Computer and Information Science; Vol. 5, No. 6; 2012
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

Evaluation of Efficiency of Linear Techniques to Optimize Attribute Space in Machine Learning: Relevant Results for Extractive Methods of Summarizing

Jesus Antonio Motta[1], Laurence Capus[1] & Nicole Tourigny[1]

[1] Département d'informatique et de génie logiciel, Université Laval, Québec (QC), Canada

Correspondence: Laurence Capus, Département d'informatique et de génie logiciel, Université Laval, Québec (QC), Canada. E-mail: laurence.capus@ift.ulaval.ca

Received: June 12, 2012 Accepted: July 9, 2012 Online Published: October 21, 2012
doi:10.5539/cis.v5n6p58 URL: http://dx.doi.org/10.5539/cis.v5n6p58

Abstract

One major challenge in the field of machine learning, especially in classification problems, is to optimize the attribute space in order to obtain a classification function, which will be used to discriminate future items. Several approaches to optimize the attribute space can be used: some of them select the most relevant attributes and the other ones extract certain attributes to create a new smaller set of variables. These classification approaches have recently been implemented in the automatic summarization process with promising results. This paper enriches these first results with another new experiment. Five well-known linear methods were exploited to optimize the attribute space in an original manner on a corpus of 1250 text documents. These methods, used in data clustering and unsupervised machine learning, allow either attribute selection (Singular Value Decomposition, K-Means, Kohonen Neural Networks) or new attribute extraction (Principal Component Analysis, Factor Analysis). After having applied these methods to optimize attribute space, the validation phase was focused on the discrimination power of the obtained classification function. For that, six techniques of machine learning were used to abduce the classification function. Its performance was evaluated with the metric F_{mesure} and ROC curves. The results show that the application of the five chosen linear methods for optimizing attribute space in the automatic summarization process by extraction is relevant. They also show which machine learning technique is preferable to use with each linear method to obtain a better efficiency.

Keywords: attribute space, machine learning, classification, abduction, automatic summarization

1. Introduction

In machine learning problems and specially classification problems, a space of concepts, variables or features, is used to induce or to abduce the classification function. This space may be used to extract the training set during the inductive/abductive process. Unfortunately, this initial space is generally too large and entropic. In other words, this space contains too much noise and numerous irrelevant features, thus creating the well-known problem of the curse of dimensionality (Smale, 1997). When data is too scattered, good estimations are not easier and consequently the elaboration of good classification models. If one applies a machine learning method to such spaces, one could have an over-fitting problem of the training set, which will generalize poorly the new items or examples to be classified (Langley, 1994). In contrast, if one tries to travel through the entire space or a large part of this space in order to find optimal solutions, one will be face to a polynomial, exponential or combinatorial explosion problem of the search time and the saving space of intermediate states, making this problem intractable, in most of cases. Let us remind that an intractable problem is a polynomial problem with a solution in theory but not in practice (Hastie et al., 2009).

Several approaches are used to optimize the attribute space either by selecting the most relevant attributes or by extracting certain attributes to create a new smaller set of variables. This way of doing has been recently apply to automatic summarization process (Motta et al., 2011). In the present paper, we propose to enrich these first results by adding an experiment we conducted with methods for optimizing attribute space, well-known but not enough used in this field. We have chosen five methods: three to select attributes that are Singular Value Decomposition (SVD) (Golub & Van Loan, 1996; Stewart, 1973), K-Means (KM) (Hastie et al., 2009) and Kohonen Neural Networks (KNN) (Kohonen, 1990), and two others that allow attributes extraction, i.e. Principal

Component Analysis (PCA) (Jolliffe, 2002) and Factor Analysis (FA) (Kim & Mueller, 1978). These methods commonly used for data clustering and unsupervised machine learning will be described in the next section.

From a corpus of 1250 textual documents from DUC 2006 (NIST, 2006), we applied these five methods to optimize the space of attributes. In fact, this attribute space is a matrix, where the rows represent the sentences of the documents and the columns are the words of these sentences. Each item of the matrix corresponds to the frequency of the word in the sentence in a vector space model (Motta et al., 2011). By applying one of the five methods on this matrix, we obtained an optimized attribute space composed of sentences with important information. Rather than evaluating produced summaries, we induced a classification function and assessed its performance. This way, we were able to determine the power to discriminate of the five chosen methods in order to optimize the attribute space, given that a good performance of these functions depends largely on the choice of the training set (Ikonomakis et al., 2005). By focusing our interest on this principle, we did not thus used methods that allow the evaluation of the quality of produced summaries, as ROUGE (Lin, 2004) for instance. We preferred this other protocol of validation. First, we applied six current techniques of data mining on the optimized attribute space to induce a classification function. Next, we evaluated the classification performance with the metric F_{mesure} (harmonic mean of precision and recall) and ROC curves. We can conclude that the five chosen methods are relevant and appropriate with the application of machine learning techniques for automatic summarization process by extraction.

Section 2 presents an overview of the chosen reduction methods. Section 3 describes the experiment that we conducted. Section 4 shows the tables and graphs produced with the experimental values and the discussion of the results obtained. Section 5 concludes on the importance of such an experiment.

2. The Five Methods Chosen for Our Experiment

This section presents the five methods we used for the reduction of the attribute space and how we applied them. To well understand, it is important to know that each method was applied to a matrix E in the vector space model, where the rows represent the sentences of the documents and the columns are the set of attributes or vocabulary of the corpus of documents. The presence of every word in every sentence of the corpus of documents has been measured by multiplying the word frequency with the frequency inverse of the words in the document, i.e.:

$$tf_t \times idf = tf_t \times \ln\left(\frac{|P|}{|\{t \in p\}|}\right)$$

where tf_t is the frequency of word *t* in the sentence *f*, $|P|$ is the totalnumber of sentences in the corpus and $|\{t \in p\}|$ is the number of sentences containing the word *t*.

To apply the five methods on this matrix, we then created algorithms based on them. Following, we explain the principle of each method and how we applied it to our research.

2.1 K-means (KM) (Hastie et al., 2009)

It is a kind of clustering based on centroids, which groups *n* objects or points $(x_1, x_2, \dots, x_n)$ that have certain characteristics in *k* partitions, groups or clusters $(k < n)$. The objects in the cluster are correlated with it, but they are not correlated with other clusters. The optimization problem that arises is to find those *k* clusters of the set $S = \{S_1, S_2, \dots, S_k\}$ and their centers, to assign the objects to the nearest cluster center, so that the square of their distances from the cluster is minimal. That is, it tries to minimize total intra - cluster variance or square error function:

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} \left(x_j - \mu_i\right)^2$$

where there are *k* clusters S_i, $i = 1, 2, \dots, k$ and μ_i is the centroid or mean point of all points x_j in S_i. A simple algorithm would consist in determining the centroid coordinates and the distance between each object and the centroid, and next grouping the objects with the minimum distance. This is a heuristic algorithm and there is no guarantee that it converges to a global optimum. The distance between points and the centroid can be obtained in several ways: Euclidean, sum of absolute differences, cosine (*1 – angle between points*), correlation (*1 – correlation between points*), Hamming (*percentage of bits that differ*) and others. Obviously the cluster centroids are different depending on the distance measure used.

In our research, as already mentioned, the items of the matrix *E* are calculated by using the formula $tf * idf$. Our clustering technique is applied to this matrix defining $k = 2$ groups. The idea is to partition the matrix

E into two groups that correspond to the very important sentences in an information viewpoint and the less important sentences. This way, we can discriminate the relevance of the sentences of the corpus in order to eliminate those that provide less information. For the process of discrimination of sentences, we find the silhouette (Rousseeuw, 1986) of each cluster and identify the sentences that have greater dissimilarity with the neighboring cluster. Actually, separating the important sentences from those unimportant is a difficult process because in reality it can be subjective and involves a high degree of abstraction. For this reason, we preferred to use the concepts of 'very important sentences', and 'less important sentences', and identify more clearly their group membership. To represent the silhouette $s(i)$ of a cluster, we proceed as follows:

Let:

- i an object inthe data set
- A the cluster to which i has been assigned
- $a(i)$ the mean dissimilarity of i to all other objects of A
- C a different cluster of A
- $d(i, C)$ the average dissimilarity of i to all objects of C
- $b(i) = \min_{C \neq A} d(i, C)$
- B the cluster, where the minimum is reached (i.e.$d(i, B) = b(i)$)

Then the silhouette$s(i)$ obtained by combining $a(i)$ and $b(i)$ is as follows:

$$s(i) = \begin{cases} 1 - \dfrac{a(i)}{b(i)} & if\ a(i) < b(i) \\ 0 & if\ a(i) = b(i) \\ \dfrac{b(i)}{a(i)} - 1 & if\ a(i) > b(i) \end{cases}$$

where one can see clearly that $1 \leq s(i) \leq 1$ for each object *i*.

To analyze how $s(i)$ works, one takes for instance the case when $s(i)$ is close to 1, that is $s(i)$ has its maximum value. This implies that the intra-cluster dissimilarity $a(i)$ is much smaller than the inter-cluster dissimilarity $b(i)$. One can say *I* is 'well clustered' and doubt is small compared to the question whether it was well clustered, because the other cluster in any case is far from the first. When $s(i)$ is closeto 0, then $a(i)$ and $b(i)$ are approximately equal and there is no clarity to what cluster allocates *i*. When $s(i)$ is negative andclose to -1, then $a(i)$ is much larger than $b(i)$, so on average *i* is closer to *B* than *A*. It would have been then better to assign *i* to *B* before than *A*. One could conclude that this object has been wrongly classified.

2.2 Kohonen Neural Networks (KNN) (Kohonen, 1990)

It is an artificial neural network and as such it must find common features, regularities, correlations or categories in the input data and incorporates them into its internal structure of connections. Therefore, neurons are self-organized according to stimuli from outside.

In this type of process, neurons compete with one another to advance a given task. When they discover an input pattern, only one neuron (BMU, Best Matching Unit) or group of neighboring neurons is activated. Neurons compete for being activated and at the end only one is winner and the others are forced to produce a minimum response level. The ultimate goal of this process is categorizing the data which enters the network. Similar values are classified into the same category and thus should activate the same output neuron. In essence, it is intended to map similar patterns in the input signal space (vectors patterns) in contiguous locations in the output space that is much smaller.

The K-NN method consists of an input layer composed of *N* neurons (one for each input variable), which receives and transmits to the output layer (made up of *M* neurons) outside information. This last layer processes information and forms the pattern map. Each input neuron *i* is connected to an output neuron *j* through aW_{ji}weight. In this way, the output neurons have an associated weighting vector W_j,called reference vector that corresponds to an average of the category represented by the output neuron *j*. As one can see in Figure1, each node has a specific topological position (*x* and *y*, its coordinates in the lattice) and contains a weight vector of the same dimension of input vectors.

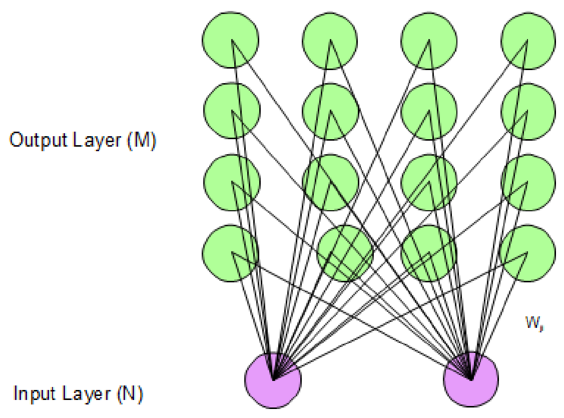

Figure 1. A simple Kohonen Neural Nerwork

The classification process of the network corresponds to the SOM algorithm. First, it chooses at random a weight vector of the nodes W_j. It makes a loop through the nodes to choose an input vector x using Euclidean distance to find the similarity between this vector and vector weights W_j. Next, it identifies the node that produces the smallest distance (node BMU) and moves this node BMU and its neighbors near the input vector x. This approach as a time function t is called the learning rate $\alpha(t)$. As this approach is being updated and new vectors are assigned to the map, the learning rate approaches 0. Along with it, the radius of neighborhood also decreases. This update is performed using the following formula:

$$W_j(t+1) = \begin{cases} W_j(t) + \alpha(t)\Theta(t)(x(t) - W_j(t)) & if\ j \in N_j(t) \\ W_j(t) & if\ j \notin N_j(t) \end{cases}$$

where t is the current iteration, N_j is the vicinity of the neuron j, $\Theta(t)$ is the neighborhood function. Finally, the SOM algorithm increases t and repeats from the loop to $t < \lambda$ (time limit). The number of iterations is fixed a priori. Once the process is finished, the map is sorted topologically speaking, grouping n vectors corresponding to n next adjacent neurons or in the same neuron.

In our research, we started from the matrix E and defined the number of neurons equal to the number of words (variables). We tried to discriminate the important sentences of those less significant from the vector of centroids of the output neurons, in order to find the correlation of each variable to the neuron and then collect the sentences that contain these variables.

2.3 Factor Analysis (FA) (Kim & Mueller, 1978)

This statistical method attempts to describe the influence between correlated variables and uncorrelated latent variables or constructs called factors. It distinguishes between common variance and unique variance. Common variance is the part of the variation of the variable that is shared with the other variables. The unique variance is the variation of the variable that is unique to that variable. The FA method aims to find a new set of variables, fewer in number than the original variables, to express what is common to these variables.

There are basically two types of factor analysis: exploratory, which attempts to discover the nature of the latent variables that influence a set of responses, and confirmatory that tests whether a set of constructs has influence over a set of answers. These two types are based on the model shown in Figure 2 (Joreskog & Van Thillo, 2010; Kim & Mueller, 1978). This model proposes that each response observed from measure 1 to 5 is influenced in part by the latent factors Factor 1 and Factor 2 and the errors E1 to E5.

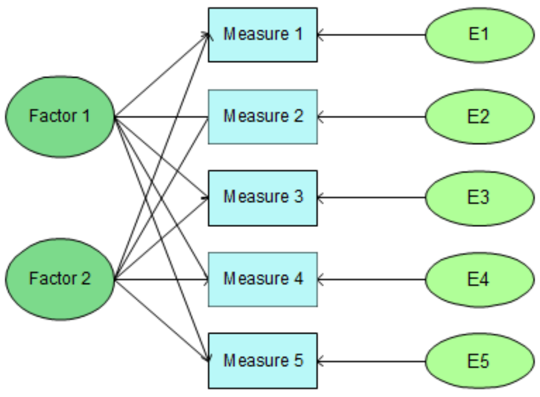

Figure 2. Model for the two types of factor analysis

The FA method is performed by finding the correlation patterns or covariance between the values of measured variables. The high correlation measures are the most likely influenced by the same factors, while measures with low correlation are probably influenced by other factors.

Formalizing these above ideas:

Let $X_1, X_2, \ldots, X_p$ be the variables under analysis, then one can express the components (measurements) of the first variable (vector) X_1 as:

$$\begin{gathered} x_{11} - \mu_1 = \lambda_{11}F_1 + \lambda_{12}F_2 + \cdots + \lambda_{1k}F_k + \varepsilon_1 \\ x_{21} - \mu_1 = \lambda_{21}F_1 + \lambda_{22}F_2 + \cdots + \lambda_{2k}F_k + \varepsilon_2 \\ \cdots \\ x_{p1} - \mu_1 = \lambda_{p1}F_1 + \lambda_{p2}F_2 + \cdots + \lambda_{pk}F_k + \varepsilon_p \end{gathered}$$

where $F_1, \ldots, F_k$are the common factors, $\mu_1$is the mean of the variable X_1 and the coefficients $\lambda_{ij}(i = 1, \ldots, p;\ j = 1, \ldots, k)$ are the factor loadings.

Generally, one can write:

$$x_i = \sum_{j=1}^{k} \lambda_{ij} f_j + \varepsilon_i + \mu_i$$

The k factors $F_1, \ldots, F_k$ are assumed to be uncorrelated random variables with mean 0 and variance 1. The errors $\varepsilon_1, \ldots, \varepsilon_p$ are uncorrelated random errors with mean 0and variances $\psi_1 \ldots \psi_p$, as well as the ε_i are uncorrelated with the factors F_j.

One can write the above equation in matrix form as follows:

$$X = \Lambda f + u + \mu$$

Where X, ε and μ are vectors of dimension $(p \times 1)$ and f is also a vector of dimension $(k \times 1)$. Λ is a matrix of unknown constants $p \times k$ $(k < p)$, called the matrix of factor loadings.

As stated:

$$Var(\varepsilon_i) = \psi_i$$

then:

$$Var(\varepsilon) = \psi = Diag\big(Cov(\varepsilon)\big) = \psi_1, \psi_2, \ldots, \psi_p.$$

Assuming also that$Cov(F) = 1$and, as established $Cov(j, u) = 0,$ then a solution to the set of equations defined above subject to the restrictions for F, is the factors and Λ is the loading matrix.

In our research, as proceeding, we applied the FA method on the matrix E. We tried to discover the latent patterns that could discriminate the sentences carrying information of those phrases that do not carry it.

2.4 Principal Components Analysis (PCA) (Jolliffe, 2002)

This method studies the relationships that exist between p correlated variables, finding another set of new uncorrelated variables called principal components. These components successively explain most of the total variance, unlike the previous method that distinguishes two types of variance: common and unique. The new variables are linear combinations of the foregoing and are constructed successively in order of significance, determined by extracting the total variability of the sample. In other words, the method seeks to find $m < p$ variables that are combinations of the original p variables which are not correlated and containing the most of the information or data variability.

By specifying the above ideas, one could have:

Consider a number of variables $x_1, x_2, \dots, x_p$ on a group of objects or individuals. From this group, calculate a new set of variables $y_1, y_2, \dots, y_p$ that do not correlate with each other and whose variances successively will decrease. The first variable contains all the possible variation of the total amount of variation; the second contains the largest possible amount of the remaining variation and so on.

Each y_j $(j = 1, \dots, p)$ is a linear combination of the original x_i as follows:

$$y_j = a_{j1}x_1 + a_{j2}x_2 + \dots + a_{jp}x_p = a_j x$$

where $a'_j = [a_{1j}, a_{2j}, \dots, a_{pj}]$ is a constant vector and $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}$.

As it is necessary to ensure the orthogonality of the transformation (the new variables are uncorrelated, and its magnitude is unitary), then the norm of the vector a'_j must be equal to 1, i.e.:

$$a'_j a_j = \sum_{k=1}^{p} a_{kj}^2 = 1$$

The first component is calculated by choosing a_1 such as y_1 so that it has the greatest variance. The second component is calculated by choosing a_2 so that it is uncorrelated with y_1 and forth so that the variables obtained will have increasingly less variance. For the first component, for example, we obtain a_1 such as it maximizes the variance y_1 subject to the constraint $a'_1 a_1 = 1$. Knowing that $a'_1 x$ is a linear combination, then $Var(a'_1 x) = a'_1 \sum a_1$. Thus, the problem is to maximize this function subject to the restriction $a'_1 a_1 = 1$. It is noted that the unknown is precisely the vector a_1, the vector that will give the optimal linear combination.

Applying quadratic programming techniques finally leads to a diagonal matrix of eigenvalues, whose elements represent the obtained major variances that are associated with the corresponding eigenvectors, i.e. the principal components sought.

By applying this method to our matrix E, we identified the first principal components (which contain more variance), and the words of highest correlation with these components for later use them, that is to discriminate the important phrases of those unimportant.

2.5 Singular Value Decomposition (SVD) (Golub & Van Loan, 1996; Stewart, 1973)

This method is based on the Spectral Theorem to factor matrices. Thus, the singular values of a given matrix A are obtained from a diagonalization process of this matrix and its singular values are precisely the elements of this diagonal. These values are sorted from highest to lowest and are associated with corresponding singular vectors.

By specifying the above ideas, one has:

Let A be an $m \times n$ matrix, then there are two orthogonal matrices U and V such that $D = U^T AV$, where D is a diagonal matrix whose entries d_i are called singular values of matrix A and are arranged in decreasing order. U is an $m \times m$ square matrix whose columns are the left singular vectors and V is also a square matrix of order $n \times n$ whose columns contain the right singular vectors. D is a diagonal matrix of order $m \times n$.

One could say that the singular values of matrix A are the lengths of the semi-axes of the hyperellipse that maps the unit sphere by the matrix A. They are therefore non-negative real numbers. To better understand this statement, consider a pair of vectors x and y in Euclidean space $\mathbf{R}^2$ which are orthogonal. Imagine further that these vectors have the images Ax and Ay by the matrix A. If one rotates these vectors, they will describe the unit circle and their images Ax and Ay will describe an ellipse centered at the origin of coordinates (See Figure 3).

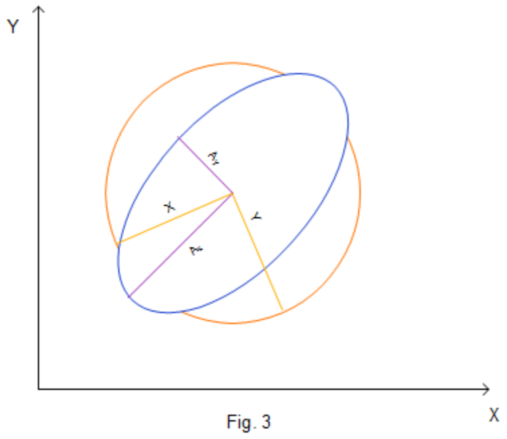

Figure 3. Sphere transformation for matrix A

In fact, there is a position of the vectors x and y such as their images are the semi-axes of the ellipse, i.e., they are perpendicular too. This fact can be exploited to the deductions that follow:

Set s_1 and s_2 are the singular values of A (the lengths of the semi-axes), then there are unit vectors u_1 and u_2 such as s_1u_1 and s_2u_2 are the vectors that represent the semi-axes of the ellipse. Therefore, u_1 and u_2 are orthogonal.

Let $x = v_1$ and $y = v_2$, then Av_1 and Av_2 are the semi-axes of the ellipse. Then $Av_1 = s_1u_1$ and $Av_2 = s_2u_2$. By expressing these equations in matrix form, one has:

$$A[v_1v_2] = [u_1u_2]\begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} = AV = US$$

Knowing that U and V are orthogonal and the diagonal matrix $S = \sum$ and by doing $U^T = \begin{bmatrix} u_1^T \\ u_2^T \end{bmatrix}$ and $V = [v_1v_2]$, then $U^TAV = diag(\sigma_1, \sigma_2) = \Sigma$ and $A = U\Sigma V^T$.

Obviously the above result can be generalized to the space $\mathbf{R}^n$.

To apply this method in our research, we also used the matrix E. We factored this matrix, we next founded the largest singular values associated with the right singular vectors V and finally we identified the components of these vectors containing the highest correlation. In other words, we started to identify the elements of highest variation to determine the information-bearing phrases.

3. Experiment

Our experiment was performed on a corpus of documents from DUC 2006 (NIST, 2006). This collection consists of 1250 documents that are grouped into 50 topics. We can resume the experiment by the following steps. We built the matrix $tf * idf$ and applied each of the chosen methods on this matrix. The training set was then created by selecting the sentences that contain more information to label them as important sentences and sentences less carriers of information as non-important. From this training set, we used machine learning techniques to abduce classification functions. We selected six machine learning techniques based on different approaches in order to obtain representative results. These six techniques are efficient and then largely used in the data mining field. For each chosen method, six classification functions were abduced and evaluated. We compared the results between them.

Following, we present briefly the abduction functions retained. We described next the evaluation criteria used to calculate the performance of the classification obtained. The results are described and discussed in Section 4.

3.1 Abduction Functions

We wanted to try techniques from several approaches: probabilistic, linear regression, trees and neural networks. Our idea is to determine not only the most appropriate technique but also its relationship with the most appropriate approach. With the application of these machine learning techniques, we abducted a function that enables to measure the discriminatory power according to the space reduction method used, as well as establish the effectiveness of the reduction-optimization methods chosen. For the application of each classification technique, we started from a set of instances which will carry information labeled each as important class (the subspace selected by applying the reduction technique) and instances of less information carriers classed as are not important (the subspace discarded by applying the reduction technique).

The first technique used is Support Vector Machine (SVM), which may be linear or nonlinear, using polynomial functions for example (Cortes & Vapnik, 1995; Vapnik, 1995). In our case, we used it as a linear technique. The second technique, Naïve Bayes (NB), based on the Bayes' theorem provides a way of calculating the probability of a hypothesis (a posteriori) based on their previous or a priori probability (Hastie et al., 2009). The Logistic Regression (LR), the third technique, is a probabilistic regression model whose dependent variable Y can only take two values that are explained by a set of predictors $x_1, x_2, \dots, x_k$ (Agresti, 2007). The fourth technique, Random Forest (RF), combines a selection method called bagging with a tree induction technique (Breiman, 2001). The bagging produces replicas of the training set sampling with replacement of training instances. The classifier Multilayer Perceptron (MLP), the fifth technique, is a neural network containing hidden layers that interact with the input layer and output layer of the network, allowing classifying elements of a state space that are not linearly separable (Rosenblatt, 1957; Rumelhart, Hinton, & Williams, 1986). The sixth and last technique is the Radial Basis Function Neural Networks (RBFNN). This network is a universal classifier composed of three layers: input, hidden and output (Buhmann, 2003). Some processing is not performed in the input layer. The hidden layer performs a nonlinear and local transformation on local data or input signals. The output layer allows a linear combination of activations of the hidden layer, that is gives the output of the network.

3.2 Evaluation Criteria

We used different metrics to evaluate the performance and the quality of classification for different machine learning techniques, which are implemented on the training sets obtained from spaces subject to the attribute optimization-reduction process. It is very important to note that a good performance of a machine learning technique is based on a good choice of the training set. Given our classifiers are binary, we use a contingency table where the entries are the two classes obtained by classification functions: important sentence and non-important sentence. This type of table, called confusion matrix (Table 1), displays four sets of sentences:

- True Positive (TP): if the function has correctly predicted the phrase labeled as important;
- True Negative (TN): if the function has predicted a phrase labeled as non-important when it is not important;
- False Positive (FP): if the function has predicted a phrase not important as being important; and,
- False Negative (FN), if the function has predicted that a sentence is important when it is not important.

The values obtained by grouping the sentences in this way will be used to obtain three metrics for performance evaluation of classifiers: precision, recall and F$_{\text{mesure}}$. Also from these data, we calculated the additional metrics sensitivity and specificity to construct ROC curves (Receiver Operation Characteristic) to try to visualize the quality of classifiers (Hastie et al., 2009; Lasko et al., 2005).

Table 1. Confusion matrix

Sentence Class	Predicted Class	
	Important	***Not important***
Important	True Positive Cases	False Positive Cases
Not important	False Negative Cases	True Negative Cases

Recall (R) is the proportion of positive cases in the total of cases classified as positive (True Positives + False Negatives). It informs about the ability of the classification function to correctly classify the important phrases. We calculate the Recall as:

$$R = \frac{TP}{TP + FN}$$

Precision (P) informs us of the ability of the function to classify the important phrases when they are truly, i.e. the proportion of correctly classified phrases in relation to the total important phrases (True Positives + False Positives). Precision is calculated as:

$$P = \frac{TP}{TP + FP}$$

The average of the two previous metrics calculated with its harmonic mean that is called F_{mesure} (also known as F_{score} or F1 score) and it is obtained as:

$$F_{mesure} = 2\frac{R \times P}{R + P}$$

In addition to the above metrics, we thought it is necessary to use another tool to analyze the quality of the classification functions. This tool has its origin in the analysis of signals from radar to distinguish true signals from which are not (noise). It is called curve analysis of Receiver Operation Characteristic (ROC). Its implementation in our research enables us to assess the accuracy of the classification models obtained, and to obtain a unified model of the evaluation process.

The curves are produced by calculating the sensitivity (true positive rate) versus specificity (1-false positive rate) and show a continuous variation of the observation points obtained (Lasko et al., 2005). From the observation of several curves, we obtain a qualitative comparison knowing that a curve on the top and to the left has the greatest accuracy. Additionally, obtaining the area under the curve (AUC) indicates the probability of success of the function to identify a sentence that is important. These metrics are obtained as follows (Spackman, 1989) :

$$sensitivity = TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

$$specificity = TNR = \frac{TN}{N} = \frac{TN}{TN + FP} = 1 - FPR$$

$$FPR = 1 - specificity$$

$$FPR = \frac{FP}{FP + TN}$$

Thus, Recall, Precision, F_{mesure}, sensitivity, specificity and ROC curves, provide us the values necessary for assessing the quality of the selected training set, by analyzing the performance of classifiers applied on them.

4. Results and Discussion

This section aims to show the results obtained from our experiment. As first results, we present the confusion matrix for each method chosen to optimize the attribute space. We also give the values computed for Recall, Precision and F_{mesure}. Next, all the ROC curves are given and discussed. We then conclude on the relevance of using the five chosen methods (KM, K-NN, FA, PCA, and SVD) to optimize the attribute space in extractive summarization.

4.1 Predictions and Confusion Matrices for the Different Method Analyzed

For the selection of instances to be classified, we used sub-sampling technique that selects the 2/3 of the training set for inducing the classification function. The remaining 1/3 was used as a sub-sample for the corresponding test. To obtain the evaluation metrics, we relied on three different software that yielded similar results: Tanagra (Rakotomalala, 2005), Weka (Holmes et al., 1994), and Orange (Demzar & Zupan, 2010).

The five following tables (Tables 2 to 6) show the values of the performance of classifiers applied to the spaces which have been optimized through the use of the five chosen methods. Each table presents the corresponding confusion matrix that has two values as input, class important and non-important class, and the values of the predictions: Recall, Precision and F_{mesure}. As we already mentioned, we wanted to examine the predictive power of each function associated with each classifier with respect to the attribute space that acts on each function.

More precisely, Table 2 presents the confusion matrix and the predicted values for the algorithms applied to the matrix $tf * idf$ of attributes optimized by applying the KM method. By noting the values of the table, we realize that the values of Precision, Recall and F_{mesure} for each algorithm are very high, with values of 1 or approaching this maximum. This would indicate that the KM method is quite efficient for the selection of spaces that can be used by all these algorithms in the way of classifying the sentences of a set of documents as important and not important. It is important to note here that the KM method can be used as a classification technique in an unsupervised machine learning approach.

Table 2. K-means method (predictions and confusion matrices)

Algorithm	*Class*	*Confusion Matrix*		*Predictions*		
		Important	*Not Important*	*Recall*	*Precision*	F_{mesure}
SVM	Important	91	1	0.9891	1	0.9945
	Not important	0	94	1	0.9895	0.9949
Naive Bayes	Important	91	1	0.9891	1	0.9945
	Not important	0	94	1	0.9895	0.9947
Logistic Regression	Important	92	0	1	1	1
	Not important	0	94	1	1	1
Random Forest	Important	92	0	1	1	1
	Not important	0	94	1	1	1
Multilayer Perceptron	Important	91	1	0.9891	1	0.9945
	Not important	0	94	1	0.9895	0.9947
RBF Neural Networks	Important	92	0	1	1	1
	Not important	0	94	1	1	1

From observation of Table 3, we conclude that the method KNN applied to spaces of attributes may produce smaller spaces with high information content. The discriminatory task of classification algorithms is made easier as demonstrated with the F_{mesure} calculated values for each of them: all values are above 93%, with values 1 or close to it.

Table 3. Kohonen Neural Networks method (predictions and confusion matrices)

Algorithm	*Class*	*Confusion Matrix*		*Predictions*		
		Important	*Not Important*	*Recall*	*Precision*	F_{mesure}
SVM	Important	42	0	1	1	1
	Not important	0	42	1	1	1
Naive Bayes	Important	42	0	1	0.9333	0.9655
	Not important	3	39	0.9286	1	0.9630
Logistic Regression	Important	42	0	1	1	1
	Not important	0	42	1	1	1
Random Forest	Important	42	0	1	0.9767	0.9882
	Not important	1	41	0.9762	1	0.9879
Multilayer Perceptron	Important	37	5	0.8810	0.9737	0.9250
	Not important	1	41	0.9762	0.8913	0.9318
RBF Neural Networks	Important	42	0	1	1	1
	Not important	0	42	1	1	1

Table 4 shows the performance of classifiers applied to an attribute space that is optimized with the FA method. The observed values of F_{mesure} are 1 or close to this value.

Table 4. Factor analysis method (predictions and confusion matrices)

Algorithm	*Class*	*Confusion Matrix*		*Predictions*		
		Important	*Not Important*	*Recall*	*Precision*	*F_{mesure}*
SVM	Important	56	0	1	1	1
	Not important	0	44	1	1	1
Naive Bayes	Important	55	1	0.9821	1	0.9909
	Not important	0	44	1	0.9777	0.9887
Logistic Regression	Important	56	0	1	1	1
	Not important	0	44	1	1	1
Random Forest	Important	56	0	1	0.9180	0.9572
	Not important	5	39	0.8864	1	0.9397
Multilayer Perceptron	Important	56	0	1	0.875	0.9333
	Not important	8	36	0.8182	1	0.9001
RBF Neural Networks	Important	56	0	1	1	1
	Not important	0	44	1	1	1

Table 5 illustrates the values of performance obtained by applying classification algorithms to attribute space optimized by the PCA method. The values obtained for F_{mesure}, except for the values obtained for MLP (92.03% and 92.55%), are all equal to 1 or close to this maximum value.

Table 5. Principal components analysis method (predictions and confusion matrices)

Algorithm	*Class*	*Confusion Matrix*		*Predictions*		
		Important	*Not Important*	*Recall*	*Precision*	*F_{mesure}*
SVM	Important	61	0	1	1	1
	Not important	0	56	1	1	1
Naive Bayes	Important	61	0	1	0.9531	0.9759
	Not important	3	53	0.9464	1	0.9724
Logistic Regression	Important	61	0	1	1	1
	Not important	0	56	1	1	1
Random Forest	Important	60	1	0.9836	1	0.9917
	Not important	0	56	1	0.9825	0.9911
Multilayer Perceptron	Important	52	9	0.8525	1	0.9203
	Not important	0	56	1	0.8615	0.9255
RBF Neural Networks	Important	61	0	1	1	1
	Not important	0	56	1	1	1

The performance values obtained based on F_{mesure} applying the SVD method, presented in Table 5, show that all are close to 1, except for the values obtained for non-important class of the classifiers RT (90.91%) and MLP (89.76%).

Table 6. Singular value decomposition (predictions and confusion matrices)

Algorithm	*Class*	*ConfusionMatrix*		*Predictions*		
		Important	*Not Important*	*Recall*	*Precision*	*F_mesure*
SVM	Important	196	0	1	0.9949	0.9974
	Notimportant	1	67	0.9853	1	0.9925
NaiveBayes	Important	196	0	1	0.9849	0.9923
	Notimportant	3	65	0.9559	1	0.9774
LogisticRegression	Important	196	0	1	0.9949	0.9974
	Notimportant	1	67	0.9853	1	0.9925
RandomForest	Important	192	4	0.9796	0.9600	0.9697
	Notimportant	8	60	0.8824	0.9375	0.9091
MultilayerPerceptron	Important	194	2	0.9898	0.9463	0.9675
	Notimportant	11	57	0.8382	0.9661	0.8976
RBF Neural Networks	Important	194	2	0.9898	1	0.9949
	Notimportant	0	68	1	0.9714	0.9854

4.2 ROC Curves

Figures 4 to 8 show the ROC curves obtained for each classifier on different spaces of attributes. AUC values (Area Under Curve) are shown in the box located at the bottom right. We thought it is important to present these graphs, in order to analyze the results in a different viewpoint. If we fix our gaze for example the important class, these graphs would show the probability that the function classifies an important phrase that has been labeled as such. The optimal point of each classifier can be found by identifying the highest and farthest to the left. Thus, this measure can distinguish performance between algorithms.

If we look at the ROC curves in Figure 4, we find that all AUC registered values above 75%. The maximum values were obtained with the algorithms SVM (99.88%), MLP (98.41%) and the LR (97.44%), which means that these last three classifiers should be used in conjunction with the KM method. In Figure 5, we verify that all AUC values of the graphics are above 75%. The maximum values are for the algorithms NB (99.85%), RT (99.85%) and SVM (94.69%), indicating that the latter three algorithms should be used with the KNN method.

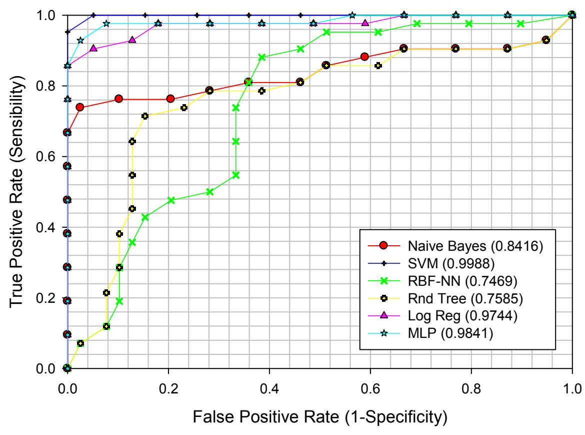

Figure 4. K-means (ROC Curves)

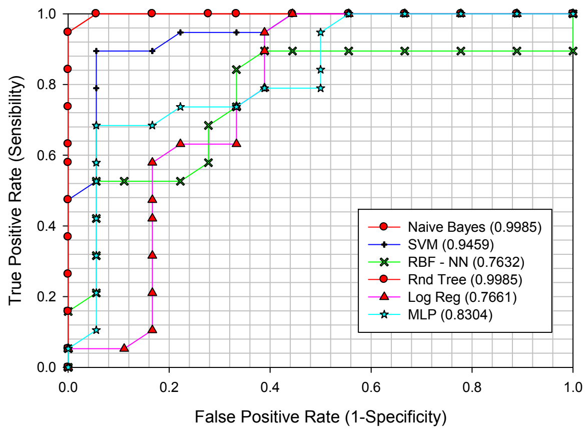

Figure 5. Kohonen-Neural Networks (ROC Curves)

Unlike the previous methods, the AUC values observed in Figure 6 show different results. However, the values of the NB, RBF and RT algorithms are really acceptable: 90.8%, 72.5% and 72.2% respectively. Therefore, the FA method is recommended to be applied only with these two classifiers. By analyzing the graphs of Figure 7, we note that AUC values ranged from 72.1% (SVM) and 81% (RT). We conclude the PCA method could be well applied in conjunction with all the algorithms analyzed with similar results.

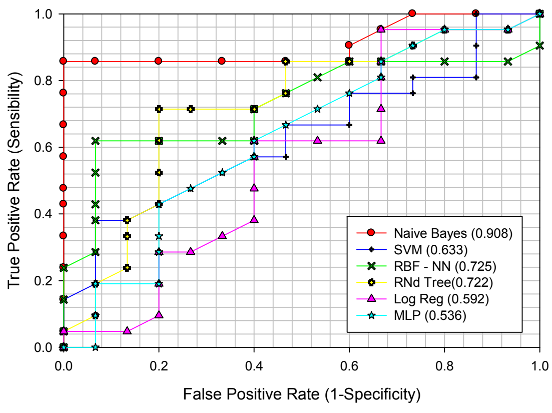

Figure 6. Factor Analysis ROC Curves

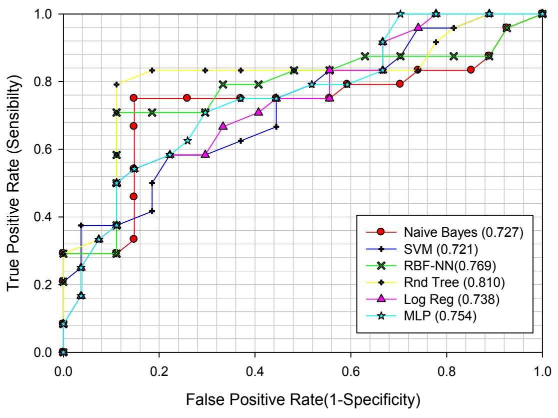

Figure 7. Principal Components Analysis ROC Curves

If we observe AUC values for the different algorithms in Figure 8, we note that their values are between 72.1% for RBFNN to 84.8% for MLP and LR. Thus the SVD method may be well applied with all algorithms studied.

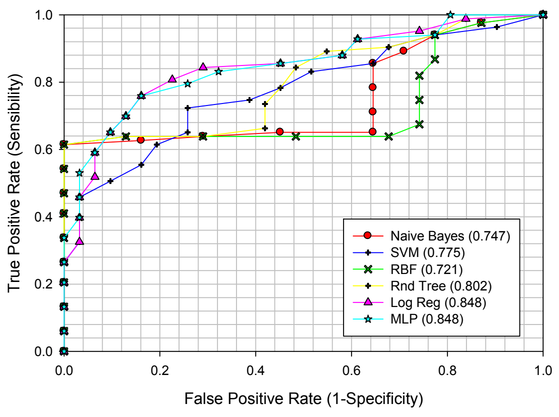

Figure 8. Singular Value Decomposition (ROC Curves)

Table 7 shows the AUC values for each classification algorithm based on the reduction method applied. In the rows, one can observe the AUC values obtained for each machine learning algorithm. The columns give the AUC values of each algorithm for one reduction method given.

Table 7. AUC values by reduction/optimization method for the classification algorithms

Algorithm	*Method*				
	K-means	*Kohonen-NN*	*FA*	*PCA*	*SVD*
SVM	0.999	0.946	0.967	0.721	0.775
Naive Bayes	0.842	0.999	0.560	0.727	0.747
Logistic Regression	0.974	0.766	0.510	0.738	0.848
Random Forest	0.759	0.999	0.814	0.810	0.802
Multilayer erceptron	0.984	0.830	0.536	0.754	0.848
RBF-Neural Networks	0.747	0.763	0.455	0.769	0.721

Generally the average value of the exactness and completeness is represented by the metric F_β (F_1 in our case). We calculated this metric for each classification algorithm applied on each method of reduction and the values obtained are quite high for all. We also use the ROC values of Table 7 to conclude about the best methods to use.

By observing this table, we could say that the reduction methods, which give greater performance to the classification algorithms, are the KNN and KM methods. The values obtained are above 75%. We could even select one of two methods depending on the algorithm used. For instance, by considering that NB presents a performance of 84.2% with the KM method, then NB may be used with the KNN method with a performance of 99.9%. It is important to note also that the use of the other methods give acceptable results with some algorithms. For instance, SVM with the FA method presents a performance of 96.7% and LR and MLP a performance of 84.8% with the SVD method.

5. Conclusion

In order to obtain smaller spaces or attribute subsets with an important information content and abduce classification functions, we used five different methods (KM, KNN, FA, PCA and SVD) that fall within the attribute selection and extraction approaches. As mentioned at the beginning of this article, the classification functions obtained will be used to produce automatically summaries in an unsupervised approach. The methods used to optimize the attribute space are based on low-dimensional projections, transformations and partitioning of this space for the selection of a representative subset. We call this process reduction/optimization in the sense that, to obtain the corresponding subsets, we use known techniques of mathematical optimization and local optimization. It is important to note that the chosen methods are not commonly used for the reduction of attribute spaces. We then created algorithms based on these methods for this purpose.

By considering that a good subset of attributes should be highly correlated with the classification process (Hall, 1999; Ikonomakis et al., 2005), we relied on the values of performance of classification functions obtained from these subsets to estimate the quality of the reduction method used. We studied the application of six current machine learning techniques, which are SVM, NB, LR, RF, MLP and RBFNN, in order to obtain significant results.

From the analysis of the experimental results, we conclude that the two best methods for reducing the space of attributes, between the methods we have studied, are KM and KNN, which produced excellent results for the six machine learning techniques applied. In addition, the better results were obtained with SVM, MLP and LR for KM, and NB, RT and SVM for KNN.

References

Agresti, A. (2007). *Building and applying logistic regression models. An Introduction to Categorical Data Analysis*. Hoboken, New Jersey: Wiley.

Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5-32. http://dx.doi.org/10.1023/A:1010933404324

Buhmann, M. D. (2003). *Radial Basis Functions: Theory and Implementations*. Cambridge University Press. http://dx.doi.org/10.1017/CBO9780511543241

Lin, C. Y. (2004). ROUGE: a Package for Automatic Evaluation of Summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, Barcelona, Spain, July 25-26, 2004.

Cortes, C., & Vapnik, V. (1995). *Support-Vector Networks*. Holmdel, NJ 07733, USA: AT & T Bell Labs.

Demzar, J., & Zupan, B. (2010). *ORANGE*: A Software Developed at Laboratory of Artificial Intelligence. Retrieved from http://orange.biolab.si/

Golub, G. H., & Van Loan, C. F. (1996). *Matrix Computations*. *Johns Hopkins Studies in Mathematical Sciences* (3rd ed.). The Johns Hopkins University Press.

Hall, M. A. (1999). *Correlation-based Feature Subset Selection for Machine Learning*. PhD dissertation, Department of Computer Science, University of Waikato.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning* (2nd ed.). New York: Springer. http://dx.doi.org/10.1007/978-0-387-84858-7

Holmes, G., Donkin, A., & Witten, I. H. (1994). *WEKA*: A Software Developed by Machine Learning Group). Retrieved from www.cs.waikato.ac.nz/ml/weka

Ikonomakis, M., Kotsiantis, S., & Tampakas, V. (2005). Text Classification Using Machine Learning Techniques. Paper presented at the WSEAS Transations on Computers.

Ikonomakis, M., Kotsiantis, S., & Tampakas, V. (2005). Text Classification Using Machine Learning Techniques. *WSEAS Transactions on Computers, 8*(4), 966-974.

Jolliffe, I. T. (2002). *Principal Component Analysis*. New York: Springer.

Joreskog, K. G., & Van Thillo, M. (2010). *LISREL: A General Computer Program for Estimating a Linear Structural Equation System Involving Multiple Indicators of Unmeasured Variables*. Princeton, NJ: Educational Testing Service.

Kim, J., & Mueller, C. (1978). *Factor analysis: statistical methods and practical issues*. Newbury Park, California: Sage Publications Inc.

Kohonen, T. (1990). The Self-Organizing Map. *Proceedings of the IEEE, 78*(9), 1464-1480. http://dx.doi.org/10.1109/5.58325

Langley, P. (1994). Selection of Relevant Features in Machine Learning. *Proceedings of the AAAI Fall Symposium on Relevance*. New Orleans, LA: AAAI Press

Lasko, T. A., Bhagwat, J. G., Zou, K. H., & Ohno-Machado, L. (2005). The use of receiver operating characteristic curves in biomedical informatics. *Journal of Biomedical Informatics, 38*(5), 404-415. http://dx.doi.org/10.1016/j.jbi.2005.02.008

Motta, J. A., Capus, L., & Tourigny, N. (2011). Insertion of Ontological Knowledge to Improve Automatic Summarization Extraction Methods. *Journal of Intelligence Learning Systems and Applications, 3*, 131-138. http://dx.doi.org/10.4236/jilsa.2011.33015

NIST. (2006). *Document Understanding Conferences – DUC2006*. Retrieved from http://www-nlpir.nist.gov/projects/duc

Rakotomalala, R. (2005). Tanagra: un logiciel gratuit pour l’enseignement et la recherche. *Proceedings of the EGC’2005 Conference*, Amsterdam, RNTI-E-3, vol. 2, 697-702.

Rosenblatt, F. (1957). *The Perceptron-a perceiving and recognizing automaton*. New York: Cornell Aeronautical Laboratory.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics, 20*(1), 53-65. http://dx.doi.org/10.1016/0377-0427(87)90125-7

Rumelhart, D., Hinton, G., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*, 533-536. http://dx.doi.org/10.1038/323533a0

Smale, S. (1997). Complexity Theory and Numerical Analysis. *Acta Numerica, 6*, 523-551. http://dx.doi.org/10.1017/S0962492900002774