# Práctica 8

# Aprendizaje Automático

Javier Pellejero Ortega & Zhaoyan Ni

Inteligencia Artificial

Grupo 11

Doble grado Matemáticas e ingeniería informática

1. Agrupamiento.

Para el agrupamiento hemos optado por una base de datos de **semillas** que podemos encontrar en el link [**http://archive.ics.uci.edu/ml/datasets/seeds**](http://archive.ics.uci.edu/ml/datasets/seeds), y mide una serie de atributos de hasta 3 tipos distintos de semillas de trigo (*Kama*, *Rosa* y *Canadian*) que comentaremos a continuación. En total hay datos de 70 de cada uno de los subtipos de semilla.

|  |  |  |
| --- | --- | --- |
| Nombre del Atributo | Descripción | Tipo |
| **area** | área de la semilla | Real |
| **perimeter** | perímetro de la semilla | Real |
| **compactness** | compacidad que viene dada por la fórmula C=4·pi·Area/perimeter2 | Real |
| **length** | longitud de la semilla | Real |
| **width** | ancho de la semilla | Real |
| **assymmetryCoeffi** | coeficiente de asimetría | Real |
| **lengthGrove** | longitud de la curvatura central de la semilla | Real |
| **varieties** | variedad de la semilla: 1-Kama, 2-Rosa y 3-Canadian | Categórico |

Comentemos ahora la preparación que hemos realizado sobre los datos. Para empezar, hemos optado por una normalización de los datos. Esto es debido a que vamos a utilizar el **algoritmo jerárquico** que maneja distancias y es preferible que todo esté en un mismo rango para que unos atributos no tengan más peso que otros en dicha distancia.

En cuanto al número de *clusters* hemos decidido que sean 3. La razón es simple: tenemos 3 clases de semilla y queremos ver cuánto se ajustan a los *clusters*. Además, hemos realizado el agrupamiento con los enlaces *centroide* y *completo* pues tienden a generar *clusters* más homogéneos que es lo que nos interesa. Los resultados que mostramos son los del enlace *centroide* que es el que ha mostrado mejores resultados.

A continuación, mostramos la información que resulta de dicho agrupamiento con el mencionado enlace y su relación con el atributo que nos señala la variedad de la semilla

=== Model and evaluation on training set ===

Clustered Instances

0 67 (32%)

1 76 (36%)

2 67 (32%)

Class attribute: varieties

Classes to Clusters:

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster 0 | Cluster 1 | Cluster 2 |  |
| 60 | 8 | 2 | Semilla 1 |
| 2 | 68 | 0 | Semilla 2 |
| 5 | 0 | 65 | Semilla 3 |

Cluster 0 <-- 1

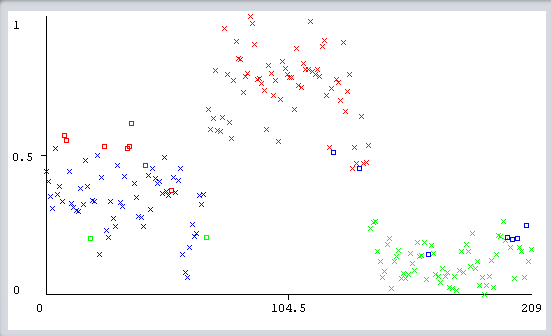
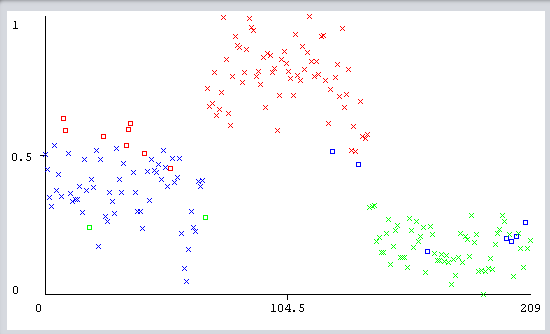
Cluster 1 <-- 2

Cluster 2 <-- 3

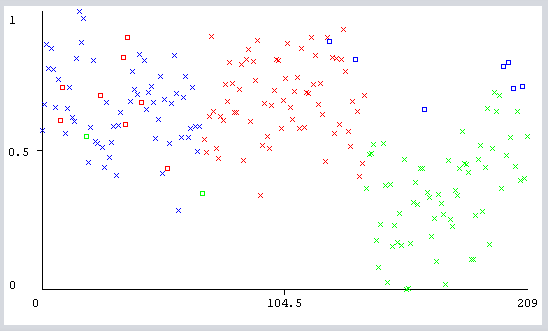
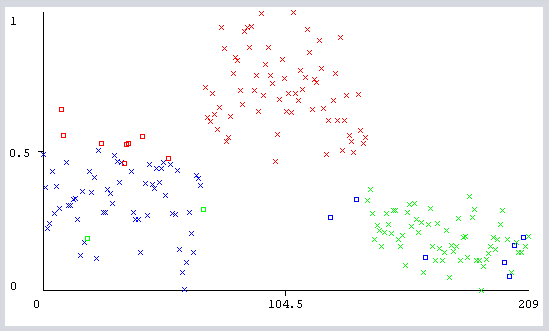
Incorrectly clustered instances: 17.0 8.0952 %

Podemos observar un buen ajuste entre *clusters* y tipos de semillas. Tan sólo 17 de las 210 semillas no se ajustan a su *cluster*. Como observamos, el peso del *cluster* 0 es de la semilla 1 (*kama*), el *cluster* 1 está conformado mayoritariamente por el tipo 2 (*rosa*) y el último por el tipo 3 (*Canadian*).

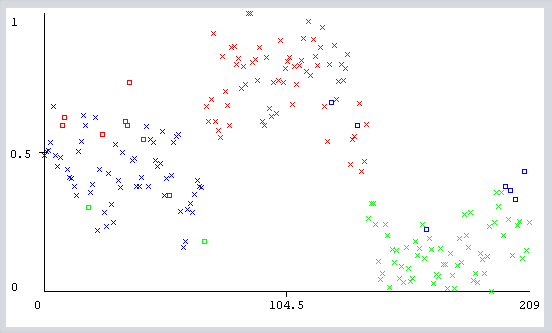
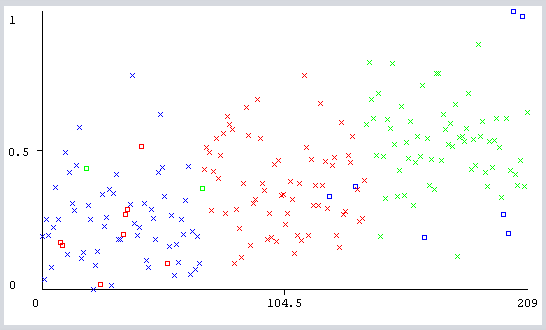
Debido a este buen ajuste, puesto que teníamos mismo número de semillas de cada tipo, los *clusters* quedan de tamaños homogéneos, rondando el correspondiente 33%. Por último, presentamos unas gráficas que representan en el eje de abscisas las instancias, y en el eje de ordenadas los valores de los diversos atributos. Además, son representados por colores los distintos *clusters*.

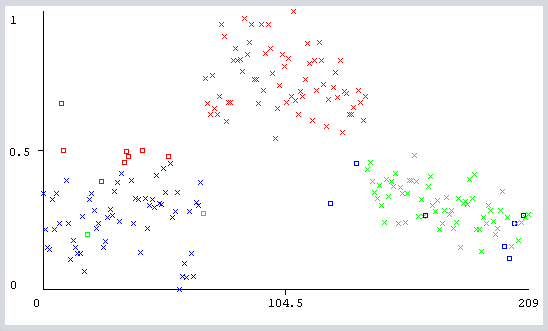
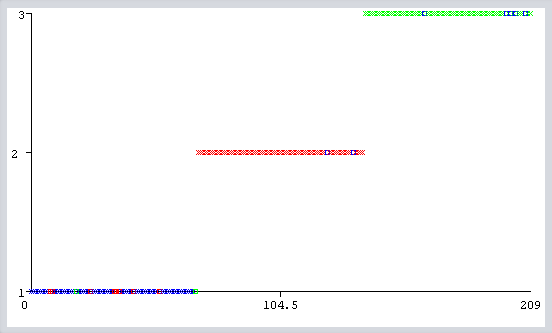
Área Perímetro

Compacidad Longitud de la semilla

Anchura Coeficiente de Asimetría

Longitud curvatura central Tipo de semilla

Como vemos, hay una homogeneidad indiscutible en todas las gráficas, tan sólo encontrando cierto ruido en zonas donde los atributos entre distintos tipos de semillas son similares.

Cabe destacar que, fijándonos únicamente en los atributos de las semillas, las de tipo 2 (*rosa*, que han ido a parar al *cluster* 1, en rojo), tiene los valores más diferenciados con respecto a las otras dos: mayor área, perímetro, longitud, anchura, etc. No es así con las de tipo 1 y 3 que tienen atributos similares tales como el área, perímetro o longitud y que sin embargo el algoritmo ha conseguido encontrar las pequeñas diferencias entre ambos tipos como la anchura o la compacidad. Hay que recalcar que el atributo referido a la variedad **no ha sido usado** para calcular las diversas medidas del algoritmo.

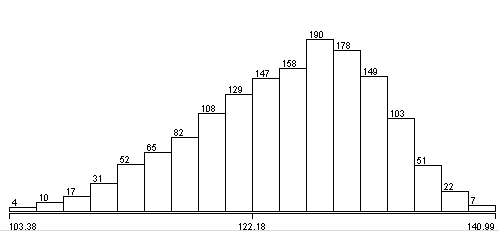
1. Agrupamiento
2. Regresión.

Para la regresión hemos escogido una base de datos de un estudio realizado sobre **perfiles alares**, lo que la aeronáutica denomina a la forma del área transversal de un elemento, que al desplazarse a través del aire es capaz de crear a su alrededor una distribución de presiones que genere sustentación, en túneles de viento. Podemos encontrar esta base de datos en el link <https://archive.ics.uci.edu/ml/datasets/Airfoil+Self-Noise>. Tenemos un total de 1503 instancias y todos los atributos, hasta un total de 6, son valores reales.

|  |  |  |
| --- | --- | --- |
| Nombre del Atributo | Descripción | Tipo |
| **frequency** | Frecuencia de vibración, medida en hercios | Real |
| **angle** | Ángulo de ataque, medido en grados | Real |
| **length** | Longitud de cuerda, en metros | Real |
| **velocity** | Velocidad de flujo libre, en metros segundo | Real |
| **suction** | Espesor de desplazamiento, en metros | Real |
| **sound** | Nivel de presión sonora a escala, en decibelios | Real |

Para entender mejor qué son cada uno de estos parámetros se puede visitar <https://es.wikipedia.org/wiki/Perfil_alar> o <https://en.wikipedia.org/wiki/Airfoil>.

Por último, comentamos que la variable de salida es la última explicada: *sound*. A continuación, presentamos el histograma de dicha variable.



Vayamos ahora con la ejecución de los algoritmos. No hemos normalizado ni estandarizado los atributos, previa experimentación con *perceptrón multicapa* arrojando similares resultados, así hemos optado por ésta como una opción por defecto.

Para el *perceptrón multicapa* hemos cogido una *p* de valor 15. Con esta asignación de partes en las que se realizará la validación cruzada del algoritmo hemos conseguido un error algo menor que con las cifras cercanas. En concreto hemos probado con todos los valores entre 10 y 20.

En cuanto al *K-NN*, hemos optado de nuevo por un valor de 15 para la *k* pues era el que menor valor arrojaba. No hemos precisado probar con distintos subconjuntos pues esta ya arrojaba un error bastante inferior que el *perceptrón multicapa*.

=== Cross-validation ===

=== Summary ===

Correlation coefficient 0.7912

Mean absolute error 3.3103

Root mean squared error 4.2755

Relative absolute error 58.6891 %

Root relative squared error 61.9336 %

Total Number of Instances 1503

Los datos anteriores son los arrojados por *Weka* al aplicar *perceptrón multicapa* con los parámetros antes dichos.

=== Cross-validation ===

=== Summary ===

Correlation coefficient 0.9309

Mean absolute error 2.0495

Root mean squared error 2.6084

Relative absolute error 36.3373 %

Root relative squared error 37.784 %

Total Number of Instances 1503

Mientras que estos últimos datos son los correspondientes a *K-NN*.

Podemos observar unos errores altos, sobre todo en el primer algoritmo y una mejora considerable con el *K-NN* en ambos parámetros, tanto la raíz cuadrada del error cuadrático, como el error absoluto. Tanto en valores medios como absolutos. Pese al error de ambos algoritmos, es muy destacable la mejora del *K-NN* reduciendo el error a prácticamente la mitad.