A Comparative Study of Different Classifiers in Text Categorisation

Madhukshara Sarkar (B18735)

Ramakrishna Mission Vivekananda Educational and Research Institute

M.Sc. Big Data Analytics

April 2019

1 INTRODUCTION

Over the recent years, text classification has become one of the key techniques for organizing online information. It can be used to organize document databases, filter spam from e-mails, or learn user's news reading preferences [1], personal information agents and assistants, information retrieval and automatic indexing [2]. Text Classification is the problem of assigning a text document into one or more topic categories or classes [3].

The automated categorisation (or classification) of texts into predefined categories has witnessed a booming interest in the last 10 years, due to the increased availability of documents in digital form and the ensuing need to organize them. In the research community the dominant approach to this problem is based on machine learning techniques: a general inductive process automatically builds a classifier by learning, from a set of pre-classified documents, the characteristics of the categories. The advantages of this approach over the knowledge engineering approach (consisting in the manual definition of a classifier by domain experts) are a very good effectiveness, considerable savings in terms of expert labour power, and straightforward portability to different domains [4].

In this study we apply 4 different classification algorithms viz. Naïve Bayes, k-Nearest Neighbour, Logistic Regression and Support Vector Machine, to categorize the Reuters-21578 data set. We shall briefly discuss the data set in this section, and also explain how the data is cleaned and prepared for classification. In Section 2 we shall see in some details about the methods, i.e., classifiers used and the validation techniques which we would use to confirm our results. In Section 3 we learn about the evaluation criteria used to interpret the performance of our classifiers. Section 4 presents the results obtained after performing the classifiers on the dataset, and the analysis performed based on those results. Finally, in Section 5 we conclude our study with some discussions and future scope (if any).

1.1 DATASET DESCRIPTION

The documents in the Reuters-21578 collection appeared on the Reuters newswire in 1987. The documents were assembled and indexed with categories by personnel from Reuters Ltd. (Sam Dobbins, Mike Topliss, Steve Weinstein) and Carnegie Group, Inc. (Peggy Andersen, Monica Cellio, Phil Hayes, Laura Knecht, Irene Nirenburg) in 1987. The corpus originally contains 135 categories and it is a multi-class (e.g. there are multiple classes), multi-label (e.g. each document can belong to many classes) dataset. Hence we consider the ModApte version of Reuters. The ModApte version contains 12902 documents with 90 categories and the corpus is divided into training and test sets.
In this study we will consider only the following 10 categories of the ModApte version of Returns-21578 corps:

alum, barley, coffee, dmk, fuel, livestock, palm-oil, retail, soybean, veg-oil

The class label of a document is the name of the directory to which it belongs to. There are 499 documents in the training set and 185 documents in the test set.

1.2 DATA PREPARATION

The first step in text categorisation is to transform documents, which typically are strings of characters, into a representation suitable for the learning algorithm and the classification task [5]. During the first step, each document in the corpus is processed to extract its most representative keywords (terms). As each term has a different relevance when it is used to describe the document content, a numerical weight is assigned. This weight quantifies the importance of the term for describing the document semantic. Moreover, a data normalizing process is used to transform term weights into a new unified value range, with TF-IDF (term frequency-inverse document frequency) being the most used normalization process [6].
As a result, each document is represented by a m-dimensional vector (instance), where m is the total number of terms in the corpus, and an associated class (relevant or non-relevant). The similarity between two documents is computed based on the distance of their representative vectors.

1.2.1 Stemming and Stopwords

In many cases, irrelevant terms are included on the sparse matrix, thus decreasing the classification results. In order to partially remove the noise, some stemming techniques and stopword removal are used.

Stemming techniques [7] morphologically identify terms and their variants (nouns, adjectives, adverbs, etc.) and reduce the data dimensionality through a step called conflation. It is to extract the stem of all the terms and apply a matching process to fuse or combine the terms, avoiding variants in the final representation.

Stopword lists [8] are wordlists composed of irrelevant terms such as articles, determiners, or interrogative particles. These terms are usually excluded during the document matrix generation.

In this way, combining stopword filtering and stemming techniques helps to avoid non-useful terms and to significantly improve the information retrieval systems and their results.

1.2.2 tf-idf Weight

The term frequency (tf) of a term t in a document d is defined as the number of times that t occurs in d.

The inverse document frequency (idf) estimates the rarity of a term in the whole document collection. (If a term occurs in all the documents of the collection, its idf is zero).

Let, the number of times the i^{th} term t_i occurs in the j^{th} document be denoted by $tf_{ij}, i = 1,2, \dots, m, j = 1,2, \dots, n$, where m denotes the number of terms in a corpus and n denotes the number of documents in a corpus. Document frequency df_i is the number of documents in which the term t_i occurs. Then the inverse document frequency idf_i is given as

$$idf_i = \log\left(\frac{n}{df_i}\right)$$

The tf-idf weight of a term is the product of its tf weight and its idf weight.

$$w_{ij} = tf_{ij} \times idf_i = tf_{ij} \times log\left(\frac{n}{df_i}\right), \quad \forall i = 1,2,\dots,m;\ j = 1,2,\dots,n$$

2 METHODS

In this study, we perform the Naïve Bayes, k-Nearest Neighbour, Logistic Regression and the Support Vector Machine classifiers to categorize the test data set. Let us take a brief outlook of these classifiers and the basic concepts of parameter tuning and cross-validation algorithms.

2.1 NAÏVE BAYES CLASSIFIER

The Naive Bayes Classifier technique is based on the Bayesian theorem and is particularly suited when the dimensionality of the inputs is high. Despite its simplicity, Naive Bayes can often outperform more sophisticated classification methods.

Let $\vec{X} = [x_1, x_2, \dots, x_m] \in \mathbb{R}^m$ be any m dimensional feature vector and let $w_1, w_2, \dots, w_c$ be c number of classes in the dataset. Following the Bayes theorem, we have

$$p(w_k|\vec{X}) = \frac{p(w_k)p(\vec{X}|w_k)}{p(\vec{X})}, \forall k = 1,2 \dots, c$$

According to the assumption of the Naïve Bayes classifier each feature x_i is conditionally independent of every other feature x_j , $\forall j \neq i$ and $\forall i, j = 1,2, \dots, m$ given class $w_k, \forall k$. Therefore,

$$p(w_k|\vec{X}) \propto p(w_k) \prod_{i=1}^{m} p(x_i|w_k) \ , \forall k$$

The Naïve Bayes classifier finds the best class for the feature vector $\vec{X}$ as the most likely or maximum a posteriori (MAP) class w_{map} as follows

$$w_{map} = \max_{k \in \{1,2,\cdots,c\}} p(w_k) \prod_{i=1}^{m} p(x_i|w_k)$$

The priors, $p(w_k)$, can be easily computed from the training set as,

$$p(w_k) = (Number\ of\ samples\ in\ w_k)/(Total\ number\ of\ samples\ in\ the\ data), \forall k$$

There exists different naive Bayes classifiers based on different assumptions on the distribution of $p(x_i|w_k), \forall i, k$.

2.1.1 Multinomial Naïve Bayes Classifier

When the features are discrete, for example in the case of text classification, we use the *Multinomial Naïve Bayes Classifier*. In this case, we estimate the conditional probability $p(x_i|w_k), \forall i, k$ as the relative frequency of feature x_i in the documents belonging to class w_k as

$$p(x_i|w_k) = \frac{F_{ik}}{\sum_{i=1}^{m} F_{ik}}$$

Here F_{ik} is the frequency of feature x_i in class w_k. In a text data F_{ik} is the sum of frequencies of a term $\overrightarrow{x_i}$ in all the documents in class w_k. Note that the text data are generally sparse i.e., a term may not occur in any document of a class (but they appear in some other classes) and this is true for all terms in the vocabulary [9]. Thus $p(x_i|w_k) = 0$, if $F_{ik} = 0$, which results to $\prod_{i=1}^{m} p(x_i|w_k) = 0$, even if for some $x_j \neq x_i$, $F_{ik} \neq 0$ i.e., $p(x_j|w_k) \neq 0, \forall j$. To avoid this situation, we add one to the frequency of a term as follows

$$p(x_i|w_k) = \frac{F_{ik} + 1}{\sum_{i=1}^{m}(F_{ik} + 1)} = \frac{F_{ik} + 1}{\sum_{i=1}^{m} F_{ik} + m}$$

This is known as *Laplace Smoothing* [9]. Here m is the number of terms in the vocabulary of the given training set.

2.1.2 Bernoulli Naïve Bayes Classifier

In the multivariate Bernoulli event model, features are independent Booleans (binary variables) describing inputs. Like the multinomial model, this model is popular for document classification tasks where binary term occurrence features are used rather than term frequencies. If x_i is a Boolean expressing the occurrence or absence of the i^{th} term from the vocabulary, then the likelihood of a document given a class w_k is given by

$$p(x_i|w_k) = p_{ki}^{xi}(1 - p_{ki})^{(1-xi)}, \qquad \forall i, k$$

where p_{ki} is the probability of class w_k generating the term x_i. This event model is especially popular for classifying short texts. It has the benefit of explicitly modelling the absence of terms.

2.2 K-Nearest Neighbours Classifier

The intuition underlying k-nearest neighbour classification is quite straightforward. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbour. The method finds the k-nearest neighbours of the test data point from the training samples using a distance function.

We describe the process of kNN decision rule below:

Let $(\overrightarrow{x_i}, y_i)$ be given, where, $\overrightarrow{x_i} \epsilon \mathbb{R}^m$, $\forall i = 1,2, \dots, m$ is m dimensional feature vectors and $y_i \in \{1,2, \cdots, c\}$ is the class label of $\overrightarrow{x_i}, \forall i$, where n is the number of samples in the training set. Here c is the number of classes and let us assume $c \geq 2$. Let us consider $\overrightarrow{x_0}$ be a test data point for which the class label is to be determined. The steps of the kNN algorithm is as follows:

1. Calculate $\rho(\overrightarrow{x_0}, \overrightarrow{x_i}), \forall i = 1,2, \dots, n$, where ρ is the distance function e.g., Euclidean distance.
2. Arrange n number of ρ values in non-decreasing order.
3. Consider the first k (say) ρ values and find the k data points corresponding to these k distances.
4. Let L_i denotes the number of points belonging to y_i among the k points, for $i = 1,2, \dots, c$.
5. Now assign $\overrightarrow{x_0}$ to y_i if $L_i > L_j, \forall j \neq i$.

Different values of k can change the classification result and hence choice of k is crucial for kNN rule. The cross validation technique is used to tune the value of k from the training samples.

2.3 Logistic Regression

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. It is used for classification of data samples e.g., filtering emails, finding malignant tumours etc.

Let $(\overrightarrow{x_i}, y_i)$ be given, where, $\overrightarrow{x_i} \epsilon \mathbb{R}^m$, $\forall i = 1,2, \dots, m$ is m dimensional feature vectors and $y_i \in \{0,1\}$ is the class label of $\overrightarrow{x_i}, \forall i$, where n is the number of samples in the training set. Let us consider the parameter vector $\vec{\beta} = [\beta_0, \beta_1, \cdots, \beta_m]$. The aim here is to map the input x to either 0 or 1. Hence we have to compute the posterior probabilities $p(y = 0|x)$ and $p(y = 1|x)$ and x is to be classified to the class which has maximum posterior probability. We can calculate the posterior probabilities as follows:

$$p(y = 1|x) = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)}, \forall x$$

$$p(y = 1|x) = \frac{1}{1 + \exp(\beta_0 + \beta^T x)}, \forall x$$

This is known as the *sigmoid function* or the *logistic function*. The aim is to estimate $\vec{\beta}$ using the training samples.

2.4 Support Vector Machine

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labelled training data, the algorithm outputs an optimal hyperplane which categorizes new examples.

Let $(\overrightarrow{x_i}, y_i)$ be given, where, $\overrightarrow{x_i} \epsilon \mathbb{R}^m$, $\forall i = 1,2, \dots, m$ is m dimensional feature vectors and $y_i \in \{-1,1\}$ is the class label of $\overrightarrow{x_i}, \forall i$, where n is the number of samples in the training set. Let $\vec{w} = [w_1, w_2, \cdots, w_m]$ be the m dimensional weight vector. The feature vector x_i for which $y_i(w^T x_i + b) = 1, \forall\, i = 1,2, \dots, m$, are known as the support vectors. Support vectors are data points that are closer to the hyperplane and influence the position and

orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

The support vector machines require the solution of the following optimization problem:

$$\min \ \frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i$$

$$\text{subject to } y_i(w^T\phi(x_i) + b) \ \geq 1 - \xi_i \ ; \ \xi_i \geq 0$$

Here training vectors x_i are mapped to higher (maybe infinite) dimensional space by the function ϕ. SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. $C > 0$ is the penalty parameter of the error term. Furthermore $K(x_i, x_j) \equiv \phi(x_i)^T\phi(x_j)$ is called the kernel function [10]. The four basic kernels are:

- Linear: $K(x_i, x_j) = {x_i}^T x_j$
- Polynomial: $K(x_i, x_j) = (\gamma {x_i}^T x_j + r)^d, \gamma > 0$
- Radial basis function (RBF): $K(x_i, x_j) = \exp\left(-\gamma\|x_i - x_j\|^2\right), \gamma > 0$
- Sigmoid: $K(x_i, x_j) = \tanh(\gamma {x_i}^T x_j + r)$

Here, γ, r and d are kernel parameters.

2.5 Parameter Tuning and Cross Validation Algorithms

In real-life applications, estimators used by practitioners always depend on unknown parameters that have to be chosen, which is called “parameter tuning”. For instance, when estimating a density with a histogram (or a kernel estimator), the partition (resp. the bandwidth) has to be specified beforehand. Other instances are LASSO and SVM algorithms: They both depend on a “regularization parameter” that has to be chosen by practitioners. Moreover, the final performance of the estimator crucially depends on that unknown parameter. This tuning step is often performed by use of Cross-Validation (CV) in practice, but without any real guaranty of performance [11].

Cross-validation (CV) is a popular strategy for algorithm selection. It is a statistical method used to estimate the skill of machine learning models. The main idea behind CV is to split data, once or several times, for estimating the risk of each algorithm: Part of data (the training sample) is used for training each algorithm, and the remaining part (the validation sample) is used for estimating the risk of the algorithm. Then, CV selects the algorithm with the smallest estimated risk [12]. The widely used forms of cross-validation are leave one out method and k-fold cross-validation technique. In this study we have made use of the k-fold cross validation technique to obtain the optimum values of the hyper-parameter of the classifiers.

In this study we use the grid search technique in Python Scikit-Learn Module for cross validation purpose.

3 EVALUATION CRITERIA

Evaluation of classification algorithms is one of the key points in any process of data mining. The most commonly tools used in analysing the results of classification algorithms applied is the confusion matrix [13].

The confusion matrix displays the number of correct and incorrect classifications made by the classifier as compared to the actual classifications in the data [13]. A general form of a confusion matrix is presented below:

	Classes Predicted		
Actual Classes		Positive	Negative
	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Here *true positive* (TP) counts the number of data points correctly predicted to the positive class. *False positive* (FP) counts the number of data points that actually belong to the negative class, but predicted as positive (i.e., *falsely predicted as positive*). *False negative* (FN) counts the number of data points that actually belong to the positive class, but predicted as negative (i.e., *falsely predicted as negative*). *True negative* (TN) counts the number of data points correctly predicted to the negative class.
The accuracy is the proportion of the total number of correct predictions and calculated as the ratio between the number of cases correctly classified and the total number of cases.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN}$$

The error indicates the proportion of classes classified incorrectly.

$$\text{Error} = 1 - Accuracy$$

The performance of a classifier in classifying the feature vectors can be measured by its precision, recall and f-measure.

The precision gives a measure of the proportion of positive samples which are correctly classified.

$$\text{Precision} = \frac{TP}{TP+FP}$$

Note that, if FP = 0, then Precision = 1 i.e., all negative samples are classified correctly.

The recall gives a measure of the proportion of correct classification amongst all positive classes.

$$\text{Recall} = \frac{TP}{TP+FN}$$

If FN = 0, then Recall = 1 i.e., all positive samples are classified correctly.

F-measure combines precision and sensitivity as the harmonic mean of the two parameters.

$$\text{F-measure} = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

The closer the values of precision and recall, the higher is the f-measure. F-measure becomes 1 when the values of precision and recall are 1 and it becomes 0 when precision is 0, or recall is 0, or both are 0. Thus f-measure lies between 0 and 1. A high f-measure value is desirable for good classification.
The above measures are corresponding to a two class classification problem. But they can be extended in a generalised fashion for an m-class classification problem. There are two conventional methods to evaluate the performance of a classifier aggregated over all classes, namely *macro-averaging* and *micro-averaging*.
Macro-averaging gives equal weight to each class, whereas micro-averaging gives equal weight to each per-document classification decision. Because the f-measure ignores true negatives and its magnitude is mostly determined by the number of true positives, large classes dominate small classes in micro-averaging.

The macro-averaged precision and recall for a set of m classes are computed as follows:

$$\text{Macro-averaged Precision} = \frac{1}{m}\sum_{i=1}^{m} \frac{TP_i}{TP_i + FP_i}$$

$$\text{Macro-averaged Recall} = \frac{1}{m}\sum_{i=1}^{m} \frac{TP_i}{TP_i + FN_i}$$

The micro-averaged precision and recall for a set of m classes are computed as follows:

$$\text{Micro-averaged Precision} = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} TP_i + \sum_{i=1}^{m} FP_i}$$

$$\text{Micro-averaged Recall} = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} TP_i + \sum_{i=1}^{m} FN_i}$$

4 Analysis of Results

In this section we present the analysis and the results obtained, in terms of confusion matrix and other performance measures, for each of the classifiers, in classifying the given data set.

4.1 Data Preparation

In order to make the data suitable for classification we first process the data (both training set and test set). We remove all special characters, numeric characters and punctuations. The stopwords are removed and the words are stemmed to their roots with the help of Porter Stemmer. There are 499 documents in the training set and 185 documents in the test set. After cleaning of data, there are 4647 words in the vocabulary of the training set and 3045 words in the test set.

We convert the training data into the tf-idf matrix and then using this matrix train the classifiers, which we shall apply on the test data to classify them. Information Retrieval research suggests, to avoid unnecessarily large feature vectors, words are considered as features only if they occur in the training data at least 3 times [5]. Thus we shall consider only those features whose document frequency is at least 3.

While creating the tf-idf matrix we consider only unigrams. By using bigrams, researchers obtain a certain improvement in text categorisation results only on rarely used datasets for which the baseline is very low and usually obtained by a weak classification method. On well-known benchmark corpora, such as Reuters-21578, statistically significant improvement has never been reported by research groups that employed bigrams in their document representations [15]. The results obtained by taking bigrams into considerations are presented in Appendix A.

Let us now see the results obtained for the various classifiers.

4.1.1 Multinomial Naïve Bayes Classifier

One of the best classification technique for text categorisation is Multinomial Naïve Bayes Classifier. We perform a 10-fold cross validation to obtain the optimum value of the smoothing parameter alpha. In the grid search we take alpha: [0.001, 0.01, 0.1, 1].

The best alpha is found to be 0.1.

The confusion matrix is as follows:

$$\begin{bmatrix} 16 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 5 \\ 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 5 & 1 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 18 & 0 & 0 & 2 & 4 \\ 0 & 0 & 1 & 0 & 0 & 0 & 3 & 0 & 1 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 24 & 6 \\ 0 & 0 & 2 & 0 & 0 & 1 & 3 & 0 & 8 & 23 \end{bmatrix}$$

There are 128 correct classifications and 57 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-scores are

	Micro	Macro
Precision	0.691892	0.780711
Recall	0.691892	0.641598
F-measure	0.691892	0.663334

4.1.2 Bernoulli Naïve Bayes Classifier

We perform a 10-fold cross validation to obtain the optimum value of the smoothing parameter alpha. In the grid search we take alpha: [0.001, 0.01, 0.1, 1].

The best alpha is found to be 0.01.

The confusion matrix is obtained as follows

$$\begin{bmatrix} 15 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 2 & 2 \\ 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 1 & 25 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 4 & 2 & 0 & 0 & 0 & 2 \\ 0 & 1 & 2 & 0 & 0 & 16 & 0 & 0 & 4 & 1 \\ 0 & 0 & 2 & 0 & 0 & 0 & 4 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 1 & 6 & 0 & 1 & 0 & 0 & 0 & 0 & 18 & 7 \\ 0 & 1 & 3 & 0 & 0 & 0 & 5 & 0 & 9 & 19 \end{bmatrix}$$

There are 113 correct classifications and 72 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-scores are

	Micro	Macro
Precision	0.610811	0.669289
Recall	0.610811	0.596352
F-measure	0.610811	0.611476

4.1.3 k-Nearest Neighbours Classifier

We perform a 10-fold cross validation to obtain the optimum value of k and the best distance metric.

Duda and Hart suggest, albeit in the context of probability density estimation within pattern classification, the use of $k \approx \sqrt{N}$, where N in our case corresponds to the number of neighbours [14]. In our analysis, instead, we vary the range of k from 1 to approximately $\sqrt{N}$. Here, $N = 499 \Longrightarrow \sqrt{N} = 22$. So we in the grid search we take the range of k from 1 to 22. Also, to choose to best distance metric, we grid search through the Euclidean metric, Cosine metric, Minkowski metric and Manhattan metric.

The best value of k was found to be 21 and the best distance metric was found to be Euclidean.

The confusion matrix obtained is the following

$$\begin{bmatrix} 13 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 3 & 5 \\ 0 & 7 & 2 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 4 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 18 & 0 & 0 & 1 & 5 \\ 0 & 0 & 1 & 0 & 0 & 0 & 4 & 0 & 0 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 & 1 & 1 & 0 & 22 & 6 \\ 0 & 0 & 3 & 0 & 0 & 0 & 4 & 1 & 8 & 21 \end{bmatrix}$$

There are 121 correct classifications and 64 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-score are

	Micro	Macro
Precision	0.654054	0.730604
Recall	0.654054	0.649945
F-measure	0.654054	0.634339

4.1.4 Logistic Regression

We perform a 10-fold cross-validation to obtain the best value of the regularization parameter C. In the grid search we take C: [0.001, 0.01, 0.1, 1, 10, 100, 1000].

The best value of C is found to be 1000.

The confusion matrix is as follows

$$\begin{bmatrix} 18 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 3 \\ 0 & 11 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 7 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 23 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 3 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 25 & 5 \\ 0 & 0 & 2 & 0 & 0 & 2 & 3 & 0 & 7 & 23 \end{bmatrix}$$

There are 144 correct classifications and 41 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-score are

	Micro	Macro
Precision	0.778378	0.840262
Recall	0.778378	0.790585
F-measure	0.778378	0.806718

Note that the value of C is quite large. This implies the regularization strength is low. Thus the model is prone to overfitting the data

4.1.5 Support vector Machine

We perform a 10-fold cross validation to obtain the optimum value of the parameter C. Here we use linear kernel to classify our data. This is because most of the text classification data are linearly separable [5]. Also, the linear kernel is good when there are a lot of features. That's because mapping the data to a higher dimensional space does not really improve the performance [10]. In text classification, both the number of instances (documents) and the number of features (words) is very large.

In the grid search we take C: [0.001, 0.01, 0.1, 10, 100, 1000].

The optimum value of C is found to be 10.

The confusion matrix is as follows

$$\begin{bmatrix} 18 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 2 \\ 0 & 11 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 6 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 23 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 23 & 7 \\ 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 7 & 26 \end{bmatrix}$$

There are 139 correct classifications and 46 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-score are

	Micro	Macro
Precision	0.751351	0.776419
Recall	0.751351	0.702633
F-measure	0.751351	0.721057

It must be noted that for each of the classifier, the precision is always high and the recall is always low. This implies amongst the misclassified data points, there are less number of false positives and more number of false negatives. A possible reason could be that the data is unbalanced. A quick check of the training data reveals that that is indeed true. The number of documents belonging to each of the 10 classes in the training set are:

35, 37, 111, 10, 13, 75, 30, 23, 78, 87

5 DISCUSSION AND CONCLUSION

We were given the Reuters 21578 data set for classification using 4 of the most popular classifiers in the supervised learning domain viz., the Naïve Bayes classifier, the k-Nearest Neighbour classifier, the Logistic Regression classifier and the Support Vector Machine.

Based on the measures of f-score, we conclude that logistic regression gives the best performance (macro-averaged f-score 80%) in classifying the test set. But it is to be noted that the optimum value of the parameter C is found to be 1000 which is extremely high. This implies the regularization strength, lambda, is very low (since, $C = \frac{1}{\lambda}$). Thus we can conclude that the model is overfitting the data, i.e. the model fits the training data too well but might not perform well for unseen data. This is undesirable. In order to avoid this overfitting problem, one could try increasing the size of the training samples, and/or dimensionality reduction, i.e. decrease the number of features in the data, and test the logistic classifier again.

The next best performance (macro-averaged f-score 72%) is given by the support vector machine. Here the optimum value of C is obtained as 10. Thus the model sets a soft margin for the separating hyperplane. This implies the model is not prone to over-fitting or under-fitting the data. It is to be noted here that since the model sets a soft margin, it is quite possible that the data is not completely linearly separable. Thus one can check the performance of SVM using the non-linear kernels (see Appendix B). Since the logistic classifier has poor generalization accuracy, we shall consider the performance of SVM as the best for our analysis.

A very important point to be noted is that the Multinomial Naïve Bayes classifier provides a relatively poor result (macro-averaged f-score 66%) which is absurd, since given how large the data set is, multinomial naïve Bayes is theoretically the best classifier for text categorisation. The performance of the Bernoulli Naïve Bayes classifier is the poorest (macro-averaged f-score 61%). It can be justified to some extent given that the data set is large. This is because theoretically Bernoulli naïve Bayes classifier works very well for short texts. Since both the naïve Bayes classifiers generate relatively low results, it may be safe to assume that the features in the dataset are perhaps not mutually independent. One might extend this study further to check for the exact relationship between the features in the dataset. Also, non-linear classification problems can lead to very poor performances of the naïve Bayes classifiers. This once again implies an apparent non-linearity of the given data.

The performance of kNN classifier is not too high (macro-averaged f-score 63%). A plausible explanation for this could be the large number of features in the dataset. Nearest neighbour depends greatly on distances between points. As the number of dimension increases, the distances are going to be less representative. Thus the performance of kNN might be improved through dimensionality reduction.

In conclusion we can say that none of the classifiers perform their very best in classifying the given dataset. However based on preliminary analysis and the above discussions we consider SVM to be the optimum classifier for classification in our study.

6 APPENDICES

APPENDIX A

Performance Measures of the classifiers considering both Unigrams and Bigrams

Multinomial Naïve Bayes

The optimum value alpha is found to be 0.01.

The confusion matrix is as follows:

$$\begin{bmatrix} 18 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 1 \\ 0 & 0 & 27 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 6 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 20 & 0 & 0 & 1 & 3 \\ 0 & 0 & 1 & 0 & 0 & 0 & 4 & 0 & 1 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 25 & 5 \\ 0 & 1 & 2 & 0 & 0 & 0 & 4 & 0 & 8 & 22 \end{bmatrix}$$

There are 133 correct classifications and 52 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-scores are

	Micro	Macro
Precision	0.718919	0.778851
Recall	0.718919	0.675383
F-measure	0.718919	0.689524

Bernoulli Naive Bayes

The best alpha is found to be 0.01.

The confusion matrix is obtained as follows

$$\begin{bmatrix} 17 & 0 & 3 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 1 & 23 & 1 & 0 & 1 & 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 4 & 3 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 & 16 & 0 & 0 & 2 & 3 \\ 0 & 0 & 1 & 0 & 0 & 0 & 3 & 0 & 1 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 21 & 8 \\ 0 & 1 & 2 & 0 & 0 & 0 & 3 & 0 & 9 & 22 \end{bmatrix}$$

There are 118 correct classifications and 67 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-scores are

	Micro	Macro
Precision	0.637838	0.625483
Recall	0.637838	0.605104
F-measure	0.637838	0.584034

k-Nearest Neighbours

The best value of k was found to be 21 and the best distance metric was found to be Euclidean.

The confusion matrix obtained is the following

$$\begin{bmatrix} 14 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 3 & 3 \\ 0 & 7 & 1 & 0 & 0 & 0 & 0 & 0 & 5 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 2 & 0 & 5 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 18 & 0 & 0 & 1 & 5 \\ 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 1 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 & 1 & 0 & 0 & 22 & 6 \\ 0 & 0 & 3 & 0 & 0 & 0 & 2 & 1 & 8 & 23 \end{bmatrix}$$

There are 123 correct classifications and 62 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-score are

	Micro	Macro
Precision	0.664865	0.743395
Recall	0.664865	0.634698
F-measure	0.664865	0.644958

Logistic Regression

The best value of C is found to be 1000.

The confusion matrix is as follows

$$\begin{bmatrix} 19 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 2 \\ 0 & 11 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 6 & 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 23 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 3 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 22 & 8 \\ 0 & 0 & 2 & 0 & 0 & 1 & 3 & 0 & 7 & 24 \end{bmatrix}$$

There are 142 correct classifications and 43 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-score are

	Micro	Macro
Precision	0.767567	0.805986
Recall	0.767567	0.778545
F-measure	0.767567	0.778141

Support Vector Machine

The optimum value of C is found to be 10.

The confusion matrix is as follows

$$\begin{bmatrix} 17 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 3 \\ 0 & 10 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 5 & 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 23 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 23 & 7 \\ 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 8 & 25 \end{bmatrix}$$

There are 139 correct classifications and 46 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-score are

	Micro	Macro
Precision	0.724324	0.766465
Recall	0.724324	0.653439
F-measure	0.724324	0.672239

APPENDIX B

Support Vector Machine using Non-Linear Kernels

We take, in grid search, the non-linear kernels: RBF, polynomial and sigmoid and perform the classifier.

We obtain the best kernel for classification as RBF and the optimum value of C is found to be 1000, which is quite clearly extremely high. This implies the model allows for a very small-margin separating hyperplane.

The confusion matrix is as follows

$$\begin{bmatrix} 18 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 3 \\ 0 & 9 & 1 & 0 & 0 & 0 & 0 & 0 & 3 & 1 \\ 0 & 0 & 28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 0 & 4 & 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 21 & 0 & 0 & 1 & 2 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 23 & 7 \\ 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 7 & 26 \end{bmatrix}$$

There are 133 correct classifications and 52 misclassifications.

The micro-averaged and macro-averaged precision, recall and f-score are as follows

	Micro	Macro
Precision	0.718919	0.765059
Recall	0.718919	0.660014
F-measure	0.718919	0.683135

Based on the f-measure scores we can clearly conclude that the performance of the classifier using non-linear kernel is poorer than its performance while using a linear kernel.

[1] Joachims, Thorsten. “Transductive Inference for Text Classification using Support Vector Machines”.

[2] Scott, Sam, Stan Matwin. “Text Classification using WordNet Hypernyms”.

[3] McCallum A.K. “Multi-label Text Classification with a Mixture Model Trained by EM”.

[4] Sebastini, Fabrizio. “Machine Learning in Automated Text Categorisation”. ACM Computing Surveys. Volume 34 Issue 1, March 2002.

[5] Joachims, Thorsten. “Text Categorisation with Support Vector Machines: Learning with Many Relevant Features”.

[6] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” Information Processing and Management, vol. 24, no. 5, pp. 513–523, 1988.

[7] J. B. Lovins. “Development of a stemming algorithm,” Mechanical Translation and Computational Linguistics, vol. 11, 1968.

[8] M. F. Porter. “An algorithm for suffix stripping,” Program, vol. 14, no. 3, 1980.

[9] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, New York, 2008.

[10] Hsu, Chih-Wei, Chih-Chung Chang and Chih-Jen Lin. “A Practical Guide to Support Vector Classification”.

[11] Celisse, Alain. “Parameter Tuning and Cross Validation Algorithms”.

[12] Arlot, Sylvian, Alain Celisse. “A survey of cross-validation procedures for model selection”. *Statistical Surveys* Vol. 4 (2010) ISSN 1935-7516.

[13] Oprea, Cristina. “Performance Evaluation of the Data Mining Classification Methods”. Annals of the „Constantin Brâncuşi” University of Târgu Jiu, Economy Series, Special Issue (2014): *Information society and sustainable development.*

[14] Duda, R. O. and Hart, P. E. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[15] Bekkerman, Ron, James Allan. “Using Bigrams in Text Categorisation”.