | youth | high | no | fair | no |
| 2 | youth | high | no | excellent | no |
| 3 | middle_aged | high | no | fair | yes |
| 4 | senior | medium | no | fair | yes |
| 5 | senior | low | yes | fair | yes |
| 6 | senior | low | yes | excellent | no |
| 7 | middle_aged | low | yes | excellent | yes |
| 8 | youth | medium | no | fair | no |
| 9 | youth | low | yes | fair | yes |
| 10 | senior | medium | yes | fair | yes |
| 11 | youth | medium | yes | excellent | yes |
| 12 | middle_aged | medium | no | excellent | yes |
| 13 | middle_aged | high | yes | fair | yes |
| 14 | senior | medium | no | excellent | no |

*Figure 7: Training data from customer database [2]*

1. As shown in Figure 7, the class label is buys_computer with two value {yes, no}. The questions becomes which class has the higher probability in the condition of the customer is a medium income student youth with fair credit rating. That is, to calculate the probabilities of *P(buys_computer = "yes"|X)* and *P(buys_computer = "no"|X)* and choose class of the higher probability. Let $C_1$ represents the class " buys_computer = 'yes' " and $C_2$ represents the class "buys_computer = 'no' ".

2. According to the Bayes theorem, we could calculate the following probabilities instead:

$$P(C_1|X) >< P(C_2|X)$$

that is: $P(X|C_1)P(C_1)/P(X) >< P(X|C_2)P(C_2)/P(X)$

A "><" B means A may more than B or it could be less than B, this inequation is our classifier. Note *P(X)* is the same in both problems, so we only need to compute

$$P(X|C_1)P(C_1) \quad >< \quad P(X|C_2)P(C_2)$$

3. We can easily compute from the data that the prior probabilities are:

$$P(C_1) = 9/14 ; \quad P(C_2) = 5/14$$

4. Then we have only to examine the value of $P(X|C_1)$ and $P(X|C_2)$ . Now we can use the class conditional independent to reduce the expensive computation. It is very clear to see that we can calculate the following probabilities instead:

$$P(X|C_1)=\prod_{k=1}^{n} P(x_k|C_1); P(X|C_2)=\prod_{k=1}^{n} P(x_k|C_2)$$

And the $P(x_k|C_1)$ ... can be estimated by the training sample. Finally, we can use " $P(X|C_i)PC_i>P(X|C_j)PC_j 1\le j\le 2, i\ne j$ " as a classifier to classify new unlabelled data

**Characteristics of Naive Bayes Classifiers:**

- **ACCURACY:** Naïve Bayes has best accuracy in theory but less accuracy in practice mainly because of the class independent assumption.
- **SPEED:** fast speed because of the class independence assumption.
- **ROBUSTNESS:** "strong robust to isolated noise and irrelevant attributes"[8]. Because the a attributes are estimated as average when are computed the conditional probabilities.

**Incremental algorithm**

Incremental algorithm is used to handle large amount of data which classify the new arrival dataset without having to mine the whole data. It classifies the updated dataset based on the training data, and uses the classified dataset to generate new classification rules to modify the old rules and improve the classifier. When it implemented with Naïve Bayes for classifier, it is known as Incremental Naïve Bayes. It is very desirable to use incremental algorithm to handle large database, especially when some data processing cost high because of complexity computation and so on. Hence, it is very useful in real-world data processing.

### 3.3.3 Decision Tree (DT)

Decision Tree uses attributes measurements to split the problems into different subset to build a tree structure. The Figure 8 below is a typical example of decision tree. It predicts whether or not a customer is likely to purchase a computer.

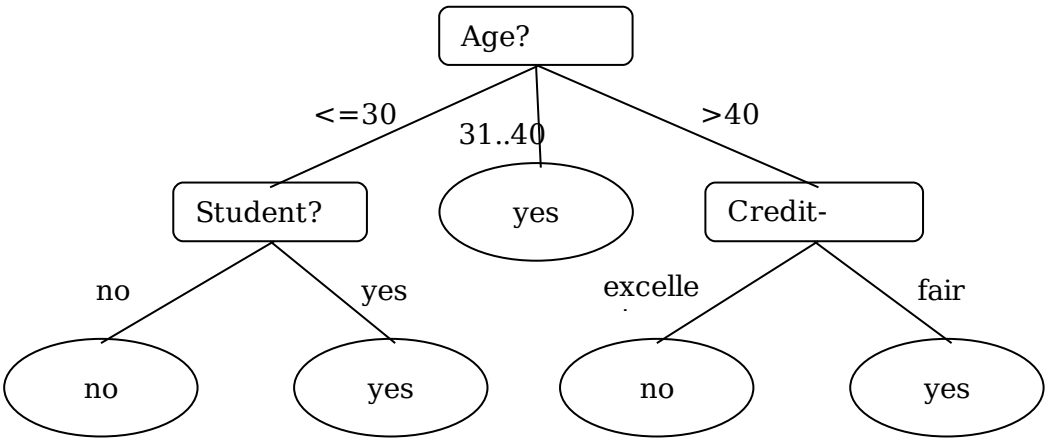


*Figure 8: An Example of Decision Tree about purchase computer*

It is the same example as in Naïve Bayes classifier. We use the attributes from the training data to construct the decision tree. And for a unknown given sample X, we could estimate its class label using this tree.
*X = (age = "<=30", income = "medium", student = "yes", credit_rating = "fair")*

The prediction of the sample's class is made by tracing from the root to a leaf node. According to the tree, we start at root node: {Age} => {<=30} => {Student} => {yes}. then we get the classified class label is "yes".

In a word, there are two basic steps in the technique:

- constructing the tree from the given data
- applying the tree to the unknown data with a categories label.

**Characteristics of Decision Tree**

- **ACCURACY:** Decision Tree is very efficient, it performs with high accuracy.
- **SPEED:** Decision Tree don't have a complex algorithm, so that the computation is not expensive. Therefore, it has high speed with no doubt.
- **ROBUSTNESS:** Decision Tree has difficulties in handling missing data and over-fit data. It is hard to identify correct and incorrect branches. Tree pruning algorithm could overcome this problem.
- **SCALABILITY:** When it applied to handle very large amount of data, the efficiency and scalability could become problems due to its restriction of training data's location should be in memory.

### 3.3.4 k-Nearest Neighbor Classifiers

k-Nearest Neighbor classifier memorizes the entire training data and performs classification when it find relatively similar attributes from the training set for the unknown sample. These examples, which are considered as nearest neighbors, can be used for determination of the class label of the test sample. It computes the unknown sample for its proximity to the other data points in the training set. The proximity is defined in terms of "Euclidean distance" between two points, $X=(x_1, x_2, \ldots, x_n)$ and $Y=(y_1, y_2, \ldots, y_n)$ is:

$$d(X,Y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

*Equation 6: Euclidean distance [2]*

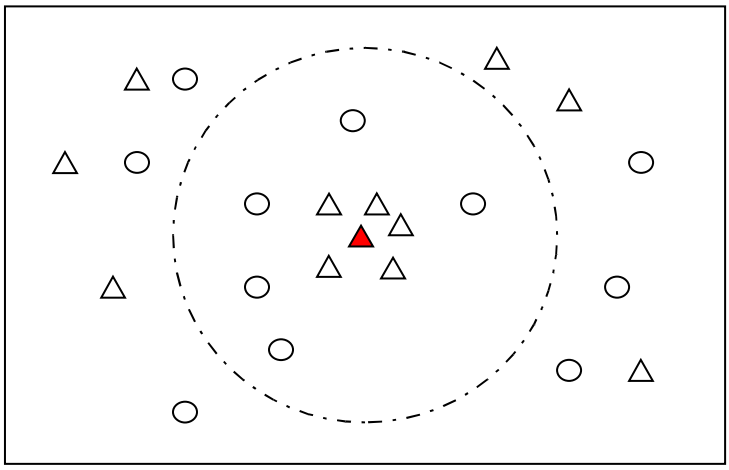

*Figure 9: k-Nearest Neighbor classification with large k [8]*

Above figure 9 shows a example of Nearest Neighbor classification. There are a unknown data surrounded by several examples that belong to two different classes, represented as triangles and circles. The red one is the test sample, and the big circle around it is represented its neighorhood. It is clear that $k = 5$ and the test sample should be assigned in triangle.[8],[2]

**Characteristics of Nearest-Neighbor Classifiers:**

- **SPEED:** K-Nearest Neighbor classifier has no time for training, and yet its speed of classification could be very slow when dealing with large amount of data.
- **ROBUSTNESS:** k-Nearest Neighbor could be weak to noise mainly due to it only depend on k, especially with a small value.

- **HARDWARE REQUIREMENTS:** Since k-Nearest Neighbor only store the training data without modelling, its requirements of hardware is much higher than the other classifiers. Else it will influence its accuracy and efficiency.

### 3.3.5 Support Vector Machine

Support Vector Machines is an algorithm with widely usage ranging from the classification of both linear and non-linear data. “It transforms the original data in a higher dimension, from where it can find a hyperplane for separation of the data using essential training tuples as support vectors.” [2]

Let's use two linear classes separation for example.

It is obvious that several possible hyperplanes available to divide the training samples into two classes in a linearly separated way as shown in the following figure 10. SVM searches for the **Maximum Margin Hyperplane** as the best hyperplane because large margin could be more accurate for classifying the new tuples than small one. The distance between these two hyperplanes is known as the margin of the classifier. Support Vectors are the training tuples fall on the maximum margin hyperplane.

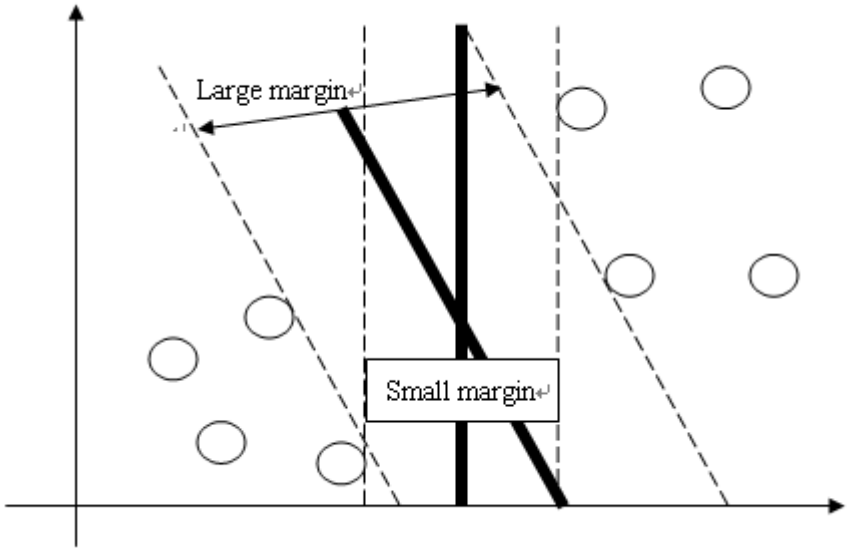


*Figure 10: Hyperplanes for linearly separable dataset*

Above figure 10 are the two hyperplanes to separate two different class.

It maximize the distances between categories by finding the optional classification hyperplanes.

The basic idea of Linear SVM is that searches for a hyperplane with the largest margin for building a model form the training data and performs classification mapping with the decision boundary according to the hyperplane.

**Characteristics of SVM**

- **ACCURACY:** SVM has shown promising results with outstanding accuracy among other classification methods.
- **SPEED:** Considering its computation expensive, even the fastest SVM could suffer from extremely slow speed.
- **ROBUSTNESS:** SVM is quite sensitive to noise due to it only depend on few support vectors. However, it has strong ability to prevent over-fit data.
- **SCALABILITY:** When it applied in large amount of data, its efficiency and scalability would face a great challenge due to its complexity and computation expensive.

### 3.3.6 Comparison of text categorization algorithms

Since different classifiers have different characteristics, we need to make analysis a comparisons on these methods to find out the most suitable classifier for our classification task. As introduced before, we mainly consider the subjects of accuracy, speed, robustness, and scalability as main measurements. Therefore we provide the evaluation and comparison matrix as below.

| | Accuracy | speed | robust-ness | Scal-ability | hardware requirements | require sample | Sum |
|---|---|---|---|---|---|---|---|
| Decision Tree | 9 | 9 | 7 | 5 | 8 | 6 | 44 |
| Naive Bayes | 8 | 9 | 9 | 9 | 9 | 7 | 51 |
| SVM | 9.5 | 2 | 8 | 7 | 8 | 8 | 42.5 |
| KNN | 8.5 | 9 | 5 | 9 | 3 | 5 | 39.5 |
| Incremental Naive Bayes | 9.5 | 7 | 9 | 9 | 8 | 9 | 51.5 |

*Table 2: Comparison table of classification methods*

Here is the specific description of this table:

**Accuracy:**

All classifiers in the comparison matrix have high accuracy. However SVM has the highest accurate. It can be expected to reach the true value during classification.

**Speed:**

It indicates the complexity of the classifier. All the classifiers are very fast except SVM. Due to the expensive computation, even the fastest SVMs can be extremely slow sometimes. KNN may be extremely slow when classifying testing data, but it usually don't need time in handling the training data.

**Robustness:**

Naïve Bayes has strong ability in resisting noise. SVM is much sensitive to the noise data mainly due to it only depends on few support vectors in training set. KNN is much worse than SVM when lacking of information. Decision tree need tree pruning algorithm to solve over-fit data problem. However, SVM is much stronger to prevent over-fitting problem than other methods.

**Scalability:**

Decision Tree has scalability problem from large database because data may not fit the data type in memory. SVM also suffer with scalability problems due to its complexity computation of high dimensional space.

**Hardware requirements:**

KNN requires efficient storage techniques to implement on hardware. In order to keep the same standard to calculate, the higher requirements on hardware and samples, the less score gets.

**Sample require:**

They all need training data sample to predict unclear data. In sense, Incremental Naïve Bayes needs the smallest data sample because it could learn from the old data to predict new data, and use the new data and the old data as knowledge to predict more new data.[2]

According to the table 2, it is clear that Decision Tree, Naïve Bayes and Incremental Naïve Bayes are the most promising classifiers in the comparison. Decision tree is a good classifier when we don't have large amount of data to handle in URL repository and it has higher score in accuracy than Naïve Bayes as shown in the table. We will discuss more in the solution and implementation section. Incremental Naïve bayes has the highest performance in this evaluation. In fact, it is the improved Naïve Bayes classifier. However, since this field is still new and don't have much research yet. There are some uncertainties with this algorithm. So we didn't implement it in our solution. Therefore, Naïve Bayes becomes the desirable choice for classification task for its great qualities and simple algorithm. We have implemented with both Naïve Bayes and Decision Tree, and provide the performance comparison in our experiment in chapter 4.

# 4 Solution

In the chapter, we propose a classification scheme to automatically classification web sites in NACE and NUTS. The main purpose is to develop a model that could perform classification task with an acceptable accuracy, so that we could use it to predicate the unknown web sites in the future.

In section 4.1 we introduce our suggestion model, and we also provide a specific description in the section 4.1.1, 4.1.2 and 4.1.3. We illustrate our implementation work in section 4.2, we used an existing application Mallet as tool in our experiment. The main experiment task is shown in the section 4.3. A specific result and performance matrix is provided for NACE and NUTS classification in section 4.3.1 and 4.3.2.

## 4.1 Design Specification

This chapter presents an automatic document classification systems, which classify web sites according to NACE and NUTS code. This system constructs a model based on the training data and then classifies the documents based on information from the model.

The system consists of three major components, NLP component, feature selection component and the classification component, as illustrated in figure 11:

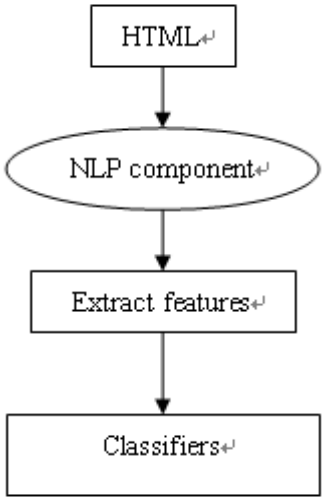


*Figure 11: Structure of proposed model*

### 4.1.1 NLP component

The NLP (Natural Language Processing) component is used in using key vocabulary extraction process from documents as Information Retrieval works. It involves several methods, such as stop words and stemming, both could be used as the pre-classification procedures as well as the application of information retrieval measurement. As we introduced in section 3.2, removing stopwords is a process of removing the most frequent word that exists in a web page document such as ‘to’, ‘and’, ‘it’, etc. It helps to save spaces of storage and reduce features space, get rid of noise and redundant data. Stemming is also could be used for the same purpose. According to the previous introduction, it is a process of extracting each word from a web page document by reducing it to a possible root word. After stemming and

stopping process in each document, we will continue to the next component, the Feature extraction.

### 4.1.2 Feature Selection

After the process of stop word and stemming, we could assume the web page document turned out with clean and informative terms only exist. As we know, Feature Selection is used to reduce the features space without sacrificing classification accuracy. Therefore, we need to choose a method to extract features from the document after NLP process. According to the comparison research, Information Gain is designed to choose terms as vectors for classifier for its remarkable qualities and high performance.

### 4.1.3 Classification

After the first two process, we gain the vectors. Therefore, we need to choose a classifier with reliable accuracy and performance. Since we discussed the comparison of classification methods, there are two algorithm has most promising qualities, Naïve Bayes and Decision Tree. As introduced, Naïve Bayes classifier is a simple classifier based on Bayes theorem from probability theory. Decision Tree splits the class following the classification rules in training data and construct a tree model to classify testing data. In this system, the stem forms of words after extracted features occurring in the training documents is used as the vectors.

The basic steps in the model are as follows:

**Training:**

1. After stopping process, identify the individual stem words occurring in all the training documents in the training set.

2. Generate the feature vectors for each document in the training document set in using Mutual Information and store them along with the correct indexes in the knowledge base.
3. Use the vectors to generate classification model in the classifier. Naïve Bayes calculate the conditional probability for each index and the terms in document. While Decision Tree build a tree model.

**Testing:**

1. Use stop word and stem algorithms to process the given test document
2. Generate the feature vectors in each document of the class. Calculate the probability for this document given each index
3. Perform classification rules on the testing data in classifier. Naïve Bayes calculates the probability for each index in the set of indexes for this document and normalize it with Bayes's theorem. And select the class label with the highest value of probability among the conditional probabilities of all classes as the prediction class for this document. While, Decision Tree classifies each term from root node to leaf node to identify its class label.

Figure following shows the methods used in each system component. Note that Stemming and Information Gain is not implemented in our experiment, because these methods are not included in Mallet. However, as we introduced before, since stemming can only improve less than 10% the accuracy, it would not make much differences without this method. Moreover, Mallet uses Mutual Information instead of Information Gain. Although the performance of Mutual Information is not as good as Information Gain as shown in section 3.2, their algorithm is very similar. Information Gain could be viewed as average Mutual Information. Therefore, the outcome would not change much comparing these two methods. Note that we could choose the method in classifier. It is not possible to use both methods in the classification task.

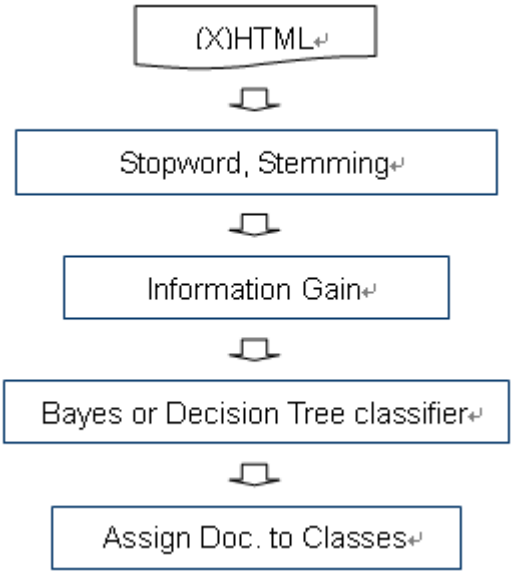


*Figure 12: Flow strategy of the proposed model*

## 4.2 Implementation

In this section, we introduces a program named mallet which we used as testing tool for classification.

### 4.2.1 Introduction

"Mallet is an integrated collection of Java code useful for statistical natural language processing, document classification, clustering, information extraction, and other machine learning applications for text class."[9][10]

We list some advantages and disadvantages for comparison.

**Advantages:**

- Mallet is implemented with *remove-stopwords* and *skip-html* in the NLP component process.
- Mallet contains several classifiers, including Naïve Bayes and Decision Tree.
- Mallet provide detailed classification result, including accuracy, average accuracy, standard deviation and a confusion matrix.
- Mallet can easily been written of new code within the existing infrastructure, so that it is possible for us to program with it and make it to meet our need in experiment in the future.

**Disadvantages:**

- Mallet is not implemented with stemming. Note that this contract to the suggested approach presented in figure 11. However, as we discussed before, Stemming combined with remove-stopwords can only provide 10% improvement at most. [5]
- Mallet chooses mutual information as feature selection method which we introduced before has a poor performance. However, it is not difficult to be improved into information gain.

Considering the advantages and disadvantages, we decide to use Mallet.

### 4.2.2 How to use Mallet:

There are several ways of using Mallet. User can choose to use command-line in Linux, or they could write Java code to call Java classes

directly. If they are not familiar with Java, they can use other languages (e.g. Python) to call Java classes.

A typical usage of Mallet for classification involves two steps:

1. Prepare the documents or other objects to be classified into MALLET, and convert these to a list of instances, where each instance is a feature vector.
2. Classify the feature vectors. Mallet can also compute diagnostic information from an instance list, such as the list of word sorted by mutual information, or printing the label associated with each instance.

### 4.2.3 How Mallet works:

When Mallet use Naïve Bayes for classification, it uses the features selected by mutual information as vectors from the training data to build the classification model applied in Naïve Bayes method, then it classifies the testing samples which had been convected into features and reduced by mutual information in using the classification model. On the output part, we can see the result matrix describing where the classified documents are in every trial. And we can also get the accuracy value of each trail and an average accuracy at last. Besides, we can even know the value of “standard deviation and standard error”.

When Mallet handles dataset with Decision Tree, it uses the vectors to build a decision tree. Let's take out an example from our experiment for explanation. In the following figure 13, at first it split the training data into two classes using a vector “Leisure” selected by mutual information with value of 0.77 from the web documents. That is, 45 documents contain vector “Leisure” are classified in a class from the training data, and the rest of training data had been classified to another class. And it split again the class without vector “Leisure”, with another vector “environment” in the

same way. As shown in the figure 13, it is clear how class J&L and K had been split. The number indicates how many documents related to the vectors. As result, documents contain “Leisure” or “environment” occur without “Leisure” are classified in class J&L. The documents which neither “Leisure” nor ”environment” appeared are categorized in class K (Police). Mallet use this decision tree to decide the class of testing samples.

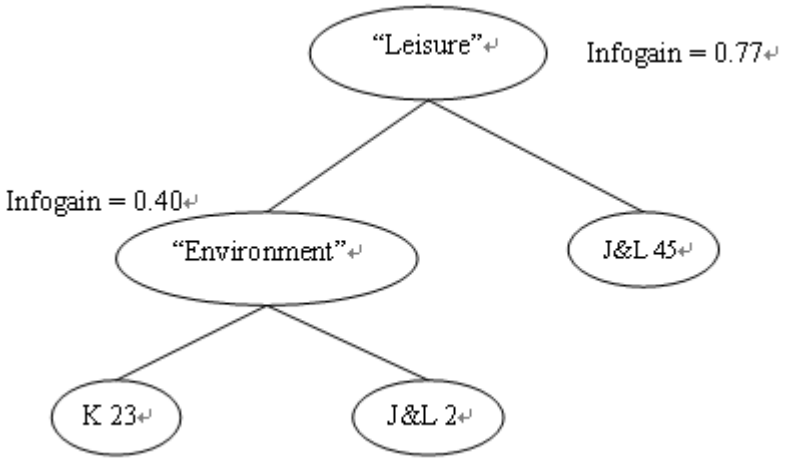


*Figure 13: An example of Decision Tree used in Mallet*

## 4.3 Validation and Testing

In this section we first present the process of pre-processing data and performance matrix of classification. It is covered in NACE. Then, we go on in discussion of performance with different datasets.

### 4.3.1 NACE

In this section of experiments, we mainly focused on NACE classification. We used the web sites in EIAO URL repository collection.

Before starting classification task, we need to prepare the pre-classified data in NACE code so that the classification system to learn from. The class of UK was selected because its languages is English. There are 247 web sites in the class and been classified in alphabet directories according to their Service domains. Then, we searched each class for their NACE code and reclassified again. The figure 14 below illustrates the relationship among the former categories, NACE code and new categories. There are several classes which shared the same NACE code and classified in a new class in the new categories represented in red colour. Also there are several categories we are not sure about their NACE code. So we remained their class and assume they are independent from other classes as shown in the yellow colour. For this reason, we still use alphabet to represent the classes instead of NACE code in the following experiment.

| Category | Service | Level | NACE code | Representation |
|---|---|---|---|---|
| A | Income taxes | 1 | 74.10 | A |
| B | Job search services | 1 | 85.30 | B |
| C | Unemployment benefits | 1 | 85.30 | B |
| D | Child allowances | 1 | 85.30 | B |
| F | Student grants | 1 | 65.11 | F |
| G | Passports | 1 | | G |
| H | driver's license | 1 | 80.41 | H |
| I | Car registration | 1 | 80.41 | H |
| J | Building permission | 3 | 45.20 | J |
| K | Declaration to police | 1 | 75.23 | K |
| L | Public libraries | 3 | 92.50 | L |
| M | Certificates | 1 | | M |
| N | Enrollment in higher education | 1 | 80.30 | N |
| Q | Social contribution for employees | 1 | 74.12 | Q |
| R | Corporation tax | 1 | 74.12 | Q |
| S | VAT | 1 | | S |
| T | Registration of a new company | 1 | | T |
| V | customs declaration | 1 | | V |
| W | Environment-related permits | 1 | | W |

*Figure 14: Prepare the pre-classified data in NACE*

After prepared the dataset, we started the classification task. We chose Mallet for classification tool in the experiment.

**NLP component**

As defined in the designed system in section 4.1, the NLP component is used to deal with pre-classified data. As methods specified in Mallet, we could use “skip-html” or “remove-stopwords”. “skip-html” is used to skip all the words in “<>” which is useful for tokenizing HTML files. Because the default treatment of the words in “<>” as text terms. “remove-stopwords”to remove all the stopwords in the text. The stoplist contains 524 words. [11]

The following table 3 presents the different performance with “skip-html” and “remove-stopwords”. we have classified data using both Naïve Bayes and Decision Tree, and table of accuracy comparison below shown in both methods. The accuracy for all test is a average of 10 trails in order to avoid individual exception and gain a more reliable average performance. It is clear that the NLP component is actually helpful for improve the accuracy. The test is performed on the classes of J&L and K, which contain the cleanest and enough web documents classes in the data collection. we will explain the class J&L in section of performance comparison later. It is clear that the Naïve Bayes shown improved in these methods while Decision Tree is not influenced much.

| | | Accuracy |
|---|---|---|
| | naïve Bayes | Decision Tree |
| Stopwords | 89.79% | 97.66% |
| skip html | 88.09% | 97.23% |
| None | 86.81% | 95.32% |
| Both | 90.28% | 97.23% |

*Table 3: Accuracy of "Skip-html","Remove-stopwords","None" and "Both"*

**Feature Selection**

In order to remove the features with low information from the relative documents containing only relative terms after NLP component, Mallet use mutual information for features selection. It sorts the feature vectors from highest to lowest value according to the associated relationship between each term to the class.

**Portion**

As introduced before, when performing classification with Mallet , before performing the actual classification or diagnostics, a list of feature vectors must be is split into training and testing portions. In order to ensure the best split portion for classification. We present the accuracies with different portion in the following table. This test is performed also in classes of J&L and K.

| | Portion | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| naïve Bayes | 67.33% | 66.49% | 78.78% | 74.86% | 81.86% | 87.23% | 93.14% | 91.30% | 95.83% |
| Decision Tree | 86.00% | 93.51% | 93.05% | 95.29% | 96.61% | 97.02% | 96.00% | 99.13% | 98.33% |

*Table 4: Accuracy with different portions*

As shown in the table, the accuracy increases as portion increases. However, the accuracy increases not very steady. Sometimes, it drops even with increasing portion as marked with red colour. Whereas, we should also consider that the training data should not be less than testing data in order

to avoid the risks of lacking enough learned information for new testing samples. Hence, we chose 0.6 portion to expect the best split for classification, it also applied in all the following experiments.

**Classification**

Mallet specified several classifiers, and we only use Naïve Bayes and Decision Tree as example for classification and comparison. The classifier in Mallet collects statistics from the training set and then apply classifying methods the testing set and output the classification results.

The following table 5 is performed on the original data which had collected and classified manually in NACE categories by EIAO.

Table below plots the result matrix of testing set in NB and DT. As the shown results, it is clear to find out that in every experiment, every testing web sites are all been categorized in the classes of J (Building Permission), K (Police) and L (Library) which has the most highest accuracy among the classes in both classification methods.

| Original class label \ Classified class label | A | B | F | G | H | J | K | L | M | N | Q | S | T | V | W | X | Total | % |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | | | | | 0.2 | | | | | | | | | | | 0.2 | 0% |
| B | | | | | | 0.4 | 0.2 | 0.4 | | | | | | | | | 1 | 0% |
| F | | | | | | | 0.2 | | | | | | | | | | 0.2 | 0% |
| G | | | | | | 0.6 | | 0.6 | | | | | | | | | 1.2 | 0% |
| H | | | | | | 1.8 | | | | | | | | | | | 1.8 | 0% |
| J | | | | | | 6.6 | | 27.6 | | | | | | | | | 34.2 | 19% |
| K | | | | | | 3.8 | 9.6 | 5.4 | | | | | | | | | 18.8 | 51% |
| L | | | | | | 20 | | 7.4 | | | | | | | | | 27.4 | 27% |
| M | | | | | | 1.4 | | 0.4 | | | | | | | | | 1.8 | 0% |
| N | | | | | | 0.8 | | 0.4 | | | | | | | | | 1.2 | 0% |
| Q | | | | | | 0.8 | | 0.2 | | | | | | | | | 1 | 0% |
| S | | | | | | 0.2 | | 0.8 | | | | | | | | | 1 | 0% |
| T | | | | | | 0.4 | | | | | | | | | | | 0.4 | 0% |
| V | | | | | | | | | | | | | | | | | 0 | 0% |
| W | | | | | | 1.4 | | 0.4 | | | | | | | | | 1.8 | 0% |
| X | | | | | | 1 | | 1 | | | | | | | | | 2 | 0% |
| Total | 0 | 0 | 0 | 0 | 0 | 39.4 | 10 | 44.6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **94** | |
| % | 0% | 0% | 0% | 0% | 0% | 17% | 96% | 17% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | | 25.11% |

*Table 5: Classification results of Naïve Bayes*

The column line indicate the document's class label. The row line refers to the class label of document is assigned to. Therefore, the bottom row line is the recall value of each class, while the left line is their precision value. The accuracy lies in the right bottom with black borders as all the other test. The red colour is represented the documents had been classified with wrong label. And the number of testing document is shown in bold number of 94.

Compare the table 5 and 6, it is obvious that Naïve Bayes classify every document into class J or L, while DT almost classify each document into random class although with a higher accuracy than NB. It is because Naïve Bayes does not handle small samples sizes very well. All but J, K and L have

very small sample sizes. That the main reason that they have most maximum precision and recall values as shown in blue colour.

| Original class label \ Classified class label | A | B | F | G | H | J | K | L | M | N | Q | S | T | V | W | X | Total | % |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | | | | | | | | | | 0.2 | | | | | | 0.2 | 0% |
| B | 0.2 | | | | 0.2 | 0.2 | | | 0.2 | | | | | | | 0.2 | 1 | 0% |
| F | | | | | | | | | 0.2 | | | | | | | | 0.2 | 0% |
| G | | 0.2 | | | | 0.6 | | 0.2 | | 0.2 | | | | | | | 1.2 | 0% |
| H | | 0.4 | | | | | | | 0.4 | | 0.8 | | 0.2 | | | | 1.8 | 0% |
| J | | 0.4 | | | | 10.0 | | 22.6 | | | 0.4 | | 0.4 | | | 0.4 | 34.2 | 29% |
| K | | | | | | 0.6 | 18.0 | | | 0.2 | | | | | | | 18.8 | 96% |
| L | 0.2 | 0.4 | | | 0.2 | 15.8 | | 10.2 | | 0.2 | | | 0.2 | | | 0.2 | 27.4 | 37% |
| M | | | | | 0.6 | | | 0.4 | 0.2 | 0.4 | 0.4 | | | | | | 2 | 0% |
| N | | | | | | | | | 0.2 | | 0.6 | | 0.2 | | | 0.2 | 1.2 | 0% |
| Q | | | | | | | | | 0.4 | | | | | | | 0.4 | 0.8 | 0% |
| S | | | | | | 0.2 | | | 0.2 | | 0.6 | | 0.2 | | | | 1.2 | 0% |
| T | | | | | | | | | | | 0.2 | | | | | 0.2 | 0.4 | 0% |
| V | | | | | | | | | | | | | | | | | 0 | 0% |
| W | | | | | | | | 0.2 | | | 0.8 | | 0.2 | | | 0.6 | 1.8 | 0% |
| X | | | | | | 0.4 | | 1.0 | | | | | | | | 0.4 | 1.8 | 0% |
| Total | 0.4 | 1.4 | 0 | 0 | 1 | 27.8 | 18 | 34.6 | 1.8 | 1 | 4 | 0 | 1.4 | 0 | 0 | 2.6 | **94** | |
| % | 0% | 0% | 0% | 0% | 0% | 36% | 100% | 29% | 11% | 0% | 0% | 0% | 0% | 0% | 0% | 15% | | 41.28% |

*Table 6: Classification results of Decision Tree*

**Performance with different datasets**

1. *Data without clean up*

By examining the original data, refer to table 5 and 6, it had shown very low over all accuracy in both tables. However, we found out there are three classes J (Building Permission), K (Police) and L (Library) has most maximum precision and recall value far more than other classes as shown in the blue colour. The main reason is they have more than half of the whole data, so that the classifiers almost assign every document into these classes. Therefore, we decide to perform a classification only on these classes to evaluate the performance.

It is obvious that class J , K and L have the highest accuracy. The reason is that we have enough training data for these classes. Therefore, we select J , K and L which are the classes in NACE code for a new trial of experiment. Figure below shows the result matrix and the accuracy of testing. We can observe from the experiments that the average accuracy is very low and especially in classes of J and L. In particular, a great number of web sites in class L are categorized in class J , whereas a lot of documents in class J also are classified in class L. This indicates that class J and L has something similar so that the program can’t distinguish the differences between them and give bad result of the classification. Therefore, we checked the web sites in the these two classes and found out that there are a large number of the web sites overlapping in both these two classes. The shadowed line means the average number of web sites that are correct class. And the Column and Row indicate the average number of web sites that are classified in each class.

| Original class label | Classified class label | | | | |
|---|---|---|---|---|---|
| | J | K | L | Total | % |
| J | 9.6 | 0.3 | 18.3 | 28.2 | 34% |
| K | 3.9 | 12.6 | 1.9 | 18.4 | 68% |
| L | 30.2 | 0.1 | 2.1 | 32.4 | 6% |
| Total | 43.7 | 13 | 22.3 | **79** | |
| % | 22% | 97% | 9% | | 30.76% |

*Table 7: Classification in Naïve Bayes*

| Original class label | Classified class label | | | | |
|---|---|---|---|---|---|
| | J | K | L | Total | % |
| J | 12.8 | 0.3 | 15.3 | 28.4 | 45% |
| K | 1.2 | 16.8 | 0.2 | 18.2 | 92% |
| L | 25.8 | 0.5 | 6.1 | 32.4 | 19% |
| Total | 39.8 | 17.6 | 21.6 | **79** | |
| % | 32% | 95% | 28% | | 45.19% |

*Table 8: Classification in Decision Tree*

*2. Clean Data*

It is clear that the classification of class J (Building Permission) and L (Library) is now, the over all accuracy is very low to less than 40% while the accuracy of class K (Police) is surprisingly high. This is due to the overlapping samples in class J and L. It is further evident that we cannot

expect high accuracy with overlapping samples in classes. Therefore, we need to clean out the overlapping samples. Because of this, we moved the overlapping data into a new class called J&L to distinguish from class J and L, in order to avoid further interference. Figure below illustrates the relationship among these three classes.

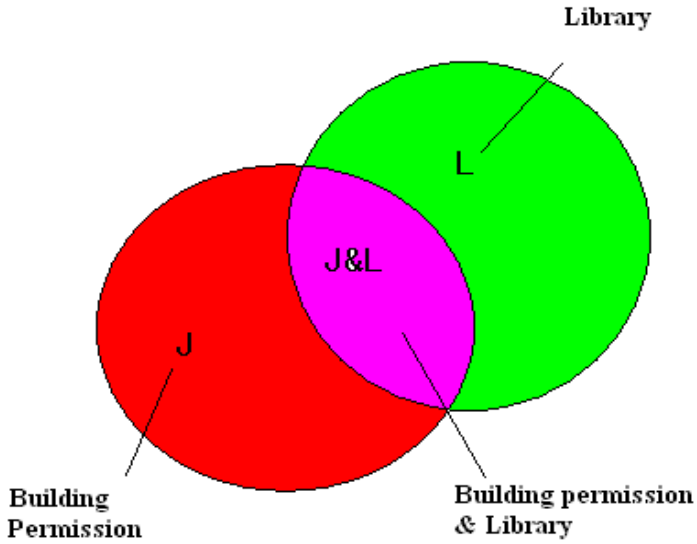


*Figure 15: The relationship among classes of J (Building Permission), L (Library) and J&L*

Figure below is the matrix result and testing accuracy after cleaning. Since class J and L only has 5,6 web sites while class J&L has more than 70 web sites, the testing result categorized all the pages into J&L without surprise in Naïve Bayes. However, the accuracy is much higher than unclean data. This indicates that breaking the interference classes into independent classes did indeed contribute to improved accuracy and performance. This

also shows that the Naive Bayes algorithm of independent assumption. Decision Tree also shows the increasing accuracy although its less than Naïve Bayes a little.

| Original class label | Classified class label | | | | |
|---|---|---|---|---|---|
| | J | J&L | L | Total | % |
| J | | 2.3 | | 2.3 | 0% |
| K | | 28.9 | | 28.9 | 100% |
| L | | 0.8 | | 0.8 | 0% |
| Total | 0 | 32 | 0 | **32** | |
| % | 0% | 90% | 0% | | 90.31% |

*Table 9: Classification result in Naïve Bayes*

| Original class label | Classified class label | | | | |
|---|---|---|---|---|---|
| | J | J&L | L | Total | % |
| J | 0.6 | 1.2 | 0.5 | 2.3 | 26% |
| K | 1.9 | 26.2 | 0.8 | 28.9 | 91% |
| L | 0.3 | 0.5 | | 0.8 | 0% |
| Total | 2.8 | 27.9 | 1.3 | **32** | |
| % | 21% | 94% | 0% | | 83.75% |

*Table 10: Classification result in Decision Tree*

*3 Only enough pages*

It is clear when the number of samples/pages for J and L is low, the over all accuracy is very low in both methods. It further indicates that we

cannot expect accurate results with large differences among the class sample size. Because of this, we remove all categories with sample size less than 40 (J and L) from the testing/ training data. Since class K contains sample size of 46, we add class K with J&L to perform a classification and get an over all accuracy of 80% as shown in the following tables.

This shows that whenever we have enough samples used for training, we can expect an over all high accuracy with our approach. Tables below shows the result. We can observe from the experiments that the testing data has been categorized in both classes instead of only one class while the accuracy doesn't change much in Naïve Bayes. This also shows the Naive Bayes algorithm's robustness of information retrieval that the size of classes does not impact the result of classification. However, the accuracy of Decision Tree increases much higher than the "unbalanced dataset". It indicates that the requirements of good quality dataset for Decision Tree is much more higher than Naïve Bayes.

| Original class label | Classified class label | | | |
|---|---|---|---|---|
| | J&L | K | Total | % |
| J&L | 27.7 | 0.8 | 28.5 | 97% |
| K | 4.5 | 14 | 18.5 | 76% |
| Total | 32.2 | 14.8 | **47** | |
| % | 86% | 95% | | 88.72% |

*Table 11: Classification result in Naïve Bayes*

| Original class label | Classified class label | | | |
|---|---|---|---|---|
| | J&L | K | Total | % |
| J&L | 27.8 | 0.7 | 28.5 | 98% |
| K | 0.5 | 18 | 18.5 | 97% |
| Total | 28.3 | 18.7 | **47** | |
| % | 98% | 96% | | 97.45% |

*Table 12: Classification result in Decision Tree*

*4. Summery*

Here is the overview of how the accuracy changes with different datasets. At first we classified the original data in Naïve Bayes and Decision Tree. Both of them provide a very low over all accuracy. Then we notice the amount sample size contained in classes J, K and L. So we classified only with these classes. However, the over all accuracy still low due to class J and L contain a great number of overlapping samples. That is the reason we split the class J and L into three classes. We separated the overlapping samples from the original class and form three new classes: J, L and J&L. Now the class J, L is not the original data but without overlapping samples. As the shown in table, the over all accuracy increases rapidly on the data of the new class J, L and J&L. However, we found out that the new class J and L has low precision and recall value due to the small sample size than J&L. Therefore, we perform a classification on class J&L and K which both contain enough dataset with the highest over all accuracy of 97% on Decision Tree.

| Data description | Over all accuracy | |
|---|---|---|
| | naïve Bayes | Decision Tree |
| Original data classified in NACE | 0.2169 | 0.3838 |
| Overlapping classes(J, L) and class K | 0.3076 | 0.4508 |
| Better representation of overlapping categories.( new class J, L and J&L) | 0.9031 | 0.8375 |
| Only NACE categories where we have more than 10 samples (class J&L, K) | 0.8872 | 0.9745 |

*Table 13: Comparison matrix with accuracy of differences dataset*

Note that the accuracy of Naïve bayes does not show a better result in the best dataset J&L and K. It has the best accuracy on the class J, L and J&L. As shown in the table 9, Naïve Bayes categories every document into class J&L. It indicates that Naïve Bayes does not have ability to distinguish between class J, L to class J&L. The high accuracy only due to the small size of class J, L but the misleading performance of Naïve Bayes. It further implies that if class J and L have bigger size, the accuracy of Naïve Bayes would be much worst than now. However, the last dataset shows the true performance of Naïve Bayes since it contain enough information for every class without overlapping data. Hence, the accuracy of Naïve Bayes is 88%.

### 4.3.2 NUTS

We also had experiments with classification dataset in NUTS. Since only two countries in NUTS using English languages, so we have only two categories in NUTS. And we still perform the classification task in using Mallet. In this time, we don't have any interference classes with overlapping data. So as you can see in the following figure 14 and 15, the accuracy of original data is much higher than in NACE classification. The results of classification is shown in the table.

| Original class label | Classified class label | | | |
|---|---|---|---|---|
| | Ireland | UK | Total | % |
| Ireland | 1.1 | 23 | 24.1 | 5% |
| UK | | 64.9 | 64.9 | 100% |
| Total | 1.1 | 87.9 | **89** | |
| % | 100% | 74% | | 74.16% |

*Table 14: Classification in NUTS using Naïve Bayes*

| Original class label | Classified class label | | | |
|---|---|---|---|---|
| | Ireland | UK | Total | % |
| Ireland | 20.3 | 4 | 24.3 | 84% |
| UK | 2 | 62.7 | 64.7 | 97% |
| Total | 22.3 | 66.7 | **89** | |
| % | 91% | 94% | | 93.26% |

*Table 15: Classification in NUTS using Decision Tree*

Note that NUTS is further organised into smaller categories. Hence, we only show the top-most level, categorising between two countries.

Nevertheless our results indicate that NUTS classification using Naïve Bayes and Decision Tree is clearly possible with this approach.

# 5 Discussion

In this chapter, we will discuss the performance matrix shown in testing experiments in section 4.3. Decision Tree had a best accuracy of 97% in NACE, while Naïve Bayes had a highest but misleading accuracy of 90%. This is due to Naïve Bayes is a classifier based on Bayes theorem which depends on probabilities. When a class contain huge data size than other classes, Naïve Bayes is leaded by probabilities to classify every document into the biggest class as the classification performed on dataset of class J, L and J&L shown in table 9. This indicates the main reason of highest accuracy of Naïve Bayes is because of the small data size of class J and L but the misleading performance of classifier. It further indicates that if the size of class J and L is bigger, the accuracy would be worst than now. Therefore, the true accuracy is performed on the best dataset J&L and K, not only because it contains enough information on each class but also without overlapping data. Hence, the true accuracy in NACE classification of Naïve Bayes is 88%.

## 5.1 Remove Stopwords VS. Skip html

As shown in the comparison table 3 in the section 4.3.1, the method of *Remove-stopwords* and *skip-html* only has slightly little change in accuracy. They are basically in a same manner of searching keywords which indicated the non-informative terms and remove them all from the dataset. As we introduced in the *Remove-stopwords* in section 3.2.1, stoplist should change when dealing with different objects. In this case, we are dealing with web page document, which includes a lot of tags and also the normal text. Because of this, we also tried both methods for classification. As shown in the table, Naïve Bayes

shows a improvement in accuracy, however Decision Tree don't have a obvious change. This is believed to be because Naïve Bayes shows a more dependent on keywords than Decision Tree. Beside, Decision Tree had shown a great accuracy nearly perfect in the experiment, it is difficult to improve more accuracy. Therefore, Naïve Bayes is dependent more on removing stopwords.

## 5.2 Naïve Baye VS. Decision Tree

According to the comparison matrix in Chapter 3.4, we will mainly discuss the following issues: Accuracy, Require sample and Over-fitting which we feel are the most important criteria.

1. **Accuracy**: Figure 5.1 shows the accuracy per test run in order to show the stability of the algorithms. The accuracy of Naïve Bayes is in red colour while Decision Tree is in blue. It is obvious that in the following performance table 16, Decision tree has higher average accuracy than Naïve Bayes, which corresponds to the comparison matrix table 2, Decision Tree has a estimated accuracy of 9 which Naïve Bayes has 8. Note that Naïve Bayes has a higher value of the standard deviation than Decision Tree, which indicates that Naïve Bayes is not so steady as Decision Tree.

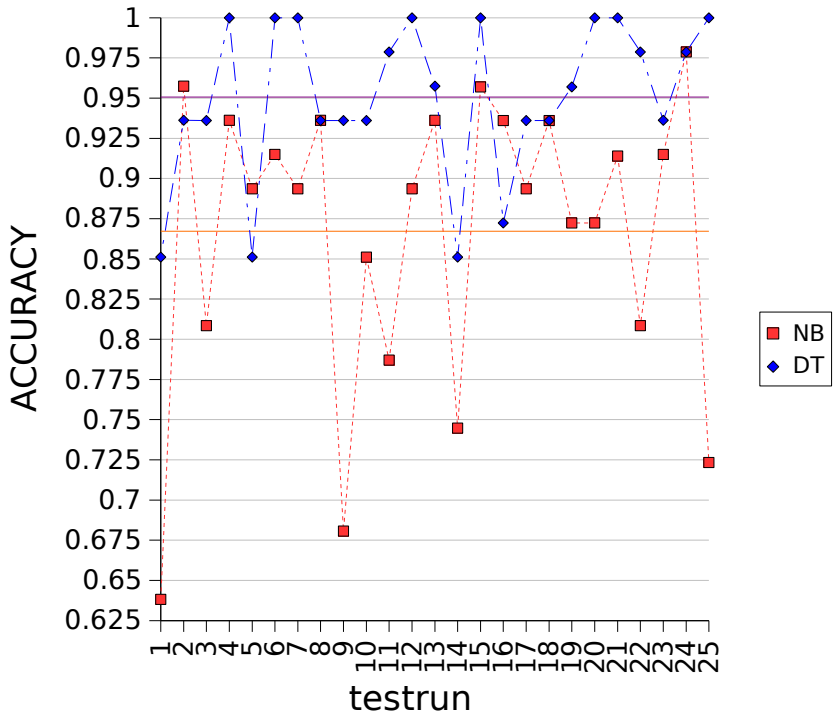


*Table 16: Performance of Naïve Bayes and Decision Tree*

2. **Sample Requirement :** The requirements of sample for both classification methods are almost the same. However, note the accuracy of Decision Tree is slightly less than Naïve Bayes when handling the clean dataset with classes J, L, J&L. This indicates that Decision Tree has difficulties in dealing with unknown class samples when lack of enough information for the class than Naïve Bayes. This also cross-reference to the comparison matrix table 2 shows Decision Tree require more quality of dataset for classification.

3. **Over-fitting:** Over-fit data is the noise exists in the training data which might mislead the classifier. As shown in the comparison matrix in section 3.4, Decision Tree is weaker when facing the noisy data compare to Naïve Bayes. However, we found out that the accuracy of Decision Tree is higher than Naïve Bayes when dealing with overlapping data. In our opinion, considering the overlapping data mainly gathered in class J (Building Permission) and L (Library ), which is not difficult for Decision Tree to make decision of the class label between class J and L. However, Naïve Bayes is a typical statistical classifier. It views the attributes as equal importance related to class. The overlapping data confuses the statistical relationship between terms and class. Because of this, the classification result will be rather arbitrary in this case.

## 5.3 Summary

It has been shown that the proposed solution performs well. The pre-processing procedures is necessary and both of the approaches used in the experiments have shown good quality during classification task in handling web page documents. Especially, the accuracy of Decision Tree is nearly to 100% which is almost perfect to applied in predication the unknown web page document in the future work. Naïve Bayes and Decision Tree all shown their abilities in dealing with noise data which is a serious problem in real-word database. We have the best accuracy of NACE code classification at 97% in Decision Tree and 88% in Naïve Bayes. While NUTS classification has 73% accuracy in Naïve Bayes and 93% accuracy in Decision Tree. We believe the main reason keywords more important for NACE than NUTS due to the keywords was extracted

automatically. There is no clear boundaries between the keywords for economic activities and geographical location. Since our classifier uses nature language processing to generate keywords, it is clearly that NACE is much more related to these keywords than NUTS. Therefore, the keywords for NACE should not be used in NUTS classification. Our experiments indicates that the keywords for NUTS should be more specific and related to geographic terms.

# 6 Conclusion

## 6.1 Conclusion

In this thesis, we have proposed and evaluated a possible solution to perform automatic web page classification in NACE and NUTS for EIAO.

We discussed the difficulties of Web Mining, and use keyword-based document classification scheme, combined with preprocessing techniques, then use the pre-classified training data to built a model to perform the classification task on the testing data.

Due to the characteristics of the Web, we discussed the necessary of preprocessing procedure, and compared several methods of searching for keywords and features selection, then proved their contribution for improving the effectiveness and performance of classification task based on our experiment.

In our heart of the solution is the classifiers. We evaluated several methods based on their accuracy and scalability, also their abilities to handle with large database. Then we tested with the two most promising classifiers in our experiment, Naïve Bayes and Decision Tree, to evaluate the solution's accuracy and performance. They had shown better performance than other methods in the comparison table specified in section 3.3.6.

Our proposed solution was shown to successfully to classify web pages with high accuracy for NACE. Additionally, NUTS classification is also shown a good result in our experiment. At best NACE classification has an accuracy of 97% with Decision Tree and 88% with Naïve Bayes. While NUTS

classification has 73% accuracy with Naïve Bayes and 93% accuracy with Decision Tree. Since we mainly focused our work in NACE, the performance of NUTS is not as good as NACE's. But they are built in the similar structured, it would be easy to improve the performance with NUTS in using the same strategy in future work.

## 6.2 Future work

Based on our work, we could continuously explore in depth. In this section, we will briefly introduce some domains that could improve accuracy in the future work.

- Stemming technique could be added in the NLP component, it will surely improve the accuracy even better.

- Since Mutual Information is evaluated as a poor automatic features selection algorithm, we could use other methods such as Document Frequency (DF) or Information Gain (IG) instead.

- For NUTS classification, we could insert with geographical relevant term as keywords added in the features space. We could expect a higher accuracy with help.

- Since the accuracy of Naïve Bayes is less than Decision Tree, it indicates that the accuracy could be more improved. We could implemented in hierarchical classification approaches for improvement. In sense, the classification method combined with two

strategy is surely better than pure keyword classification. Also NACE and NUTS are all hierarchical structured. The hierarchical classification could be expected with much higher accuracy.

- Moreover, we could also implemented with Incremental learning algorithm as the best promising classifier in our evaluation.

- To expend the application domains, we could also work with more languages other than only English, so that we could evaluate the web pages in European area level.

# 7 Appendices

[1]: EIAO, European Internet Accessibility Observatory , http://www.eiao.net/
[2]: Jiawei Han; Michiline Kamber, Data Mining Concepts and Techniques
[3]: European Commissions, NACE, http://circa.europa.eu/irc/dsis/nacecpacon/info/data/en/2007%20introduction.htm
[4]: Mallet, Introductory guidelines-EN,
[5]: European Commissions, NUTS, http://ec.europa.eu/eurostat/ramon/nuts/codelist_en.cfm?list=nuts
[6]: Yiming Yang; Jan O. Pedersen, A Comparative Study on Feature Selection in Text Categorization, 1997
[7]: wikipedia, Feature Selection, http://en.wikipedia.org/wiki/Feature_selection
[8]: Margaret H. Dunham, Data Mining Introductory and Advanced Topics,
[9]: Pang-ning Tan; Michael Steinbach; Vipin Kumar, Introduction to Data Mining
[10]:Mallet, Documentation, http://mallet.cs.umass.edu/index.php/Command_line_tutorial
[11]:Fei Xia, Mallet Guides for LING 572, 2007
[12]:Mallet, Documentation, http://mallet.cs.umass.edu/index.php/Main_Page