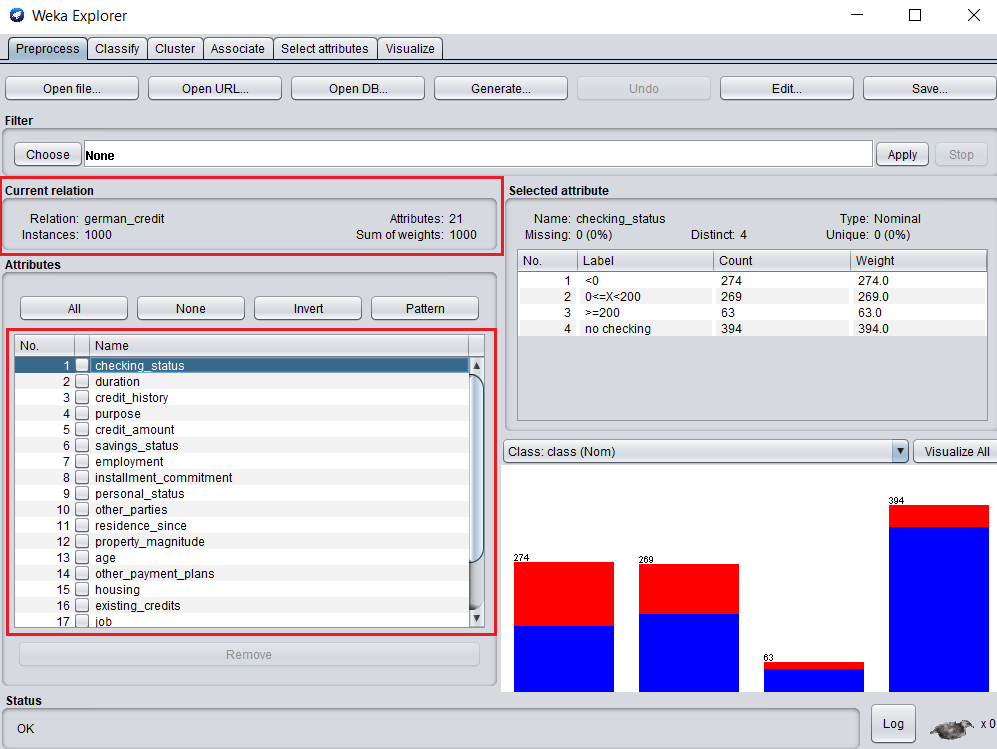
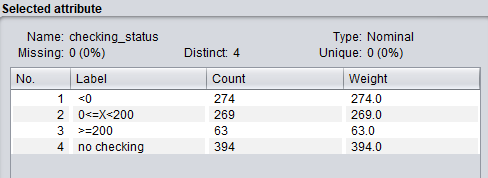
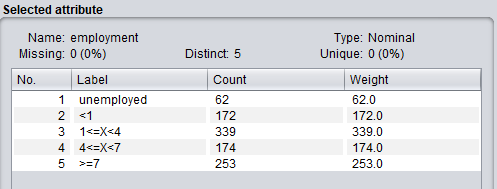
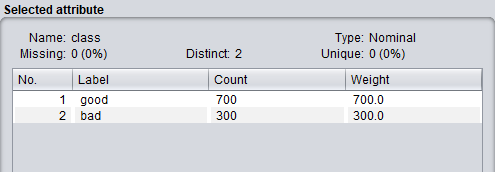
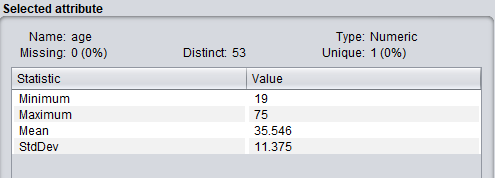
**TÉCNICAS SUPERVISADAS EN WEKA**

Inicialmente se carga el dataset y se miran sus características principales:



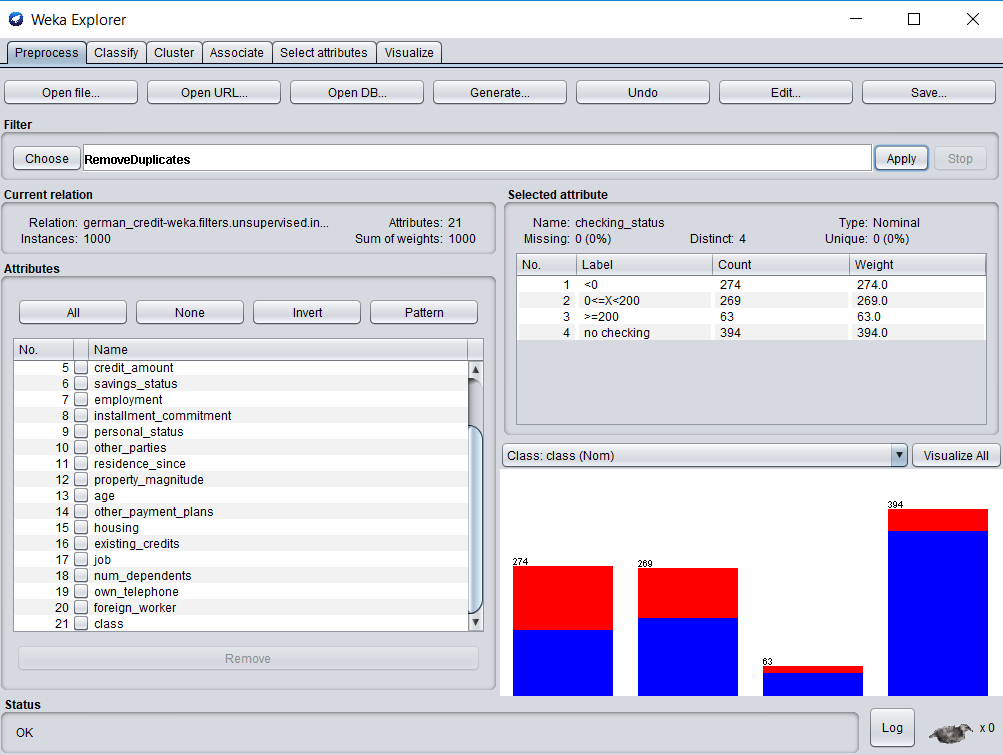
En general se observa que el dataset tiene 1000 instancias con 21 atributos cada una. Y además que ninguna de los atributos contiene valores faltantes. A continuación, se muestran algunos ejemplos.

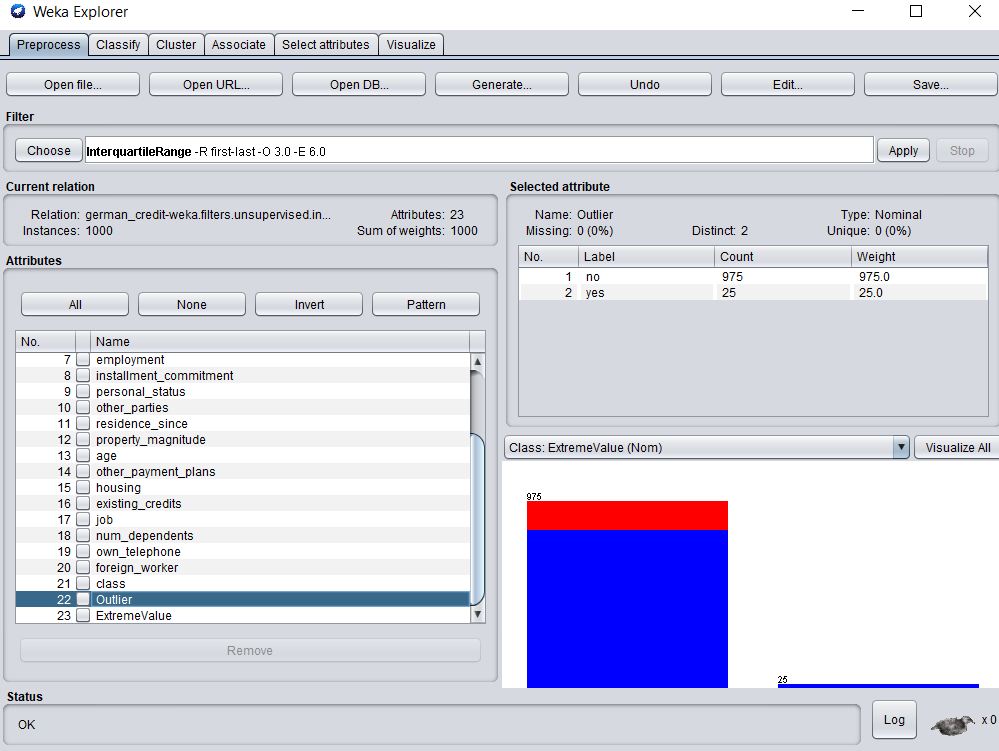
1. **Preprocesamiento**
   1. **Manejo de datos duplicados**

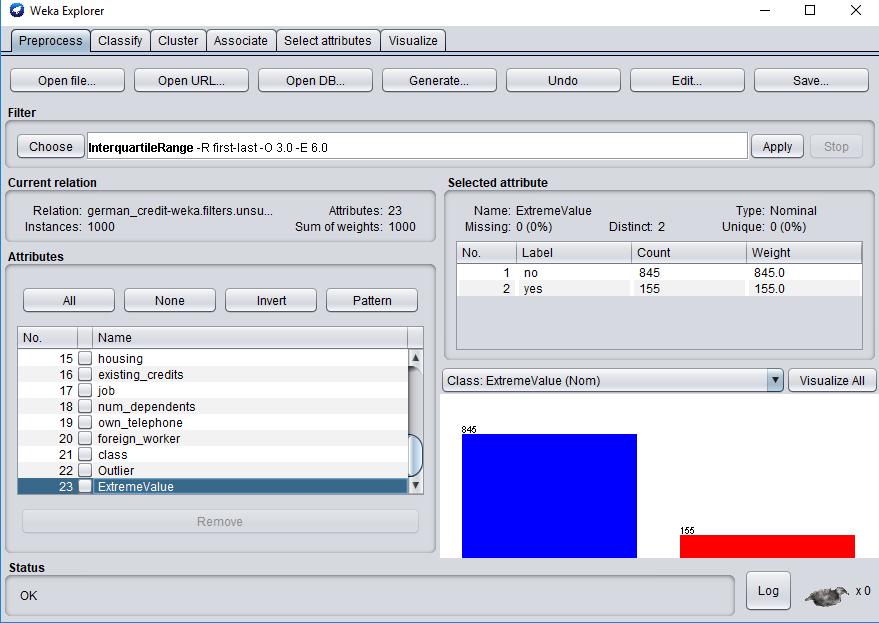
Se aplica el filtro RemoveDuplicates para remover las instancias que están duplicadas:

-

Se observa que el número de instancias se mantuvo igual, debido a que no existen instancias duplicadas.

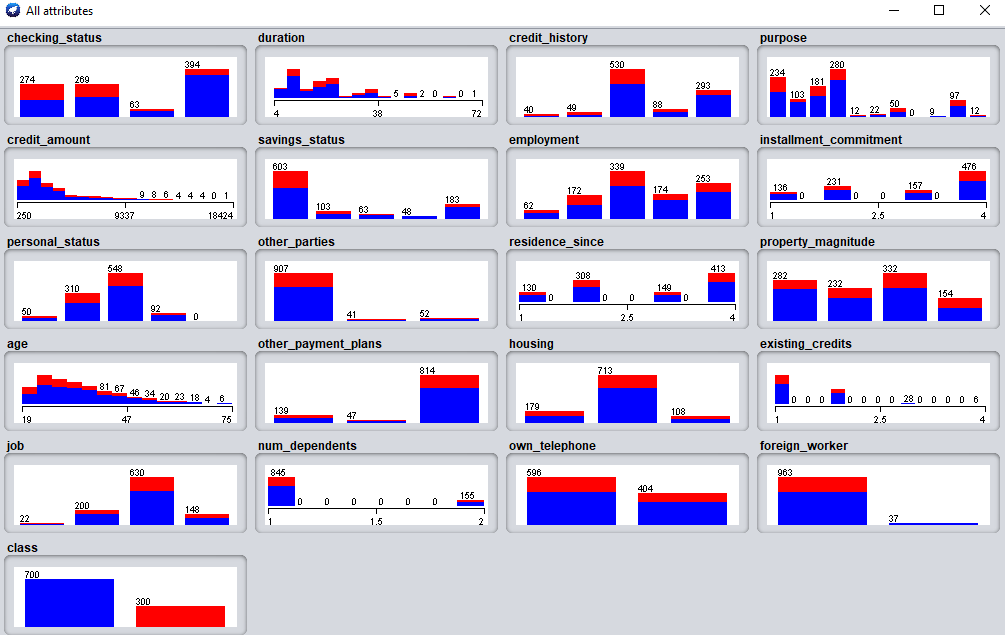
* 1. **Manejo de Outliers y valores extremos:**





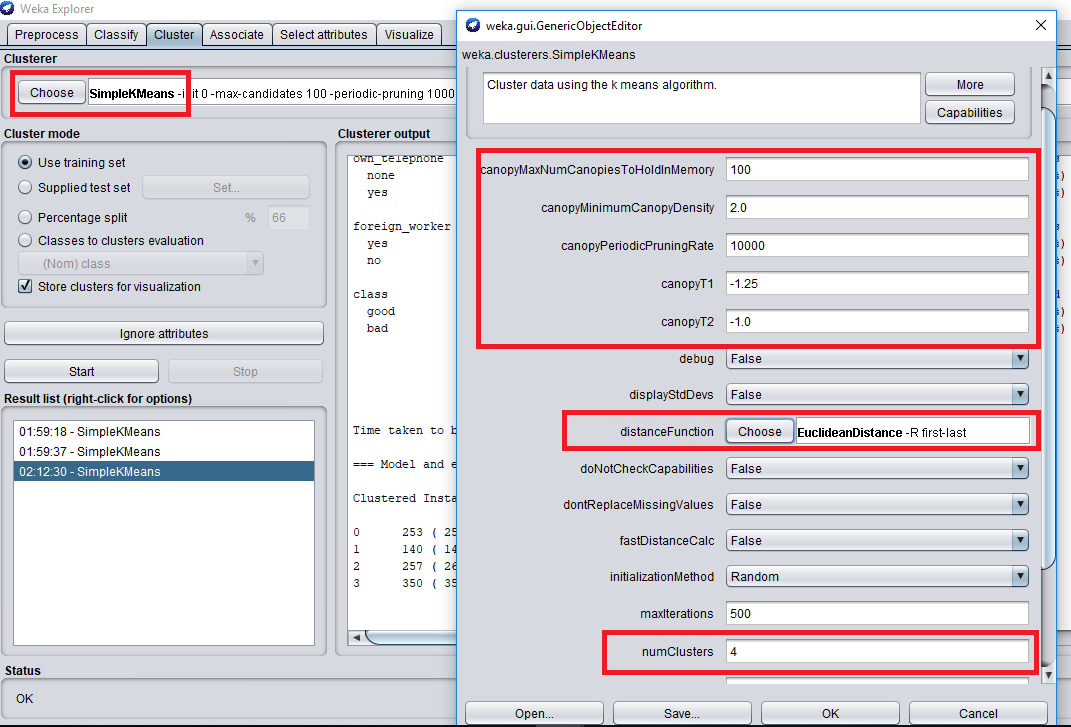
Se usa el filtro Interquartilerange para comprobar la cantidad de valores extremo y outliers, sin embargo, no se eliminan.

1. **Visualización de los datos**

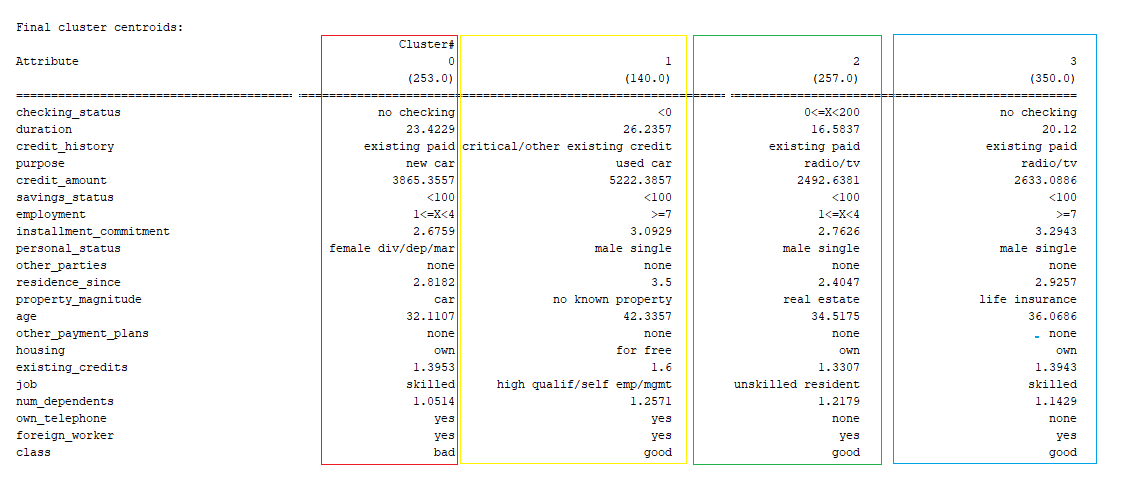
****

****

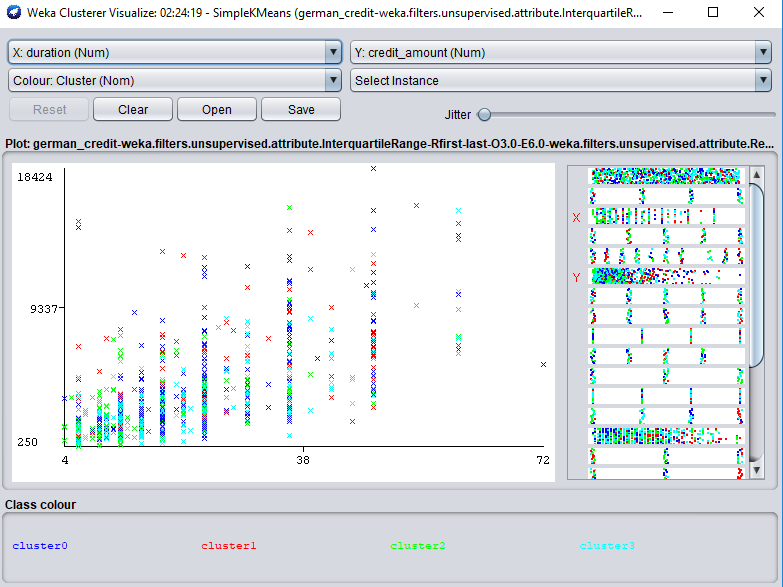
1. **Aplicación de los algoritmos de clustering**
   1. **Simple Kmeans**

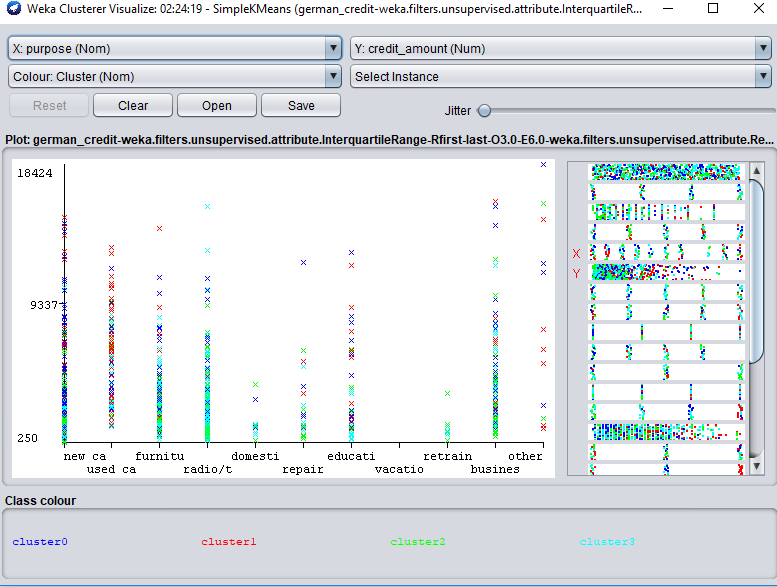
****

Se especifica el número de clusters en 4 (teniendo en cuenta el procedimiento realizado en Python), la distancia escogida es la Euclidiana y se configura el preprocesamiento para acelerar el clustering, eliminando registros que son muy parecidos (distancia menor a T2). Los resultados obtenidos fueron los siguientes:

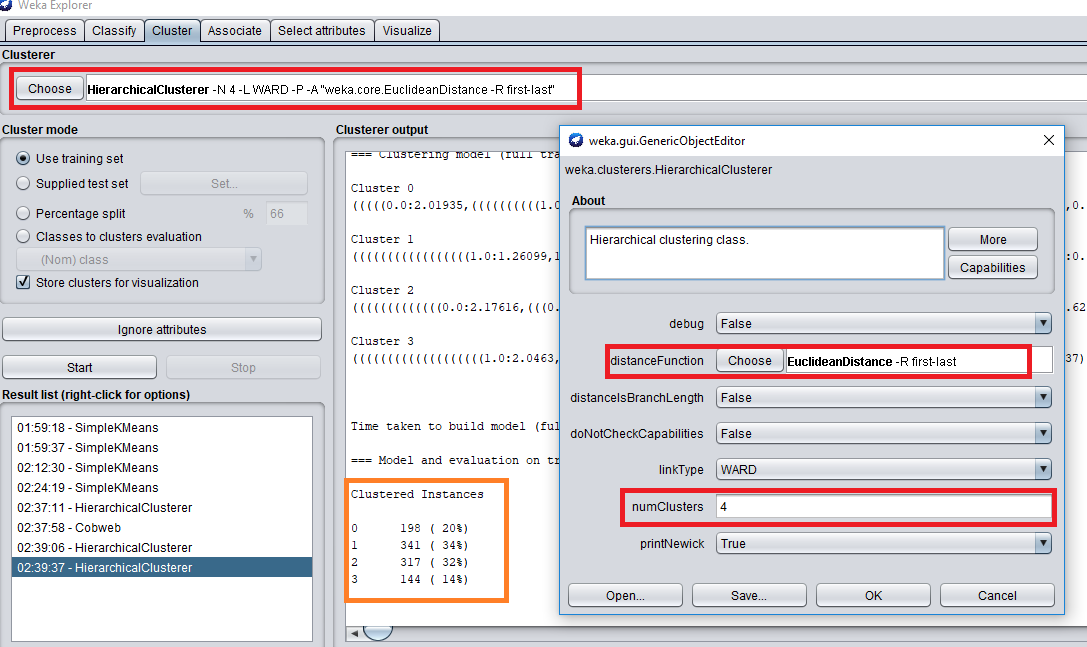


A continuación, se muestran algunas gráficas con base en los resultados obtenidos:

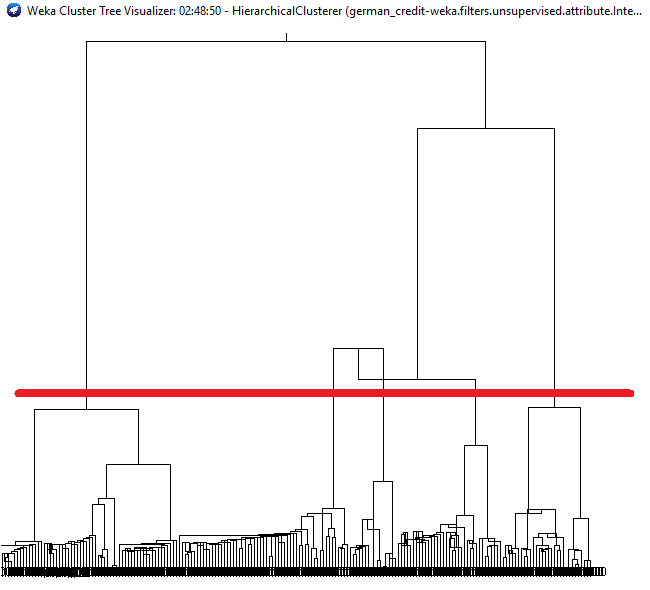


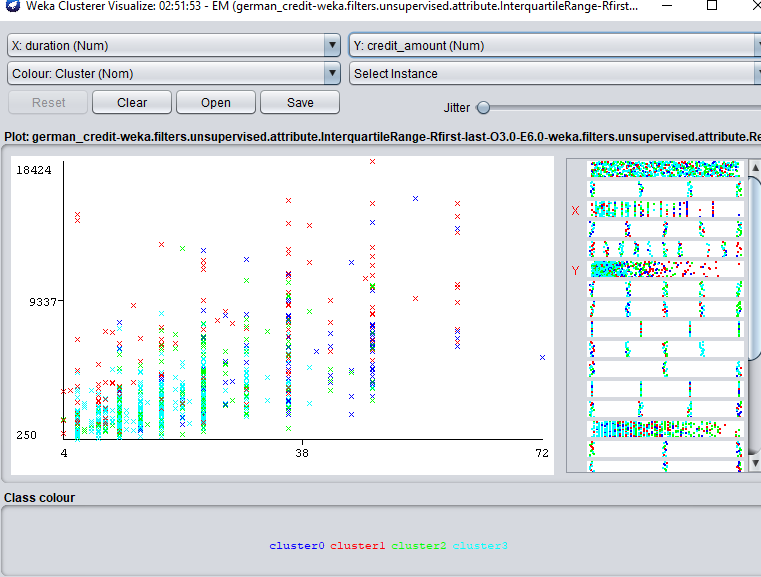


* 1. **Método Jerárquico**

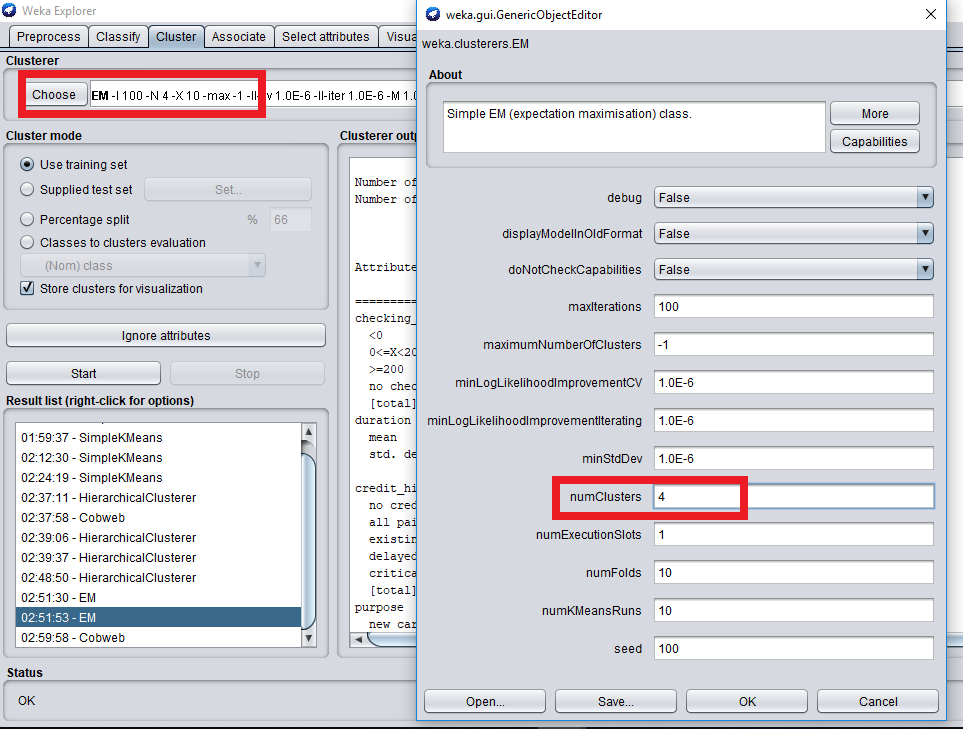
****

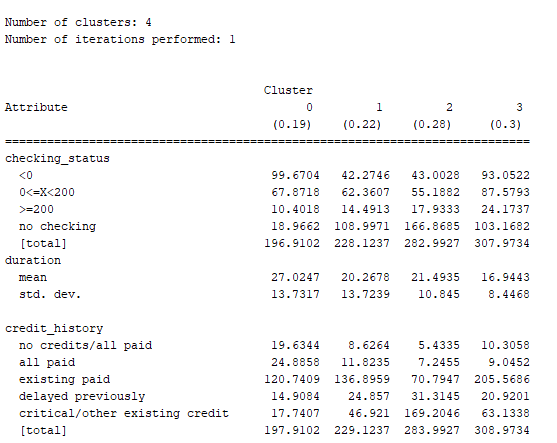
Se configuró el método con un número de clusters igual a 4, la distancia elegida fue la euclidiana y además el tipo de enlace escogido fue Ward. El dendograma resultante se muestra a continuación:

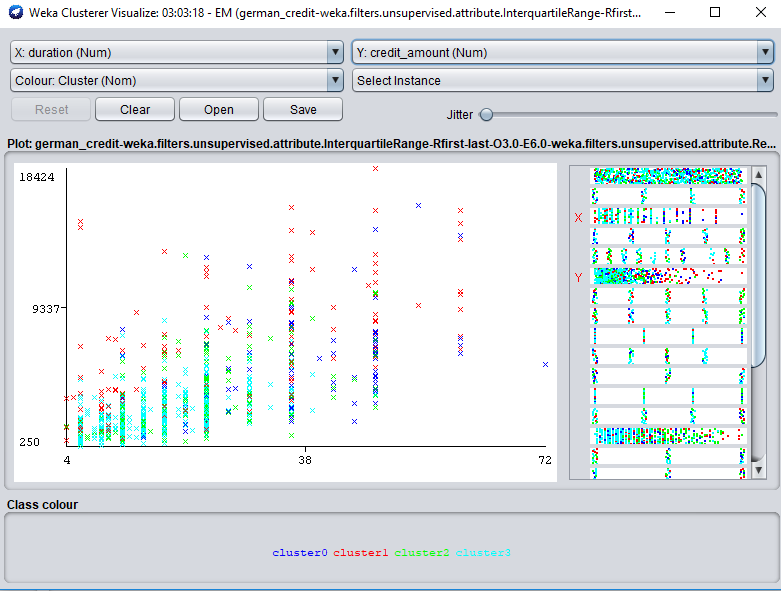




* 1. **Método Probabilístico**

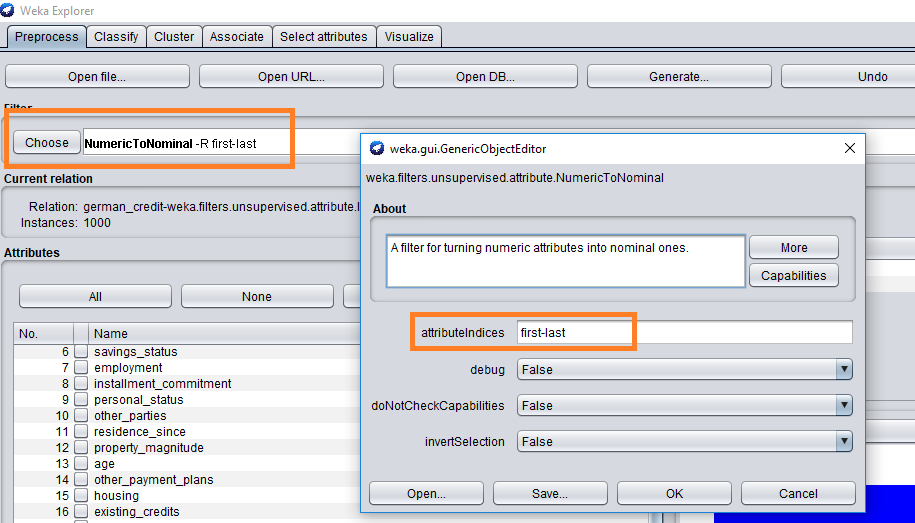
****

****

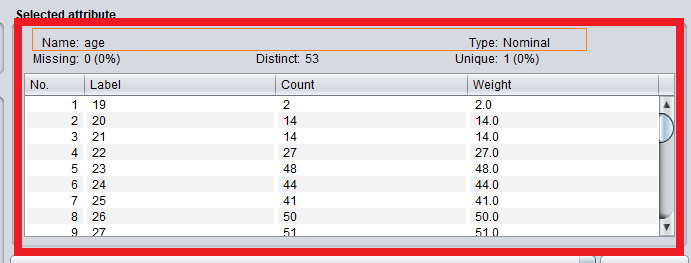
****

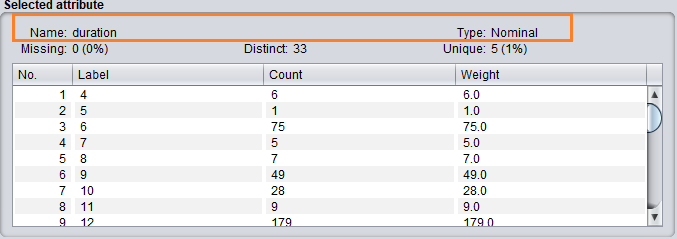
* 1. **Apriori**

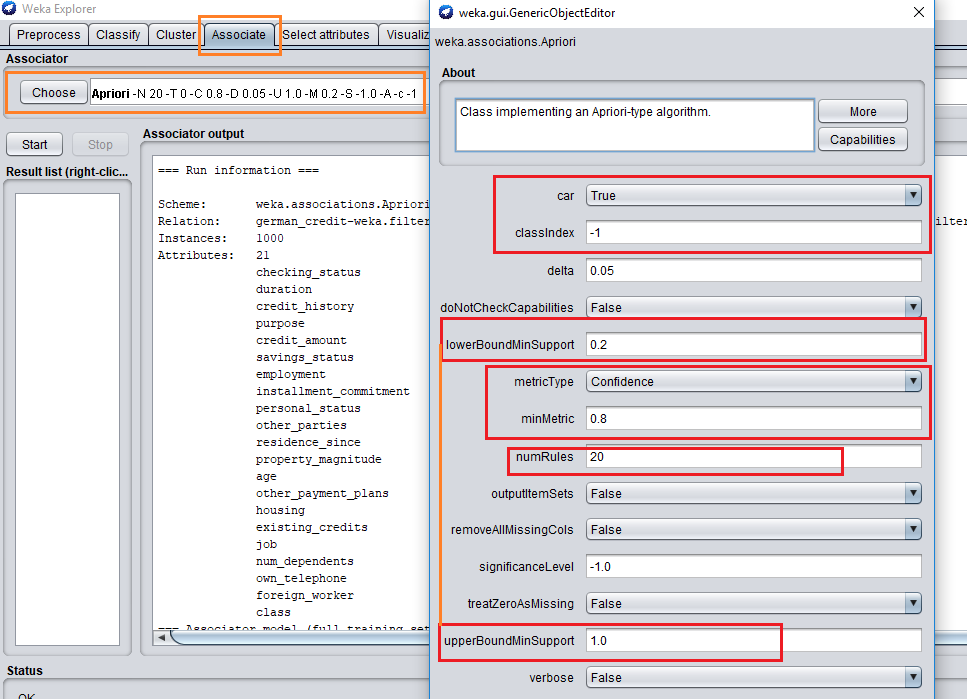
Antes de aplicar el método, fue necesario realizar una modificación a los datos y convertirlos todos en tipo categórico ya que este modelo solo funciona con este tipo de datos. Para lograr esto se utilizó el filtro NumericToNominal:



Una vez aplicado el filtro todos los atributos del dataset quedaron de tipo nominal y se procedió a aplicar el método. A modo de ejemplo se muestra que las variables edad y duración que en principio eran numéricas, fueron convertidas a tipo nominal.

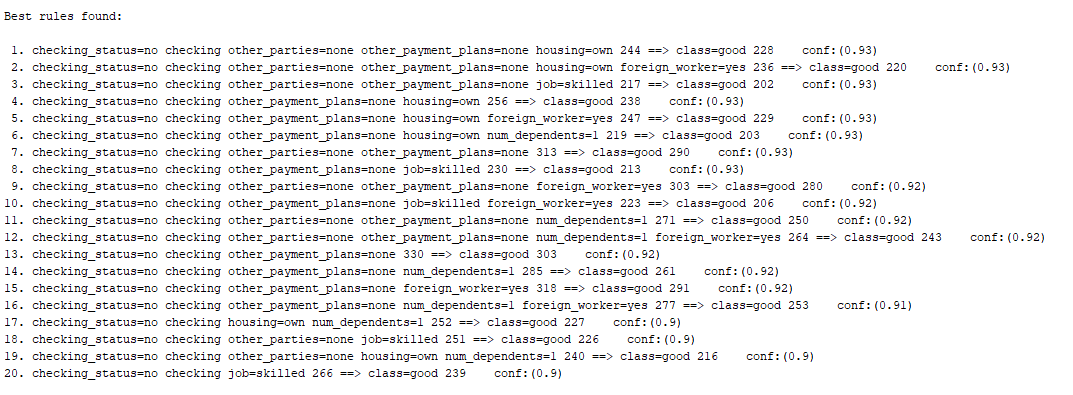




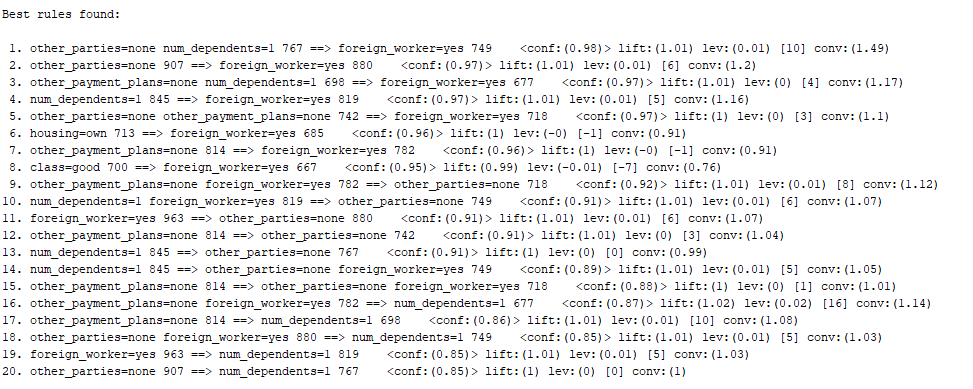


* Se puso la opción car en true, debido a que se tiene una variable de clase. (el index se configuró en -1 ya que la clase es el último atributo)
* El support se configuró entre 0.2 y 1
* La métrica utilizada fue el confidence y su valor mínimo se estableció en 0.8
* Se configuró el número de reglas en 20

Resultados con el car configurado en true (orientado a la clase):



Resultados con el car configurado en false (no orientado a la clase):



1. **Conclusiones**

* En general, los resultados en WEKA son difíciles de interpretar en la medida en que se carece de métricas que permitan comparar los distintos métodos que se usan y además la herramienta de visualización es poco flexible.
* El método que resultó más sencillo para su interpretación es el de apriori ya que los resultados están muy bien especificados.
* Una vez más se comprueba que Python ofrece mucha más flexibilidad en cuanto a la manipulación de los datos y la evaluación de los métodos, sin embargo, para este tipo de técnicas que requieren poca preparación de la información, WEKA puede ser una muy buena alternativa como opción para dar una mirada rápida a los resultados obtenidos.
* No fue posible identificar los clusters visualmente debido a que no hay dos variables que los describan de buena forma y la herramienta solo permite visualizar en dos dimensiones.