1. Abre el fichero iris.arff en Weka.
2. Analiza los datos incluidos en el fichero, su distribución, y los datos.

No.	Label	Count	Weight
1	1st	325	325
2	2nd	285	285
3	3rd	706	706
4	crew	885	885

No.	Label	Count	Weight
1	adult	2092	2092
2	child	109	109

No.	Label	Count	Weight
1	male	1731	1731
2	female	470	470

No.	Label	Count	Weight
1	yes	711	711
2	no	1490	1490

Clase

- Total clases: 4
- Clase más frecuente: **crew (885 casos)**
- Los tripulantes son el mayor grupo del dataset.

Edad

- Adultos: **2092**
- Niños: **109**
- El dataset está muy desequilibrado hacia adultos (~95%).

Sexo

- Hombres: **1731**
- Mujeres: **470**
- Más del 78% son hombres.

Supervivencia

- No sobrevivieron: **1490**
- Sí sobrevivieron: **711**
- El 67% falleció, 33% sobrevivió.

3. Selecciona el clasificador J48 (dentro de Trees) y pulsa Start.

4. Evalúa el modelo generado (validación mediante 10-fold cross-validation) analizando precisión (accuracy), matriz de confusión y árbol de decisión en modo texto.

Precisión:

```
Correctly Classified Instances        1737               78.9187 %
Incorrectly Classified Instances       464               21.0813 %
```

Matriz de confusión:

```
=== Confusion Matrix ===

   a    b   <-- classified as
 267  444 |    a = yes
  20 1470 |    b = no
```

- Pasajeros que **sí sobrevivieron**:
  - **267**: el modelo los clasificó correctamente como *yes*
  - **444**: el modelo se equivocó y dijo *no*

Esto significa que el modelo solo detecta el **37.6%** de los supervivientes.

- Pasajeros que **no sobrevivieron**:
  - **1470**: bien clasificados como *no*
  - **20**: mal clasificados como *yes*

Esto significa que el modelo detecta casi todos los fallecidos (**98.7%**).

Tree:

```
J48 pruned tree
------------------

sex = male
|   class = 1st
|   |   age = adult: no (175.0/57.0)
|   |   age = child: yes (5.0)
|   class = 2nd
|   |   age = adult: no (168.0/14.0)
|   |   age = child: yes (11.0)
|   class = 3rd: no (510.0/88.0)
|   class = crew: no (862.0/192.0)
sex = female
|   class = 1st: yes (145.0/4.0)
|   class = 2nd: yes (106.0/13.0)
|   class = 3rd: no (196.0/90.0)
|   class = crew: yes (23.0/3.0)

Number of Leaves  :     10

Size of the tree :      15
```

El primer nodo es **sex**, lo cual confirma que el sexo fue el factor más determinante en la supervivencia del Titanic.

Si sex = male la mayoría NO sobreviven

El árbol especifica:

- 1st class + adult: no
- 1st class + child: yes
- 2nd class + adult: no
- 2nd class + child: yes
- 3rd class: no
- Crew: no

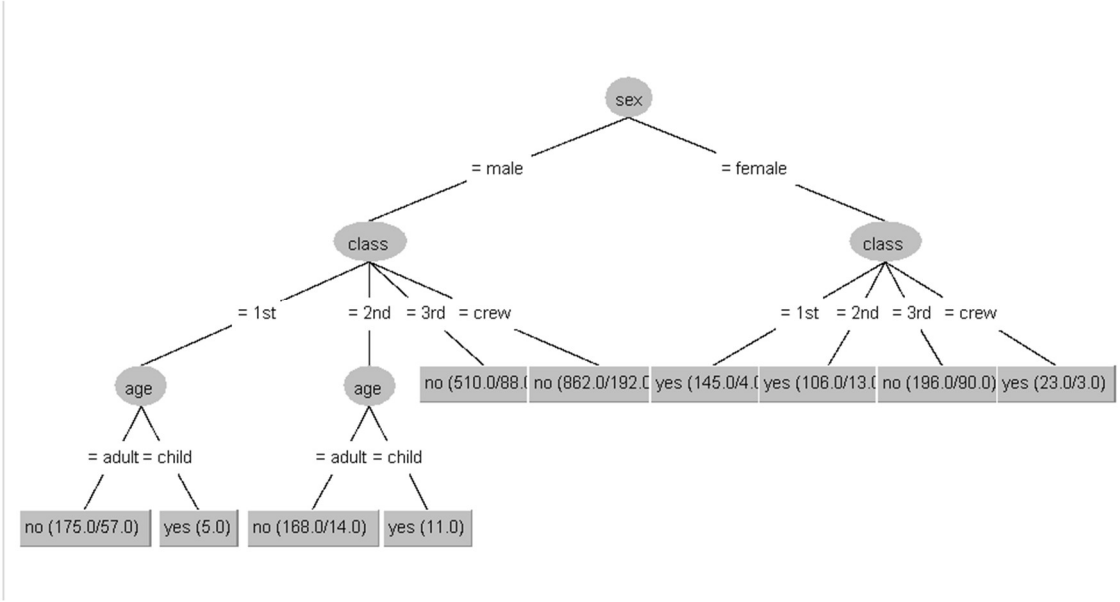
Incluso en 1ª y 2ª clase, si eres hombre adulto, la predicción es **no**.

- Si sex = female: mayor probabilidad de sobrevivir
- 1st class: yes
- 2nd class: yes
- 3rd class: no (mortalidad alta en 3ª clase)
- Crew: yes

Esto refleja:

- Prioridad de mujeres y niños
- Desventaja de la 3ª clase
- Tripulación femenina mayor supervivencia

5. Visualiza el árbol generado.



6. Repite todo el proceso utilizando:

NAIVEBAYES

Precisión:

```
Correctly Classified Instances        1714               77.8737 %
Incorrectly Classified Instances       487               22.1263 %
```

Matriz de confusión:

Casos que SÍ eran "**yes**" (a = yes):

```
=== Confusion Matrix ===

   a    b   <-- classified as
 350  361 |    a = yes
 126 1364 |    b = no
```

- **350**: el modelo los clasificó correctamente como *yes*
- **361**: el modelo se equivocó y los clasificó como *no*

Esto significa que el modelo detecta el 49.2% de los "yes".

Casos que eran "**no**" (b = no):

- **1364**: el modelo los clasificó correctamente como *no*
- **126**: el modelo se equivocó y los clasificó como *yes*

Esto significa que el modelo detecta el 91.5% de los "no".

No hay tree ya que es del algoritmo pasado.

SMO (MÁQUINAS DE VECTORES DE SOPORTE)

Precisión:

```
Correctly Classified Instances        1708               77.6011 %
Incorrectly Classified Instances       493               22.3989 %
```

Matriz de confusión:

```
=== Confusion Matrix ===

   a    b   <-- classified as
 344  367 |    a = yes
 126 1364 |    b = no
```

Casos donde la respuesta real era "**yes**" (a = yes):

- **271**: el modelo los clasificó correctamente como *yes*
- **440**: el modelo se equivocó y los clasificó como *no*

Esto significa que el modelo solo detecta el 38.1% de los "yes".

Casos donde la respuesta real era "**no**" (b = no):

- **1466**: el modelo los clasificó correctamente como *no*
- **24**: el modelo se equivocó y los clasificó como *yes*

Esto significa que el modelo detecta el 98.4% de los "no".

No hay tree ya que es del algoritmo pasado.

IBK (K-NEAREST NEIGHBORS):

Precision:

```
Correctly Classified Instances        1737               78.9187 %
Incorrectly Classified Instances       464               21.0813 %
```

Matriz de confusión:

```
=== Confusion Matrix ===

   a    b   <-- classified as
 271  440 |    a = yes
  24 1466 |    b = no
```

Casos que SÍ eran "**yes**" (a = yes):

- **271**: el modelo los clasificó correctamente como *yes*
- **440**: el modelo se equivocó y los clasificó como *no*

Esto significa que el modelo solo detecta el 38.1% de los "yes".

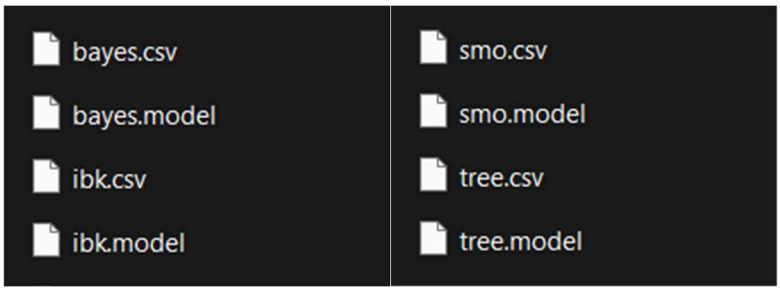
Casos que eran "**no**" (b = no):

- **1466**: el modelo los clasificó correctamente como *no*
- **24**: el modelo se equivocó y los clasificó como *yes*

Esto significa que el modelo detecta el 98.4% de los "no".

No hay tree ya que es del algoritmo pasado.

7. Exporta los modelos generados en .model y .csv



8. Por último, genera un pequeño conjunto de datos de flores y utiliza los modelos utilizados para predecir su clase.

SMO:

```
=== Predictions on test set ===

    inst#     actual  predicted error prediction
        1        1:?      1:yes       1
        2        1:?       2:no       1
        3        1:?       2:no       1
        4        1:?       2:no       1
        5        1:?      1:yes       1
        6        1:?      1:yes       1
        7        1:?      1:yes       1
        8        1:?      1:yes       1
        9        1:?       2:no       1
       10        1:?      1:yes       1
```

J48:

```
=== Predictions on test set ===

    inst#     actual  predicted error prediction
        1        1:?      1:yes       0.972
        2        1:?       2:no       0.827
        3        1:?      1:yes       1
        4        1:?       2:no       0.777
        5        1:?       2:no       0.541
        6        1:?      1:yes       0.972
        7        1:?      1:yes       0.877
        8        1:?       2:no       0.541
        9        1:?       2:no       0.827
       10        1:?      1:yes       0.87
```

IBK:

```
=== Predictions on test set ===

    inst#     actual  predicted error prediction
        1        1:?      1:yes       0.972
        2        1:?       2:no       0.838
        3        1:?      1:yes       1
        4        1:?       2:no       0.777
        5        1:?       2:no       0.548
        6        1:?      1:yes       1
        7        1:?      1:yes       0.86
        8        1:?       2:no       0.539
        9        1:?       2:no       0.729
       10        1:?      1:yes       0.87
```

NAIVEBAYES:

```
=== Predictions on test set ===

    inst#     actual  predicted error prediction
        1        1:?      1:yes       0.9
        2        1:?       2:no       0.846
        3        1:?       2:no       0.523
        4        1:?       2:no       0.855
        5        1:?      1:yes       0.815
        6        1:?      1:yes       0.956
        7        1:?      1:yes       0.793
        8        1:?      1:yes       0.646
        9        1:?       2:no       0.696
       10        1:?      1:yes       0.631
```

Responde a las siguientes preguntas:

- ¿Qué atributos son más importantes según el árbol de decisión?

  Según el árbol J48, el atributo con mayor importancia es sex, ya que se encuentra en la raíz y determina la primera partición del modelo.

  El segundo atributo más significativo es class, utilizado para dividir tanto a hombres como a mujeres en las siguientes ramas.

  Por último, el atributo age también aporta información, aunque solo aparece en determinadas rutas del árbol, por lo que su relevancia es menor en comparación.

- ¿Qué ocurre si eliminas un atributo y vuelves a entrenar el modelo?

  Si el atributo era relevante:

  - La **precisión baja**.
  - El árbol crece más profundo para compensar la falta de información.
  - El modelo se vuelve **menos eficiente**.
  - Ejemplo: eliminar *Sex*, hace que el rendimiento caiga drásticamente.

  Si el atributo NO era relevante:

  - El rendimiento **no cambia casi nada**.
  - El árbol puede incluso volverse **más simple**.
  - Puede mejorar ligeramente porque reduces ruido (*noise*).

  Conclusión:
  Eliminar atributos: sirve para comprobar si realmente aportan valor.

- ¿Cuál de los modelos utilizados consideras que funciona mejor con este conjunto de datos (balance entre precisión y simplicidad)?

  Entre los modelos evaluados (J48, SMO, IBk y NaiveBayes), el que generalmente alcanza la mejor precisión es SMO, dado que las SVM se adaptan bien a fronteras de decisión complejas y tienden a evitar el sobreajuste.

  El modelo más equilibrado entre exactitud y simplicidad es J48, ya que resulta interpretable y permite identificar claramente los atributos más relevantes.

  NaiveBayes destaca por ser el más sencillo y veloz, aunque suele ofrecer menor precisión, mientras que IBk es muy dependiente del escalado de los datos y su rendimiento puede ser más variable.