Statistical Learning Assignment
Universidad Carlos III de Madrid
January 2020
Marta Cortés

Leaf Type Plant Classification

1 Introduction and Data Set

Plants, as creatures in the animal kingdom, are all unique regarding external appearance, structure and physiological behaviour. Also, they differ in their habitats and requirements. With that diversity, the exactly way to classifies them has always been a big question. Classifying plants is considered as one of the oldest approaches in studying botany.

Traditionally, plant recognition has been done by specialized taxonomist, who use attributes like the shape of leaves, color of flowers, aspect of fruits, and others criteria.

The technology and image and signal processing techniques development has motivated a growing tendency in automation, replacing the conventional methodologies, likewise it provides a good tool with reduced or, even, no costs.

The data set used for this project has been extracted from the Master's Thesis 'Development of a System for Automatic Plant Species Recognition' by Pedro Filipe Barros Silva (2013).

This data set consists in a collection of shape and texture features extracted from digital images (previous photographed) of leaf specimens originating from a total of 30 different plant species shown in the Figure 1.

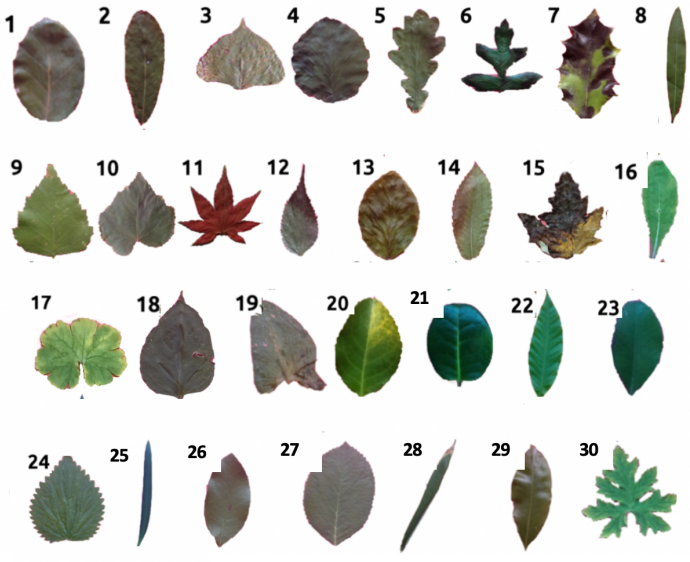

Figure 1: Leaf database overview of each plant specie

The data contains 340 records of leaves, homogeneously distributed respect the quantity of each plant species (average of 10 records per specie). The species names are listed in the Table 1.

Table 1: Leaf database: plant specie scientific name

Class	Scientific Name	Class	Scientific Name	Class	Scientific Name
1	Quercus suber	11	Acer palmatum	21	Ilex perado
2	Salix atrocinera	12	Celtis sp.	22	Magnolia soulangeana
3	Populus nigra	13	Acer palmatum	23	Buxus sempervirens
4	Alnus sp.	14	Castanea sativa	24	Urtica dioica
5	Quercus robur	15	Populus alba	25	Podocarpus sp.
6	Crataegus monogyna	16	Primula vulgaris	26	Acca sellowiana
7	Ilex aquifolium	17	Erodium sp.	27	Hydrangea sp.
8	Nerium oleander	18	Bougainvillea sp.	28	Pseudosasa japonica
9	Betula pubescens	19	Arisarum vulgare	29	Magnolia grandiflora
10	Tilia tomentosa	20	Euonymus japonicus	30	Geranium sp.

The features of the leaves have been registered as predictors variables. Also, in order to be able to perform this project, it has been add another categorical variable as the response. This variable is the type of the leaf for each specie: Evergreen or Deciduous.

All of the variables are described in the Table 2 and 3.

Table 2: Statistical Variables of the Data Set

Variable	Description*(*)*
Eccentricity	Eccentricity of the ellipse with identical second moments to I *(*)*.
Aspect Ratio	Consider any X,Y E ∂I. Choose X and Y such that d(X,Y)=D(I). The aspect ratio is defined as the quotient $D(I)/D^{\perp}$. Values close to 0 indicate an elongated shape.
Alongation	Compute the maximum escape distance $dmax = maxXEId(X, \partial I)$. Elongation is obtained as $1 - 2dmax/D(I)$ and ranges from 0 to 1. The minimum is achieved for a circular region.
Solidity	The ratio $A(I)/A(H(I))$ is computed, which can be under- stood as a certain measure of convexity.
Stochastic Convexity	The aim is to estimate the probability of a random segment $[XY], X, YEI$, to be fully contained in I.
Isoperimetric Factor	The ratio $4\pi A(I)/L(\partial I)2$ is calculated. The maximum value of 1 is reached for a circular region.
Maximal Indentation Depth	The indentation function can then be defined as $[d(X, C_{H(I)}) - d(Y, C_{H(I)})]/L(H(I))$, which is sampled at one degree intervals. The maximal indentation depth D is the maximum of this function.
Lobedness	Calculate lobedness as $F * D2$, where F stands for the smallest frequency at which the accumulated energy exceeds 80%. This feature characterizes how lobed a leaf is.
Average Intensity	Average intensity is defined as the mean of the intensity image
Average Contrast	Average contrast is the the standard deviation of the intensity image
Smoothness	Smoothness is defined as $R = 1 - 1/(1 + \sigma 2)$ and measures the relative smoothness of the intensities in a given region.
Third moment	is a measure of the intensity histogram's skewness.
Uniformity	Uniformity's maximum value is reached when all intensity levels are equal.

Table 3: Statistical Variables of the Data Set

Entropy	A measure of intensity randomness.
Leaf Type	Evergreen if the plant retain leaves at all times, or Deciduous if it shed its leaves at the end of the growing season.

(*)Let I denote an object of interest in a binary image, ∂I its border, D(I) its diameter, i.e.
(*)For further information about the calculus of the predictors variables, see: Master's Thesis 'Development of a System for Automatic Plant Species Recognition', 2013.

The main objective of this project is to provide an statistical classification analysis using the image features database of the collected leaves.
The developing of an automatic prediction model for classifying a new leaf observation regarding its leaf type, is also developed using the R tool.

2 Data Processing and Statistical Visualization

Initially, the data has been reviewed in order to treat the skewed values. No missing values has been found as well as some other wrong data.

As it has been said before, one more column about the leaf type has been added into the data set for the response analysis.

In order to visualize the data, extract useful information from it, and be able to explain the behaviour of our variables, some graphs has been performed through R tool.

First, it has been represented in the Figure 2 the distribution of each type of leaf from all the recorded species. As it can been seen, there are more or less the same quantity of records for each type of leaf, which is a good balance the next prediction sections.

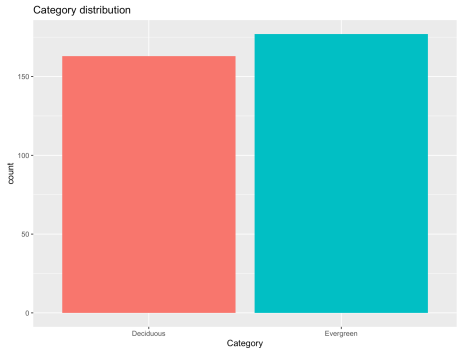

Figure 2: Leaf database overview of each plant specie

A multi-chart of the densities (figure 3) and the QQ-plots (figure 4) of the variables has been performance since it is very important to know the distribution and the behaviour of our predictors.

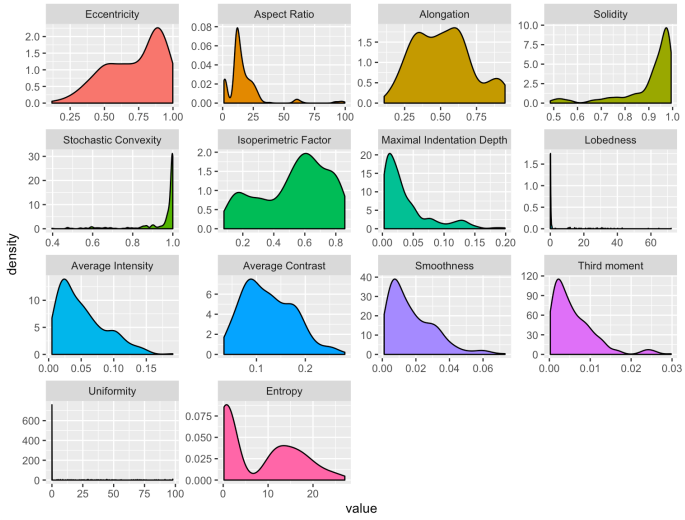

Figure 3: Distributions of the predictor variables

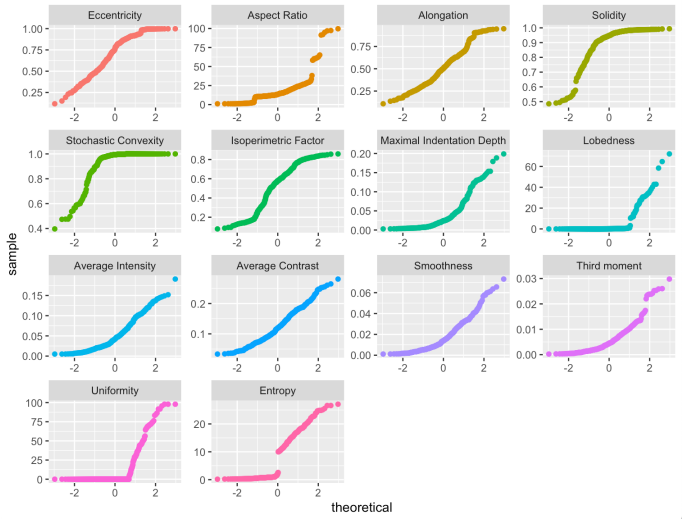

Figure 4: QQ-Plots of the predictor variables

Since the distribution of some variables don't follow normality, we have transformed them. The next table shows the transformations we have applied.

Table 4: Transformation of the variables

Variable	Transformation
Eccentricity	$(100 - Eccentricity)^2$
Aspect Ratio	$log(AspectRatio + 1)$
Solidity	$(2 - log(Solidity))^{-50}$
Stochastic Convexity	$-1 * (log(StochasticConvexity + 1))^{50}$
Isoperimetric Factor	$-1 * (IsoperimetricFactor)^{1.5}$
Maximal Indentation Depth	$log(MaximalIndentationDepth)$
Lobedness	$log(8 + log(Lobedness))$
Smoothness	$log(Smoothness)$
Third moment	$log(Thirdmoment)$
Uniformity	$-1 * (1 + log(Uniformity))^4$
Entropy	$-1 * (log(Entropy - 0.15))^2)$

The figures 5 and 6 show the same graphs than before with the densities and QQ-plots of the variables already transformed.

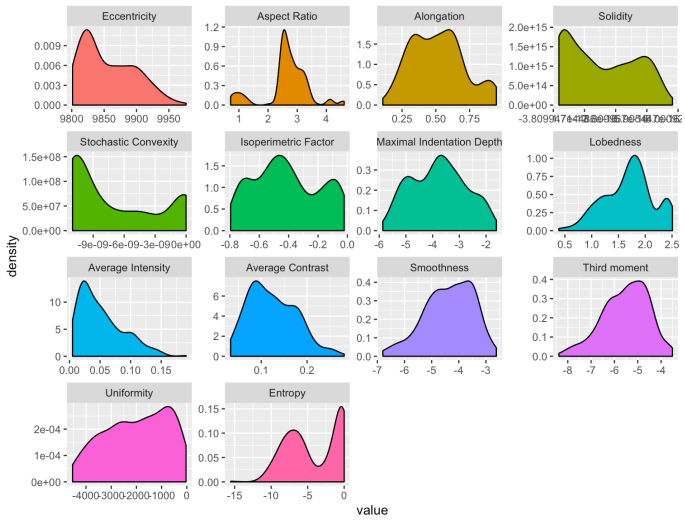

Figure 5: Distributions of the transformed variables

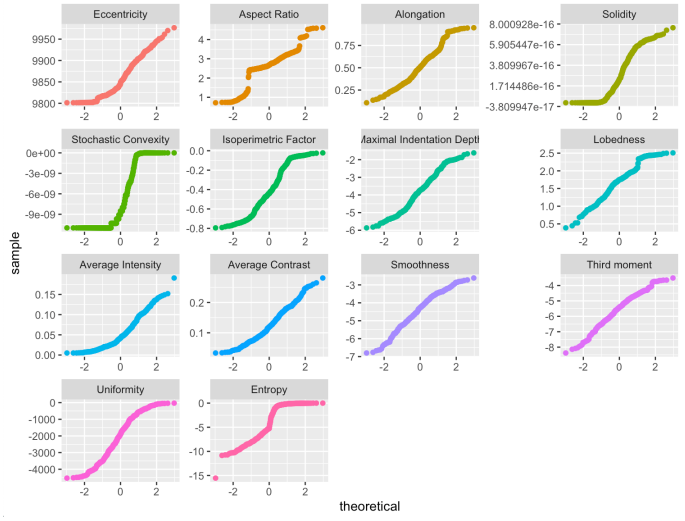

Figure 6: QQ-plots of the transformed variables

The variables, also, should be scaled in order to standardize and to have them with the same mean (0) and standard deviation (1). We use the function "scale" in R Studio to do so.
In the next figure, we can observe the boxplots of the variables and as they are attuned.

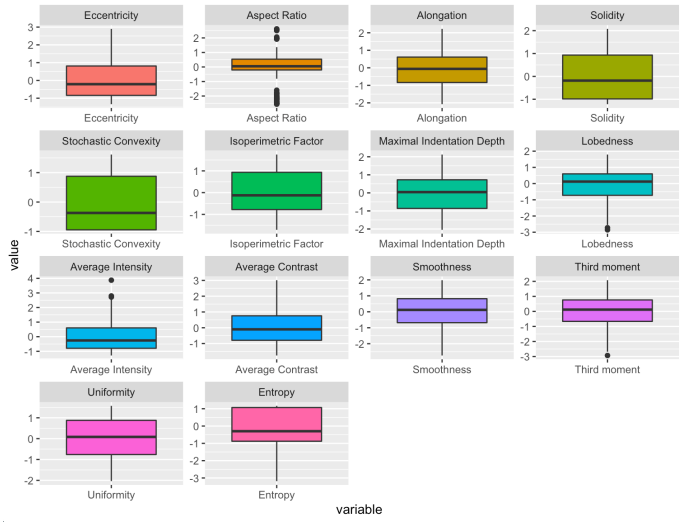

Figure 7: Boxplots of the transformed predictor variables

A summary graph about the densities distribution of the transformed variables regarding the two categories of the leaf type is shown in the next figure (Figure 8).

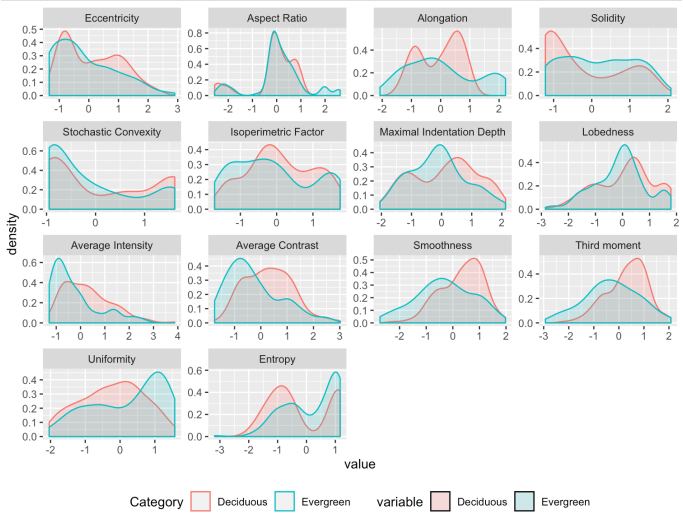

Figure 8: Distribution of the predictor variables per each category

It can been said that the two groups aren't too well-separated in most of the predictors as they don't conform two independent groups, but the distributions overlap each other. This, could hinder the good performance of our prediction in the next sections.

In any case, it can been said that Alongation, Smoothness and Third Moment are the categories that classify the better.

Finally, a correlation matrix (Figure 5) has been performed in order to show the association between the predictors.

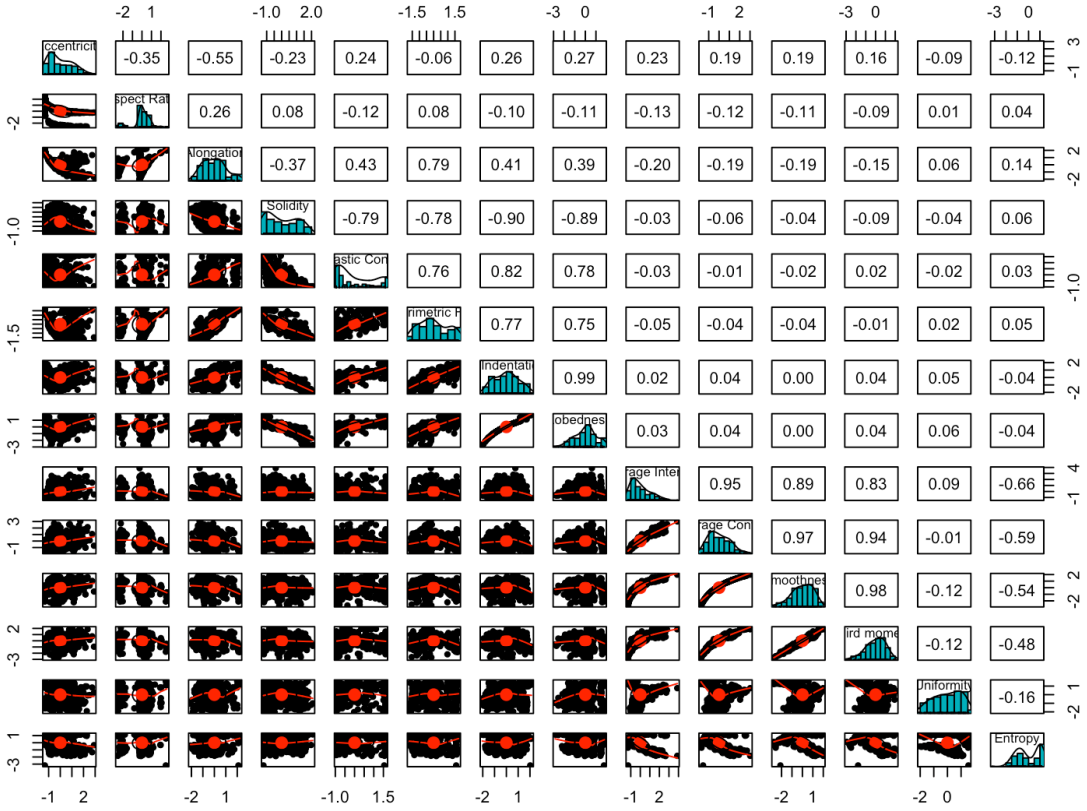

Figure 9: Correlation Matrix of the predictor variables

As it can been observed, there is high multicollinearity, more than 80% of correlation between some variables in few cases, as *Convexity* with *Indentation*, *Solidity* with *Lobedness* or *Indentation* with *Lobedness*, etc.

This can give redundant information about the response. Sometimes more information (more variables) is not necessary better, because this could produce much more noise in our prediction model. To avoid this, a PCA has been developed in the next section as effectively technique for dimensionality reduction and variable decorrelation.

2.1 PCA

A PCA (Principal Component Analysis) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. This transformation is defined in such a way that the first principal component has the largest possible variance, and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components.

Applying this technique through R, it can be said only 3 components are really uncorrelated and explain more than the 70% of the data variability as we can see in the figure 10.

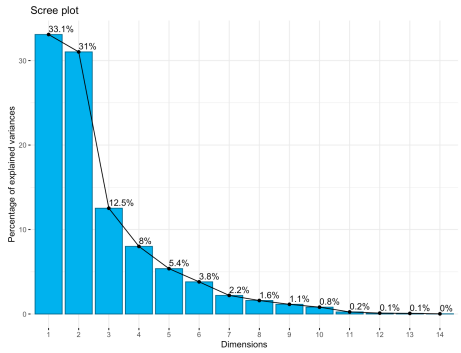

Figure 10: Percentage of variability explained for each PCA

This four principal components are compounded for particular weights of the data variables, as we can see in the figure 11, in which we have plotted the first PCA vs the second, we can find the variables that influence the most in the first (the ones that are close to the X axis) and that one for the second (closed to the Y axis). The figure 12 shows the correlations of the variables with each PCA.

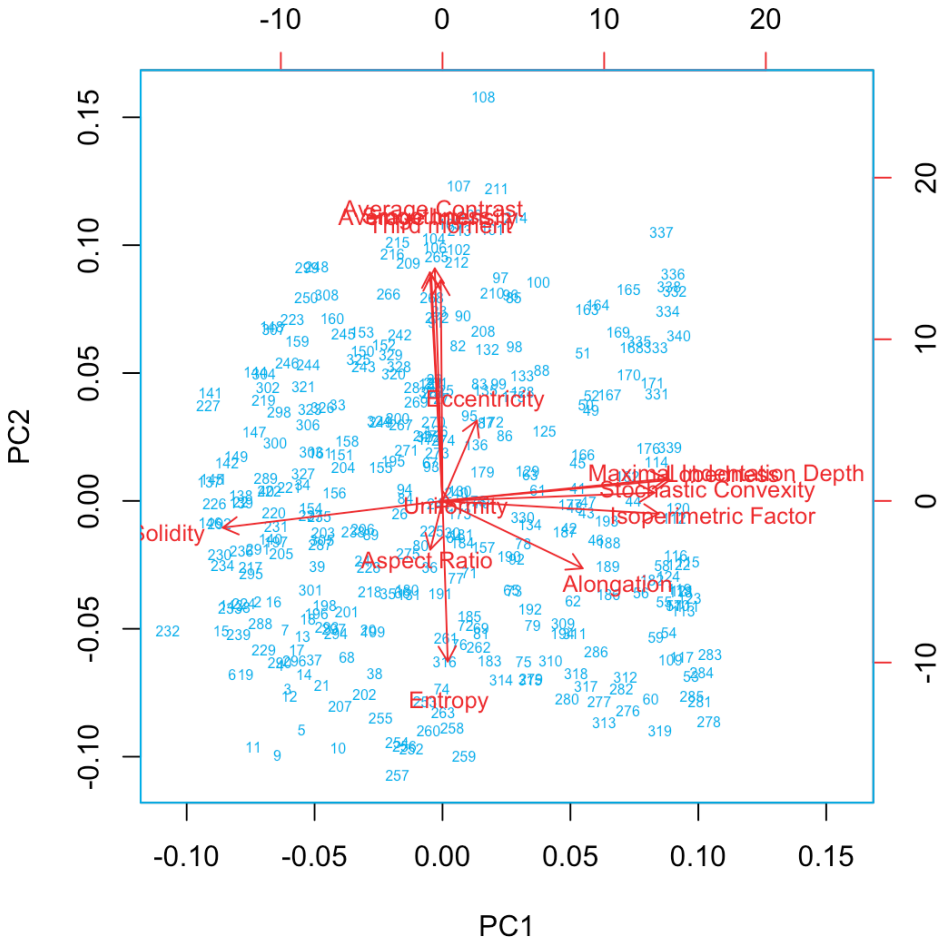

Figure 11: First two PCAs and the variables

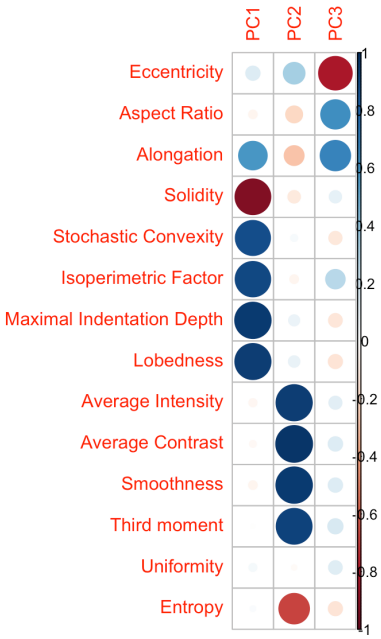

Figure 12: Correlation between variables and PCAs

In the next sections it will be seen how the selection of less variables can affect in the achievement of the predictions.

3 Statistical Classifications

Classification is the matter of identifying to which of a set of categories (sub-populations), in this case, leaf type, a new observation belongs, on the basis of a training set of data containing observations whose category membership is known.

In this model, the variable response (y) is the leaf type, while the rest of the variables are the predictors (x).

There are few techniques that let us forecast to which category belong a new observation. These techniques can be classified as following:

1. Logistic Regression: it is more used with continuous quantitative response variables, but it can be applied for qualitative responses after transformation to binary form. It works better for binary classification. It uses a logistic function to model a binary dependent variable.

2. Bayes Classifiers: are a family of simple "probabilistic classifiers" based on the conditional probability, applying Bayes' theorem with strong (Naïve) independence assumptions between the features. It works better with multi-classification.

3.1 Logistic Regression

In logistic regression, we use the logistic function, to fit the model:

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

The logistic function will always produce an S-shaped curve, and so regardless of the value of X, we will obtain a sensible prediction.

The quantity p is called the odds: $\frac{p}{1-p}$
We assume a linear relationship between the predictor variables, and the log-odds of the event that $Y = 1$. This linear relationship can be written in the following mathematical form (where ℓ is the log-odds, b is the base of the logarithm, and β_i are the parameters of the model):

$$\ell = \log_b \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

In this case, a negative value indicates most likely y = 0, whereas a positive value indicates most likely y = 1 (more symmetry).

If we increase an x-variable by one unit, the logit is increased by the corresponding β.

In order to perform the predictions, the data has been separated in "Training" observations (the 80% of the data) and "Testing observations", so by this way, we use the "training" ones to create the regression model and the "testing" to check how good the performance is.

The model has been done with all the variables and with the variables of the PCAs, in order to see if we can get better results with less dimensionality.

The R tool and the function "log.fit" has been used to perform the regression.

With all variables, the summary of the fit is detailed below:

```
Call:
vglm(formula = Category ~ ., family = multinomial(refLevel = 1),
    data = scaled.Leaves3)

Pearson residuals:
                      Min      1Q Median     3Q   Max
log(mu[,2]/mu[,1]) -5.195 -0.7534 0.1242 0.7568 2.743

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)         0.187656   0.130857   1.434  0.15156
Eccentricity        0.022655   0.243974   0.093  0.92602
AspectRatio         0.086683   0.135324   0.641  0.52181
Alongation         -0.245901   0.437758  -0.562  0.57430
Solidity            1.198661   0.371840   3.224  0.00127 **
StochasticConvexity -0.020404   0.255128  -0.080  0.93626
IsoperimetricFactor  0.577670   0.511113   1.130  0.25838
MaximalIndentation  -1.779678   0.932254  -1.909  0.05626 .
Lobedness            2.154818   0.848206   2.540  0.01107 *
AverageIntensity    -2.120141   1.050784  -2.018  0.04363 *
AverageContrast      4.127737   1.528981   2.700  0.00694 **
Smoothness          -2.097166   1.318864  -1.590  0.11181
ThirdMoment         -0.903544   1.403158  -0.644  0.51962
Uniformity           0.506144   0.155490   3.255  0.00113 **
Entropy             -0.003244   0.178217  -0.018  0.98548
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Name of linear predictor: log(mu[,2]/mu[,1])

Residual deviance: 381.6023 on 325 degrees of freedom

Log-likelihood: -190.8011 on 325 degrees of freedom

Number of Fisher scoring iterations: 5

No Hauck-Donner effect found in any of the estimates
```

Figure 13: Summary of the log.fit for all variables

The exponential of the coefficients (odds) can be calculate in order to better interpret the model. These values show the relation of the variable whether one group or the other, in

fact, if the values are higher than 1, the features better account for Deciduous leaves, since the reference level is Deciduous, and therefore, the odds less than one, are closer to Evergreen leaves.

(Intercept)	Eccentricity	AspectRatio	Elongation	Solidity	StochasticConvexity
1.2064181	1.0229136	1.0905514	0.7819998	3.3156756	0.9798029
IsoperimetricFactor	MaximalIndentation	Lobedness	AverageIntensity	AverageContrast	Smoothness
1.7818815	0.1686925	8.6263189	0.1200148	62.0373486	0.1228040
ThirdMoment	Uniformity	Entropy			
0.4051313	1.6588826	0.9967615			

Figure 14: Summary of the log.fit for all variables

Then, we can obtain the predictions of the testing data, with the function "predict" in R Studio, and assign to each observation, the corresponding category through maximum probability.

In order to measure the goodness of the different models, we have used as a metric, the accuracy, the error and the kappa coefficient (this last one, only for Bayes models).

- The accuracy ratio is the degree of closeness of measurements of a quantity, to that quantity's true value. The higher the better.

- The error ratio is the number of error occurrences divided by the total number of the observations. The lower the better.

- The Kappa coefficient is a measure of the agreement between two raters. In this case, the interpretation is more complicated, as it depends on the type carried analysis, so, for instance, health or science topics should need higher values of kappa while social topics can be validated with lower kappa coefficients. Cohen suggested the Kappa result be interpreted as follows: values ≤ 0 indicate no agreement, 0.01–0.20 as none to slight, 0.21–0.40 as fair, 0.41– 0.60 as moderate, 0.61–0.80 as substantial, and 0.81–1.00 as almost perfect agreement.

For this model, the confusion matrix, the errors and the accuracy are detailed below:

```
> ConfMat1

pred.test1 Deciduous Evergreen
         0        30        15
         1         2        20
> n = length(LeavesTest$Category)
> prop.errors1 <- (n - sum(diag(ConfMat1))) / n
> prop.errors1
[1] 0.2537313
> accuracy1 <- sum(diag(ConfMat1)) / n
> accuracy1
[1] 0.7462687
```

Figure 15: Results of the log.fit for all variables

The results are reasonable, but let's see next the rest of the performances.

So, we have done the same for the 8 predictors that contribute the most in the three PCAs.

In this case, this last model has performed worse than the one with all predictors (only 64% of accuracy). We can conclude, the predictions work better with more variables.

Also, we have tried with some interaction between the predictors. The results have been better than the previous model but this still being a bit worse than the one with all variables without interactions. So, from now, we have only considered all the variables, without interactions, to

still performing the next models.

The summary of the results are all compiled in the table 5 and all the details about the code can be founded in the Appendix.

3.2 LDA, QDA and Naive Bayes

These methods classify qualitative variables inside two or more known groups by the highest probability of belonging to the groups.

First, we model the (multivariate) distribution of predictors for each g: $f_g(x)$.

Then, we apply Bayes Theorem to obtain the posterior probabilities given a new observation x:

$$p_g(x) = P(y \in g|X = x) = \frac{f_g(x)\pi_g}{\sum_k f_k(x)\pi_k}$$

Finally, as before, we apply Bayes' rule to assign new observation x to that group with largest $p_g(x)$, i.e. $maxg\ fg(x)\pi g$
In practice, we just approximate or estimate $f_g(x)$ and πg. For instance, πg can be estimated using the proportion of training observations that belong to class: $g : \hat{\pi}_g = n_g/n$

LDA method is good and better than the Logistic Regression when the classes are well-separated, the number of observations is lower, and the distribution of the predictors follows a normality behaviour in each class.

In order to perform this method, we need to assume normality and equal variance in our data, even if we don't have normality at all.
This method has been developed by R tool using the function: "lda.model" and separating the sample in "Training" and "Testing" as we did before for the Logistic.

The results for this model are quite good, 0.73 of accuracy, nevertheless, the logit fit model has a bit better performance. This essentially could be because our groups aren't to well separated as we saw in the figure 8.

The kappa value is 0.46, which is moderately good.

QDA method is more unstable with large dimensions, although it provides less variance than the LDA.

While LDA can only learn linear boundaries, QDA can learn quadratic boundaries and is, therefore, more flexible.
This method has been developed by R tool using the function: "qda.model".

As we could expected, since the QDA is more wide, it has much better results than LDA, and also than the logit regression, with an accuracy of 78% and a kappa coefficient of 0.55 (slightly better than the previous).

Naives Bayes Classification can also be performance if we assume independent variables, since it considers that each of these variables contribute independently to the probability that the observation is from one or other category, regardless of any possible correlations between the features.

The method has also been applied in R by the function: "naive.model".

The accuracy in this case is only 0.73, the same performance than LDA but not better than QDA, this could be because, in fact, our features are highly correlated as we sow in the plot 9 of the correlation matrix.

The summary of the results of the models is gather in the table 5.

Table 5: Summary of Predictions efficiency by Classification methods

	Logit			Bayes		
	All predictors	Eight predictors	With interactions	LDA	QDA	Naïve
Accuracy	0.75	0.64	0.67	0.73	0.78	0.73
Error	0.25	0.36	0.32	0.27	0.22	0.27
Kappa	/	/	/	0.46	0.55	0.46

4 Machine Learning Classifications

Machine Learning uses statistical models that can be applied through computer systems for classification approaching. It uses automatic learning to make forecasting of group membership for data instances, based on past observations.

There are few different algorithms that can be developed as classification techniques, some of them are:

- K-nearest neighbor (KNN)
- Support Vector Machines (SVM)
- Decision Trees
- Random Forest
- Gradient Boosting
- Neural Network

Further information about them and its corresponding application performance details are gathered in the next sections.

4.1 KNN

In the K-nearest neighbor (KNN) technique, nearest neighbor is measured with respect to value of k, that define how many nearest neighbors need to be examine to describe class of a sample data point.

The entire data is categorized into sample and training data, distance is calculated between sample points and all training points and the point with smallest distance is known as nearest

neighbor.

One of the main advantage of KNN technique is that it is effective for large training data and robust to noisy training data.

As a first step, we have applied the algorithm through R Studio by the function 'knn' of the package 'class'. We have selected as 'k' parameter, the value 3, and we have checked the results, the confusion matrix of the performance model is detailed below.

```
Confusion Matrix and Statistics

          Reference
Prediction  Deciduous Evergreen
  Deciduous        24        13
  Evergreen         8        22

               Accuracy : 0.6866
                 95% CI : (0.5616, 0.7944)
    No Information Rate : 0.5224
    P-Value [Acc > NIR] : 0.004694

                  Kappa : 0.3761

 Mcnemar's Test P-Value : 0.382733

            Sensitivity : 0.7500
            Specificity : 0.6286
         Pos Pred Value : 0.6486
         Neg Pred Value : 0.7333
             Prevalence : 0.4776
         Detection Rate : 0.3582
   Detection Prevalence : 0.5522
      Balanced Accuracy : 0.6893

       'Positive' Class : Deciduous
```

Figure 16: Results of the KNN without tuning

As we can observed, the kappa value is 0.38, and then, we have calculated the accuracy level (0.69) and the errors (0.31).

They are lower than the ones achieved by the statistical tools, but we didn't tune the parameter k yet. Let's see how the model can be improved if we select the optimal value of k.

For doing this, we have used a train control algorithm based on the cross-validation technique which use a limited sample in order to estimate how the model is expected to perform in general when is used to make predictions on data that is not used during the training of the model.

Then, we have applied the grid search of the package 'caret' in R Studio for finding the optimal hyper-parameter, using the metric "accuracy".

We have not used a economic cost metric because a failure in our study doesn't has economic impact or some kind of loss for none company. In addition, regardless to the type prediction the difference in impact is meaningless and does not affect anyhow a potential criteria.

Due to the random initialization of the weights, the final results after applying the tuning can differ a bit, so, we have made a loop of 30 iterations and we have obtained the most repeated value of 'k' and we have calculated the result with it.

We have preferred to do that instead of fixing the data with 'set.seed', because if the first time you execute the code with 'set.seed' the algorithm don't select the best parameter, the study is not going to be as upright as possible since possibly it is going to compare not the most optimal results of each model.

The optimal value of 'k' is 7.
Then, the accuracy, kappa and error results for the tuned model are: 0.78 of accuracy, 0.56 of kappa and 0.22 of errors.

As we can see, these results are rather better than the first KNN model and equally or even a bit better than the ones got by the best classification model (QDA).

All the details about the code can be founded in the Appendix.

4.2 Support Vector Machines

Support vectors are the data points that lie closest to the decision surface. It executes the classification of data vectors by a hyper plane in immense dimensional space. Maximal margin classifier is the basic form of SVM that helps to determine the most simple classification problem of linear separable training data with binary classification.

The main advantage of SVM is its capability to deal with wide variety of classification problems includes high dimensional and not linearly separable problems, but one of its major disadvantage is that it requires the key parameters to set correctly to attain excellent classification results.

As the same way developed with KNN, we have performance a first model using the values of gamma (radial basis function as it has been the selected method for SVM) equal to 0.01 and cost equal 1. The results are gather in the table 6.

Then, we have tuned the parameters as the same way than KNN. Obtaining a gamma and cost optimal value of 0.05 and 1, respectively. The results are much better with the tuning as we could expected. The details are gathered in table 6.

4.3 Decision Trees

The decision tree is transparent mechanism which facilitate users to follow a tree structure easily in order to see how the decision is made.

The tree is terminated by leaf nodes that denote the result of the combination of decisions.

Data that is long to be classified, is placed at the root node where it is passed through the various decisions in the tree according to the values of its features. The path that the data takes funnels each record into a leaf node, which assigns it a predicted class.

As, we did before, we have applied first version of a decision tree without tuning, in order to compare later with the optimized one.

The graph of the tree is shown in the figure 17.

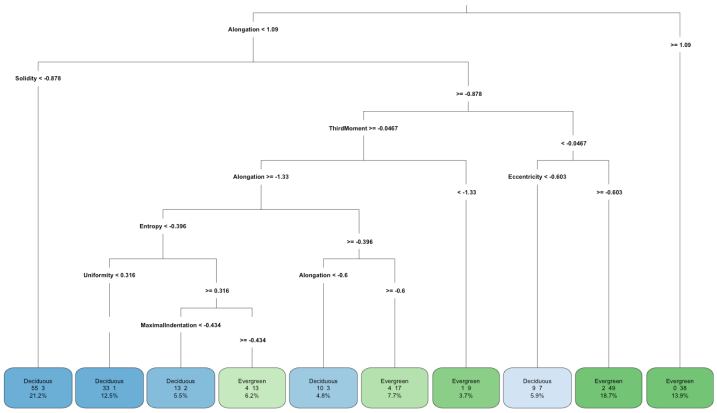

Figure 17: Decision tree without tuning

The results of the performance are detailed in the table 6.

Then, the advanced version of decision trees, C5.0 algorithm has been used for improve the results of the previous one.

A slightly better output has been achieved versus the classic decision tree. We can observed the results in the table 6.

4.4 Random Forest

Random decision forest operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees.

Random decision forests correct for decision trees' habit of over-fitting to their training set.

The first model we have built up has been with the default hyper-parameter 'mtry' which is the number of variables available for splitting at each tree node, and it is calculated as the number of predictor variables divided by 3 and rounded down. So, the chosen value has been equal to 4.

Also, the cutoff in RF controls the probability to belong a group, since there is only two categories, it should be (0.5,0.5).

The obtained results are collected in the table 6.

In order to improve the previous model, we have performed the loop with 30 iterations as with the previous cases, using the values 2, 4, 8, 10, 12 and 14 for the hyper-parameter 'mtry', and we have finally choose the optimal one which has been 14.

So, the obtained results has been rather better now. They are detailed in the table 6.

4.5 Ensemble Prediction

Sometimes when you combine the methods that have been performed better, you can get greater results making them work together.

We have resembled the algorithms of LDA, tuned KNN, tuned SVM, C5.0 and tuned RF.

In this case, we have obtained the same effect than the tuning, since the measures have been more or less the same than the best previous models.

The final figures are in the table 6.

4.6 Gradient Boosting

This machine learning technique produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds a model step by step like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

The default used model is built with Bernoulli distribution, 250 trees, shrinkage of 0.01, interaction depth of 2 and 8 number of minobsinnodoes.
We have obtain an 0.71 of accuracy.

Then, optimizing the hyper-parameters with caret and specifying a threshold of 0.5 (two balanced class) in the variable importance, we have got an accuracy of 0.77, a bit better. The rest of the measures are in the table 6.

In addition, the next plot shows the contribution weight of each variable in the classification of the model.

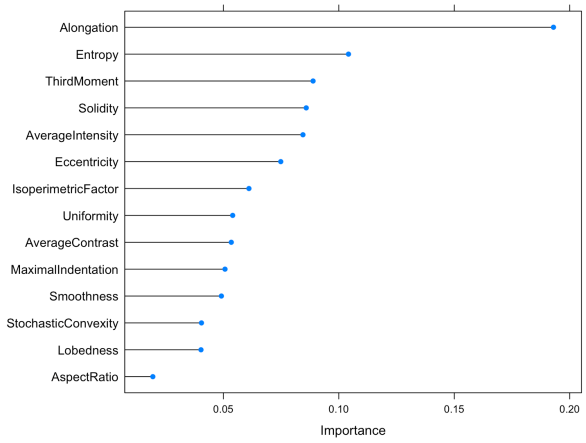

Figure 18: Importance of the variables in the classification

4.7 Neural Networks

A Neural Network (NN) is based on a collection of connected nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a

signal then processes it and can signal neurons connected to it.

We have used the average neural network function in R to compute the model, over more, we have tune the hyper-parameters selecting the optimal between some options of 'size' and 'decay'.

The figure 19 show the performance of the two different decay parameters in the model computation.

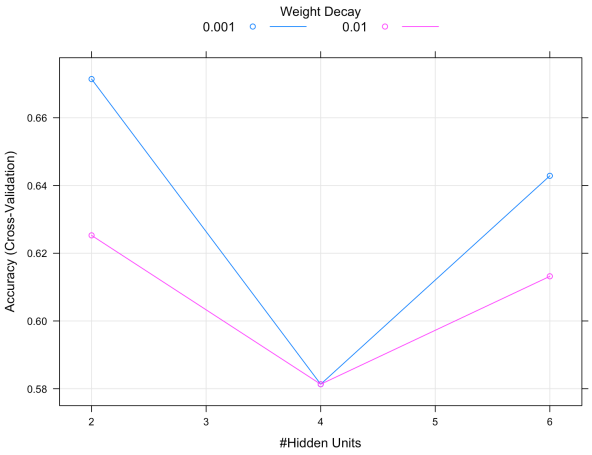

Figure 19: Decay parameter performances

Finally, in the figure 20, we can see the importance of the variables in the model performance classification. They are the same as the Gradient boosting technique.

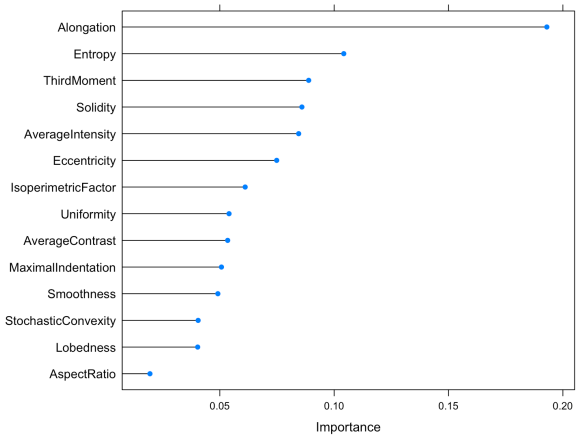

Figure 20: Importance of the variables in the classification

The achieved accuracy is the best of all methods with a value of 0.97.

The next table summarise all the results of the implemented models.

Table 6: Summary of prediction efficiency by Machine Learning methods

		KNN	SVM	DT	RF	Ensemble	GB	NN
No tun	Accuracy	0.69	0.75	0.77	0.75	/	0.71	/
	Error	0.31	0.25	0.25	0.25	/	0.29	/
	Kappa	0.38	0.49	0.55	0.51	/	0.42	/
Tuning	Accuracy	0.78	0.81	0.78	0.81	0.79	0.77	0.97
	Error	0.22	0.19	0.22	0.19	0.21	0.23	0.03
	Kappa	0.56	0.61	0.56	0.61	0.58	0.55	0.95

5 Conclusions

As we can observe in the table 5 and 6, the accuracy of the models is between 0.69-0.78 with default parameters or non-tuning parameter, and the range error values is 0.25-0.36. On the contrary, the accuracy range when using tuned parameters is 0.78-0.81 and 0.19-0.23 of errors. So, we can conclude, as expected, using the optimal parameters improve a lot the performance of the models.

As a summary about the different techniques, it can been said that in statistical classification, although the Logistic Regression model usually works better for classification of two variables than the Bayers Classifiers, in this case, the results for the Logistic one have been worse than the QDA (0.78 of accuracy for QDA in comparison with 0.75 of the Logistic), this can be explained because QDA is more flexible and works with quadratic boundaries.

Regarding the number of variables, it works better with all of them, and also, without interactions.

Regarding to machine learning techniques, the best one is the Neural Network obtaining an accuracy almost of 1 (a perfect model).

This machine learning techniques works much better when you tune the hyper-parameters and in these cases, they are better than statistical classification methods.

The ensemble configuration didn't work better than some individual techniques at all, and the Support Machine Vectors and Random Forest process work quite well comparing with the rest.

Some limitations has been found like the lack of normality for the predictors distribution, equal variances, well-separated groups, etc, however, only a 3% of errors has been obtained in the better model which can be considered quite good enough for develop accurate predictions.

References

[1] Master's Thesis 'Development of a System for Automatic Plant Species Recognition' by Pedro Filipe Barros Silva (2013). *https* : *//hdl.handle.net*/10216/67734