Universidad de Alcalá
Escuela Politécnica Superior

GRADO EN INGENIERÍA DE SISTEMAS DE INFORMACIÓN

Trabajo Fin de Grado

Análisis de herramientas para el estudio de técnicas de aprendizaje automático

Autor: Tatiana Alonso Vegas

Tutor/es: Julia María Clemente Párraga

2021

UNIVERSIDAD DE ALCALÁ
Escuela Politécnica Superior

GRADO EN INGENIERÍA DE SISTEMAS DE INFORMACIÓN

Trabajo Fin de Grado
Análisis de herramientas para el estudio de técnicas de aprendizaje automático

Autor: Tatiana Alonso Vegas

Tutor/es: Julia María Clemente Párraga

TRIBUNAL:

Presidente: María del Mar Lendínez Chica

Vocal 1º: Concepción Batanero Ochaíta

Vocal 2º: Julia María Clemente Párraga

FECHA: 29 de septiembre 2021

A mi hermana, por luchar sin descanso salvando vidas en medio de una pandemia.

A Lidia, por su apoyo incondicional y sus cuidados diarios.

A Paloma, gracias por todo el amor que diste, te llevaré siempre en mi corazón.

Agradecimientos

Esta parte está dedicada a agradecer a todas aquellas personas que estuvieron apoyándome en estos años que fueron realmente difíciles.

En primer lugar dar las gracias a Lidia por estar a mi lado y darme su apoyo diario. En estos últimos años de pandemia has estado a mi lado dándome fuerzas, ánimos y sobre todo enseñarme a como ver la vida desde otra perspectiva. Cuando me sentía bloqueada o pensaba que no podía dar más de mí, ahí estabas tu para ver qué es lo que realmente me estaba ocurriendo, dándome apoyo y comprensión.

A mi familia por darme su apoyo incondicional incluso en los momentos más difíciles. Me habéis enseñado valores y aprendizajes que hacen que sea la persona que soy ahora. Obviamente, no dejaré de aprender a lo largo de mi vida ya que esta vida es un constante aprendizaje y quiero verla junto a vosotros.

A mis amigos y amigas que estuvieron a mi lado en cada momento compartiendo anécdotas, risas, críticas, frustraciones. Algunos de vosotros\as fuisteis un gran apoyo en los últimos años haciendo que la universidad fuera un lugar menos hostil. Siempre os recordaré con una sonrisa y alegría los buenos momentos que pasamos.

A Paloma. Gracias por todos tus apoyos, ánimos y optimismo que siempre me diste. Cada vez que te contaba alguna noticia sobre mí me dabas fuerzas y ánimos. Siempre me saludabas con una sonrisa, te preocupabas por mí y me enviabas siempre besos y corazones cuando hablábamos por chat. Todavía recuerdo cuando salíamos de fiesta las ganas que tenías de bailar y de pasártelo bien. Gracias a ti pude disfrutar y aprender de cosas nuevas que no había vivido anteriormente. Hoy día, te tengo muy presente en mi pensamiento y en mi vida. Siempre te recordaré por haber sido una bellísima persona y haber podido formar parte de mi familia y de mi vida. Gracias por todo. Que la tierra te sea leve.

Índice General

Índice de Figuras

Índice de Tablas

Resumen

El Aprendizaje Automático es una rama de la Inteligencia Artificial que consiste en identificar patrones y recopilar información útil de grandes conjuntos de datos para un uso futuro. Disponemos de una serie de herramientas de aprendizaje automático para poder definir que técnicas de aprendizaje darán paso a la obtención de predicciones que serán de gran utilidad. Estas herramientas disponibles funcionan mediante una interfaz para recibir datos y extraer resultados significativos. La selección de una herramienta no es tarea fácil por lo que averiguaremos cual es la herramienta que más se adecua en base a unos criterios preseleccionados.

Abstract

Machine Learning is a branch of Artificial Intelligence that consists of identifying patterns and gathering useful information from large data sets for future use. We have a series of machine learning tools to be able to define which learning techniques will lead to obtaining predictions that will be very useful. These available tools work through an interface to receive data and extract meaningful results. Selecting a tool is not an easy task, so we will find out which is the most suitable tool based on some pre-selected criteria.

Palabras clave

Aprendizaje automático, Herramientas, Análisis, Algoritmos, Supervisado, No Supervisado, Inteligencia Artificial, Técnicas, Weka, Orange

PARTE I

Resumen Extendido

La Inteligencia Artificial (IA) según la Comisión Europea se ha referido recientemente a la IA como “sistemas de software (y posiblemente hardware) diseñados por humanos que ante un objetivo complejo, actúan en la dimensión física o digital percibiendo su entorno, a través de la adquisición e interpretación de datos estructurados o no estructurados, razonando sobre conocimiento, procesando la información derivada de estos datos y decidiendo las mejores acciones para lograr el objetivo dado. Los sistemas de IA pueden usar reglas simbólicas o aprender un modelo numérico, y también puede adaptar su comportamiento al analizar como el medio ambiente se ve afectado por sus acciones previas” (España, 2020).

Aprender puede ser definido como el proceso de mejorar nuestra habilidad para hacer un trabajo. El objetivo del aprendizaje puede ser adquirir el conocimiento de tipos diferentes: aprender un concepto, aprender reglas de clasificación, aprender relaciones de causa-efecto, etc..

El Aprendizaje Automático (Machine Learning, ML) es una rama de la Inteligencia Artificial que consiste en identificar patrones y recopilar información útil de grandes conjuntos de datos para un uso futuro. Un sistema puede aprender adquiriendo nuevo conocimiento o modificando el conocimiento que tiene para hacerlo más útil. El efecto del aprendizaje es dotar a la máquina de un nuevo conocimiento que permite dar solución a un rango mayor de problemas, obtener soluciones precisas o simplificar el conocimiento almacenado.

Aprender a partir de ejemplos es uno de los paradigmas más utilizados hoy en día y constituye la base para estrategias como las de aprender en la enseñanza

Para la obtención de predicciones necesitamos herramientas para el análisis de datos. Hoy en día hay una gran variedad de herramientas con las que podemos adquirir experiencia. Algunas de ellas son de código abierto por lo que no suponen un coste inicial y están al alcance del usuario. El aprendizaje es un proceso por el cual se adquieren habilidades, destrezas, conocimientos e incluso actitudes por medio del estudio, experiencia o enseñanza. En ámbito de la IA, el aprendizaje automático desempeña un papel importante en la actualidad y brinda a los usuarios información sobre los datos observados y tratados.

El objetivo de este proyecto es realizar un análisis de algunas de las herramientas de aprendizaje automático más utilizadas en el ámbito de la aprendizaje/enseñanza. Estas

herramientas serán seleccionadas en base a unos criterios previamente investigados en términos de accesibilidad, intuición, simple de entender y comprender, seguridad, flexibilidad, facilidad toma de decisiones, explicabilidad e interpretabilidad (comprensión del modelo).

El trabajo comenzará con un estudio detallado sobre las técnicas de aprendizaje automático y su uso actual. Describirán algunas de las técnicas de aprendizaje automático más manejadas, poniendo de manifiesto sus ventajas y para qué se utilizan.

Para realizar el análisis previo es necesario seleccionar un conjunto de datos de entrenamiento realizando una adecuada limpieza de los mismos. Esta parte es una de las más compleja del proceso, conseguir una buena calidad de los datos a partir del conjunto escogido, es fundamental. Cuanta mejor calidad, conocimiento, limpieza de los datos, mejores resultados podremos esperar.

Para evaluar las herramientas se van a realizar pruebas partiendo de varios casos de estudios y aplicando las técnicas adecuadas a esos problema en las herramientas seleccionadas con el fin de evaluarlas de acuerdo con los criterios previamente investigados. Además, se dará especial relevancia a los resultados de otros autores en sus casos de estudio en términos de usabilidad, amigabilidad, etc.

La propia evolución de las herramientas de aprendizaje automático a lo largo de los años hace interesante este proyecto. El avance que han tenido las herramientas tecnológicamente permite expandir el aprendizaje con las diferentes funcionalidades que presentan en la actualidad.

Con todo lo anterior se pretende llega a un análisis crítico. Este concepto de análisis crítico es la evaluación del planteamiento o propuestas de un autor, es decir, la interpretación personal respecto a la posición de un autor, a partir de datos principales, extraídos del texto. Este concepto es muy utilizado en el ámbito de la Inteligencia Artificial ya que se incluyen en los avances, los errores, las mejoras, propuestas en un futuro, etc.

Las herramientas que se han elegido en este trabajo a analizar son dos: Weka y Orange. Además, se hará un estudio previo de las herramientas describiéndolas; cuáles son sus

funcionalidades y características principales. Cada una de ellas muestra distintas formas de extraer la información que será visualizada por el usuario. Tener una interfaz sencilla y si el resultado es fácil de interpretar es muy útil sobre todo para aquellos usuarios que no son expertos en el área del aprendizaje automático.

Como conclusión de este trabajo es necesario mencionar el éxito que estan teniendo las herramientas de aprendizaje automático en las diversas áreas. Estan sirviendo de soporte principal en el trabajo humano a la hora de recopilar datos y obtener resultados beneficiosos de tal magnitud que podemos prevenir, por ejemplo, enfermedades mortales como el cáncer, el cambio climático, estudios de mercados, marketing y ventas, etc. El análisis y evaluación de estas herramientas será de gran ayuda para impulsar la mejora del proceso de aprendizaje/enseñanza en la actualidad.

Como futuras líneas de trabajo a este proyecto, serían necesarios añadir otras técnicas de aprendizaje automático, nuevos criterios completando su evaluación y el análisis de las herramientas de aprendizaje automático.

PARTE II

Memoria del trabajo

Capítulo 1

1. Introducción

En este capítulo se hará una breve descripción de la evolución de la historia del aprendizaje automático desde su creación hasta la actualidad. A continuación, se explica en qué consiste el aprendizaje automático, la inteligencia artificial, los objetivos que se pretenden alcanzar y las tareas que se van a realizar. Por último, se comentará sobre la estructura en la que está dividido este documento.

1.1 Historia del Aprendizaje Automático

En 1950 Alan Turing crea el "Test de Turing" para determinar si una máquina era inteligente. Para pasar el test, una máquina tenía que ser capaz de engañar a un humano haciéndole creer que era humana en lugar de un computador (Pinar Saygin, Cicekli, & Akman, 2001).

Después, Arthur Samuel en 1952 escribe el primer programa de ordenador capaz de aprender. El software era un programa que jugaba a las damas y que mejoraba su juego partida tras partida.

En 1956 Martin Minsky y John McCarthy, con la ayuda de Claude Shannon y Nathan Rochester, organizan la conferencia de Darthmouth de 1956 considerada como el evento donde nace el campo de la Inteligencia Artificial. Durante la conferencia, Minsky convence a los asistentes para acuñar el término "Artificial Intelligence" como nombre del nuevo campo. Posteriormente, en 1958 Frank Rosenblatt diseña Perceptrón, la primera red neuronal (Ramírez, 2018).

En la segunda mitad de los años 70 el campo IA sufrió su primer "invierno". Esto significó un atraso ya que diferentes agencias que financiaban la investigación de la Inteligencia Artificial cortan los fondos tras numerosos años de altas expectativas y muy pocos avances.

Más tarde en los años 80, se volvió a generar gran interés por el Aprendizaje Automático debido al nacimiento de los sistemas expertos basados en reglas. En el año 1981, Gerald Dejong introduce el concepto "Explanation Base Learning" (EBL) donde un computador analiza datos de entrenamiento y crea reglas generales que le permiten descartar los datos menos importantes. Después, en el año 1985, Terry Sejnowski inventa NetTalk, que aprende a pronunciar palabras de la misma manera que lo haría un niño.

A finales de los años 80 y durante la primera mitad de los años 90 llegaría el segundo "invierno" de la Inteligencia Artificial y más prolongado que el anterior. Esta vez sus efectos se extenderán durante muchos años y la reputación del campo no se recuperará del todo hasta entrados los 2000. Durante esta época en 1997 el ordenador Deep Blue, de IBM vence al campeón mundial de ajedrez Gary Kaspárov (IBM, 1997).

Desde el año 2006 hasta la actualidad el aumento de la potencia de cálculo junto con la gran abundancia de datos disponibles ha vuelto a lanzar el campo del Aprendizaje Automático. Numerosas empresas están transformando sus negocios hacia el dato y están incorporando técnicas de Aprendizaje Automático en sus procesos, productos y servicios para obtener ventajas. En 2006 Geoffrey Hinton determina el término "Deep Learning" (Aprendizaje Profundo) para explicar nuestras arquitecturas de Redes Neuronales profundas que son capaces de aprender mucho mejor modelos más planos. En 2011, el ordenador Watson de IBM vence a sus competidores humanos en el concurso Jeopardy que consiste en contestar preguntas formuladas en lenguaje natural. Más tarde, en 2012, el laboratorio de investigación Google X utiliza GoogleBrain para analizar autónomamente videos de Youtube y detectar aquellos que contienen gatos.

En 2015 Amazon lanza su propia plataforma de Aprendizaje Automático y Microsoft crea el "Distributed Machine Learning Toolkit" que permite la distribución eficiente de problemas de aprendizaje automático en múltiples computadoras.

En la actualidad estamos viviendo una tercera explosión de la IA, ya que la gran disponibilidad de datos parece ser el "diesel" que está alimentando los motores de los algoritmos que, a su vez, han roto las limitaciones que existían antes de la computación distribuida. Todo parece indicar que seguiremos disponiendo de más y más datos con los que alimentar nuestros algoritmos y que los próximos años prometen ser realmente frenéticos.

Figura 1. Línea Temporal del Aprendizaje Automático *(Gonzalez Pachecho, 2019).*

Sobre definiciones de Inteligencia Artificial se encuentran autores como Rich y Knigth (Rich & Knight, 1994), y Stuart (Rusell & Meter, 1996) quienes definen en forma general la Inteligencia Artificial como la capacidad que tienen las máquinas para realizar tareas que en el momento son realizadas por los seres humanos. Otros autores como Nebendah (Dieter, 1988) y Delgado (Delgado, 1998), la definen como el campo de estudio que se enfoca en la explicación y emulación de la conducta inteligente en función de procesos computacionales basados en la experiencia y el conocimiento continuo del ambiente.

Se encuentran sobre el Aprendizaje Automático definiciones de autores como (Herbrich & Thore, 2015) que definen que el aprendizaje automático trata de hacer que las computadoras modifiquen o adapten sus acciones (ya sean predicciones o controlen un robot) para que estas acciones sean más precisas. Otro autor M. Mitchell (M.Mitchell, 1997) define el aprendizaje automático de esta forma: *"Se dice que un programa aprende de la experiencia E con respecto a alguna clase de tareas T y medida de rendimiento P, si su rendimiento en tareas en T, medida por P, mejora con la experiencia E"*.

1.2 Herramientas de Aprendizaje Automático

Las herramientas de aprendizaje automático se utilizan para el análisis de datos ya que aumentan la precisión, la eficiencia de su investigación en el ámbito del aprendizaje automático. Algunas de ellas sin requerir costos iniciales, ni a posteriori en su utilización. Actualmente existe una amplia gama de herramientas de aprendizaje automático, por lo que se debe elegir una o varias de ellas para adquirir experiencia.

Con las herramientas podemos explorar los datos interactivos y visualización de resultados del modelo, comparar de diferentes modelos de aprendizaje y tener una plataforma integrada en nuestro equipo para la automatización del proceso de datos a elegir.

1.3 Objetivos

El objetivo principal del presente trabajo es la evaluación y análisis de las herramientas de aprendizaje automático para su posterior estudio y uso de las principales técnicas en este campo. Se crearán pruebas de utilización de las herramientas y después se realizará una comparativa de las herramientas.

Para lograr el objetivo previo, se han establecido varios subobjetivos:

- Realizar un estudio detallado y comparativa sobre las herramientas de aprendizaje automático más relevantes utilizadas en el manejo e interpretación de técnicas de aprendizaje automático para el análisis de datos.

- Realizar un estudio de criterios de evaluación de las herramientas a analizar; que se pretende valorar en las herramientas elegidas. Este estudio puede orientar al usuario a enriquecer el proceso de aprendizaje/enseñanza y la elección de las herramientas.

- Aplicar las técnicas de aprendizaje automático en las diversas herramientas seleccionadas en estudio. Se realizarán pruebas en base a varios casos de estudios y aplicando las técnicas adecuadas a esos problema en las herramientas seleccionadas con el fin de evaluarlas de acuerdo con los criterios previamente investigados y finalmente seleccionados. Con estas pruebas se consigue examinar la herramienta al detalle.

- Recopilar todos los datos obtenidos en las pruebas y los conocimientos que se poseen de las herramientas, redactar un informe comparativo de las diferentes herramientas utilizadas y un análisis crítico. También se realizará un análisis de los resultados obtenidos en la evaluación diseñado mediante el planteamiento de diversos casos de estudio.

1.4 Planificación inicial

El objetivo principal del proyecto, como ya se ha descrito en la sección 1.3 Objetivos, que consiste en analizar las distintas herramientas de aprendizaje automático en base a unos criterios previamente seleccionados. Así como evaluación y la realización de una comparativa entre las herramientas.

La planificación de este trabajo es la siguiente:

- Fase de Planificación:
  - Consulta Bibliográfica.
  - Estudio Marco Tecnológico-Teórico:
    - Estudio y Comparativa de las herramientas más relevantes de aprendizaje automático y su uso actual.
    - Búsqueda y Evaluación de criterios que deberán cumplir las herramientas en términos de accesibilidad, intuición, simple de entender y comprensión.
- Fase de Pruebas:
  - Pruebas con diferentes escenarios de técnicas de aprendizaje automático y de las diversas herramientas evaluadas en el proyecto de acuerdo con los criterios previamente seleccionados.
- Documentación:
  - Memoria del proyecto, Conclusiones y Recopilación Bibliográfica.

1.5 Estructura del trabajo

El Proyecto Final de Grado que se presenta se ha estructurado en diferentes partes. La primera parte presenta el resumen extendido del proyecto cuya finalidad es sintetizar el resto del trabajo. La segunda parte representa la memoria del proyecto, es decir, el cuerpo del proyecto. Por último, se incluyen los ficheros utilizados en las pruebas y las referencias que engloban a las diferentes partes en las que queda dividido el proyecto.

La parte que recoge lo referente a la memoria del trabajo se ha dividido en 6 puntos clave. El primero de ellos presenta la introducción en la que está incluida la historia y las herramientas, los objetivos y planificación inicial del proyecto y la propia estructura del trabajo. A continuación, el segundo capítulo incluye los fundamentos teóricos necesarios para el buen entendimiento del proyecto. Este segundo capítulo abarca el marco teórico referente al aprendizaje automático, tipos y algoritmos de aprendizaje automático.

El tercer capítulo incluye el análisis de herramientas de este proyecto, es decir, el estudio de las herramientas de aprendizaje automático, funcionalidades y características generales, con el objetivo de entender y conocer mejor las herramientas y además el estudio de otras propuestas interesantes planteadas en el mismo área, con el objetivo de aprovechar los conocimientos de otros investigadores y de especificar aspectos que este trabajo ofrece. En el cuarto capítulo se detalla la evaluación de las herramientas que consistirá en poner a prueba con diferentes fuentes seleccionadas las técnicas de aprendizaje automático más utilizadas actualmente.

El quinto capítulo se exponen las conclusiones que han sido obtenidas a lo largo del proyecto así como las posibles líneas a futuro que pueden seguirse para continuar el estudio de herramientas de aprendizaje automático. Por último, se añade la parte que comprende el glosario, anexo y bibliografía. El anexo contiene los ficheros utilizados en las pruebas.

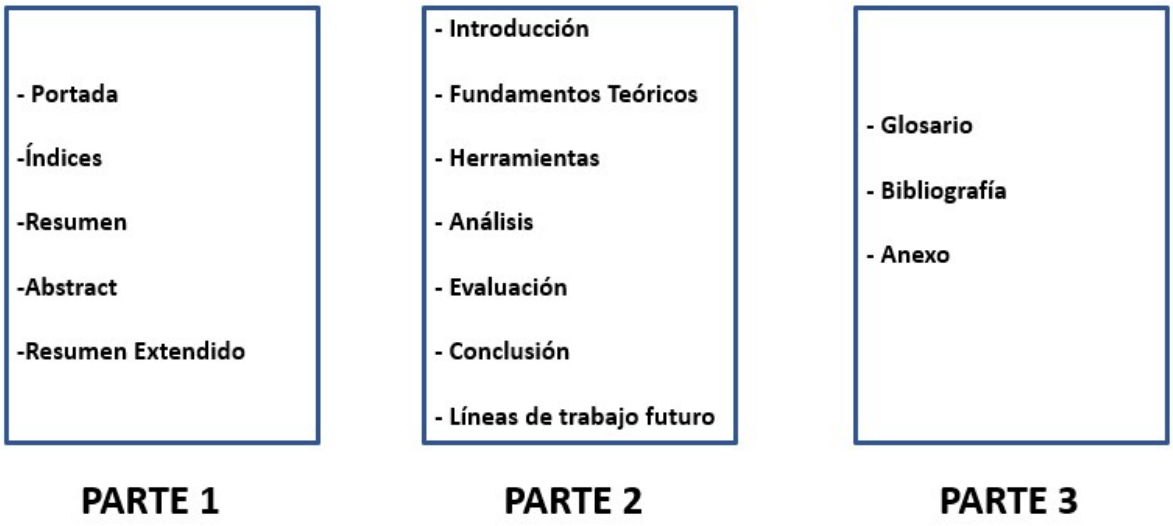


Figura 2. Estructura del trabajo (Elaboración propia).

Capítulo 2

2. Fundamentos Teóricos

Tras la introducción, que incluye el contexto general de este trabajo, es necesario especificar los conocimientos fundamentales dentro del marco teórico. En este capítulo se describirá el aprendizaje automático, los tipos de aprendizaje automático y sus características, ventajas, el aprendizaje automático en diferentes sectores, cuál es su uso en la actualidad y estudio de criterios previamente seleccionados para el análisis de las herramientas.

2.1 Qué es el Aprendizaje Automático

Antes de conocer que es el aprendizaje automático nos hacemos las siguientes preguntas: ¿Las máquinas realmente aprenden?, si es así ¿Cómo aprenden? ¿Cuáles son aquellas características importantes que se requieren para definir correctamente un problema de aprendizaje?

El aprendizaje se definiría como *"Proceso a través del cuál se adquieren o modifican habilidades, destrezas, conocimientos, conductas o valores como resultado del estudio, la experiencia, la instrucción, el razonamiento y la observación"*. No se considera aprendizaje aquello innato, si no aquello que procede de la experiencia con el entorno (Lázaro Enguita, 2018).

El aprendizaje automático se podría resumir en una frase muy utilizada por gran parte de los especialistas en el campo del aprendizaje automático: *"Se dice que un programa de computación aprende de la experiencia E con respecto a una tarea T y alguna medida de rendimiento P, si es que el rendimiento en T, medido por P, mejora con la experiencia E"* (Dutt, Chandramouli, & Das, 2020).

Después de saber qué es y en lo que consiste el aprendizaje automático, se puede considerar que una máquina aprende si es capaz de acumular experiencia al realizar una determinada

tarea y mejorar su rendimiento al realizar tareas en un futuro. Es por eso por lo que la máquina estará dotada de una serie de herramientas que acompañarán a su proceso de aprendizaje definiendo problemas reales.

El proceso básico de aprendizaje automático se puede dividir en tres partes (Dutt, Chandramouli, & Das, 2020):

- **Entrada de datos:** los datos o la información pasados se utilizan como base para la toma de decisiones futuras.

- **Abstracción:** los datos de entrada se representan de una manera más amplia a través del algoritmo subyacente.

- **Generalización:** la representación abstraída se generaliza para formar un marco para la toma de decisiones.

En la Figura 3 es la representación esquemática el proceso de aprendizaje automático.

Figura 3. Proceso de aprendizaje automático *(Dutt, Chandramouli, & Das, 2020).*

Para detallar la recopilación de datos paso a paso, la preparación de datos y el diseño del programa para resolver el problema, a continuación, resumimos los pasos por los cuales tendremos que pasar (Dutt, Chandramouli, & Das, 2020):

- Paso 1: ¿Cuál es el problema? Describir el problema de manera informal y formal y enumerar suposiciones y problemas similares.

- Paso 2: ¿Por qué es necesario resolver el problema? Enumerar la motivación para resolver el problema, los beneficios que proporcionará la solución y cómo se utilizará la solución.

- Paso 3: ¿Cómo solucionar el problema? Describir cómo se resolvería el problema manualmente para eliminar el conocimiento del dominio.

2.2 Tipos de Aprendizaje Automático

El aprendizaje automático se puede clasificar en tres amplias categorías, tal y como se destaca la Figura 4 (Dutt, Chandramouli, & Das, 2020):

- *Aprendizaje Supervisado:* también conocido como aprendizaje predictivo. Una máquina predice la clase de objetos desconocidos basándose en información previa relacionada con la clase de objetos similares.

- *Aprendizaje No supervisado:* también conocido como aprendizaje descriptivo. Una máquina encuentra patrones en objetos desconocidos agrupando objetos similares.

- *Aprendizaje por refuerzo:* una máquina aprende a actuar por sí misma para lograr los objetivos establecidos.

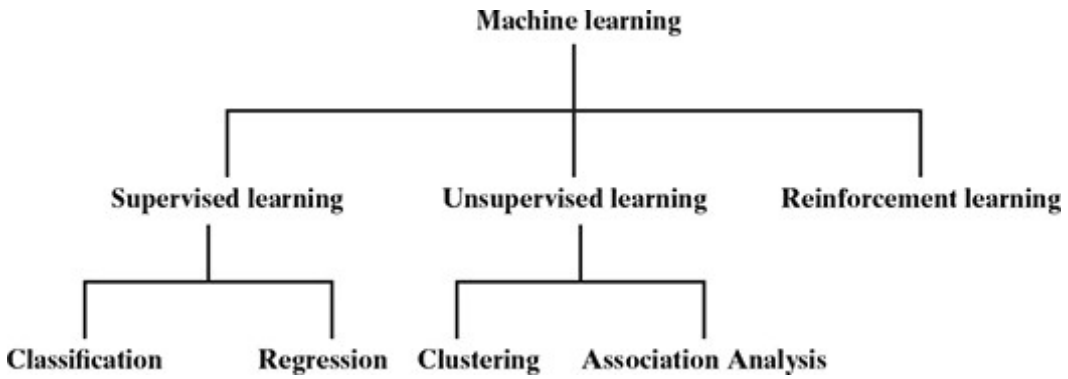


Figura 4. Tipos de aprendizaje automático *(Dutt, Chandramouli, & Das, 2020).*

2.2.1 Aprendizaje Supervisado

La motivación principal del aprendizaje supervisado es aprender de la información pasada. Es por eso por lo que nos hacemos esta pregunta: ¿Qué tipo de información pasada necesita la máquina para el aprendizaje supervisado? Es la información sobre la tarea que debe ejecutar la máquina. Esta información pasada es la experiencia.

Un ejemplo de su aplicación sería el siguiente: una máquina está obteniendo imágenes de diferentes objetos como entrada y la tarea es segregar las imágenes por la forma o el color del objeto. Si es por forma, las imágenes que son de objetos de forma redonda deben separarse de las imágenes de objetos de forma triangular, etc. Si la segregación debe ocurrir según el color, las imágenes de objetos azules deben separarse de las imágenes de objetos verdes. Pero ¿cómo puede la máquina saber qué es la forma redonda o la forma triangular? De la misma manera, ¿cómo puede la máquina distinguir la imagen de un objeto en función de si es de color azul o verde? Una máquina se parece mucho a un niño pequeño cuyos padres o adultos necesitan guiarlo con la información básica sobre la forma y el color antes de que pueda comenzar a realizar la tarea. Una máquina necesita que se le proporcione la información básica. Este insumo básico, o la experiencia en el paradigma del aprendizaje automático, se da en forma de datos de entrenamiento. Los datos de entrenamiento son la información pasada sobre una tarea específica. En el contexto del problema de la segregación de imágenes, los datos de entrenamiento tendrán datos anteriores sobre diferentes aspectos o características en varias imágenes, junto con una etiqueta sobre si la imagen es redonda o triangular, o de color azul o verde. La etiqueta se llama "etiqueta" y decimos que los datos de entrenamiento están etiquetados en caso de aprendizaje supervisado (Dutt, Chandramouli, & Das, 2020).

La Figura 5 expone una descripción simple del proceso de aprendizaje supervisado. Los datos de entrenamiento etiquetados que contienen información pasada vienen como entrada. Después en función de los datos de entrenamiento, la máquina crea un modelo predictivo que se puede utilizar en los datos de prueba para asignar una etiqueta a cada registro en los datos de prueba (Dutt, Chandramouli, & Das, 2020).

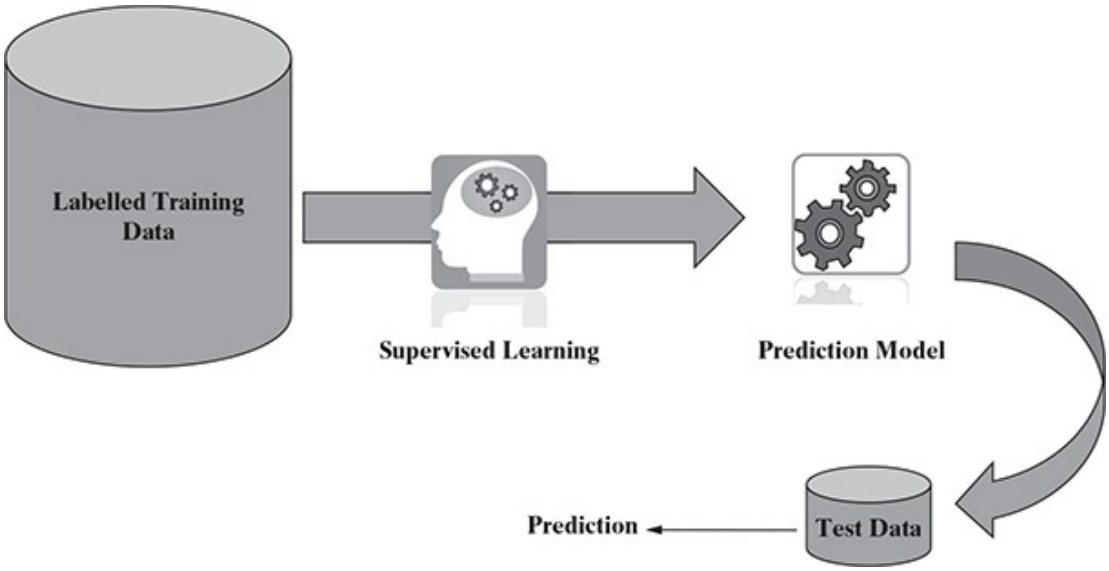


Figura 5. Aprendizaje Supervisado (Dutt, Chandramouli, & Das, 2020).

Algunos ejemplos del aprendizaje supervisado son:

- Predecir los resultados de un juego.
- Predecir si un tumor es maligno o benigno.
- Predecir el precio de las acciones, inmuebles.
- Clasificar textos, conjunto de correos como *spam* o no *spam*.

2.2.2 Aprendizaje No Supervisado

A diferencia del aprendizaje supervisado, el aprendizaje no supervisado no contiene datos de entrenamiento etiquetados de los que aprender ni predicciones qué hacer. El objetivo es tomar un conjunto de datos como entrada y tratar de encontrar agrupaciones o patrones naturales dentro de los elementos o registros de datos. Por lo tanto, el aprendizaje no supervisado se denomina modelo descriptivo y su proceso se denomina descubrimiento de patrones o conocimientos. Una aplicación fundamental del aprendizaje no supervisado es la segmentación de clientes.

La agrupación en clústeres es el principal tipo de aprendizaje no supervisado. Tiene la intención de agrupar u organizar similares juntos. Es por eso por lo que los objetos pertenecen al mismo grupo son bastantes similares entre sí mientras que los objetos que pertenecen a diferentes grupos son bastante diferentes.

De ahí el objetivo de la agrupación para descubrir la agrupación intrínseca de datos no etiquetados y formar agrupaciones como se muestra en la Figura 6. Se pueden aplicar diferentes medidas de similitud para la agrupación.

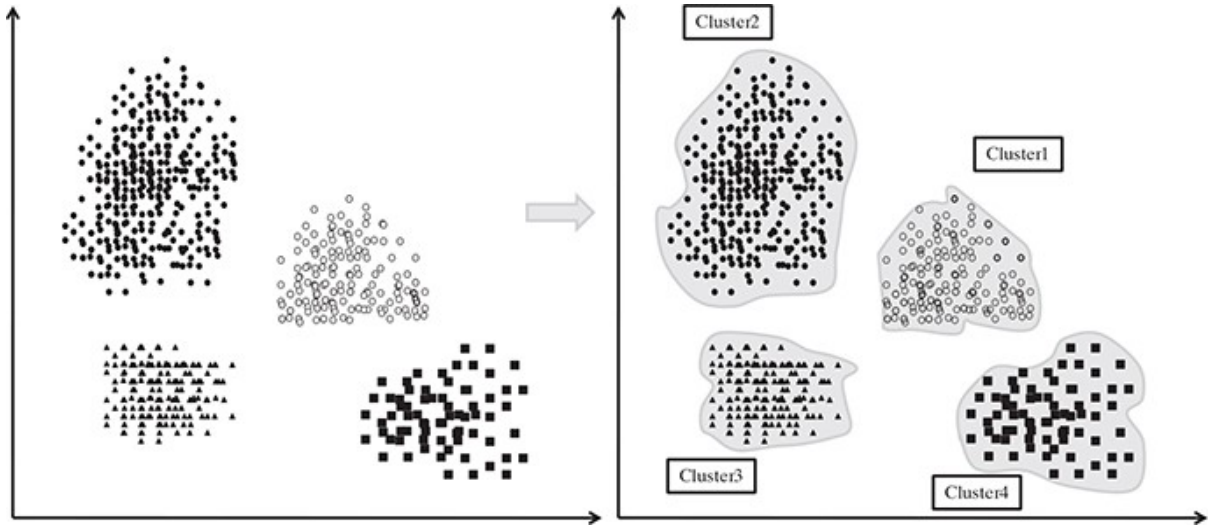


Figura 6. Agrupación basada en la distancia (Dutt, Chandramouli, & Das, 2020).

Una de las medidas de similitud más comúnmente adoptadas es la distancia. Dos elementos de datos se consideran parte del mismo grupo si la distancia[1] entre ellos es menor. De la misma forma, si la distancia entre los elementos de datos es alta, los elementos generalmente no pertenecen al mismo grupo. Se conoce esto también como agrupación en clústeres basada en la distancia. La Figura 7 describe el proceso de agrupamiento en un nivel alto.

[1] Para más detalles véase sección 4.2 Métricas de Distancias (Arriagada Rodríguez, 2015)

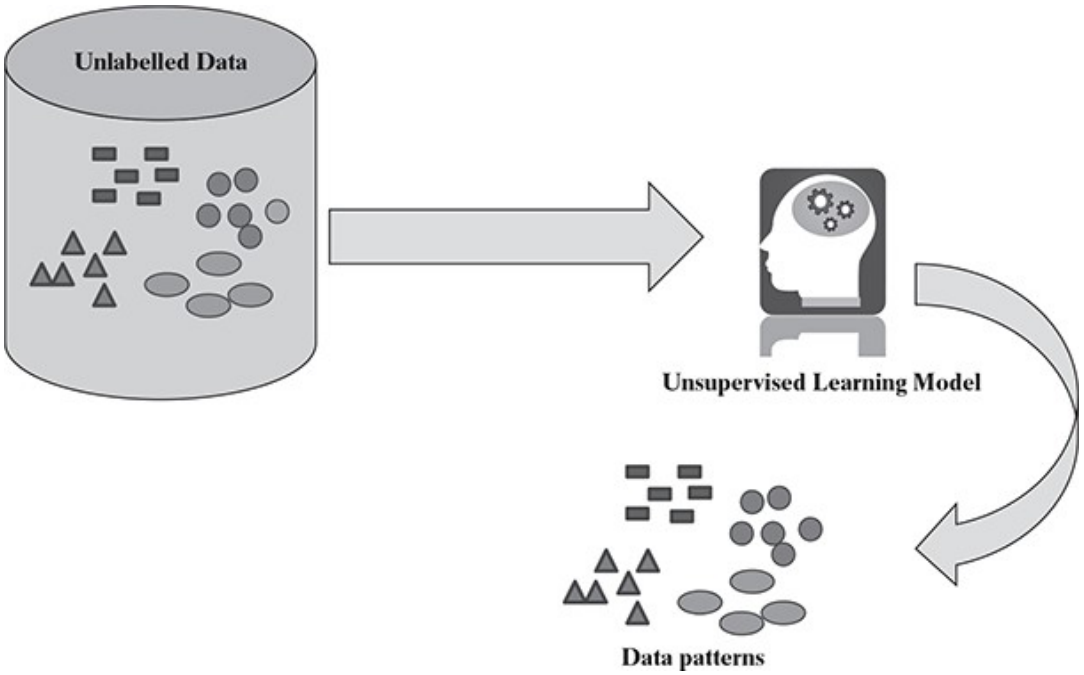


Figura 7. Aprendizaje No Supervisado (Dutt, Chandramouli, & Das, 2020).

Además de agrupar los datos y obtener una vista resumida de ellos, una variante más del aprendizaje no supervisado es el análisis de asociación. Se identifica entre elementos de datos.

2.2.3 Aprendizaje Reforzado o Aprendizaje por Refuerzo

El aprendizaje reforzado o por refuerzo es aprender que hacer para maximizar una señal de recompensa numérica. Se debe descubrir que acciones producen la mayor recompensa probándolas. En los casos más interesantes y desafiantes, las acciones pueden afectar no solo la recompensa inmediata sino también la siguiente situación y, a través de ella, todas las recompensas posteriores. Estas dos características, búsqueda de prueba y error y recompensa retrasada, son las dos características distintivas más importantes del aprendizaje por refuerzo. El aprendizaje por refuerzo no se define caracterizando los métodos de aprendizaje, sino caracterizando un *problema de aprendizaje*. Cualquier método que sea adecuado para resolver ese problema, se considera un método de aprendizaje por refuerzo. Una especificación completa del problema del aprendizaje por refuerzo en términos de control óptimo de los procesos de decisión de Markov[2], pero la idea básica es capturar los aspectos más importantes del problema real que enfrenta un agente de aprendizaje al interactuar con su entorno para lograr un objetivo. El agente debe poder sentir el estado del medio ambiente hasta cierto

punto y debe poder tomar acciones que afecten al estado. El agente también debe tener una meta o metas relacionadas con el estado del medio ambiente. La formulación está destinada a incluir solo estos tres aspectos: sensación, acción y objetivo, en sus formas más simples posibles sin trivializar ninguno de ellos. Uno de los desafíos que surgen en el aprendizaje por refuerzo y no en otros tipos de aprendizaje es el compromiso entre *exploración* y *explotación*. Para obtener una gran recompensa, un agente de aprendizaje por refuerzo debe preferir las acciones que ha intentado en el pasado y ha encontrado que son efectivas para producir recompensas. Pero para descubrir tales acciones, tiene que probar acciones que no haya seleccionado antes. El agente tiene que *explotar* lo que ya sabe para obtener una recompensa, pero también tiene que *explorar* para hacer mejores selecciones de acción en el futuro. El dilema es que ni la exploración ni la explotación pueden perseguirse exclusivamente sin fallar en la tarea. El agente debe intentar una variedad de acciones que favorezca progresivamente las que parezcan mejores. En una tarea estocástica, cada acción debe intentarse muchas veces para obtener una estimación confiable de su recompensa esperada (Sutton & Barto, 2005).

Una buena forma de entender el aprendizaje reforzado es considerar algunos ejemplos y posibles aplicaciones que han guiado a su desarrollo que son los siguientes (Sutton & Barto, 2005):

- Un jugador de ajedrez maestro hace un movimiento. La elección se basa tanto en la planificación (anticipando posibles respuestas y contrarrespuestas) como en juicios intuitivos sobre la conveniencia de determinadas posiciones y movimientos.

- Un controlador adaptativo ajusta los parámetros del funcionamiento de una refinería de petróleo en tiempo real. El controlador optimiza el equilibrio rendimiento/costo/calidad sobre la base de costos marginales especificados sin ceñirse estrictamente a los puntos de ajuste sugeridos originalmente por los ingenieros.

- Una cría de gacela lucha por ponerse en pie minutos después de nacer. Media hora más tarde, corre 20 km por hora.

[2] Véase capítulo 3 del libro Reinforcement Learning: An Introduction (Sutton & Barto, 2005).

En conclusión, en la Tabla 1 podemos ver detalladamente la comparación de los distintos tipos de aprendizaje comentados anteriormente.

Aprendizaje Supervisado	**Aprendizaje No Supervisado**	**Aprendizaje Reforzado**
Este tipo de aprendizaje se utiliza cuando queremos saber cómo clasificar un dato dado, o en otras palabras, hay clases o etiquetas disponible.	Este tipo de aprendizaje se utiliza cuando no hay idea de la clase o etiqueta de un dato en particular. El modelo tiene que encontrar un patrón en los datos.	Este tipo de aprendizaje se utiliza cuando no hay idea de un dato en particular. El modelo tiene que hacer la clasificación; será recompensado si la clasificación es correcta, de lo contrario será castigado.
Se necesitan datos de entrenamiento etiquetados. El modelo se basa en datos de entrenamiento.	Cualquier conjunto de datos desconocido y sin etiquetar se proporciona al modelo como entrada y los registros se agrupan	El modelo aprende y se actualiza a sí mismo a través de recompensas / castigos
El rendimiento del modelo se puede evaluar en función de la cantidad de clasificaciones erróneas que se hayan realizado en función de una comparación entre los valores predichos y reales.	Es difícil medir si el modelo hizo algo útil o interesante. La homogeneidad de los registros agrupados es la única medida.	El modelo se evalúa mediante la función de recompensa después de haber tenido algún tiempo para aprender
Hay dos tipos de clasificación y regresión de problemas de aprendizaje supervisado.	Hay dos tipos de clasificación y regresión de problemas de aprendizaje no supervisado.	No hay tipos
El más simple de entender.	Más difícil de entender e implementar que el aprendizaje supervisado.	Más complejo de entender y aplicar.
Los algoritmos estándar incluyen: • Naive Bayes • *k*-vecinos cercanos kNN • Árbol de decisión	Los algoritmos estándar son: • *k*-means • Análisis de componentes principales (PCA) • Mapa Autoorganizado (SOM) • Algoritmo A priori • DBSCAN, etc.	Los algoritmos estándar son: • Q-learning • Sarsa
Las aplicaciones prácticas incluyen: • Reconocimiento de escritura a mano • Predicción del mercado de valores • Predicción de enfermedades • Detección de fraude	Las aplicaciones prácticas incluyen: • Análisis de mercado • Sistemas de recomendación • Segmentación de clientes, etc.	Las aplicaciones prácticas incluyen: • Coches autónomos • Robots inteligentes • AlphaGo Zero (la última versión del sistema al de DeepMing jugando Go)

Tabla 1. Comparación Aprendizaje Supervisado, Aprendizaje no Supervisado y Aprendizaje Reforzado (Traducido) (Dutt, Chandramouli, & Das, 2020).

2.3 Ventajas del Aprendizaje Automático

El aprendizaje automático tiene muchas aplicaciones y sus posibilidades se expanden continuamente.

Estas son algunas de las principales ventajas (Microsoft, 2020):

- **Descubrir la información:** El aprendizaje automático puede ayudar a identificar un patrón o una estructura en datos estructurados o no estructurados, lo que ayuda a entender lo que los datos están diciendo.

- **Mejorar la experiencia del usuario:** Algunos ejemplos pueden ayudar a optimizar la experiencia.

- **Prever el comportamiento:** El aprendizaje automático puede extraer datos relacionados con un patrón para ayudar a identificar patrones y comportamientos, lo que le permite optimizar las recomendaciones de productos y proporcionar la mejor experiencia posible.

- **Mejora la integridad de datos:** El aprendizaje automático es excelente en la minería de datos y puede llevarlo más lejos al mejorar sus capacidades con el tiempo.

- **Reducir el riesgo:** Puesto que las tácticas de fraude cambian constantemente, el aprendizaje automático mantiene el ritmo: supervisa e identifica nuevos patrones para detectar intentos antes de que se concreten.

- **Reducir los costes:** Una aplicación del aprendizaje automático es la automatización de procesos, que puede liberar tiempo y recursos para que su equipo pueda dedicarse a lo que más importa.

2.4 Aprendizaje Automático en diferentes sectores

El aprendizaje automático facilita que los diferentes sectores de la sociedad puedan hacer predicciones, detectar fallos y tomar decisiones. A continuación, se muestran los sectores donde se aplica el aprendizaje automático (Cámara Madrid, 2019):

- **Sector Bancario:** puede implementar soluciones de aprendizaje automático para detección del fraude, incumplimiento de normativas, *credit scoring*[3] o asistencia virtual para los clientes.

- **Turismo:** el aprendizaje automático está ayudando a aerolíneas, hoteles, agencias turísticas, etc., a realizar predicciones para mejorar los servicios, como estrategias de segmentación de clientes, personalización de la estancia de huéspedes, predicción de cancelaciones o pernoctaciones y mejora de la eficiencia de las campañas online.

- ***E-Commerce*:** el aprendizaje automático es inseparable de las aplicaciones de *e-commerce* en el sector minorista, donde podemos ver soluciones como los ya populares *chatbots*, que proporcionan atención personalizada al cliente de forma automática, o tecnología de aprendizaje automático orientada a mejorar la experiencia de usuario o la optimización de los KPIs[4].

- **Industria:** el aprendizaje automático y el *Big Data* ayudan a mejorar los sistemas robóticos del sector industrial, facilitando la transformación digital de la industria y ayudando a una toma de decisiones inteligente que ayude a optimizar los costes, reducir los tiempos de producción y conseguir los mejores resultados.

- **Salud:** el sector de la salud es otra de las ramas que se está beneficiando de las ventajas del aprendizaje automático, también del *Deep Learning*, y la multitud de aplicaciones en medicina preventiva basándose en modelos de redes neuronales.

[3] *El credit scoring es un sistema de modelos de decisión a través del cual se calcula la probabilidad de que un sujeto sea capaz de devolver o no un crédito comercial. (Bilbao, 2015)*
[4] *Key Performance Indicator, también conocido como indicador clave de rendimiento, es una medida del nivel del rendimiento de un proceso.*

2.5 Actualidad y Aplicaciones del Aprendizaje Automático

"*Más del 60% de los CEOs Españoles ya están aplicando la Inteligencia Artificial en sus procesos de automatización y el 25% de las empresas invierten hasta 44 millones de euros en modificar y reorientar sus modelos de negocio a los algoritmos*" (KPMG, 2020)

El concepto de Aprendizaje Automático, una de las ramas de la Inteligencia Artificial que dota a los ordenadores de un aprendizaje automático sin necesidad de ser programados de forma continuada, ha tomado mayor protagonismo en la última década. En pocos años, los algoritmos catalogados como Aprendizaje Automático han evolucionado para conseguir manejar grandes volúmenes de datos, obtener mejores resultados y resolver problemas de manera más eficiente (BDM, 2020).

El uso del Aprendizaje Automático cada vez es más variado. Según estimaciones de la consultora Accenture, su aplicación aumentará la productividad de las empresas de un 40% para el año 2035. Además, en la actualidad más del 60% de los CEOs Españoles ya están aplicando la Inteligencia Artificial en sus procesos de automatización y el 25% de las empresas invierten hasta 44 millones de euros en modificar y reorientar sus modelos de negocio a los algoritmos, según el último informe de KPMG (BDM, 2020).

La práctica del Aprendizaje Automático forma parte del día a día de la población mundial. Desde **Ironhack** -escuela líder en formación de talento digital de forma intensiva-, han recopilado **7** ejemplos que demuestran que el Aprendizaje Automático forma parte de la vida cotidiana **(**BDM, 2020):

- **Detección de rostro:** El reconocimiento facial es una de las revoluciones más importante de la década. Se usa para desbloquear el móvil, probar filtros de Snapchat o Instagram e, incluso, para intentar predecir cómo se envejece. Si bien parece algo nuevo, la primera vez que se utilizó fue a finales del siglo XIX por el oficial de policía francés Alphonse Bertillon con el objetivo de identificar el rostro de criminales y sustituir el método de huellas dactilares. El software identifica las caras mediante un grupo de 68 referencias o puntos concretos, más o menos, cuya configuración es diferente en cada persona.

- **Reconocimiento de voz:** Los primeros sistemas de reconocimiento de voz fueron creados en 1952 y se basaban en la potencia de voz del hablante. En la actualidad, se cuenta con sistemas como: "Ok Google" u "Oye Siri", entre otros. Este es uno de los mejores ejemplos de Aprendizaje Automático. Con el objetivo de entender mejor qué es lo que se necesita cuando formula una pregunta, estos asistentes virtuales terminan conociendo todo del usuario como: patrones de sueño, mensajes, calendario, recordatorios, mails, etc.

- **Gmail:** Al marcar los correos como *malware*, el sistema termina entendiendo y aprendiendo a enviar dichos mensajes directamente a la carpeta de "no deseados" para mantener al usuario protegido de virus, fraudes o mensajes que no le interesan.

- **Marketing personalizado:** Basado en la actuación del usuario cuando utiliza Internet, sus redes sociales o cómo interacciona, el Aprendizaje Automático aprende de ese comportamiento para recomendarle productos o servicios que encajen con él y se produce así un marketing personalizado basado en patrones de conducta. Empresas como Google, Amazon e Instagram, entre otras, trabajan con estos datos, ya que incrementa la eficiencia y productividad de las campañas. De hecho, gracias a la IA, las empresas pueden conocer las necesidades del usuario antes que él mismo lo sepa.

- **Google Maps para el tráfico:** Cada día se recorren más de 1.000 millones de km alrededor del mundo utilizando Google Maps. Esta herramienta muestra las rutas más seguras y eficientes utilizando tecnologías basadas en patrones de tráfico y de movilidad recopilados a lo largo del tiempo y combinándolo con condiciones de tráfico en vivo. De esta forma, es como se aplica el *Machine Learning* para poder generar pronósticos apoyados en ambos conjuntos de datos.

- **Coches autónomos:** En la actualidad, existen coches capaces de ser conducidos de manera autónoma, adelantar, aparcar o realizar cualquier tipo de maniobra. Este tipo de automóviles ofrecen la posibilidad de disminuir las incidencias de tráfico e incluso el número de accidentes, ya que, al eliminar el factor humano de la ecuación, el margen de error es prácticamente inexistente.

- **Diagnósticos médicos:** El uso de sistemas inteligentes dentro de la medicina tiene un gran potencial, ya que permiten procesar una gran cantidad de información y generar diagnósticos ayudando a detectar patologías con mayor rapidez y menor margen de error de lo que lo haría un ser humano.

Las principales áreas en las que se usa el Aprendizaje Automático son**: oncología**, donde se ha demostrado una eficacia del 90% en detección de cáncer de mama y próstata; **neurología**, donde se han conseguido grandes avances en diagnóstico y tratamiento de ictus, alzhéimer o demencia senil; **ginecología,** ayudando a detectar malformaciones o problemas durante el embarazo, y **genética**, con programas capaces de detectar mediante el rostro más de 8.000 trastornos genéticos y enfermedades raras (BDM, 2020).

En la actualidad se está empezando a experimentar las aplicaciones del aprendizaje automático en una gran variedad de ámbitos o campos. A continuación, se exponen estos ámbitos (Cámara Madrid, 2019):

- **Estudios de mercados:** las aplicaciones permiten la segmentación de clientes y la predicción de la demanda a partir de proyectos de Big Data con Inteligencia Artificial de cara a identificar mejor a los grupos de clientes.

- **Marketing y ventas:** hay multitud de aplicaciones que se pueden implementar en el ámbito digital, como las recomendaciones personalizadas según el perfil del usuario, o la detección de patrones de venta cruzada.

- **Atención al cliente:** mediante sistemas de reconocimiento de voz, texto o incluso vídeo, las máquinas pueden ofrecer soluciones y respuestas a los clientes de acuerdo con un aprendizaje previo, maximizando la satisfacción de los clientes.

- **Sistemas de calidad:** otro de los campos en los que el aprendizaje automático está teniendo éxito es en el control de calidad, detección de fraudes y descubrimiento de irregularidades, ayudando a mejorar la experiencia.

- **Automatización de procesos:** desde el control de accesos y la gestión de los Recursos Humanos, pasando por el proceso logístico, hay multitud de funciones en la empresa que se pueden optimizar al máximo gracias a la extracción de información implícita y descubrimiento de reglas, o el uso de fuentes de datos heterogéneos.

- **Producción:** la industria 4.0 se une de la mano de la robótica y el aprendizaje automático para mejorar procesos, aumentar la productividad, reducir costes o prevenir fallos que retrasen o paralicen la producción, ayudando a aumentar la competitividad de las empresas.

2.6 Herramientas de Aprendizaje Automático Seleccionadas

Para saber cuáles son las herramientas más utilizadas se ha tenido que recurrir a encuestas en las cuales hay una gran participación de la comunidad. En mayo de 2019, KDnuggets (portal web conocido por sus enlaces y recursos de análisis y minería de datos) realizó su encuesta anual de software preguntando: ¿Qué analítica, ciencia de datos, software/herramientas de aprendizaje automático ha utilizado en los últimos 3 años (2017-2019) para un proyecto real? Esta encuesta recibió más de 1.800 participantes. La Figura 8 muestra los resultados de la encuesta para las herramientas que se ubicaron en el top 40, según el número de votos que recibieron. El gráfico muestra el número de votos para cada una de estas herramientas (KDnuggets, 2019).

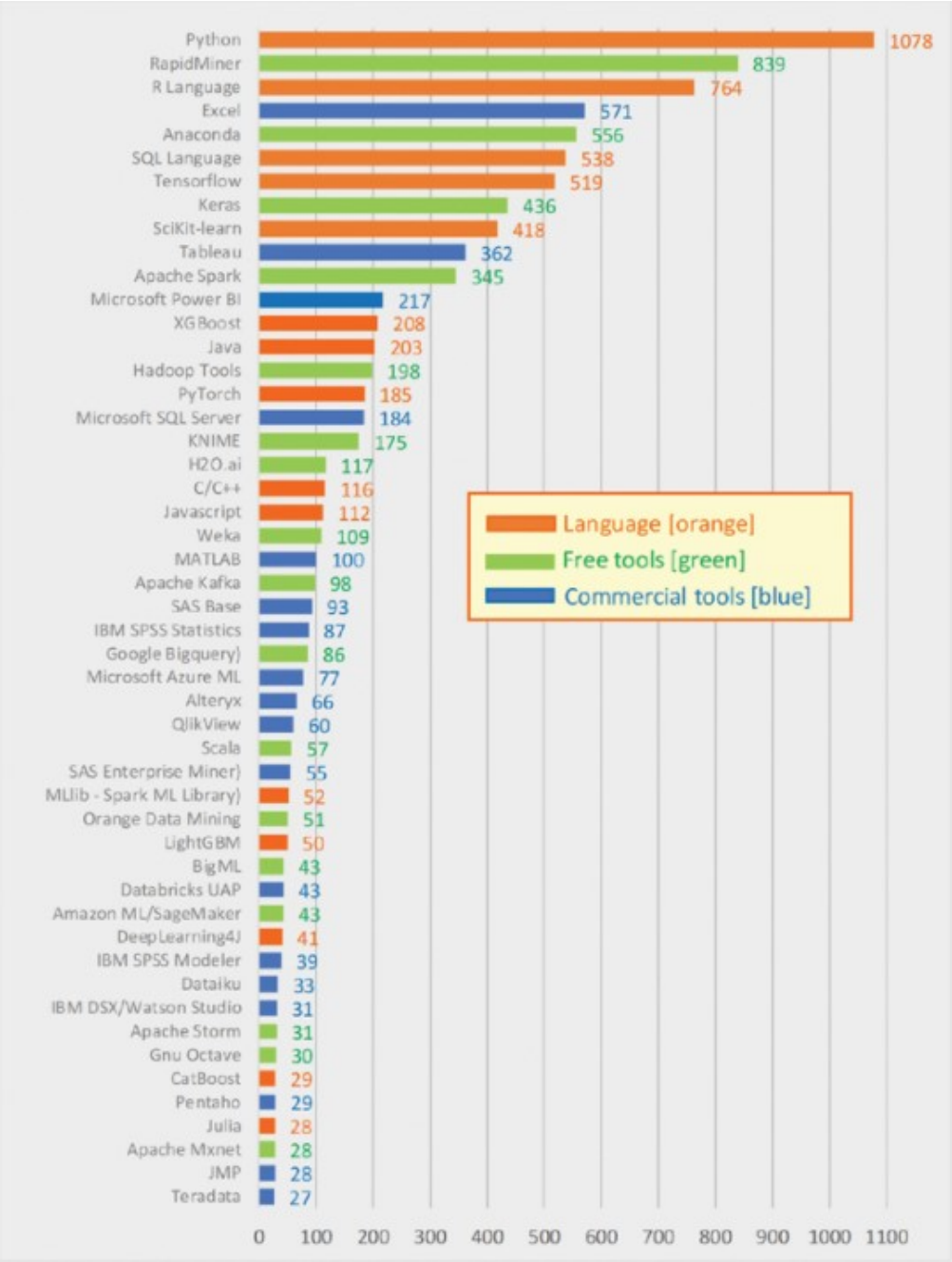


Figura 8. Herramientas de software de análisis y minería de datos más populares (resultados encuesta de usuarios) (KDnuggets, 2019).

Contrastando los resultados de la encuesta de KDnuggets y verificando que en la actualidad se encuentran herramientas que no implican realizar tareas de programación, que van dirigidas a usuarios sin conocimientos avanzados se han definido estas herramientas más utilizadas (datos.gob.es, 2021):

- Weka: software multiplataforma de aprendizaje automático y minería de datos. Se puede acceder a sus funcionalidades a través de su interfaz gráfica, una línea de comandos o una API de Java.

- Knime: software de minería de datos, que permite el análisis de datos y la realización de visualizaciones a través de una interfaz gráfica.

- Orange: software abierto de aprendizaje automático y minería de datos. Se puede acceder a sus funcionalidades a través de su interfaz gráfica y crear tus propios flujos en su lienzo.

Estas tres herramientas (Weka, Knime y Orange) además de ser unas de las más utilizadas ya que no implica programación y que se encuentran en el top 40 de la encuesta KDnuggets se han elegido para este trabajo dos herramientas de las tres comentadas anteriormente de aprendizaje automático a analizar que son: Weka y Orange. Ambas, se caracterizan por ser públicamente disponibles, es decir, son herramientas *Open Source*. Son fáciles de instalar, de utilizar (mediante interfaz gráfica o por comandos de consola), de elevada precisión y fiable. Además, sirven para entrenar de manera sencilla por lo que se configura fácilmente.

El estudio se ha limitado a estas herramientas mencionadas anteriormente por la envergadura de trabajo pero es fácilmente extensible a otras herramientas.

2.7 Criterios para el análisis de las herramientas disponibles

En este apartado se van a describir los criterios elegidos para analizar las herramientas que se han seleccionado. Estos criterios han sido escogidos en base a la necesidades de este trabajo y a los estudios de investigación relevantes realizados por determinados autores como son los siguientes:

- *Study of Open-Source Data Mining Tools for Forecasting* de Hasim Nurdatillah y Abu Haris Norhaidah.

- *A comparison of Contemporary Data Mining Tools* de Dakić Dušanka, Stefanović Darko, Sladojević Srdjan, Arsenović Marko, Lolić Teodora.

- *Performance Evaluation of Select Data Mining, Software tools for Data Clustering* de AO Ameen *, AO Bajeh, BA Adesiji, AO Balogun and MA Mabayoje.

- *An Extensive Study of Data Analysis Tools* (RapidMiner, Weka, R Tool, Knime, Orange) de Venkateswarlu Pynam, R Roje Spanadna, Kolli Srikanth.

- *Experimental Evaluation of Open-Source DATA Mining Tools (Weka and Orange) de* Ritu Ratra y Preeti Gulia.

- *Detailed Analysis of Data Mining Tools* de Rohit Ranjan, Swati Agarwal y Dr. S. Venkatesan.

- *A Study Data Mining in Knowledge Discovery Process* de Y. Ramamohan, K. Vasantharao, C. Kalyana Chakravarti, A.S.K. Ratnam.

- *Analysis of Data Minng Tool Orange* de Padmavaty, C. Geetha, N. Priya.

- *An overview of free software tools for general data mining* de A. Jović, K. Brkić and N. Bogunović.
- *Open-Source Data Mining Tools, a comparative study* de Hussah A. Al-Odan, Ahmad A. Al-Daraiseh King Saud.

- *A comparative Study on Data Mining Tools* de Priti S. Patel, Dr. S.G. Desai.

- *Weka Powerful Tool in Data Mining* de Eshwari Girish Kulkarni, Raj B. Kulkarni, PhD.

- Creación de perfiles de deudores de crédito universitario, para mejoramiento de campañas de cobranza, usando minería de datos de Carolina Verónica Lagos Vera.

Por lo tanto, los criterios seleccionados para analizar las herramientas según los criterios utilizados por determinados autores (que analizaremos en la sección 4.3) en sus investigaciones se describen a continuación (RAE, 2021):

- **Accesibilidad:** Cualidad de accesible y la condición que deben cumplir los entornos, productos y servicios para que sean comprensibles, utilizables y practicables por todos los ciudadanos, incluidas las personas con discapacidad.

- **Intuitivo:** Facultad de comprender las cosas instantáneamente, sin necesidad de razonamiento.

- **Simple de entender:** Saber manejar para algún fin.

- **Comprensión:** Facultad, capacidad o perspicacia para entender y penetrar en las cosas.

- **Compatible:** Dicho de una persona o de una cosa. Que puede estar, funcionar o coexistir sin impedimento con otra.

- **Rendimiento:** Producto o utilidad que rinde o proporción entre el producto o el resultado obtenido y los medios utilizados.

- **Usabilidad:** Facilidad con la que las personas empleamos una herramienta o cualquier objeto para cumplir un objetivo concreto.

- **Amigabilidad:** en término del área de la informática significaría la capacidad que tiene un programa de entregar una facilidad al usuario, es decir, fácil de utilizar.

- **Explicabilidad:** que se puede explicar, que es transparente.

- **Interpretabilidad**: que se puede interpretar.

2.8 Conclusión

Para este trabajo se han elegido dos herramientas de aprendizaje automático a analizar con unos criterios específicos. Cada una con sus propias características y funcionalidades. Según el estudio y el análisis que va a continuación veremos cuál es la que se adecúa y se adapta mejor a las necesidades del usuario.

Se ha optado por elegir **estas dos** herramientas a analizar, ya que son Open Source, es decir, de Software Libre y no requieren presupuesto inicial. Estas herramientas se adaptan perfectamente al objetivo de este proyecto:

- WEKA
- Orange

3. Algoritmos de Aprendizaje Automático

3.1 Algoritmos de aprendizaje automático supervisado

3.1.1 Random Forest

Según Leo Breiman Random Forest es una combinación de predictores de árboles tal que cada árbol depende de los valores de un vector muestreado independientemente y con la misma distribución para todos los árboles del bosque. El error de generalización para los bosques converge en un límite a medida que aumenta el número de árboles en el bosque. El error de generalización de un bosque de clasificadores de árboles depende de la fuerza de los árboles individuales en el bosque y la correlación entre ellos. Usando una selección aleatoria de características para dividir los rendimientos de cada nodo produce tasas de error que se comparan favorablemente al algoritmo Adaboost (Freund, 1996), pero son más robustos con respeto al ruido. El error de seguimiento de estimaciones internas, fuerza y correlación se utilizan para mostrar la respuesta al aumento del número de funciones utilizadas en la división. Las estimaciones internas también se utilizan para medir la importancia de la variable. Estas ideas también son aplicables a la regresión (Breiman, 2001).

Random Forest es un conjunto de árboles combinados con *bagging*, La idea principal del *bagging* es ajustar múltiples modelos, cada uno con un subconjunto diferente de datos de entrenamiento. Todos los modelos del agregado participan aportando su predicción. Se toman las medidas de todas las predicciones o la clase más frecuente.

Cada árbol se construye de la siguiente manera (Cutler & Breiman, s.f.):

1. El número de casos en el conjunto de entrenamiento es N, muestree N casos al azar, pero con *reemplazo*, a partir de los datos originales. Esta muestra será el conjunto de entrenamiento para hacer crecer el árbol.

2. Si hay *M* variables de entrada, se especifica un número *m* << *M* de modo que en cada nodo, *m* variables se seleccionan al azar de *M* y se usa la mejor división de estas *m* para dividir el nodo. El valor de *m* se mantiene constante durante el crecimiento del bosque.

3. Cada árbol se cultiva en la mayor medida posible. No hay poda.

Se demuestra además que la tasa de error forestal depende de dos factores:

- La **correlación** entre dos árboles cualesquiera en el bosque. El aumento de la correlación aumenta la tasa de error forestal.

- La **fuerza** de cada árbol individual del bosque. Un árbol con una tasa de error baja es un clasificador sólido. El aumento de la fuerza de los árboles individuales disminuye la tasa de error forestal.

Reducir *m* disminuye tanto la correlación como la fuerza. Incrementarlo aumenta ambos. En algún punto intermedio hay un rango "óptimo" de *m*, generalmente bastante amplio. Utilizando la tasa de error de oob[5] se puede encontrar rápidamente un valor de m en el rango. Este es el único parámetro ajustable al que los bosques aleatorios son algo sensibles.

Las características del algoritmo de aprendizaje supervisado *Random Forest* son (Cutler & Breiman, s.f.):

- Su precisión es insuperable entre los algoritmos actuales.

- Funciona de manera eficiente en grandes bases de datos.

- Puede manejar miles de variables de entrada sin eliminarlas.

- Da estimaciones de que variables son importantes en la clasificación.

[5] Out of bag error estimate

- Genera una estimación interna no sesgada del error de generalización a medida que avanza la construcción del bosque.

- Tiene un método eficaz para estimar datos faltantes y mantiene la precisión cuando falta una gran proporción de los datos.

- Tiene métodos para equilibrar el error en conjuntos de datos no equilibrados de población de clases.

- Los bosques generados se pueden guardar para uso futuro entre otros datos.

- Se calculan prototipos que dan información sobre la relación entre variables y la clasificación.

- Calcula las proximidades entre pares de casos que se pueden utilizar para agrupar, localizar valores atípicos o (mediante escalado) proporcionar vistas interesantes de los datos.

- Las capacidades de lo anterior pueden extenderse a datos no etiquetados, lo que lleva a agrupaciones en clústeres no supervisadas, vista de datos y detección de valores atípicos.

- Ofrece un método experimental para detectar interacciones variables.

Veamos a continuación un ejemplo. En la Figura 9 se muestra un árbol sobre si se debe de jugar al tenis. Si el clima esta nublado, entonces “Si” debemos de jugar al tenis. Si el clima es soleado y la humedad es alta entonces “No” debemos jugar al tenis (M.Mitchell, 1997).

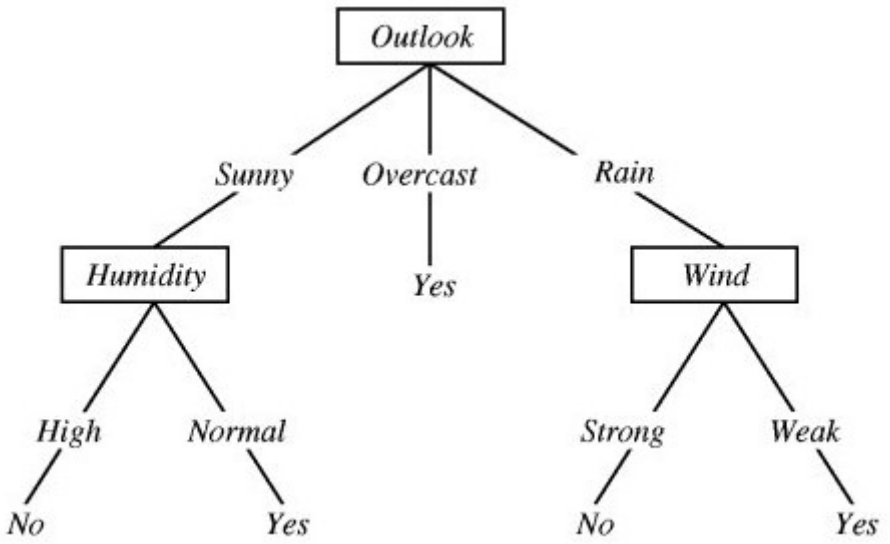


Figura 9. Random Forest para el concepto Jugar al Tenis (M.Mitchell, 1997).

3.1.2 Árboles de decisión ID3

El investigador J. Ross Quinlan desarrolló el algoritmo conocido como ID3 (Induction Decision Trees). Según Ross J. Quinlan en (Quinlan, 1986) el algoritmo ID3 se diseñó donde hay muchos atributos y el conjunto de entrenamiento contiene muchos objetos, pero donde se requiere un árbol de decisiones razonablemente bueno sin mucho cálculo. Se sabe que construye árboles de decisiones simples, pero el enfoque que utiliza no puede garantizar que no se hayan pasado por alto mejores árboles. La estructrura básica del ID3 es iterativa. Un subconjunto del conjunto de entrenamiento llamado ventana se elige al azar y se forma un árbol de decisión a partir de él. Este árbol clasifica correctamente todos los objetos de la ventana. Todos los demás objetos del conjunto de entrenamiento se clasifican mediante el árbol. Si el árbol da la respuesta correcta para todos estos objetos, entonces es correcto para todo el conjunto de entrenamiento y el proceso termina. De lo contrario, se agrega a la ventana una selección de los objetos clasificados incorrectamente y el proceso continua.

De esta manera se han encontrado árboles de decisión correctos después de solo unas pocas interacciones para conjuntos de entrenamiento hasta treinta mil objetos descritos en términos de hasta 50 atributos. La evidencia empírica sugiere que un árbol de decisiones correcto generalmente se encuentra más rápidamente con este método iterativo que formado un árbol directamente a partir de todo el conjunto de entrenamiento.

El algoritmo ID3 construye lo que llamamos los árboles *top-down* (de arriba a abajo), utilizando un método de selección de atributos basado en la teoría de la información. Este algoritmo se considera como heurística que el atributo cuyo conocimiento aporta más información en la clasificación es el más útil.

En este algoritmo el atributo mejor clasificado se convierte en la condición del nodo raíz que da lugar a distintas ramas, una por cada valor posible del atributo. Nunca da marcha atrás para reconsiderar decisiones previas. Hay que considerar que este algoritmo utiliza la ganancia de información para seleccionar en cada paso según se va generando el árbol que aquel atributo.

Las características principales del algoritmo ID3 son las siguientes (López Takeyas, 2013):

- Pertenece a la familia TDIDT (*Top-Down Induction of Decision Trees*).

- Su objetivo es construir un árbol de decisión que explique cada instancia de la secuencia de entrada de la manera más compacta posible. Utilizando en cada momento el mejor atributo dependiendo de una determinada heurística.

- El inconveniente es que favorece indirectamente aquellos atributos con muchos valores, los cuales no tienen que ser los más útiles.

- Genera árboles de decisión a partir de ejemplos de partida.

- Intenta encontrar el árbol más sencillo que separa mejor los ejemplos.

- Recursividad.

- No se realiza "*Backtracking*"[6].

- Utiliza la entropía[7].

[6] Vuelta atrás.
[7] Medida de la incertidumbre que hay en un sistema. Ante una determinada situación, la probabilidad de que ocurra cada uno de los posibles resultados.

Veamos a continuación un ejemplo. En la Figura 10 se describe un mini-dominio compuesto por los datos de 8 personas, correspondientes a su altura, color de cabello y color de ojos distribuidas en dos clases, *C*+ y *C*-, y se busca el mejor árbol de decisión que lo caracteriza.

Clase	Elemento	Altura	Cabello	Ojos
C^+	1	bajo	rubio	azules
	2	alto	pelirrojo	azules
	3	alto	rubio	azules
C^-	4	alto	rubio	marrones
	5	bajo	castaño	azules
	6	alto	castaño	azules
	7	alto	castaño	marrones
	8	bajo	rubio	marrones

Figura 10. Ejemplo Tabla Algoritmo ID3 (Moreno, y otros, 1994).

Así pues, $C = \{C+ ; C\}$, $X = \{1; 2; 3; 4; 5; 7; 8\}$ y su partición en las dos clases existentes sería $Pc\ (X) = \{\{1, 2, 3\}, \{4, 5, 7, 8\}\}$. Por consiguiente,

$$I(P_{\mathcal{C}}(\mathcal{X})) = -3/8log_2\ 3/8 - 5/8log_2\ 5/8 = 0.954$$

Analicemos ahora los atributos:

$$E(\mathcal{X}, \texttt{Altura}) = 3/8I(P_{\mathcal{C}}(\{1, 5, 8\}) + 5/8I(P_{\mathcal{C}}(\{2, 3, 4, 6, 7\}) = 0.951$$

Con

$$I(P_{\mathcal{C}}(\{1, 5, 8\})) = -1/3log_2\ 1/3 - 2/3\ log_2 2/3 = 0.918$$
$$I(P_{\mathcal{C}}(\{2, 3, 4, 6, 7\})) = -2/5log_2\ 2/5 - 3/5log_2\ 3/5 = 0.971$$

Finalmente, la ganancia generada por **Altura** sería:

$$G(\mathcal{X}, \texttt{Altura}) = 0.954 - 0.951 = 0.003$$

Similarmente,

$$E(\mathcal{X}, \texttt{Cabello}) = 0.454$$
$$E(\mathcal{X}, \texttt{Ojos}) = 0.347$$

Por tanto, se elegirá atributo raíz Cabello. El proceso continuaría ahora para generar los 3 subárboles correspondientes a los 3 valores de Cabello, utilizando para ello los conjuntos de instancias A^{-1}(X, castaño), A^{-1}(X, pelirrojo) y A^{-1}(X, rubio), respectivamente. El proceso completo se puede observar en las Figura 11 y la Figura 12 (Moreno, y otros, 1994).

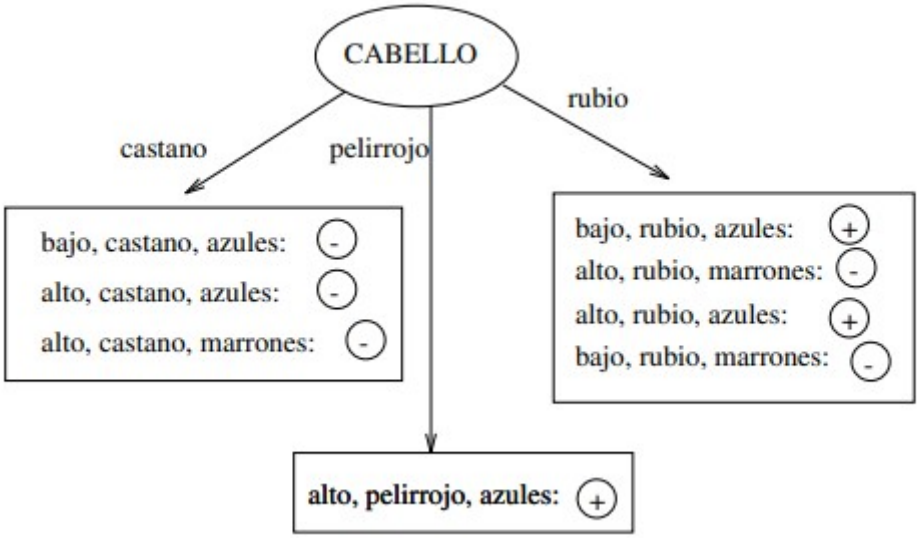


Figura 11. Paso según los cálculos del texto (Moreno, y otros, 1994).

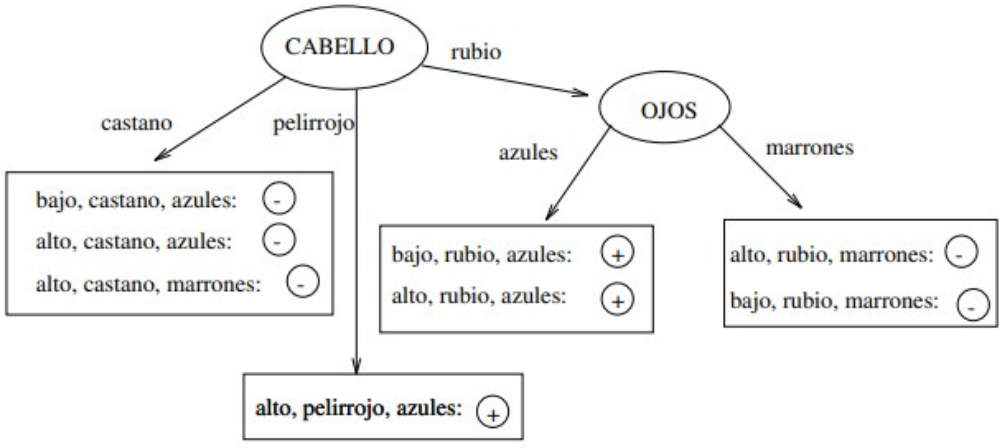


Figura 12. Árbol de decisión generado por ID3 (Moreno, y otros, 1994).

3.1.3 Algoritmo J48

El algoritmo J48 está basado en el algoritmo C4.5 de J.R Quinlan. Es una implementación del algoritmo C4.5, además de ser uno de los algoritmos más utilizado actualmente y es conocido en el sistema Weka por J48.

El algoritmo C4.5 es un método de inducción de árboles de decisión basado en ID3, el cual acaba con muchas limitaciones del ID3. Permite trabajar con valores continuos para los atributos, separando los posibles resultados en dos ramas en función de un umbral. Los árboles son menos frondosos porque cada hoja no cubre una clase en particular sino una distribución de clases.

El C4.5 genera un árbol de decisión a partir de los datos mediante particiones realizadas recursivamente, según la estrategia de profundidad-primero (*Depth-First*). Antes de cada partición de datos, el algoritmo considera todas las pruebas posibles que pueden dividir el conjunto de datos y selecciona la prueba que resulta en la mayor ganancia de información o en la mayor proporción de ganancia de información. Para cada atributo discreto, se considera una prueba con *n* resultados, siendo *n* el número de valores posibles que se puede tomar el atributo. Para cada atributo continuo se realiza una prueba binaria sobre cada uno de los valores que toma el atributo en los datos.
El algoritmo J48 tiene estas características:

- Puede procesar datos categóricos.

- No puede ser actualizado de forma incrementar (es decir, añadir nuevos datos sin reclasificar a los anteriores).

- Permite en Weka utilizar un árbol podado o no podado, se puede impedir el aumento de los subárboles, lo que desemboca en algoritmos más eficientes.

- Permite fijar el umbral de confianza para el proceso de poda y el número mínimo de instancias permitido en cada hoja.

- Se permite una opción que disminuye el error de poda, realizándose una poda del árbol de decisión que del árbol de decisión que optimiza el rendimiento en un conjunto fijo. Se puede fijar el tamaño de este grupo, el conjunto de datos se divide por igual en el número de grupos fijado y la última parte se usa como conjunto fijo.

- Puede manejar instancias ponderadas.

- Permite la construcción de árboles binarios

A continuación veamos un ejemplo. La estructura del árbol (ver Figura 13) está compuesta por dos tipos de nodos (Vizcaino Garzon, 2008):

- Una hoja (nodo terminal), que indica una clase.
- Un nodo de decisión, que especifica una comprobación a realizar sobre el valor de una variable. Tiene una rama y un subárbol para cada resultado posible de la comprobación.

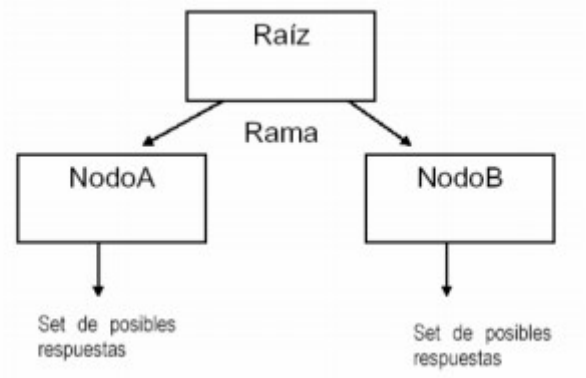


Figura 13. Ejemplo aplicado árbol de decisión adaptado para J48 (Vizcaino Garzon, 2008).

3.1.4 Reglas de clasificación PRISM

El algoritmo PRISM (Cendrowska, 1987) es uno de los algoritmos más simples de recubrimiento secuencial. Se describe el algoritmo PRISM, que, aunque está basado en ID3, utiliza una estrategia de inducción diferente para inducir reglas que son modulares, evitando así muchos problemas relacionados con los árboles de decisión.

PRISM toma como entrada un conjunto de entrenamiento ingresado como un archivo de conjuntos ordenados de valores de atributos y, cada conjunto termina con una clasificación. El conjunto de reglas para este algoritmo sería si el conjunto de entrenamiento contiene instancias de más de una clasificación, entonces para cada clasificación δ_n se realizan los siguientes pasos (Cendrowska, 1987):

- Paso 1: calcular la probabilidad de ocurrencia, $p(\delta_n \mid \alpha_x)$, de la clasificación δ_n para cada par atributo -valor α_x,

- Paso 2: seleccionar α_x para cual $p(\delta_n \mid \alpha_x)$ es un máximo y crea un subconjunto del conjunto de entrenamiento que comprende todas las instancias que contiene el seleccionado α_x,

- Paso 3: repetir los pasos 1 y 2 para este subconjunto hasta que solo contenga instancias de la clase δ_n. La regla inducida es una conjunción de todos los pares atributo-valor utilizados para crear el subconjunto homogéneo.

- Paso 4: eliminar todas las instancias cubiertas por esta regla del conjunto de entrenamiento.

- Paso 5: repetir los pasos del 1 al 4 hasta que se hayan eliminado todas las instancias de la clase δ_n .

Cuando se han inducido las reglas para una clasificación, el conjunto de entrenamiento se restablece a su estado inicial y el algoritmo se aplica nuevamente para inducir un conjunto de reglas que cubren la siguiente clasificación. Aunque el algoritmo de inducción básico utilizado por PRISM se basa en técnicas empleadas por ID3, es bastante diferente en muchos aspectos. La principal diferencia es que PRISM se concentra en encontrar solo valores relevantes de atributos, mientras que ID3 se ocupa de encontrar el atributo que es más relevante en general, aunque algunos valores de ese atributo pueden ser irrelevantes. ID3 divide un conjunto de entrenamiento en subconjuntos homogéneos sin referencia a la clase de este subconjunto, mientras PRISM debe identificar subconjuntos de una clase específica. Esto

tiene la desventaja de un esfuerzo computacional mayor, pero la ventaja de una salida en forma de reglas modulares en lugar de un árbol de decisiones.

Si se hace una elección incorrecta en PRISM, entonces el resultado es que un par atributo-valor irrelevante puede ser elegidos. Esta característica afortunadamente se puede evitar incorporando algunas heurísticas en el algoritmo básico.

PRISM produce sus resultados como un conjunto de reglas modulares que son máximamente generales cuando el conjunto de entrenamiento es completo. La precisión de las reglas inducidas por un conjunto de entrenamiento incompleto depende del tamaño de ese conjunto de entrenamiento (como con todos los algoritmos de inducción) pero es comparable a la precisión de un árbol de decisiones inducido por ID3 del mismo conjunto de entrenamiento, a pesar de la gran reducción en el número y extensión de las reglas. Este algoritmo tiene la característica de eliminar los ejemplos que va cubriendo por las reglas conformadas, por lo cual las reglas deben mostrarse e interpretarse en el orden que se van cubriendo.

```
PRISM (ejemplos) {
   Para cada clase (C)
     E = ejemplos
     Mientras E tenga ejemplos de C
        Crea una regla R con parte izquierda vacía y clase C
        Hasta R perfecta Hacer
           Para cada atributo A no incluido en R y cada valor v de A
             Considera añadir la condición A=v a la parte izquierda de R
             Selecciona el par A=v que maximice p/t
                (en caso de empates, escoge la que tenga p mayor)
           Añadir A=v a R
     Elimina de E los ejemplos cubiertos por R
```

Figura 14. Pseudocódigo del Algoritmo de PRISM (Robles Aranda & R. Sotolongo, 2013).

El algoritmo de PRISM tiene estas características:

- Crea reglas que cubren mayor parte de observaciones separando instancias para analizarlas por separado.

- Ventajas al no considerar el ruido en los datos.

- Se elije la condición que maximiza la precisión de la regla.

- Cualquier regla con precisión menor que el 100% es incorrecta.

- En PRISM se continúa asignando reglas hasta que su precisión es del 100%.

- PRISM añade condiciones a reglas que maximicen la relación *p/t* (la relación entre los ejemplos positivos cubiertos y los ejemplos cubiertos en total).

Veamos a continuación un ejemplo. Considerando los datos de la Figura 15, en la Figura 16, si empezamos con la Clase *P,* construimos todas las posibles combinaciones de atributo valor y evaluamos su predicción sobre la clase *P*.

A_1	A_2	A_3	A_4	Clase
1	x	△	a	P
0	x	◯	a	N
1	y	□	a	P
1	y	△	b	P
1	x	□	b	N
0	y	◯	a	P
0	x	△	b	N
1	y	◯	a	P

Figura 15. Datos para predicción algoritmo PRISM (Morales & Escalante).

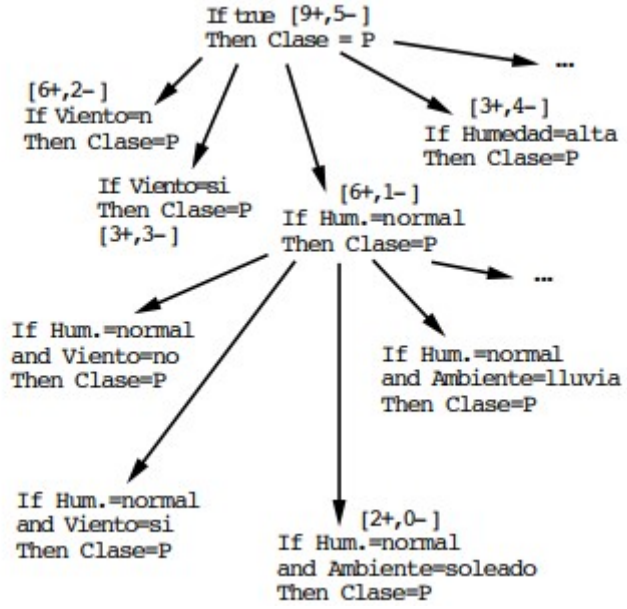


Figura 16. Resultado sobre clase P (Morales & Escalante).

3.1.5 Algoritmo A Priori

El algoritmo A Priori (Agrawal, Imielinski, & Swami, 1993) es uno de los más utilizados y conocidos dentro de los que se consideran algoritmos para generar reglas de asociación.

Este algoritmo está diseñado para intervenir en bases de datos que contiene transacciones. Se basa en el conocimiento previo o "a priori" de los conjuntos de datos que aparecen con mayor frecuencia, esto sirve para reducir el espacio de búsqueda y aumentar la eficiencia.

El algoritmo A Priori pretende generar *ítem–sets* que cumplan una *cobertura* mínima de manera eficiente. Un ítem es un par atributo-valor mientras que un ítem-set es un conjunto de pares atributo-valor. Un k-ítem-set es un conjunto de "k" pares atributo-valor. La cobertura de un ítem-sets se refiere al número de instancias que cumplen los valores en el *ítem-set* y va a determinar la cobertura de las reglas generadas a partir de dicho *ítem-set*.

El proceso del algoritmo se resume en dos pasos:

1. Generación de *ítems-set*.

2. Generación de reglas a partir de los *ítems-sets* generados en la fase 1.

Las características del algoritmo A Priori son:

- Encuentra aquellas asociaciones con mayor frecuencia.

- Su método es simple.

- Es intuitivo.

- Hay que especificar las reglas.

- Se establecen valores mínimos.

- Itera sobre bases de datos.

Veamos un ejemplo a continuación. Considerando el conjunto de datos tamaño 1 que aparece en la Figura 17 y sabiendo que el soporte mínimo es 0.6, se obtienen los siguientes conjuntos de ítems frecuentes: {a},{b} y {d}. A partir de estos conjuntos, se obtienen los conjuntos candidatos de tamaño 2, los cuales se establecen a través de la combinación de los conjuntos frecuentes de tamaño 1. En la Figura 18 se muestran los conjuntos candidatos de tamaño 2 (Pinho Lucas, 2010).

Conjunto de ítems	Num. de transacciones	Soporte
{a}	7	0,7
{b}	8	0,8
{c}	5	0,5
{d}	7	0,7
{e}	2	0,2

Figura 17. Conjunto de datos de tamaño 1 (Pinho Lucas, 2010).

Conjunto de ítems	Num. de transacciones	Soporte
{a, d}	4	0,6
{b, d}	4	0,6
{a, b}	4	0,6

Figura 18. Conjunto de datos tamaño 2 (Pinho Lucas, 2010).

Los conjuntos de ítems frecuentes obtenidos serían los siguientes: {b,d}, {a,b} y {a,d}. Partiendo de dichos conjuntos, se obtiene solamente un conjunto de tamaño 3, el cual se muestra en la Figura 19.

Conjunto de ítems	Num. de transacciones	Soporte
{a, b, d}	4	0,4

Figura 19. Conjunto de datos tamaño 3 (Pinho Lucas, 2010).

Una vez obtenido el conjunto de tamaño 3, no sería posible obtener conjuntos de ítems frecuentes de tamaño 4, por lo que la obtención de ítems frecuentes finalizaría.

3.2 Algoritmos de aprendizaje automático no supervisado

3.2.1 Algoritmo K-Means

El algoritmo K-Means es uno de los algoritmos más utilizados de tipo *clustering* y una técnica clásica hoy en día. Este tipo de algoritmo especifica de antemano cuantos clústeres se buscan; éstos vendrían representados por el parámetro "*k*". Los "*k*" puntos se eligen al azar como centros de clúster. Todas las instancias se asignan a su centro de clúster más cercano de acuerdo con la métrica de distancia euclidiana. A continuación, se calcula el centroide o la media de las instancias de cada grupo (esta es la parte de "Medias-Means"). Estos centroides se toman como nuevos valores centrales para sus respectivos grupos. Por último, todo el proceso se repite con los nuevos centros de clúster. Este proceso iterativo continúa hasta que se asignan los mismos puntos a cada grupo en una determinada iteración, en cuyo paso los centros del grupo se han estabilizado y seguirá siendo los mismos para siempre (Witten, Frank, & Hall, 2011).

El algoritmo K-Means resuelve un problema de optimización. Se busca minimizar la suma de las distancias de cada objeto al centroide de su clúster.

Las características del algoritmo de agrupamiento K-Means son:

- Se utiliza la distancia euclídea, dando buenos resultados.
- Tiene buena escalabilidad con la cantidad de datos.
- Agrupa los datos.
- Es simple y efectivo.
- Es un método iterativo.

Un ejemplo es el siguiente, relativo a la segmentación de imágenes en color. Suponiendo que la imagen original utiliza una paleta de 255 colores, podemos decidir usar menos bits por

píxel, es decir, usar menos colores. En la Figura 20, se puede observar cómo quedaría la foto utilizando menos colores. Empezando por *k*=2, seguido de *k*=3 y *k*=10.

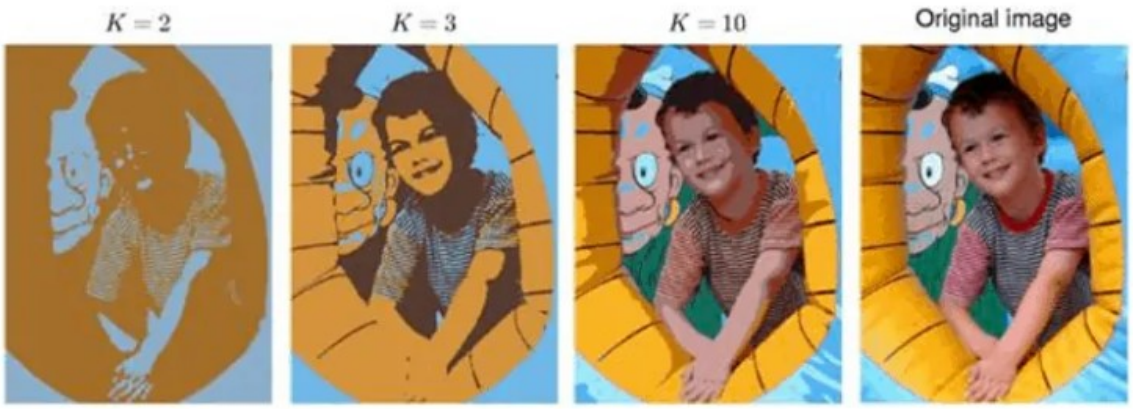


Figura 20. Imágenes con diferentes Ks para el algoritmo K-Means (Ma, 2016).

3.2.2 Algoritmo Cobweb

El algoritmo Cobweb es un algoritmo de *clustering* jerárquico, que se caracteriza porque utiliza aprendizaje incremental. Este consiste en realizar las agrupaciones instancia a instancia. Durante su ejecución se forma un árbol de clasificación, donde las hojas representan los segmentos y el nodo raíz engloba por completo el conjunto de datos de entrada. A priori, el árbol consiste en un único nodo raíz. Las instancias se van añadiendo una a una y el árbol se va actualizando en cada paso. La actualización consiste en encontrar el mejor sitio donde incluir la nueva instancia, la operación que puede necesitar de la reestructuración de todo el árbol o solamente la inclusión de la instancia en un nodo que ya existía (Garre, Cuadrado & Sicilia).

Para saber cómo y dónde se debe actualizar el árbol, se proporciona la medida denominada utilidad de categoría, que mide la calidad general de una partición de instancias en un segmento. Este algoritmo es muy sensible a dos parámetros:

- **Acuity:** se basa en la estimación de la media y la desviación estándar del valor de los (González & Pérez, 2006) atributos. Representa la media de error de un nodo con una sola instancia, es decir, establece la varianza mínima de un atributo.

- **Cut-off:** se utiliza para evitar el crecimiento desmesurado del número de segmentos. Este indica el grado de mejoría que se debe producir en la utilidad de categoría para que la instancia sea tenida en cuenta de manera individual.

A este algoritmo no hay que darle el número exacto de clústeres a priori, sino que, en base a los parámetros mencionados con anterioridad encuentra el número óptimo.

Las características del algoritmo Cobweb son:

- Pertenece a los métodos de aprendizaje conceptual o basados en modelos.
- No hay que proporcionar número exacto de clústeres.
- Utiliza aprendizaje incremental.
- Tiene un comportamiento bidireccional.
- Permite una reorganización estructural.

A continuación, se presenta un ejemplo. En la construcción del árbol (ver Figura 21), incrementalmente se incorpora cada ejemplo al mismo, donde cada nodo es un concepto probabilístico que representa una clase de objetos. Cobweb desciende por el árbol buscando el mejor lugar o nodo para cada ejemplo. Esto se basa en medir en cual se tiene la mayor ganancia de utilidad de categoría (Molina López & García Herrero, 2004).

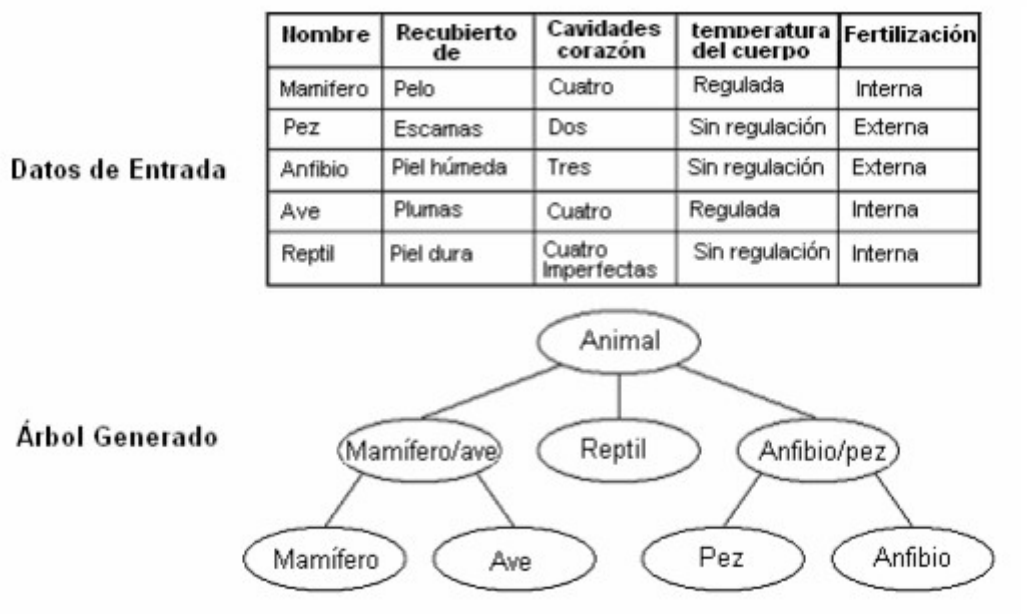


Figura 21. Ejemplo algoritmo Cobweb (Molina López & García Herrero, 2004).

3.2.3 Algoritmo EM

El algoritmo EM (*Expectation Maximitation*) pertenece a los llamados "Finite Mixture Models"; mezclas finitas. Estos se pueden utilizar para segmentar conjuntos de datos además de indicar la probabilidad de aquellas instancias pertenezcan a cada uno de los clústeres.

Principalmente, este algoritmo se basa en asociar a un problema de datos incompletos un problema de datos completados en el que la estimación por Máxima Verosimilitud sea operable. Se establece relación entre las verosimilitudes de estos dos problemas con la finalidad de que la estimación vía Máxima Verosimilitud sea más simple en cada iteración. Este algoritmo se utiliza en estadística ya que la verosimilitud del conjunto de datos completo tiene una distribución fácil de operar y conocida, lo que hace que el algoritmo EM sea eficiente (Pavon, 2014).

Consta de dos etapas:

- **Expectation (Esperanza):** estimar el conjunto de datos utilizando el subconjunto de datos observados y los parámetros correspondientes, ya que los datos faltantes son reemplazados por su esperanza dados los datos conocidos.

- **Maximization (Maximización):** se maximiza la función de verosimilitud con los datos del paso anterior y de esta forma obtener un nuevo conjunto de parámetros que serán utilizados para actualizar la estimación de la esperanza dada de los datos desconocidos en la siguiente iteración.

Hasta que la verosimilitud converja, el paso Expectation y Maximization se repiten iterativamente.

Las características del algoritmo EM son:

- Es simple de utilizar.

- Presenta el problema de no poder converger a un óptimo global y si a un óptimo local.

- Resulta ser efectivo frente a otros métodos de clustering.

- Puede tener un alto coste computacional si lo que hay que modelar es alto.

Por ejemplo, la Figura 22 muestra un ejemplo de algoritmo EM sobre el lanzamiento de una moneda. Considere un simple experimento de lanzamiento de monedas en el que se nos da un par de monedas A y B de sesgos desconocidos, θA y θB respectivamente, es decir, en cualquier lanzamiento, la moneda A aterrizará en la cara con probabilidad θA y cruz con probabilidad 1- θA y de manera similar para la moneda B. Nuestro objetivo es estimar θ = (θA,θB) repitiendo el siguiente procedimiento diez veces: Elige al azar una de las dos monedas (con igual probabilidad) y realiza diez lanzamientos de moneda independientes con la moneda seleccionada. Por lo tanto, todo el procedimiento implica un total de 50 lanzamientos de moneda (B Do & Batzoglou, 2008).

Figura 22. Estimación Máxima Verosimilitud (B Do & Batzoglou, 2008).

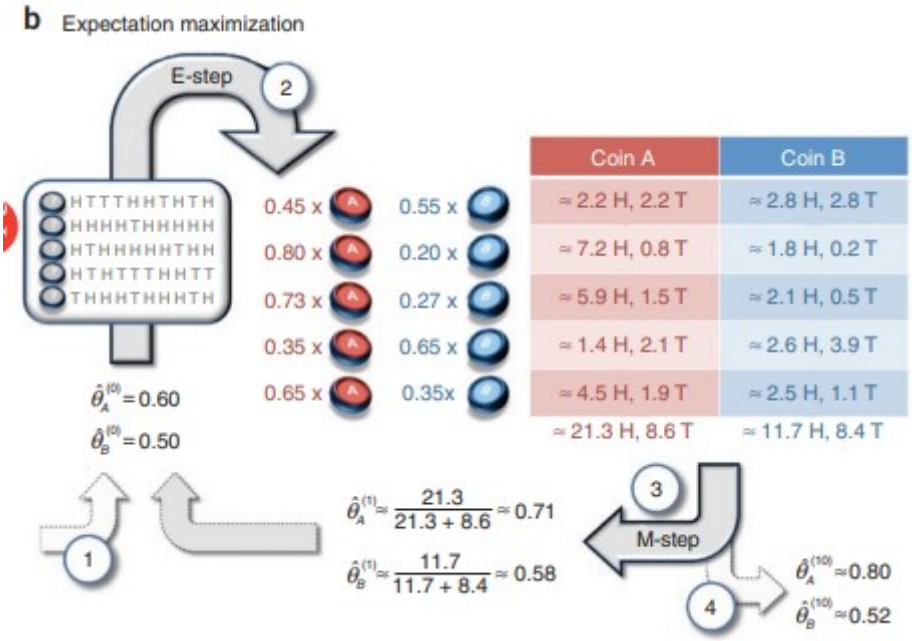


Figura 23. Expectación Maximización (B Do & Batzoglou, 2008).

3.2.4 Algoritmo Fuzzy C-Means

El algoritmo Fuzzy C-Means es un método de agrupamiento o *clustering* de partición difusa más difundido. Su idea es obtener particiones difusas a partir de un conjunto de datos, además el algoritmo Fuzzy C-Means modela los clústeres con forma circular.

Fuzzy C-Means se inspira en el agrupamiento de k-means en donde los elementos de datos pueden pertenecer a más de un grupo. Cada dato tiene asociado un conjunto de niveles de pertenencia a cada grupo. En estos niveles se indican la fuerza de asociación entre un dato

particular a uno o más grupos. Este agrupamiento difuso es un proceso de asignación de estos niveles de pertenencia para luego usarlos para asignar un dato a uno o más grupos. Para comprobar los grupos obtenidos se hace una validación por medio de índices de desempeño. Estos índices valoran si la partición de los grupos encontrada es la mejor para los datos asignados a cada grupo (Villazana, Arteaga, Seijas, & Rodríguez, 2012).

Dado un conjunto de objetos x_1, x_2,..x_i,..x_n, un agrupamiento mediante k clústeres difusos C1, C2, … C_j… C_K se puede representar utilizando una matriz divisoria, M = [p_{ij}], tal que $1 \leq i \leq n$ y $1 \leq j \leq k$, siendo p_{ij} el grado de pertenencia del objeto xi al clúster C_j. El algoritmo Fuzzy C-Means se basa en minimizar la siguiente función objetivo que aparece a continuación:

$$f_m = \sum_{i=1}^{n} \sum_{j=1}^{K} p_{ij}^{m} \|x_i - c_j\|^2$$

Tal que:

1. m es un número real mayor o igual a 1 que gobierna la influencia de los grados de pertenencia.

2. P_{ij} es el grado de pertenencia del objeto x_i al clúster C_j incluido en la matriz divisoria antes mencionada.

3. x_i es la i-ésima instancia.

4. c_j es el centro del clúster C_j que se puede calcular como la media, el centroide o cualquier otra fórmula adecuada al problema en concreto.

5. || || se refiere a cualquier medida relativa a la similitud entre la instancia y el centro del clúster.

Los pasos que sigue son:

1. Inicializar la matriz divisoria de manera aleatoria, pero cumpliendo que:

a. Cada valor asignado p_{ij} está entre 0 y 1.

b. Para cada objeto $x_i \sum_j^K = 1 \; p_{ij} = 1$

2. En la iteración t, calcular los centros de los clústeres en base a la siguiente expresión:

$$c_j = \frac{\sum_{i=1}^{n} p_{ij}^m x_i}{\sum_{i=1}^{n} p_{ij}^m}$$

3. Actualizar la matriz $M^{(t)}$ obteniendo $M^{(t+1)}$ con la siguiente fórmula:

$$p_{ij} = \frac{1}{\sum_{l=1}^{k} \frac{\|x_i - c_j\|^{\frac{2}{m-1}}}{\|x_i - c_l\|}}$$

4. Finalizar si se alcanza el criterio de parada ya que puede establecerse en un número determinado de iteraciones o en que la variacion de la matriz sea demasiado pequeña en la iteración tal que $\| M^{(t+1)} - M^{(t)} \| < \varepsilon$. Si no volver al paso 2.

Hay que tener en cuenta que Weka no tiene la opción para implementar el algoritmo Fuzzy C-Means. Se podría elegir el algoritmo EM o K-Means con el que tiene similitudes. Por otra parte, con Orange también pasaría lo mismo que con Weka, tendría que elegir aplicar el algoritmo K-Means.

Las características del algoritmo Fuzzy C-Means son las siguientes:

- Cada observación puede pertenecer a varios clústeres.
- Se asemeja a K-Means.
- Es iterativo.
- Se utiliza en la segmentación de imágenes.

3.2.5 Agrupamiento Jerárquico

El agrupamiento jerárquico es un método que busca construir una jerarquía de grupos, es decir, produce representaciones jerárquicas en la que los grupos en cada nivel de la jerarquía se crean fusionando grupos en el siguiente nivel inferior. En el nivel más bajo, cada grupo contiene una única observación. En el nivel más alto solo hay un grupo que contiene todos los datos. Las estrategias para la agrupación jerárquica se dividen en dos paradigmas o algoritmos básicos: *algoritmos aglomerativos* (de abajo hacia arriba) y *algoritmos divisorios* (de arriba hacia abajo). A continuación se explica en qué consiste cada uno de ellos:

- **Algoritmos aglomerativos:** estos comienzan de abajo hacia arriba, en cada nivel fusionan recursivamente un par seleccionado de grupos en un solo grupo. Esto produce una agrupación en el siguiente nivel superior con un grupo menos. El par elegido para la fusión consta de dos grupos con la menor disimilitud[8] intergrupal.

- **Algoritmos divisorios:** estos comienzan arriba hacia abajo y en cada nivel dividen de forma recursiva uno de los grupos existentes en ese nivel en dos grupos nuevos. La división se elige para producir dos nuevos grupos con la mayor disimilitud entre grupos.

Con ambos algoritmos hay N-1 niveles en la jerarquía. Cada nivel de la jerarquía representa una agrupación particular de los datos en grupos de observaciones disjuntos. Toda la jerarquía representa una secuencia ordenada de tales agrupaciones. Depende del usuario decidir qué nivel (si lo hay) realmente representa un agrupamiento "natural" en el sentido de que las observaciones dentro de cada uno de sus grupos son suficientemente más similares entre sí que las observaciones asignadas a diferentes grupos en ese nivel. Los nodos de los árboles representan grupos. El nodo raíz representa el conjunto de datos completo. Los N nodos terminales representan cada una de las observaciones individuales. Este tipo de visualización gráfica se llama dendograma (Hastie, Tibshirani, & Friedman, 2008).

[8] Falta de semejanza o de parecido entre dos cosas o más cosas.

Un dendograma proporciona una descripción completa altamente interpretable de la agrupación jerárquica en un formato gráfico. Esta es una de las principales razones de la popularidad de los métodos de agrupación jerárquica. En la Figura 24 se muestra como ejemplo un dendograma de agrupamiento aglomerativo y divisorio.

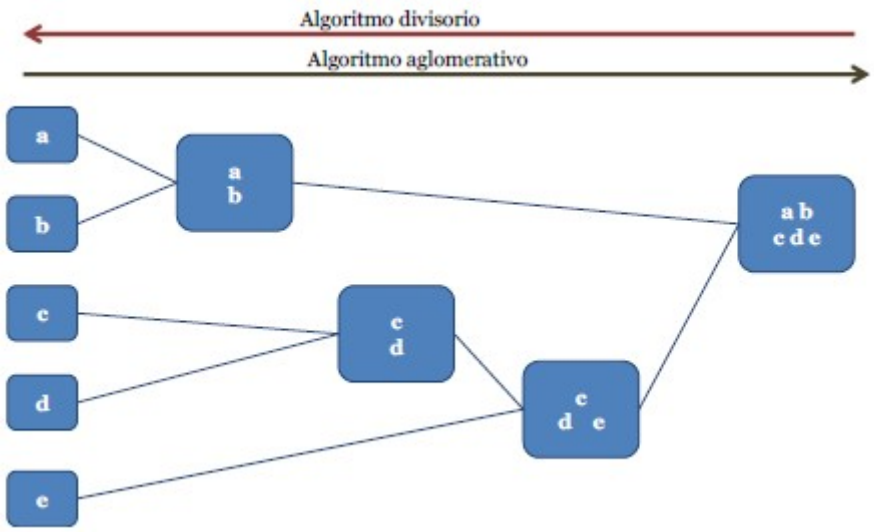


Figura 24. Agrupamiento Jerárquico aglomerativo y divisorio.

Cuando se trabaja con estos algoritmos jerárquicos normalmente se observará que su resultado es el mostrado en la Figura 25. Se observan los diferentes niveles conforme se van construyendo los nodos en cada iteracion.

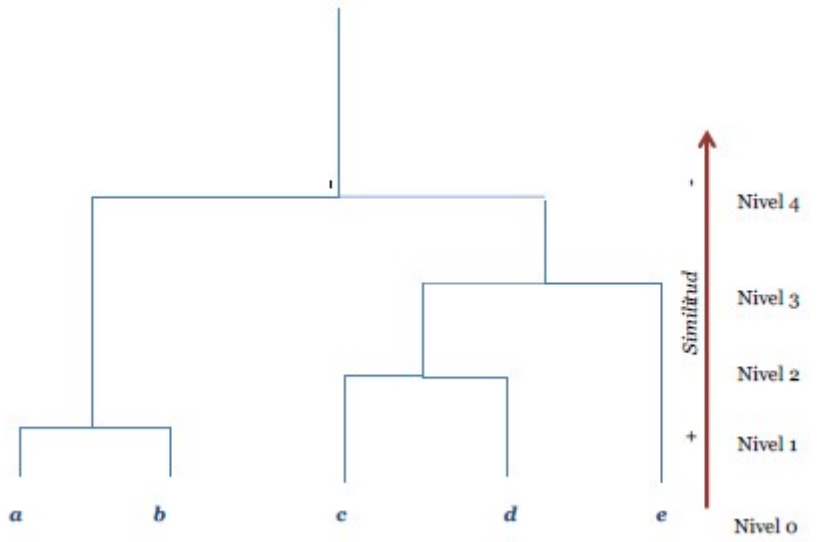


Figura 25. Dendograma referenciando la jerarquía correspondiente a la Figura 26.

A continuación, se describe el funcionamiento general de un método de aglomeración que consiste en: 1) inicializar *n* grupos activos, cada uno con un punto de datos y 2) repetir las siguientes operaciones exactamente *n* – 2 veces (H. Press, A. Teukolsky, T. Vetterling, & P. Flannery, 2007):

1. Encontrar los dos clústeres activos más cercanos por alguna medida de distancia prescrita.

2. Cree un nuevo clúster activo que combine las dos.

3. Conecte el nuevo grupo, como padre, a los dos grupos más cercanos como hijos, con alguna prescripción para las dos longitudes de rama.

4. Elimine a los dos hijos de las lista activa.

5. Calcule las distancias desde el nuevo grupo hasta los grupos activos que quedan. En caso contrario el algoritmo finaliza.

Cada repetición de estos pasos reduce la lista de grupos activos en exactamente uno (una adición, dos eliminaciones), por lo que después de *n-2* repeticiones habrá exactamente dos grupos activos.

Capítulo 3

4. Análisis de las Herramientas

4. 1 Weka

En esta sección se va a explicar, ¿qué es WEKA?

Su nombre Weka (*Gallirallus australis*) se debe a un ave endémica de Nueva Zelanda, de aspecto pardo y tamaño similar a una gallina, se encuentra en peligro de extinción y es famosa por su curiosidad y agresividad.

Este ave da como nombre a una extensa colección de algoritmos de máquinas de conocimiento desarrollados por la Universidad de Waikato (Nueva Zelanda) implementados en Java. Además, Weka contiene las herramientas necesarias para realizar transformaciones sobre los datos, tareas de clasificación, regresión, *clustering*, asociación y visualización. Weka está diseñado como una herramienta orientada a la extensibilidad por lo que añadir nuevas funcionalidades es una tarea sencilla. El objetivo es explicar el funcionamiento básico de esta herramienta y sentar unas bases de su funcionamiento (García Morate, 2006).

Esta herramienta proporciona una interfaz uniforme para muchos algoritmos de aprendizaje diferentes, junto con métodos para el procesamiento previo y posterior y para evaluar el resultado de los esquemas de aprendizaje en cualquier conjunto de datos (Witten, Frank, & Hall, 2011).

4.1.1 Conociendo Weka

Weka es un sistema multiplataforma y de amplio uso probado bajo sistemas operativos Linux, Windows y Macintosh. Puede ser usado desde la perspectiva usuario mediante cinco interfaces que ofrece. Por otra parte, la línea de comando posibilita llamar a cada uno de los algoritmos en la herramienta como programas individuales y mediante la creación de un

programa Java que llame a las funciones que se desee. La versión 3.8.4 dispone de cinco interfaces diferentes que pueden ser accedidas mediante la ventana de selección (*Weka GUI Chooser*). Estas constituyen la interfaz de usuario gráfica GUI[9], tal y como se muestra en la Figura 26. A continuación, se explica brevemente cada una de ellas (Witten, Frank, & Hall, 2011):

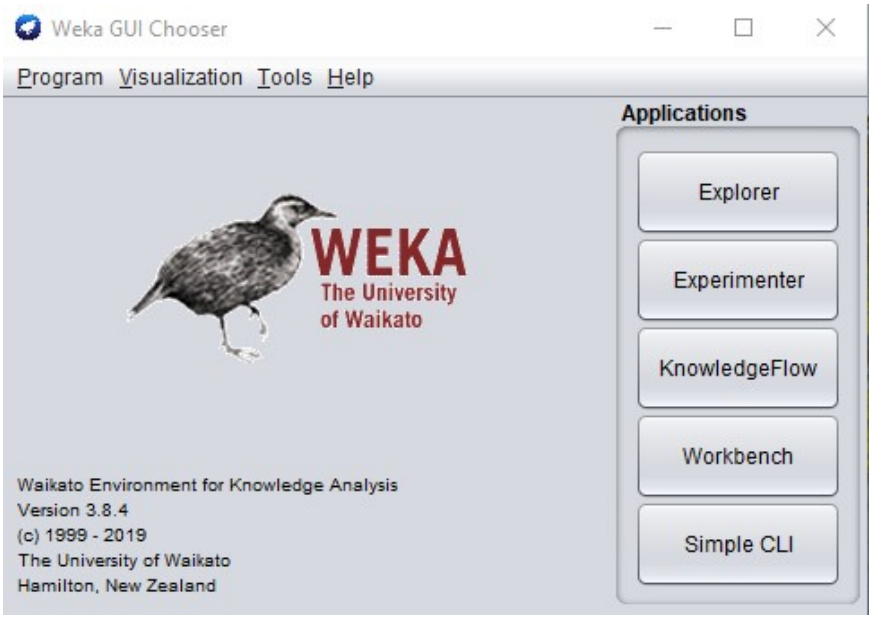


Figura 26. Ventana de selección de interfaces (Weka GUI Chooser).

- **Explorer:** es la forma mas sencilla de utilizar Weka es a través de esta interfaz. Esto le da acceso a todas sus instalaciones mediante la selección de menú y el completado de formularios. Por ejemplo, puede leer rápidamente un conjunto de datos de un archivo ARFF (u hoja de cálculo) y crear un árbol de decisiones a partir de él. Aparecen consejos útiles sobre heramientas cuando el ratón pasa sobre los elementos de la pantalla para explicar lo que hacen. Los valores predeterminados garantizan que pueda obtener resultados con un mínimo de esfuerzo pero tendrá que pensar lo que está haciendo para comprender el significado de los resultados. Weka proporciona un entorno a los usuarios que compara una variedad de técnicas de aprendizaje. Esto se puede hacer de forma interactiva mediante la interfaz *Explorer*.

- **Experimenter:** está diseñada para ayudar a responder una pregunta practica básica al aplicar técnicas de clasificación y regresión: ¿Qué métodos y valores de parámetros

[9] GUI: Grafic User Interface

funcionan mejor para el problema dado? Por lo general, no hay forma de responder a esta pregunta a priori. La interfaz *Experimenter* le permite automatizar el proceso al facilitar clasificadores y filtros con diferentes configuraciones de parámetros en un conjunto de datos, recopilar estadísticas de rendimiento y realizar pruebas significativas. Los usuarios avanzados pueden emplear *Experimenter* para distribuir la carga informática entre varias máquinas utilizando la invocación de método remoto (RMI) de Java. De esta forma, puede configurar experimentos estadísticos a gran escala y dejar que se ejecuten.

- **Knowledge Flow:** permite diseñar configuraciones para el procesamiento de datos transmitidos. Cuando abre un conjunto de datos, lo carga todo inmediatamente. Esta interfaz premite especificar un flujo de datos conectando componentes que representan fuentes de datos, herramientas de preprocesamiento, algoritmos de aprendizaje, métodos de evaluación y módulos de visualización. Si los filtros y los algoritmos de aprendizaje son capaces de un aprendizaje incremental, los datos se cargarán y procesarán de forma incremental.

- **Simple CLI** (*Command Line Interface*): permite invocar desde la línea de comandos cada uno de los algoritmos incluidos en Weka como programas individuales. Los resultados se muestran únicamente en modo texto. A pesar de ser en apariencia muy simple es extremadamente potente ya que, permite realizar y soporta cualquier operación en Weka de forma directa. Sin embargo, es muy complicada de manejar ya que es necesario un conocimiento completo de la aplicación. Su utilidad es pequeña y actualmente es solo útil como una herramienta de ayuda en la fase de pruebas. Es beneficiosa para los sistemas operativos que no proporcionan su propia interfaz para la línea de comandos.

- **Workbench:** Entorno que combina todas las interfaces GUI en una única interfaz. Es muy útil si se está entre dos o más interfaces, es decir, si se empieza a utilizar la interfaz *Explorer* y luego se cambia a *Experimenter*.

4.1.2 Preparación y entrada de datos

La preparación de la entrada para una investigación suele consumir la mayor parte del esfuerzo invertido en todo el proceso de aprendizaje: entre otros, el formato de archivo de entrada, en particular. El formato de archivos de relación de atributos que utiliza Weka y que se describirá más adelante.

Podemos utilizar datos de fuentes abiertas como son Kaggle[10], datos de la Comunidad de Madrid[11], datos del Gobierno de España[12] o datos que nos proporcionen terceras personas para hacer el estudio. De esta forma, se dispondrá de unos datos fiables para poder empezar nuestras pruebas en Weka.

En Weka se denominan a cada uno de los casos ya proporcionados, en el conjunto de datos de entrada, instancias. Cada una de ellas posee unas propiedades que las definen, denominadas atributos en cada conjunto de datos.

El formato con el que trabaja Weka es ARFF (*Attribute Relation File Format*). Este formato se compone por una estructura dividida en tres partes:

- *Encabezado:* se define el nombre de la relación
- *Declaración de atributos:* se definen los atributos a utilizar especificando su tipo
- *Declaración de los datos:* se definen los datos que componen la relación

Por ejemplo, en la Figura 27 se muestra un archivo ARFF para los datos meteorológicos. Las líneas que comienzan con un signo % son comentarios. A continuación de los comentarios, al principio del archivo, se establece el nombre de la relación (clima) y un bloque que define los atributos (panorama, temperatura, humedad, viento, juego). Los atributos nominales van seguidos del conjunto de valores que pueden tener, entre llaves. Los valores pueden incluir

[10] https://www.kaggle.com/datasets
[11] https://datos.comunidad.madrid/dataset
[12] https://datos.gob.es/

espacios. Si es así, deben colocarse entre comillas. Los valores numéricos van seguidos de la palabra clave numérico (Witten, Frank, & Hall, 2011).

```
% ARFF file for the weather data with some numeric features
%
@relation weather

@attribute outlook { sunny, overcast, rainy }
@attribute temperature numeric
@attribute humidity numeric
@attribute windy { true, false }
@attribute play? { yes, no }

@data
%
% 14 instances
%
sunny, 85, 85, false, no
sunny, 80, 90, true, no
overcast, 83, 86, false, yes
rainy, 70, 96, false, yes
rainy, 68, 80, false, yes
rainy, 65, 70, true, no
overcast, 64, 65, true, yes
sunny, 72, 95, false, no
sunny, 69, 70, false, yes
rainy, 75, 80, false, yes
sunny, 75, 70, true, yes
overcast, 72, 90, true, yes
overcast, 81, 75, false, yes
rainy, 71, 91, true, no
```

Figura 27. Fichero ARFF con datos metereológicos (Elaboración propia).

El formato de los ficheros por defecto que utiliza Weka, como ya se ha mencionado, es **ARFF**, pero no es el único que admite. En Weka también son válidos otros formatos como **CSV**[13], archivos que son separados por comas, tabuladores (en la primera línea aparecen los atributos), **C4.5** que son archivos codificados según el formato C4.5 donde los datos estarán agrupados en un fichero *.names* o *.data.* En el primero de los anteriores estarán los nombres de los atributos y en el segundo los datos. Además, también admite los ficheros de instancias **JSON**[14], ficheros de datos **LIBSVM**[15] o también denominada biblioteca de aprendizaje automático, fichero ASCII **Matlab**, **SVM Light Data Files**[16] (implementación de una máquina de vectores para los problemas del reconocimiento de patrones, problemas de regresión y problemas para aprender una función de clasificación), **Instancias Binarias**

[13] Comma-Separated Values
[14] JavaScript Object Notation
[15] A Library for SVM in TypeScript (Lin, s.f.)

Serializadas y ficheros de datos **XRFF**[17]**,** extensión basada en XML. Este último tipo de fichero XRFF tiene dos extensiones *.xrff* y *.xrff.gz,* donde la primera es la extensión predeterminada y la segunda es la extensión para los archivos comprimidos con *gzip*.

Por otra parte, las instancias pueden leerse desde una URL o de una base de datos en SQL usando JDBC.

4.1.3 Manejo de Weka

Los algoritmos de Aprendizaje Automático implementados en Weka permiten realizar tareas como: Preprocesar, Clasificación, Búsqueda de Asociaciones, Selección de Atributos y Visualizar. En la Figura 28 se puede ver la ventana **Explorer**, el entorno básico de Weka, se pueden encontrar algoritmos de aprendizaje automático, de acuerdo a las tareas mencionadas, para su posterior selección. En las ventanas: *Preprocess*, *Classify, Clúster, Associate, Select Attributes* y *Visualize*.

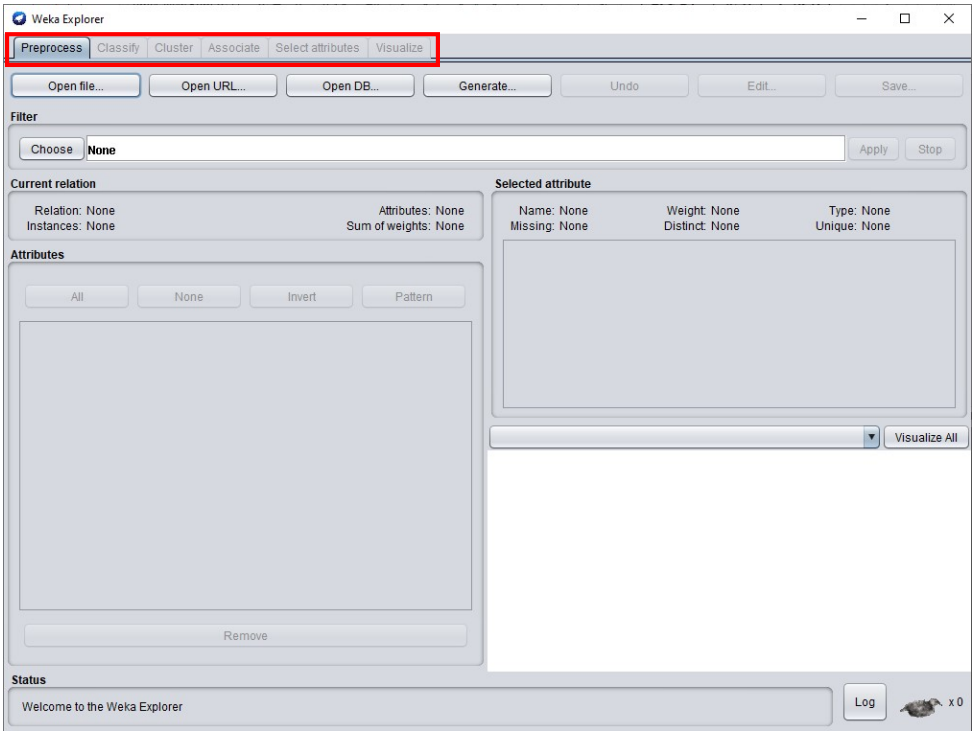


Figura 28. Ventana Explorer en Weka (Elaboración propia).

[16] SVM Light es una implementación de Support Vector Machine en C (Joachims, s.f.)

A continuación, se explican cada una de las ventanas del Explorer en Weka tal y como aparecen en la Figura 28:

- **Preprocess:** esta ventana es la única que se encuentra activa cuando se abre la ventana Explorer. Permite cargar un archivo de datos y, a partir de ella se muestran los atributos, así como la forma gráfica, la relación que existe entre los atributos y la clase. A través de esta interfaz de preprocesamiento se tiene la posibilidad de aplicar cualquiera de los algoritmos de preprocesamiento implementados en Weka, denominados filtros. Estos algoritmos transforman de alguna manera el conjunto de datos de entrada, modificando sus atributos o sus instancias.

- **Classify:** esta ventana pone a disposición del usuario los algoritmos de clasificación implementados, los cuales se agrupan dependiendo la técnica que se emplee. Weka tiene implementado clasificadores basados en redes bayesianas, análisis de regresión, árboles de decisión, basados en reglas, redes neuronales meta-clasificadores, etc. Una vez seleccionado el algoritmo a utilizar será aplicado sobre las instancias del conjunto de datos activo.

- **Clúster:** esta ventana es similar a la de *Classify*. En ella se encuentran los algoritmos de agrupamiento que serán aplicados al conjunto de datos introducidos activo. En esta ventana se pueden observar, en forma gráfica, la asignación de las muestras en grupos. Se pueden aplicar algoritmos como son SimpleKMeans, Cobweb, FilteredClusterer, entre otros.

- **Associate:** en esta ventana se permiten el uso de métodos orientados a la búsqueda de asociaciones entre datos. Es simple de manejar ya que carece de opciones. Estos algoritmos sólo funcionan con datos nominales. Los algoritmos que se encuentran en esta ventana de búsqueda de asociaciones son: A priori, FilteredAssociator y FPGrowth.

- **Select Atributes:** la ventana de selección de atributos tiene como objetivo identificar, mediante el conjunto de datos, cuáles son los atributos que tienen más relevancia a la

[17] eXtensible attribute-Relation File Format

hora de determinar si los datos son de una clase u otra. Algunos de los algoritmos que se encuentran en esta ventana son: CfsSubsetEval, ClassifierAttributeEval, ClassifierSubsetEval, CorrelationAttributeEval, entre otros.

- **Visualize:** en esta última ventana se muestra gráficamente la distribución de todos los atributos mediante gráficas en dos dimensiones, en las que se representan dichos atributos. Se permiten ver correlaciones y asociaciones entre los atributos de forma gráfica.

A través de esta interfaz de preprocesamiento se tiene la posibilidad de aplicar cualquiera de los algoritmos de preprocesamiento implementados en Weka, denominados filtros. Estos algoritmos transforman de alguna manera el conjunto de datos de entrada, modificando sus atributos o sus instancias.

Algunos de los algoritmos de preprocesamiento en Weka son (González & Pérez, 2006):

- **Discretize:** la discretización de un rango de atributos numéricos del conjunto de datos, transformándolos en atributos nominales.

- **Normalize:** normaliza todos los valores numéricos del conjunto de datos dados.

- **ReplaceMissingValues:** reemplaza aquellos valores perdidos del conjunto de datos. Los atributos nominales se remplazan por la moda y los numéricos por la media.

- **NominalToBinary:** convierte aquellos atributos nominales en atributos numéricos binarios

Una vez cargado el archivo con los datos, podemos aplicar los filtros sobre el archivo o bien, pasar a las siguientes secciones y realizar otras tareas. Además, se pueden elegir las opciones del test, es decir, la manera de calcular el porcentaje esperado de aciertos o el error cuadrático medio entre otros. A continuación, se explican estas opciones:

- **Use training set:** Para esta opción se utiliza el mismo conjunto que el de entrenamiento para hacer el test. Esta opción nos dará un resultado o porcentaje optimista. Si se desean resultados más fiables, es conveniente no utilizarlo.

- **Supplied test set:** En este caso se utiliza si tenemos un fichero con datos de test distintos a los de entrenamiento.

- **Cross-validation:** Calcula el porcentaje de aciertos esperado haciendo validación cruzada de *k* particiones (podemos utilizar k=10).

- **Percentage Split:** se dividirá el conjunto de entrenamiento en dos partes: por ejemplo, los primeros 66% de los datos para construir el clasificador y el 33% restante para hacer el test. Es importante tener en cuenta que al utilizar esta opción Weka desordena aleatoriamente el conjunto inicial y después parte en 66% para entrenamiento y 33% para test. Así, si se construyera el clasificador dos veces, se obtendrían dos desordenaciones diferentes y por lo tanto dos porcentajes de aciertos de test diferentes.

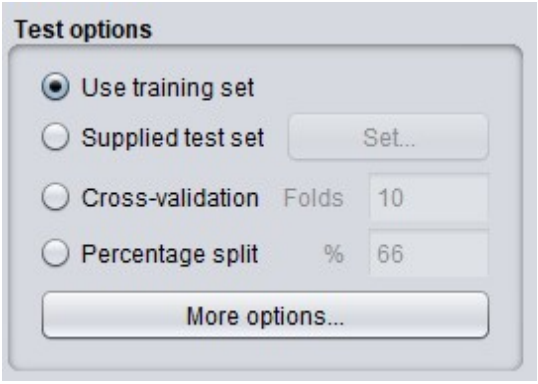


Figura 29. Opciones de test en Weka (Elaboración propia).

Después, una vez elegido el algoritmo a utilizar y mostrados los resultados en el sumario, se obtienen aquellas instancias correcta e incorrectamente clasificadas y, **Kappa Statistic,** parámetro que mide la coincidencia de la predicción con la clase real. Si éste es, 1.0, significa que hubo coincidencia absoluta (Carletta, 1996). También se encuentran el error absoluto medio, error cuadrático medio, error cuadrático relativo de raíz y número de instancias totales.

Después, se muestra la precisión detallada por clase con los siguientes indicadores representados en columnas. A continuación, se explica el significado de cada una de ellas (Corso, 2009):

- **TP Rate (True Positive Rate):** es la proporción de ejemplos que fueron clasificados como clase x, de entre todos los ejemplos que de verdad tienen clase x, es decir, qué cantidad de la clase X ha sido clasificada. En la matriz de confusión, es el valor del elemento de la diagonal dividido por la suma de la fila relevante. Esta medida es considerada como una de las más valiosas, ya que, nos dice el número de predicciones correctas e incorrectas que se resume con valores de recuento y que se desglosa por clase.

- **FP Rate (False Positive Rate):** proporción de ejemplos que fueron clasificados como clase x, pero en realidad pertenecen a otra clase. En la matriz de confusión es la suma de la columna menos el valor del elemento de la diagonal dividido por la suma de las filas de las otras clases.

- **Precision:** mide el número de *términos*, instancias, correctamente reconocidos respecto al total de *términos* predichos.

- **Recall:** proporción de *términos* correctamente reconocidos respecto al total de *términos* reales.

- **F-Measure:** La medida F se define como una media armónica de precisión (P) y recuperación (R) (Sasaki, 2007).

- **MCC[18]:** coeficiente de correlación de Matthews.

- **ROC Area:** Curva de ROC (Receiver Operating Characteristic) para el gráfico de los verdaderos positivos frente a los falsos positivos para el umbral de clasificación.

[18] https://scikit-learn.org/stable/modules/model_evaluation.html#matthews-corrcoef

- **PRC Area:** Curva de recuperación de precisión, para el gráfico de precisión frente a recuperación para todos los puntos de corte potenciales de la clasificación.

- **Class:** clase elegida para el análisis.

Otra forma de utilizar los algoritmos en WEKA es desde la línea de comandos **MS-DOS**. En ocasiones resulta más práctico ya que si se requiere ejecutar muchas veces un clasificador para obtener los resultados, éste se puede automatizar en bucle.

Para ello hay que situarse en el directorio de instalación de WEKA. En Windows, normalmente es el mostrado en la Figura 30.

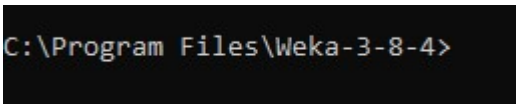


Figura 30. Directorio de instalación WEKA (Elaboración propia).

Una vez dentro del directorio, se introduce el comando: *java -cp weka.jar weka.classifiers.trees.J48 -t data/weather.nominal.arff* . El resultado será el siguiente:

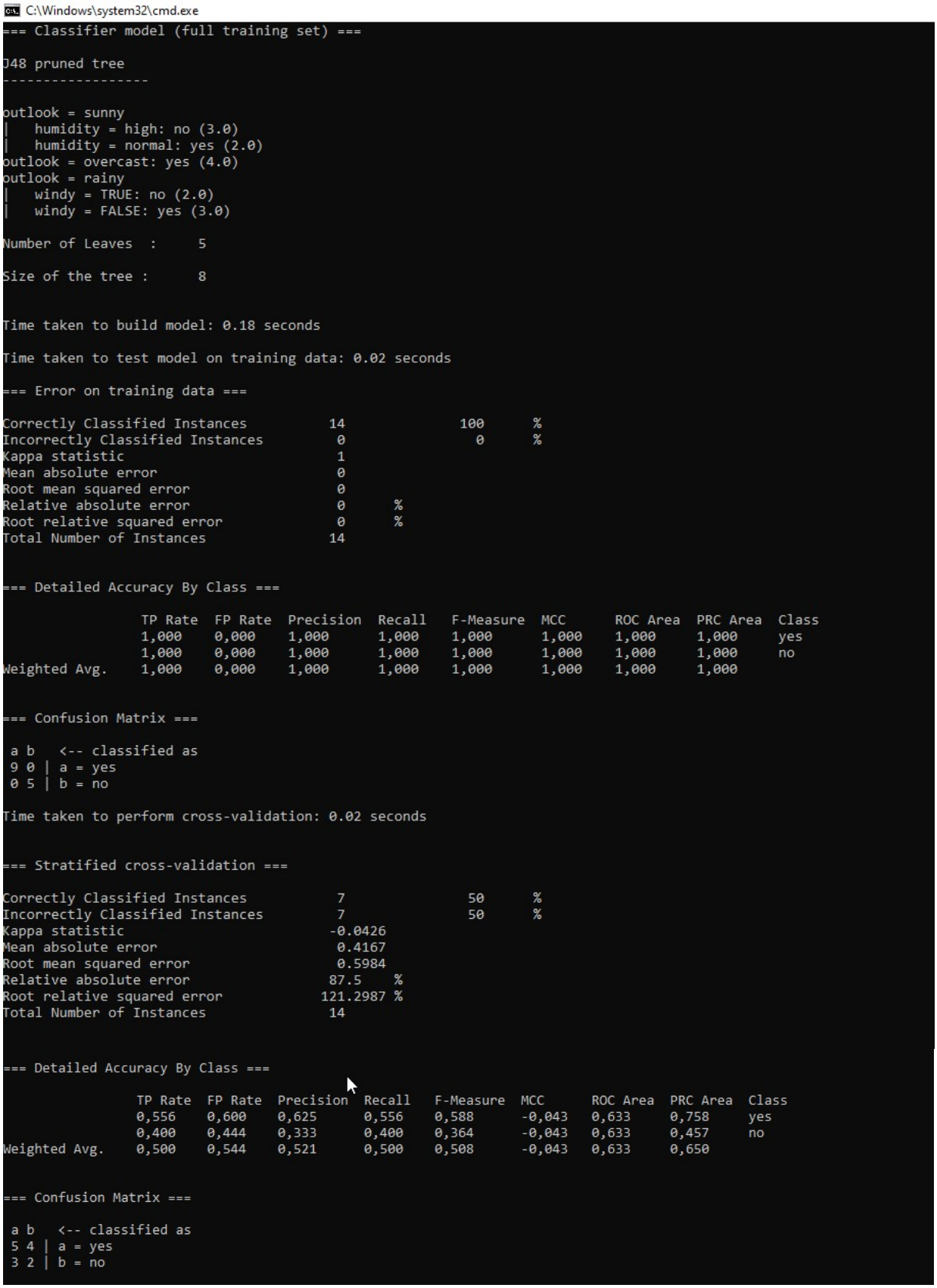


Figura 31. Resultado fichero weather.nominal.arff en MS-DOS WEKA (Elaboración propia).

En resultado de la Figura 31 el mismo que se obtendría desde el entorno gráfico. A continuación se analiza en detalle la orden que se ha introducido en MS-DOS:

- **java:** ejecuta el intérprete java.
- **-cp weka.jar:** ejecutar *weka.jar* en el directorio de instalación.
- **weka.classifiers.trees.J48:** clasificador elegido en este caso.
- **-t data/weather.nominal.arff:** el fichero de datos de entrenamiento que se va a utilizar.

Dentro del directorio *C:\Program Files\Weka-3-8-4\doc\weka* se localiza cualquier otro clasificador o cualquier otra función de WEKA. En este directorio existen los siguientes subdirectorios:

- **Associations:** métodos orientados a la búsqueda de asociaciones entre datos.
- **AttributeSelection:** selección de características.
- **Clasifiers:** algoritmos clasificadores.
- **Clusterers:** agrupación de datos.
- **Core:** aquellas funciones del núcleo de WEKA.
- **Datagenerators:** la generación automática de los datos.
- **Estimators:** estimadores estadísticos.
- **Experiments:** interfaz de usuario.
- **Filters:** filtros correspondientes con los atributos y datos.

- **Gui:** interfaz de usuario.

- **Knowledgeflow:** contiene todo lo referente a diseño de configuraciones para el procesamiento de datos transmitidos.

4.2 Orange

Orange[19] es una herramienta de minería de datos y de aprendizaje automático de uso general. Caracterizada por ser de código abierto y software de visualización con una comunidad activa y que ayuda a principiantes y expertos en su análisis. Hay que destacar que esta herramienta es compatible con los sistemas operativos Mac OS, GNU/Linux y Windows.

Esta herramienta es adecuada especialmente para principiantes ya que su interfaz es intuitiva y además por la visualización de patrones y facilidad de interpretación de los datos sin saber codificar. También es muy útil para aquellos procesos analíticos que tienen programación visual sencilla de utilizar o *scripting* en Python. Además, esta herramienta permite aplicar todos los principales algoritmos de minería de datos.

Orange es muy eficaz cuando el concepto innovación, fiabilidad o calidad está involucrado. Proporciona diferentes características tal y como se pueden ver en la Figura 32. Características tales como: visualización de datos, clasificación, evaluación, aprendizaje, asociación, visualización sin supervisión usando Qt y las implementaciones de prototipos.

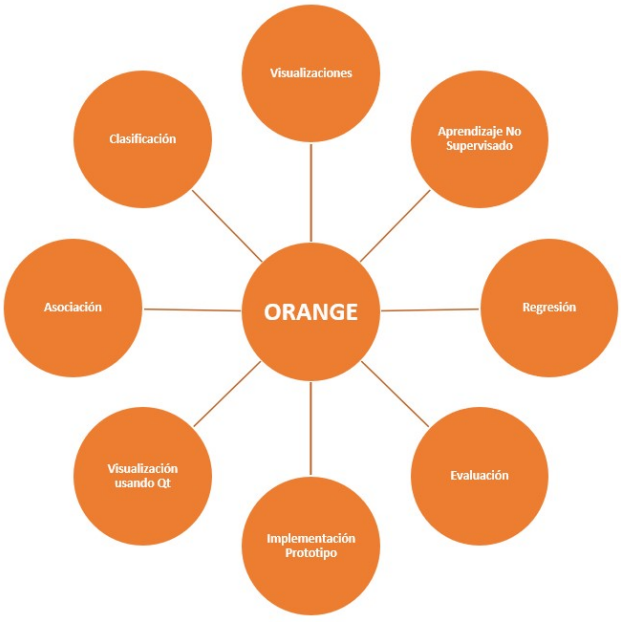


Figura 32. Características de la herramienta Orange (Ratra & Gulia, 2020).

[19] https://orangedatamining.com/

Orange se puede ampliar con módulos adicionales. Actualmente dispone de una amplia colección de módulos. Es utilizada en bioinformática, biomedicina, enseñanza, investigación genómica entre otros.

En la actualidad, el socio industrial más notable Astra-Zeneca, utiliza Orange en el desarrollo de fármacos y patrocina el desarrollo de varias partes relacionadas de Orange (Zupan & Demšar, 2013).

4.2.1 Conociendo Orange

Una de las características más destacadas de Orange es su interfaz gráfica de usuario CANVAS y su entorno de programación visual. También contiene una amplia colección de *widgets* para la visualización y exploración. Estos *widgets* dan flexibilidad a la hora de crear flujos para la entrada y salida de datos. Cada widget se explica por sí mismo, ya que posee una breve descripción en la interfaz. Además, Orange se puede ampliar con módulos adicionales para minería de datos, aprendizaje *multi-target*[20], bioinformática.

Para instalar este software es necesario ir al sitio oficial: https://orangedatamining.com/download/ y descargar el paquete de instalación. El proceso de instalación es sencillo; solo se deben de seguir los pasos de instalación que te pida el programa. Finalizada la instalación, se podrá acceder a la interfaz de inicio o pantalla de bienvenida.

Cuando se abre Orange, lo primero que aparece es la pantalla de bienvenida tal y como aparece en la Figura 33.

[20] Permite la clasificación de conjuntos de datos con múltiples clases.

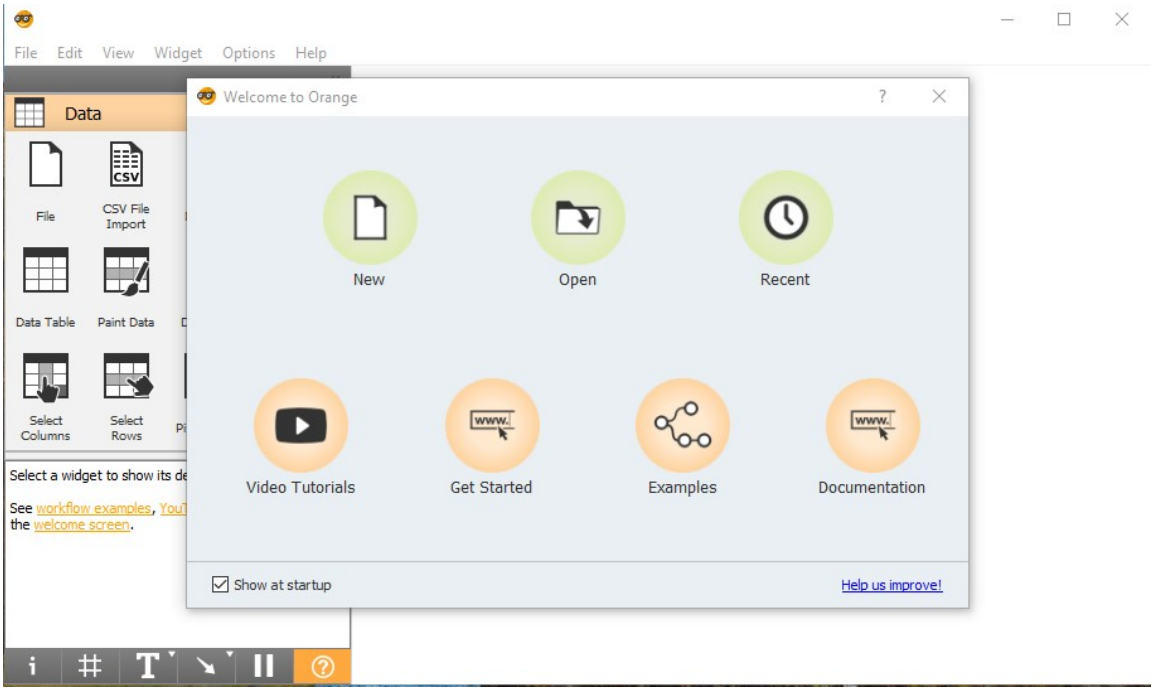


Figura 33. Pantalla de bienvenida de Orange (Elaboración propia).

En la pantalla de bienvenida aparece iniciar un flujograma de análisis de datos (*New*), abrir uno (*Open*), ejemplos (*Examples*), video tutoriales (*Video Tutorials*), documentación (D*ocumentation*), abrir uno reciente (*Recent*), ir a la página web (*Get Started*). Una vez cerrada la ventana de bienvenida se muestra un lienzo en blanco es donde se encuentran los *widgets* utilizados para leer los datos, procesar, visualizar, crear modelos predictivos y también ayudar a explorar los datos de diferentes maneras. En la Figura 34 es donde se muestran todos los *widgets*.

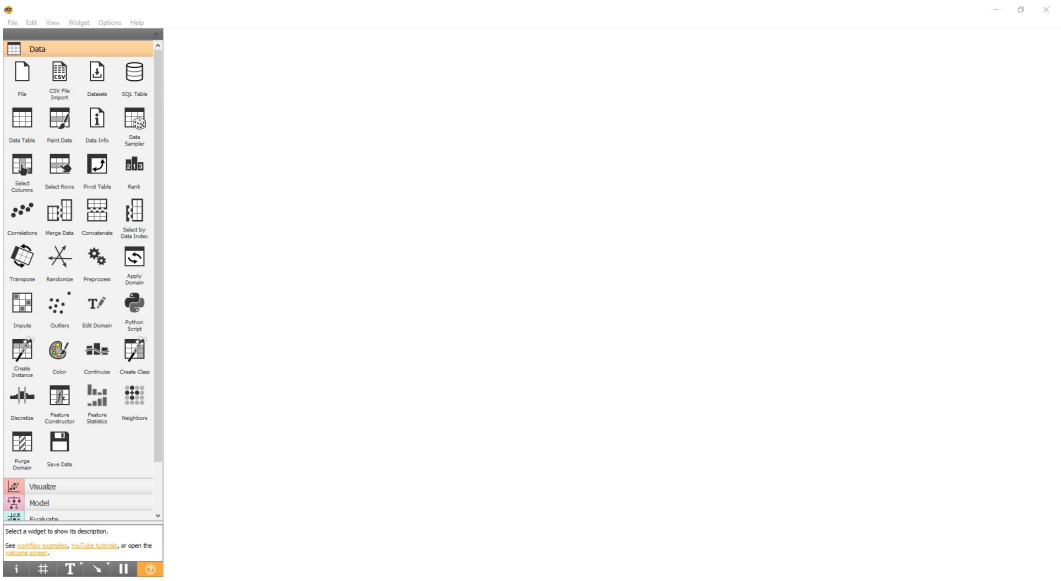


Figura 34. Pantalla inicial de Orange (Elaboración propia).

En la Figura 34, los widgets que aparecen en el lado izquierdo son aquellos que se arrastrarán y soltarán de la caja *Data* al lienzo, es decir, se colocan los *widgets* en el lienzo. Luego las entradas y salidas se conectan. El *widget* es el punto básico de procesamiento de cualquier manipulacion de datos y, dependiendo de lo que se seleccione, se pueden realizar una serie de acciones. En Orange todos los *widgets* que aparecen son:

- **Data:** para la entrada de datos, muestreo, selección y manipulacion de caracteristicas, filtrado de datos, etc.

- **Visualize:** visualización de diagrama de caja, histogramas, diagrama detamiz, diagrama de dispersión y mosaico.

- **Model:** modelos adaboost, regresión lineal, random forest, redes neuronales, etc.

- **Evaluate:** procedimientos basados en muestreo, validación cruzada, Confusion Matrix, ROC Análisis, predicciones.

- **Unsupervised:** algorimos de aprendizaje no supervisado k-means, agrupación jerárquica, y técnicas de proyección de datos.

- **Prototypes:** crear prototipos, explicación de modelos y predicciones, Oracle SQL, etc.

- **Image Analytics:** procedimientos para trabajar con imágenes.

- **Time Series:** análisis y modelado de series.

- **Textable:** Preprocesamiento de texto, segmentación, unión, importar datos desde un archivo de texto sin procesar, etc.

- **Text Mining:** procesamiento del lenguaje natural y minería de texto.

- **Single Cell:** cargar un conjunto de datos desde un repositorio en línea, preprocesar un solo conjunto de datos, normalización del efecto por lotes en un conjunto de datos de una sola celda, etc.

- **Networks:** gráficos y análisis de redes.

- **Geo:** para trabajar con datos geoespaciales.

- **Explain:** inspeccionar el modelo usando la técnica de importancia de característica de permutación, explicación del modelo y de predicción.

- **Educational:** agrupamiento de k-means, regresión polinominial, descenso de gradiente, fusionar diferentes conjuntos de datos, etc.

- **Bioinformatics:** análisis del conjunto de genes, enriquecimiento y acceso a bibliotecas de vías.

- **Associate:** aprendizaje de reglas de asociación y extracción de conjuntos de elementos frecuentes.

- **Spectroscopy:** visualización de análisis de datos espectrales.

Los *widgets* disponibles están limitados. El flujo de trabajo será una serie de acciones o pasos para realizar una tarea concreta. Es por eso que entre las principales funcionalidades, están las siguientes:

- Se adapta la herramienta a las necesidades del usuario o de la empresa y también permite rediseñarse.

- Los usuarios crean sus propios flujos interactivos de trabajo con el objetivo de visualizar y analizar los datos fácilmente.

- Permite la visualización de los datos mediante distintos formatos, como por ejemplo, histogramas, diagramas de dispersión, árboles, mapas de color, redes. Esto permite una mayor claridad de la compresión de los datos a la hora de su interpretación.

Orange tiene un gran número de *widgets*. Cada uno para distintas categorías de análisis, como, por ejemplo, para redes, bioinformática, texto, educación, etc.

4.2.2 Preparación y entrada de datos

La preparación de entrada de datos suele consumir la mayor parte del esfuerzo invertido en todo el proceso de aprendizaje. El formato de archivo de entrada, en particular, el formato de archivos por defecto (**TAB**[21]) que utiliza Orange, son archivos de texto o datos delimitados por tabuladores (valores separados por tabuladores). También se pueden introducir formatos **CSV**, **DAT**[22], **TXT**, **XLSX**, **MAP**[23], entre muchos más. Además de introducir archivos de distintos formatos, en Orange pueden introducirse URLs para la entrada de datos.

En Orange se pueden utilizar los *Datasets* que vienen por defecto en su instalación, tal y como vienen en la Figura 35, o bien, se pueden utilizar datos de fuentes abiertas como ya hemos comentado anteriormente, tales como: *Kaggle*, Datos de la Comunidad de Madrid, Datos del Gobierno de España o aquéllos que nos proporcionen terceras personas para hacer el estudio. Con este conjunto de datos se puede comenzar a realizar pruebas en Orange.

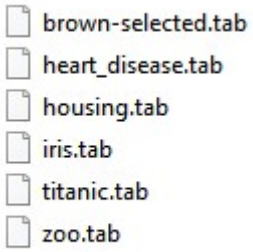


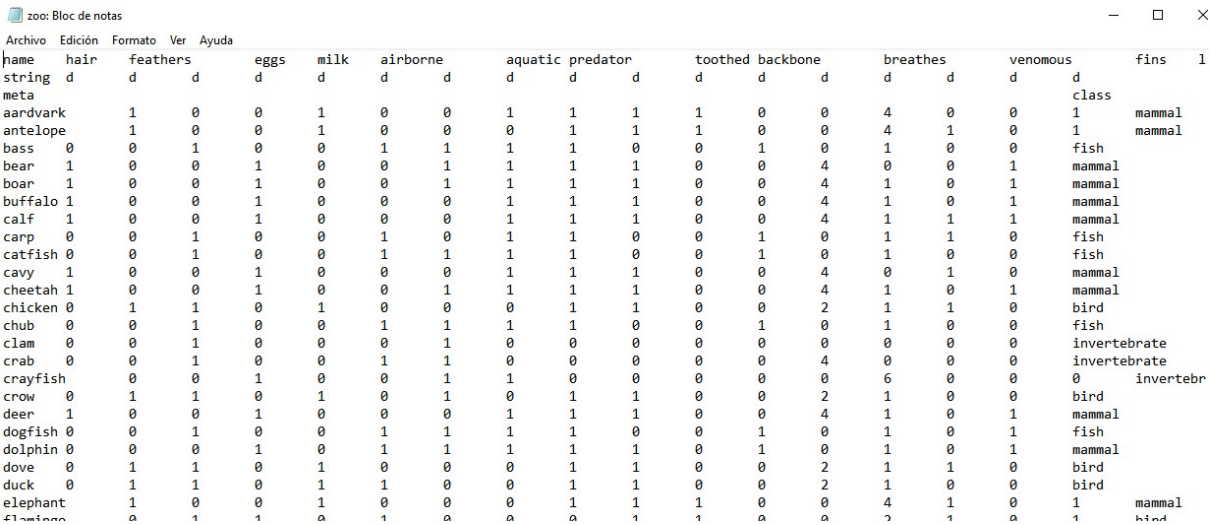
Figura 35. Datasets por defecto en Orange (Elaboración propia).

[21] TAB abreviatura de "tabulador".
[22] Data File.
[23] Archivo de mapas de Quake Engine.

El formato con el que trabaja por defecto Orange (TAB), posee una estructura dividida en las siguientes partes:

- Encabezado: se definen el nombre de cada uno de los datos en su columna correspondiente.
- Declaración de atributos: se definen los atributos a utilizar especificando su tipo.
- Declaración de la clase: se define la clase por la cual se realizará la clasificación.
- Declaración de datos: se definen los datos que componen la relación.

Por ejemplo, la Figura 36 muestra cómo sería un archivo TAB para datos de un Zoo. En la primera fila está el encabezado nombrando qué es cada columna. En la segunda fila se definen el tipo de dato al que corresponde esa columna. La tercera fila define la clase, la cual será decisiva para el tipo de dato que compone la relación. Por último, en la cuarta fila estarán los datos.


zoo: Bloc de notas

Archivo Edición Formato Ver Ayuda

name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	venomous	fins	l
string	d	d	d	d	d	d	d	d	d	d	d	d	
meta											class		
aardvark	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	mammal	
antelope	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	mammal	
bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	fish	
bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	1	mammal	
boar	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	1	mammal	
buffalo	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	mammal	
calf	1	0	0	1	0	0	0	1	1	1	0	0	4	1	1	1	mammal	
carp	0	0	1	0	0	1	0	1	1	0	0	1	0	1	1	0	fish	
catfish	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	fish	
cavy	1	0	0	1	0	0	0	1	1	1	0	0	4	0	1	0	mammal	
cheetah	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	1	mammal	
chicken	0	1	1	0	1	0	0	0	1	1	0	0	2	1	1	0	bird	
chub	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	fish	
clam	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	invertebrate	
crab	0	0	1	0	0	1	1	0	0	0	0	0	4	0	0	0	invertebrate	
crayfish		0	0	1	0	0	1	1	0	0	0	0	0	6	0	0	0	invertebr
crow	0	1	1	0	1	0	1	0	1	1	0	0	2	1	0	0	bird	
deer	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	mammal	
dogfish	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	1	fish	
dolphin	0	0	0	1	0	1	1	1	1	1	0	1	0	1	0	1	mammal	
dove	0	1	1	0	1	0	0	0	1	1	0	0	2	1	1	0	bird	
duck	0	1	1	0	1	1	0	0	1	1	0	0	2	1	0	0	bird	
elephant		1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	mammal
flamingo		0	1	1	0	1	0	0	0	1	1	0	0	2	1	0	1	bird

Figura 36. Fichero TAB con datos Zoo (Elaboración propia).

Una vez se tenga el archivo limpio, se procederá a introducirlo en Orange para su posterior análisis.

4.2.3 Manejo de Orange

Los algoritmos de aprendizaje automático implementados en Orange nos permiten realizar entrada/salida de datos, visualizar, modelos de regresión y clasificación, evaluaciones al conjunto de datos, algoritmos no supervisados, entre otros. En la Figura 37 aparece la ventana principal de Orange con los distintos componentes explicados en la sección 4.2.1:

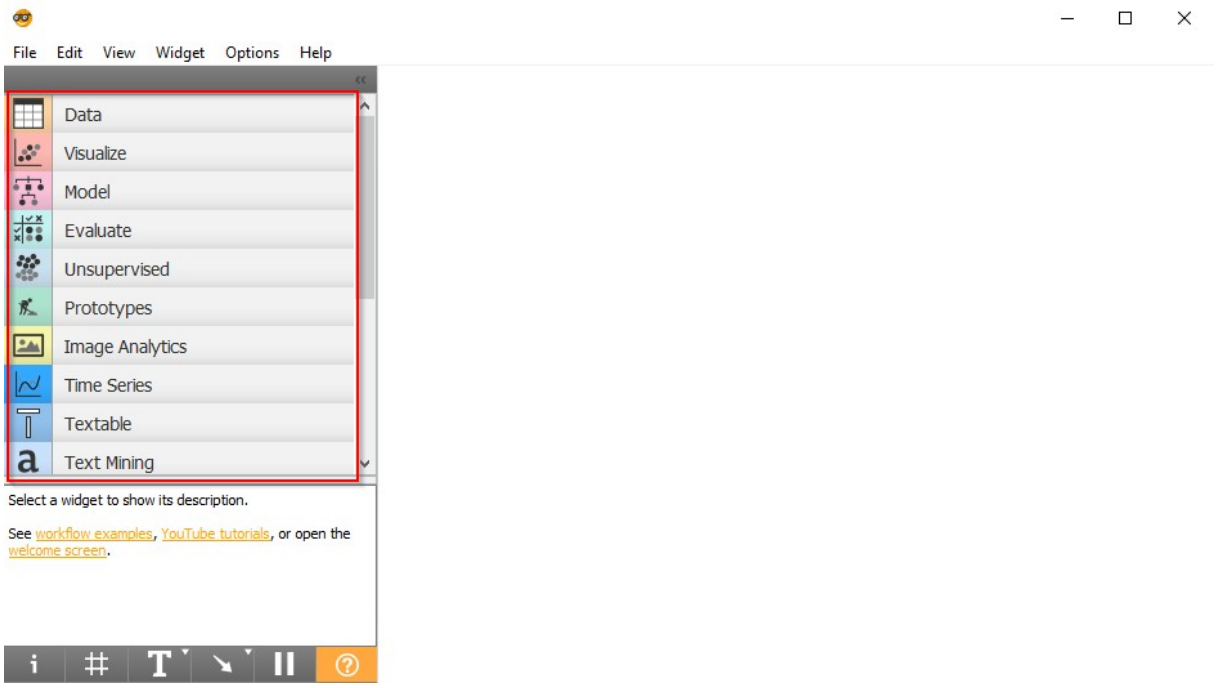


Figura 37. Ventana Principal en Orange (Elaboración propia).

Dentro de estos componentes mencionados ya en la sección 4.2.1, éstos leen, procesan y visualizan los datos. Todo ello se llevará a cabo con los *widgets*, que se colocarán en el lienzo. Estos se comunicarán entre sí enviando información por un canal. De esta forma, se construye un flujo de trabajo arrastrando *widgets* al lienzo y conectándolos entre sí, dibujando una línea desde el *widget* transmisor al *widget* receptor. En la Figura 38 se muestra un flujo de trabajo con dos *widgets* conectados. Las entradas aparecen a la derecha y las salidas aparecen a la izquierda.

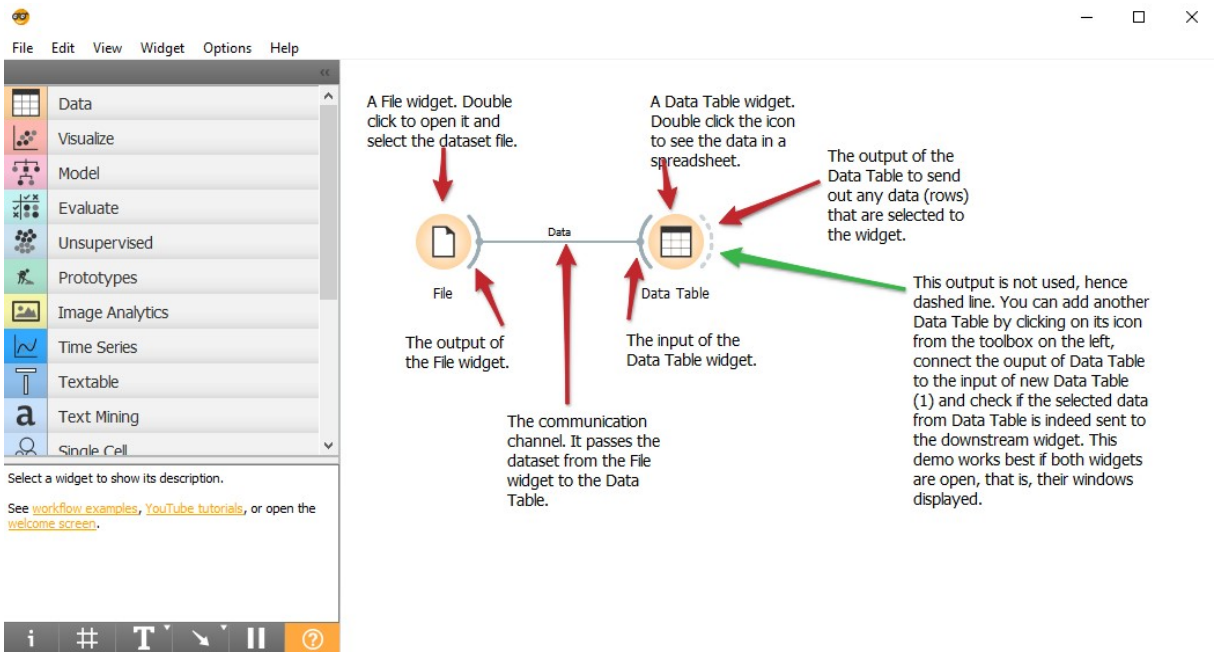


Figura 38. Ejemplo Flujo de Trabajo Orange (Elaboración propia).

Es importante señalar que Orange, contiene un fichero que se llama Introducción a la Minería de Datos[24], que nos explica Lección a Lección los pasos para la creación de los flujos de trabajo junto con su explicación correspondiente. Esto nos ayudará a entender mejor la herramienta y su entorno a la hora de aplicar el problema.

4.3 Análisis en función de los criterios seleccionados

En este apartado se expone, en base a los criterios seleccionados, el análisis de las herramientas de aprendizaje automático explicadas en los puntos anteriores. Este análisis se realiza en función de los criterios tomados por autores que han publicado sobre estas herramientas, y estudiados en profundidad (véase sección 5) más las aportaciones del autor del presente trabajo.

Accesibilidad

Se encuentran autores como (Ameen*, Bajeh, Adesiji, Balogun, & Mabayoje, 2018), (Ranjan & Agarwal, 2017), (Dušanka, Darko, Srdjan, Marko, & Teodora, 2017), (Ramamohan, Vasantharao, Kalyana Chakravarti, & Ratnam, 2012), (Kulkarni & Kulkarni, 2016), que

explican cómo la interfaz de usuario de Weka es totalmente funcional y flexible por lo que hace más fácil el acceso a los componentes principales. Además, mencionan que al contener interfaz gráfica hace más fácil su utilización y desempeño. Otros autores como (Jović, Brkić, & Bogunović, 2014), (Ratra & Gulia, 2020), explican como Weka es una herramienta potente y versátil, pero que carece de buena visualización de datos. Otro autor (Nehru, 2018) comentan que Weka no requiere conocimientos de codificación por lo que es fácil de utilizar. En Orange, autores como (Padmavaty, Geetha, & Priya, 2020), (Ratra & Gulia, 2020), explican que es fácil de usar ya que Orange proporciona una interfaz bien estructurada con diferentes caracteristicas. Lo que facilita el trabajo de análisis. Finalmente autores como, (Al-Odan & Saud, 2015), (Jović, Brkić, & Bogunović, 2014), (Ranjan & Agarwal, 2017), comentan que Orange está por encima de Weka al navegar por todas sus funcionalidades con facilidad y además, la interfaz gráfica es visualmente atractiva, lo que ofrece una experiencia de usuario agradable. En la Tabla 2 podemos observar el nivel de accesibilidad de ambas herramientas.

Herramienta	Accesibilidad
Weka	Alta
Orange	Alta

Tabla 2. Criterio Accesibilidad en Weka y Orange (Dušanka, Darko, Srdjan, Marko, & Teodora, 2017).

Esto supone una ventaja para ambas herramientas, ya que poseen una interfaz de usuario sencilla que permite su uso hasta por usuarios sin experiencia. Además, esta característica permite acercar al usuario a una gran gama de técnicas de aprendizaje y procesamiento de datos.

Intuitivo

En términos de intuición, los autores (Al-Odan & Saud, 2015), hicieron una encuesta donde los usuarios eligieron a Orange como la herramienta más intuitiva. No se puede decir lo mismo de Weka ya que los usuarios la calificaron por debajo de Orange. En la Tabla 3 se puede ver el nivel de intuición de ambas herramientas.

[24] https://file.biolab.si/notes/2018-05-intro-to-datamining-notes.pdf

Herramienta	Intuitivo
Weka	Media
Orange	Alta

Tabla 3. Criterio Intuitivo en Weka y Orange (Al-Odan & Saud, 2015).

Simple de entender

Sobre este criterio, autores como (Ranjan & Agarwal, 2017), (Jović, Brkić, & Bogunović, 2014), explican como Orange es más sencilla de aprender que Weka debido a su interfaz gráfica visualmente atractiva. Por lo tanto, la interfaz de Orange es mucho más intuitiva y simple de entender que la interfaz de Weka. Por lo tanto, en base a los estudios previamente observados se concluye que Orange es más simple de entender que Weka (ver Tabla 4).

Herramienta	Simple de Entender
Weka	Alta
Orange	Muy Alta

Tabla 4. Criterio Simple de entender en Weka y Orange (Elaboración propia).

Comprensión

Sobre este criterio, los autores mencionados en la sección 2.7 no hicieron reseña por lo que este criterio es propio del autor de este trabajo. A nivel de comprensión de las herramientas según su diseño percibimos que son fáciles de comprender (ver Tabla 5).

Herramienta	Comprensión
Weka	Alta
Orange	Alta

Tabla 5. Criterio Comprensión en Weka y Orange (Elaboración propia)

Compatible

En términos de compatibilidad, los autores (Ramamohan, Vasantharao, Kalyana Chakravarti, & Ratnam, 2012), (Patel & Desai, 2015), (Padmavaty, Geetha, & Priya, 2020), (Ranjan &

Agarwal, 2017), explicaron que Weka y Orange son compatibles con todos los sistemas operativos actualmente disponibles: Windows, Linux y Mac OS.

Rendimiento

En términos de rendimiento, los autores (Ameen*, Bajeh, Adesiji, Balogun, & Mabayoje, 2018), han descrito un buen rendimiento en los algoritmos y en su ejecución (ver Tabla 6) en ambas herramientas. Pero cabe destacar que en Orange no se mide el tiempo de ejecución porque no tiene la funcionalidad incorporada de medición del tiempo de ejecución. Esto implica que Weka es superior a Orange.

Herramienta	Rendimiento
Weka	Alta
Orange	Media

Tabla 6. Criterio Rendimiento en Weka y Orange (Ameen*, Bajeh, Adesiji, Balogun, & Mabayoje, 2018).

Usabilidad

Basándonos en los resultados de una encuesta de los autores (Al-Odan & Saud, 2015) donde los usuarios eligieron a Orange como la herramienta más usable por encima de Weka. Otro autor (Lagos Vera, 2011) concluye en su investigación que Weka y Orange son las herramientas más satisfactorias de usar. Sin embargo, en término de la usabilidad se observó a Orange por encima de Weka, debido a la mala visualización de los datos en Weka. En base a estas investigaciones podemos decir que Orange en términos de usabilidad está por encima de Weka (ver Tabla 7).

Herramienta	Usabilidad
Weka	Media
Orange	Media-Alta

Tabla 7. Criterio Usabilidad en Weka y Orange (Lagos Vera, 2011).

Amigabilidad

Este término está relacionado con la usabilidad. Permite que una persona que tiene poca experiencia interactúe de forma exitosa con la herramienta. En este caso, este criterio es propio del autor de este trabajo. En este caso podemos decir que Weka a veces es poco amigable ya que carece de buena visualización de datos y ausencia de muchos datos y visualización de métodos, según los autores (Jović, Brkić, & Bogunović, 2014), (Ratra & Gulia, 2020). En Orange se cumple este criterio por lo que se considera Orange más amigable que Weka.

Explicabilidad

Sobre este término ningún autor hizo referencia a este criterio por lo que es propio del autor de este trabajo. Sobre este término debemos preguntarnos si los algoritmos son explicables lo suficiente como para verificar que sus resultados son adecuados para el propósito que hemos planteado. En ambas herramientas, la utilización de los algoritmos y su explicación se adecua a los resultados obtenidos por lo que, en conclusión, ambas herramientas cumplen este criterio.

Interpretabilidad

Este término está relacionado con la interpretación de los resultados. Varios autores como (Al-Odan & Saud, 2015), (Padmavaty, Geetha, & Priya, 2020), (Nehru, 2018), (Ratra & Gulia, 2020), (Kulkarni & Kulkarni, 2016) explican sus investigaciones en base a lo que interpretan de los resultados sin dificultad. Por lo tanto, ambas herramientas cumplen este criterio.

4.4 Análisis Crítico

Después del análisis en función de los criterios seleccionados de las herramientas de aprendizaje automático, cabe destacar que en la actualidad es importante resaltar la importancia de elegir una herramienta adecuada que permita extraer toda la información posible. Tanto Weka como Orange proporcionan una amplia colección de funcionalidades

para investigadores, profesionales así como para usuarios sin experiencia. Cuentan también con número amplio de técnicas de aprendizaje automático para ampliar su potencial, lo que las convierte en un referente importante en herramientas de aprendizaje automático de código abierto.

Otra conclusión que se obtiene es que se han analizado las herramientas en base a unos criterios que han sido seleccionados en base a la necesidad de este trabajo y a las referencias por otros autores. Por este motivo, esto nos hace tomar importancia de cuál es la herramienta que más se adecua al aprendizaje/enseñanza y con qué propósito.

Posteriormente, en base a las pruebas realizadas en ambas herramientas en el capítulo siguiente se añade a este análisis los resultados obtenidos.

Capítulo 4

5. Evaluación herramientas WEKA y ORANGE

Este apartado se enfoca en pruebas y análisis de los algoritmos de aprendizaje automático supervisado y no supervisado que hemos comentado con anterioridad y que son los más utilizados actualmente.

Para ello dispondremos de un conjunto de datos fiables que utilizaremos para hacer la evaluación de las herramientas. Estos conjuntos de datos se encuentran en el pequeño directorio[25] que proporciona Weka una vez se ha instalado en nuestro computador:

- **Breast-cancer (Cáncer de Mama[26]):** este fichero contiene los detalles médicos de pacientes y muestras de su tejido tumoral. Hay nueve variables de entrada, todas aquellas son nominales. Se utiliza para predecir si la paciente tiene cáncer de mama o no.

- **Iris:** Cada instancia describe las medidas de las flores iris. Hay cuatro variables de entrada numéricas con las mismas unidades y mayormente la misma escala. Se utiliza para predecir a que especie de flor de iris pertenece la observación.

- *Dataset* público sobre la **tasa de actividad, empleo y paro** de los graduados universitarios por CCAA[27] de su universidad y ámbito de estudio[28] en el 2020: Tabla de INEbase Tasas de actividad, empleo y paro de los graduados universitarios por CCAA de su universidad y ámbito de estudio. Encuesta de Inserción Laboral de Graduados Universitarios.

[25] Directorio **Data.**
[26] Los datos se obtuvieron del Centro Médico Universitario, Instituto de Oncología, Ljubljana en Yugoslavia.
[27] Comunidad Autónoma.
[28] https://datos.gob.es/es/catalogo/ea0010587-tasas-de-actividad-empleo-y-paro-de-los-graduados-universitarios-por-ccaa-de-su-universidad-y-ambito-de-estudio-identificador-api-t13-p100-2019-p02-l0-03011-px

Para introducir el fichero Breast-cancer en la herramienta Orange necesitaremos convertirlo en otro formato, por ejemplo *.csv* o *.tab*, ya que este ficheros por defecto vienen en formato *.arff*. y Orange no admite este formato. El fichero Iris ya viene por defecto en ambas herramientas por lo que no es necesario convertir. El Dataset público tiene formato .csv por lo que es válido en ambas herramientas. **Muy importante** comentar que se debe hacer una limpieza exhaustiva de los datos ya que necesitamos que el Dataset este limpio para el posterior análisis.

Además, estos conjuntos de datos se deben introducir en las herramientas y aplicar los diferentes algoritmos. Posteriormente, se discutirán los resultados obtenidos con cada uno de ellos. Todo esto nos servirá para valorar, junto con el análisis de los criterios previamente seleccionados, cuál es la mejor opción a utilizar para el aprendizaje/enseñanza de los algoritmos y su aplicación en un contexto determinado para resolver problemas para los que sea adecuado su uso.

Los algoritmos de aprendizaje automático supervisado escogidos son los siguientes:

- Random Forest.
- Algoritmo J48.

Los algoritmos de aprendizaje automático no supervisado escogidos son los siguientes:

- K-Means.
- Agrupamiento Jerárquico Aglomerativo.

5.1 Prueba con Algoritmo J48

5.1.1 Weka

Se utilizará para el algoritmo J48 el fichero **Iris** que se encuentra, como se ha comentado con anterioridad, en el directorio *Data* que proporciona Weka. Dicho conjunto de datos tiene una extensión *.arff* para luego ser procesado en la herramienta tal y como aparece en la Figura 39.

```
@RELATION iris

@ATTRIBUTE sepallength  REAL
@ATTRIBUTE sepalwidth   REAL
@ATTRIBUTE petallength  REAL
@ATTRIBUTE petalwidth   REAL
@ATTRIBUTE class        {Iris-setosa,Iris-versicolor,Iris-virginica}

@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
```

Figura 39. Dataset Iris (Elaboración propia).

Después este conjunto de datos se carga en la herramienta Weka a través de la ventana *Explorer* obteniendo la siguiente vista reflejada en la Figura 40.

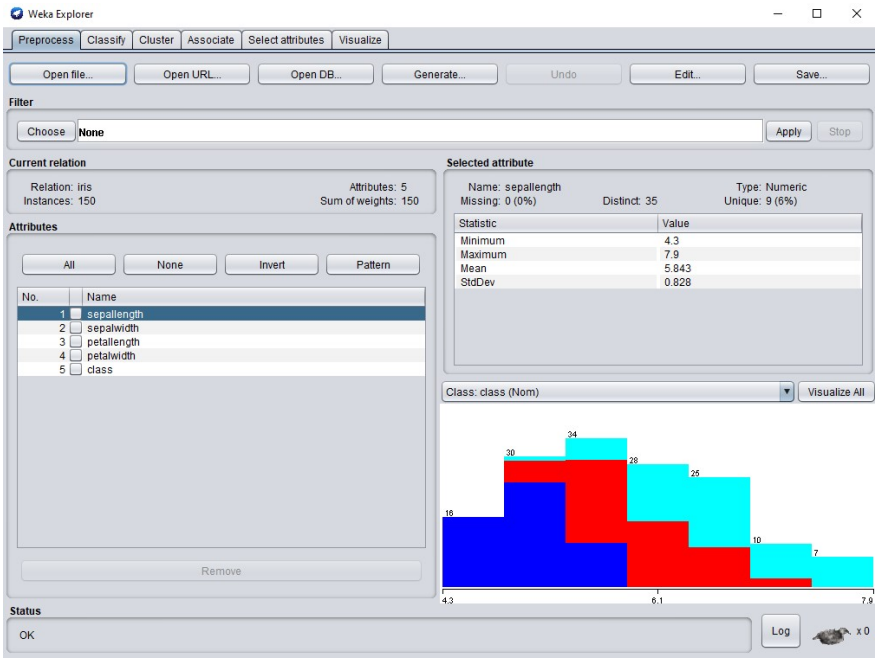


Figura 40. Dataset Iris cargado en Weka (Elaboración propia).

Una vez cargado el conjunto de datos Iris. Se elige el algoritmo J48 que aparece en la ventana *Classify. S*e incia el proceso mostrado en la Figura 41.

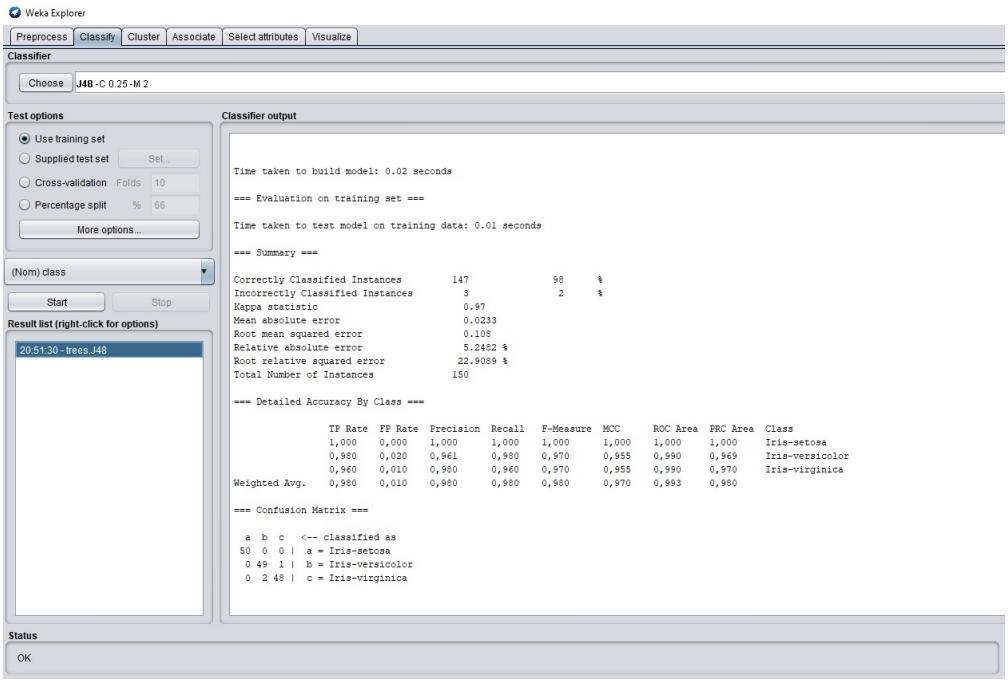


Figura 41. Resultados Algoritmo J48 con dataset Iris en Weka (Elaboración propia).

Una vez aplicado el algoritmo J48, este nos proporciona datos importantes. Podemos ver que el porcentaje de acierto del método es bastante bueno (Figura 42). No obstante no debemos ser optimistas con estos resultados ya que se han utilizado todos los datos de entrenamiento y sería necesario aplicar la validación cruzada = 10 o *percentage split* para saber si estos resultados son los que esperamos:

```
Correctly Classified Instances         147               98       %
Incorrectly Classified Instances         3                2       %
```

Figura 42. Instancias Clasificadas Algoritmo J48 en Weka (Elaboración propia).

Si nos fijamos en la matriz de confusión, se observa claramente que hay instancias mal clasificadas (Figura 43):

```
=== Confusion Matrix ===

  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 49  1 |  b = Iris-versicolor
  0  2 48 |  c = Iris-virginica
```

Figura 43. Confusion Matrix Algoritmo J48 en Weka (Elaboración propia).

Es por eso que podemos deducir que dos versiones han sido clasificadas como *virginicas* y, a su vez, dos *versicolor* pero todas las *setosas* han sido clasificadas correctamente. En la Figura 44 se visualiza el árbol. Este nos indica como ha clasificado el algoritmo el conjunto de datos en base al resultado obtenido.

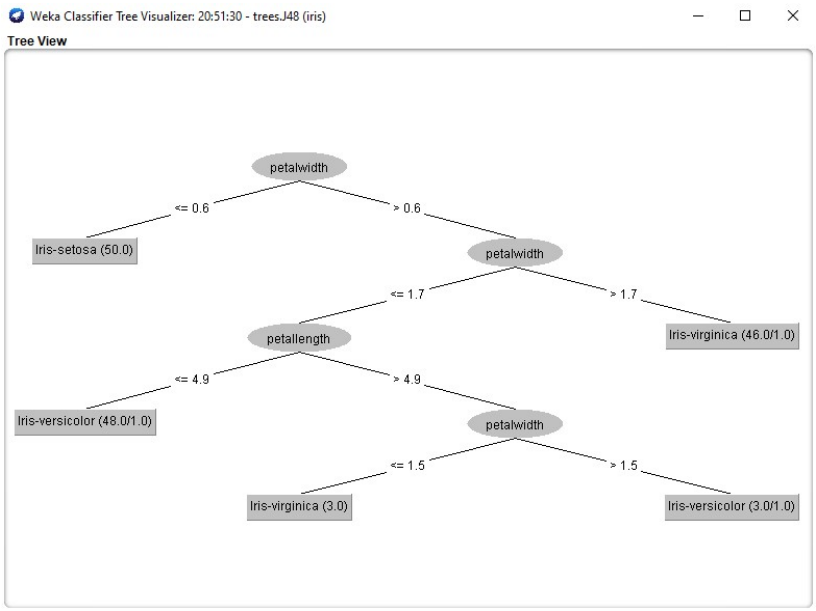


Figura 44. Visualización Árbol Algoritmo J48 en Weka (Elaboración propia).

Comprobamos en el árbol generado en la Figura 44 que hay 3 clases (*virginica, setosa y versicolor*). También comprobamos que hay 9 nodos y el árbol es de altura = 5. Los números entre paréntesis junto a su clase correspondiente muestran el número de instancias clasificadas dentro de esa clase. A veces aparece un número y en otras ocasiones aparecen dos números. Siguiendo el contenido de la Figura 45 en el árbol generado de la Figura 44, se comprueba como *Iris-versicolor* tiene 48.0 datos clasificados correctamente y 1.0 datos clasificados incorrectamente:

If Petalwidth > 0.6 : Nodo 3

If Petalwidth <= 1.7 : Nodo 4

If Petallength <= 4.9 : Nodo 5

Figura 45. Ruta Árbol Algoritmo J48 en Weka (Elaboración propia).

Después de haber hecho este análisis con el algoritmo J48, ¿podemos considerar que el árbol generado utilizando los datos de entrenamiento son optimistas? Tendríamos que considerar otras opciones como, por ejemplo, utilizar la validación cruzada con 10 iteraciones o aplicar percentaje split. Lo que si se ha comprobado, y que según los autores (Jović, Brkić, & Bogunović, 2014), (Ratra & Gulia, 2020), es que Weka a veces es poco amigable con la visualización de datos. Además, los informes no se pueden exportar externamente desde Weka. En términos de rendimiento, el algoritmo ha sido bueno en su ejecución. También, en términos de explicabilidad e interpretabilidad de los resultados se puedieron obtener conclusiones sin dificultad.

5.1.2 Orange

En Orange no existe como tal el algoritmo J48, como en Weka, por lo que se deberá utilizar los *widgets* para crear este algoritmo. Considerando que el algoritmo J48 es la implementación *open source* del algoritmo C4.5 y que en Orange hay documentación[29] para poder implementar este algoritmo, seguiremos estos pasos. Utilizaremos el fichero *Iris.tab* que tiene por defecto, una vez instalada la herramienta, en el directorio *Data*. En la Figura 46 se puede ver el flujo resultante.

[29]https://buildmedia.readthedocs.org/media/pdf/orange-visual-programming/latest/orange-visual programming.pdf

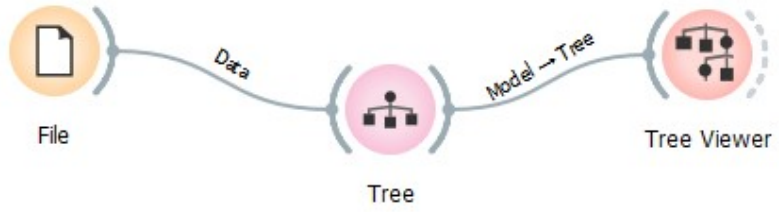


Figura 46. Flujo con widgets C4.5 en Orange (Elaboración propia).

Además, igual que en Weka, en Orange podemos visualizar el árbol generado (ver en Figura 47). En este árbol aparecen 3 clases (virginica, setosa y versicolor). También se puede comprobar que hay 9 nodos y el árbol es de altura = 5. Los números que aparecen junto a su clase correspondiente muestran el número de instancias localizadas dentro de esa clase más el porcentaje de aciertos. Siguiendo el contenido de la Figura 45 en la Figura 47 comprobamos que Iris-versicolor tiene un porcentaje de aciertos del 97.9% y además 47 datos clasificados correctamente de los 48 datos clasificados en el entrenamiento.

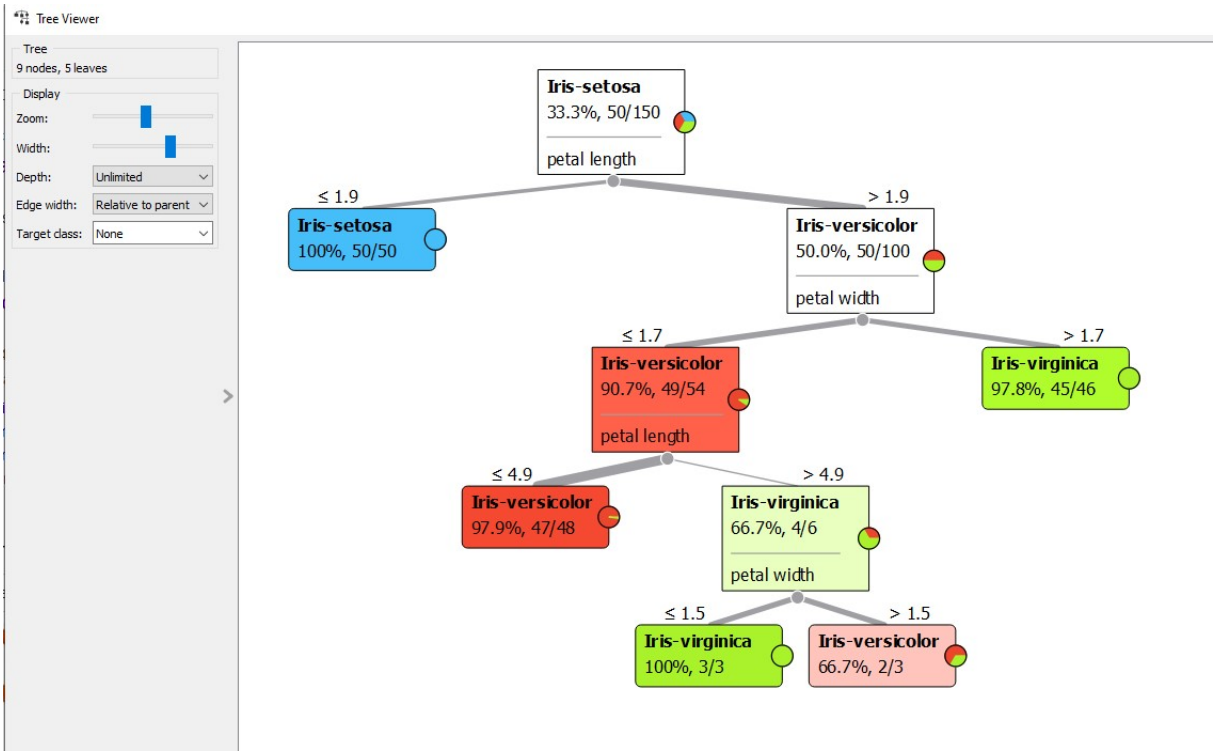


Figura 47. Visualización Árbol Algoritmo C4.5 en Orange (Elaboración propia).

En el flujo mostrado anteriormente en la Figura 47 no se reflejan los resultados de la evaluación ni tampoco la *Confusion Matrix* donde observar qué instancias específicas de una determinada clase se clasificaron bien y cuáles no. Es por eso por lo que se añadirán *widgets* para poder ver estos resultados. Con el *widget Test & Score* se obtienen los resultados de la evaluación y con el *widget Confusion Matrix* se podrán observar qué instancias se clasificaron errónea y correctamente. En la Figura 48, aparece el flujo con los *widgets* comentados anteriormente.

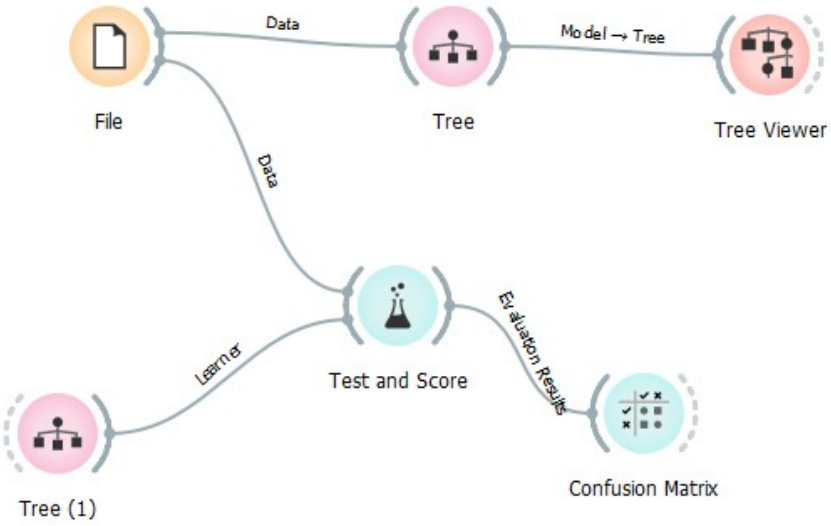


Figura 48. Visualización resultados y Confusion Matrix (Elaboración propia).

La Figura 49 muestra la matriz de confusión para el modelo *Tree* entrenado. Cada fila corresponde a una clase mientras que las columnas representan las clases predichas. Se puede observar que cuatro instancias de *Iris-versicolor* fueron clasificadas erróneamente como *Iris-virginica*. La columna a la derecha con el símbolo sumatorio ∑ contiene el número de instancias de cada clase. En este caso, son 50 instancias *Iris* de cada una de las tres clases. La fila inferior nos muestra el número de instancias clasificadas en cada clase como, por ejemplo, 51 clasificadas en *versicolor*.

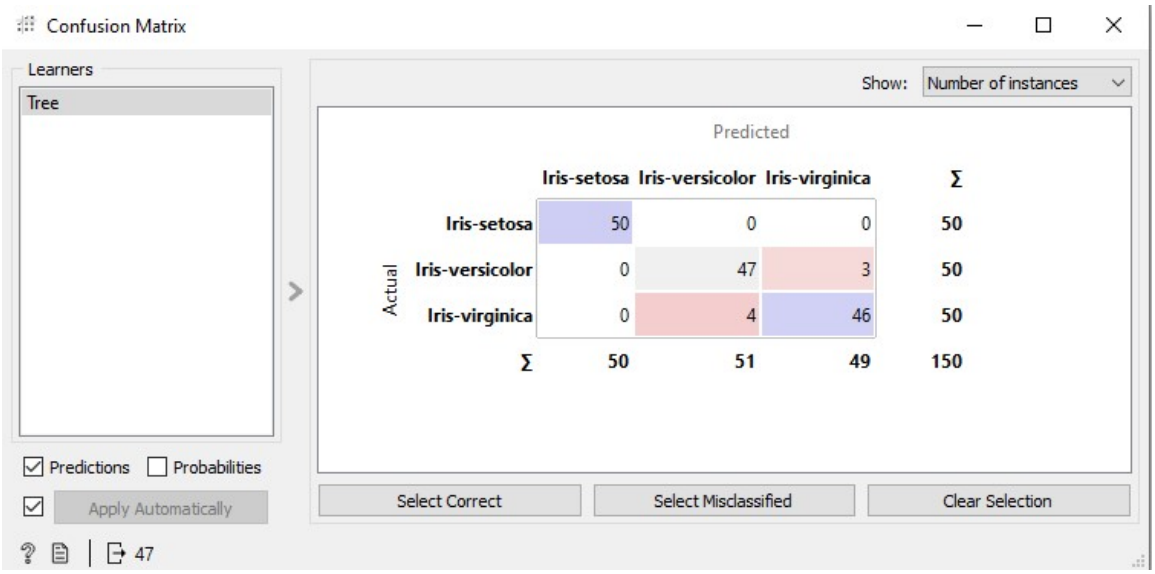


Figura 49. Confusion Matrix Algoritmo C4.5 en Orange (Elaboración propia).

En *Show* se selecciona qué datos se desean ver en la matriz:

- **Número de instancias:** nos muestra las instancias clasificadas correcta e incorrectamente.

- **Proporción de predicción:** se muestra cuántas instancias son clasificadas como, *Iris-versicolor* en esa clase (Figura 50). Observamos que el 92.2% de *Iris-versicolor* predicho era verdaderamente *Iris-versicolor* y el 7.8% del *Iris-versicolor* predicho donde actualmente es *Iris-virginica*.

- **Proporción real:** estas muestran la relación opuesta (Figura 51). La muestra real del 94% se predijo *Iris-versicolor*, pero el 6% se predijo *Iris-virginica*.

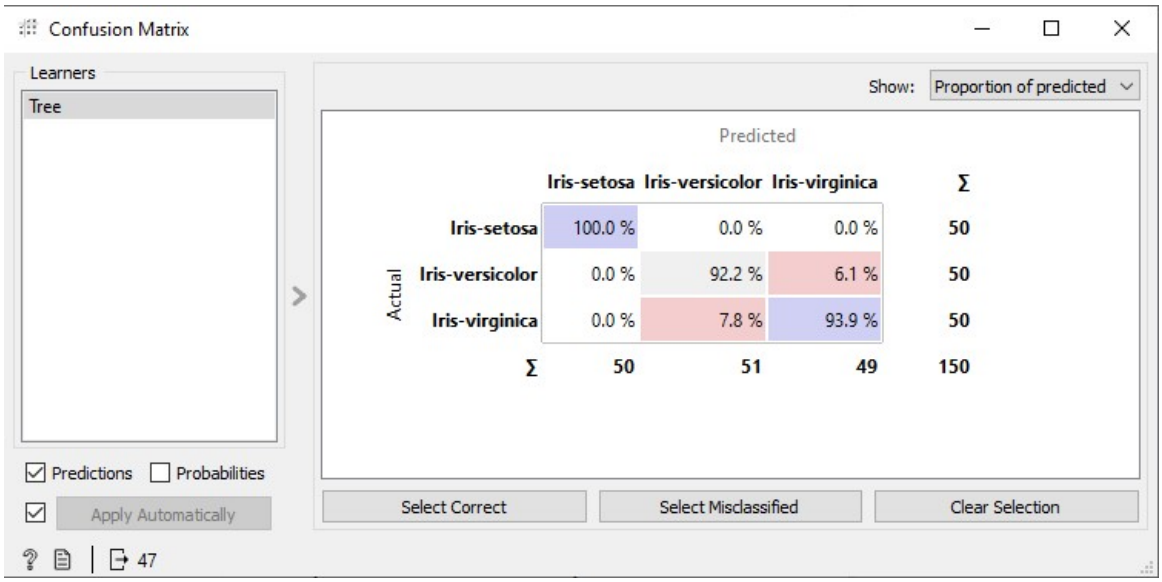


Figura 50. Proporción de predicción en Orange (Elaboración propia).

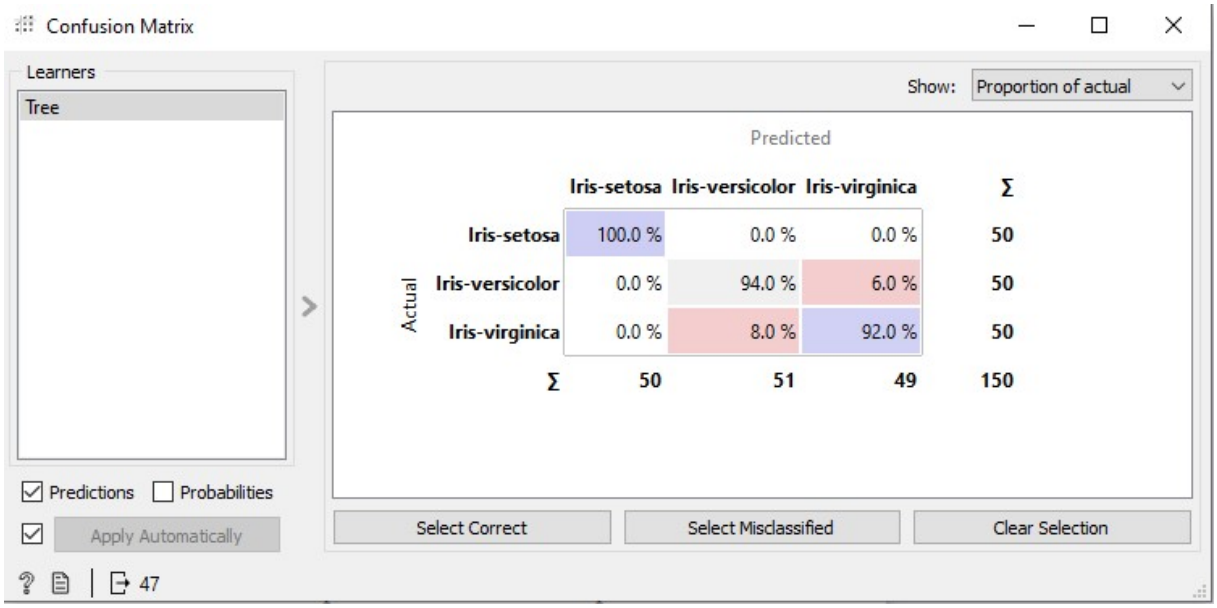


Figura 51. Proporción real en Orange (Elaboración propia).

Por último, en *Test & Score* (Figura 52) aparecen datos importantes. Son los resultados de validación cruzada que hemos utilizado. La segunda columna, denominada AUC, informa sobre la proporción de instancias de datos clasificadas correctamente. La precisión de la clasificación fue de 96,5%.

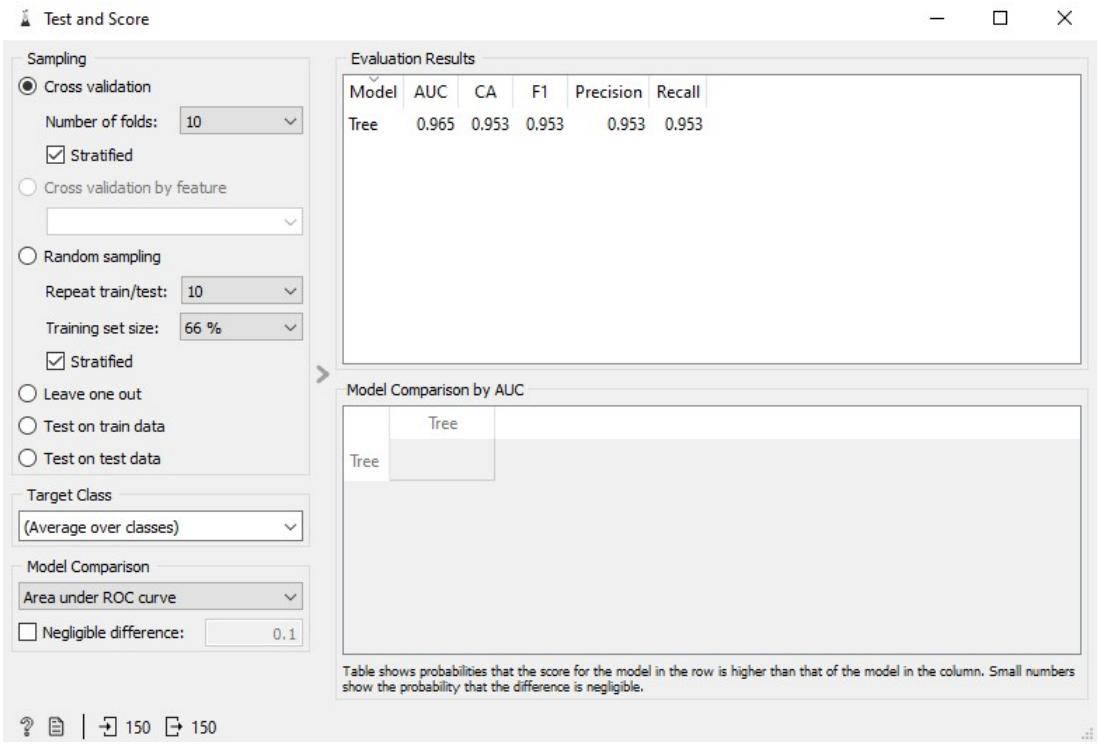


Figura 52. Test & Score Algoritmo C4.5 en Orange (Elaboración propia).

Después de realizar todos los análisis correspondientes podemos sacar un *Report* con todos los resultados obtenidos del entrenamiento. Para ello, se debe ir a cada uno de los widgets y hacer *Click* en el icono de *Report* (ver Figura 53).

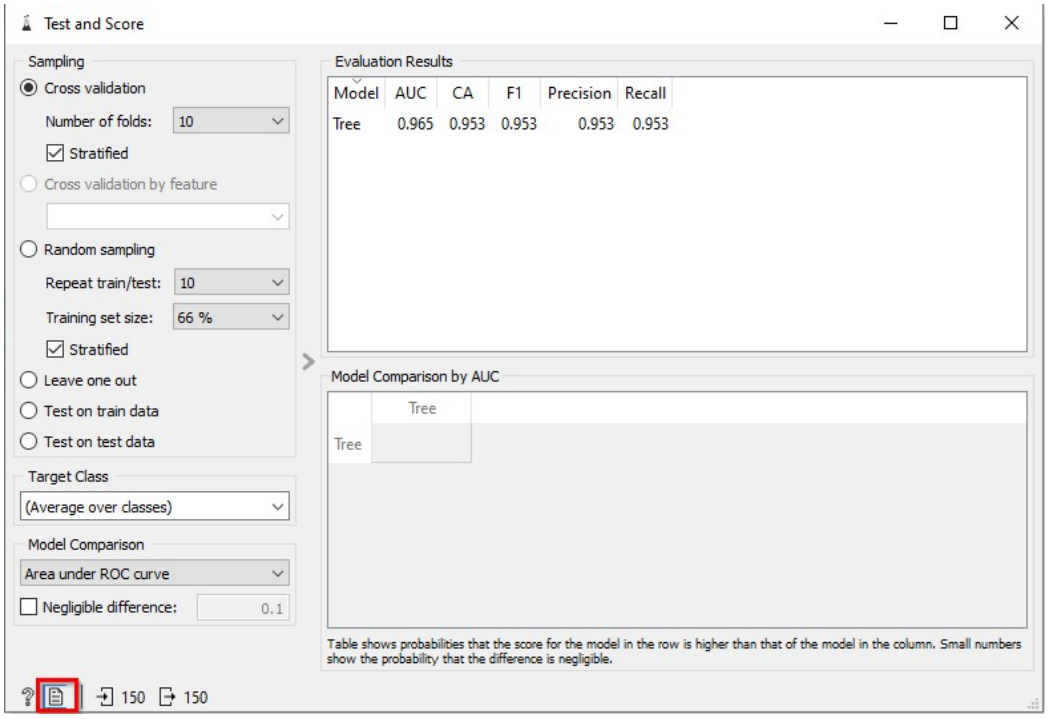


Figura 53. Icono Report en Orange (Elaboración propia).

Una vez seleccionados cada uno de los resultados al *Report*, estos se pueden ver conjuntamente tal y como aparece en la Figura 54:

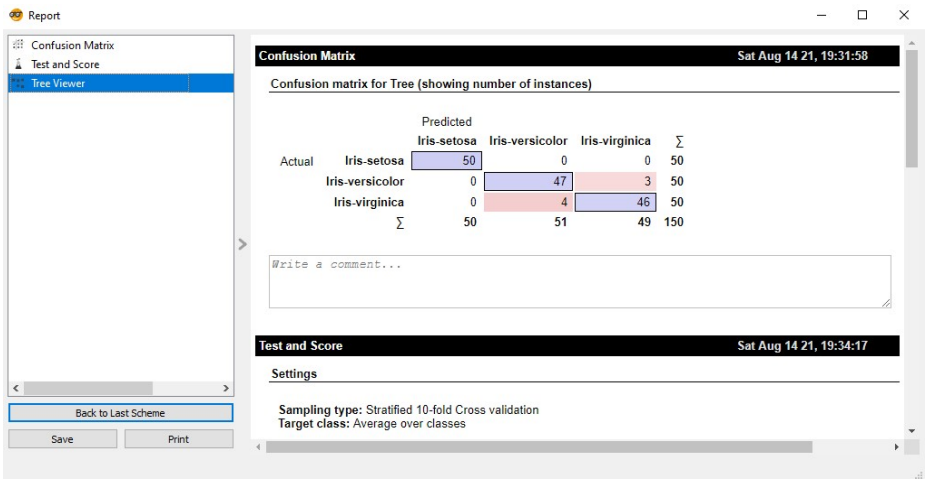


Figura 54. Reports en Orange (Elaboración propia).

Después de haber hecho este análisis con el algoritmo C4.5 en Orange se observa como se crea el flujo del algoritmo. Después como se ha visualizado el árbol con los resultados junto con la Matrix Confusion, y por último la extracción de los Reports elegidos con los resultados. Se ha comprobado que en términos de intuición, según los autores (Al-Odan & Saud, 2015), (Jović, Brkić, & Bogunović, 2014) Orange es más intuitiva y simple de entender que Weka por su interfaz gráfica, ya que es visualmente atractiva. También, se ha comprobado términos de accesibilidad, según los autores (Al-Odan & Saud, 2015) (Jović, Brkić, & Bogunović, 2014) (Ranjan & Agarwal, 2017) que al navegar por su interfaz ofrece una experiencia de usuario agradable. En términos de interpretabilidad, amigabilidad y explicabilidad del algoritmo ha sido bueno ya que a medida que se han ido observando los resultados se han ido sacando las conclusiones del entrenamiento.

5.2 Prueba con Algoritmo Random Forest

5.2.1 Weka

Se utilizará para el algoritmo Random Forest el fichero **breast-cancer (cáncer de mama)** que se encuentra, como se ha comentado con anterioridad, en el directorio *Data* que proporciona

Weka. Dicho conjunto de datos tendrá una extensión *.arff* para luego ser procesado en la herramienta. En la Figura 55 aparece el contenido de dicho fichero.

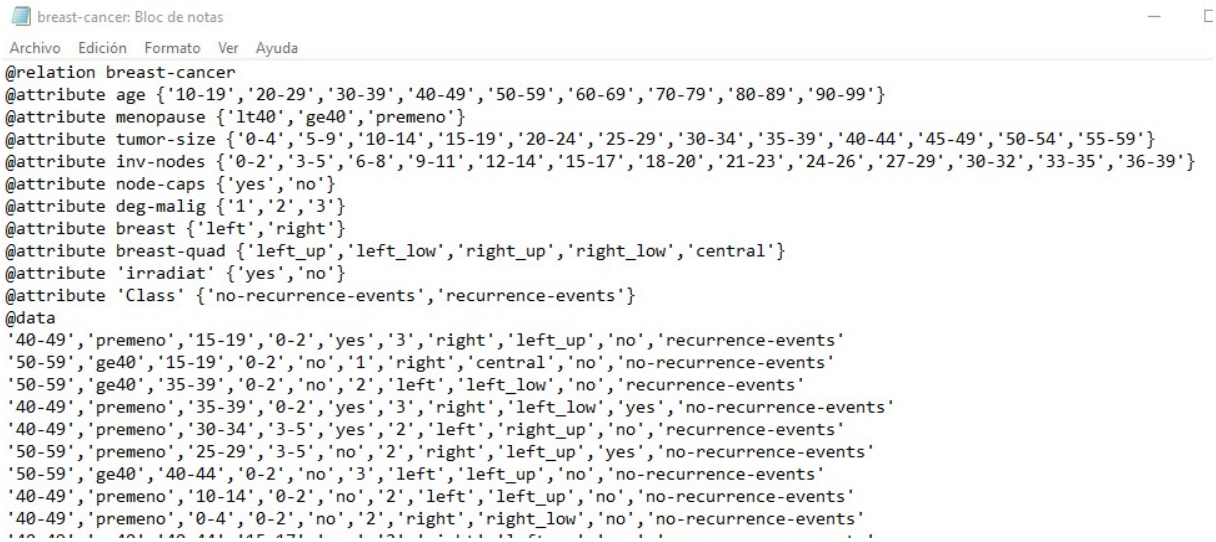


Figura 55. Dataset breast-cancer (Elaboración propia).

Este mismo conjunto de datos se carga en la herramienta Weka en la ventana *Explorer* obteniendo la siguiente vista reflejada en la Figura 56.

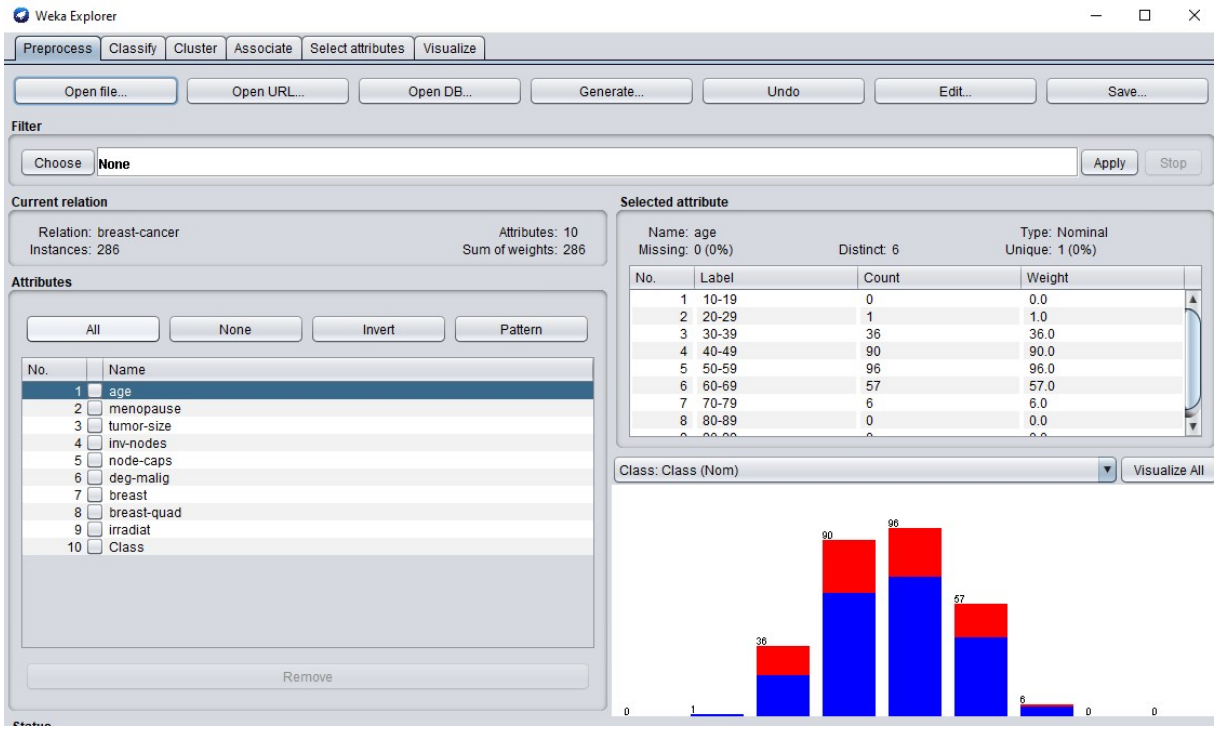


Figura 56. Dataset breast-cancer cargado en Weka (Elaboración propia).

Una vez cargado el conjunto de datos *breast-cancer* elegimos el algoritmo *Random Forest* que aparece en la ventana *Classify* y se incia el proceso mostrado en la Figura 57.

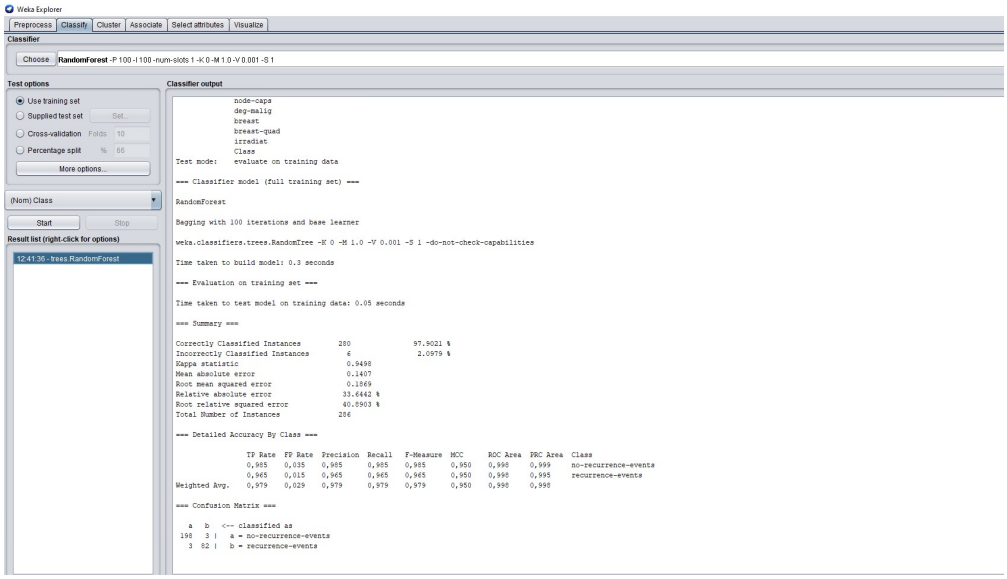


Figura 57. Resultados Random Forest breast-cancer en Weka (Elaboración propia).

Aplicado el algoritmo *Random Forest*, el informe correspondiente en Weka nos proporciona datos importantes. Podemos ver que el porcentaje de acierto del método es optimista (Figura 58), no obstante debemos de realizar varias predicciones ya que se han utilizado todos los datos de entrenamiento y sería necesario aplicar la validación cruzada = 10 o *aplicar percentaje split* para saber si estos resultados son los que esperamos.

```
Correctly Classified Instances         280               97.9021 %
Incorrectly Classified Instances         6                2.0979 %
```

Figura 58. Instancias Clasificadas Random Forest en Weka (Elaboración propia).

En la matriz de confusión se ve claramente aquellas instancias que han sido clasificadas erróneamente y correctamente (Figura 59):

```
=== Confusion Matrix ===

   a   b   <-- classified as
 198   3 |   a = no-recurrence-events
   3  82 |   b = recurrence-events
```

Figura 59. Confusion Matrix Random Forest en Weka (Elaboración propia).

Aplicando la validación cruzada = 10 se observa (ver la Figura 60) el porcentaje de acierto del método es más bajo que en el anterior análisis.

```
Correctly Classified Instances         199               69.5804 %
Incorrectly Classified Instances        87               30.4196 %
```

Figura 60. Instancias Clasificadas Random Forest validación cruzada = 10 en Weka (Elaboración propia).

Y que además, en la matriz de confusión se observan aquellas instancias que han sido clasificadas erróneamente y correctamente (ver Figura 61).

```
=== Confusion Matrix ===

   a   b   <-- classified as
 175  26 |   a = no-recurrence-events
  61  24 |   b = recurrence-events
```

Figura 61. Confusion Matrix Random Forest validación cruzada = 10 en Weka (Elaboración propia).

Por lo tanto, se puede concluir que se interpretan los resultados obtenidos: 1) en el entrenamiento sin validación cruzada, tres versiones no han sido clasificadas como evento de recurrencia y a su vez tres versiones no han sido clasificadas como evento de no recurrencia, y 2) con validación cruzada, sesenta y una versiones no han sido clasificadas como evento de recurrencia y a su vez veintiseis versiones no han sido clasificadas como evento de no recurrencia. Esto quiere decir que el primer análisis es más optimista que el segundo. Para el algoritmo *Random Forest* no se puede generar el árbol, lo cual es un punto negativo para continuar con el análisis, tal y como comentaban (Jović, Brkić, & Bogunović, 2014), (Ratra & Gulia, 2020), en sus investigaciones, Weka es poco amigable. Por otra parte en término de rendimiento ha sido bueno.

5.2.2 Orange

En Orange existe como tal el *widget Random Forest*, por lo que será fácil implementarlo. De nuevo, se utilizan los *widgets* para crear este algoritmo y que nos dé los resultados de la predicción. En Orange hay documentación[30] con ejemplos sobre *Random Forest* para poder

[30]https://orange3.readthedocs.io/projects/orange-visual-programming/en/latest/widgets/model/randomforest.html

implementar este algoritmo. Por tanto, seguiremos estos pasos. Se utilizará el mismo fichero *breast-cancer.tab* que se manejó en Weka anteriormente. En la Figura 62 se puede ver el flujo resultante.

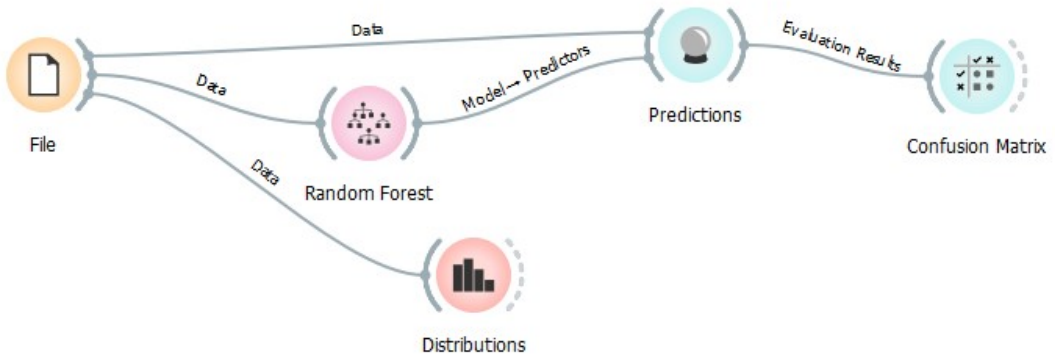


Figura 62. Flujo con widgets Random Forest en Orange (Elaboración propia).

En el flujo mostrado en la Figura 62 está añadido la predicción, la distribución y la confusión matrix para obtener los resultados y observar qué instancias específicas se clasificaron correctamente y cuáles no.

La Figura 63 muestra la matriz de confusión para el modelo *Random Forest* entrenado. Cada fila corresponde a una clase mientras que las columnas representan las clases predichas. Podemos ver que 32 instancias de evento no recurrente fueron clasificadas erróneamente como evento recurrente. La columna de la derecha con el símbolo sumatorio ∑ contiene el número de instancias de cada clase. En este caso son 201 de evento no recurrente y 85 de evento recurrente del cáncer de mama y en la fila inferior se muestra el número de instancias clasificadas en cada clase como, por ejemplo, 58 clasificadas en evento recurrente.

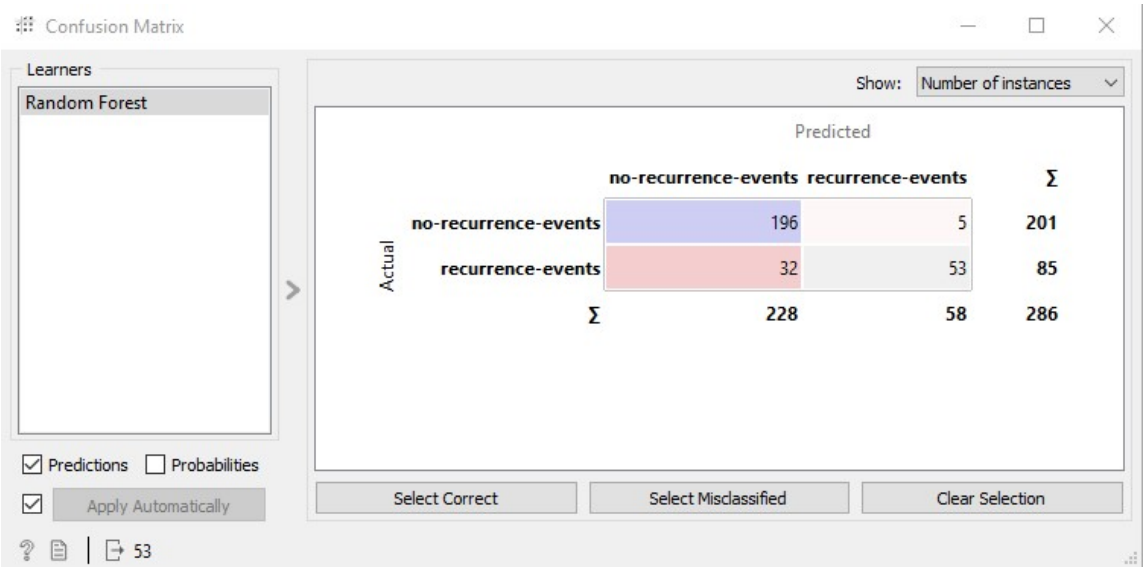


Figura 63. Confusion Matrix Algoritmo Random Forest en Orange (Elaboración propia).

Además, en *Show* seleccionamos los datos que nos gustaría observar en la matriz:

- **Número de instancias:** nos muestra las instancias clasificadas correcta e incorrectamente.

- **Proporción de predicción:** muestra cuantas instancias son clasificadas como, por ejemplo, evento recurrente en esa clase (ver Figura 64). La muestra del 91.4% de evento recurrente predicho era verdaderamente esa clase y el 14.0% de evento recurrente predicho donde se encuentra el evento no recurrente.

- **Proporción real:** estas muestran la relación opuesta (Figura 65). La muestra real del 97.5% se predijo como evento no recurrente.

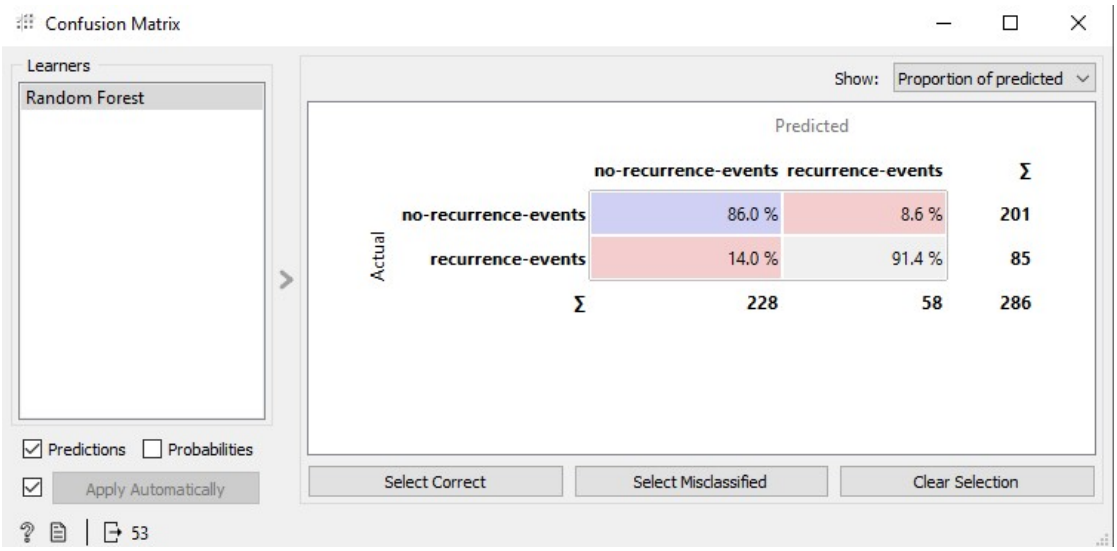


Figura 64. Proporción de predicción en Orange (Elaboración propia).

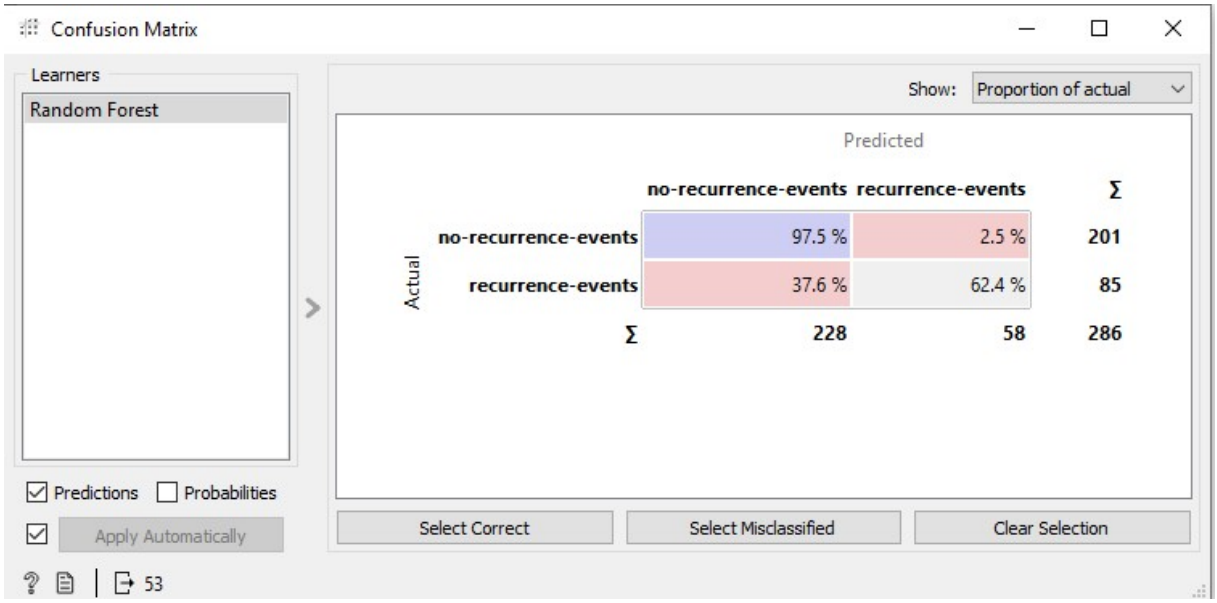


Figura 65. Proporción real en Orange (Elaboración propia).

En el *widget Predictions* (Figura 66) aparecen también datos importantes como los resultados de la predicción. En la segunda columna nombrada AUC, se informa sobre la proporción de instancias de datos predichas correctamente. El porcentaje de esta predicción es de 92,9%.

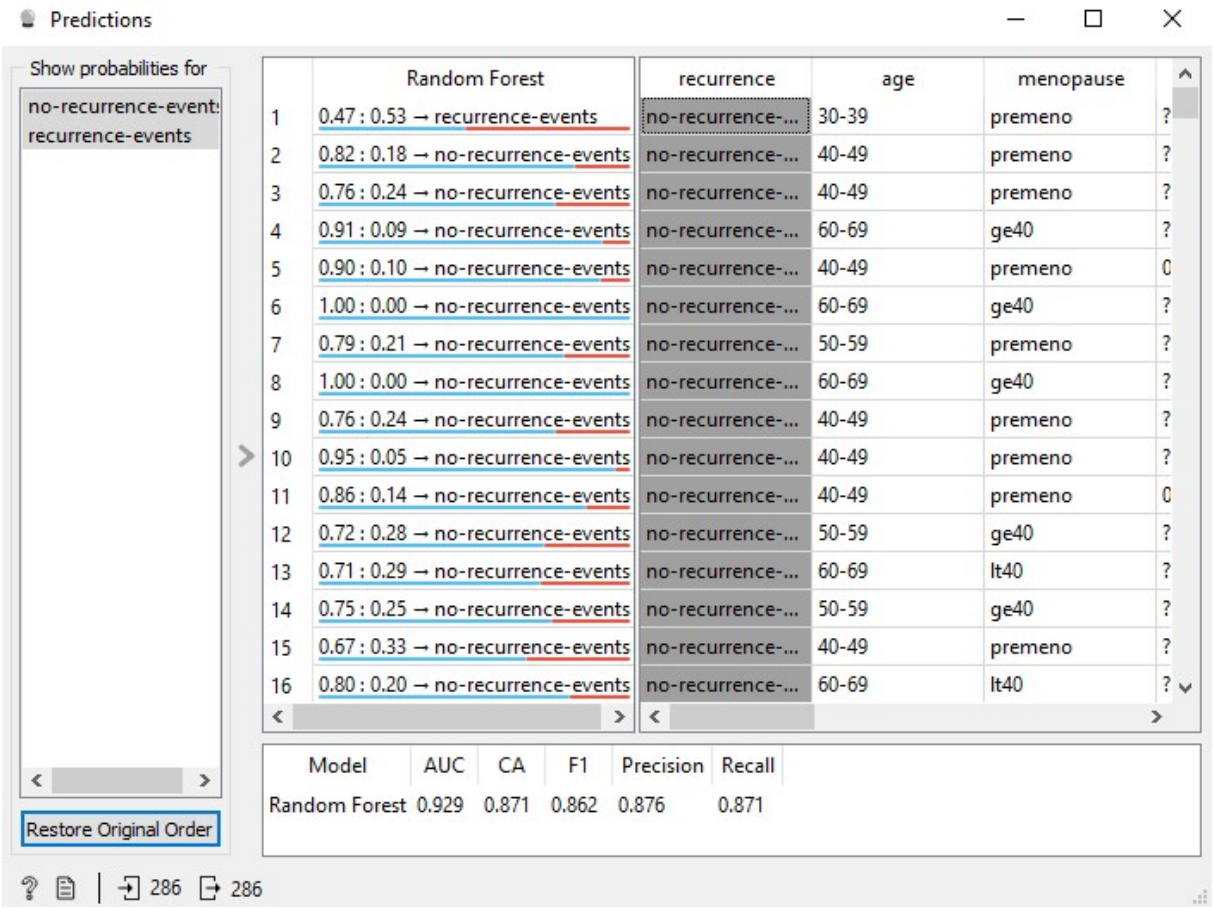


Figura 66. Prediction Random Forest en Orange (Elaboración propia).

Por último, se ha utilizado el *widget Distribution* (Figura 67) para observar cual es la distribución del conjunto de datos. Este *widget* nos muestra un histograma clasificado por edad y frecuencia en los eventos recurrente y eventos no recurrente. Observamos que entre los 40-49 años la incidencia de los eventos recurrentes es mayor que en el resto de edades y que los eventos recurrentes entre 50-59 años es mayor que el resto. Esto nos hace tener una idea de en qué edades incide más el cáncer respecto al resto. Además, podemos crear un *Report* incluyendo todos los resutlados obtenidos en el análisis y de esta forma detallar en profundidad todas las conclusiones posibles del análisis.

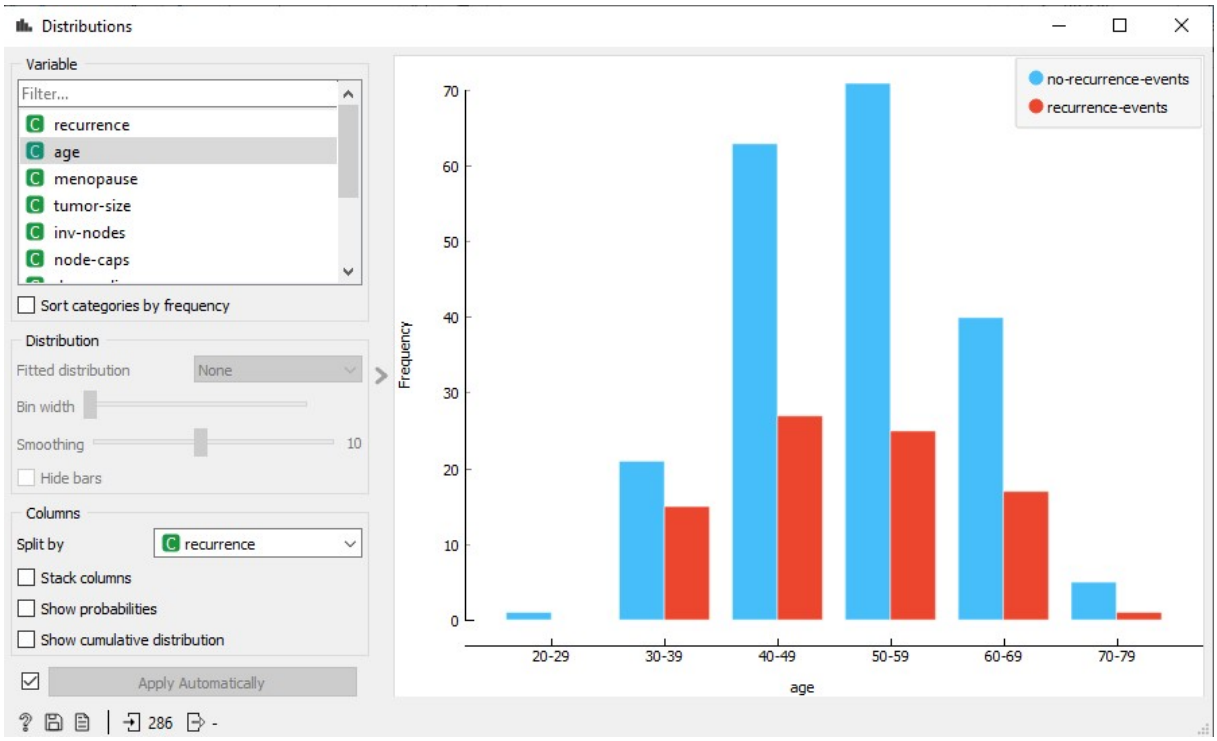


Figura 67. Distribution Random Forest en Orange (Elaboración propia).

Concluyendo, observamos que en este análisis se han interpretado y explicado varios resultados. De ahí se han obtenido unas conclusiones sobre el conjunto de datos introducido. Se ha comprobado, y según los autores (Jović, Brkić, & Bogunović, 2014), (Ratra & Gulia, 2020), (Al-Odan & Saud, 2015) Orange presenta facilidades al navegar por todas sus funcionalidades, y una interfaz bien estructurada con diferentes características por lo que es intuitiva, simple de entender y comprender. Cabe destacar que el rendimiento, y según el autor (Al-Odan & Saud, 2015), en Orange no se mide el tiempo de ejecución porque no tiene la funcionalidad incorporada, en cambio Weka si la tiene incorporada. Por lo que Weka en términos de rendimiento es superior a Orange.

5.3 Prueba con algoritmo K-Means

5.3.1 Weka

Principalmente saber que el algoritmo *K-Means* en Weka se le conoce como *Simple K-Means*. Es por eso por lo que se seleccionará esta opción para esta prueba. Se utilizará para la prueba del algoritmo *K-Means* en Weka el fichero **Tasa de actividad, empleo y paro** que se

encuentra público en la página oficial del Gobierno de España[31]. Este conjunto de datos tiene una extensión *.csv* para luego ser procesado en la herramienta.

Hay que tener en cuenta que se agrupan los datos mediante el centroide. Conociendo el número de clases por las que se tendrían que agrupar los datos, para esta prueba serían uno por clase ($k = 3$), pero en este caso no se introducirá todavía para ver cómo es la evolución de los resultados y también cómo se comporta Weka. No siempre se conocen las clases de agrupamiento, es por eso por lo que todavía no lo aplicamos.

Introduciendo el fichero (Figura 68) y aplicando el algoritmo *K-Means* en Weka podemos visualizar los clústeres que ha descubierto el algoritmo (Figura 69). Una vez mostrados los resultados del conjunto de entrenamiento en el panel de lista de resultados, se debe hacer *Click* en el botón derecho y se selecciona **Visualize Clúster Assignments**.

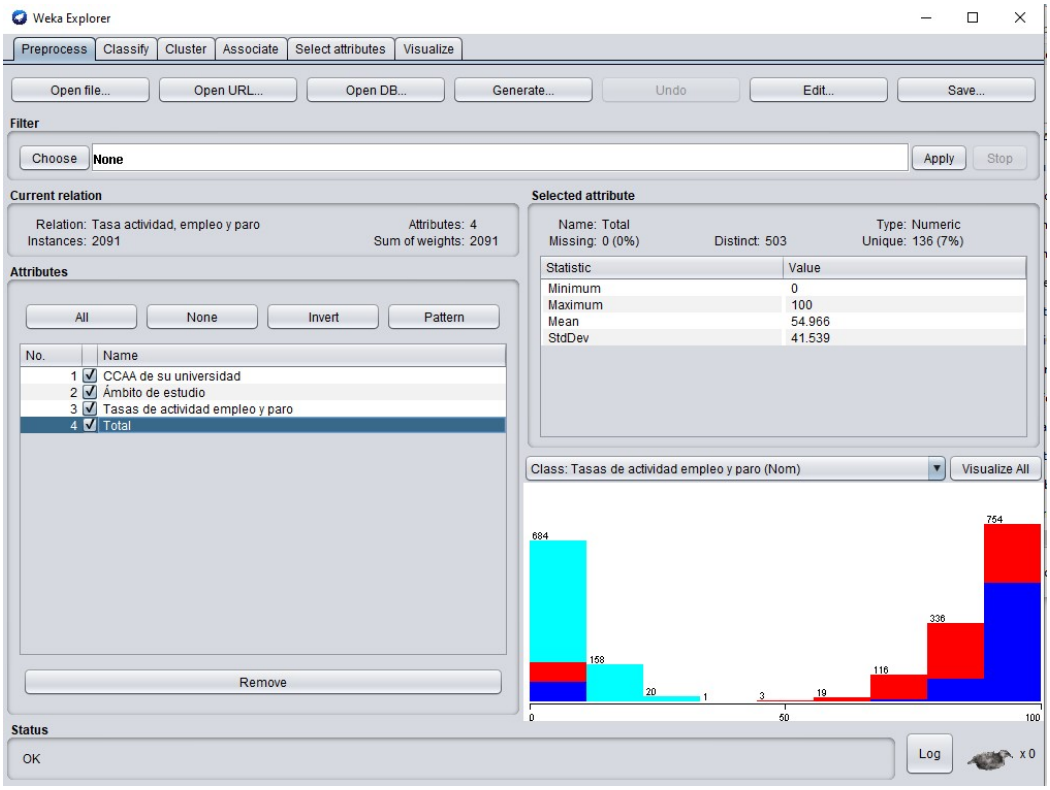


Figura 68. Dataset introducido en Weka (Elaboración propia).

[31] https://datos.gob.es/es/catalogo/ea0010587-tasas-de-actividad-empleo-y-paro-de-los-graduados-universitarios-por-ccaa-de-su-universidad-y-ambito-de-estudio-identificador-api-t13-p100-2019-p02-l0-03011-px

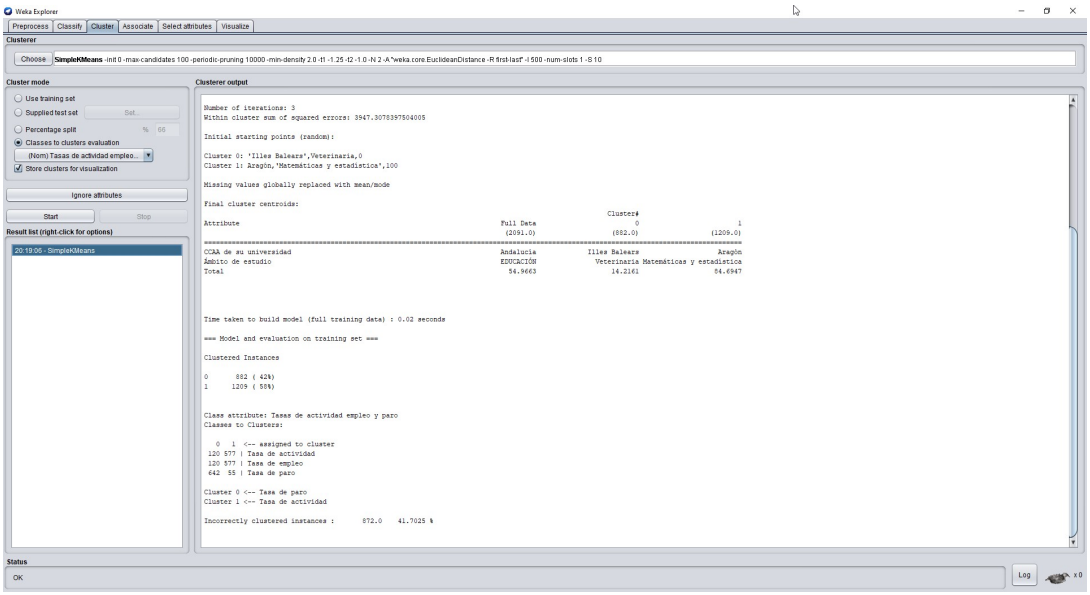


Figura 69. Resultados K-Means en Weka (Elaboración propia).

Como se comentó con anterioridad en la Figura 69, no se ha introducido el número de clases definido. En los resultados se observa como se han construido dos clústeres cada uno con el número total de instancias. Para el clúster 0 sería un total de 882 instancias y para el clúster 1 un total de 1209 instancias. Además, observamos en los resultados, el número de iteraciones es de 3, la suma de errores dentro del grupo es bastante alta y que los puntos de partida iniciales (aleatorios) son en el Clúster 0: 'Illes Balears',Veterinaria,0. Clúster 1: Aragón,'Matemáticas y estadística',100. Una vez observados todos estos resultados se visualizará como se han agrupado los diferentes clústeres (ver Figura 70) y el diagrama de dispersión (ver Figura 71). A simple vista en la Figura 71 se pueden diferenciar los dos clústeres; en el clúster 0 aparecen las instancias mas frecuentes en verde (Tasa de paro) y algunas instancias de rojo (Tasa de empleo) y azul (Tasa de actividad). En el clúster 1 aparecen las instancias mas frecuentes; las instancias en color rojo (Tasa de empleo) y color azul (Tasa de actividad), seguidas de algunas instancias de color verde (Tasa de paro). Además, si navegamos por los diferentes gráficos se observa cómo estan distribuidas las instancias por CCAA (Figura 72) o por ámbito de estudios (Figura 73).

Figura 70. Visualización Clústeres K-Means en Weka (Elaboración propia).

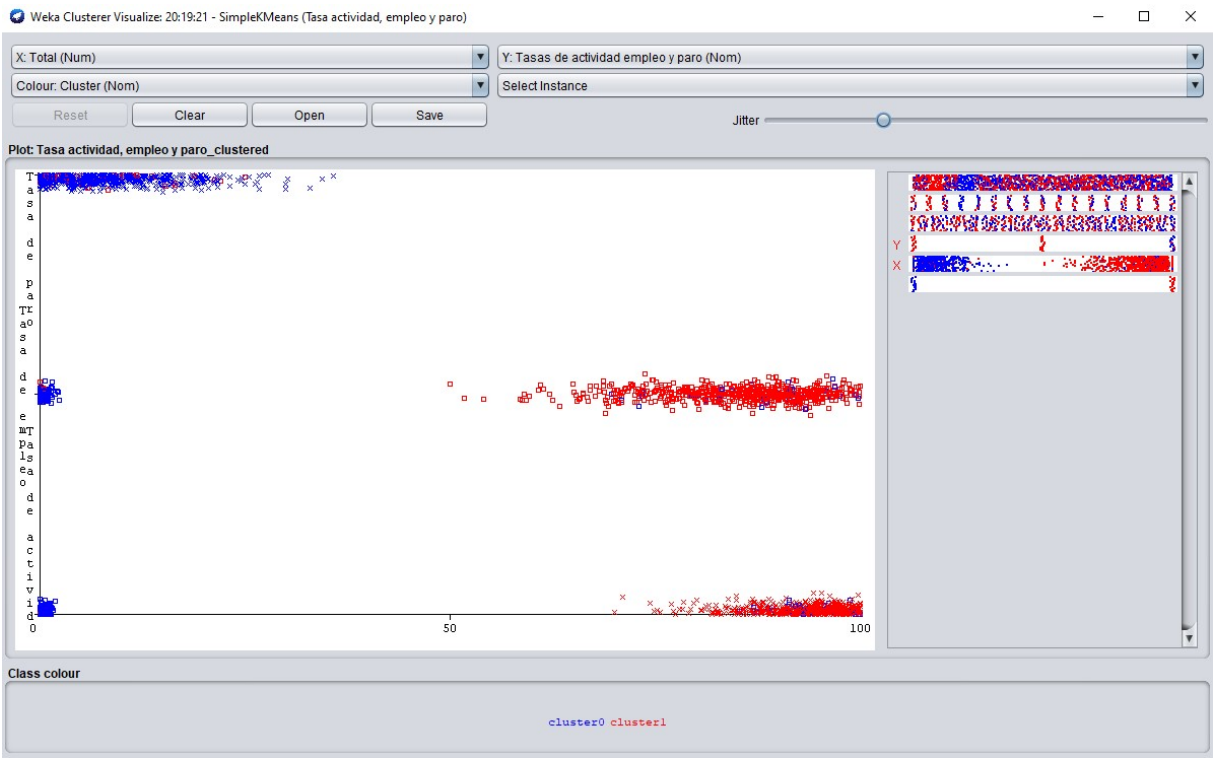


Figura 71. Diagrama de dispersión de tasa de actividad, empleo y paro K-Means en Weka (Elaboración propia).

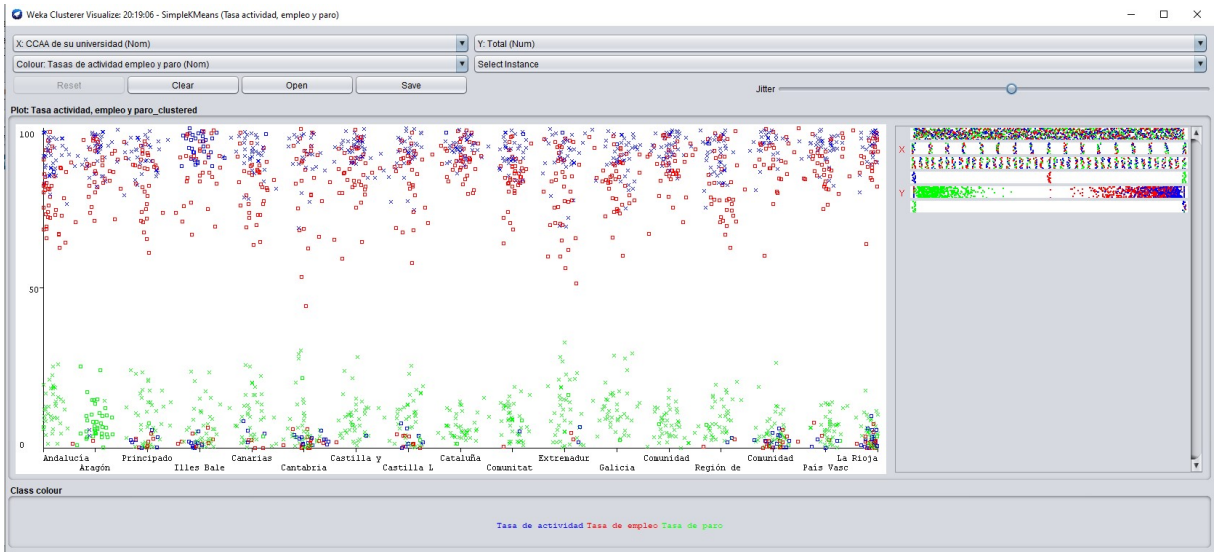


Figura 72. Visualización por CCAA K-Means en Weka (Elaboración propia).

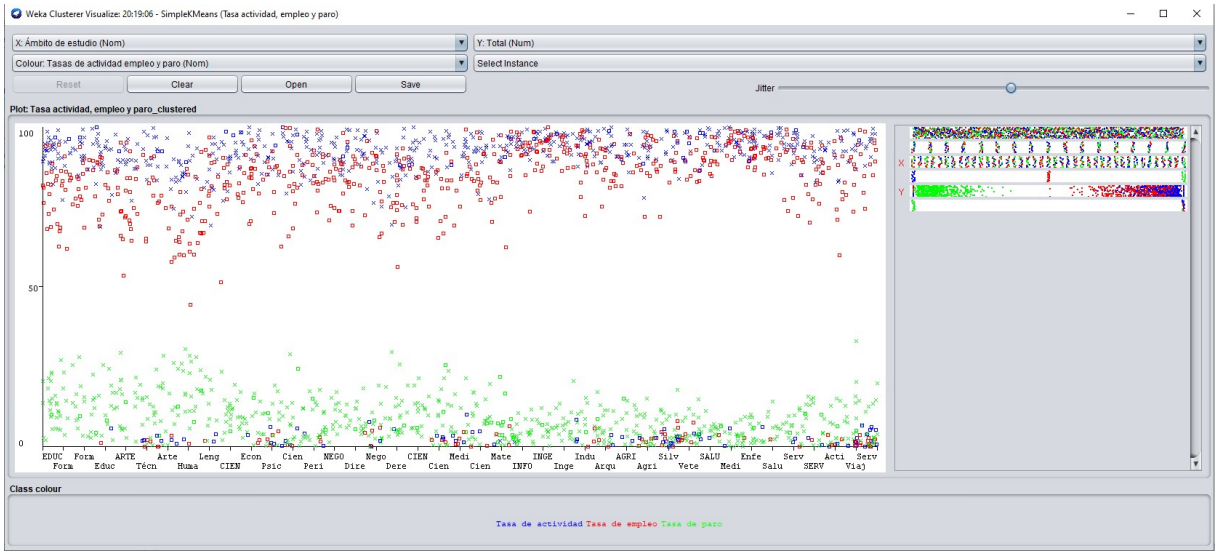


Figura 73. Visualización por ámbito de Estudio K-Means en Weka (Elaboración propia).

Otro dato característico en los resultados es identificar qué casos de *Tasa de actividad, Tasa de empleo y Tasa de paro* han sido clasificados de forma correcta. Se observa cómo el clúster 0 está identificado como *Tasa de paro* y el clúster 1 como *Tasa de actividad*. Los casos de T*asa de paro* son de 642 instancias clasificadas correctamente y los casos de *Tasa de actividad* son de 577 instancias clasificadas correctamente. Por último, las instancias que se han agrupado incorrectamente son de 872, lo que supone un 41,70% de los datos (Figura 74).

```
Class attribute: Tasas de actividad empleo y paro
Classes to Clusters:

   0   1  <-- assigned to cluster
 120 577 | Tasa de actividad
 120 577 | Tasa de empleo
 642  55 | Tasa de paro

Cluster 0 <-- Tasa de paro
Cluster 1 <-- Tasa de actividad

Incorrectly clustered instances :       872.0    41.7025 %
```

Figura 74. Clústeres definidos K-Means en Weka (Elaboración propia).

Ahora probemos con (k = 3) tal y como aparece en la Figura 75. Este cambio se hace antes de ejecutar el algoritmo con el conjunto de datos del entrenamiento.

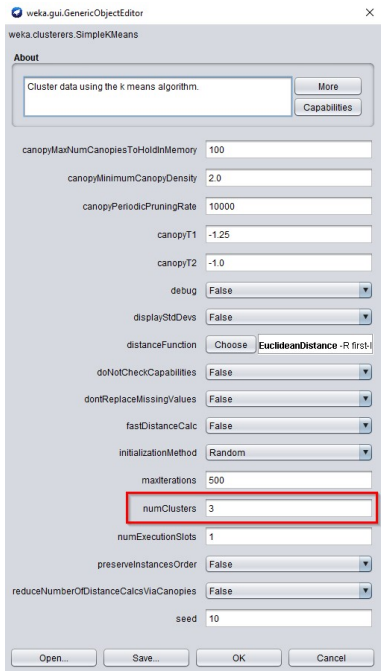


Figura 75. NumClusters k=3 K-Means en Weka (Elaboración propia).

Observando los resultados obtenidos en la Figura 76 se distinguen 3 clústeres. En el clúster 0, 817 instancias, en el clúster 1, 1064 instancias y en el clúster 2, 210 instancias. Además, se observa en los resultados: el número de iteraciones, 8; la suma de errores dentro del grupo es más baja que en la prueba anterior y los puntos de partida iniciales (aleatorios) son: Clúster 0: 'Illes Balears', Veterinaria, 0. Clúster 1: Aragón, 'Matemáticas y estadística', 100. Clúster 2: Canarias, Humanidades, 'Tasa de actividad', 80.2.

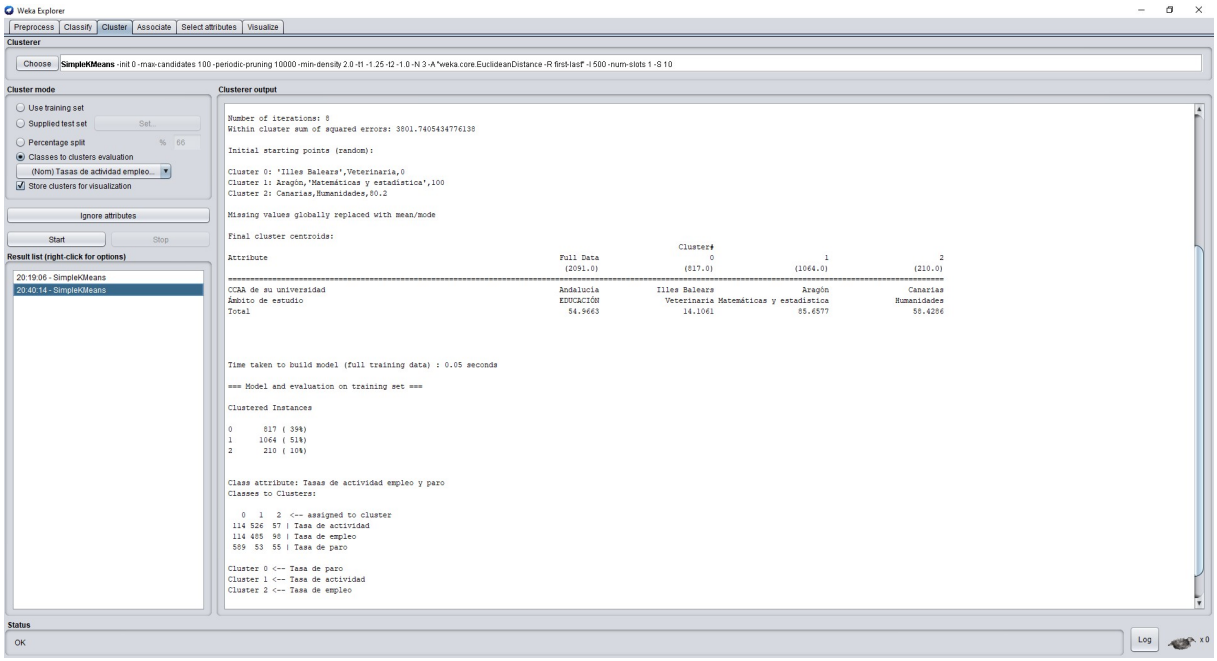


Figura 76. Resultado k=3 K-Means en Weka (Elaboración propia).

Veamos ahora en la Figura 77 como están agrupados los clústeres y su diagrama de dispersión en la Figura 78.

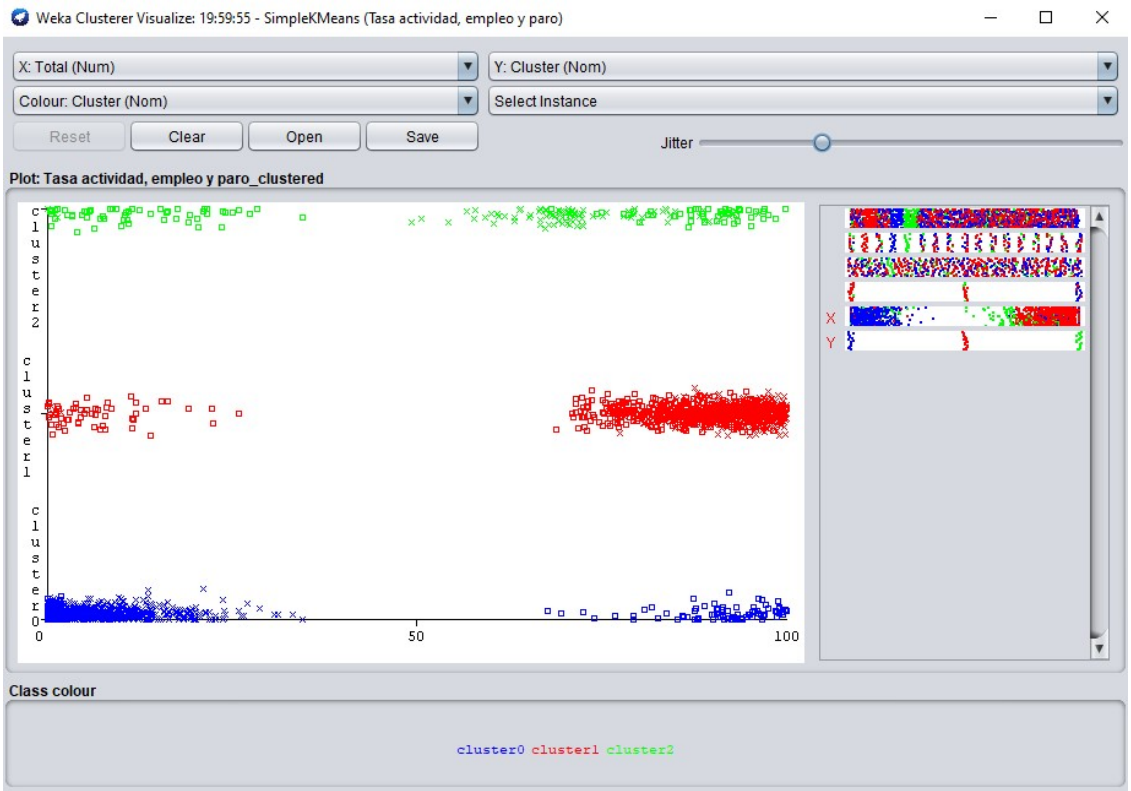


Figura 77. Visualización Clústeres k = 3 K-Means en Weka (Elaboración propia).

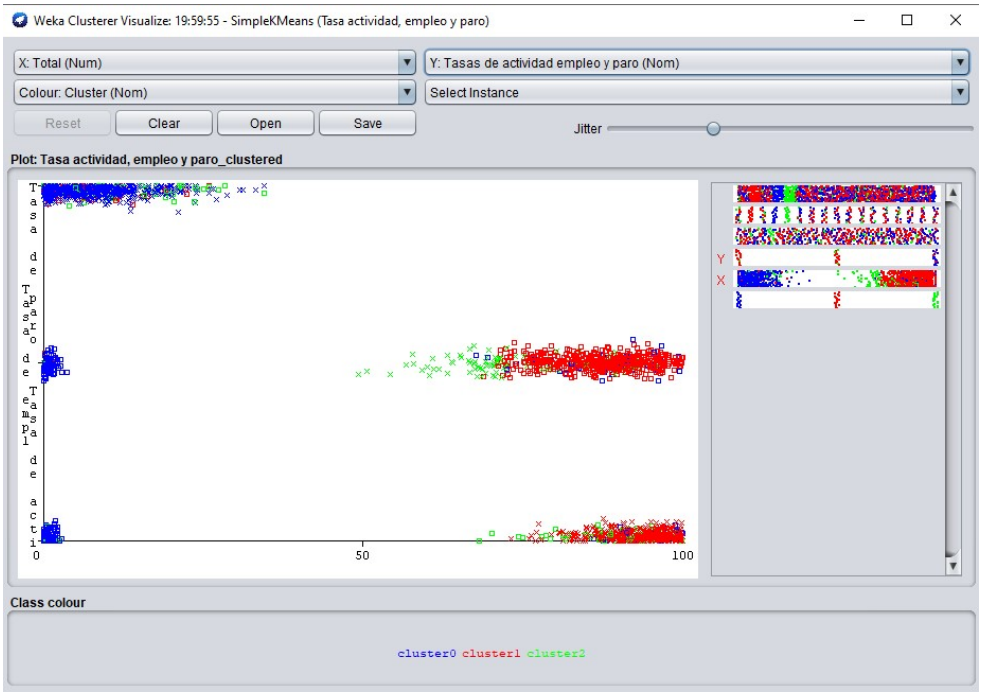


Figura 78. Diagrama de dispersión de tasa de actividad, empleo y paro k = 3 K-Means en Weka (Elaboración propia).

Observamos que aparecen 3 clústeres perfectamente definidos y, en cada uno de ellos, se visualiza un tipo de instancias frecuentes. En el clúster 0, las instancias más frecuentes son las de tasa de paro, en el clúster 1, las instancias de tasa de empleo y en el clúster 2, las instancias de tasa de empleo. Si navegamos por los diferentes gráficos observamos como están distribuidas las instancias por CCAA (Figura 79) o por ámbito de estudios (Figura 80) y no se aprecia diferencia con el análisis anterior.

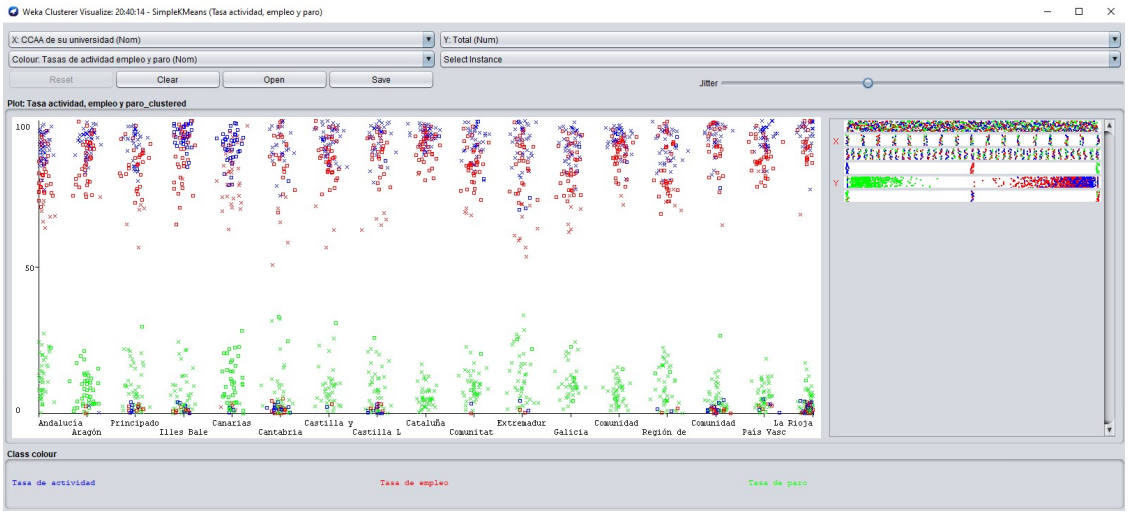


Figura 79. Visualización por CCAA k = 3 K-Means en Weka (Elaboración propia).

Figura 80. Visualización ámbito de Estudio k= 3 K-Means en Weka (Elaboración propia).

Por último, se observa en los resultados los casos de *Tasa de actividad, Tasa de empleo y Tasa de paro* han sido clasificados de forma correcta. Se observa cómo el clúster 0 está identificado como *Tasa de paro*, el clúster 1 como T*asa de actividad* y el clúster 2 como *Tasa de empleo*. Los casos de tasa de paro son de 589 instancias clasificadas correctamente, los casos de tasa de actividad son de 526 instancias clasificadas correctamente y los casos de tasa de empleo son 98 instancias clasificadas correctamente. Por último, las instancias que se han agrupado incorrectamente son 878, más que en el análisis anterior lo que supone un 41,98% de los datos (ver Figura 81).

```
Class attribute: Tasas de actividad empleo y paro
Classes to Clusters:

   0   1   2  <-- assigned to cluster
 114 526  57 | Tasa de actividad
 114 485  98 | Tasa de empleo
 589  53  55 | Tasa de paro

Cluster 0 <-- Tasa de paro
Cluster 1 <-- Tasa de actividad
Cluster 2 <-- Tasa de empleo

Incorrectly clustered instances :       878.0    41.9895 %
```

Figura 81. Clústeres definidos k = 3 K-Means en Weka (Elaboración propia).

Se ha comprobado, y según los autores (Ameen*, Bajeh, Adesiji, Balogun, & Mabayoje, 2018) (Dušanka, Darko, Srdjan, Marko, & Teodora, 2017) la interfaz de usuario de Weka es flexible y totalmente funcional lo que hace más fácil el acceso a sus componentes principales. También, hace más fácil su utilización y desempeño. Además, la ejecución del algoritmo ha sido buena por lo que su rendimiento ha sido bueno. Se observa en este análisis varios resultados. De ahí son sacadas unas conclusiones sobre el conjunto de datos. Interpretando los resultados obtenidos: 1) en el primer análisis no se tuvo en cuenta el número de clases, y 2) en el segundo análisis sí se aplicó el número de clases exactas. Comparando y comprendiendo el primer análisis con el segundo se observa que las instancias clasificadas incorrectamente son mayores en el segundo que en el primero. Por lo que los resultados del primer análisis son optimistas.

5.3.2 Orange

En Orange existe el *widget K-Means*, por lo que será fácil implementarlo. De igual forma tendremos que utilizar los widgets para crear este algoritmo y que nos dé los resultados del entrenamiento. Además, en Orange hay documentación[32] con ejemplos sobre K-Means para poder implementar este algoritmo. Por tanto, se seguirán estos pasos. Se utilizará el fichero **tasa de actividad, empleo y paro.csv** que se manejó en Weka anteriormente. El *widget K-Means* aplica el algoritmo de agrupación a los datos y genera un nuevo conjunto de datos en el que el índice de clúster se utiliza como atributo de la clase. Las puntuaciones de los resultados de la agrupación para varios *k* también se muestran en el *widget*. En la Figura 82 se puede ver el flujo del algoritmo K-Means resultante dividido en tres grupos, uno con la tabla de datos donde se pueden ver qué instancias entraron en cada grupo, otro con un gráfico de dispersión con los puntos coloreados, de acuerdo con los clústeres encontrados, las filas de selección con reglas que se pueden introducir para ver su posterior distribución y por último Box Plot, llamado también diagrama de caja, que muestra la distribución de los valores de los atributos (se puede hacer más simple pero se eligió hacerlo así, con estos *widgets*, para ver con detalle todos los resultados).

[32]https://buildmedia.readthedocs.org/media/pdf/orange-visual-programming/latest/orange-visual-programming.pdf

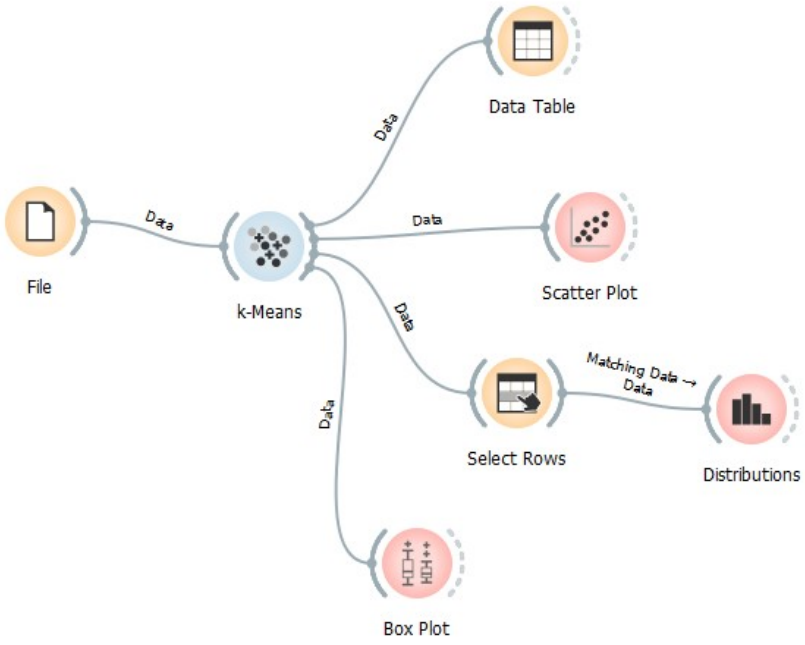


Figura 82. Flujo con widgets K-Means en Orange (Elaboración propia).

Se parte, por defecto con el números de clústeres que se van a utilizar en esta prueba. Esto se hará con el widget de *K-Means* (ver Figura 83).

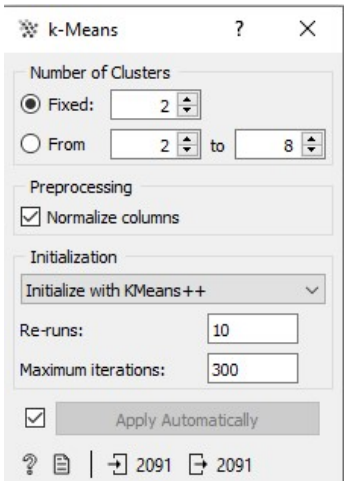


Figura 83. Opciones K-Means en Orange (Elaboración propia).

Una vez seleccionado el fichero de entrenamiento y el algoritmo, se procede a visualizar el *widget Data Table*. Este indica cuáles son los datos que se han agrupado en cada uno de los clústeres (ver Figura 84).

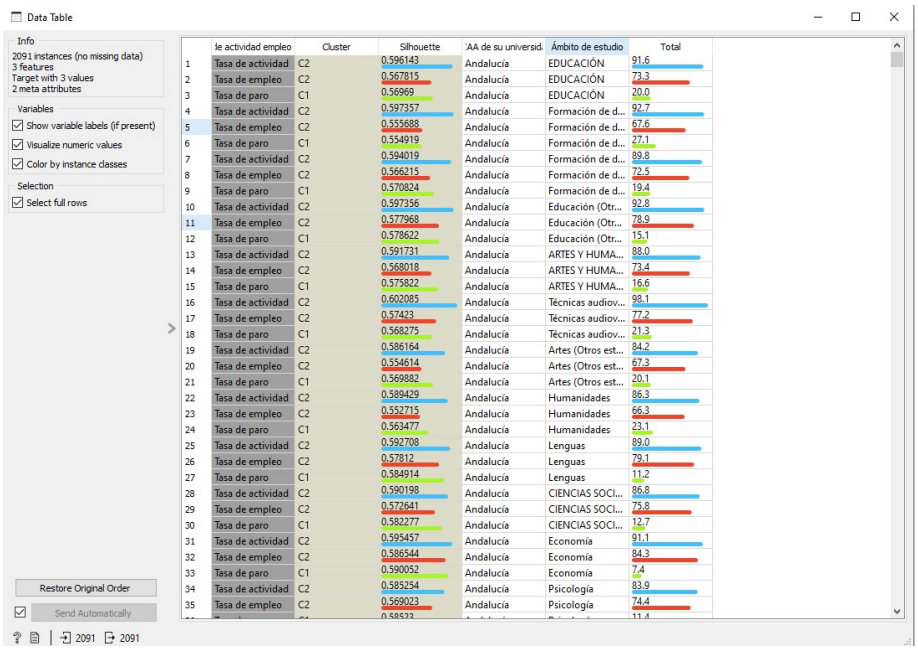


Figura 84. Data Table K-Means en Orange (Elaboración propia).

En la Figura 85 se observa el agrupamiento de los datos en el diagrama de dispersión. Se aprecian cada una de las clases identificadas por diferentes colores; verde para tasa de paro, azul para tasa de actividad y rojo para tasa de empleo. Además, se identifican que datos son los correspondientes a cada clúster, los datos del C1, su símbolo es un círculo y los del C2, una cruz. Además, se aprecia el resultado de la línea de regresión con unos valores entre 1 y -1, significa que son datos fiables. También, se observan tres zonas coloreadas en el diagrama correspondientes a las clases definidas.

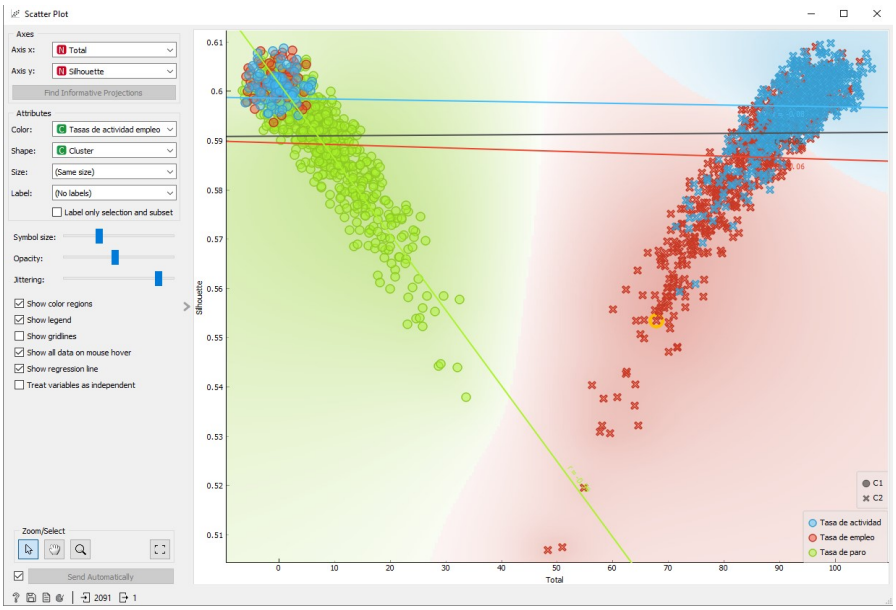


Figura 85. Diagrama de Dispersión K-Means en Orange (Elaboración propia).

Con el *widget Select Rows* se pueden seleccionar las clases individuales y tener los puntos marcados en el diagrama de distribución que se usará más adelante. En el widget *Distributions* se observa cómo estan agrupados los datos en cada clúster. Se visualizan los atributos tal y como aparece en la Figura 86. Todas las instancias de tasa de paro, 697 (en color verde) están incluidas en el clúster C1, además de 83 instancias de tasa de actividad (en color azul) y 83 instancias de tasa de empleo (en color rojo). Sin embargo, en el clúster C2 solo hay 614 instancias de tasa de empleo (en color rojo) y 614 instancias de tasa de actividad (en color azul).

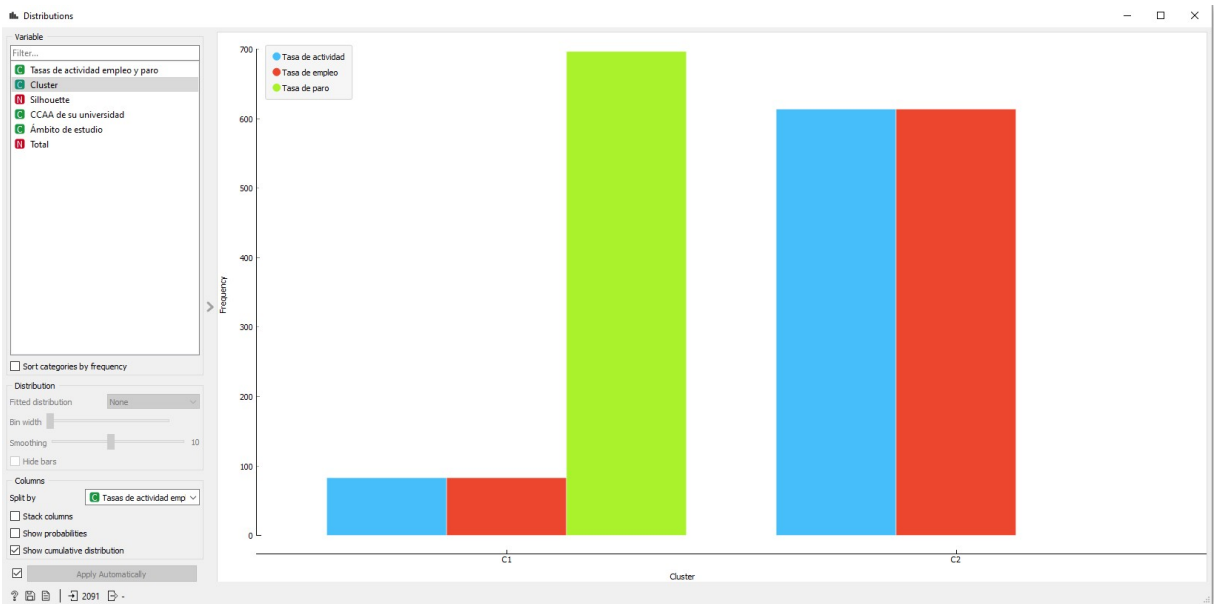


Figura 86. Distribution Clúster K-Means en Orange (Elaboración propia).

Si se desea saber el total de cada una de las clases identificadas se seleccionaría *Total* y en el recuadro de *Split by,* se seleccionaría *Tasa de actividad empleo y paro*. De este modo, se puede observar cuál es su distribución en base a la frecuencia y su total. Podemos deducir de la Figura 87 que la tasa de paro es superior que la tasa de actividad y empleo.

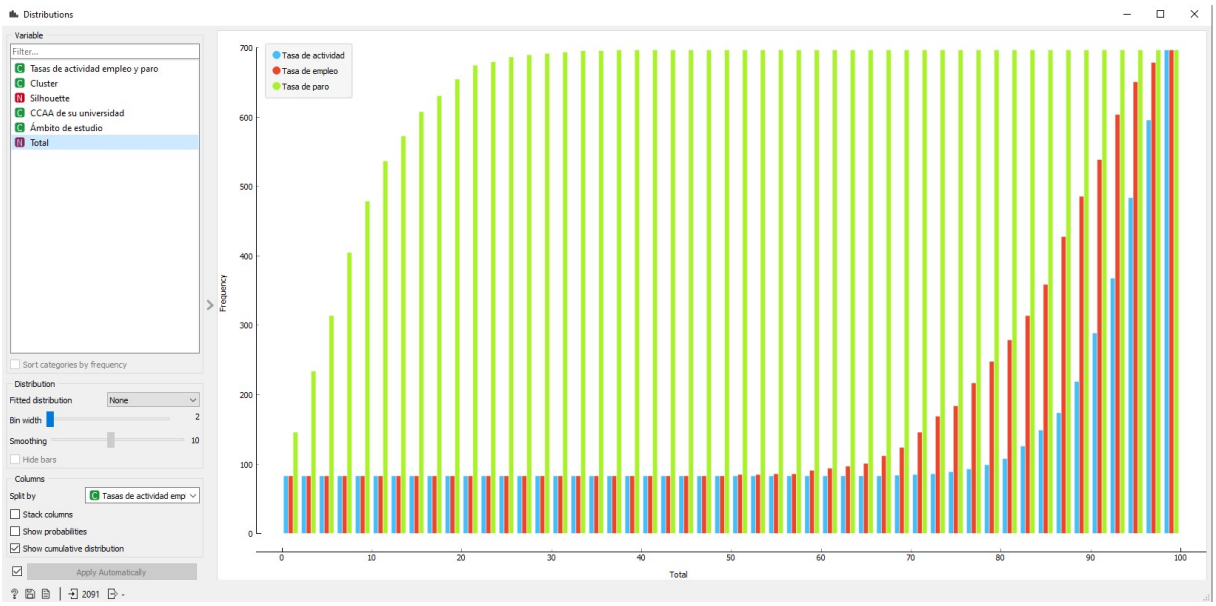


Figura 87. Distribución algoritmo K-Means en Orange (Elaboración propia).

Con el widget *Select Rows* se pueden seleccionar aquellos datos que se quieren visualizar en el gráfico de distribución. Por ejemplo, si se quieren los totales de la tasa de empleo de la Comunidad de Madrid se introducen estas condiciones en el *widget Select Rows* y después, se visualizan en el gráfico de distribución (ver Figura 88).

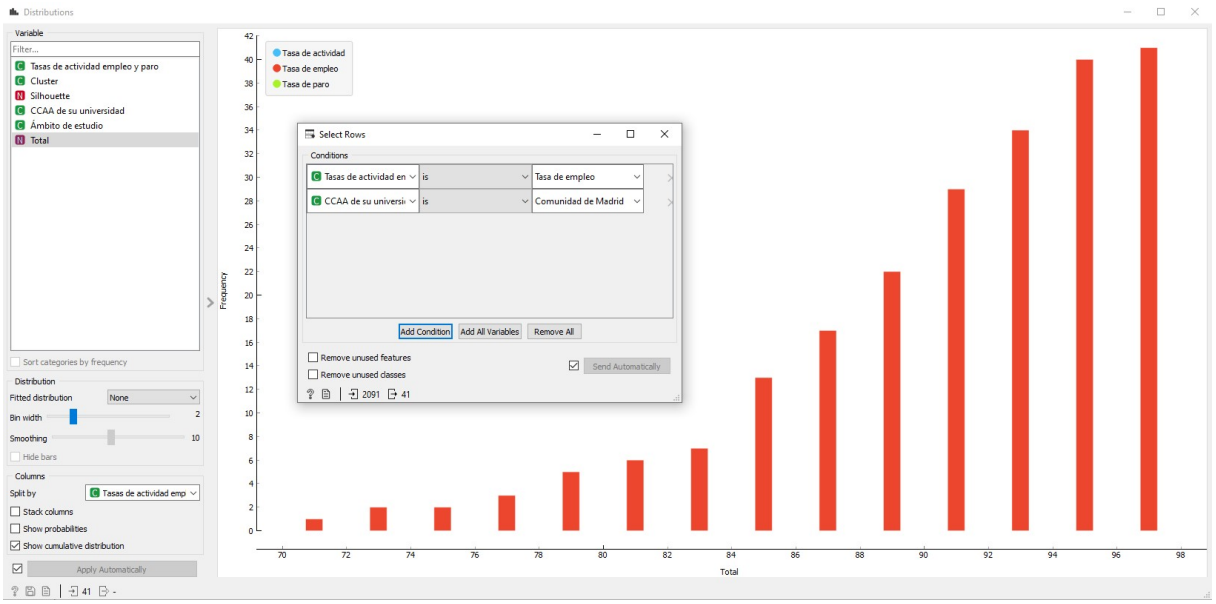


Figura 88. Gráfico de distribución Tasa de empleo C. Madrid en Orange (Elaboración propia).

Se puede comprobar que la tasa de empleo empieza cuando el total es superior a 70. Esto significa, que hay una tasa de empleo positiva. Si se introduce la condición sobre la tasa de

paro (ver Figura 89), observamos como empieza el total superior a 0 y así progresivamente hasta 17 por lo que deducimos que la tasa de paro es baja.

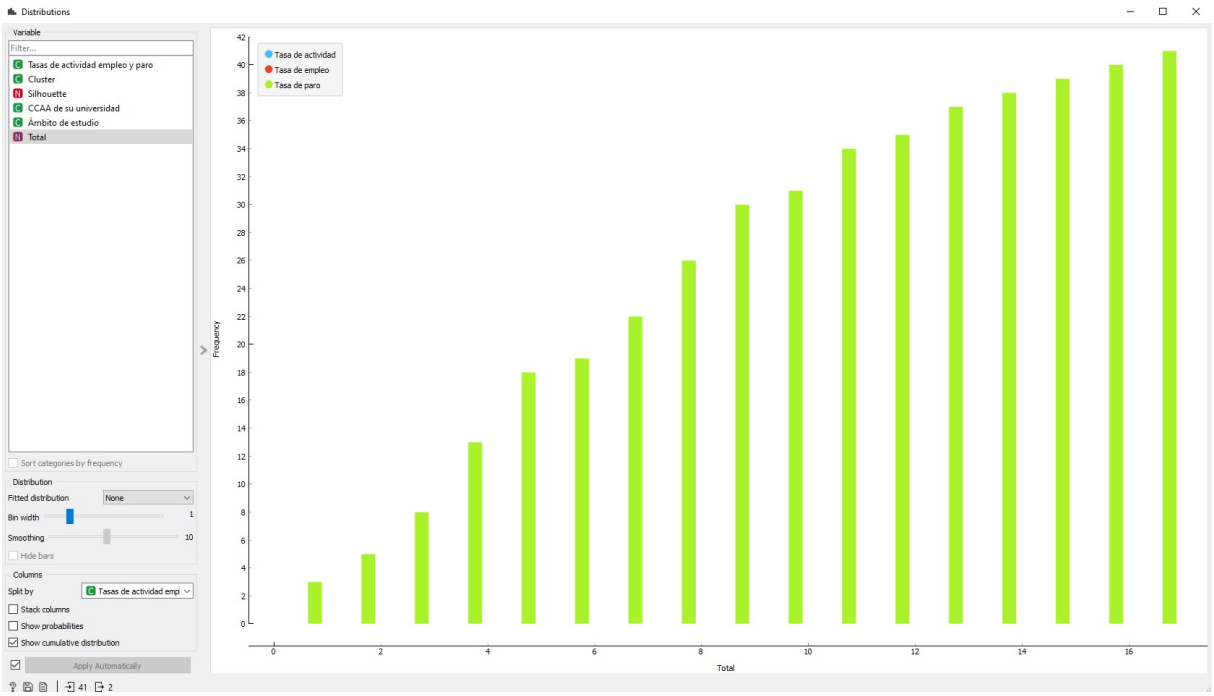


Figura 89. Gráfico de distribución Tasa de paro C. Madrid en Orange (Elaboración propia).

Podemos incluir más *widgets* si se desea profundizar más en el análisis del aprendizaje. Es el caso del *widget Box Plot* (véase Figura 90). Se trata de un, diagrama de caja que muestra la distribución de los valores de los atributos. Es una buena práctica para verificar cualquier dato anómalo, como por ejemplo valores duplicados, valores similares o atípicos. Se seleccionan valores para datos categóricos o el rango de cuartiles para datos numéricos, además de seleccionar las barras (ver Figura 91).

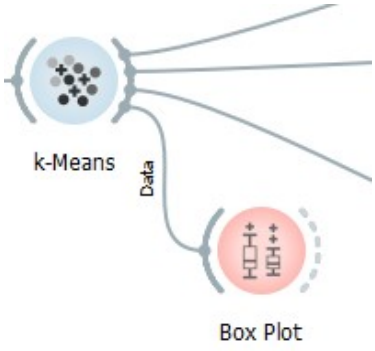


Figura 90. Widget Box Plot en Orange (Elaboración propia).

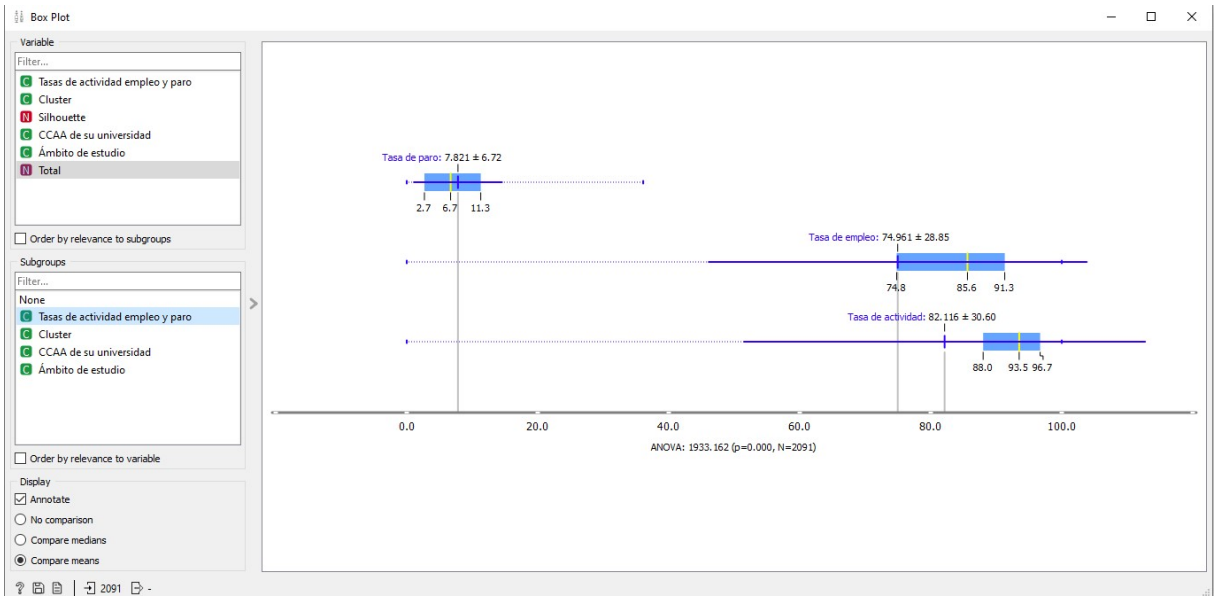


Figura 91. Box Plot K-Means en Orange (Elaboración propia).

En la Figura 91 se observa la media que es la línea vertical azul oscuro y la delgada línea azul representa la desviación estándar. Los valores del primer y tercer cuartil y el área resaltada en azul representando los valores entre el primer y tercer cuartil. Por último, la mediana[33] está representada por la línea vertical amarilla. Se observa, que hay valores de *tasa de empleo y tasa de actividad* duplicados sobre todo entre el tercer cuartil de *tasa de empleo* y el primer cuartil de *tasa de actividad*. Además, se visualizan qué valores se comprenden cada una de las clases.

En el diagrama de barras dentro del *widget Box Plot* se observan aquellas instancias que se encuentran en cada uno de los clústeres (Figura 92).

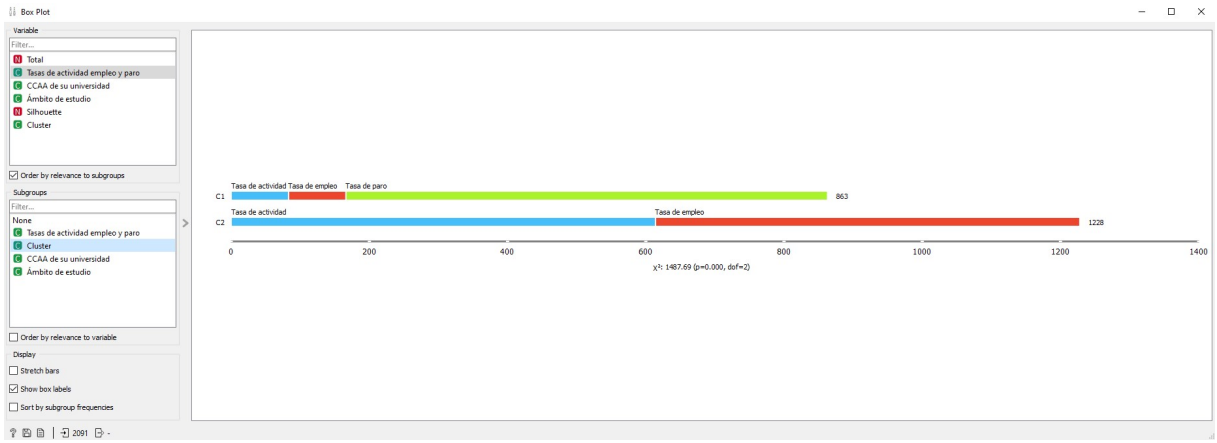


Figura 92. Box Plot diagrama clústeres K-Means en Orange (Elaboración propia).

Ahora definimos $k = 3$, conociendo el número de clases real que tiene el fichero. En el diagrama de dispersión correspondiente se observan diferencias con el análisis anterior (Figura 93). Se puede ver que los agrupamientos están bien definidos, cada uno con su símbolo correspondiente y que el resultado de la línea de regresión se comprende entre 1 y -1, por lo que deducimos que son datos fiables. Se aprecian tres zonas coloreadas que corresponden a las clases definidas. Además, se interpreta que $k = 3$ parece el correcto.

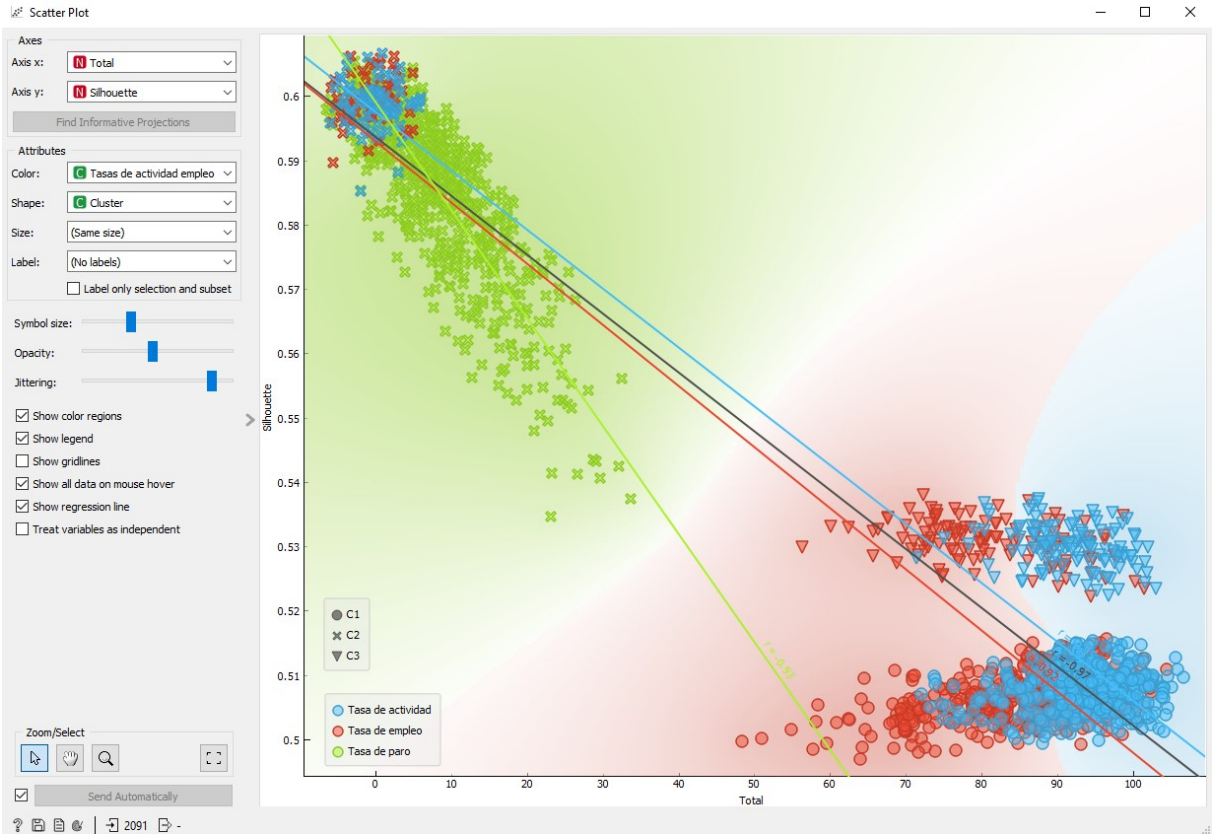


Figura 93. Diagrama de Dispersión k=3 K-Means en Orange (Elaboración propia).

Veamos en el *widget* de *Distribution* cómo están agrupados los datos en cada clúster (Figura 94). En el clúster C1, se incluyen datos de *Tasa de actividad y Tasa de empleo*. En el clúster el C2, se incluyen datos de *Tasa de actividad, Tasa de empleo y Tasa de paro*. Por último, en el clúster C3, se incluyen datos de *Tasa de actividad y Tasa de empleo*. Comparando con el anterior análisis se observa que la *Tasa de paro* solo está definida en un clúster, lo mismo que en análisis anterior. En cambio, la *Tasa de actividad y Tasa de empleo* están distribuidas por todos los clústeres.

[33] Mediana o segundo cuartil: divide en dos partes la distribución.

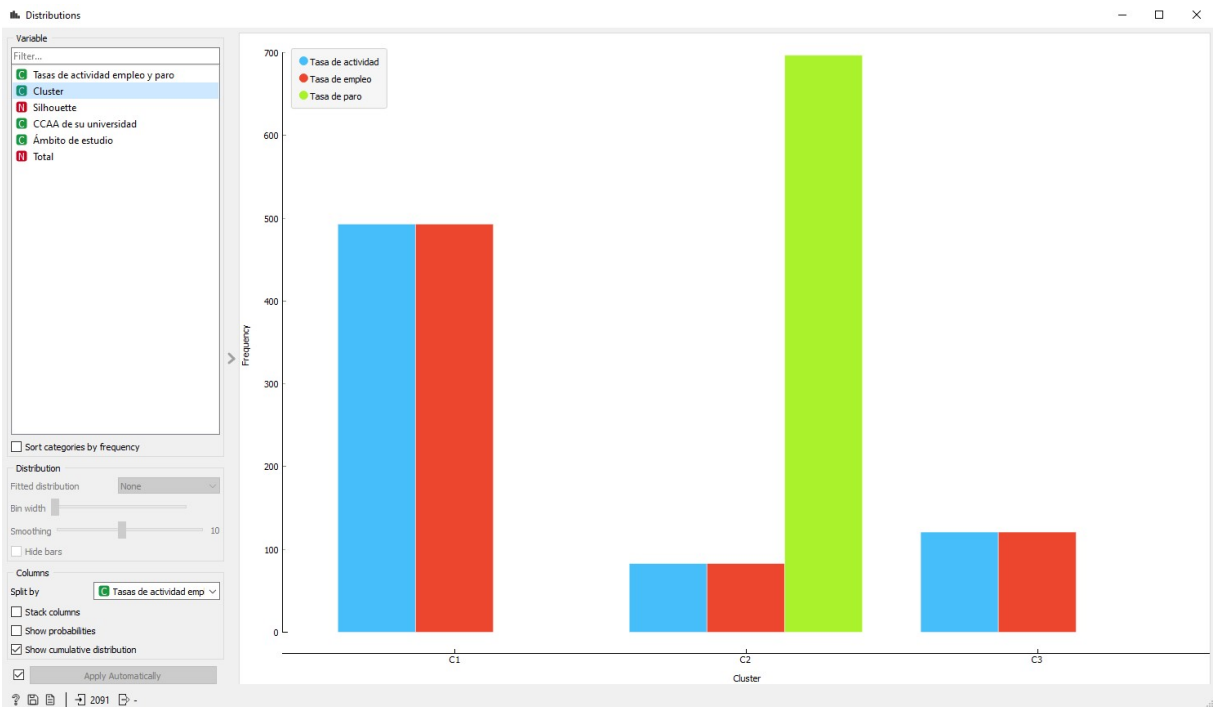


Figura 94. Distribution Clúster k=3 K-Means en Orange (Elaboración propia).

Si se hiciera el mismo análisis anterior observando la *Tasa de paro y Tasa de empleo* de la Comunidad de Madrid, se concluiría que el resultado es el mismo. Sin embargo, en el *widget Box Plot* hay cambios sobre las instancias agrupadas en cada uno de los clústeres, tal y como se muestra en la Figura 95.

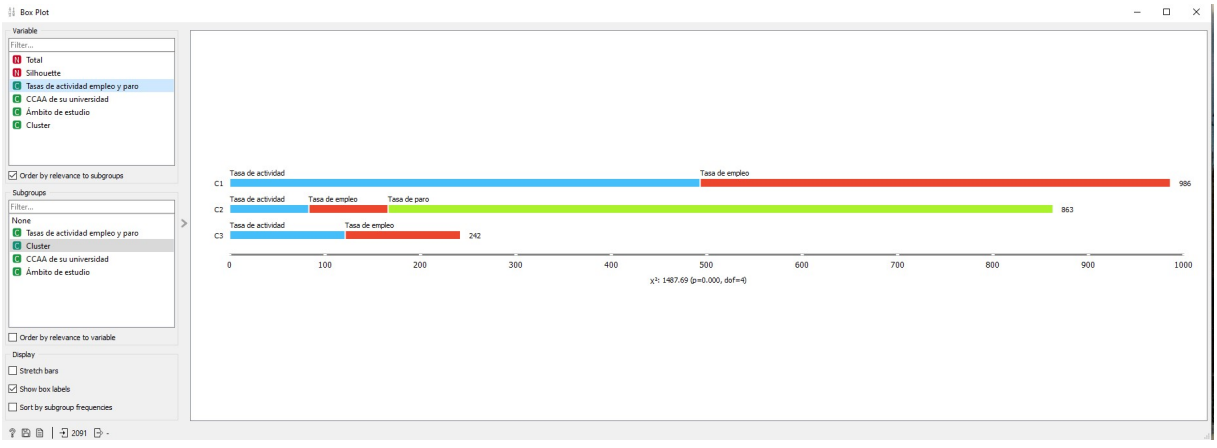


Figura 95. Box Plot diagrama clústeres k=3 K-Means en Orange (Elaboración propia).

Concluyendo en los análisis se observa e interpreta el agrupamiento de *Tasa de paro, Tasa de actividad y Tasa de empleo* en diferentes clústeres. También, se observa cuál es la incidencia de las diferentes tasas en la Comunidad de Madrid por lo que deducimos que hay una tasa de

paro baja y una tasa de empleo alta. Una vez sacadas estas conclusiones, se observa la explicabilidad de varios resultados obteniendo unas conclusiones. Se observa, y según los autores (Al-Odan & Saud, 2015), (Jović, Brkić, & Bogunović, 2014), (Padmavaty, Geetha, & Priya, 2020), (Ratra & Gulia, 2020), al navegar por Orange presenta facilidades, y una interfaz bien estructurada con diferentes características por lo que sus resultados son interpretables, es intuitiva, simple de entender y comprender. El rendimiento del algoritmo ha sido bueno pero cabe destacar, y según el autor (Al-Odan & Saud, 2015), en Orange no se mide el tiempo de ejecución porque no tiene la funcionalidad incorporada. Por lo que no podemos medir el rendimiento en Orange

5.4 Prueba con algoritmo Agrupamiento Jerárquico Aglomerativo

5.4.1 Weka

Utilizaremos para la prueba del algoritmo Agrupamiento Jerárquico en Weka utilizaremos el fichero **tasa de actividad, empleo y paro** que se encuentra público como se ha comentado con anterioridad en la página oficial del Gobierno de España. Una vez cargado el conjunto de datos elegimos en algoritmo *HierarchicalClusterer* que aparece en la pestaña *Clúster* y hacemos click en *Start*. Después comprobamos los resultados obtenidos una vez aplicado el algoritmo *HierarchicalClusterer* este nos proporciona datos importantes. Fijándonos en los resultados obtenidos se aprecia el número de clústeres identificados.

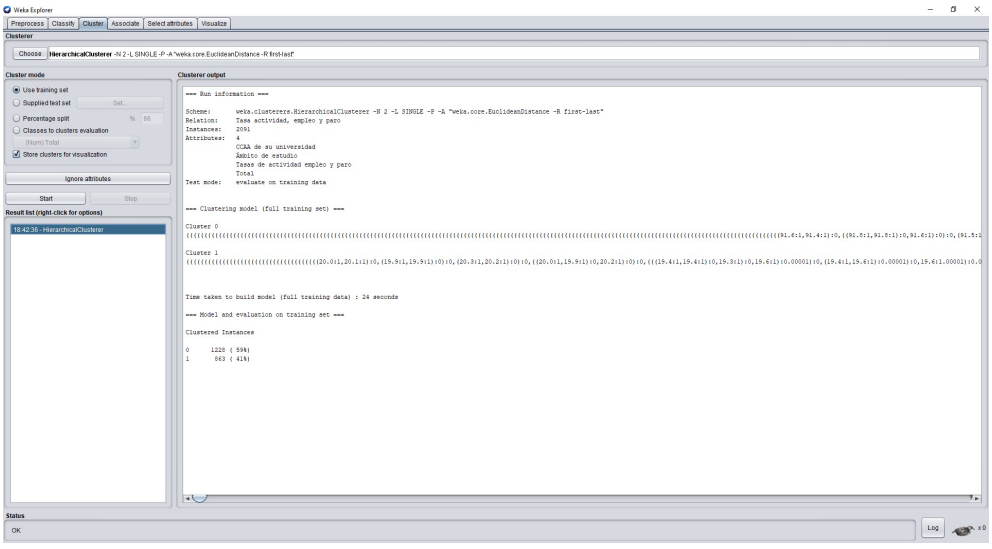


Figura 96. Clústeres Agrupamiento Jerárquico en Weka (Elaboración propia).

Observamos además de los clústeres creados, las instancias que hay dentro de cada clúster. Se crearon dos clústeres porque por defecto el algoritmo contiene ese número de clústeres. En principio no sabemos que número de clases hay en un conjunto de datos por lo que lo dejamos por defecto. En el clúster 0 se encuentran 1228 instancias, esto sería un 59% de los datos totales. En el clúster 1 se encuentran 863 instancias, lo que sería un 41% de los datos totales. Una vez obtenidos estos resultados, visualizamos como estan agrupados estos datos en cada clúster. En la Figura 97, se aprecia claramente como estan definidos los clústeres junto con su grupo de datos.

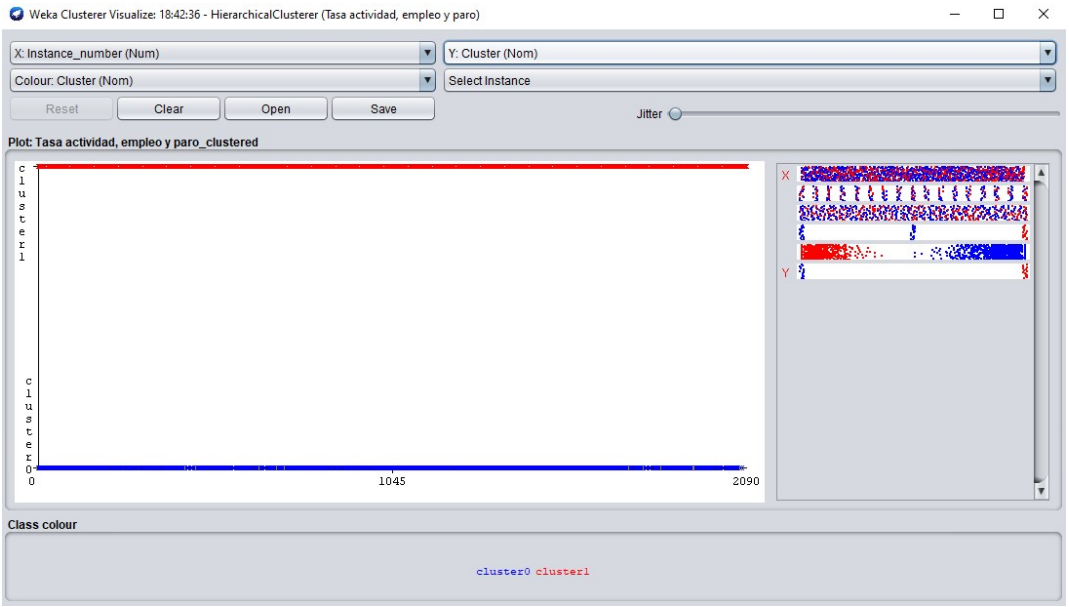


Figura 97. Visualización clústeres Agrupamiento Jerárquico en Weka (Elaboración propia).

Debido a la alta cantidad de datos desde Weka no se puede visualizar el dendrograma con la agrupación de datos mas cercanos. Es por eso por lo que tendríamos que utilizar un conjunto de datos con una cantidad de datos más pequeña. Ahora probemos a cambiar el número de clúster igual a tres (Figura 98).

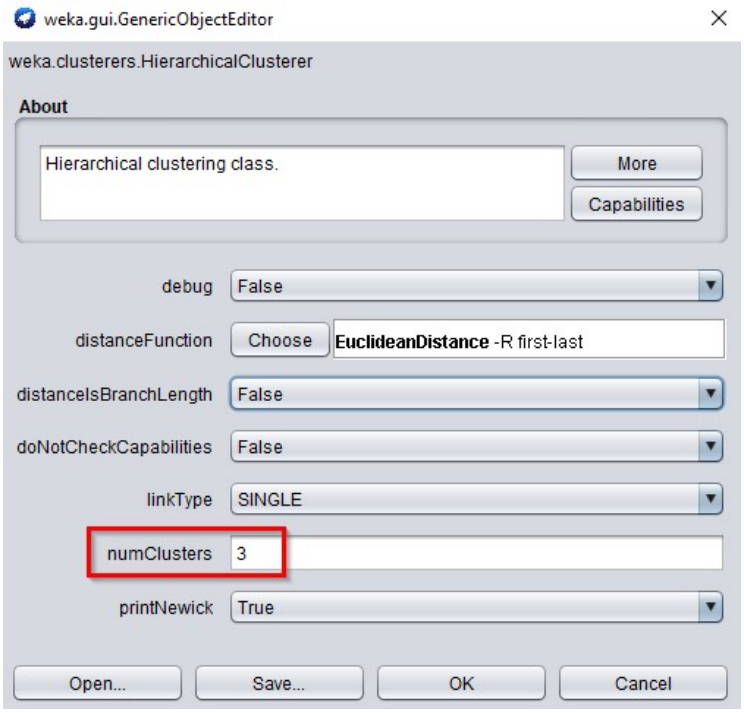


Figura 98. Número clústeres=3 Agrupamiento Jerárquico en Weka (Elaboración propia).

Observamos los resultados (Figura 99) una vez cambiado el número de clúster. Apreciamos como el clúster 0 contiene 1226 instancias lo que supone un 59% de los datos totales. El clúster 1, contiene 863 instancias lo que supone un 41% de los datos totales. Por último, el clúster 2, contiene 2 instancias que suponen un 0%.

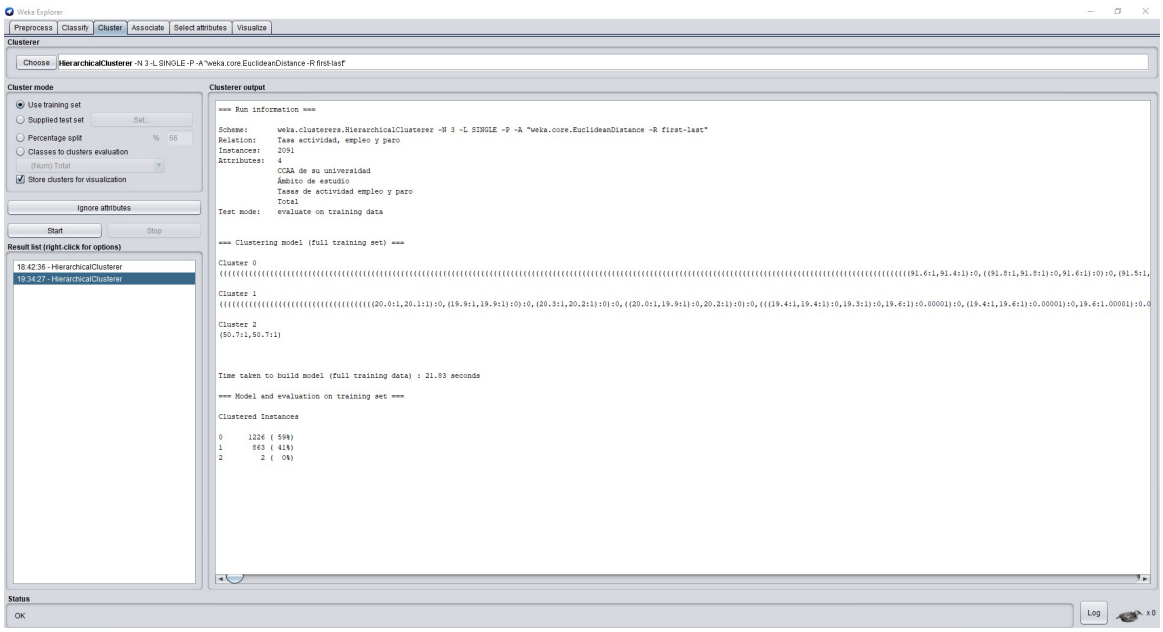


Figura 99. Resultados Agrupamiento Jerárquico num clústeres=3 en Weka (Elaboración propia).

Después, visualizamos como estan agrupados estos datos en cada clúster. En la Figura 100, se aprecia claramente como estan definidos los clústeres junto con su grupo de datos.

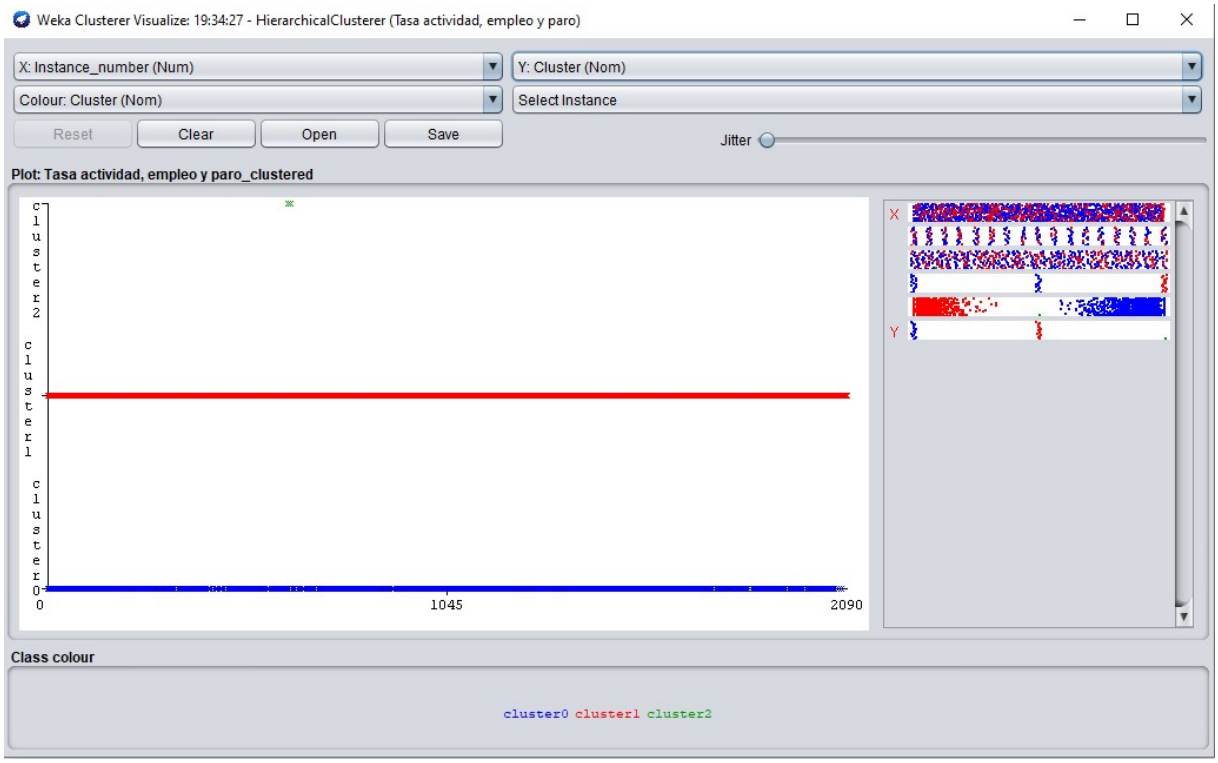


Figura 100. Visualización num clústeres = 3 Agrupamiento Jerárquico en Weka (Elaboración propia).

Para poder visualizar el dendograma nos encontramos con un obstáculo. Se necesita un conjunto de datos más pequeño para poder visualizar el dendrograma. Se comprueba de esta forma, lo que comentan los autores en sus investigaciones (Jović, Brkić, & Bogunović, 2014), (Ratra & Gulia, 2020). Weka a veces es poco amigable y carece de buena visualización de datos. Interpretando y comprendiendo los resultados del análisis se pueden ver el agrupamiento de tasa de paro, actividad y empleo en diferentes clústeres, igual que en K-Means. También se observa las instancias que se encuentran en cada clúster junto con su porcentaje. Faltaría conocer como se ha aplicado el dendrograma ya que la cantidad de datos de este conjunto de datos es alto. Si utilizaramos un conjunto de datos más pequeño podríamos observar como se ha creado el dendrograma.

5.4.2 Orange

En Orange existe como tal este widget por lo que tendremos que crear el flujo con los *widgets* que tenemos en Orange para crearlo. Este workflow se encuentra en los ejemplos ya definidos en Orange por lo que tomaremos ese mismo. Utilizaremos el fichero **tasa de actividad, paro y empleo.csv** para hacer esta prueba. En la Figura 101, se muestra el flujo que utilizaremos para el algoritmo de *Agrupamiento Jerárquico*.

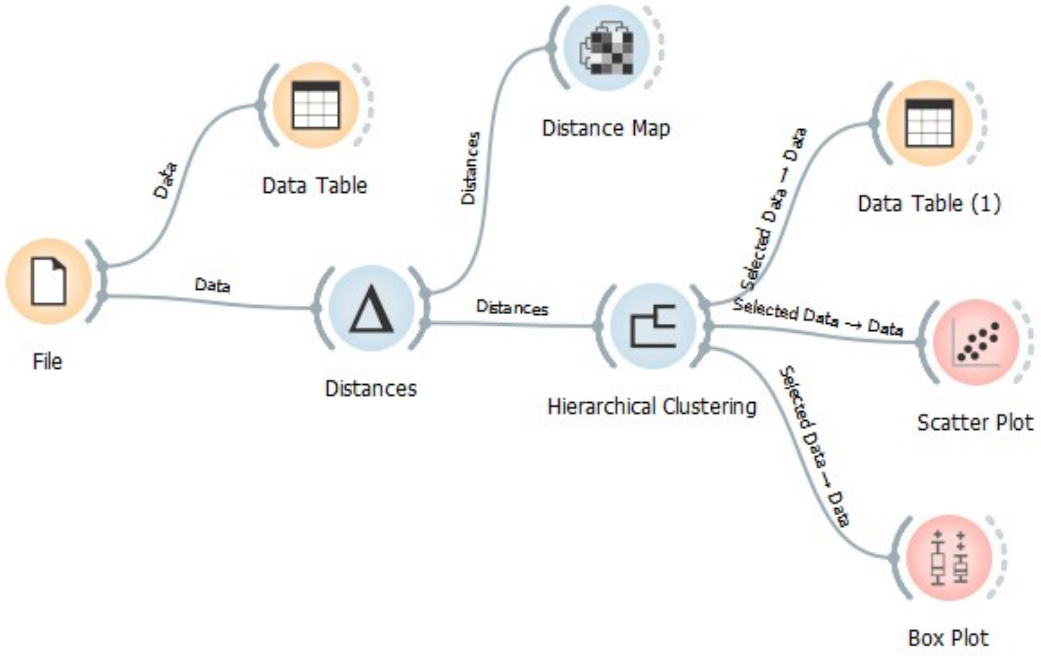


Figura 101. Flujo Algoritmo Agrupamiento Jerárquico en Orange (Elaboración propia).

Empezamos explicando cada uno de los *widgets* que observamos en la Figura 98. Una vez introducido el fichero mediante el widget *File*, se tiene en la parte superior el *widget Data Table*. Este *widget* nos muestra los datos de entrenamiento introducidos. Porteriormente, nos encontramos con el *widget Distances* que calcula las distancias entre filas/columnas en un conjunto de datos. Con este widget los datos se normalizarán para garantizar un tratamiento equitativo de las características individuales. En la Figura 102, se observa que es lo que contiene este *widget.* Vemos como se ha elegido que las distancias sean entre filas, la métrica de distancia es la euclidiana ("línea recta", distancia entre dos puntos).

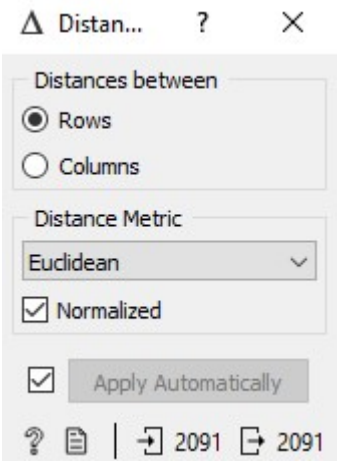


Figura 102. Widget Distances Agrupamiento Jerárquico en Orange (Elaboración propia).

El siguiente *widget* es *Distance Map*. Con este *widget* visualizamos las distancias entre elementos o atributos. En la Figura 103, se observan las distancias entre filas en los datos de *Tasa de actividad, Tasa de empleo y Tasa de paro*, donde las distancias más pequeñas se representan en amarillo y las más grandes en naranja oscuro.

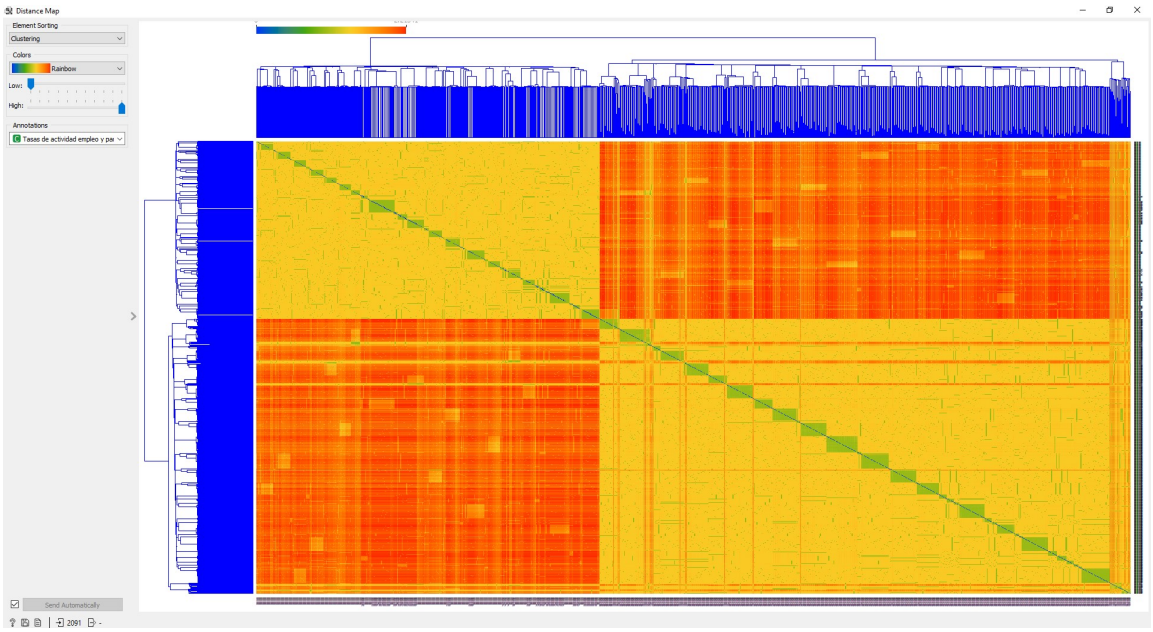


Figura 103. Distance Map Agrupamiento Jerárquico en Orange (Elaboración propia).

La clasificación está organizada mediante *clustering* que agrupa los datos por similitud. Las anotaciones están por *Tasa de actividad, Tasa de empleo y Tasa de paro*. La línea diagonal nos indica los atributos más cercanos. Podemos seleccionar una región en el mapa y arrastrar del cursor. Este procedimiento selecciona una parte del mapa y el widget genera todos los elementos de las celdas seleccionadas. El siguiente *widget* es *Hierarchical Clustering*. Este

agrupa elementos mediante un algoritmo de agrupación jerárquica y muestra un dendrograma con las agrupaciones correspondientes.

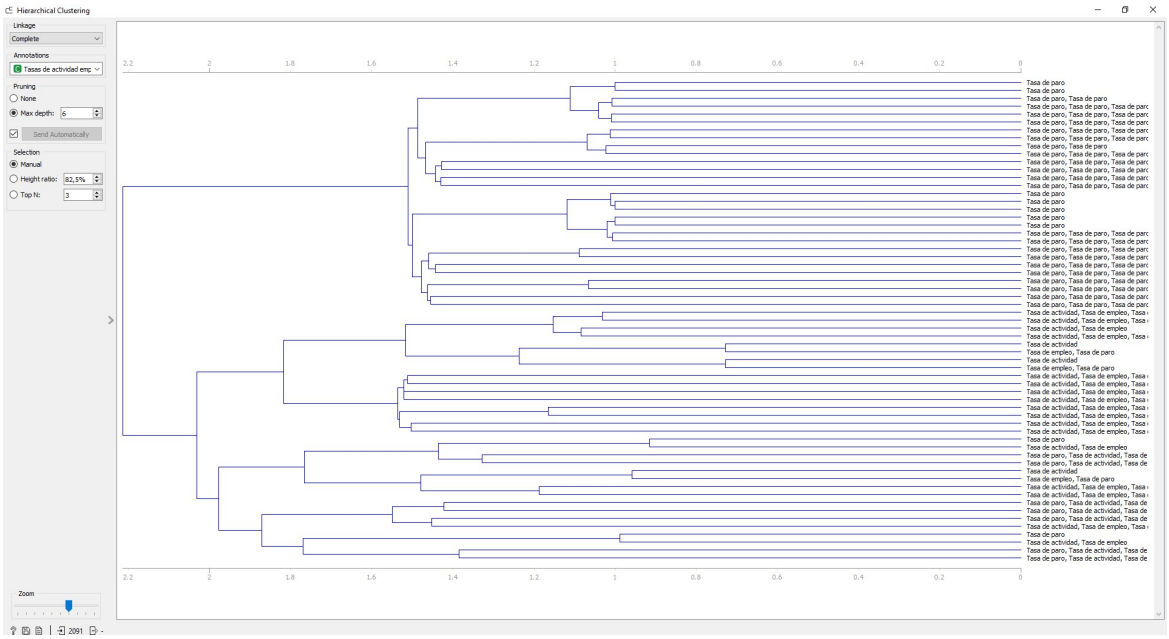


Figura 104. Hierarchical Clustering Agrupamiento Jerárquico en Orange (Elaboración propia).

En la Figura 104 aparece el dendrograma de la *Tasa de actividad, Tasa de empleo y Tasa de paro*. Con un máximo de profundidad de seis para que se pueda visualizar correctamente. Esto no afecta al agrupamiento si no a la visualización del dendrograma. También, la selección es *Manual* (si hacemos click en el dendrograma se pueden seleccionar varios grupos manteniendo presionado Ctrl). Se podría seleccionar *Height ratio* (aparece una línea de corte en el gráfico seleccionando los elementos a la derecha de la línea) tal y como aparece en la Figura 105. Según avanzamos la línea de corte hacia la derecha se observa cómo crece el número de clústeres. También, podemos elegir *Top N* que selecciona el número de nodos superiores (ver Figura 106). En el caso de *Top N* se eligieron cuatro nodos.

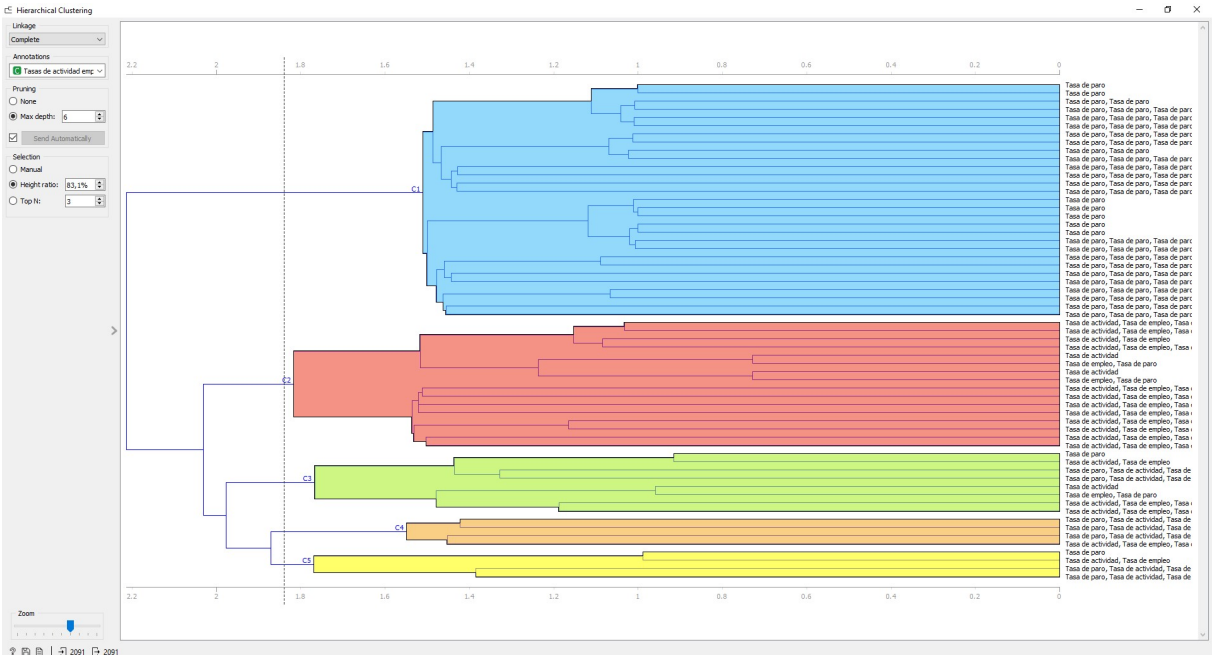


Figura 105. Height Ratio Agrupamiento Jerárquico en Orange (Elaboración propia).

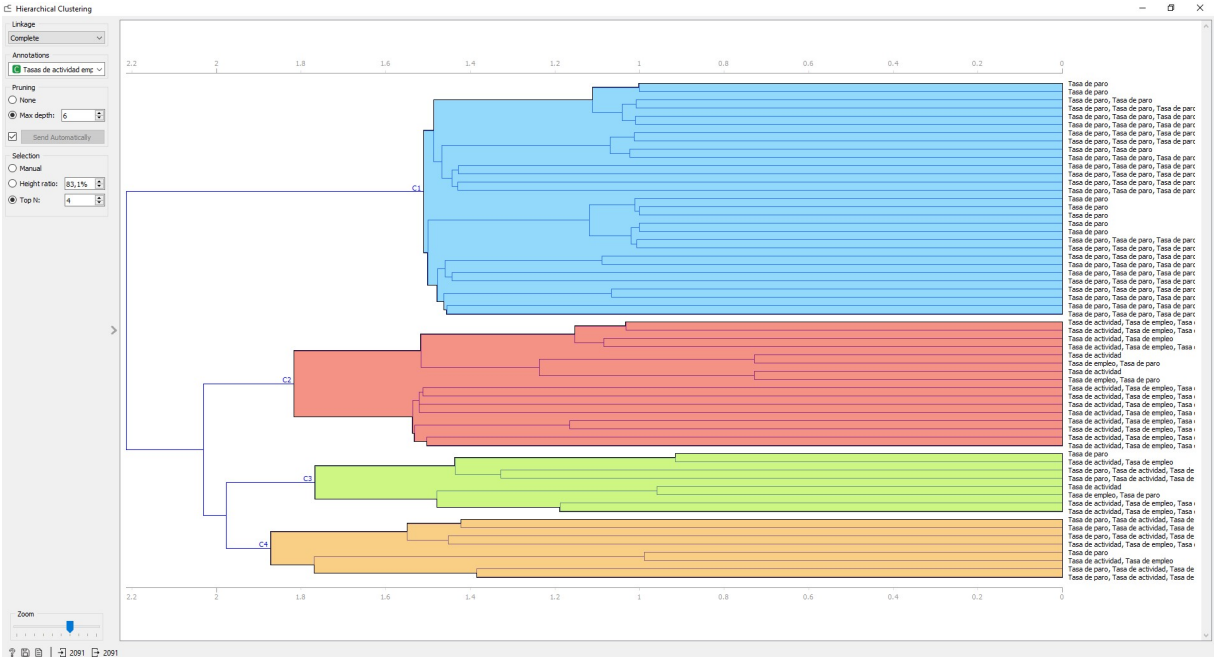


Figura 106. Top N = 4 Agrupamiento Jerárquico en Orange (Elaboración propia).

Luego tendríamos *Data Table (1)* que nos indica que instancias pertenecen a cada clúster. El diagrama de dispersión muestra los clústeres creados más las clases identificadas. Se aprecia como la línea de regresión es igual a uno (ver Figura 107).

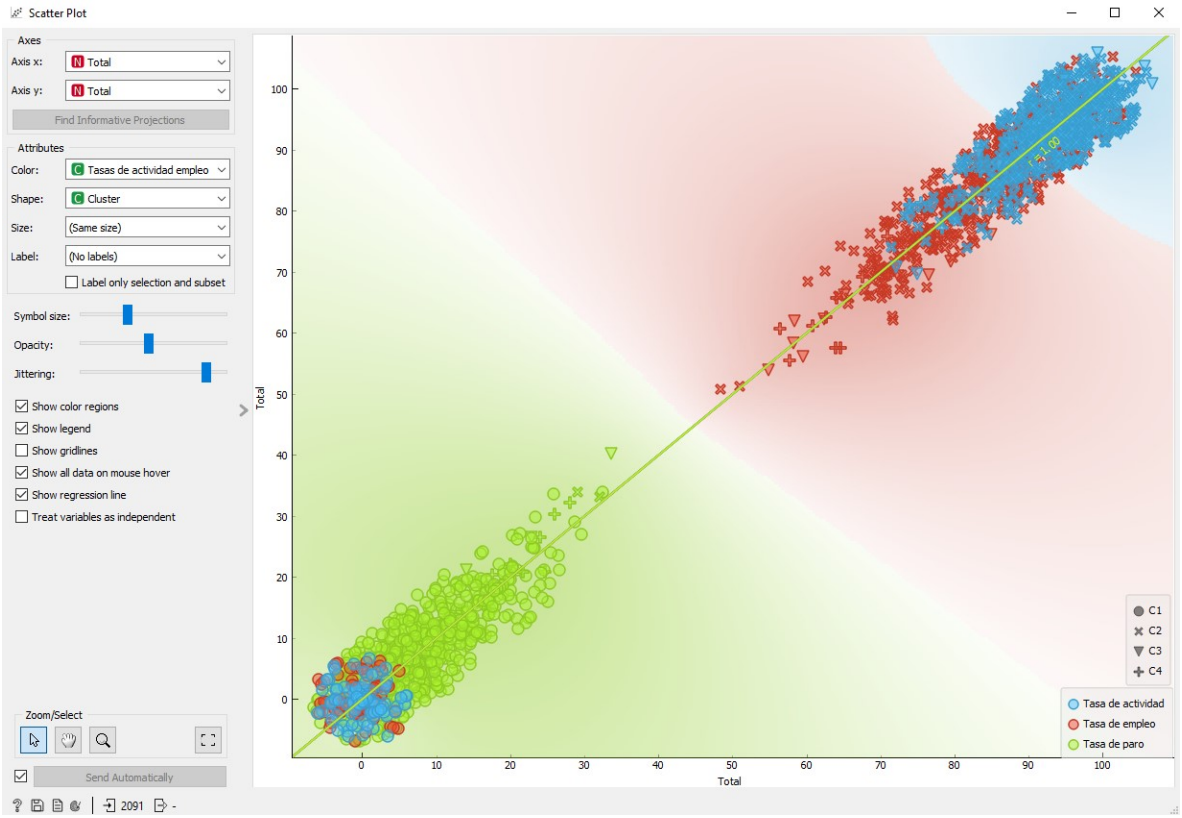


Figura 107. Diagrama de dispersión Agrupamiento Jerárquico en Orange (Elaboración propia).

Por último, visualizamos el *widget Box Plot*. En él observamos cuantas instancias se encuentran en cada uno de los clústeres. En el clúster C1, se encuentran 850 instancias. En el clúster C2, 1134 instancias. En el clúster C3, 74 instancias. Y, en el clúster C4, 33 instancias (ver Figura 108).

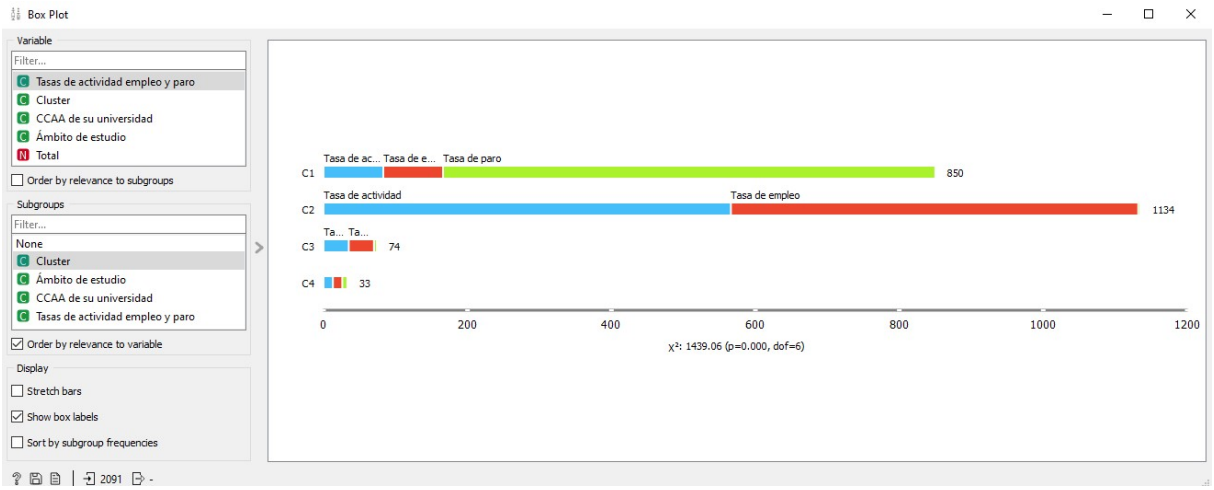


Figura 108. Box Plot Clústeres Agrupamiento Jerárquico en Orange (Elaboración propia).

Además, al observar la media y desviación estándar comprobamos que son los mismos valores que cuando aplicamos el algoritmo *K-Means*. Los valores del primer y tercer cuartil y el área resaltada en azul representando los valores entre el primer y tercer cuartil. La mediana también es la misma y los valores de tasa de empleo y tasa de actividad hay duplicados sobre todo entre el tercer cuartil de tasa de empleo y el primer cuartil de tasa de actividad (ver Figura 109).

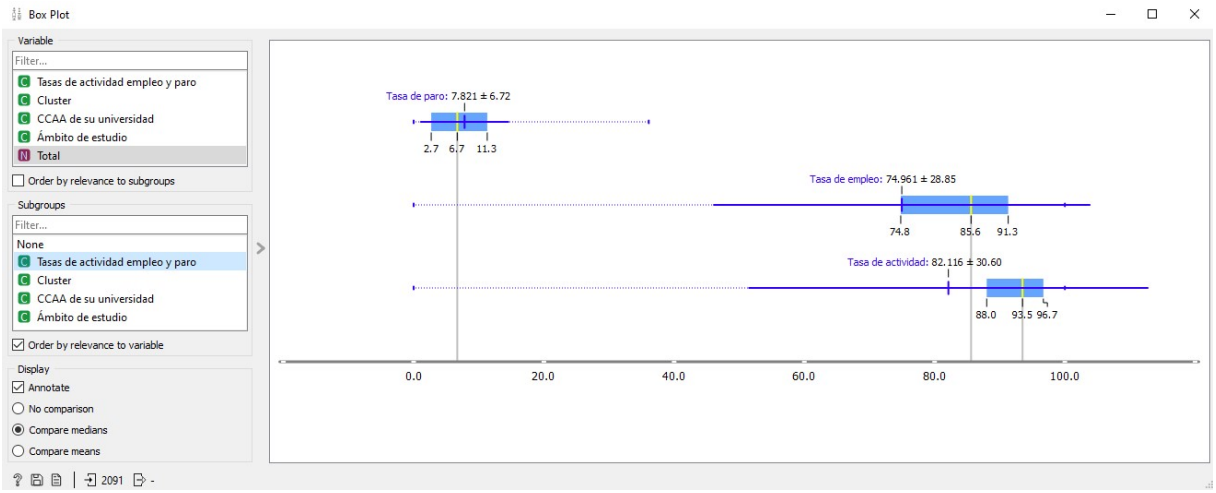


Figura 109. Box Plot Agrupamiento Jerárquico en Orange (Elaboración propia).

Concluyendo el análisis se observa e interpreta, visualizando los datos, el agrupamiento de tasa de paro, actividad y empleo en diferentes clústeres y también, mediante el dendrograma aumentar o disminuir el número de clústeres. También, se observan cuáles son las instancias en cada uno de los clústeres. Mediante el diagrama de dispersión se aprecia que hay poco paro y mucho empleo y actividad en general. Además observa, y según los autores (Al-Odan & Saud, 2015), (Jović, Brkić, & Bogunović, 2014), (Padmavaty, Geetha, & Priya, 2020), (Ratra & Gulia, 2020), Orange presenta facilidades al navegar por todas sus funcionalidades. Su interfaz bien estructurada hace que sus resultados sean interpretables y explicables. Además su interfaz gráfica es intuitiva y simple de entender. La visualización de los datos en Orange supera a Weka debido a lo atractiva que es su interfaz.

Capítulo 5

6. Conclusiones

En la actualidad el aprendizaje automático es una tecnología que ofrece beneficios, como por ejemplo, ahorrando grandes cantidades de dinero para las empresas. Abre nuevas oportunidades a los negocios, para los investigadores ampliando conocimientos y búsqueda de soluciones a problemas reales en áreas como medicina, industria, medio ambiente, etc.

Después de haber realizado un estudio, evaluación y análisis de las herramientas Orange y Weka en este trabajo y reconocidas sus amplias capacidades de aprendizaje automático, se han obtenido las siguientes conclusiones:

Ambas herramientas son accesibles en cualquier Sistema Operativo y de fácil instalación. En términos de intuición, Orange es considerada más intuitiva que Weka. Orange posee una interfaz gráfica visualmente atractiva. Además, los gráficos hacen que sea destacable en intuición. No se puede decir lo mismo de Weka. En técnicas de aprendizaje, ambas herramientas contienen aquellas técnicas más utilizadas actualmente. Las dos herramientas contienen una interfaz gráfica, que permite el procesamiento, análisis y exploración de datos. Ambas herramientas son utilizadas por empresas y también por usuarios. La visualización de la información en distintos formatos como gráficos de barras, árboles de decisión, diagramas de dispersión o diagramas de caja permiten mostrar aquellos resultados en ambas herramientas. En Weka se comprobó que la visualización de los datos es poco amigable.

Hay muchos tutoriales disponibles en internet sobre ambas herramientas, lo que hace crecer el entendimiento y comprensión de los usuarios. En término de entendimiento sencillo o simple de entender Orange es considerada más sencilla o simple de entender ya que posee una interfaz gráfica visualmente atractiva que hace todo más fácil a la hora de entender los resultados obtenidos. Es por eso por lo que Orange es más sencilla de entender que Weka. No es necesario disponer de conocimientos de programación en ambas herramientas para utilizarlas, esto facilita el trabajo y lo convierte en herramientas muy útiles para los usuarios. En términos de accesibilidad Weka y Orange al contener interfaz gráfica hace más fácil su

utilización, y ofrece facilidades de acceso a los componentes principales. Lo que facilita el trabajo de análisis. Todo ello, dando la posibilidad de que pueda ser usado por el mayor número de personas posibles. En términos de rendimiento cabe destacar como ya dijimos con anterioridad que Orange no mide el tiempo de ejecución porque no tiene la funcionalidad incorporada de medición del tiempo de ejecución. En Weka, esta funcionalidad está incluida. Lo que indica una clara ventaja sobre rendimiento en Weka. Sobre el término usabilidad contando con un estudio previo que estudia esta característica y comprobando en este análisis podemos decir que Orange supera a Weka en términos de usabilidad.

Sobre los términos amigabilidad, cabe destacar que Weka a veces es poco amigable ya que carece de buena visualización de datos. En cambio, Orange no tiene este problema. Además en Orange se pueden extraer los informes. Es por eso por lo que Orange es más afable que Weka. En términos de explicabilidad e interpretabilidad ambas herramientas cumplen este criterio ya que se consideran herramientas cómodas, muestra facilidades al usuario, la interacción del usuario con la herramienta es exitosa, etc.

Con esto se puede concluir que ambas herramientas son de gran utilidad ya que ambas permiten aplicar técnicas que facilitan la comprensión de los datos explorados. No obstante, no existe ninguna herramienta mejor que otra en este caso si no que cada una aporta distintas formas de aprendizaje y entrenamiento.

Finalmente, con este proyecto se pretende también incentivar a los estudiantes a conocer el uso de las herramientas de aprendizaje automático para poder emplearlas en su aprendizaje y saber elegir cual es la que se adecua a su necesidad particular.

Con este trabajo he tenido la oportunidad de aprender las diversas herramientas de aprendizaje automático y su utilidad actualmente. Lejos de ser un obstáculo debido a no tener los conocimientos necesarios de Inteligencia Artificial y Aprendizaje Automático me ha sido de gran valor para emprender este proyecto. Además me ha ayudado a ampliar mis conocimientos en este campo.

Se ha llegado a conseguir el objetivo planteado desde el principio sobre el análisis de herramientas para el estudio de técnicas de aprendizaje automático y esto se desglosa en los siguientes subobjetivos:

- Estudio de las herramientas a analizar y evaluar.
- Análisis Crítico.
- Definición de las herramientas y profundización de su estructura.
- Evaluación de las herramientas sobre técnicas de aprendizaje automático.
- Utilización de estas herramientas en la actualidad.

7. Futuras líneas de trabajo y mejoras

Seguidamente, se muestran aquellas posibles líneas de mejora que no han podido realizarse en este proyecto:

Aplicar otras técnicas de aprendizaje automático con Weka y Orange

En este trabajo se han aplicado algunas de las técnicas más utilizadas actualmente con las herramientas de aprendizaje automático Weka y Orange. Sin embargo, se pueden aplicar otras técnicas de aprendizaje como *árboles de decisión ID3*, algoritmo *A Priori*, algoritmo *Cobweb*, etc. Para poder extraer el análisis de las herramientas con otros aspectos específicos de dichos algoritmos de aprendizaje. Del mismo modo, se podría ampliar el estudio a otras herramientas bien conocidas y utilizadas como *RapidMiner*, *Knime* etc. no comentadas en este trabajo. Todo ello ampliando los conocimientos de las herramientas. De esta forma podremos aprovechar todas esas nuevas formas de aprendizaje.

Aplicar otros criterios sobre las herramientas de aprendizaje automático

Se aplicaron los criterios más utilizados hoy en día, pero se podrían añadir otros como, por ejemplo, criterios de medición, seguridad de los datos, etc.

PARTE III

Glosario, Anexo y Bibliografía

Glosario

- Algoritmo: Conjunto ordenado y finito de operaciones sistemáticas que permite hacer un cálculo y hallar la solución de un tipo de problemas.

- Aprendizaje Automático: Rama de la Inteligencia Artificial, en concreto es el campo de estudio que proporciona a las computadoras las capacidad de aprender sin haber sido explícitamente programadas para ello.

- Aprendizaje No Supervisado: tipo de Aprendizaje Automático en el cual el algoritmo no recibe información sobre cómo deben ser los datos de salida.

- Aprendizaje Reforzado: tipo de Aprendizaje Automático que se utiliza cuando no hay idea de la clase o etiqueta de un dato en particular.

- Aprendizaje Supervisado: tipo de Aprendizaje Automático en el cual el algoritmo recibe previamente información sobre relaciones existentes entre los datos de entrada y salida y sobre cómo deben ser estos últimos.

- Dataset: conjunto de datos con una serie de variables que corresponden a cada miembro del conjuntos de datos.

- Entrenamiento: proceso por el cual se forma un algoritmo con un conjunto de datos.

- Gráfico: aquello que se representa por medio de signos o dibujos.

- Herramienta: conjunto de programas, aplicaciones o instrucciones usadas para ejecutar tareas concretas.

- Inteligencia Artificial: disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana, como el aprendizaje o el razonamiento lógico.

- Matriz Confusión: matriz de valoración de un modelo de clasificación basado en el aprendizaje automático. Este valora como de bueno o malo es el modelo.

- Minería de Datos: consiste en la extracción de datos procesable de los conjuntos grandes de datos, todo ello para deducir patrones y tendencias que existen en los datos.

- Modelo: algoritmo de aprendizaje automático que construye su propia comprensión de un tema.

- Widget: pequeña aplicación o programa que facilita el acceso a las funciones.

Anexo

Fichero Iris.arff

Fichero Breast-Cancer.tab

Fichero tasa de actividad, empleo y paro

Bibliografía

Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. ACM SIGMOD Conference Washington DC, USA, May 1993.

Al-Odan, H. A., & Saud, A. A.-D. (2015). *Open Source Data Mining Tools, A Comparative Study.* Arabia Saudí.

Ameen*, A. O., Bajeh, A. O., Adesiji, B. A., Balogun, A. O., & Mabayoje, M. A. (2018). *Performance evaluation of select data mining software tools for data clustering.* Kwara State, Nigeria.

Arriagada Rodríguez, M. (2015). *Comparación de métricas de distancia en el algoritmo K-Vecinos más cercanos para el problema de reconocimiento automático de dígitos manuscritos.*

B Do, C., & Batzoglou, S. (2008). What is the expectation maximization. *Nature Publishing Group, http://www.nature.com/naturebiotechnology*.

BDM, R. (15 de 12 de 2020). *Big Data Magazine*. Obtenido de Big Data Magazine: https://bigdatamagazine.es/7-aplicaciones-practicas-del-machine-learning-en-la-vida-cotidiana-de-las-que-pocos-son-conscientes

Breiman, L. (2001). *Random Forest.* Statistics Department, University of California Berkeley, CA 94720.

Cámara Madrid. (30 de 12 de 2019). Obtenido de Cámara Madrid: https://www.mba-madrid.com/empresas/impacto-del-machine-learning-ambito-empresarial/

Carletta, J. (1996). Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics, 22(2)*, 249–254.

Cendrowska, J. (1987). PRISM: An algorithm for inducing modular rules. C/O The Faculty of Mathematics, The Open University, Walton Hall, Milton Keynes,.

Corso, C. L. (2009). *Aplicación de algoritmos de clasificación supervisada usando Weka.* Universidad Tecnológica Nacional, Facultad Regional Córdoba Argentina. Obtenido de https://www.academia.edu/9538896/Aplicaci%C3%B3n_de_algoritmos_de_clasificaci%C3%B3n_supervisada_usando_Weka

Cutler, A., & Breiman, L. (s.f.). *Random Forest, Leo Breiman and Adele Cutler*. Obtenido de Random Forest, Leo Breiman and Adele Cutler: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#intro

datos.gob.es. (21 de 04 de 2021). *datos.gob.es reutiliza la información pública.* Obtenido de https://datos.gob.es/es/blog/las-herramientas-de-analisis-de-datos-mas-populares

Delen, D. (12 de 2020). *Informit, libros, libros electrónicos y aprendizaje digital.* Obtenido de https://www.informit.com/articles/article.aspx?p=3100071&seqNum=5

Delgado, A. (1998). *Inteligencia Artificial y Mini Robots. Segunda Edición.* Eco Ediciones.

Dieter, N. (1988). *Sistemas Expertos. Ingeniería y Comunicación.* Editores Marcombo.

Dušanka, D., Darko, S., Srdjan, S., Marko, A., & Teodora, L. (2017). *A Comparison of Contemporary Data Mining Tools.* Serbia.

Dutt, S., Chandramouli, S., & Das, A. K. (2020). *Machine Learning.* Pearson.

España, G. d. (Noviembre de 2020). *La Moncloa.* Obtenido de https://www.lamoncloa.gob.es/presidente/actividades/Documents/2020/ENIAResumen2B.pdf

Espinosa-Zúñiga, J. J. (2020). Aplicación de algoritmos Random Forest y XGBoost en una base de solicitudes de tarjetas de crédito. *Ingeniería Investigación y Tecnología*, 16.

Freund, Y., & Schapire, R. (1996). *Experiments with a new boosting algorithm, Machine Learning: Proceedings of the Thirteenth International Conference, pp. 148-156.*

Fuentes Flores, G. R., & Rosas Arévalo, D. A. (2018). *Algoritmo De Minería De Datos, Búsqueda de patrones periódicos en función del tiempo en bases de datos Espacio-Temporales.* Chile.

García Jímenez, M., & Álvarez Sierra, A. (2010). *Análisis de Datos en WEKA – Pruebas de Selectividad.* Madrid.

García Morate, D. (2006). *Manual WEKA.*

García Pichardo, V. H. (2005). *Algoritmo ID3 en la detección de ataques en aplicaciones web.* Atizapán de Zaragoza, Edo. Méx.

Garre, M., Cuadrado, J. J., & Sicilia, M. A. (s.f.). *Comparación de diferentes algoritmos de clustering en la estimación de coste en el desarrollo de software.* Obtenido de Dept. de Ciencias de la Computación, ETS Ingeniería Informática, Universidad de Alcalá: http://www.sc.ehu.es/jiwdocoj/remis/docs/GarreAdis05.pdf

Gonzalez Pachecho, V. (18 de 01 de 2019). *Telefónica Tech.* Obtenido de https://empresas.blogthinkbig.com/una-breve-historia-del-machine-learning/

González, H. M., & Pérez, L. I. (2006). *Extensiones al Ambiente de Aprendizaje Automatizado WEKA.*

H. Press, W., A. Teukolsky, S., T. Vetterling, W., & P. Flannery, B. (2007). *Numerical Recipes The Art os Scientific Computing Third Edition.* Cambridge University Press.

Hastie, T., Tibshirani, R., & Friedman, J. (2008). *The Elements of Statistical Learning Data Mining, Inference and Prediction Second Edition.* Springer.

Herbrich, R., & Thore, G. (2015). *Aprendizaje automático: una perspectiva algorítmica.* CRC Press.

Hernández Cáceres, J. (2015). *Clustering basado en el algoritmo K-means para la identificación de grupos de pacientes quirúrgicos.* Bucaramanga, Colombia.

https://orange.biolab.si/. (s.f.).

https://www.cs.waikato.ac.nz/ml/weka/. (s.f.).

IBM. (1997). *Deep Blue*. Obtenido de https://www.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/

Joachims, T. (s.f.). *SVMlight* . Obtenido de SVMlight : http://svmlight.joachims.org/

Jović, A., Brkić, K., & Bogunović, N. (2014). *An overview of free software tools for general data mining.* Opatija, Croatia.

KDnuggets. (05 de 2019). *KDnuggets.* Obtenido de https://www.kdnuggets.com/2019/05/poll-top-data-science-machine-learning-platforms.html/2

KPMG. (10 de 3 de 2020). *Home KPMG*. Obtenido de Home KPMG: https://home.kpmg/es/es/home/sala-de-prensa/notas-de-prensa/2020/03/ceos-implantan-ia-automatizacion-procesos.html

Kulkarni, E. G., & Kulkarni, R. B. (2016). *WEKA Powerful Tool in Data Mining.* International Journal of Computer Applications (0975 – 8887).

Lagos Vera, C. V. (2011). *Creación de perfiles de deudores de crédito universitario, para mejoramiento de campañas de cobranza, usando minería de datos.* Valdivia-Chile.

Lázaro Enguita, P. (2018). *Machine Learning en la industria del automóvil.* Alcala de Henares.

Lin, C.-J., & Chang, C.-C. (s.f.). *LIBSVM -- A Library for Support Vector Machines*. Obtenido de LIBSVM -- A Library for Support Vector Machines: https://www.csie.ntu.edu.tw/~cjlin/libsvm/

López Takeyas, B. (2013). Obtenido de http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas_alumnos/ID3/ID3.pdf

M.Mitchell, T. (1997). *Machine Learning.* McGraw-Hill.

Ma, A. (2016). *Using Python and K-means to find the colors in movie posters.*

Microsoft. (2020). *Microsoft Azure*. Obtenido de https://azure.microsoft.com/es-es/overview/what-is-machine-learning-platform/#benefits

Molina López, J., & García Herrero, J. (2004). *Técnicas de análisis de datos, aplicaciones prácticas utilizando microsoft Excel y Weka.*

Morales, E., & Escalante, H. J. (s.f.). *Aprendizaje de Reglas.* INAOE.

Moreno, A., Armengol, E., Béjar, J., Belanche, L., Cortés, U., Gavaldà, R., . . . Sánchez, M. (1994). *Aprendizaje automático.* Barcelona: Edicions UPC.

Mota Aragón, B., & Núñez, J. A. (2020). Estimación de la distribución multivariada de los rendimientos de los tipos de cambio contra el dólar de las criptomonedas Bitcoin, Ripple y Ether. *Revista mexicana de economía y finanzas*.

Muñoz Pérez, C., Cabrera Padilla, D., Carvajal-Gámez, B. E., Gallegos-Funes, F., & Gendron, D. (2014). *Segmentación automática en imágenes RGB aplicando la técnica Fuzzy C-means de la morfología matemática para la ayuda de la fotoidentificación de cetáceos.* México.

Nehru, J. (2018). *An Extensive Study of Data Analysis Tools (Rapid Miner, Weka, R Tool, Knime, Orange).*

Padmavaty, V., Geetha, C., & Priya, N. (2020). *Analysis of Data Mining Tool Orange.* Chennai, Tamilnadu, India: International Journal of Modern Agriculture.

Patel, P. S., & Desai, S. (2015). *A Comparative Study on Data Mining Tools.* Surat, India: International Journal of Advanced Trends in Computer Science and Engineering.

Pavon, M. F. (2014). *Estimación de datos faltantes con el Algoritmo EM.* México.

Pinar Saygin, A., Cicekli, I., & Akman, V. (2001). *Turing Test: 50 Years Later.* Department of Cognitive Science, University of California, San Diego, La Jolla, CA.

Pinho Lucas, J. (2010). *Métodos de clasificación basados en asociación aplicados a sistemas de recomendación.*

Quinlan, J. (1986). Induction of Decision Trees. *Machine Learing 1: 81 - 106.*

RAE, R. A. (2021). *Diccionario de la lengua española*. Obtenido de Diccionario de la lengua española: https://dle.rae.es/

Ramamohan, Y., Vasantharao, K., Kalyana Chakravarti, C., & Ratnam, A. (2012). *A Study of Data Mining Tools in Knowledge Discovery Process.* International Journal of Soft Computing and Engineering (IJSCE).

Ramírez, F. (20 de 07 de 2018). *Historia de la IA: Frank Rosenblatt y el Mark I Perceptrón, el primer ordenador fabricado específicamente para crear redes neuronales en 1957*. Obtenido de https://web.archive.org/web/20180722124753/https://data-speaks.luca-d3.com/2018/07/historia-de-la-ia-frank-rosenblatt-y-el.html

Ranjan, R., & Agarwal, S. (2017). *Detailed Analysis of Data Mining Tools.* Bangalore: International Journal of Engineering Research & Technology (IJERT).

Ratra, R., & Gulia, P. (2020). Experimental Evaluation of Open Source Data Mining Tools (WEKA and Orange). *International Journal of Engineering Trends and Technology (IJETT) – Volume 68 Issue 8 - Aug 2020*, 30-35.

Rich, E., & Knight, K. (1994). *Inteligencia Artificial. Segunda Edición.* McGraw Hill.

Robles Aranda, Y., & R. Sotolongo, A. (2013). Integración de los algoritmos de minería de datos 1R, PRISM e ID3 a Postgresql. *Revista de Gestão da Tecnologia e Sistemas de Informação*, 7.

Rusell, S., & Meter, N. (1996). *Inteligencia Artificial: Un Enfoque Moderno.* Prentice Hall.

Sánchez Bilbao, P. (2015). *Credit Scoring.* Santander.

Sasaki, Y. (2007). *The truth of the F-measure.* School of Computer Science, University of ManchesterMIB, 131 Princess Street, Manchester, M1 7DN.

Suca, C., Córdova, A., Condori, A., Cayra, J., Sulla, & José. (2016). *Comparación de algortimos de clasificación para la predicción de casos de obesidad infantil.* Obtenido de Comparación de algortimos de clasificación para la predicción de casos de obesidad infantil: https://www.researchgate.net/profile/Abel-Condori-Castro/publication/301567339_COMPARACION_DE_ALGORITMOS_DE_CLASIFICACION_PARA_LA_PREDICCION_DE_CASOS_DE_OBESIDAD_INFANTIL/links/571a985c08aee3ddc568f97d/COMPARACION-DE-ALGORITMOS-DE-CLASIFICACION-PARA-LA-PR

Sutton, S. R., & Barto, G. A. (2005). *Reinforcement Learning: An Introduction.* The MIT Press.

Villazana, S., Arteaga, F., Seijas, C., & Rodríguez, O. (2012). Estudio Comparativo entre Algoritmos de Agrupamiento Basado en SVM y C-Medios Difuso Aplicados a Señales Electrónicas. *Revista Ingeniería UC, Vol. 19, No.1, Abril 2021 16*, 2.

Vizcaino Garzon, P. A. (2008). *Aplicación de técnicas de inducción de árboles de decisión a problemas de clasificación mediante el uso de Weka (Waikato Enviroment for Knowledge Analysis).* Bogotá.

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining Practical Machine Learning Tools and Techniques 3rd ed.* Morgan Kaufmann.

Zupan, B., & Demšar, J. (2013). *Orange: Data mining fruitful and fun.* Tržaška 25, 1000 Ljubljana, Slovenia,University of Ljubljana, Faculty of Computer and Information Science.

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR

Universidad
de Alcalá