Laboratory 4

Introduction to Artificial Intelligence

Document Classification

Ernesto Vieira Manzanera. Student id: 251663

1. Overview

This paper aims to describe the process of setting up the experiments oriented to analyse two trivial document classification algorithms: Naïve Bayes Classification and Decision Trees Classification. The set of documents used for training and performance testing are obtained from the 20 News Groups, and the categories **`science`, `baseball`, `politics.misc`, `medicine`** and **`mac.hardware`** have been chosen.

2. Text representation and feature selection

In order to represent the documents in a format which is accepted by weka, the following process is carried out:

- Header, footer and quoting elimination from the texts, using the remove parameter in the 20 News Groups fetching function.
- Elimination of escape characters (\n, \t, \r), and non-informative characters . ( ) 's / , == — _ _ * > < , ; ? ¿ `` # ^ | -: : ! ¡ " % { } º
- Splitting of documents in words.
- Filtering of words to avoid unsupported elements: '', ' ', '-', '\x0c'.
- Converting each document to an array of integers representing the occurrence of each word in the text.

Afterwards, an evaluation of each term in terms of each category is performed based on three methods, each of them defining an evaluation function $A(t,c)$ where t is a term and c is a class. Three methods have been tested:

<u>Document Frequency:</u>
Assigns a value to each term which represents the number of documents in a class c that contain t.

<u>Collection Frequency:</u>
Assigns a value to each term which represents the number of occurrences of the term t in the documents belonging to c.

<u>Mutual Information:</u>
Assigns a value to each term t in terms of a class c based on the statistical dependence between those two. The Mutual Information index is defined as follows[1]:

[1] Introduction to Information Retrieval, Christopher D., Manning Prabhakar, Raghavan Hinrich Schütze, Cambridge

$$I(U;C) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} P(U = e_t, C = e_c) \log_2 \frac{P(U = e_t, C = e_c)}{P(U = e_t)P(C = e_c)},$$

However, for implementation purposes, the expression above is rewritten in the way:

$$\begin{aligned} I(U;C) &= \frac{N_{11}}{N} \log_2 \frac{NN_{11}}{N_{1.}N_{.1}} + \frac{N_{01}}{N} \log_2 \frac{NN_{01}}{N_{0.}N_{.1}} \\ &+ \frac{N_{10}}{N} \log_2 \frac{NN_{10}}{N_{1.}N_{.0}} + \frac{N_{00}}{N} \log_2 \frac{NN_{00}}{N_{0.}N_{.0}} \end{aligned}$$

This feature selection method measures the degree of independence between the term and the class, that is to say, to what extent we can infer knowledge about one variable knowing a fact from the other. If two variables are completely independent from each other, the expression inside the logarithmic part will become 1 as the numerator will tend to show the behaviour of the denominator (independent events), which will make the logarithm 0.

These last functions are implemented with the methods `createDocumentFrequency`, `createCollectionFrequency` and `createMutualInformation` respectively, and the three accept the vectorised texts and a list of categories, and output a term ranking for each class based on the above explained methods.

Posteriorly, the function `chooseVocab(vocab, k)` receives a class-term ranking object and a number of terms k to be chosen, and outputs a vocabulary of k words that are most fitted in terms of the previous evaluation.

Lastly, the function createARFF receives the chosen vocabulary and the vectorised texts, and creates a .arff file that is readable by Weka. This function can be set to represent each term in the text with a dichotomous variable (0 or 1) representing the existence or absence of the term in the document, or allowing in to have more values, representing the number of occurrences of the term in the document.

The difference between the functions can be observed easily if we check the terms chosen for each class.

The Figure below shows the first terms chosen by the Document Frequency (2) and Collection Frequency (1) ranker

the	of	to	and	a	in	is	that	i	it	for	this	are	you	be	with	not	have	or	on
5164.0	3276.0	2876.0	2635.0	2482.0	1959.0	1898.0	1521.0	1518.0	1457.0	1084.0	848.0	811.0	786.0	780.0	773.0	724.0	698.0	639.0	629.0
522.0	501.0	477.0	471.0	467.0	459.0	415.0	402.0	395.0	356.0	314.0	299.0	289.0	289.0	284.0	282.0	279.0	268.0	264.0	253.0

The tables below show the terms chosen by the Mutual Information Selector for each class:

sci.med	
gordon	1.1681
shameful	1.1642
intellect	1.1631
surrender	1.1623
banks	1.1608
disease	**1.1574**
doctor	**1.1535**
medical	**1149**
patients	**1145**
soon	1.1414
treatment	**1.1384**
medicine	**1.1378**
patient	**1.1361**
diet	**1.1361**
symptoms	**1.1313**
blood	**1.1312**
she	1.13
pain	**1.13**
syndrome	**1.1298**
food	1.1289

politics.misc	
clinton	**1.6156**
government	**1.615**
tax	**1.6059**
state	**1.6045**
people	1.6041
law	**1.6031**
president	**1.6018**
crime	**1.6002**
economic	**1.5993**
rights	**1.5978**
bush	**1.5973**
congress	**1.5968**
laws	**1.5965**
we	1.5965
taxes	**1.5964**
deficit	**1.5946**
economy	**1.5938**
homosexuals	**1.5937**
federal	**1.593**
house	1.5929

sci.space	
space	**1.2462**
nasa	**1.1832**
launch	**1.1663**
moon	**1.1581**
earth	**1.152**
flight	**1.1393**
satellite	**1.136**
mission	**1.1355**
station	**1.132**
commercial	1.1316
mars	**1.1312**
project	1.1311
launched	**1.1302**
rocket	**1.1302**
ames	1.129
astronomy	**1.1281**
sun	**1.1274**
vehicle	**1.1273**
exploration	**1.1271**
apollo	**1.1264**

mac.hardware	
mac	**1.251**
apple	**1.2425**
drive	**1.2063**
video	**1.1917**
he	1.1901
ram	**1.1885**
monitor	**1.1873**
card	1.1866
his	1.1857
thanks	1.1839
disk	**1.1836**
was	1.1833
chip	**1.1819**
software	**1.1798**
memory	**1.1796**
upgrade	**1.1782**
se	1.1779
slot	**1.1778**
who	1.1766
modem	**1.1765**

sport.baseball	
baseball	**1.1661**
game	**1.1639**
games	**1.1597**
season	**1.1544**
team	**1.1499**
players	**1.1467**
league	**1.1459**
pitching	**1.1421**
he	1.1349
runs	**1.1326**
teams	**1.1324**
player	**1.1323**
hit	1.1311
win	**1.1305**
mets	**1.1266**
year	1.1259
fans	**1.1226**
it	1.122
his	1.1219
sox	1.121

3. Naïve Bayes Classification

Bayes-based classification algorithms are based on the evidence-based knowledge relation provided by Bayes' Theorem. Given a set of documents $\mathbb{X}$, and a set of classes $\mathbb{C}$, text classification algorithms' goal is to find a function $\gamma : \mathbb{X} \mapsto \mathbb{C}$ as a result of the application of a Learning Method to a training set $\mathbb{D} = \{\langle d, c\rangle, \langle d, c\rangle \in \mathbb{X} \times \mathbb{C}\}$:

$$\Gamma(\mathbb{D}) = \gamma$$

In this case, the method is based on finding the class that maximises the conditional probability of a class being a document's class:

$$argmax_{c\in\mathbb{C}}\ P(c\,|\,d) = argmax_{c\in\mathbb{C}}\ \frac{P(c)\ P(d\,|\,c)}{P(d)} = argmax_{c\in\mathbb{C}}\ P(c)\ P(d\,|\,c)$$

As every document d can be represented as a sequence of terms $\langle t_1 \ldots t_n\rangle$, the previous expression can be rewritten to be expressed in terms of those terms. Furthermore, Naïve Bayes is based on the **Assumption of Conditional Independence**, by which the existence of a term is not determined by the other terms. Thus, the Bayes method can be expressed:

$$P(d\,|\,c) = P(\langle t_1 \ldots t_n\rangle\,|\,c) = \prod P(t_k\,|\,c)$$

$$argmax_{c\in\mathbb{C}}\ P(c)\ P(d\,|\,c) = argmax_{c\in\mathbb{C}}\ P(c)\ \prod P(t_k\,|\,c) = argmax_{c\in\mathbb{C}}\ \log(P(c)) + \sum \log(P(t_k\,|\,c))$$

Lastly, Bayes' method relies on one more assumption: the **Assumption of Positional Independence**, by which the position of the term in the text does not affect the probability of it belonging to a specific class.

Experiments on the Bayes' model are carried out with the NaiveBayes classifier implemented in Weka software. In order to perform the tests, k-fold cross-validating in included. The effect of k's choice in the classifier's performance is presented in the following figures:

5000 words, Collection Frequency Feature Selection, RawNaiveBayes			
k = 3	Correctly Classified Instances Incorrectly Classified Instances	1540 1287	54.4747 % 45.5253 %
k = 5	Correctly Classified Instances Incorrectly Classified Instances	1581 1246	55.925 % 44.075 %
k = 7	Correctly Classified Instances Incorrectly Classified Instances	1587 1240	56.1372 % 43.8628 %
k = 10	Correctly Classified Instances Incorrectly Classified Instances	1605 1222	56.774 % 43.226 %

k = 13	Correctly Classified Instances Incorrectly Classified Instances	1608 1219	56.8801 % 43.1199 %
k = 15	Correctly Classified Instances Incorrectly Classified Instances	1588 1239	56.1726 % 43.8274 %
k = 20	Correctly Classified Instances Incorrectly Classified Instances	1608 1219	56.8801 % 43.1199 %

The previous test where made on a 1000 words feature set, where each feature represents the amount of repetitions of the word in the document. As we can see, the accuracy of the classifier raises up when using higher number of folds. However, from k = 10-13 folds, the behaviour starts declining, but no significantly. Experimentally, it has been showed that the k value for cross validation should be around 10 folds.

The number of words of the training set can also be decisive when performing training. This time, Multinomial Bayes Classifier has been used with the Mutual Information feature selection.

1000 words (MNB)	Correctly Classified Instances Incorrectly Classified Instances	1633 1194	57.7644 % 42.2356 %
2500 words (MNB)	Correctly Classified Instances Incorrectly Classified Instances	1713 1114	60.5943 % 39.4057 %
6500 words(MNB)	Correctly Classified Instances Incorrectly Classified Instances	2305 522	81.5352 % 18.4648 %
10000 words (MNB)	Correctly Classified Instances Incorrectly Classified Instances	2168 659	76.6891 % 23.3109 %

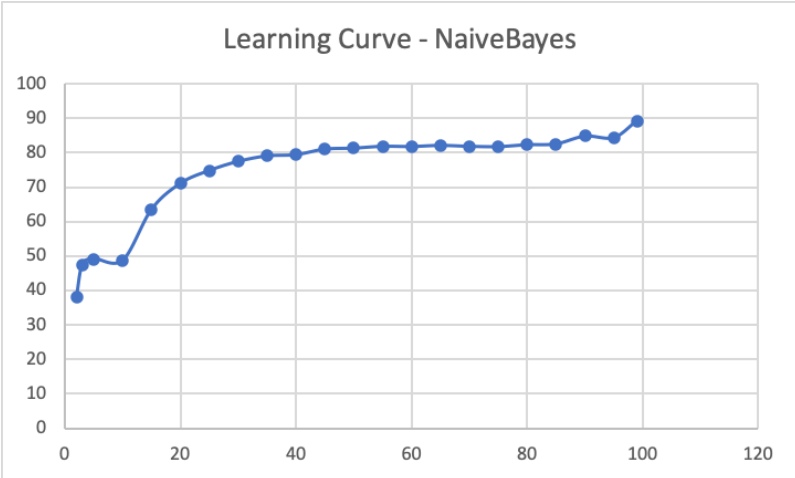
As we can see, the number of words affects positively on the performance of the algorithm up to a certain threshold, beyond which the quality start declining. This could be due to the existence of more variability and more variety of terms that confuses the algorithm.

Feature selection also plays a very important role when it comes to achieve a high quality generalisation of the documents. The three feature selection methods previously proposed are tested with a vocabulary size of 10000 words in a multinomial Bayes Classifier, obtaining the following results:

Mutual Information	Correctly Classified Instances Incorrectly Classified Instances	2327 500	82.3134 % 17.6866 %
Collection Frequency	Correctly Classified Instances Incorrectly Classified Instances	2471 356	87.4071 % 12.5929 %
Document Frequency	Correctly Classified Instances Incorrectly Classified Instances	2477 350	87.6194 % 12.3806 %

<u>Learning Curve</u>

For NaiveBayes Mutinomial Classifier, the learning curve depending on the amount of training data is the following:



As we can see, the multinomial Bayes Classifier performs highly better than the simple NaiveBayes. This is due to the nature of these two classifiers. Whereas a Bernoulli NaiveClassifier only takes into account if a specific features takes place or not, multinomial adds extra information by considering how many that feature takes place. However, dummy words with no meaning, and which appear more often in the whole dataset lead the classifier to wrong conclusions and generalizations.

When it comes to feature selection, we observe that although Mutual Information selection provides terms that concisely determine the classes, the overall performance is worse. That could be due to the fact that certain array of characters appear in few documents (which is translated in high dependance with the class), but does does not genuinely give any information about the target class. One way of overcoming the issue could be by narrowing the selected vocabulary while assuring that the the selected words are highly representative, eliminating accidental strings of characters that although rare, do not allow us to make any inference of the class.

Regarding to k-cross fold-validation technique, it is common in literature to find a recommended value of k = 10. Empirically we have observed some evidence that supports it, as the overall performance of the classifier increased up to k = 10-13. However, if we increase k to a higher number, the division sets are smaller and might lead to wrong generalizations.

4. Decision Tree Classification

Decision tree classification models use decision tree structures in order to assign a class based on a series of inputs. In the tree we can distinguish three different elements:

- Leaf nodes: represents a class upon which the decision its stablished.
- Non-leaf nodes: represents an attribute.
- Branches: represent a single value for the father node.

At each non-leaf node, the attribute corresponding to the node is decided based on the classification features of the available attributes.

Weka's implementation of C4.5 algorithm can be found under the name J48. At each tree level, the algorithm chooses the attribute that divides the data most efficiently. For such goal, C4.5 uses the concept of Information Gain, which is defined as the difference of entropy of a given dataset as a result of a transformation. As lower entropies yield more information about the data, at each step entropy minimisation is desired in order to favour clustering. Information gain of a document set for an attribute a is defined as:

$$IG(S,a) = E(S) - \sum_{v \in Values(a)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Where the first term is the entropy of the whole document and the second is the normalised entropy of the documents conditioned to the appearance of a.

J48 allows the user to classify a training set using the C4.5 Algorithm, while establishing different values for its variables:
- confidenceFactor: determines the error threshold while pruning the tree.
- binarySplits: set True for building binary trees, or False otherwise.
- minNumOb: the minimum number of objects at each leaf.
- numFolds: determines the size of the pruning set.
- SubtreeRaising: whether to use os nor subtree raising operation.
- usePruning: whether to use pruning or not.
- useLaplace: use Laplace smoothing.

5000 words. Confidence = 0.25	
Pruned, Minimum 2, No_subtree_raising.	Correctly Classified Instances 1751 61.9385 % Incorrectly Classified Instances 1076 38.0615 %
Pruned, Minimum 2, No_subtree_raising, Laplace_smoothing	Correctly Classified Instances 1748 61.8323 % Incorrectly Classified Instances 1079 38.1677 %
Not_pruned, Minimum 2, Subtree_raising, Laplace smoothing.	Correctly Classified Instances 1737 61.4432 % Incorrectly Classified Instances 1090 38.5568 %
Pruned, Minimum 2, No_subtree_raising, Laplace_smoothing, 5 folds	Correctly Classified Instances 1737 61.4432 % Incorrectly Classified Instances 1090 38.5568 %
Pruned, Minimum 8, subtree raising, Laplace smoothing, 7	Correctly Classified Instances 1750 61.9031 % Incorrectly Classified Instances 1077 38.0969 %

When the SubtreeRaising option is set off, the size of tree quickly escalates. We observe that we get a tree size of 711, and a number of leaves of 356. However, the overcall quality of the algorithm is kept the same. Moreover, when pruning is not chosen, the classification takes considerably more time, but the performance is barely affected. We can conclude that in terms of quality, the pruning of the tree does not yield much better results.

Following, analysis of the number of chosen words and selection method is included. Other parameters are set to the default configuration. The first table runs analysis allowing more than two branches per node, making full use of the multinomial representation of the text. The second table shows the performance of a J48 algorithm with binary branching.

Binary Splitting			
250 words, Mutual Information Selection.	Correctly Classified Instances Incorrectly Classified Instances	1778 1049	62.8935 % 37.1065 %
1000 words, Mutual Information Selection.	Correctly Classified Instances Incorrectly Classified Instances	1769 1058	62.5752 % 37.4248 %
1000 words, Collection Frequency Selection.	Correctly Classified Instances Incorrectly Classified Instances	1789 1038	63.2826 % 36.7174 %
1000 words, Document Frequency Selection.	Correctly Classified Instances Incorrectly Classified Instances	1788 1039	63.2473 % 36.7527 %
5000 words, Document Frequency Selection.	Correctly Classified Instances Incorrectly Classified Instances	1794 1033	63.4595 % 36.5405 %

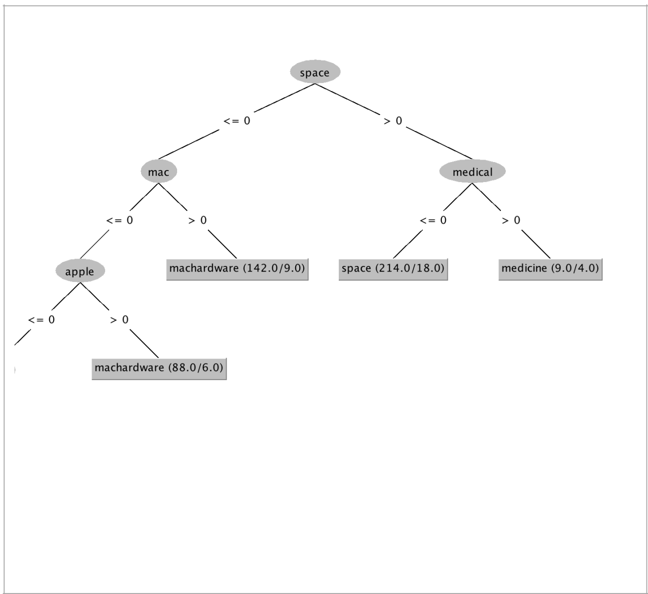
Non-Binary Splitting			
5000 words, Collection Frequency Selection.	Correctly Classified Instances Incorrectly Classified Instances	1728 1099	61.1249 % 38.8751 %
1000 words, Collection Frequency Selection.	Correctly Classified Instances Incorrectly Classified Instances	1767 1060	62.5044 % 37.4956 %
1000 words, Mutual Information Selection.	Correctly Classified Instances Incorrectly Classified Instances	1768 1059	62.5398 % 37.4602 %
500 words, Mutual Information Selection.	Correctly Classified Instances Incorrectly Classified Instances	1789 1038	63.2826 % 36.7174 %
250 words, Mutual Information Selection.	Correctly Classified Instances Incorrectly Classified Instances	1778 1049	62.8935 % 37.1065 %

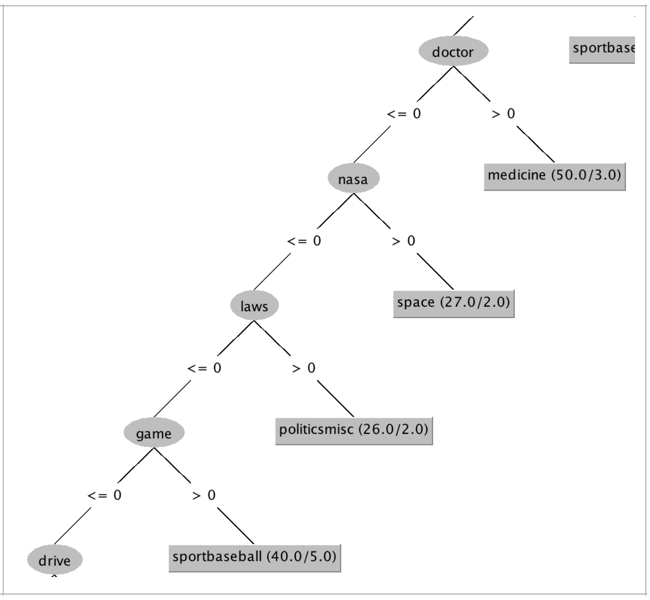
Lastly, feature selection methods were implemented in order to detect the most important words when it comes to class definition. We observe that those words are naturally found by the algorithm, as we can see that the top nodes include the attributes with higher raking values.

Information Gain	
356	orbit
352	space
348	skepticism
347	chastity
347	n3jxp
346	geb@cadredslpittedu
341	apple
0.34	nasa
334	shuttle
332	shameful
329	season
323	scsi
322	surrender
0.32	quadra
319	launch
318	intellect
316	lunar
314	senate
313	braves
311	simms
0.31	spacecraft
309	pitching
308	crime
303	lc
302	disease

Correlation	
0.1314	skepticism
0.1305	chastity
0.1305	n3jxp
0.1295	geb@cadredslpittedu
0.1283	shameful
0.1277	intellect
0.1271	surrender
0.1161	banks
0.1105	apple
0.1013	team
0.0949	quadra
0.0946	season
0.0945	pitching
0.0939	space
0.0916	players
0.0877	clinton
0.0875	braves
0.0841	player
0.0839	treatment
0.0824	mets
0.0821	orbit
0.0817	medicine
0.0797	disease
0.0795	nasa
0.0793	simms

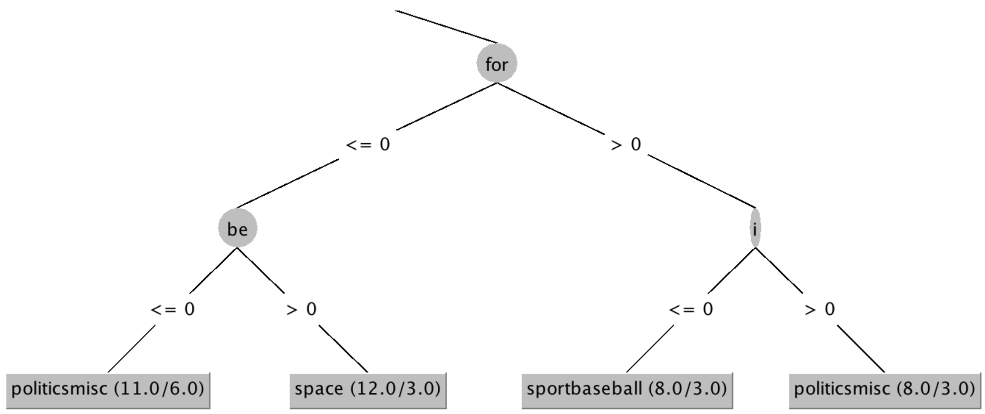
Chi-squared	
673.4904	space
423.2425	apple
353.5819	nasa
310.1599	orbit
287.1046	banks
285.6627	skepticism
281.9138	season
281.7001	n3jxp
281.7001	chastity
277.7404	geb@cadredslpittedu
277.2582	team
272.0817	shameful
271.0674	launch
270.4633	intellect
266.7156	surrender
247.3377	clinton
245.5313	disease
241.3684	players
238.8091	shuttle
216.3159	medical
211.686	pitching
198.0531	scsi
190.7341	tax
189.9941	quadra
189.4351	patients





We can observe that the J48 algorithm finds the terms that most partitions the document set. The two figures above show the tree from the root node, and the first level of classification. If we analyse the terms that have been chosen to be as first nodes, we notice that they are no trivial, and in terms of meaning, they do provide information about the target class. Furthermore, we can see that those terms are included in the first 20 terms selected by the Mutual Information Ranker algorithm that was presented in the second point.

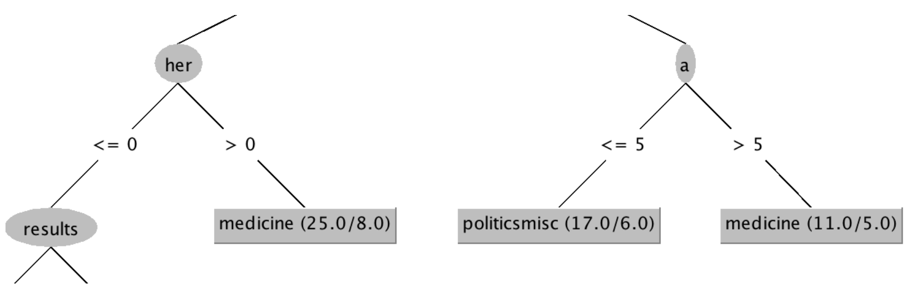
If we now go deep in the tree, we observe that the decision attributes now become more abstract and meaningless, as they cannot define a class by its own meaning. An example os such phenomenon is shown in the next Figure:



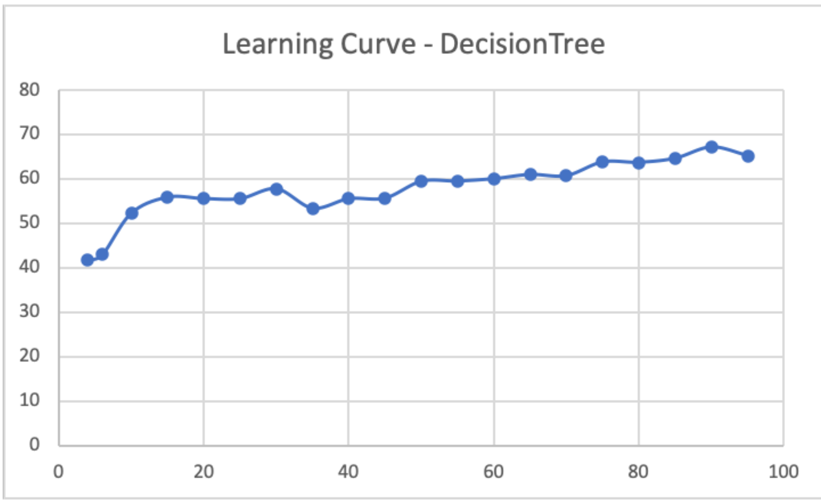
Here, we observe how the words for, be and I are used to distinguish between classes, but in fact the carry no information whatsoever.

If we now run the same algorithm with the same settings over a multinomial representation of the texts we obtain the results:

```
Correctly Classified Instances         1584               56.0311 %
Incorrectly Classified Instances       1243               43.9689 %
Number of Leaves  :     105
Size of the tree :      209
```



We can see that the results worsen although more information is provided (i.e the number of occurrences of the term in the document). However, figures like the one above shows that this information is frequently used in words that carry no meaning, and thus the improvement is no noticeable. What is more, although the binarySplits is set of off, the tree never yields more than two children for one node, which can be interpreted as the algorithm finding no useful information in the discretisation of values higher than 1.



5. Random Tree Classifier

Another algorithm that could be studied is Random Tree Classifier. This learning method is based on creating a set of n trees which model a section of the training data. Although the single performance of a single tree can be cuestionable, the final decision is made based on the partial decisions of each tree, as a result of a divide and conquer problem.

Random Tree Classifier is run with a feature set of 10000 words, applying a 10-fold cross validation, with 100 trees generated in the forest. The performance, compared to the basic Decision tree, is highly better.

```
Correctly Classified Instances        2126               75.2034 %
Incorrectly Classified Instances       701               24.7966 %
```

With a tree number of 50 trees, the performance is as follow:

```
Correctly Classified Instances        2043               72.2674 %
Incorrectly Classified Instances       784               27.7326 %
```

If we decrease the number of generated trees to 10 trees

```
Correctly Classified Instances        1524               53.9087 %
Incorrectly Classified Instances      1303               46.0913 %
```

6. Model Overfitting

During the experiments, three methods of feature selection have been presented in order to validate the models with different degrees of specific vocabulary. For the methods of

Document Frequency and Collection Frequency, it can be observed that very common terms like “is”, “for”, “to”, “he”, “she”, etc, that carry almost no meaning have been included. The presence of those terms has been overcame by the algorithms as they have been able to generalise the classes up to a considerable performance (in some example 88% of accuracy).

However, the Mutual Information Selection method, although gave very concise vocabulary, obtained less accuracy when k-cross validation was performed. Instead of the 20 words lists that were presented previously, the algorithms was inputted with an average of 1000 words representative of the each class. Overviewing that list, we come accross many numerical values that are set as attributes, which can mislead the classifier provoking overfitting. Although numerical values are very specific of the documents, and can help to determine the class, they are not good for generalisation of the documents as they provide out-of-context information.

Typos and word modifications are not well generalised by the algorithms either, as we can see in the 20 chosen terms list for baseball category. The terms “player” and “players” are selected by the feature selection, but do not add much help in the decision-making process.

7. Model Choosing

We have observed the performance of two classification methods on a specific set of tests. Yielded results suggest that Naïve Bayes Classifier achieves higher accuracy percentages than Decision Trees. Furthermore, when it comes to knowledge representation, multinomial version of Bayes Classifier outputs considerably better results.

When it comes to feature selection, a greedy selection was implemented with the createMutualInformation function, which provided specific terms that supported classification process. At the same time, feature selection with Weka tools yielded similar words to the one selected by the Mutual Information technique.

Finally, it order to validate our models, k-cross validation has been carried out in all the experiments. This method can help to detect overfitting, and the value k has been empirically showed to be recommendable around 10-13.