**INFORME FINAL**

# Universidad Nacional de La Matanza

## Departamento de Ingeniería e Investigaciones Tecnológicas

**Título del proyecto de investigación:** Análisis Comparativo *de Modelos de Clasificación de Minería de Datos (Data Mining*). Su aplicación en la predicción de perfiles de alumnos en riesgo de deserción.

**Código del proyecto: C176**

**Programa de acreditación: PROINCE**

**Director del proyecto: RYCKEBOER, Hugo Emilio**

**Co-Director del proyecto: SPOSITTO, Osvaldo Mario**

**Integrantes del equipo:**

**CASTRO, Hugo Martín**

**BOSSERO, Julio César**

**GARGANO, Cecilia Victoria**

**MATTEO, Lorena Romina**

**PRILUSKY, Elisa Mirta**

**PROCOPIO, Gastón Emanuel**

**QUINTANA, Fabio Hernán**

**Fecha de inicio:** 01/01/2015

**Fecha de finalización:** 03/12/2016

**Resumen**:

El presente trabajo muestra los resultados de la comparación entre diferentes técnicas de minería de datos (MD) con el objetivo de identificar en forma automática a los estudiantes con mayor riesgo de deserción de las carreras de Ingeniería de la UNLaM, a partir de los datos socioeconómicos y académicos de los mismos.

Para desarrollar este trabajo se aplicó la metodología para proyectos de MD conocida como KDD. Dicha metodología estructura el proceso en seis fases, que interactúan entre ellas de forma iterativa. Se aplicaron las técnicas de Redes Neuronales, Árboles de decisión y K-vecino más próximo, como algoritmos supervisados y K-Means, como no supervisado, para analizar el comportamiento de los estudiantes evaluando la calificación obtenida en las asignaturas comunes del primer año de las distintas carrera, y además otros datos como la edad, el sexo, etc.

Los datos se obtuvieron de un almacén de datos construido para tal fin en proyectos anteriores. La muestra que se utilizó para entrenar los algoritmos fue de 1499 sujetos (cohorte 2013 y 2014). Mientras que para testear los modelos creados se utilizaron 793 alumnos de la cohorte 2015.

Se empleó la herramienta libre Weka y ejecutar y evaluar el desempeño de los algoritmos. Se encontró que el algoritmo del J48 permitió obtener efectividades mayor que los otros algoritmos.

Se observó que todos los algoritmos supervisados, en su etapa de entrenamiento, arrojan un desempeño general, con porcentajes mayores al 90% de exactitud, mientras que estos mismos algoritmos cuando son convertidos a modelos y son testeados con nuevos lotes de prueba, el porcentaje de acierto baja a valores apenas superiores al 50%.

El conocimiento generado permitirá soportar la toma de decisiones eficaces de las directivas universitarias enfocadas a formular políticas y estrategias relacionadas con los programas de retención estudiantil que actualmente se encuentran establecidos.

**Palabras claves:** KDD, Algoritmos Supervisados, Algoritmos No Supervisados, Weka

**Área de conocimiento:** Ingeniería de Comunicaciones, Electrónica y Control

**Código de Área de conocimiento:** 1800

**Disciplina de conocimiento:** Computación

**Código Disciplina de conocimiento:** 1802

**Campo de Aplicación:** Computación

**Código Campo de Aplicación:** 1802

**Otras dependencias de la UNLaM que intervinieron en el Proyecto:** No Aplica.

**Otras instituciones intervinientes en el Proyecto:** No Aplica.

**Otros proyectos con los que se relaciona:** No Aplica.

3. **Resumen**:

**Título del proyecto de investigación**: *Análisis Comparativo de Modelos de Clasificación de Minería de Datos (Data Mining). Su aplicación en la predicción de perfiles de alumnos en riesgo de deserción.*

Se estudiaron y analizaron tres diferentes técnicas de clasificación del tipo supervisado: Árbol de Decisión, Redes Neuronales Artificiales y K Vecinos más Próximo, y una técnica del tipo aprendizaje no supervisado: K-Means.

Como herramienta de entrenamiento y testeo de los algoritmos se utilizó el software Weka, que es una plataforma para el aprendizaje automático.

El desempeño de los algoritmos se analizó desde el resultado obtenido en las matrices de confusión. Se encontró que el algoritmo del J48 permitió obtener efectividades mayores a las de la red neuronal y a K vecino.

Se observó que todos los algoritmos supervisados, en su etapa de entrenamiento, arrojan un desempeño general, con porcentajes mayores al 90% de exactitud, mientras que estos mismos algoritmos cuando son convertidos a modelos y son testeados con nuevos lotes de prueba, el porcentaje de acierto baja a valores apenas superiores al 50%.

Asimismo se cree que con estos resultados, la Universidad podrá generar mejoras en los procesos de predecir los potenciales alumnos que abandonen sus estudios, permitiendo realizar tareas de intervención anticipando la deserción de los mismos.

**Palabras claves:** KDD, Algoritmos Supervisados, Algoritmos No Supervisados, Weka

**Memoria descriptiva**

En esta segunda etapa se llevaron a cabo todas las actividades planeadas en el Gantt informado en el Protocolo de presentación del proyecto. A grandes rasgos se concluyó con el desarrollo de las últimas etapas del desarrollo de un proceso KDD: La etapa de Mineria de Datos y la de Interpretación y evaluación de resultados.

En cuanto a las actividades de difusión de resultados preliminares, se presentó parte de este trabajo en los siguiente eventos:

- "***Predicción del riesgo de abandono universitario utilizando métodos supervisados***". Trabajo presentado en el Workshop de la V Jornadas Nacionales y I Latinoamericanas de Ingreso y Permanencia en Carreras Científico – Tecnológicas. Facultad Regional Bahía Blanca. Universidad Tecnológica Nacional. Bahía Blanca. Mayo de 2016. IPECyT 2016. Disponible en http://www.frbb.utn.edu.ar/ipecyt2016/

- "***Comparación de Algoritmos de Aprendizaje Supervisado para la obtención de perfiles de alumnos desertores***". Trabajo presentado en el "Workshop del V Congreso Nacional de Ingeniería en Informática/Sistemas de Información. Publicación on line - ISSN 2347-0372. CONAIISI 2016. Salta. Disponible en: http://www.ucasal.edu.ar/conaiisi2016/book/memorias.html

## 4. Organización del Informe Final

**Introducción:**

- **Selección del Tema**

La Minería de Datos Educacional (MDE), es una rama de la Minería de Datos (MD) o Data Mining (DM), la cual se ha dedicado a aplicar diversas técnicas para analizar datos provenientes de ambientes relacionados a la educación y a extraer la mayor cantidad de conocimiento para tratar de entender mejor a los estudiantes, profesores y actores relacionados, con el fin de mejorar los procesos educativos [1,2]. Ante un creciente volumen de información por centralizar y estudiar, y en aras de desempeñar una labor más efectiva, se hace necesario emplear técnicas de análisis de datos más complejas y dinámicas a la estadística, para lograr, finalmente, la extracción de conocimiento no implícito. Concretamente, se trata de herramientas como la Minería de Datos, la cual apoya el proceso de KDD, que consiste en analizar grandes volúmenes de datos para generar conocimiento útil a favor de la toma de decisiones.

- **Definición del Problema**

El Departamento de Ingeniería e Investigaciones Tecnológicas (DIIT), de la Universidad Nacional de La Matanza (UNLaM), en el año 2009 realizó un estudio acerca de la utilización de técnicas de MD para la evaluación del rendimiento académico y la deserción universitaria [3]. En esa ocasión se convino en la necesidad de desarrollar herramientas informáticas para la toma de decisiones. Las mismas se llevaron a cabo en años sucesivos, llegando a implementar un Almacén de Datos (AD) o Data Warehouse (DW) departamental. [Anu10] y [Anu12].

Según Hernandez, un DW que es la base para desarrollar un proyecto de MD [Her04]. Uno de los objetivos que se fijó el DIIT fue buscar la forma de prevenir el fenómeno de abandono en las carreras de Ingeniería de la de UNLaM a través de algoritmos de MD, los cuales puedan detectar factores que incidan en la deserción de alumnos universitarios, analizando datos socio-demográficos y académicos de los alumnos en su primer año de estudio.

- **Justificación del Estudio**

La utilización de técnicas de MD en el desarrollo de modelos para estudiar la correlación entre los factores de predicción y determinar la posibilidad de deserción universitaria ha sido objeto de diversos estudios e investigaciones [4,5,6,7,8,9 y 10], entre otros. Esto hace pensar que la cantidad de técnicas distintas que se han ensayado, pone en evidencia el carácter no absoluto de las mismas.

Estudiar varias de estas técnicas sobre una misma población, con una problemática bien definida y en un contexto sobre el cual se puede actuar, puede agregar luz sobre los fundamentos de las mismas y las condiciones que hacen a una, preferible sobre otra. A partir de lo anteriormente planteado, en este trabajo se propone realizar un estudio comparativo sobre algunos de los algoritmos más utilizados en Minería de Datos.

Para ello se creará una vista minable, con la que aplicarán algoritmos para generar modelos. Luego, examinado los datos de la cohorte 2015, se analizará cuál de los modelos creados entrega una respuesta más beneficiosa.

- **Limitaciones**

Para realizar el estudio se analizaron los datos de alumnos de las carreras de Ingenierías Electrónica, Informática, Industrial y Civil, haciendo foco en la cohorte 2013 y 2014, para la construcción de los modelos. Utilizando la cohorte 2015 para validar los modelos construidos.

- **Alcances del Trabajo**

Como ya se comentó en la etapa anterior, los datos se obtuvieron de un almacén de datos institucional. En el primer año del trabajo, se contó con una población de 718 alumnos solo de la cohorte 2013. De este total, 119 estudiantes no presentaron actividad académica en el departamento en el año académico 2014. En el segundo año de este trabajo, a esta cantidad, se le agregaron 781 alumnos, los cuales ingresaron en el año 2014. Un total de 112 alumnos no siguieron con sus estudios en el año 2015, considerándose así, como que los mismos abandonaron sus estudios. Según Tinto [Tin82] "*…un alumno desertor es aquel individuo que siendo estudiante de una institución de educación superior no presenta actividad académica durante tres semestres académicos consecutivos…*". Como la UNLaM cuenta con un curso de verano, el requisito se cumple en un año civil.

Por último, para la evaluación del modelo se utilizó datos de los alumnos ingresantes en el ciclo lectivo 2015.

- **Objetivos**

El proyecto ha planteado inicialmente los siguientes objetivos:

- Los Objetivos Principales:

  - ✓ Estudiar y analizar distintos Algoritmos de Data Mining.
  - ✓ Crear un Modelo de Datos o Vista Minable. Aplicarlo al problema elegido como testeo y, sobre éste, aplicar técnicas de Minería de datos y comparar los resultados que se obtengan.

- Los Objetivos Secundarios:

  - ✓ Llevar a cabo un estudio descriptivo/comparativo de las características particulares de los Algoritmos comúnmente utilizados en Minería de Datos.
  - ✓ Conocer distintos programas informáticos de DM para ser utilizado para el aprendizaje automático de la información.
  - ✓ Enfrentar los resultados computacionales de la minería de datos con estudios extracomputacionales para mejor valoración de los resultados obtenidos.

- **Hipótesis**

  - ✓ "*Los diferentes Modelos de Clasificación de Minería de Datos difieren significativamente en sus predicciones*".
  - ✓ "*El comportamiento de los alumnos actuales respecto de los aspectos analizados en este proyecto es análogo al de los que aparecen en los datos históricos*".

**Desarrollo:**

- **Materiales y Métodos**

Una metodología no solo define las fases de un proceso sino también las tareas que deberían realizarse y cómo llevarlas a cabo. Las mismas permiten llevar a cabo el proceso de minería de datos en forma sistemática y no trivial, ayudan a las organizaciones a entender el proceso de descubrimiento de conocimiento y proveen una guía para la planificación y ejecución de los proyectos. Existen diferentes metodologías para llevar a cabo un proceso de Data Mining. Entre las más conocidas se encuentran:

- ✓ CRISP-DM: (de Cross Industry Standard Process for Data Mining): creada por el grupo de empresas SPSS[1], NCR y Daimer Chrysler en el año 2000, es actualmente la guía de referencia más utilizada en el desarrollo de proyectos de minería de datos.
- ✓ SEMMA: (Sample, Explore, Modify, Model, Assess) se la define como el proceso de selección, exploración y modelado de grandes cantidades de datos para descubrir patrones de negocio desconocidos. Es una metodología utilizada por la firma SAS.
- ✓ KDD: (Knowledge Discovery in Databases o Descubrimiento de Conocimiento en Bases de Datos)

La metodología seguida por este grupo es KDD, la misma es propuesta por Hernández en [Her04]. Esta metodología contiene seis etapas que pretenden gestionar de forma global un proyecto de minería de datos. Las fases son las que aparecen en la figura 1.

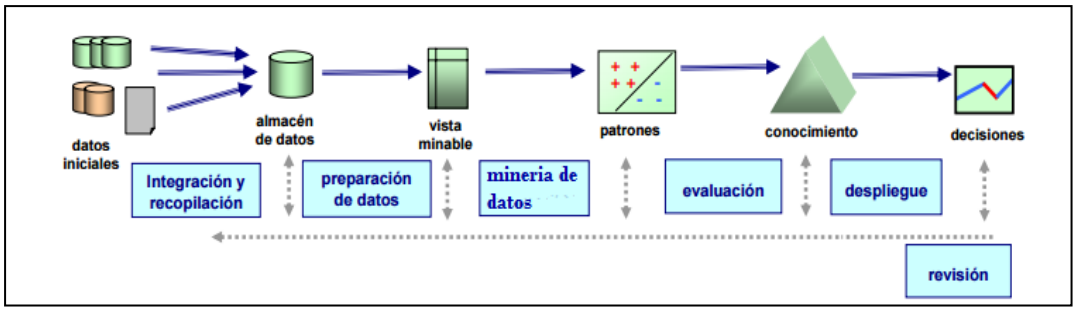



**Figura 1: Etapas que componen un proceso KDD.[Her04]**

Determinar las fuentes de información que pueden ser útiles y dónde conseguirlas, es parte de la primer etapa. En este trabajo los datos fueron extraídos de un almacén de datos, construido, como ya se comentó, por el DIIT en años anteriores. En la figura 2 se puede observar el diseño lógico del mismo.

[1] **SPSS** es un programa estadístico informático muy usado en las ciencias exactas, sociales y aplicadas, además de las empresas de investigación de mercado

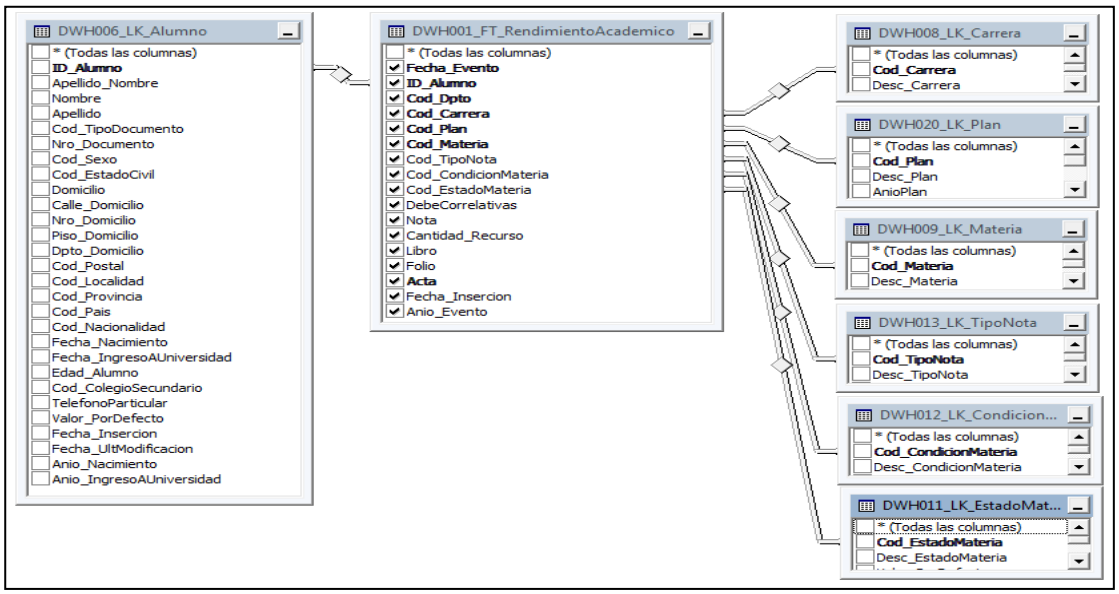



**Figura 2: Esquema relacional de las tablas utilizadas.**

| Variable | Tipo de datos | Tipo de contenido | Valores |
|---|---|---|---|
| Cod_sexo | Numérico | Discreta | 1-2 |
| Estcivil | Numérico | Discreta | 1-2-3 |
| Carrera | Numérico | Discreta | 201-202-203-207 |
| Edad | Numérico | Continuo | 17 a 65 |
| Alg_Geo_Analitica_I | Texto | Discreta | Cursada<br>Promocionada<br>Final Aprobado<br>Cursada Aprobada<br>Final Reprobado<br>Final Ausente<br>Cursada Reprobada<br>Cursada Ausente<br>No Cursada |
| Alg_Geo_Analitica_II | Texto | Discreta | |
| Analisis_Mat_I | Texto | Discreta | |
| Analisis_Mat_II | Texto | Discreta | |
| Computacion_Niv_I | Texto | Discreta | |
| Elementos_Prog_I | Texto | Discreta | |
| Mat_Discreta | Texto | Discreta | |
| Química_Gral | Texto | Discreta | |
| Tecn_Ing_Soc | Texto | Discreta | |
| Sist_Rep | Texto | Discreta | |
| Ingles_Tecn_I | Texto | Discreta | |
| Fisica_I | Texto | Discreta | |
| TICs | Texto | Discreta | |
| Situacion | Texto | Discreta | Abandono<br>No_Abandono |

**Tabla 1. Nómina de variables utilizadas y sus valores.**

Como herramienta de entrenamiento y testeo de los algoritmos se utilizó Weka (Waikato Environment for Knowledge Analysis, en español «Entorno para análisis del conocimiento de la Universidad de Waikato»), que es una plataforma de software para el aprendizaje automático, escrito en Java y desarrollado en la Universidad de Waikato, Nueva Zelanda. Weka es un software libre distribuido bajo la licencia GNU-GPL.[11]

- **Lugar y tiempo de la investigación**

Esta investigación se llevó a cabo en el DIIT, perteneciente a la UNLaM, utilizando los datos de los alumnos de ingeniería de los años 2013,2014 y 2015.

- **Descripción del objeto de estudio**

Esta investigación desde el punto de vista metodológico, es tanto descriptiva como experimental. Se estudiaron y analizaron tres diferentes técnicas de clasificación del tipo Supervisado: Árbol de Decisión, Redes Neuronales y K-vecinos más Próximo, y una técnica del tipo aprendizaje No Supervisado: K-Means, estas técnicas se eligieron por considerarse que son las más utilizadas para resolver este tipo de problema [Her04].

Frawley y sus colegas, en el artículo escrito el año 1992 titulado "*Descubrimiento de conocimiento en bases de datos*", define formalmente a la minería de datos como *"…un conjunto de técnicas y herramientas aplicadas al proceso no trivial de extraer y presentar conocimiento implícito, previamente desconocido, potencialmente útil y humanamente comprensible, a partir de grandes conjuntos de datos, con objeto de predecir, de forma automatizada, tendencias o comportamientos y descubrir modelos previamente desconocidos…*"[12].

Dependiendo del objetivo del análisis de los datos, las técnicas de MD se clasifican en dos grandes categorías de algoritmos de aprendizaje: los No Supervisados y los Supervisados. [Her04]

- **Supervisados:** estos algoritmos predicen el valor de un atributo, llamado etiqueta, dentro de un conjunto de datos, a partir de otros atributos denominados atributos descriptivos. Esta técnica permite deducir una función a partir de un conjunto de datos de entrenamiento.
- **No Supervisados**: Están compuestos por algoritmos que aspiran a descubrir patrones y tendencias sobre el conjunto de datos, sin tener ningún tipo de conocimiento previo de la situación a la cual se quiere llegar. Quedando librado al investigador decidir si los subconjuntos obtenidos tienen una semántica interesante.

**Breve descripción de los algoritmos.**

Es usual en esta disciplina designar clase a cada alternativa que puede tomar el atributo etiqueta, esto permite decir que los métodos de clasificación particionan los datos en clases.

El atributo etiqueta es el objetivo de la predicción en los problemas de clasificación supervisados. Al recibir instancias de valores descriptivas unidas con su etiqueta ya asignada se espera que el método descubra que combinaciones de valores descriptiivas hacen propicio caer en cada una de las clases.

Los algoritmos que estudiaremos a continuación, son dedicados al problema de la clasificación supervisada operan usualmente sobre la información suministrada por un conjunto de muestras, patrones, ejemplos o prototipos de entrenamiento que son asumidos como representantes de las clases (class), y los mismos poseen por definiciòn una etiqueta de clase correcta. A este conjunto de prototipos correctamente etiquetados se le llama conjunto de entrenamiento (TS, training set), y es el conocimiento empleado para la clasificación de nuevas muestras. Estos algoritmos tienen como objetivo determinar cuál es la clase, de las que ya se tiene conocimiento, a la que debe pertenecer una nueva muestra, teniendo en cuenta la información que se puede extraer del conjunto de entrenamiento.

A continuación se presentan los modelos propuestos con sus respectivas características:

- **Árboles de Decisión (AD). Generalidades.**

Este algoritmo pertenece a la familia de algoritmos TDIDT (Top Down Induction of Decision Trees). Es un método inductivo del Aprendizaje Automático que aprende a partir de ejemplos preclasificados. También es denominado Árbol de Clasificación. Existe una serie de algoritmos desarrollados desde los principios de los 60's para la construcción de AD: CLS, ID3, CART, ACLS, ASSISTANT y C4.5, entre otros. [Her04]. Estos algoritmos generan árboles y reglas de decisión a partir de datos preclasificados. Para construir los árboles se utiliza el método de aprendizaje "***divide y reinarás***". Particiona el conjunto de muestras en subconjuntos a medida que avanza; trabajar sobre cada subconjunto es más sencillo que trabajar sobre el total de los datos. Los subconjuntos son elegidos de modo tal de concentrar elementos de una misma etiqueta en cada subconjunto. A su vez cada subconjunto volverá a ser procesado salvo que ya sea puro. Divide utilizando una variable por vez Las variables continuas le permiten planear divisiones en dos ramas: una para aquellos $A_i <= N$ y otra para $A_i > N$. Las variables discretas dividen en tantas ramas como instancias de la misma se halle en el conjunto. Este algoritmo fue propuesto por Quinlan en 1993 [13].

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 puede tomar el atributo. Para cada atributo ordenado, se realiza una prueba binaria sobre cada uno de los valores que toma el atributo en los datos.La idea subyacente al algoritmo es que, mientras que todos los patrones que se correspondan con una determinada rama del árbol de clasificación no pertenezcan a una misma clase, se seleccione la variable que, de entre las no seleccionadas en esa rama sea la más informativa o la más idónea con respecto de un criterio previamente establecido. La elección de esta variable sirve para expandir el árbol en tantas ramas como posibles valores toma dicha variable [Her04].

Un esquema sencillo puede apreciarse en la Figura 3, este ejemplo para el concepto "***buen día para jugar tenis***", muestra unos patrones de entrenamiento caracterizados por 4 variables predictoras denotadas por: ***Ambiente, Temperatura, Humedad y Viento*** y una variable clase ***C*** con dos valores posibles denotados por ***N*** *(No puedo Jugar)* y ***P*** *(Puedo Jugar)*.

Una posible representación del conocimiento subyacente al dominio del ejemplo mostrado en la Figura 3 lo podemos ver por medio del árbol de clasificación de la Figura 4.

| Ambiente | Temp. | Humedad | Viento | Clase |
|---|---|---|---|---|
| soleado | alta | alta | no | N |
| soleado | alta | alta | si | N |
| nublado | alta | alta | no | P |
| lluvia | media | alta | no | P |
| lluvia | baja | normal | no | P |
| lluvia | baja | normal | si | N |
| nublado | baja | normal | si | P |
| soleado | media | alta | no | N |
| soleado | baja | normal | no | P |
| luvia | media | normal | no | P |
| soleado | media | normal | si | P |
| nublado | media | alta | si | P |
| nublado | alta | normal | no | P |
| lluvia | media | alta | si | N |

**Figura 3: Conjunto de ejemplos de entrenamiento para el concepto objetivo ¿jugar tenis?**

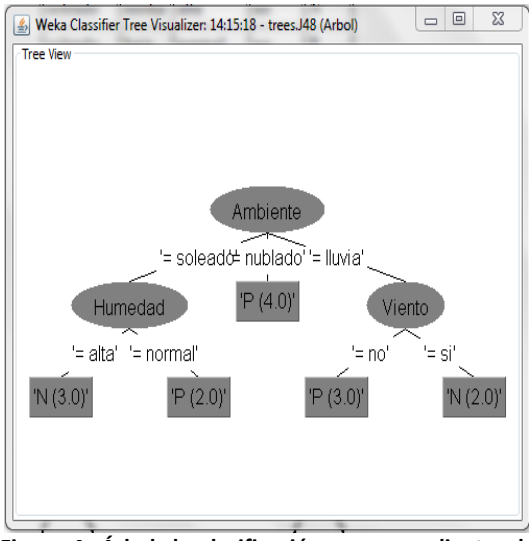



**Figura 4: Árbol de clasificación correspondiente al ejemplo representado en la Figura 2.**

Tal y como puede verse en la Figura 4, las variables predictoras se van a representar en el árbol de clasificación insertadas en un círculo, mientras que las hojas del árbol se representan por medio de un rectángulo en el cual se inserta el valor de la variable clase que el AD asigna a aquellos casos que bajan por las correspondientes ramas del Arbol. La Figura 4 tiene para todas las ramas una profundidad de 2, siendo este concepto de profundidad el que proporciona una idea de la complejidad del árbol y del tiempo de ejecució de una clasificación. Por otra parte en este ejemplo se puede notar que no existe ruido en el sentido de que las hojas son puras al contener tan sólo patrones de un determinado tipo respecto de la variable clase. Finalmente, se puede expresar el árbol de clasificación de la Figura 4 por medio de un conjunto de reglas.

Así las reglas R1 a R4 consideradas en su globalidad son equivalentes al AD anterior:

- R1: ***Si*** (Ambiente = Soleado y Humedad = Alta → Jugar Tenis = No )
- R2: ***Si*** (Ambiente = Soleado y Humedad = Normal → Jugar Tenis = Si )
- R3: ***Si*** (Ambiente = Nublado → Jugar Tenis = Si )
- R4: ***Si*** (Ambiente = Lluvioso y Viento = Si → Jugar Tenis = No)
- R5: ***Si*** (Ambiente = Lluvioso y Viento = No → Jugar Tenis = Si)

- **K-Vecino más próximo (K-NN, K -Nearest Neighbour). Generalidades.**

La idea básica sobre la que se fundamenta este paradigma es que un nuevo caso se va a clasificar en la clase más frecuente a la que pertenecen sus K vecinos más cercanos. El algoritmo se fundamenta, por tanto, en una idea muy simple e intuitiva, lo que unido a su fácil implementación hace que sea un paradigma clasificatorio muy extendido [14].

El algoritmo K-NN básico funciona de la siguiente forma:[Her04]

- ***D*** indica un fichero de ***N*** casos, cada uno de los cuales está caracterizado por ***n*** variables predictoras, cada una acompañada de una instancia de la variable a predecir, una de las clases de ***C***.
- Los ***N*** casos se denotan por

  $(\mathbf{x}_1, \mathbf{c}_1), \ldots, (\mathbf{x}_N, \mathbf{c}_N)$ donde

  $\mathbf{x}_i = (x_{i,1} \ldots x_{i,n})$ para todo $i = 1, \ldots, N$

  $\mathbf{c}_i \in C$ para todo $i = 1, \ldots, N$

  $x_{i,j}$ es un valor de la *j*-ésima variable predictora.

  (en nuestro caso ***C*** es de cardinalidad 2 : abandona no-abandona)
- El nuevo caso que se pretende clasificar se denota por $\mathbf{x} = (x_1 \ldots x_n)$

Una vez determinado el conjunto de los *K* casos ya clasificados, $D_K(\mathbf{x})$ más cercanos al nuevo caso $\mathbf{x}$, a éste se le asignará la clase (valor de la variable C) más frecuente de entre los *K* objetos, $D_K(\mathbf{x})$. Si hubiera empate se replantea con un *K* menor. La Figura 5 muestra de manera gráfica un ejemplo de lo anterior.

| | | $X_1$ | … | $X_j$ | … | $X_n$ | $C$ |
|---|---|---|---|---|---|---|---|
| $(\mathbf{x}_1, c_1)$ | 1 | $x_{11}$ | … | $x_{1j}$ | … | $x_{1n}$ | $c_1$ |
| | ⋮ | ⋮ | | ⋮ | | ⋮ | ⋮ |
| $(\mathbf{x}_i, c_i)$ | $i$ | $x_{i1}$ | … | $x_{ij}$ | … | $x_{in}$ | $c_i$ |
| | ⋮ | ⋮ | | ⋮ | | ⋮ | ⋮ |
| $(\mathbf{x}_N, c_N)$ | $N$ | $x_{N1}$ | … | $x_{Nj}$ | … | $x_{Nn}$ | $c_N$ |
| $\mathbf{x}$ | $N+1$ | $x_{N+1,1}$ | … | $x_{N+1,j}$ | … | $X_{N+1,n}$ | ? |

**Figura 5: Notación para el Algoritmo K-NN**

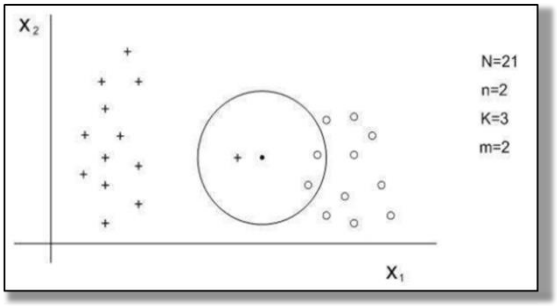



**Figura 6: Ej. de aplicación del algoritmo K-NN básico**

Tal y como puede verse en la Figura 5 tenemos 24 casos ya clasificados en dos posibles valores (***m*** = ***2***).

Las variables predictoras son $X_1$ y $X_2$, y se ha seleccionado ***K***=3. De los 3 casos ya clasificados que se encuentran más cercanos al nuevo caso a clasificar, x (representado por un punto "●" ), dos de ellos pertenecen a la clase circulito "○", por tanto el clasificador 3-NN predice la clase ○ para el nuevo caso.

Nótese que el caso más cercano a $x$ pertenece a la clase **+**. Es decir, que si se hubiese utilizado un clasificador 1-NN, x se hubiese asignado a **+**.

- **Redes Neuronales Artificiales. Generalidades.**

Las Redes Neuronales Artificial (RNA o ANN en inglés Artificial Neural Networks) son modelos artificiales de las redes neuronales biológicas. Estas redes están constituidas por innumerables unidades procesadoras, llamadas neuronas, Figura 7, las cuales se interconectan a través de conexiones [Her04]. En una neurona se pueden distinguir las siguientes partes:

- ✓ **Dendritas.** Recogen señales procedentes de otras neuronas, o del mundo externo y las transmiten al cuerpo de la neurona.
- ✓ **Soma o Cuerpo.** Recibe las señales de todas las dendritas y según la tipología de la neurona determina una respuesta global.
- ✓ **Axón.** Transmite la respuesta global generada por el cuerpo de la neurona. La respuesta se transmite a otras neuronas o a algún miembro efector.
- ✓ **Sinapsis.** Es el punto de unión entre dos neuronas (dendrita - axón), y por el cual el impulso transmitido por el axón se pasa a otras neuronas.

Las neuronas establecen conexiones entre ellas (sinapsis), de manera que cuando un animal recibe un estímulo, ciertas conexiones se refuerzan más que otras, provocando una cierta respuesta. Siempre que el animal reciba un estímulo (entrada) similar, generará la misma respuesta (aprendizaje): se puede decir que el cerebro reconoce diferentes patrones.

**Neurona artificial**

En la actualidad existen dos conceptos bastante aceptados de neurona artificial.

- Unidad de procesamiento encargada de simular el comportamiento de una neurona biológica.
- Modelo matemático que trata de emular de manera simplificada el comportamiento de una neurona biológica.

**Modelo de McCulloch-Pitts**

Modelo de neurona binaria propuesto por McCulloch y Pitts en 1943 [15], Figura 7, el cual calcula la suma ponderada de sus entradas producidas por otras unidades, y da como salida un uno (1) si ésta se encuentra por encima de un valor denominado umbral, o un cero (0) si está por debajo. La ecuación que gobierna el funcionamiento de dicho modelo de neurona es la siguiente:

$$n_i(t+1)=f(\Sigma_j(w_{ij}*n_j(t)) - u_i) \quad (1)$$

donde,

$w_{ij}$ es el peso de la conexión entre la neurona *j* y la neurona *i*

$n_j(t)$ es la salida producida por la neurona *j*

$u_i$ es el umbral de la neurona *i*

$f(x) = 0$ *si* $x < u$

1 en otro caso. Esta es la función umbral.

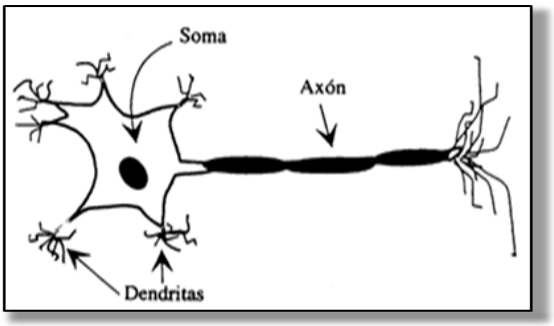



**Figura 7: Neurona física.**

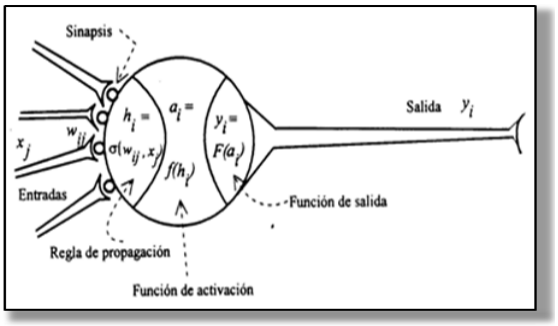



**Figura 8: Representación matemática de una neurona.**

En esencia las redes neuronales son capaces de realizar dos tareas diferentes: el reconocimiento de patrones y la síntesis funcional. Aunque parecen tener la misma capacidad de cómputo que una ***máquina de Turing***[2] no se debe esperar que una red neuronal realice tareas que ya tienen una solución algorítmica buena, por ejemplo, invertir una matriz.

- **Reconocimiento de patrones**: El reconocimiento de patrones, implica la clasificación de información según ciertas características. Aplicaciones típicas son la diferenciación de sonidos muy similares, el reconocimiento de escritura manuscrita, interpretación de encefalogramas, reconocimiento de la voz y procesamiento de imágenes.

- **Síntesis Funcional**: La aproximación de funciones, consiste en establecer relaciones entre varias entradas continuas (discretas) y una o más salidas continuas (discretas), por ejemplo, estimar la demanda de un producto, filtrar el ruido de una señal, controlar un proceso y simular el comportamiento de un sistema dinámico.

**Antecedentes:**

**El Perceptron**

Una de las características más significativas de las redes neuronales es su capacidad para aprender a partir de alguna fuente de información interactuando con su entorno. En 1958 el psicólogo Frank Rosenblatt [16] desarrolló un modelo simple de neurona basado en el modelo de McCulloch y Pitts y en una regla de aprendizaje basada en la corrección del error. [Her04]. A este modelo le llamó Perceptrón. Figura 9. Una de las características que más interés despertó de este modelo fue su capacidad de aprender a reconocer patrones.

El Perceptrón está constituido por un conjunto de sensores de entrada que reciben los patrones de entrada a reconocer o clasificar y una neurona de salida que se ocupa de clasificar a los patrones de entrada en dos clases, según que la salida de la misma sea 1 (activada) o 0 (desactivada). Sin embargo, dicho modelo tenía muchas limitaciones, como por ejemplo, no ser capaz de aprender la función lógica ***XOR***.

.

[2] Es un dispositivo que manipula símbolos sobre una tira de cinta de acuerdo a una tabla de reglas. A pesar de su simplicidad, una máquina de Turing puede ser adaptada para simular la lógica de cualquier algoritmo de computador y es particularmente útil en la explicación de las funciones de una CPU dentro de un computador.

**El Perceptrón Multicapa**

El Perceptrón Multicapa (PM o MLP en inglés Multi-Layer Perceptron) es una extensión del perceptrón simple. La topología de un perceptrón multicapa está definida por un conjunto de capas ocultas, una capa de entrada y una de salida, Figura 9, esto le permite resolver problemas que no son linealmente separables, lo cual es la principal limitación del perceptrón (también llamado perceptrón simple). Este tipo de redes se caracterizan por su facilidad de implementación. [Her04]

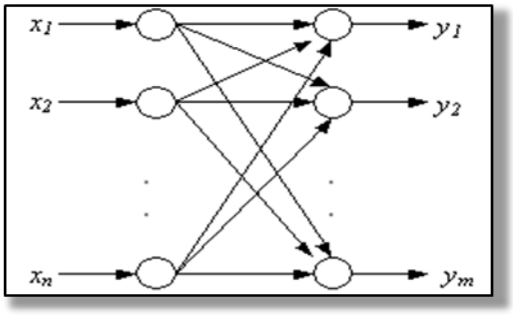



**Figura 9: Perceptrón simple.**

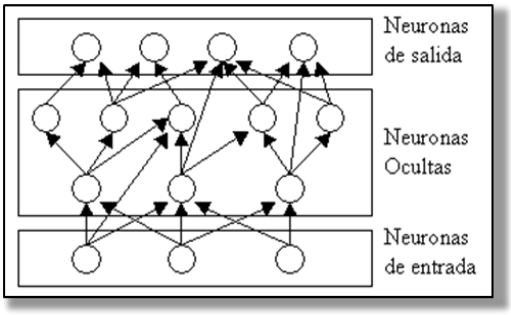



**Figura 10: Perceptrón Multicapa**

Su aprendizaje se basa en el algoritmo de retropropagación (en inglés Backpropagation). Para un conjunto de entradas se obtiene una cierta salida. Basándose en que se conoce la salida que se debería haber obtenido (patrón catalogado), es decir, aprendizaje supervisado y se calcula el error. A partir de este error se modifican los pesos siguiendo el sentido inverso al de evolución de la Red (se parte de la salida hasta llegar a la entrada). De la misma manera se opera con el resto de entradas de entrenamiento. El error irá disminuyendo a medida que se aplique el algoritmo. Entonces, cada neurona recibe un error que es proporcional a su contribución sobre el error total de la red. Basándose en el error recibido, se ajustan los errores de los pesos sinápticos de cada neurona.

- **K-Media. Generalidades.**

El análisis cluster de K-Medias (en inglés K-Means) es un método de agrupamiento por vecindad en el que se parte de un número determinado de prototipos y de un conjunto de ejemplos a agrupar, sin etiquetar. [Her04]. Este algoritmo fue creado por MacQueen en 1967, puede asignar casos a un número fijo de grupos (también se denominados ***clusters o conglomerados***) cuyas características no se conocen aún pero que se basan en un conjunto de variables especificadas. Éste es un algoritmo del tipo ***No Supervisado***.

El algoritmo parte de un conjunto de casos u objetos, cada uno de los cuales está caracterizado por varias variables, y a partir de dicha información trata de obtener grupos de objetos, de tal manera que los objetos que pertenecen a un grupo sean lo más homogéneos posibles entre sí y, por otra parte, la heterogeneidad entre los distintos grupos sea lo mas elevada posible. Expresado en términos de variabilidad se hablaría de ***minimizar la variabilidad*** dentro de los grupos, para al mismo tiempo, ***maximizar la variabilidad*** entre los distintos grupos. Un buen análisis cluster es:

- **Eficiente:** Utiliza tan pocos conglomerados como sea posible.
- **Efectivo:** Captura todos conglomerados estadística y comercialmente importantes. Por ejemplo, un conglomerado con cinco clientes puede ser estadísticamente diferente pero no muy beneficioso.

Esta técnica sigue un procedimiento simple de clasificación de un conjunto de objetos en un determinado número ***K*** de grupos o clústeres, ***K*** determinado a priori. El nombre de ***K-means*** viene porque representa cada uno de los clusters por la media (o media ponderada) de sus puntos, es decir, por su centroide. La representación mediante centroides tiene la ventaja de que tiene un significado gráfico y estadístico inmediato. Cada cluster por tanto es caracterizado por su centro o centroide (Figura 11) que se encuentra en el centro o el medio de los elementos que componen el cluster.

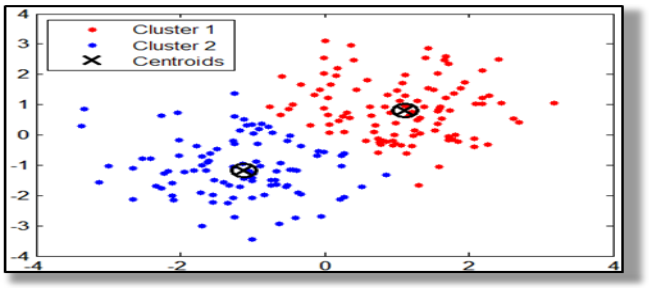



**Figura 11: Representación gráfica K-medias**

Siendo ***O*** un conjunto ***D*** de ***N*** objetos, cada uno de los cuales está caracterizado por ***n*** variables reales.

El *i*-ésimo objeto se designa $\mathbf{x}_i = (x_{i,1} \ldots x_{i,n})$. el centro del j-ésimo cluster se designará $\boldsymbol{v}_j$ .

El algoritmo del K-means se realiza en 3 etapas:

- **Etapa 1**: Elegir aleatoriamente K objetos xj que son asignados como únicos integrantes de los K clusters iniciales. Para el j-ésimo cluster, el valor inicial de su centro es vj = xj.

Repetir las etapas 2 y 3 hasta que no se hagan más reasignaciones:

- **Etapa 2**: Reasigna los objetos de D. Para cada uno de los N objetos x, el prototipo que se le asigna es aquel que corresponde al centro más próximo al objeto, según una medida de distancia, (habitualmente la medida euclidiana), y al azar si hubiera empate.
- **Etapa 3:** Una vez que todos los objetos son colocados, recalcular los centros de los ***K*** cluster's. (los baricentros).

**Nota:** Aunque el algoritmo termina siempre, no se garantiza el obtener la solución óptima. En efecto, el algoritmo es muy sensible a la elección aleatoria de los ***K*** centros iniciales. Esta es la razón por la que se utiliza el algoritmo del ***K-means*** numerosas veces sobre un mismo conjunto de datos, para intentar minimizar este efecto, sabiendo que: centros iniciales lo más espaciados posible dan mejores resultados.

El algoritmo propuesto por McQueen comienza considerando los k primeros elementos del fichero de casos como los k centroides iniciales, o dicho de forma, equivalente como conglomerados con un único elemento. A continuación, y siguiendo el orden establecido en el fichero, cada uno de los objetos se va asignando al conglomerado con centroide más próximo, con la característica de que al efectuar cada asignación, se recalculan las coordenadas del nuevo centroide.

Finalmente, una vez asignados todos los objetos, se calculan los centroides para cada uno de los conglomerados y se reasigna cada objeto al centroide más cercano sin que en este paso se lleve a cabo un recalculo del centroide para cada asignación. Los pasos anteriores se iteran hasta que se verifique un determinado criterio de parada (ejemplos de los cuales han sido comentados con anterioridad). Es importante tener en cuenta que el algoritmo de McQueen es sensible al orden con el que se encuentran los objetos en el fichero de casos, y fundamentalmente es sensible a los objetos que se encuentran en las K primeras posiciones.

- **Descripción de Población y Muestra.**

La población para la aplicación de los algoritmos en este proyecto son los datos recolectados del sistema de información SIU-Guarani[3], que es el encargado de llevar los registros de todos los estudiantes de cada una de las carreras de ingenieria de la Universidad Nacional de La Matanza. De este sistema es de donde se alimenta el Almacén de Datos o Data Warehouse (DW) Departamental del DIIT, el cual fue desarrollado en anteriores proyectos ProInce: C123 y C150. El DW contiene los datos del SIU-Guaranì desde 2013.

- **Diseño de la investigación**

En este trabajo se compararon 4 clasificadores pertenecientes a dos paradigmas de aprendizaje automático. Se utilizó el paquete de software libre Weka 3.6 que contiene implementaciones en lenguaje Java de estos algoritmos [Wit11]. A continuación se describen los pasos necesarios para ejecutar estos algoritmos, mencionando solo algunas de las características de los mismo. Asimismo los parámetros que ofrece Weka para configurar los algoritmos fueron probados para obtener y analizar los resultados arrojados. A menos que se mencione lo contrario, los restantes parámetros son los predeterminados por el software.

La primera tarea que se abordó fue la preparación y adecuación de los datos para el desarrollo de este trabajo. Para seleccionar los atributos de mayor relevancia se utilizaron variados métodos de selección de atributos disponibles en Weka (ver informe de avance 2015). La selección de atributos incluyó la combinación de una búsqueda, con la estimación de la utilidad del atributo, más su evaluación respecto a un esquema de aprendizaje específico [Wit11].

En este segundo año se incorporaron nuevos atributos (Ingles_Tecn_I, Fisica_I y TICs). Durante el primer año del proyecto se quiso tener un modelo comparable con otro prexistente. En esta nueva etapa se privilegiò la calidad de la predicciòn y por lo tanto si habìa datos acadèmicos adicionales era conveniente incorporarlos. Esta decisiòn obligò a repetir la selecciòn de atributos ya que el repertorio es más amplio.

Al igual que en el experimento realizado el año pasado, se volvieron a utilizar los mismo cuatro sistemas evaluadores de subconjuntos de atributos que se encuentran en Weka, Los dos primeros son clasificados como ***Filter*** y los restantes como ***Wrappers***, estos se detallan más adelante. Se ejecutaron en combinación con el método de búsqueda ***Best First***. Dado el tamaño moderado del conjunto de entrenamiento se decidiò utilizar la totalidad de los los datos a través de la opción ***Use full training set.*** Para el entrenamiento de estos métodos se empleó la ***validación cruzada*** en su variante , **k-fold cross validation** (la que se explica màs adelante)**,** la ùnica variante que ofrece Weka. que sirve para estimar la exactitud del

[3] http://www.siu.edu.ar/siu-guarani/

esquema de aprendizaje en cada conjunto. Los métodos de selecciòn y evaluación fueron los siguientes (acompañados de la explicaciòn que provee Weka):

- **CfsSubsetEval**: Evalúa un subconjunto de atributos considerando la habilidad predictiva individual de cada variable, así como el grado de redundancia entre ellas. Se prefieren los subconjuntos de atributos que estén altamente correlacionados con la clase y tengan baja intercorrelación.
- **ConsistencySubsetEval**: Evalúa un subconjunto de atributos por el nivel de consistencia en los valores de la clase al proyectar las instancias de entrenamiento sobre el subconjunto de atributos.
- **ClassifierSubsetEval:** Evalúa los subconjuntos de atributos en los datos de entrenamiento o en un conjunto de prueba independiente, utilizando un clasificador. En este caso utilizamos el método SimpleLogistic (Regresión Lógica).
- **WrapperSubsetEval:** Evalúa los subconjuntos de atributos utilizando un clasificador (también se volvió a utilizar el método SimpleLogistic.

En la siguiente tabla se pueden observar los atributos que arrojaron cada una de las pruebas realizadas.

| | |
|---|---|
| • Evaluator: weka.attributeSelection.CfsSubsetEval<br>• Search:weka.attributeSelection.BestFirst -D 1 -N 5 | Attribute Subset Evaluator (supervised, Class (nominal): 18 Situacion): CFS Subset Evaluator Including locally predictive attributes<br>Selected attributes: 6,9,10,11,12,13,14,15,17 : 9<br>ALG_GEO_ANALITICA_II - COMPUTACION_NIV_I<br>ELEMENTOS_PROG_I - MAT_DISCRETA<br>QUIMICA_GRAL - TECN_ING_SOC<br>SIST_REP - INGLES_TECN_I - TICs |
| • Evaluator: weka.attributeSelection.ConsistencySubsetEval<br>• Search:weka.attributeSelection.BestFirst -D 1 -N 5 | Attribute Subset Evaluator (supervised, Class (nominal): 18 Situacion): Consistency Subset Evaluator<br>Selected attributes: 4,5,7,9,10,11,12,13,14,15,17 : 11<br>Edad - ALG_GEO_ANALITICA _I - ANALISIS_MAT_I<br>COMPUTACION_NIV_I - ELEMENTOS_PROG_I<br>MAT_DISCRETA - QUIMICA_GRAL<br>TECN_ING_SOC - SIST_REP<br>INGLES_TECN_I - TICs |
| • Evaluator: weka.attributeSelection.ClassifierSubsetEval -B weka.classifiers.functions.SimpleLogistic -T -H "Click to set hold out or test instances" -- -I 0 -M 500 -H 50 -W 0.0<br>• Search:weka.attributeSelection.BestFirst -D 1 -N 5 | Attribute Subset Evaluator (supervised, Class (nominal): 18 Situacion):Classifier Subset Evaluator Learning scheme: weka.classifiers.functions.SimpleLogistic<br>Scheme options: -I 0 -M 500 -H 50 -W 0.0<br>Hold out/test set: Training data<br>Accuracy estimation: classification error<br>Selected attributes: 2,4,5,8,9,12,13,14,15,16,17 : 11<br>EstCivil - Edad - ALG_GEO_ANALITICA_I<br>ANALISIS_MAT_II - COMPUTACION_NIV_I<br>QUIMICA_GRAL - TECN_ING_SOC<br>SIST_REP - INGLES_TECN_I - FISICA_I - TICs |
| • Evaluator: weka.attributeSelection.WrapperSubsetEval –B weka.classifiers.functions.SimpleLogistic -F 5 -T 0.01 -R 1 -- -I 0 -M 500 -H 50 -W 0.0 Search:weka.attributeSelection.BestFirst -D 1 -N 5<br>• Search Method: Best first. Start set: no attributes Search direction: forward Stale search after 5 node expansions Total number of subsets evaluated: 184 | Attribute Subset Evaluator (supervised, Class (nominal): 18 Situacion): WrapperSubset Evaluator earning scheme: weka.classifiers.functions.SimpleLogistic<br>Scheme options: -I 0 -M 500 -H 50 -W 0.0 Subset evaluation: classification accuracy<br>Number of folds for accuracy estimation: 5<br>Selected attributes: 2,4,8,9,10,12,13,14,15 : 9<br>EstCivil - Edad - ANALISIS_MAT_II<br>COMPUTACION_NIV_I - ELEMENTOS_PROG_I<br>QUIMICA_GRAL - TECN_ING_SOC<br>SIST_REP - INGLES_TECN_I |

**Tabla 2. Nómina de atributos por metodo de selección.**

Una vez que se obtuvieron los distintos conjuntos de atributos, ver Tabla 2, se procedió a utilizarlos con cada uno de los algoritmos supervisados propuestos, para comprobar con cual de ellos se obtiene mejor porcentaje de casos correctamente clasificados. En esta comparación se incluyó además una prueba con todos los atributos originales. En la Tabla 3, se pueden observar los porcentajes obtenidos.

| | RNA | Arbol de Decisión | K-vecinos | K-Media |
|---|---|---|---|---|
| | Porcentaje de Instancias Correctamente Clasificadas | | | |
| **Todos los atributos** | **89.8599 %** | **91.1274 %** | **87.3249 %** | **66.52 %** |
| **CfsSubsetEval**: | **89.9933 %** | **91.4610 %** | **90.1268 %** | **68.55 %** |
| **ConsistencySubsetEval** | **89.7932 %** | **91.1941 %** | **87.7919 %** | **66.52 %** |
| **ClassifierSubsetEval** | **88.9927 %** | **90.7939 %** | **87.2582 %** | **74.45 %** |
| **WapperSubsetEva** | **96.7312 %** | **91.1941 %** | **88.7258 %** | **68.12 %** |

**Tabla 3. Porcentajes de Instancias correctamente clasificadas según algoritmo y cantidad de atributos.**

Es de destacar de este cuadro que el uso de la totalidad de los atributos no significó ninguna mejora en la calidad de la clasificación, lo que muestra la conveniencia de recurrir a una selecciòn de atributos por el ahorro de tiempo que habrà en la clasificaciòn, sin pèrdida de calidad en la misma.

Cabe aclarar que para este ensayo se usaron los 4 clasificadores adoptando en sus parámetros propios los valores sugeridos por **Weka**. Como se puede observar no hubo ningún método supervisado que haya tenido gran diferencia en cuanto a los resultado obtenido con los otros algoritmos del mismo tipo. Si hubo una diferencia considerable entre ellos y el método No Supervisado. Se optó por el método **CfsSubsetEval** que mejor clasificó en 3 de los 4 algoritmos propuestos.

La manera de clasificar los resultados académicos en el SIIU-Guaraní no ha sido la misma todos los años, lo cual impediría unir lotes provenientes de distinto año. Por este motivo la escala discreta de valores utilizada para describir los rendimientos académicos en cada asignatura ha fusionado las escalas de cada año en una única (4ta. columna de la tabla 1).

Weka tiene una opción para ver las distribución de las variables ***Clase*** (***Class),*** esta es la variable que contiene como valor quien abandonó o no, es decir, es la que utiliza Weka para entrenar a los algoritmos. En la Figura 13, se puede observar el histograma donde el color azul representa el valor "***NO Abandono***" y el rojo el valor "***Abandono"***.

```
@relation 'Datos_Inv_2013-2014_8atrib-weka.filters.unsupervised.attribute.Remove-R1-5,7-8,16'

@attribute ALG_GEO_ANALITICA_II {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute COMPUTACION_NIV_I {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute ELEMENTOS_PROG_I {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute MAT_DISCRETA {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute QUIMICA_GRAL {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute TECN_ING_SOC {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute SIST_REP {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute INGLES_TECN_I {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute TICs {'[Cursada Promocionada]','[Final Aprobada]','[Cursada Aprobada]','[Final Reprobada]','[Final Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada]}
@attribute Situacion{NO_ABANDONO,ABANDONO}

@data
[No_Cursada],'[Cursada Promocionada]',[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]','[Cursada Promocionada]','[Cursada Promocionada]',[No_Cursada],'[Cursada Reprobada]',NO_ABANDONO
[No_Cursada],[No_Cursada],'[Cursada Promocionada]','[Cursada Ausente]','[Cursada Promocionada]','[Cursada Promocionada]','[Cursada Promocionada]',[No_Cursada],'[Cursada Promocionada]',
NO_ABANDONO
[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]',[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Ausente]',ABANDONO
[No_Cursada],[No_Cursada],'[Cursada Ausente]',[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]',[No_Cursada],[No_Cursada],ABANDONO
'[Cursada Ausente]',[No_Cursada],'[Cursada Ausente]','[Cursada Promocionada]','[Cursada Reprobada]','[Cursada Aprobada]','[Cursada Promocionada]',[No_Cursada],'[Cursada Reprobada]',
NO_ABANDONO
[No_Cursada],'[Cursada Promocionada]',[No_Cursada],'[Cursada Ausente]','[Cursada Reprobada]','[Cursada Ausente]','[Cursada Promocionada]',[No_Cursada],'[Final Aprobada]',NO_ABANDONO
[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]','[Cursada Ausente]','[Cursada Ausente]','[Cursada Ausente]','[Cursada Ausente]',ABANDONO
[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Promocionada]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada],'[Cursada Promocionada]','[Cursada Ausente]',NO_ABANDONO
'[Cursada Ausente]','[Cursada Ausente]','[Cursada Ausente]','[Cursada Promocionada]','[Cursada Ausente]','[Cursada Ausente]','[Cursada Ausente]',[No_Cursada],'[Cursada Ausente]',NO_ABANDONO
[No_Cursada],'[Cursada Promocionada]','[Cursada Ausente]','[Cursada Ausente]','[Cursada Promocionada]','[Cursada Aprobada]','[Cursada Ausente]',[No_Cursada],'[Cursada Ausente]',NO_ABANDONO
[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Promocionada]','[Cursada Ausente]','[Cursada Ausente]','[Cursada Ausente]',[No_Cursada],'[Cursada Ausente]',NO_ABANDONO
'[Cursada Promocionada]','[Cursada Promocionada]',[No_Cursada],'[Cursada Ausente]','[Cursada Promocionada]','[Cursada Ausente]','[Cursada Promocionada]','[Cursada Promocionada]','[Cursada
Promocionada]', NO_ABANDONO
[No_Cursada],'[Cursada Promocionada]','[Cursada Ausente]','[Cursada Reprobada]','[Cursada Ausente]',[No_Cursada],'[Cursada Ausente]',[No_Cursada],'[Final Aprobada]',NO_ABANDONO
[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]','[Cursada Ausente]',[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]',ABANDONO
[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]',[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Ausente]',ABANDONO
[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]','[Cursada Reprobada]',[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Ausente]',ABANDONO
[No_Cursada],[No_Cursada],[No_Cursada],'[Cursada Ausente]','[Cursada Ausente]',[No_Cursada],'[Cursada Ausente]',[No_Cursada],'[Cursada Ausente]',ABANDONO
[No_Cursada],'[Cursada Ausente]',[No_Cursada],'[Cursada Promocionada]','[Cursada Ausente]','[Cursada Ausente]',[No_Cursada],'[Cursada Promocionada]','[Cursada Ausente]',NO_ABANDONO
[No_Cursada],'[Final Aprobada]','[Cursada Promocionada]','[Cursada Promocionada]','[Cursada Ausente]',[No_Cursada],'[Cursada Ausente]','[Cursada Promocionada]','[Cursada
Promocionada]',NO_ABANDONO
```

**Figura 12: Encabezado del archivo .arff usado para las pruebas.**

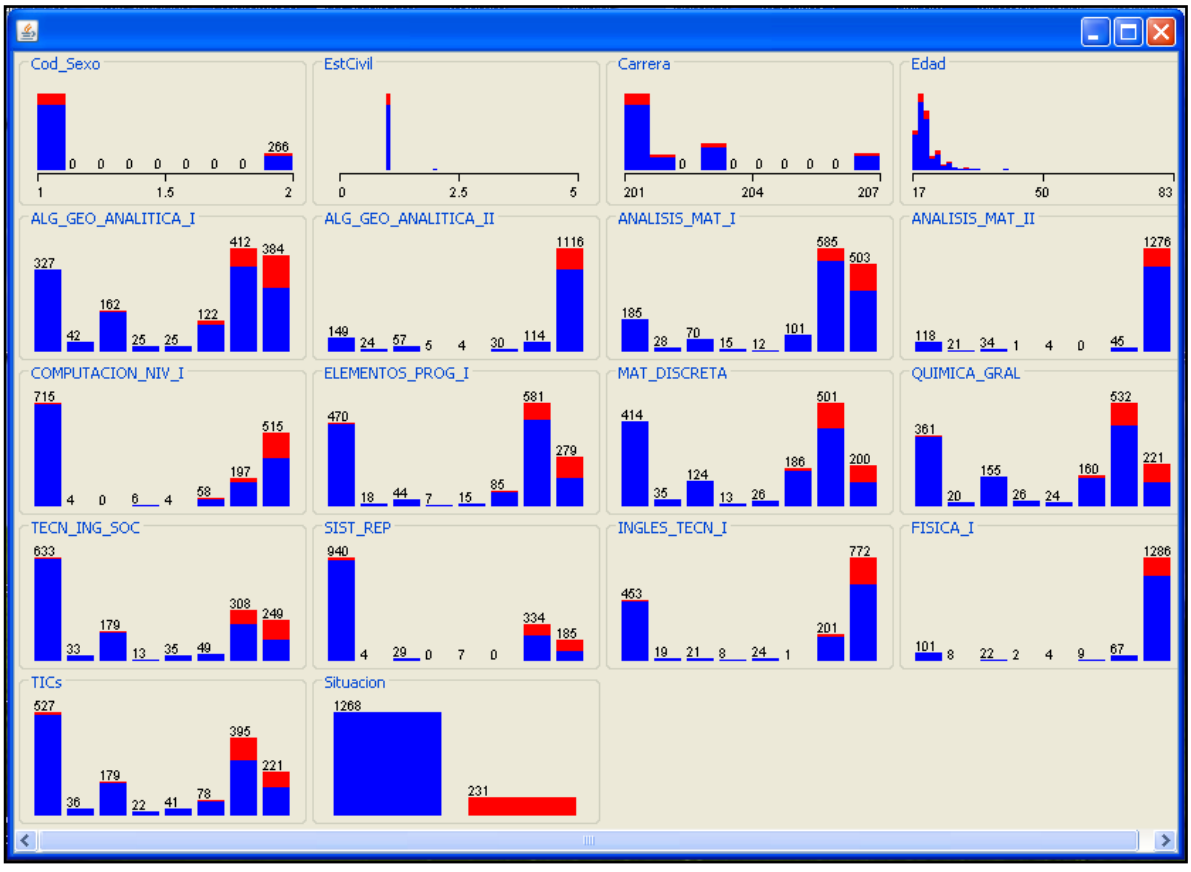



**Figura 13: Histogramas que nuestra la distribución de todos los atributos.**

El eje horizontal de las asignaturas va de mejor rendimiento a peor. Se observa que el gráfico confirma lo que es intuitivo, hay pocos desertores entre los alumnos que aprueban materias. Además, de haber desertores estos se dan en materias con alto nùmero de aprobados, lo que insinùa una dificultad relativa entre las mismas. Una vez cargado el lote con los datos recortado a la medida de lo sugerido por los algoritmos previso de selecciòn se puede pasar a la construcción de los modelos.

La cantidad de métodos de clasificación provistos por Weka es grande. Para nuestra experimentación fue necesario elegir el método detallado a utilizar. Siguiendo las pestañas ***Classify, a***parecerá una estructura de directorios en la que se seleccionará el algoritmo que se utilizará en el proceso de clasificación. En la Figura 14. Se ilustra los algoritmos particulares elegidos: Red neuronal, árbol de decisión y K vecinos más cercanos.

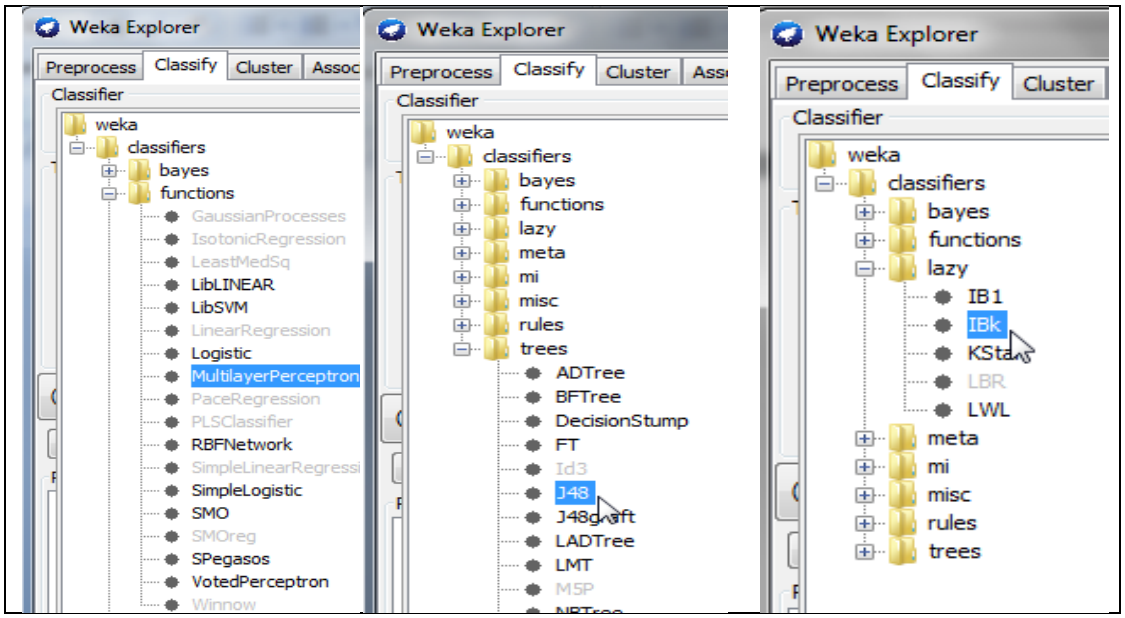



**Figura 14: Visualización de cómo se seleccionan los distintos atibutos.**

- **Instrumentos de Recolección y Medición de Datos**

Para evaluar la calidad de las distintas selecciones de atributos es necesario dividir el repositorio de datos de entrenamiento en dos conjuntos: entrenamiento y prueba, Weka provee para validación cruzada el método de n pliegues (***n-fold cross validation***) debido a su buen rendimiento computacional [Her04]. Este método consiste en dividir el conjunto de entrenamiento en n subconjuntos disjuntos de similar tamaño llamados carpetas (***folds***) de forma aleatoria. En Weka este es un número parametrizable, de subconjuntos a testear y se puede introducir el mismo a través del campo Folds. Una vez elegida la cantidad de carpetas, se realiza n iteraciones (igual al número de subconjuntos definido), donde en cada una ellas se reserva un subconjunto diferente para el conjunto de prueba y los contenidos de los restantes ***n***-1 se unen para construir el modelo (entrenamiento). En cada iteración se calcula el error de muestra parcial del modelo. Por último, se construye el modelo con todos los datos y se obtiene su error promediando los errores obtenidos anteriormente en cada una de las iteraciones. En este estudio se utilizó ***n***=10 particiones, que es el valor que comúnmente se usa y que se ha probado que da buenos resultados.

El desempeño de los algoritmos de aprendizaje automático es típicamente evaluado a través de una Matriz de Confusión (MC) como se ilustra en la Figura 15 ( la imagen de este ejemplo es para un problema de 2 clases). La teoría de esta matriz, a igual del significado de los resultados que arroja Weka, también fueron desarrollados en la etapa anterior. Solo para ilustrar esta etapa volveremos a explicar algunas conceptos. Las columnas de una MC son la Clase Real y las filas la Clase Predicha. Los componentes de una la matriz son los siguiente, los **VP** son el número de ejemplos positivos correctamente clasificados (Verdadero Positivo o True Positives), **FP** son la cantidad de ejemplos negativos incorrectamente clasificados como positivos (Falsos Positivos o False Positives), **VN** es el número de ejemplos negativos correctamente clasificados (Verdadero Negativo o True Negatives) y **FN** es el número de ejemplos incorrectamente clasificados como negativos (Falsos Negativos o False Negatives).

| Matriz de Confusión | | Clase Verdadera | |
|---|---|---|---|
| | | Positivos | Negativos |
| Clase Predicha | positivos | VP | FP |
| | negativos | FN | VN |
| Total columna | | P | N |

**Figura 15: Visualización de cómo se seleccionan los distintos atibutos.**

Como ya se mencionó Weka como resultado de la ejecución de un algoritmo nos proporciona numerosos datos estadísticos. Las medidas que tomamon para medir a los clasificadores en este trabajo son:

- **Exactitud:** Se calcula como el número de unidades clasificadas correctamente, sobre el número total de unidades consideradas. Se obtiene sumando los elementos de la diagonal divididos por el Total de observaciones. Este índice tiende a sobrestimar la bondad de la clasificación. Sus valores se encuentran en el intervalo [0, 1], siendo la clasificación mejor cuanto más se acerque a la unidad.

$$exactitud = \frac{VP + VN}{P + N} \quad \textbf{(2)}$$

- **Alcance:** Mide la probabilidad de que si un individuo pertenece a cierta categoría, el sistema lo asigne correctamente a dicha categoría. Hay una métrica para cada clase:

$$sensibilidad = \frac{VP}{P} \quad \textbf{(3)}$$

$$especificidad = \frac{VN}{N} \quad \textbf{(4)}$$

La sensibilidad de una prueba responde a las siguientes preguntas:

- ¿Cuántos resultados positivos se obtendrán en los individuos que abandonan?
- ¿Cuántos casos del total de casos en la población estudiada pueden identificarse por el resultado de la prueba?

Por lo tanto, podemos definir la sensibilidad de una prueba como la proporción de los individuos clasificados como positivos por el estándar de oro[4] que se identifican correctamente por la prueba en estudio. El valor que puede asumir la sensibilidad varía del 0 al 1 (100%), es decir, cuanto más alto es el valor, hay una mejor capacidad en la detección de aciertos por medio de la prueba. Existen áreas en las cuales una elevada sensibilidad es de especial importancia. Por ejemplo en caso de epidemias. Una baja sensibilidad deja a pacientes enfermos sin tratar como fuente de infección y propagaciòn en la comunidad. Lo que representa un costo alto. En nuestro caso no existe tal realimentación y hay una relación directa entre falta de sensibilidad y desertor no detectado oportunamente.

La especificidad de una prueba en estudio se refiere a la proporción de los individuos clasificados como negativos por el estándar de oro que se identifican correctamente por la prueba en estudio. Este parámetro responde a las siguientes preguntas:

- ¿Cuántos resultados negativos en alumnos que no abandonan?
- ¿Cuántos individuos que no abandonan sus estudios se confirmarán por el resultado de la prueba?

Al igual que la sensibilidad, el valor de la especificidad varía del 0 al 1 (100%), lo que significa que cuanto mayor sea el valor mayor capacidad de detección de sujetos sanos por la prueba. Por todo lo anterior, podemos decir, que una prueba ideal es aquella que tiene una alta ***sensibilidad*** (identifica correctamente a una alta proporción de individuos realmente expuestos) y una alta ***especificidad*** (da pocos resultados positivos en individuos no expuestos).

Por último, tenemos una combinación de ambas, ***F-measure*** que se usa para caracterizar con único valor la bondad de un clasificador o algoritmo. La formula de esta medida está establecida como:

$$F\ measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad \textbf{(5)}$$

Con Precisión nos referimos a la fracción de ejemplares que se han clasificado como de la clase correspondiente y que, en realidad, son de esa clase. Por tanto,

$$Precisión = \frac{VP}{VP + FP} \quad \textbf{(6)}$$

y luego tenemos el Recall (sensibilidad) que se refiere a la fraccióon de ejemplos de la clase de todo el conjunto que se clasifican correctamente (3).

Los modelos de clasificación efectúan evaluaciones sobre el objeto a clasificar y por comparación con un umbral deciden. Al variar ese umbral se obtienen más o menos individuos en una clase con el efecto contrario en la clase complementaria. Por cada valor de umbral se tiene una MC y a partir de ella un par de valores especificidad, sensibilidad. Esto da un punto en un diagrama, uniendo tales puntos se tiene una curva. Es práctica usual invertir el eje de especificidad colocando el 1.0 (valor ideal) a la izquierda (o llamar a las abcisas 1-especificidad). Se conoce esta curva como característica operativa del receptor (ROC, por sus siglas en inglés Receiver Operating Characteristic). La mayoría de los algoritmos de Minería de Datos

[4] *Gold standard (GS):* El rendimiento de todo test diagnóstico se basa en su comparación con un *gold standard (estándar de oro, patrón de oro, patrón de referencia).* El GS es la técnica diagnóstica que define la presencia de la condición con la máxima certeza conocida

pueden reportar sus resultados cuantitativamente, utilizando escalas contínuas. El análisis de curvas ROC constituye un método estadístico para determinar la calidad diagnóstica de estos tests, siendo utilizadas con tres propósitos específicos:

- Determinar el punto de corte (umbral) de una escala continua en el que se alcanza un compromiso entre la sensibilidad y especificidad que responda mejor a los objetivos del estudio. En el caso de la deserción si las autoridades persiguen una gran sensibilidad, o sea no perder un sòlo desertor, termianràn dedicando esfuerzos acadèmicos (en un sentido muy amplio que puede incluir motivaciòn y ayuda económica) a alumnos que sin ellos de todos modos no habrían desertado. Siendo los recursos totales acotados deberàn fijar un punto de corte aunque ello implique no atender a futuros desertores y perderlos.
- Evaluar la capacidad discriminativa del test diagnóstico, es decir en nuestro trabajo, su capacidad de diferenciar sujetos que no abandonan versus los que si lo hacen.
- Comparar la capacidad discriminativa de dos o más tests diagnósticos superponiendo sus curvas ROC, viendo a la luz del equilibrio elegido entre sensibilidad y especificidad cual da el mejor punto operativo.

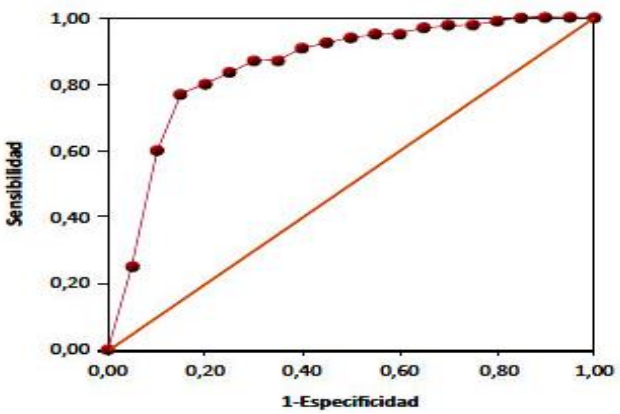



**Figura 16: Presentació Gráfica de una curva ROC.**

## Experimentación

Para cada algoritmo se utilizaron diversas configuraciones dentro de las opciones que ofrece Weka y según la bibliografía consultada. El objetivo de obtener diferentes métricas es para poder compararlas y elegir la que mejor resultado entregue. A continuación se detalla que configuración se usó en cada algoritmo.

- **Árboles de Decisión.**

En Weka encontramos los algoritmos de clasificación de árboles de decisión en la sección "***Trees***", Figura 14. En nuestra primer experimentación utilizaremos el algoritmo J48 que contiene Weka, que corresponde al algoritmo de árboles C4.5 explicado anteriormente.

En este algoritmo lo que se hace para cada nodo del árbol es elegir un atributo de los datos que divida el conjunto de muestras de forma más eficiente. Es decir que particione los datos en buenos subconjuntos de una clase u otra. El criterio que utiliza es el de ganancia de información o ***diferencia de entropía***[5],, que se utiliza para elegir el atributo que dividirá los datos. El atributo que posea la mayor ganancia de información normalizada se elige como parámetro de decisión. El algoritmo continúa de forma recursiva subdividiendo cada conjunto generado en el paso anterior. Cuando un conjunto es puro la recursiòn se detiene.

Para este algoritmo hay 3 pasos base para la recursión, éstos son:

- ✓ Todas las muestras en la lista pertenecen a la misma clase. Cuando esto sucede, simplemente crea un nodo de hoja para el árbol de decisión diciendo que elija esa clase.
- ✓ Ninguna de las características proporciona ninguna ganancia de información. En este caso, J48 crea un nodo de decisión más arriba del árbol utilizando el valor esperado de la clase.
- ✓ Instancia de la clase previamente no vista encontrada. Una vez más, J48 crea un nodo de decisión más arriba en el árbol con el valor esperado.

El parámetro más importante que deberemos tener en cuenta es el ***factor de confianza*** para la poda (***Confidence Level)***, en la Figura 17 se puede obervar todo los parámetros configurables para esta técnica. Este valor influye en el tamaño y capacidad de predicción del árbol construido. A probabilidad menor, se exigir· que la diferencia en los errores de predicción antes y después de podar sea más significativa para no podar.

El valor sugerido es del 0.25%. Según baje este valor, se permiten más operaciones de poda. Otra forma de variar el tamaño del árbol es a través del parámetro M que especifica el mínimo número de instancias o registros por nodo del árbol.

Con el fin de obtener diferentes modelos de árboles, se establecieron cinco (5) diferentes porcentajes del factor confianza C: 0.05%, 0.15%, 0.25%, 0.50% y 0.75%, manteniendo constante el factorM en 2, que es el valor que propone Weka.

[5] En el ámbito de la teoría de la información la entropía, también llamada entropía de la información y entropía de Shannon (en honor a Claude E. Shannon), mide la incertidumbre de una fuente de información.

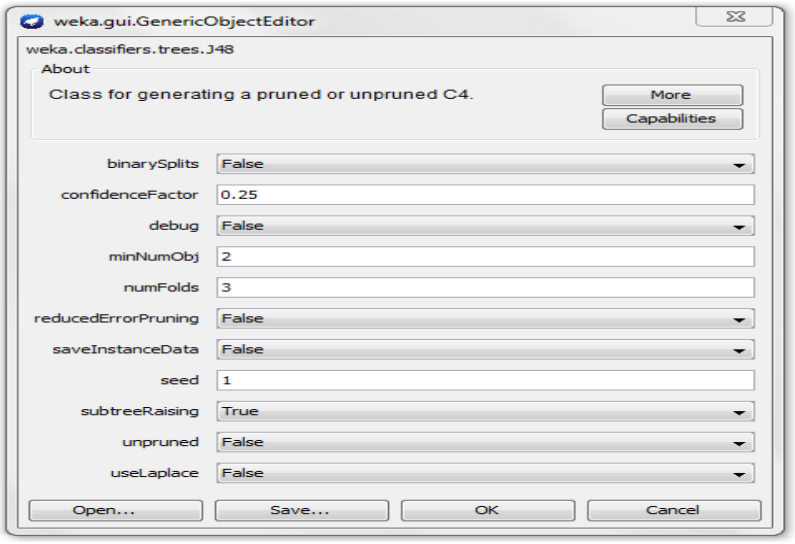



**Figura 17: Ventana para configurar los parámetros de J48.**

En la tabla siguiente se pueden ver los resultados obtenidos y en la Figura 18 se ven graficamente las distintas medidas

| | ***Factor de Confianza*** | ***factor M*** | ***% Instancias correctamente clasificadas*** |
|---|---|---|---|
| **Validación cruzada (10 folder)** | 0.05% | 2 | 90.927% |
| | 0.15% | 2 | 91.527% |
| | 0.25% | 2 | 91.461% |
| | 0.50% | 2 | 91.327% |
| | 0.75% | 2 | 89.993% |

**Tabla 4. Precisiones alcanzadas con J48 con distintos Factores de confiaza.**

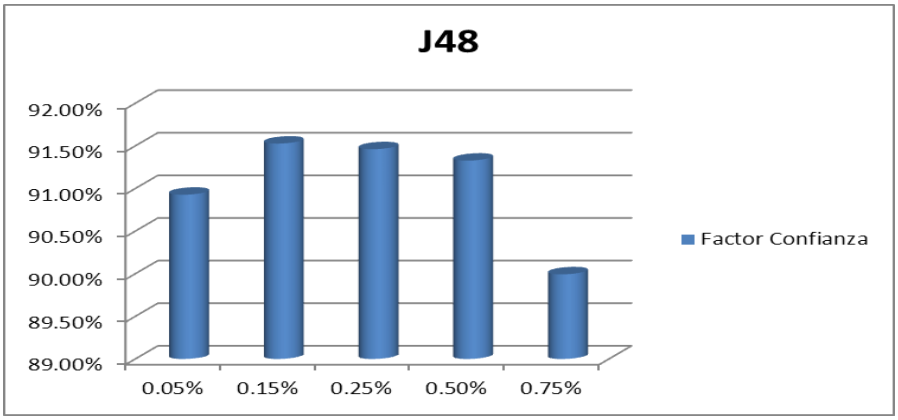



**Figura 18: Comparación de resultados del experimento realizado con J48.**

Estos ensayos muestran la baja influencia que tiene el factor de confianza y la conveniencia de aceptar el valor sugerido por Weka.

Cuando Weka termina de correr el clasificador entrega una salida, Figura 19, con los valores comentado en los apartados anteriores. Además algunos algoritmos como el J48, dejan visualizar el argoritmo, Figura 20.

Classifier output

```
Time taken to build model: 0.06 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1372               91.5277 %
Incorrectly Classified Instances       127                8.4723 %
Kappa statistic                          0.6562
Mean absolute error                      0.1345
Root mean squared error                  0.2691
Relative absolute error                 51.5385 %
Root relative squared error             74.5416 %
Total Number of Instances             1499

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.962     0.342      0.939     0.962      0.951      0.848    NO_ABANDONO
                 0.658     0.038      0.76      0.658      0.705      0.848    ABANDONO
Weighted Avg.    0.915     0.295      0.912     0.915      0.913      0.848

=== Confusion Matrix ===

    a    b   <-- classified as
 1220   48 |    a = NO_ABANDONO
   79  152 |    b = ABANDONO
```

**Figura 19: Configurar los parámetros de J48.**

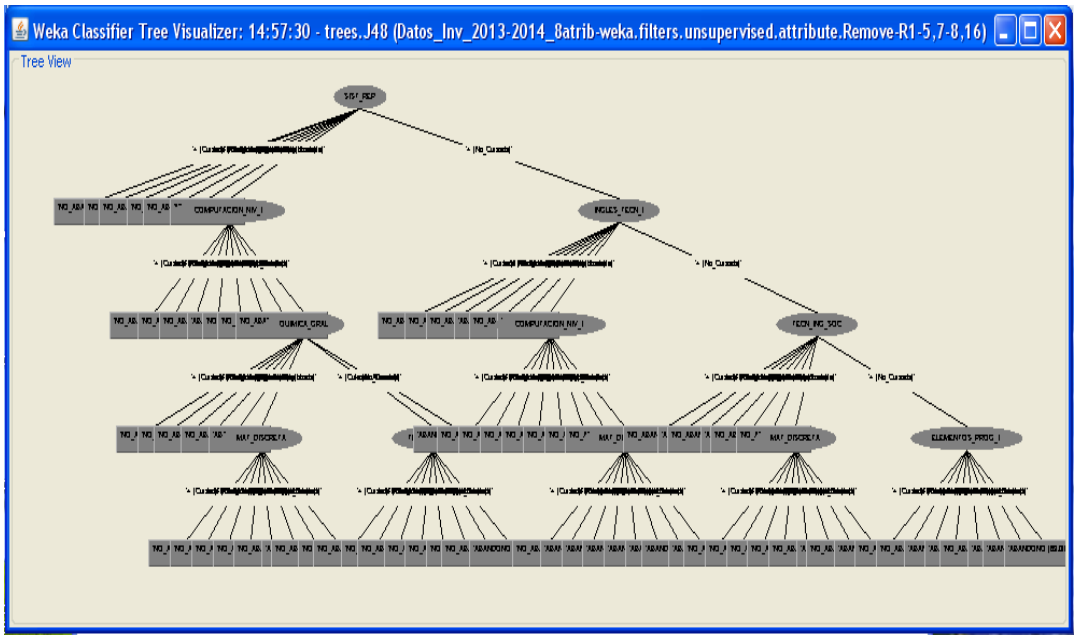



**Figura 20: Representación gráfica del árbol J48.**

- **Algoritmos K- Vecinos más próximo.**

El algoritmos K-Vecinos más próximo se encuentran en Weka en los llamados ***Lazy Classifiers*** (Clasificadores perezosos), en la Figura 14. Este nombre se debe a que realmente no se aprende nada de las muestras de aprendizaje, sino que todo el trabajo se hace al momento de clasificar un nuevo patrón.

En Weka para el caso del algoritmo ***Vecino más próximo*** el mismo se llama ***IB1*** y para ***K Vecinos más próximo, IBk***. Ambos algoritmos utilizan como medida de distancia por defecto la distancia ***Euclídea***, aunque esta opción se puede modificar. Se puede usar para el caso de ***IBk*** pesos en las medidas de distancia penalizando los patrones que se encuentran más lejos de la muestra a clasificar. Para este algoritmo utilizamos varios valores de ***k*** y dos métodos de distancia distintos.

Para la estimación de la distancia se utilizan dos estrategias conocidas como distancia euclidiana y distancia Mahalanobis, las cuales se presentan en su forma vectorial para una distancia entre dos vectores en (6) y (7), respectivamente.

$$D_{Euclidea}(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T(\vec{x}-\vec{y})} \quad (7)$$

$$D_{Mahalanobis}(x,\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T \sum\nolimits_{-1} (\vec{x}-\vec{y})} \quad (8)$$

En la figura se puede ver la venta de configuración del método.

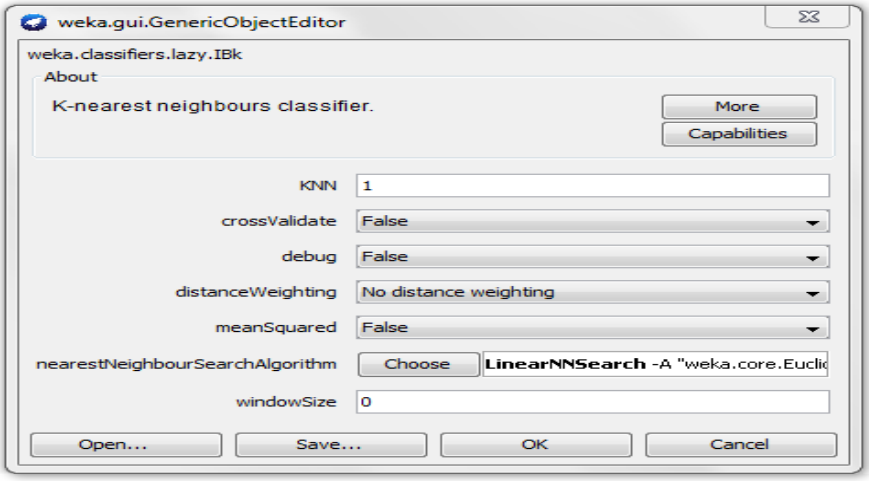



**Figura 21: Ventana para configurar los parámetros de J48.**

En la Tabla 5 se pueden observar los valores devuelto por Weka usando distintos valores de K y dos métodos de distancias distintos. Como se puede observar este último criterio devuelve los mismos porcentaje de aciertos. En la Figura 22 se presenta la representación gráfica de estos resultados.

| | Método de distincia | Valor de $K$ | % Instancias correctamente clasificadas |
|---|---|---|---|
| **Validacion cruzada** | Euclidea | 1 | 90.1268 % |
| | Euclidea | 5 | 90.5937 % |
| | Euclidea | 10 | 90.3936 % |
| | Euclidea | 15 | 90.3269 % |
| | Euclidea | 20 | 90.3269 % |
| | Mahalanobis | 1 | 90.1268 % |
| | Mahalanobis | 5 | 90.5937 % |
| | Mahalanobis | 10 | 90.3936 % |
| | Mahalanobis | 15 | 90.3269 % |
| | Mahalanobis | 20 | 90.3269 % |

**Tabla 5. Precisiones alcanzadas con J48 con distintos Factores de confianza.**


**Figura 22: Representación gráfica con K-Vecino.**

En la siguiente imagen se observan los valores que Weka ofrece como resultado de la ejecución del algoritmo.

Classifier output

```
Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1358               90.5937 %
Incorrectly Classified Instances       141                9.4063 %
Kappa statistic                          0.6509
Mean absolute error                      0.13
Root mean squared error                  0.2592
Relative absolute error                 49.7855 %
Root relative squared error             71.7836 %
Total Number of Instances             1499

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.937     0.264      0.951     0.937     0.944      0.947    NO_ABANDONO
                 0.736     0.063      0.68      0.736     0.707      0.947    ABANDONO
Weighted Avg.    0.906     0.233      0.909     0.906     0.907      0.947

=== Confusion Matrix ===

    a    b   <-- classified as
 1188   80 |    a = NO_ABANDONO
   61  170 |    b = ABANDONO
```

**Figura 23: Resultados con mejor porcentaje con K-Vecino.**

- **Redes Neuronales Artificiales.**

Si bien se ha demostrado que la propiedad de ***aproximador universal de funciones*** de la red MLP requiere de un máximo de dos capas ocultas, en la mayoria de los casos una única capa oculta resulta suficiente para conseguir buenos resultados.

**Tamaño de las Redes Neuronales**

Un Perceptrón Multicapa es una red con alimentación hacia delante, compuesta de varias capas de neuronas entre la entrada y la salida de la misma, esta red permite establecer regiones de decisión mucho más complejas que las de dos semiplanos, como lo hace el Perceptrón de un solo nivel. En este tipo de red puede haber una o más capas ocultas entre las capas de entrada y salida. El tamaño de las redes depende del número de capas y del número de neuronas ocultas por capa. El número de unidades ocultas está directamente relacionado con las capacidades de la red. Para que el comportamiento de la red sea correcto, se tiene que determinar apropiadamente el número de neuronas de la capa oculta determinación de la arquitectura optima de la red.

Si bien se ha demostrado que la propiedad de ***aproximador universal de funciones*** de la red MLP requiere de un máximo de dos capas ocultas, en la mayoria de los casos una única capa oculta resulta suficiente para conseguir óptimos resultados. [Flo08]

Para determinar el número de neuronas ocultas de cada capa suele utilizarse reglas "***ad hoc***" que, aunque no resulten matemáticamente justificables, han demostrado un buen comportamiento en diversas aplicaciones prácticas. Entre las que se encuentran:

- ***La regla de la pirámide geométrica***: se basa en la suposición de que la capa oculta ha de ser inferior al total de variables de entrada, pero superior al número de variables de salida. La fórmula es:

$$\sqrt{NxM} \qquad (9)$$

Siendo ***N*** el número de variables de entrada y ***M*** el total de neuronas de salida.

- ***La regla de la capa oculta-capa entrada:*** según esta regla el número de capas ocultas está relacionado con el número de neuronas de entrada. En particular suele aplicarse la regla **2 x 1**, de forma que el número de neuronas ocultas no puede ser superior al doble del número de variables de entrada.

En Weka se pueden configurar diferentes parámetros, Figura 24, entre los que se encuentran:

- ***Número de capas ocultas*** (hidden layers): se ofrecen varias posibilidades para aproximar correctamente el modelo buscado.
- ***Tasa de aprendizaje***: factor de actualización de pesos cuando han de ser modificados, un valor mayor converge más rápido pero puede provocar oscilaciones.
- ***Umbral de validación***: para determinar cuando dar por acabado el test de validación, cuántas veces en una fila el error puede empeorar antes de terminar el entrenamiento. En nuestro modelo se especificó que el número de capas ocultas sería '***a***' ((atributos+clases)/2), con una tasa de aprendizaje de 0.3 y un umbral de validación de 20.

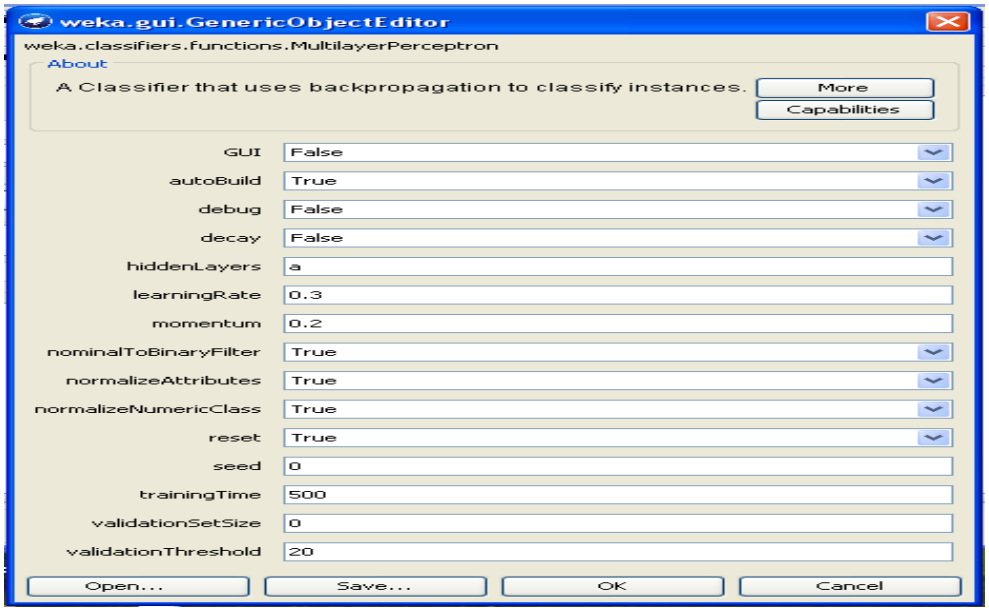



**Figura 24: Ventana para configurar los parámetros de una red MLP.**

En este trabajo solo se experimentó con el número de neuronas ocultas de cada capa. En la tabla suele utilizarse reglas "***ad hoc***" que, aunque no resulten

| | **Configuración** | **% Instancias correctamente clasificadas** |
|---|---|---|
| **Validacion cruzada** | Sin capas ocultas | **89.993%** |
| | Con 1 capa oculta con 3 neuronas | **90.583%** |
| | Con 1 capa oculta con 5 neuronas | **90.593%** |
| | Con 2 capas ocultas. Una con 5 y otra con 3 neuronas | **90.460%** |

**Tabla 6. Precisiones alcanzadas con el algoritmo con distintos configuraciones de capas.**

Esta limitada experimentación confirma que el pasar a dos capas no introduce una mejora en los resultados.

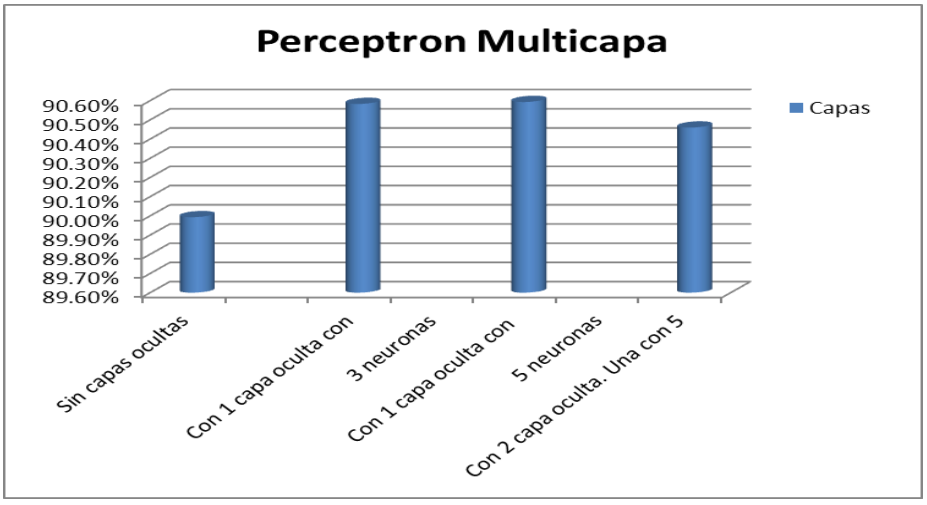
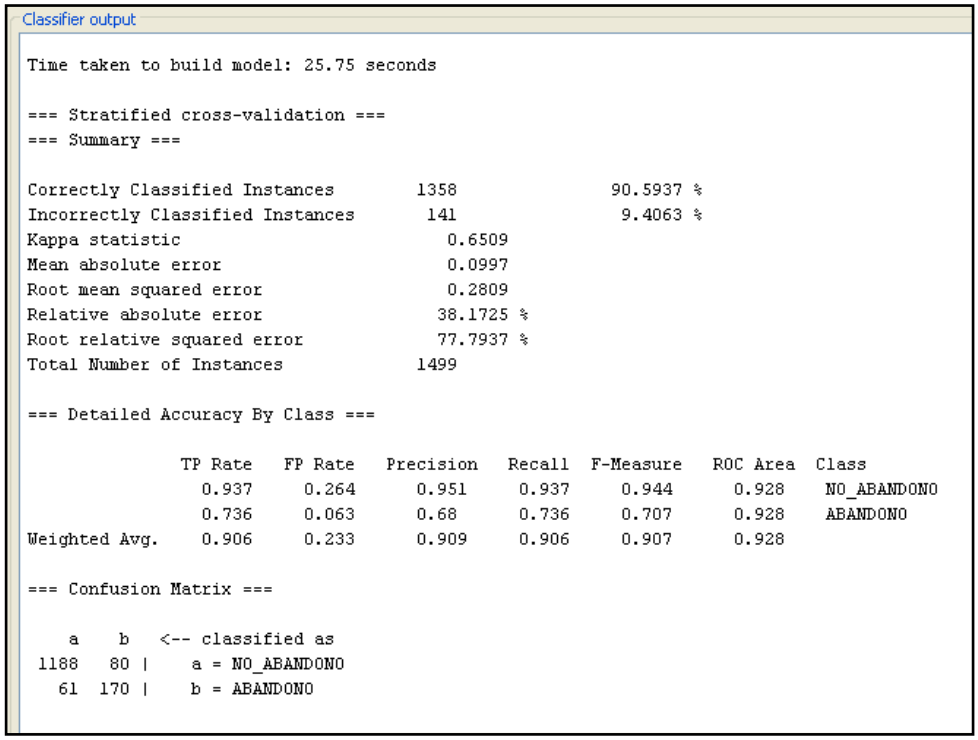



**Figura 25: Representación gráfica con distintas configuraciones de capas.**

Además al ejecutar el algoritmo la ventana del clasificador nos proporciona la siguiente información relacionada con el modelo obtenido.



Classifier output

```
Time taken to build model: 25.75 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1358               90.5937 %
Incorrectly Classified Instances       141                9.4063 %
Kappa statistic                          0.6509
Mean absolute error                      0.0997
Root mean squared error                  0.2809
Relative absolute error                 38.1725 %
Root relative squared error             77.7937 %
Total Number of Instances             1499

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.937     0.264      0.951     0.937     0.944      0.928    NO_ABANDONO
                 0.736     0.063      0.68      0.736     0.707      0.928    ABANDONO
Weighted Avg.    0.906     0.233      0.909     0.906     0.907      0.928

=== Confusion Matrix ===

    a    b   <-- classified as
 1188   80 |    a = NO_ABANDONO
   61  170 |    b = ABANDONO
```

**Figura 26: Salida del Clasificador MLP.**

Este algoritmo puede presentar una representación gráfica de una RNA. En la siguiente Figura se observa el modelo del primer experimento con RNA, con 5 neuronas en la capa oculta.

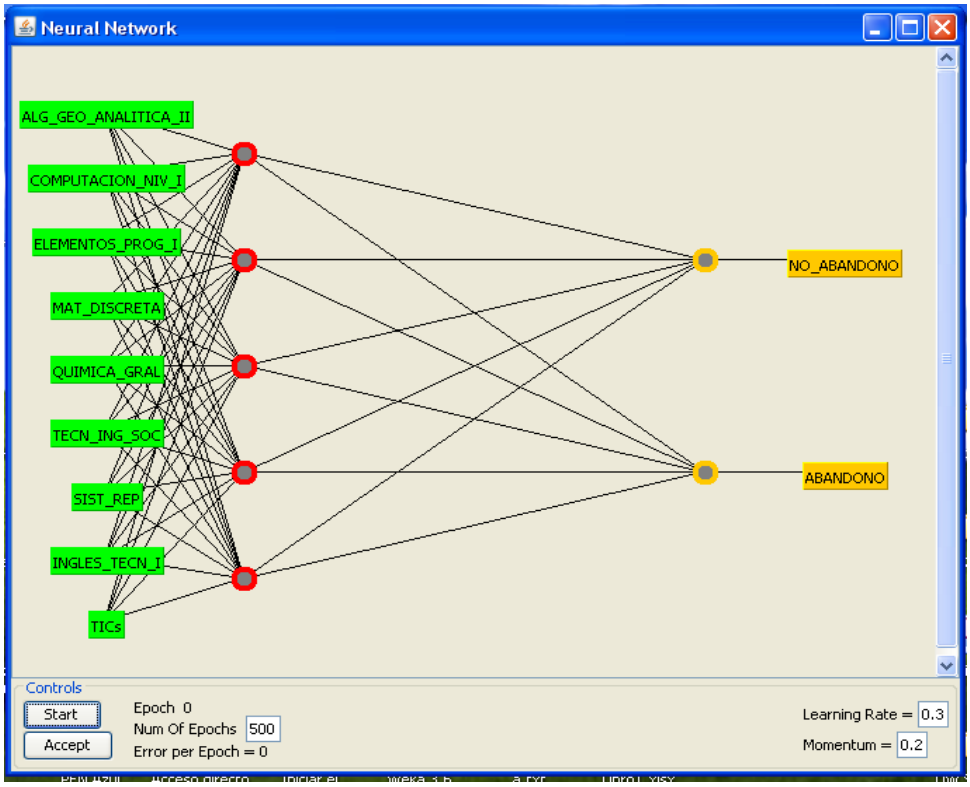



**Figura 27: Salida gráfica de una RNA con 5 neuronas en la capa oculta.**

- **K-medias**

Este es un algoritmo que pertenece al grupo de los denominados de ***Aprendizaje No Supervisados*** o de ***Agrupamiento*** (en inglés *Clustering*). Esta técnica representa la división de datos en grupos de objetos similares llamados clusters. De esta manera se busca maximizar la similitud de las instancias en cada cluster y minimizar la similitud entre clusters.

K-medias o K-means es uno de los algoritmos más utilizados para realizar , técnica de agrupamiento en Minería de Datos [Her04]. La idea del k-medias es colocar todos los objetos en un espacio determinado y dadas sus características formar grupos de objetos con rasgos similares pero diferentes a los demás que integran otros grupos. Sin embargo el algoritmo presenta algunos inconvenientes:

• El agrupamiento final depende de los centroides iniciales.
• La convergencia en el óptimo global no está garantizada, y para problemas con muchos ejemplares, requiere de un gran número de iteraciones para converger.

Algunas implementaciones de K-means sólo permiten valores numéricos para atributos. En ese caso, puede ser necesario convertir el conjunto de datos en el formato de hoja de cálculo estándar y convertir los atributos categóricos en binarios. También puede ser necesario normalizar los valores de los atributos que se miden en escalas sustancialmente diferentes (por ejemplo, "edad"). Esto se debe a que el algoritmo que viene con Weka, ***SimpleKMeans*** maneja automáticamente una mezcla de atributos categóricos y numéricos. Además, el algoritmo normaliza automáticamente los atributos numéricos cuando se hacen cálculos de distancia. El algoritmo ***SimpleKMeans*** utiliza medida de ***distancia Euclidiana,*** (7) para calcular distancias entre instancias y clusters.

**Propiedades:**

La distancia o métrica euclideana (en su caso general de ***N*** dimensiones) satisface las condiciones necesarias para ser catalogada como una métrica en términos estrictamente matemáticos que serían:

1. $d$ (A,B) mayor o igual a 0
2. $d$ (A,B) = $d$ (B,A). Llamada propiedad simétrica
3. $d$ (A,A) = 0.
4. Si $d$(A,B) ≤ $d$(A,C) + $d$(C,B), Llamada propiedad de la desigualdad triangular.
5. $d$(A,B)= 0, entonces A=B

Para utilizar este algoritmo se debe seleccionar la pestaña "***Clúster***" en el Explorador y hacer clic en el botón ***"Choose"***. De unaa lista desplegable de algoritmos de agrupación disponibles, se selecciona "***SimpleKMeans***". A continuación, haga clic en el cuadro de texto a la derecha del botón "***Elegir***" para obtener la ventana emergente mostrada en la Figura 28, para editar el parámetro de agrupación.

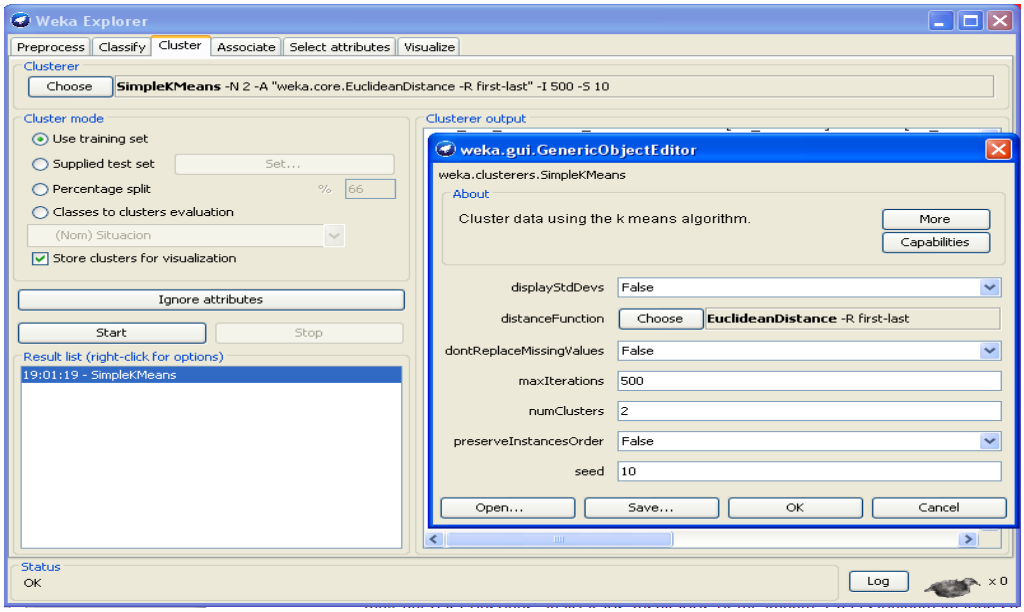



**Figura 28: Ventana para configurar los parámetros de SimpleKMeans.**

En la ventana emergente para configurar parámetros, se deja el valor que sugiere el software, un dos (2) como el número de clústeres (***NumClusters***) y dejamos también sin modificar el valor de semilla (***seed***). El valor de la semilla se utiliza para generar un número aleatorio que, a su vez, se utiliza para realizar la asignación inicial de las instancias a los clústeres. En general, K-means es bastante sensible a cómo se asignan inicialmente los clusters. Por lo tanto, a menudo es necesario probar diferentes valores y evaluar los resultados. Para realizar la agrupación, en la pestaña principal de***Explorer*** ingresamos a la opción ***Cluster***, luego se selecciona en la opción "***Cluster Mode",*** para elegir el modo de evaluar los resultados del agrupamiento. Lo más simple es utilizar el propio conjunto de entrenamiento. En este trabajo se empleó la opción ***Use tranining set.*** Existe también la opción de comparar los clusters con un atributo de clasificación (***Classes to clusters evaluation***), que es la opción utilizada en este trabajo, dicho atributo no se considera en la construcción de los clusters. Finalmente, el cuadro opcional de almacenamiento de instancias, ***Store clusters for visualization***, es muy útil para después analizar los resultados gráficamente. En la siguiente imagen se puede observar la salida que entrega Weka luego de correr el algoritmo.

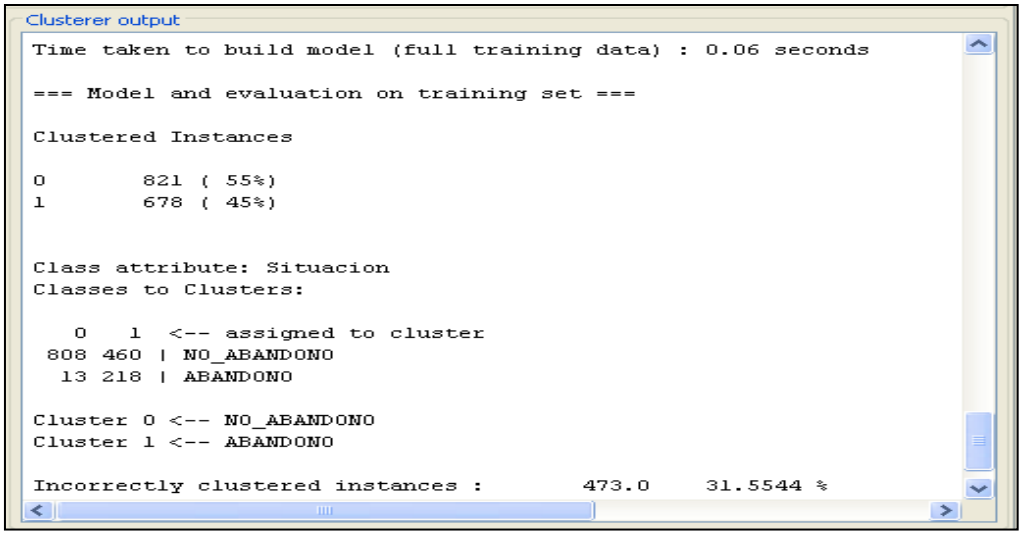
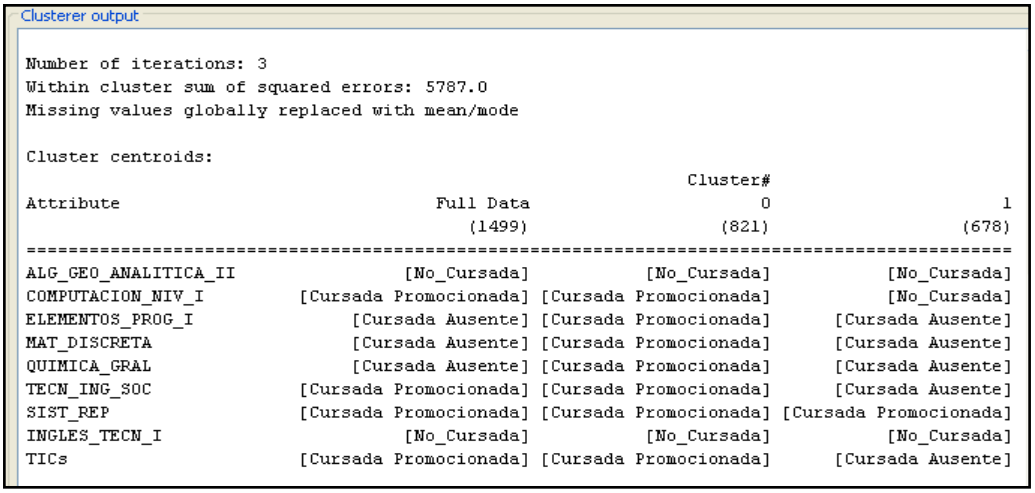



**Figura 29: Salida para simple K-Means.**

Como se puede observar en la Figura anterior, Weka entrega distinta información como resultado de la ejecución del algoritmo, si se compara esta con los resultados que informa el software ante los métodos supervisados. La ventana de resultados de la Figura 30, muestra el centroide de cada grupo. Así como estadísticas sobre el número y el porcentaje de Instancias asignadas a diferentes clusters. Los centroides de agrupamiento son los vectores medios para cada agrupación (de modo que cada valor de dimensión en el centroide representa el valor medio para esa dimensión en el grupo). Por lo tanto, los centroides se pueden utilizar para caracterizar los conglomerados.



Clusterer output

```
Number of iterations: 3
Within cluster sum of squared errors: 5787.0
Missing values globally replaced with mean/mode

Cluster centroids:
                                                        Cluster#
Attribute                               Full Data             0                    1
                                          (1499)          (821)                (678)
=============================================================================================
ALG_GEO_ANALITICA_II                [No_Cursada]    [No_Cursada]         [No_Cursada]
COMPUTACION_NIV_I         [Cursada Promocionada] [Cursada Promocionada]    [No_Cursada]
ELEMENTOS_PROG_I               [Cursada Ausente] [Cursada Promocionada]  [Cursada Ausente]
MAT_DISCRETA                   [Cursada Ausente] [Cursada Promocionada]  [Cursada Ausente]
QUIMICA_GRAL                   [Cursada Ausente] [Cursada Promocionada]  [Cursada Ausente]
TECN_ING_SOC              [Cursada Promocionada] [Cursada Promocionada]  [Cursada Ausente]
SIST_REP                  [Cursada Promocionada] [Cursada Promocionada] [Cursada Promocionada]
INGLES_TECN_I                      [No_Cursada]    [No_Cursada]         [No_Cursada]
TICs                      [Cursada Promocionada] [Cursada Promocionada]  [Cursada Ausente]
```

**Figura 30: Salida para K-Means.**

Un factor que afecta en gran medida el costo computacional del algoritmo K-means es el número de iteraciones que necesita realizar, ya que por cada iteración calcula la distancia de cada objeto a los centroides de los grupos.

Otra forma de entender las características de cada clúster a través de la visualización. Para ello, haga clic con el botón derecho del ratón en el conjunto de resultados del panel "***Lista de resultados***" de la izquierda y seleccione "***Visualizar las asignaciones de clúste***r". Esto muestra la ventana de visualización como se muestra en la Figura 31.

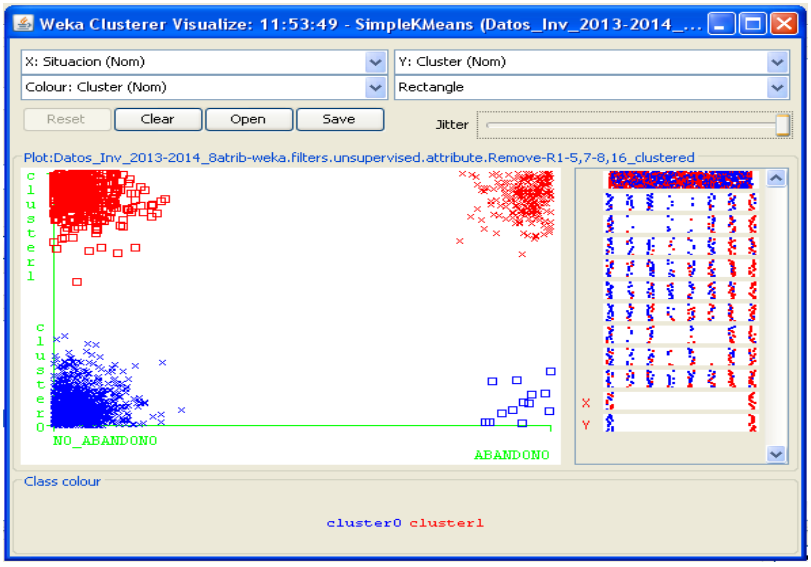



**Figura 31: Grafica sobre la distribución de Cluster.**

**Creación de modelos.**

Luego de realizado el entrenamiento de cada algoritmo con distintas configuraciones, se realizó la creación de los 4 modelos, uno para cada algoritmo. En la Figura 27, se muestra como en Weka se puede guardar un modelo.

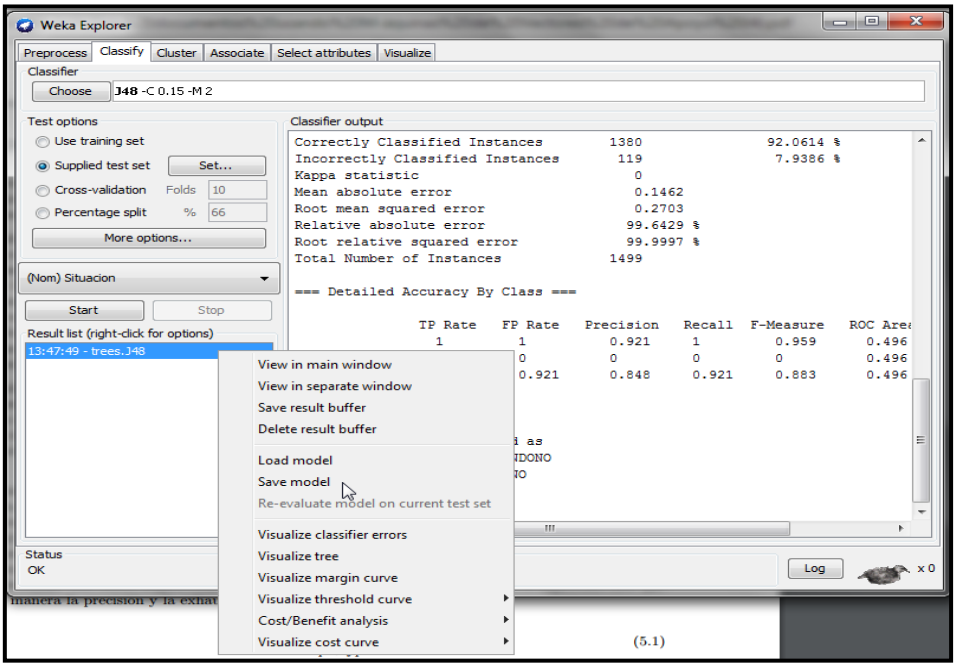



**Figura 32: Ventana para guardar los algoritmos como modelos.**

En la parte inferior izquierda (***Result List***) aparece una lista de todos los experimentos realizados, al dar clic derecho se despliega una lista de opciones que nos permite visualizar de manera diferente los resultados obtenidos incluyendo la ejecución de gráficas en aquellos algoritmos que lo permitan. A continuación se realiza una breve explicación de cada una de las opciones que aparecen en el menú anteriormente mencionado

- ***View in main window***: Muestra el resultado del experimento en la pantalla principal del clasificador.
- ***View in separate window***: Permite visualizar el resultado en una nueva ventana.
- ***Save result buffer***: Guarda el resultado del experimento en un fichero.
- ***Load model***: Carga un modelo de clasificador ya construido.
- ***Save model***: Guarda el modelo de clasificación actual.
- ***Re-evaluate model on current test set***: Enfrenta otro modelo con un conjunto de datos actual.
- ***Visualize clasiffier errors***: Se abre una nueva ventana donde se muestra una gráfica con los errores de clasificación.
- ***Visualize tree***: Esta opción mostrará un árbol de decisión generado por el clasificador. Se mostrará una ventana como la siguiente según el clasificador que se haya escogido.
- ***Visualize margin curve***: Muestra por medio de una curva la diferencia entre la probabilidad de la clase estimada y la máxima probabilidad de otras clases.

Se crearon cuatro modelos, mediante la opcion ***Save model,*** con la mejor configuración obtenida anteriormente. Luego a traves de la opción ***Supplierd test set,*** se cargo el lote de testeo, el mismo esta compuesto por los datos de los alumnos de la cohorte 2015, que cuenta con 793 alumnos.

En las siguientes imágenes se muestra como se cargan en Weka los datos para probar lo modelos creados.

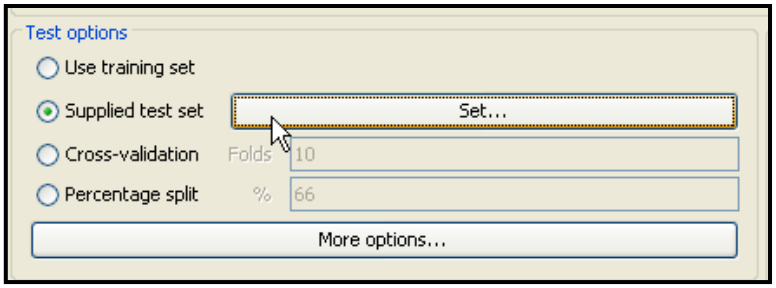



**Figura 33: Ventana para cargar el conjunto de datos para testeo.**

Una cargado el lote de prueba y el modelo mediante la opcion Reevalúe el modelo en el juego de prueba actual (***Re-evaluate model on current test set***), esta opcion se muesta en la siguiente imagen..

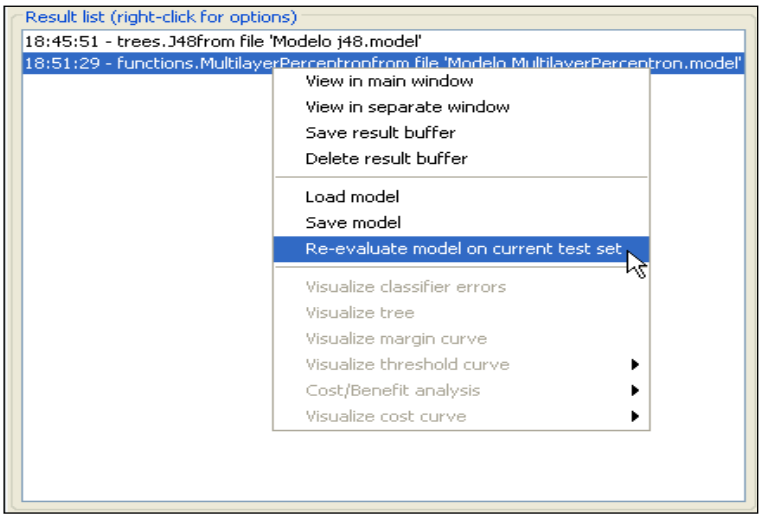



**Figura 34: Ventana para re-valuar un modelo.**

En la siguiente tabla se muestra la clasificación de los alumnos de la cohorte 2015 obtenida por la ejecución de los modelos previamente construidos, para los tres algoritmos supervisados.

| | RNA Con 1 capa oculta con 5 neuronas | J48 | K-Vecinos más próximo |
|---|---|---|---|
| VP \| FP<br>FN \| VN | 268 \| 201<br>167 \| 157 | 397 \| 72<br>257 \| 67 | 216 \| 253<br>108 \| 216 |
| **Instancias clasificadas correctamente** | ***53.593 %*** | ***58.512 %*** | ***54.476 %*** |
| **NO_ABANDONO** | | | |
| TP Rate | 0.571 | 0.846 | 0.461 |
| FP Rate | 0.515 | 0.793 | 0.333 |
| Precision | 0.616 | 0.607 | 0.667 |
| Recall | 0.571 | 0.846 | 0.461 |
| F-Measure | 0.593 | 0.707 | 0.545 |
| ROC Area | 0.554 | 0.488 | 0.595 |
| **ABANDONO** | | | |
| TP Rate | 0.485 | 0.207 | 0.667 |
| FP Rate | 0.429 | 0.154 | 0.539 |
| Precision | 0.439 | 0.482 | 0.461 |
| Recall | 0.485 | 0.207 | 0.667 |
| F-Measure | 0.460 | 0.289 | 0.545 |
| ROC Area | 0.554 | 0.488 | 0.595 |
| **Promedio ponderado** | | | |
| **TP Rate** | **0.536** | **0.585** | **0.545** |
| **FP Rate** | **0.480** | **0.532** | **0.418** |
| **Precision** | **0.544** | **0.556** | **0.582** |
| **Recall** | **0.536** | **0.585** | **0.545** |
| **F-Measure** | **0.539** | **0.536** | **0.545** |
| **ROC Area** | **0.554** | **0.488** | **0.595** |
| | | | |
| | | | |

**Tabla 7. Resultados de la ejecución de los algoritmos (1499 instancias)**

## Estado de avance del proyecto

A continuación se describe con qué nivel fueron alcanzados los objetivos propuestos.

Los Objetivos Principales que se propusieron fueron dos:

- *Estudiar y analizar distintos Algoritmos de Data Mining.*
- *Crear un Modelo de Datos o Vista Minable. Aplicarlo al problema elegido como testeo y, sobre éste, aplicar técnicas de Minería de datos y comparar los resultados que se obtengan.*

Se estudió el estado del arte en diferentes técnicas para desarrollar modelos de predicción, se eligieron 3 técnicas de aprendizaje supervisados: Árboles de Decisión, Redes Neuronales y K-Vecinos más próximo y una de aprendizaje No supervisado o de agrupamiento: K-media o K-meams.

Como la MD es una etapa más, dentro de un proceso superior conocido como metodología KDD, en esta memoria se demostró el desarrollo de dicha metodología de minería de datos, y fueron descriptas con detalle cada una de las distintas etapas que la componen.

Del estudio de las diversas técnicas se ha podido ver con un caso real y mediante la aplicación de diferentes parámetros, el comportamiento de los mismos. Se ha creado un modelo de cada algoritmo y se los ha probado con datos de la cohorte 2015.

Respecto a los Objetivos secundarios plateados oportunamente tenemos:

- *Llevar a cabo un estudio descriptivo/comparativo de las características particulares de los Algoritmos comúnmente utilizados en Minería de Datos.*
- *Conocer distintos programas informáticos de DM para ser utilizado para el aprendizaje automático de la información.*
- *Enfrentar los resultados computacionales de la minería de datos con estudios extracomputacionales para mejor valoración de los resultados obtenidos*.

El primer punto se desarrollo para los 4 algoritmos, dejando planteado tambien, cuales son los principales paramétros que se pueden configurar en cada caso.

El segundo objetivo secundario, se cumplió al investigar uno de los principales software en minería de datos, Weka (Waikato Environment for Knowledge Analysis, en español «entorno para análisis del conocimiento de la Universidad de Waikato»), que es una plataforma de software para el aprendizaje automático y la minería de datos escrito en Java y desarrollado en la Universidad de Waikato. Weka es software libre distribuido bajo la licencia GNU-GPL.

Por último en la tabla 7, se mostró el resultado arrojado por Weka luego correr cada uno de los modelos construidos a lo largo del trabajo.

En relación con las dos Hipótesis planteadas:

- "*Los diferentes Modelos de Clasificación de Minería de Datos difieren significativamente en sus predicciones".*
- *"El comportamiento de los alumnos actuales respecto de los aspectos analizados en este proyecto es análogo al de los que aparecen en los datos históricos*".

La primera hipótesis se comprueba en parte, ya que tenemos que entre algoritmos del mismo tipo de aprendizaje, es poca la diferencia entre los valores que arroja cada medida. No sucede lo mismo con el resultado que se nos presentó con la comparación con un tipo de aprendizaje No supervisado.

La segunda hipótesis se comprueba al analizar los datos de las cohortes, 2011 al 2015. En ellas se pueden observar los distintos porcentajes.

| Año | No Abandonan | Porcentaje | Abandonan | Porcentaje | Totales |
|---|---|---|---|---|---|
| **2011** | 692 | 78.64% | 188 | 21.34% | 880 |
| **2012** | 563 | 81.95% | 124 | 18.05% | 687 |
| **2013** | 599 | 83.43% | 119 | 16.57% | 718 |
| **2014** | 669 | 85.66% | 112 | 14.34% | 781 |
| **2015** | 469 | 59.14% | 324 | 40.86% | 793 |

**Tabla 8. Resultados de la ejecución de los algoritmos (1499 instancias**)

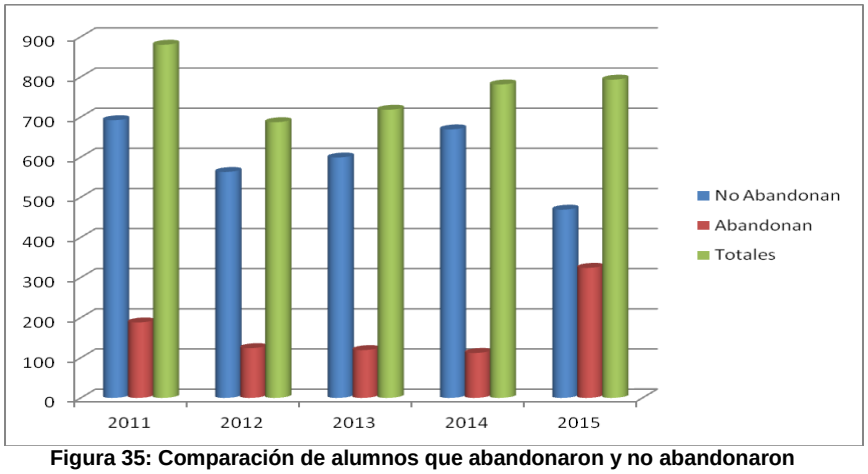



**Figura 35: Comparación de alumnos que abandonaron y no abandonaron entre diferentes ciclos lectivos.**

## Conclusiones

Las principales conclusiones obtenidas se detallan a continuación. Además se presentan algunas propuestas tendientes a mejorar futuros desarrollos de proyectos de minería de datos educativa.

- Según muestra el histagrama de la figura 13, el eje horizontal de las asignaturas va de mejor rendimiento a peor. Se observa que el gráfico confirma lo que es intuitivo, hay pocos desertores entre los alumnos que aprueban materias. Además, de haber desertores estos se dan en materias con alto número de desaprobados, lo que insinúa una dificultad relativa entre las mismas.
- Viendo la tabla 3, se destaca que el uso de la totalidad de los atributos no significó ninguna mejora en la calidad de la clasificación, lo que muestra la conveniencia de recurrir a una selección de atributos por el ahorro de tiempo que habrá en la clasificación, sin pérdida de calidad en la misma.
- El empleo de las 3 técnicas de clasificación supervisada en la realización de pruebas a estudiantes permite obtener una clasificación con aceptables porcentajes de verdaderos positivos.
- A través de esta comparación, tanto en el modo testeo como en el de evaluación, se puede concluir que las técnicas de minería de datos poseen importantes ventajas, para poder descubrir patrones de comportamiento de un estudiante que deserta una carrera, ya que brinda un alto valor agregado para el análisis y la generación del nuevo conocimiento.
- Se investigó la influencia del factor de confianza que parametriza el método J48 de árboles de decisión, concluyendo su baja incidencia y lo acertado del valor que Weka sugiere usar.
- Referente a las técnicas empleadas para la generación de los modelos, se pudo constatar que la técnica de agrupamiento (clustering) empleada, brindó mas bajo porcentaje de instancias bien clasificadas: 69%, en relación com los restantes algoritmos supervisados.
- Además se encontró que los árboles de decisión tuvieron mejor porcentaje de aciertos cuando se tienen datos que no participan en el entrenamiento.
- Según los resultados obtenidos en las cuatro carreras analizadas, se pudo constatar además que las variables: sexo y carrera; no influyen significativamente para que un estudiante deserte la carrera. En cambio Estado civil y Edad, si bien no fueron de la partida para evaluar los algoritmos, si fueron elegidos por 2 de los 4 sistemas de selección de atributos.

- En cuanto al tiempo de ejecución es considerablemente mejor en la ejecución de los algoritmos KNN y AD, dado que en todas las ejecuciones el resultado de la pruena es obtenido en pocas fracciones de seguntos, en cambio es considerablemente mayor en las RNA.

**Recomendaciones**

- En esta investigación se ha descubierto que en general las instituciones no recolectan la suficiente información referente a la caracterización del estudiante al momento de ingresar a estudiar, que permita establecer modelos de predicción de retención.
- Este trabajo permitió apreciar la importancia que tiene el proceso de recopilación de datos, abarcando las fases de análisis y preparación de los datos, descripto en el apartado proceso de extracción de conocimientos asociado a la metodología KDD.
- Como trabajo futuro se propone recoger un gran conjunto de datos reales incorporando nuevas variables a la base de datos socioeconómicos de estudiantes universitarios y aplicar los modelos a estos datos. Además, de aplicar otros métodos de clasificación para poner a prueba el método más adecuado que se adapte a la estructura de los datos de los estudiantes y dar una mejor precisión en la clasificación.
- El conjunto de datos existente para el estudio y los atributos contenidos en ellos son fundamentales para poder lograr los niveles de predicción necesarios para la validación positiva de los modelos, por lo que se propone, profundizar en la determinación de las variables relevantes para el problema de la deserción y establecer un procedimiento de captura de dichas variables, al momento que el alumno se matricule en la institución.

- **Bibliografía**

**Bibliografía Impresa:**


[Anu10] Utilización de Técnicas de Data Warehouse para la toma de decisiones en el área académica. Anuario de Investigaciones. RE 2010. ISBN: 978-987-1635-55-9. Pp.173-178 y RE 2011. ISBN: 978-987-1635-77-1. Pp.119-122

[Anu12] Implementación de un Data Warehouse para la toma de decisiones en el área académica. Anuario de Investigaciones. R. Entendidos 2012. ISBN: 978-987-3806-01-8. Pp.161-166 y RE 2012. ISBN: 978-987-3806-30-8. Pp.73-79.

[Her04] Hernández Orallo, J., Ramírez Quintana, M., and Ferri Ramírez, C. (2004). Introducción a la Minería de Datos. Ed. Pearson

[Wit11] Witten IH, Frank E, Hall MA. *Data Mining: Practical Machine Learning* Tools and Techniques Third Edition ed. Burlington: Morgan Kaufmann Publishers; 2011.


**Referencias**


[1] The 8th International Conference on Educational Data Mining Edm 2015. [En línea] http://www.educationaldatamining.org/ EDM2015/

[2] http://www.educationaldatamining.org/EDM2015/ http://www.educationaldatamining.org/EDM2016/

[3] Sositto, Osvaldo; Etcheverry, Martín; Ryckeboer, Hugo & Bossero, Julio (2010). Aplicación de técnicas de minería de datos para la evaluación del rendimiento académico y la deserción estudiantil. http://www.iiis.org/cds2010/cd2010csc/cisci_2010/paperspdf/ca156fk.pdf

[4] "*La Minería de Datos como un Método Innovador para la Detección de Patrones de Deserción Estudiantil en Programas de Pregrado en Instituciones de Educación Superior".* Disponible en: http://www.acofipapers.org/index.php/acofipapers/2013/paper/viewFile/211/112

[5] *"Redes Neuronales para predecir el rendimiento académico de los alumnos ingresantes a la carrera de Bioquímica de la FACENA-UNNE en función de sus conocimientos matemáticos previos".* Disponible en: *http://sedici.unlp.edu.ar/bitstream/handle/10915/23739/Documento_completo.pdf?sequence=1*

[6] *"Modelo neuronal para la estimación del riesgo de deserción en alumnos de grado".* Disponible en: http://42jaiio.sadio.org.ar/proceedings/simposios/Trabajos/EST/06.pdf

[7] "*Análisis de rendimiento académico estudiantil usando data warehouse y redes neuronales".* Disponible en: http://www.scielo.cl/scielo.php?pid=S0718-33052011000300007&script=sci_arttext

[8] *"Aplicación de técnicas de minería de datos para predecir la deserción".* Disponible en: http://www.utim.edu.mx/~svalero/docs/MineriaDesercion.pdf

[9] "*La Minería de Datos como un Método Innovador para la Detección de Patrones de Deserción Estudiantil en Programas de Pregrado en Instituciones de Educación Superior".* Disponible en: http://www.acofipapers.org/index.php/acofipapers/2013/paper/viewFile/211/112

[10] *"Redes Neuronales para predecir el rendimiento académico de los alumnos ingresantes a la carrera de Bioquímica de la FACENA-UNNE en función de sus conocimientos matemáticos previos".* Disponible en: *http://sedici.unlp.edu.ar/bitstream/handle/10915/23739/Documento_completo.pdf?sequence=1*

[11] http://www.cs.waikato.ac.nz/ml/weka/

[12] https://aaai.org/ojs/index.php/aimagazine/article/view/1011

[13] https://link.springer.com/article/10.1007%2FBF00993309

[14] "*Clasificadores Knn-I*" de Ricardo Aler Mur Disponible en: http://ocw.uc3m.es/ingenieria-informatica/analisis-de-datos/transparencias/KNNyPrototipos.pdf

[15] McCulloch, W.S. & Pitts, W. Bulletin of Mathematical Biophysics (1943) 5: 115. doi:10.1007/BF02478259. Disponible en: https://link.springer.com/article/10.1007/BF02478259

[16] Rosenblatt, F. "*The perceptron: A probabilistic model for information storage and organization in the brain"*. Psychological Review, Vol 65(6), Nov 1958, 386-408. Disponible en :http://psycnet.apa.org/index.cfm?fa=buy.optionToBuy&id=1959-09865-001