UNIVERSIDAD CARLOS III DE MADRID working papers

Working Paper: 11-17
Statistics and Econometrics Series 012
June 2011

Departamento de Estadística
Universidad Carlos III de Madrid
Calle Madrid, 126
28903 Getafe (Spain)
Fax (34) 91 624-98-49

HANDWRITTEN DIGIT CLASSIFICATION

Andrea Giuliodori, Rosa Lillo and Daniel Peña

Abstract

Pattern recognition is one of the major challenges in statistics framework. Its goal is the feature extraction to classify the patterns into categories. A well-known example in this field is the handwritten digit recognition where digits have to be assigned into one of the 10 classes using some classification method. Our purpose is to present alternative classification methods based on statistical techniques. We show a comparison between a multivariate and a probabilistic approach, concluding that both methods provide similar results in terms of test-error rate. Experiments are performed on the known MNIST and USPS databases in binary-level image. Then, as an additional contribution we introduce a novel method to binarize images, based on statistical concepts associated to the written trace of the digit.

Keywords: digit, classification, images

Departamento de Estadística, Universidad Carlos III de Madrid, C/ Madrid 126, 28903 Getafe (Madrid), e-mail addresses: mariaandrea.giuliodori@uc3m.es (Andrea Giuliodori), rosaelvira.lillo@uc3m.es (Rosa Lillo) and daniel.pena@uc3m.es (Daniel Peña). This research acknowledges the support of “Comunidad de Madrid" grant CCG07-UC3M/HUM-3260.

HANDWRITTEN DIGIT CLASSIFICATION

ANDREA GIULIODORI, ROSA LILLO AND DANIEL PEÑA

Abstract. Pattern recognition is one of the major challenges in statistics framework. Its goal is the feature extraction to classify the patterns into categories. A well-known example in this field is the handwritten digit recognition where digits have to be assigned into one of the 10 classes using some classification method. Our purpose is to present alternative classification methods based on statistical techniques. We show a comparison between a multivariate and a probabilistic approach, concluding that both methods provide similar results in terms of test-error rate. Experiments are performed on the known MNIST and USPS databases in binary-level image. Then, as an additional contribution we introduce a novel method to binarize images, based on statistical concepts associated to the written trace of the digit.

In the field of digital image processing, pattern recognition is the research area that studies the operation and design of systems that recognize patterns in data. Its primary goal is the classification of objects into a number of categories or classes. Depending on the use, these objects can be images, signal waveforms or any other type of measurements that need to be classified. Common applications of pattern recognition are automatic speech recognition, classification of text into several categories (e.g. spam/non-spam email messages) and the automatic recognition of handwritten postal codes on postal envelopes. Statistical approaches are one of the most widely studied and used to perform pattern classification.

In statistical pattern recognition, an image is represented by a set of f features which constitute a f-dimensional feature vector. A decision process (statistical classifier) based on this vector is used to establish boundaries between image classes and then perform the classification. Thus, the classification success depends entirely on the set of selected features and the classification scheme.

In this paper we tackle the handwritten digit recognition problem. Our purpose is to present alternative classification methods based on statistical techniques and with good performance for classifying handwritten digit images. We use two different statistical approaches. Firstly, we use a probabilistic approach assuming that the features of images in the training set have a probability distribution in order to use the Bayes's decision rule to classify images in the test set. Secondly, we conduct a supervised classification approach (where classes of sets are known), using the K-nearest neighbors rule to classify images in the test set. The classification scheme is based on a set of variables (feature vector) obtained by applying structural measures to detect the shape and geometry of the numbers. Our methodology has the advantage that is more intuitive and generalizable over other methods that require the use of scanned digit with the same size [see Lauer et al. (2007)]. Besides, our methodology do not need any pre-processing (as deskew, noise removal or shift the

Key words and phrases. **digit, classification, image**.

edges) of images from databases [see Decoste and Scholkopf (2002) and Keysers et al. (2007)].

Experiments are performed on two databases described in Section 1. The binarization of the images is required to calculate the features used in this paper. Thus, in Section 2 we propose a new binarization method to find an optimum threshold parameter for each image, which is based on the written trace of digit. Section 3 describes the construction of the variables proposed for the classification. Initially, the feature vector is composed with the variables calculated in Section 3. However, the final feature vector used to classify is selected by the application of the sequential forward selection technique. In Section 4 we present the probabilistic approach which is performed by the application of the Bayes's rule, disjointing the variables into categorical and quantitative. Section 5 is devoted to describe the application of the K-nearest method algorithm to classify the digit images. Finally, the last Section 6 provides the conclusion of the two classification methods. Both techniques provide similar results to handwritten digit classification, despite the fact that in the probabilistic approach we consider the distribution of the variables, whereas in the KNN algorithm only the distances are used in the classification.

1. Databases

The classification in this chapter is performed using two different databases consisting of scanned digits, from 0 to 9. Datasets are partitioned in training and testing sets in order to validate the technique. In general, as explained in Chapter 2, training set is used to classify the testing set, being both independent.

MNIST database. The *MNIST* database was constructed from **N**ational **I**nstitute of **S**tandards and **T**echnology (*NIST*) database of scanned handwritten digit.

NIST originally had the training set composed by a collection of digits written by paid US census workers, while the testing set was collected from digits written by uncooperative high-school students. This difference of origin of the data explains why the classification errors obtained in each group were completely different with worse performance on the test data. Therefore, the NIST database required a reorganization in order to combine adequately training and test sets, forming the *MNIST* (Modified NIST) set [LeCun et al. (1998)], which is the database that we use in this work.

The *MNIST* database is composed by 60.000 handwritten digits in the training set and 10.000 in the test set. Digits were size-normalized and centered in an image of size 28×28 by computing the center of mass of pixels, and translating the image to locate this center point at the center of the 28×28 field [LeCun et al. (1998)]. The *MNIST* set contains gray level images as a result of the anti-aliasing technique used by the normalization algorithm conducted by the authors (http://yann.lecun.com/exdb/mnist/). The distribution of digits in the training and testing sets are in Table 1.

Table 1. Distributions of *MNIST* sets

Digit	0	1	2	3	4	5	6	7	8	9	Total
Train	5923	6742	5958	6131	5842	5421	5918	6265	5851	5949	60000
Test	980	1135	1032	1010	982	892	958	1028	974	1009	10000

Some examples of *MNIST* are in Figure 1. As a consequence of the origin of

(A) MNIST training (B) MNIST test

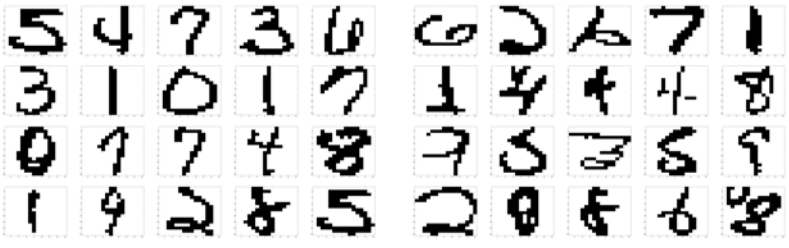
FIGURE 1. Typical Images from *MNIST* sets

the *MNIST* database, digits in the training and testing sets seem to have the same degree of difficulty to be recognized.

USPS database. The *USPS* database comes from a set of digits automatically scanned from envelopes by the **U**nited **S**tate **P**ostal **S**ervice. The original scanned digits were binary and with different sizes and orientations. The segmentation procedure performed by Postal Service caused that some digits were mis-segmented. Thus, the database was generated with extreme difficulty to be recognized and classify, with a human error-rate around the $2,5\%$ [Simard et al. (1993)]. Images used in this work were deslanted [1] and size-normalized by LeCun et al. (1990), resulting in 16 x 16 grayscale images. The training set is composed by 7291 images and the testing set contains 2007 images. The distributions of digits in the training and testing sets is showed in Table 2. Examples of this database are in Figure 2.

TABLE 2. Distributions of USPS sets

Digit	0	1	2	3	4	5	6	7	8	9	Total
Train	1194	1005	731	658	652	556	664	645	542	644	7291
Test	359	264	198	166	200	160	170	147	166	177	2007



(A) USPS training (B) USPS test

FIGURE 2. Typical Images from *USPS* sets

[1]Deslant is a term commonly used in handwritten recognition field to indicate the action of removing the slant of the text by some specific technique.

Although the completely set of numbers is not shown, it can be seen in the examples of *USPS* presented in Figure 2 some clear differences between test and training sets. In the testing set digits seem to be more unreadable than in the training set. Besides, the testing has some mis-segmented digits, such as the digit 8 shown in Figure 2b. This obvious discrepancy between sets represents a difficulty to classify both sets with the same performance in terms of classification, because the training, set used to classify images of the testing set, has less variability in the shape of digit.

2. Binarization

The variables proposed in this paper to make handwritten digit classification require images in binary level. The binarization process assumes that images contain two classes of pixel: the foreground (or white pixels, with maximum intensity, i.e., equal to 1) and the background (or black pixels with minimum intensity, i.e., equal to 0). The goal of the method is to classify all pixels with values above of the given threshold as white, and all other pixels as black. That is, given a threshold value t and an image X with pixels denoted as $x(i,j)$, the binarized image X_b with elements $x_b(i,j)$ is obtained as follows.

$$\begin{array}{ll} \text{If} \quad x(i,j) > t, & x_b(i,j) = 1 \text{ (object)} \\ \text{else} & x_b(i,j) = 0 \text{ (background)} \end{array}$$

However, this version of the algorithm assumes that we are interested in light objects on a dark background. Then, in order to obtain dark objects on a light background we would use,

$$\begin{array}{ll} \text{If} \quad x(i,j) < t, & x_b(i,j) = 1 \text{ (object)} \\ \text{else} & x_b(i,j) = 0 \text{ (background)} \end{array}$$

Then, the key problem in the binarization is how to select the correct threshold t for a given image. We observe that the shape of any object in the image is sensitive to variations in the threshold value, and even more sensitive in the case of handwritten digit. Therefore, we consider that a binary handwritten number is better recognized computationally if its trace is complete and continuous, this is the criterion that we use to the threshold, being its choice of crucial importance.

Although several methods exist for choosing a threshold, in an attempt to obtain a threshold more adequate for digit binarization, we propose a novel method to find an optimum threshold value. The procedure is based on statistical concepts that consider the handwriting trace of the digit, finding an optimum threshold value associated with each image. In the process we use the median absolute deviation[2] (mad) as a robust variability measure. The algorithm to find the optimum threshold consists of assigning an initial threshold value denoted as t and then binarize the image X, obtaining what we call the *local trace of the line* in a digit. That is, for each white pixel x_{ijt} (pixel binarized with value 1, located in row i, column j and with a given threshold t), we find the horizontal and vertical number of contiguous white pixels, denoted as h_{ijt} and v_{ijt} respectively. Then, we choose the minimum of both values, indicated as Y_{ijt}. That is, Y_{ijt} is the trace of the line at point (i,j) with threshold t. Thus, we define the median absolute deviation of the values Y_{ijt} as the *global variability of the trace of the line* , denoted as $gtr(X)_t$. The procedure

[2]In general, the mad of a variable X is defined as $mad(X) = median_i(|X_i - median_j(X_j)|)$, for $i = 1, \ldots, n$ and $j = 1, \ldots, n$.

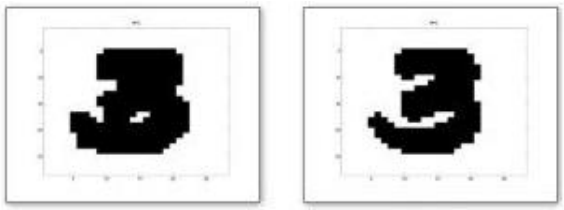
is repeated for different threshold values and the optimum threshold (t_{op}) for an image X is the value of t that has the minimum *global variability of the trace of the line*, that is, the value that makes the trace more homogeneous. Formally,

$$Y_{ijt} = \min(h_{ijt}, v_{ijt}) \tag{1}$$

$$gtr(X)_t = mad(Y_{ijt}) \tag{2}$$

$$t_{op} = \min_t(gtr(X)_t) \tag{3}$$

Figure 3 shows the same digit with two different threshold values. In both cases the digit is clear to be recognized as the number three by human eye. However, both images are not clear enough for a computer to be classified in the same class. The digit with a $t = 0.001$ is more difficult to recognize than the one that has $t_{op} = 0.30$.



(A) $t = 0.001$ (B) $t_{op} = 0.30$

FIGURE 3. The same digit with different threshold values

3. Features extraction

The group of features used in the classification scheme considers the shape and some structural characteristics of the digits. Some of them are calculated through the *Hough transform* and others are proposed, developed and programmed specifically for this work, using the software MATLAB version 7.9 [see Gonzalez et al. (2004)]. All the variables described in this section are obtained for every digit binarized with its corresponding optimum threshold value described in the previous section. Since some of the features are based on the Hough Transformation, we explain it in detail in the next subsection.

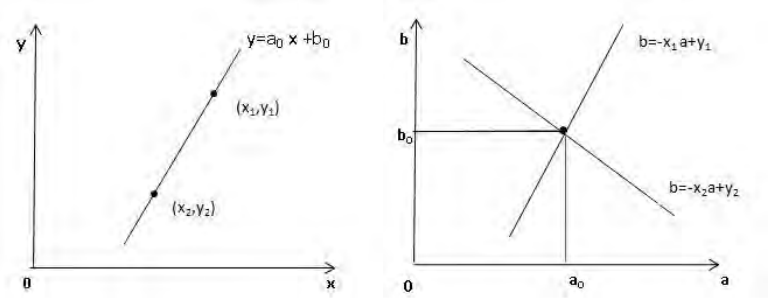
3.1. **Hough Transform.** The Hough Transformation is a technique initially implemented to the identification of lines in a binarized image. Later, it was extended to detect arbitrary shapes, generally circles or ellipses [Ballard (1981)]. The HT, as it is universally used today was developed by Duda and Hart (1972). However, the name comes from its inventor Hough (1962). In this work we use the HT to detect lines and circles in digit images.

3.1.1. *Straight lines detection.* The main tenet of the HT is to detect the occurrence of figure points (pixels for us) in an image, lying on a straight line. The equation for a straight line is represented in the Cartesian coordinates as

$$y_i = a\, x_i + b, \tag{4}$$

and is graphically plotted for a pair of points (x_1, y_1) and (x_2, y_2) as in Figure 4a. The HT aims to find points with coordinates (x, y) that satisfy the equation 4.

There exist infinite lines which pass through a particular point (x_i, y_i) in the Cartesian plane, but only one line satisfies the equation 4 for specific values of parameters a and b. Moreover, points lying on the same straight line in Cartesian plane can be represented in the space of parameters a and b as it is shown in Figure 4b. That means, two points lying on the same straight line with parameters a and



(A) Cartesian plane (B) Parameter space

FIGURE 4. Representation of a line

b in the Cartesian plane are represented in the parameter space as two lines with an interception point (a, b). Then, an arbitrary straight line can be represented by a single point in the parameter space.

A disadvantage of using the equation 4 is that the slope can be infinite if the straight line is vertical. This problem is solved by using the so-called normal representation of a straight line (also known as Hesse's normal).

The general form of the linear equation 4 is

$$Ax + By + C = 0, \tag{5}$$

and the normal representation is given by

$$x\ \cos\theta + y\ \sin\theta - \rho = 0. \tag{6}$$

As both form represent the same line, their respected coefficients must be proportional. Therefore,

$$\begin{aligned} \cos\theta &= k\ A \\ \sin\theta &= k\ B \\ -\rho &= k\ C, \end{aligned} \tag{7}$$

where k is a coefficient of proportionality. Squaring and summing both sides of the first and the second equation in 7, we obtain

$$\cos^2\theta + \sin^2\theta = k^2(A^2 + B^2). \tag{8}$$

Hence,

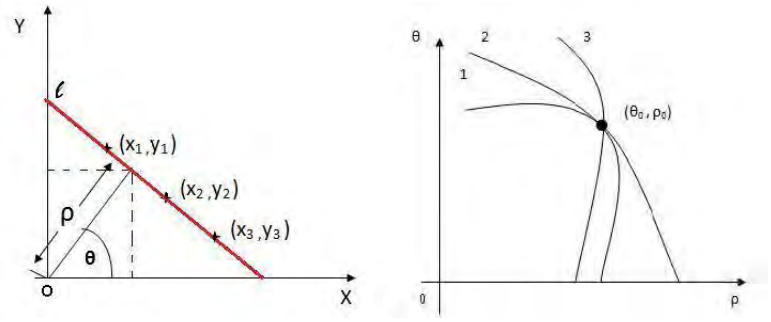
$$k = \frac{1}{\pm\sqrt{A^2 + B^2}}. \tag{9}$$

Substituting in equation 7,

$$\begin{aligned} \cos\theta &= \frac{A}{\pm\sqrt{A^2 + B^2}}, \\ \sin\theta &= \frac{B}{\pm\sqrt{A^2 + B^2}}, \\ -\rho &= \frac{-C}{\pm\sqrt{A^2 + B^2}}, \end{aligned} \tag{10}$$

which are the coefficients of equation 6. In our case, $A = a$, $B = -1$ and $C = b$.

An example of a line (called ℓ) obtained by the equation 4, with parameters a_0 and b_0 is represented in red color in Figure 5a. The parameter ρ is the vector that



(A) Cartesian plane (B) Parameter plane

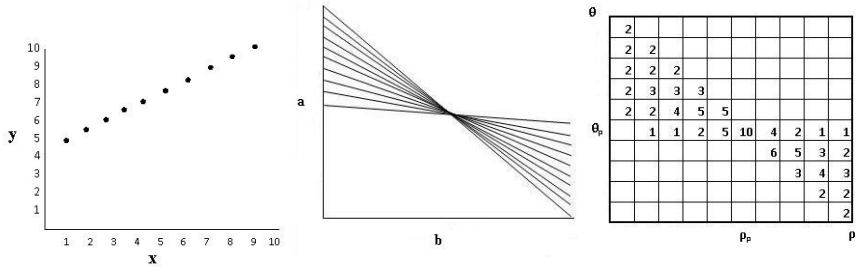
FIGURE 5. Normal representation of a line.

represents the distance between the line ℓ and the origin, while θ is the angle that forms the vector ρ with the X-axis.

As a consequence of this representation, each point (x,y) in the normal form (Figure 5a) is represented as a curve in the parameter plane (Figure 5b). If we add the restriction that θ belongs to the interval $[0, \pi]$, the normal parameters for a line are unique. With this restriction, every line in the cartesian plane corresponds to a unique point in the parameter plane. The collinear points located in a straight line in normal representation have a common point of intersection in the parameter plane. The point (θ_0, ρ_0) in this plane defines the line passing through the collinear points in the Cartesian plane. Thus, the problem of detecting collinear points can be converted into a problem of finding a common point of intersection.

The Hough transform algorithm uses an array (or matrix) called accumulator, to detect the existence of straight lines in images. The dimension of the accumulator is given by the number of unknown parameters in the equation 6, i.e. two. The parameters are quantized to be represented in a two-dimensional array with size $d_1 \times d_2$, where d_1 is the number of values of θ uniformly spaced in the interval$(0, \pi)$, and d_2 is the number of values in the ρ axis also uniformly spaced in an interval specified as $(-R, R)$. Then, for each pixel of the image, the accumulator eventually

records the total number of lines (with restricted parameters), passing through the pixel. In Figure 6a is shown an example of figure points lying on the same line in the Cartesian plane, while in **??** is the representation in parametric plane. Besides, the representation of the accumulator array is shown in Figure **??**. After all pixels



(A) (B) (C)

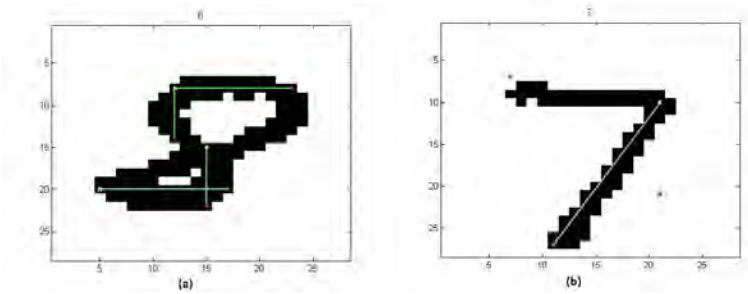
FIGURE 6. Hough Transform

are treated, the array is inspected to find cells with high counts, called peaks (p). If the counts of a given cell is 10, then, there are precisely 10 points lying along the line whose normal parameters are (θ_p, ρ_p). In Figure 6c there is an example of the accumulator array (or Hough matrix) where there is a peak $p = 10$. Thus, there is a line in the image composed by 10 points (or pixels) represented as in Figure 6a. In Figure 6b, the 10 lines are represented in the space of parameter a and b. Finally, in Figure 6c is shown the resulting accumulator array of the Hough transform. The matrix shows the lines that can be formed in a picture, where each cell represents the number of pixels composing them. The longest line is the one which has 10 points (pixels). However, there is one line with 6 points and four other lines composed by 5 points. The lines to be selected depend on the goal of the study.

In the HT calculation, the parameters ρ and θ can be restricted to find lines with a particular slope or position in an image. Also the minimum number of pixels required to conform a line (the minimum value of a peak p) can be determined. After making some analysis in our work, we consider interesting to find vertical (90^o), horizontal (0^o) and diagonal (45^o) lines to differentiate digits. Since images have small size we select lines with at least two pixels. Therefore, the Hough transform detected all possible lines with those characteristics in four stages.

(1) In the first stage the binary image is split horizontally into two rectangular equal parts and the largest horizontal line is registered from each part of the image.
(2) In the second stage, the binary image is divided vertically into two rectangular equal parts and the largest vertical line is detected from each part of the image.
(3) In the third stage, the image is divided by its principal diagonal and the largest upper and lower parallel to this diagonal are found.
(4) In the four stage, the image is divided by its secondary diagonal and the largest upper and lower parallel to this diagonal are found.

The features considered in the classification were obtained from the information of the selected lines. Every line has two points, the start and the end-point. The coordinates (x_i, y_i) which specify the start-point and end-point of a straight line are what we call *straight* (S) to specify the coordinate of lines of 0° and 90°. Besides, we called *diagonal* (D) to refer the coordinates of a diagonal lines (with 45°). We also include the *length* (Le) of each line as an additional variable. Therefore, we have a total of 5 variables for every line (4 corresponding to the coordinates and 1 corresponding to the length). Since we use four straight lines and four diagonals to defined the shape of an image, we record 40 resulting values per image. Graphically, examples of straight and diagonal lines are depicted in Figure 7.



(A) Straight lines (B) Diagonal line

FIGURE 7. Examples of lines

During the calculation of the Hough transform, we observe that some digits do not have all the lines we want to extract, that is 2 horizontal, 2 vertical, and 4 diagonal lines to define the shape of the digit. In these cases, we replace the missing information by assigning the coordinates and length of a single point. The coordinates of those points are determined by the location of them in the image. Each replacement point is located in accordance with the line that is missed.

3.1.2. *Circles detection.* We also used the HT to detect circles in images. In an xy Cartesian plane (see Figure 8), the circle with center coordinates (a, b) and radius r is the set of all points (x, y) such that

$$(x-a)^2 + (y-b)^2 = r^2 \tag{11}$$

For a circle with radius r and center in the origin $(0, 0)$, equation 11 is rewritten as,

$$x^2 + y^2 = r^2.$$

Then, the coordinates (x, y) in the Cartesian plane center in this origin are equal to

$$x = r\ \cos\phi$$
$$y = r\ \sin\phi,$$

where ϕ is the angle that the radius forms with the x-axis, defined in a range $(0, 2\pi)$. And, by axis translation, those coordinates center in (a, b) are equal to

$$x = r\cos\phi - a$$
$$y = r\sin\phi - b.$$

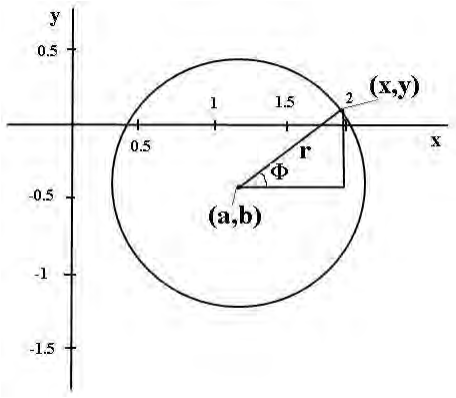


FIGURE 8. Cartesian plane

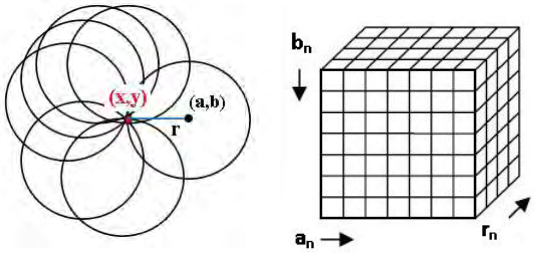
Resulting that

$$a = r\cos\phi - x$$
$$b = r\sin\phi - y.$$

The Hough transform finds all circles with an specific radius r, with an angle θ and centered in the point (a, b) (see Figure 9a). The accumulator array in this case has 3 dimensions (Figure 9b) given by the three parameters of equation 11. Then, each cell of the accumulator array gives the number of pixels lying in a circle with parameters a, b and r. The cell containing a peak represents the circle with highest number of pixels in the image. However, the existence of a peak do not imply the presence of a circle in the image, because the number of pixels may be insufficient to form a circle. For that reason, it is vital to specify for every radius, the number of pixels (value of the peak) requires to get a circle. This is done in accordance with the circumference of a circle given by

$$c = 2\ \pi\ r. \tag{12}$$

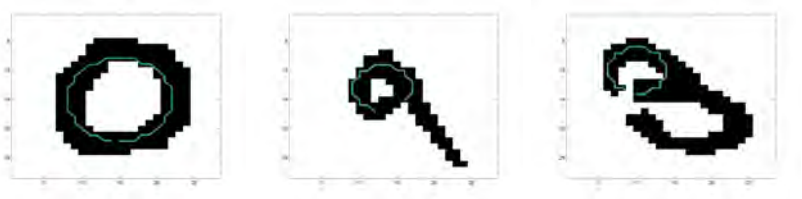
In our work, a circle is selected if at least the 80% of the points composing the circumference are lying on pixels. For our databases, we observe that in any case the 100% of the circle points are lying on the pixels of the digit. According to the



(A) (B)

FIGURE 9. Hough transform

size and shape of the handwritten numbers, we fix the frequent radiuses of circles that can be formed in the digits. We also analyze the alternative locations of those circles in order to fix the range of values of parameters a and b and select circles with radiuses equal to 4, 5, 6, 7 and 8. Those values are chosen to capture the possible circles contained in digits zero, six, eight and nine. In the case that a circle with $r = 4$ or $r = 5$ is found in the upper part of the digit, another circle with similar radius is searched in the lower part. This searching is done to find the two circles contained in digit eight. Finally, the values of the variables selected for the classification are given by the radius and the coordinates of the start and end-point of the selected circle. That represents a total of 5 variables per image corresponding to the circle detection, 4 values are the coordinates of start and end-points and 1 value is the radius. Examples of circles found in the digits are shown in Figure 10. In those cases where the digits do not have circles, these variables assume value



(A) Radius=7 (B) Radius=4 (C) Radius=4

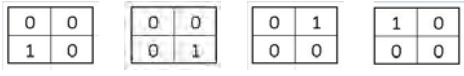
FIGURE 10. Examples of circles

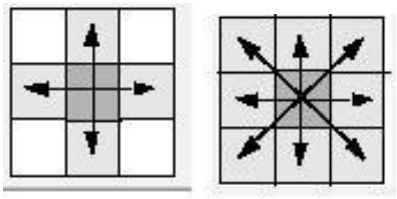
zero.

3.2. **Euler number.** The *Euler* number (E) is a measure of the topology of an image, specially used in binary representation. The *Euler* number of binary images can be calculated based on local measures, i.e., from pixel neighborhood relation. Suppose that we consider as neighbors only the four pixels that share an edge (not a corner) with a particular pixel (x, y). The neighbors are $(x + 1, y)$, $(x - 1, y)$, $(x, y + 1)$, and $(x, y - 1)$. In this situation we have 4-connected neighbors and the connection is defined as 4-connectivity (see Figure 11a). An alternative is to consider a pixel as connected not only by pixels on the same row or column, but also by the diagonal pixels. The four 4-connected pixels plus the diagonal pixels are called 8-connected neighbors, and the connection is defined as 8-connectivity (see Figure 11b). The alternative calculation of *Euler* number based on the connectivity is obtained differently in each case (Lin et al. (2006) and Lin et al. (2007)). Denoting as $E(4)$ and $E(8)$ the euler number calculated by 4 or 8-connectivity respectively, then

$$E(4) = \frac{(S_1 - S_3 + 2 \times X)}{4}$$
$$E(8) = \frac{(S_1 - S_3 - 2 \times X)}{4},$$

where S_1 is the number of the following structures in the binary image

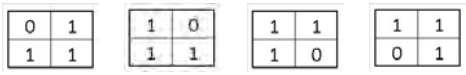




(A) 4-connectivity (B) 8-connectivity

FIGURE 11. Euler Number

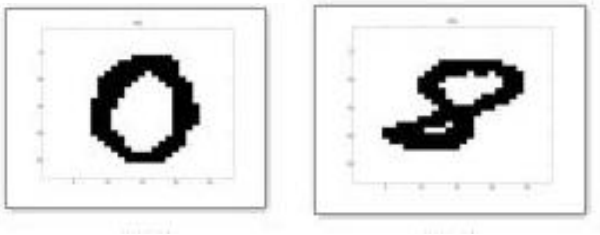
the S_3 is number of the following structures that are in the binary image



and the X is the number of the following structures that are in the binary image

In our work we use the pixels neighborhood to find the *Euler* Number in 2D images, which is the procedure used in some computational programs. Specifically, we use the 8-connectivity.

3.3. **Holes.** The variable *hole*(H) is specially programmed for this work. It finds holes in the digit and its location up and/or down. If the digit does not have a hole, the variable assumes the value zero. We also consider the case of bigger holes which characterizes the digit zero. In 12a the number zero has a bigger hole and in 12b the digit has two holes, up and down.

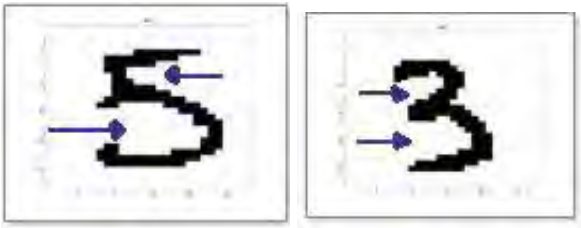


(A) Bigger hole (B) Holes up and down

FIGURE 12. Hole variable

3.4. **Right and left entries.** The feature *right entry* (R) is programmed to find if a digit has a right entry and if it is up or down. For example, the number five has a right up entry. The variable *left entry* (L) finds if a digit contains a left entry located up, down or both, like a digit three. If there is no entry, the variable assumes value zero. Examples of these variables are shown in 13. In 13a the digit

has an entry left down and also an entry right up. The digit in 13b has two entry by left (up and down) and zero entry in the right. The arrays in the figure indicate the orientation and location of the entries.



(A) L=down, R=up entry

(B) L=up and down

FIGURE 13. Entry variables

3.5. **Cross in the center.** The variable *cross in the center* (C) is defined to detect if a digit has a cross of the digit trace in its center. The shape of a digit with this characteristic is shown in Figure 14. The number that contains exactly this shape in the center is digit eight. However, we also consider a digit that has a middle cross to upwards or downwards like a nine or six respectively. If the digit does not have cross in the center such as the digit one, the variable assumes value zero.

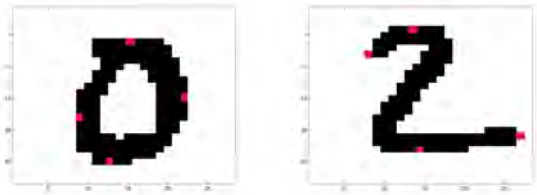
FIGURE 14. Cross in the center variable

Examples of this feature are shown in Figure 15. In (a) the number has a complete cross, in (b) the digit has a middle cross to downwards, and in (c) the nine has a middle cross upwards.

3.6. **Extremes.** The variable *extreme* (E) tries to define the contour of a digit by identifying four extreme black pixels of it: the northernmost (E_n) , the southernmost (E_s), the easternmost (E_e) and the westernmost (E_w) pixel. If there are more than one pixel occupying one of these extreme location, the pixel situated nearest the central part of the image is chosen. The values of the variable *extreme* are the coordinates of the extreme pixels. As the extreme pixels are four, this variable is characterized by eight values. Examples of this variable are shown in Figure 16 where the extreme pixels are depicted in red color.

(A) Cross in the center (B) Middle cross down (C) Middle cross up

FIGURE 15. Examples of cross in the center variable

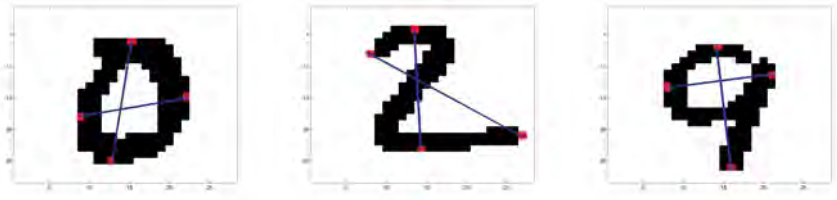


(A) (B)

FIGURE 16. Examples of extremes variable

3.7. **Intersections.** The variable *intersections* (I) is obtained considering the extreme pixels E_n, E_s, E_e, and E_w, previously defined. These pixels help to draw two imaginary lines in the image. One of them goes from E_n to E_s and the other line goes from E_e to E_w. Therefore, the *intersections* variable counts the number of times that each imaginary line is intercepted for the trace of the digit (continues trace that define the digit).

Due to we have two imaginary lines, the variable *intersections* has two values, that is the number of intersections of each line. In the example (a) of Figure 17 the variable has values $(0, 0)$ for both lines, while in example (b) the variable has values $(1, 1)$. In Figure 17c the values of the variable are $(0, 1)$.



(A) I= (0,0) (B) I= (1,1) (C) I= (0,1)

FIGURE 17. Examples of intersection variable

3.8. **Distance.** The *distance* variable is obtained by black pixels located as near as possible to the corners of the image. The corner pixels are the black pixels of the digit located at the northwest corner (co_{NW}), southwest corner (co_{SW}), northeast corner (co_{NE}) and southeast corner(co_{SE}). Two distances are obtained from the coordinates of those pixels, the distance between the northeast and southwest corner pixels, denoted as d_{EW}, and the distance between the northwest and southeast corner pixels denoted as d_{WE}. The distance used is the Euclidean. As results, two values per image are obtained, i.e. $d_{EW} = d(co_{NE}, co_{SW})$ and $d_{WE} = d(co_{NW}, co_{SE})$.

4. Probabilistic classification approach

Once the features that we use in the classification scheme are introduced, we give some details of the two statistical approaches developed in this paper.

In the first approach the classification is performed by the application of the Bayes' Theorem. Being X the feature vector extracted of an image, the decision rule is stated by deciding that an image belongs to the class c_k if $p(c_k/X) > p(c_j/X)$ for all $k \neq j$. The posterior probability $p(c_k/X)$ is calculated using Bayes' Theorem, as follows

$$p(c_k/X) = \frac{p(X/c_k)p(c_k)}{p(X)}. \tag{13}$$

Given that the ten classes of digits have almost the same proportion inside the databases, in this paper we consider the same prior probability $p(c_k)$. Then, the posterior probability can be expressed as

$$p(c_k/X) \propto p(X/c_k)\ , \tag{14}$$

where $p(X/c_k)$ is the class-conditioned probability of a feature vector X. This probability is used to find $p(c_k/X)$, i.e., the membership probability to a class c_k of an image in the testing set with feature vector X_{test}. In order to develop an appropriate probabilistic background to find this probabilities, the variables involved in the classification are divided in categorical and quantitative, since we consider that each group requires different technique to be statistically modelled. In the first group we include the variables cross in the center (C), euler (E), hole (H), right entry (R) and left entry (L), forming the feature vector called X_{cat}. In the second group, we consider the variables straight line coordinates (S), diagonal line coordinates (D), straight and diagonal line length (Le), horizontal and vertical intersections (I) and extreme (Ex), composing the feature vector denoted as X_{quant}. The training set is used to find the probability of categorical and quantitative variables of each class expressed in equation 15. After analyzing the data in training set we can conclude that each group of variables can be treated as independent. In symbols, the probability that an image from training set with feature vector X belongs to a class c_k, is given by

$$P(X/c_k) \propto P(X_{cat}/c_k) \times P(X_{quant}/c_k). \tag{15}$$

To obtain the class-conditioned probabilities of the categorical feature vector we calculate the join probabilities of occurrence of the five variables (C, E, H, R, L) for a given class c_k, using a frequentist procedure through the training set. Sometimes, this way of calculation causes a problem due to the lack of observations in the combinations. Besides, the analysis of data suggests that there exists dependence only among some variables. To solve those troubles, we state by cross-validation

that the best joint dependence structure for categorical data can be defined by the specific scheme showed in the following equation.

$$P(X_{cat}/c_k) \approx P(L/c_k) \times P(H/L,\ c_k) \times P(C/L,\ c_k) \times P(R/C,\ c_k) \times P(E/R,\ c_k) \tag{16}$$

The probability density function of quantitative data (S, D, Le, I, Ex) is modelled by parametric estimation, assuming the multivariate normal distribution to calculate the density inside each class, commonly used for integer-valued features [see Jain et al. (2000)]. The density of an image with a feature quantitative vector X_{quant} given a class c_k, is obtained as follows

$$f(X_{quant}, \mu, \Sigma/c_k) = \frac{1}{\sqrt{|\Sigma_{c_k}|(2\pi)^d}} \exp^{-\frac{1}{2}(x-\mu_{c_k})\Sigma_{c_k}^{-1}(x-\mu_{c_k})'}, \tag{17}$$

where the mean μ and covariance matrix Σ are estimated from training set in the reference class (equation 17). The probability density function of each class is valued at the quantitative feature vector for every image in the test set. Moreover, the posterior probability for categorical data in the test set is obtained by equation 16. Lastly, assuming independence between quantitative and categorical data, the final posterior probability that an image in the test set belongs to a class c_k is given by equation 15.

The classification process is performed in two stages. In the first stage, we find by cross-validation a cutting point (denoted as p) to classify a subset of the test set. The cutting point value obtained is 0.999. That is, an image with a class probability greater or equal to the cutting point is classified into this class. Otherwise, if none class achieves the 0.999 of probability or more, the image is submitted to a second stage.

In the second stage we have a subgroup of images (called $test2$) that represents the 16,29% of the test set. The procedure in this stage is similar to the previous one. However, in this case we consider as possible outcomes of each digit in $test2$, only a pair of classes. These classes, denoted as c_i and c_j, are those that have the greatest probabilities of occurrence in the first stage. Thus, we perform the application of Bayesian rule by considering a particular feature vector with the variables considered more discriminant between the classes c_i and c_j in each case. After the process is finished, a digit is classified to a class with greater probability of both. Finally, the test-error rate is calculated achieving a $4,3\%$ for *MNIST* database and a test-error rate of $9,7\%$ in *USPS* dataset. Table 3 shows the classification rates for every number.

TABLE 3. Probabilistic approach results

database	0	1	2	3	4	5	6	7	8	9
MNIST	3.57%	2.82%	5.43%	5.05%	6.62%	4.82%	3.86%	7.78%	5.54%	7.53%
USPS	7.0%	8.0%	8.1%	14.5%	18.1%	11.1%	10.3%	9.5%	9.4%	4.0%

According with the results, the procedure has better performance in *MNIST* database than in *USPS*. This behavior could be a consequence of the difficulty of digits in *USPS* to be recognized. The results show that the digits with worse error rate are the seven and nine in *MNIST* dataset and the digit three and four in the *USPS* dataset.

5. K nearest neighbor classification approach

In this approach, the classification is performed by the *nearest neighbor method* [Cover (1968)], using the variables specially proposed in Section 3 for this kind of problems [3].

By means of k-nearest-neighbor algorithm, the *training* set feature vectors are used to classify the *test* set. An image is classified by a majority vote of its neighbors, i.e.,

[3]LeCun et al. (1998) and Smith et al. (1994) used all pixels of the image as feature vector, requiring more computer memory and running time

it is assigned to the class most common among its k nearest neighbors (majority rule). The distance city-block, calculated as the sum of absolute differences, is chosen because it provides better performance with integer variables. With respect to the value of the parameter k, there is no consensus in the bibliography in defining the adequate number of k in nearest neighbor classification. Previous works [Hall et al. (2008)] on nearest-neighbor classifiers held the value of k by cross-validation which is the method that we use. Specifically, we choose 5 neighbors. The algorithm is applied to the features defined in Section 3 following the sequential forward selection [also known as wrapping procedures, see Karegowda et al. (2010) and Kohavi and John (1997)], in order to eliminate possible redundant information. We choose the *forward* selection in order to include the minimum number of variables in the classification[4]. This technique, widely used in regression problems [see Hastie et al. (2009)], permits to find the subset of variables that minimize the error rate (percentage of misclassified images in the test set). The selection of features is done sequentially [see Kohavi and John (1997), John et al. (1994) and Karegowda et al. (2010)], by adding features to the model one at a time. The procedure begins with an empty feature set and sequentially adds features. The first feature included in the model is the one which individually has the lowest error-rate. The next feature that enters the model is the one that, jointly with the first variable, has the greatest reduction in the error-rate. The process of adding features continues until including a new one does not decrease the error-rate. The resulting subset of features is the ones used to discriminate both classes.

The sequential forward selection technique requires the application of cross-validation method in order to avoid the overfitting of the error-rate. Cross-validation is used to assess how the results of a statistical analysis (in our case, KNN and LDC) will generalize to an independent data set. One round of cross-validation involves the partition of the data into subsets, performing the analysis on one subset (i.e., training set), and validating this analysis on the other subset (i.e., testing set). To reduce variability, multiple rounds of cross-validation (called K-fold cross-validation) are performed using different partitions, and the validation results are averaged over the rounds. Otherwise stated, we divide the data into K roughly equals parts and for each $k = 1, \ldots, K$ we apply the classification method using a subset of p features $Z = (Z_1, Z_2, \ldots, Z_p)$. Then, we compute the error rate for each k-fold as

$$E_k(Z) = \frac{n_{Ek}}{n_k},$$

where n_{Ek} is the number of misclassified observations (images) in the kth part (fold) and n_k is the size of the kth fold. Hence, the overall cross-validation rate is

$$CV_E(Z) = \frac{1}{K} \sum_{k=1}^{K} E_k(Z). \tag{18}$$

The procedure repeats the cross-validation for different subset of features and selects the combination of them that minimize the $CV_E(Z)$. For this work we choose the commonly used 10-fold cross-validation.

As a results of the sequential selection, 21 variables are included in the classification of *USPS* database and 27 (5 additional different) variables in the *MNIST* database. The excluded features are refereed to redundance information about coordinates of straight and diagonal lines, coordinates of the extreme variable and the information about circles in the image. The five additional variables included in the classification of *MNIST* dataset are some coordinates of the extreme variable. One possible reason to this discrepancy in the number of variables selected could be the fact that the *USPS* set has more heterogeneity in digit shape. Thus, the extremes used in the variable do not work as well as in *MNIST*

[4]We also perform the feature selection by backward elimination and some variables were retained unnecessarily making the model less efficient.

dataset. The total error rate obtained for the MNIST database is $3,65\%$ and $4,39\%$ for the *USPS* dataset. The outcomes for each database are shown in Table 4.

Table 4. K- nearest neighbors results

database	0	1	2	3	4	5	6	7	8	9
MNIST	1.33%	0.79%	3.20%	5.64%	4.07%	4.15%	1.77%	3.40%	7.08%	5.25%
USPS	0.75%	1.52%	5.9%	6.3%	6.54%	3.91%	4.63%	1.36%	7.66%	4.2%

The results show that the digits with worse error rate in both databases are the three and the eight. It is interesting to observe that in k-nearest neighbor the digits seven and nine have much better error rates than digit eight, while in the probabilistic approach occurs the opposite. This behavior may show a future line of investigation combining both approaches in order to improve results.

6. Conclusions and contributions

In general, previous work in handwritten digit recognition contemplate neural network classifiers to perform the classification. In this paper we propose a method based in a multivariate statistical approach. Our study is less sophisticated and obtain competitive results in this area. Other authors work on this database usesing a baseline nearest neighbor algorithm [see LeCun et al. (1998) and Smith et al. (1994)] to classify the digits directly by the pixels value (784 values per image). We perform the k nearest neighbors technique on the feature vectors of the images (27 values per image) which provides good results on the same dataset and requires less computer memory and recognition time. The proposed variables were specially programmed for this work and they can be easily generalized to be use in any digit database. As the variables are calculated in binary image, we propose a novel method to binarize an image through an optimization procedure that finds the best trace. In addition, we propose an alternative probabilistic approach with similar results than the k nearest rule. Differ from other methods [see Bottou et al. (1994)], the proposed techniques permit to quantify the individual contribution of the variables. They can also be applied easily to different datasets and no changes in the values of the variables are observed by resizing the image.

7. Future works

There are some natural extensions to this work that would help expand and strengthen the results.

Firstly, we aim to combine the binarization method proposed in this work with other methods. We observe that our procedure is based on the trace of the digit, while, for instance, the Otsu' method [see Otsu (1979)] is based on minimizing the variances between the blacks and white pixels. We are interested in setting out the binarization problem in terms of a family of thresholds.

Secondly, we aim to analyzed the inclusion of different coefficients of weight for each variable, in accordance with their contribution in the classification process.

Finally, we are concerned about studying the combination of the classification procedures proposed in section 4 and 5. We think that the performance of K-nearest neighbor technique could be improved, may be by adding some information about the nearest neighbors [see Domeniconi et al. (2005), Han et al. (2001), Hastie and Tibshirani (1996), Dudani (1976) and Bailey and Jain (1978)].

One extension that would be interesting is to estimate the posterior probability by the application of the K-nearest neighbor classifier [Atiya (2005) and Fukunaga and Hostetler (1975)]. In this regard, we already initiated some studies but without concluding results.

References

Atiya, A. F. (2005). Estimating the posterior probabilities using the k-nearest neighbor rule. *Neural Computation 17*, 731–740.

Bailey, T. and A. K. Jain (1978). A note on distance-weighted k-nearest neighbor rules. *IEEE Transactions on Systems, Man and Cybernetics 8*(4), 311 –313.

Ballard, D. (1981). Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition 13*(2), 111 –122.

Bottou, L., C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, and V. Vapnik (1994). Comparison of classifier method: A case study in handwritten digits recognition. *Pattern Recognition 2*.

Cover, T. (1968). Estimation by the nearest neighbor rule. *Information Theory, IEEE Transactions on 14*(1), 50 – 55.

Decoste, D. and B. Scholkopf (2002). Training invariant support vector machines. *46*, 161–190.

Domeniconi, C., D. Gunopulos, and P. Jing (2005). Large margin nearest neighbor classifiers. *Neural Networks, IEEE Transactions on 16*(4), 899 –909.

Duda, R. and P. Hart (1972). Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM 15*(1), 11–15.

Dudani, S. A. (1976). The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man and Cybernetics, SMC-6*(4), 325 –327.

Fukunaga, K. and L. Hostetler (1975). K-nearest-neighbor bayes-risk estimation. *IEEE Transactions on Information Theory 21*(3), 285–293.

Gonzalez, R., R. Woods, and S. Eddins (2004). *Digital Image Processing using MATLAB*.

Hall, P., B. Park, and R. Samworth (2008). Choice of neighbor order in nearest-neighbor classification. *Annals Statistics 36*(5), 2135–2152.

Han, E.-H., G. Karypis, and V. Kumar (2001). Text categorization using weight adjusted k-nearest neighbor classification. In *Proceedings of Conference on Knowledge Discovery and Data Mining*, pp. 53–65.

Hastie, T. and R. Tibshirani (1996). Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence 18*(6), 607–616.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

Hough, P. V. C. (1962). Method and means for recognizing complex patterns.

Jain, A. K., R. P. W. Duin, and J. Mao (2000). Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*(1), 4–37.

John, G. H., R. Kohavi, and K. Pfleger (1994). Irrelevant features and the subset selection problem. pp. 121–129. Morgan Kaufmann.

Karegowda, A. G., M.A.Jayaram, and A. Manjunath (2010). Feature subset selection problem using wrapper approach in supervised learning. *International Journal of Computer Applications 1*(7), 13–17.

Keysers, D., T. Deselaers, C. Gollan, and H. Ney (2007). Deformation models for image recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 29*(8), 1422–1435.

Kohavi, R. and G. H. John (1997). Wrappers for feature subset selection. *Artificial Intelligence 97*(1-2), 273 – 324.

Lauer, F., C. Suen, and G. Bloch (2007). A trainable feature extractor for handwritten digit recognition. *Pattern Recognition 40*(6), 1816–1824.

LeCun, Y., B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson (1990). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems 2*, 396–404.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324.

Lin, X., J. Ji, and Y. Gu (2007). The euler number study of image and its application. pp. 910 –912.

Lin, X., Y. Sha, J. Ji, and Y. Wang (2006). A proof of image euler number formula. *Science in China Series F: Information Sciences 49*, 364–371.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics 9*(1), 62–66.

Simard, P., Y. LeCun, and J. Denker (1993). Efficient pattern recognition using a new transformation distance. *Advances in Neural Information Processing Systems 5*, 50–58.

Smith, S., M. Bourgoin, K. Sims, and H. Voorhees (1994). Handwritten character classification using nearest neighbor in large databases. *IEEE Transactions on Pattern Analysis and Machine Intelligence 16*(9), 915 –919.

Universidad Carlos III de Madrid, Spain
E-mail address: mgiuliod@est-econ.uc3m.es