UNIVERSIDAD TECNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

ÁREA TÉCNICA

TITULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Análisis de Técnicas de aprendizaje automático para la clasificación de palabras en un curso virtual de la plataforma MOODLE

TRABAJO DE TITULACIÓN

AUTOR: Vivanco Castillo, Santiago Andrés

DIRECTOR: Valdiviezo Díaz, Priscila Marisela, Ing.

LOJA - ECUADOR

2015

Septiembre, 2016

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN

Ingeniera.

Prisila Marisela Valdiviezo Díaz.

DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de titulación: Análisis de Técnicas de aprendizaje automático para la clasificación de palabras en un curso virtual de la plataforma MOODLE, realizado por Santiago Andrés Vivanco Castillo ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, septiembre de 2015

f).....

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

"Yo Vivanco Castillo Santiago Andrés, declaro ser autor del presente trabajo de titulación: Análisis de Técnicas de aprendizaje automático para la clasificación de palabras en un curso virtual de la plataforma MOODLE, de la Titulación Sistemas Informáticos y de Computación, siendo Prisila Marisela Valdiviezo Díaz directora del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

Adicionalmente declaro conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja, que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado que se realicen a través, o con el apoyo financiero, académico o institucional (operativo) de la Universidad"

f).....

Autor: Santiago Andrés Vivanco Castillo

Cédula: 1105106908

DEDICATORIA

A mis padres y hermanas por su apoyo, comprensión, sacrificio, amor y constante aliento en todo momento, quienes me han apoyado incondicionalmente en cada etapa de mi vida y ahora más que nunca con mucha paciencia y cariño.

A mis amigos y compañeros quienes estuvieron siempre dispuestos a apoyarme, soportarme e impulsarme a seguir adelante.

A mis maestros que nunca desistieron al enseñarme y guiarme.

A Dios, por ser el creador de todo, por guiarme, bendecirme y acompañarme día a día en el transcurso de mi vida.

AGRADECIMIENTO

Quiero extender un agradecimiento especial a la ingeniera Prisila Valdiviezo, que como directora de tesis ha sabido compartir sus conocimientos y experiencia, para poder realizar este proyecto de la mejor manera, a mi familia que me alentó incondicionalmente a seguir adelante y a mis amigos y compañeros que colaboraron con el presente proyecto. A Dios, por la dicha de gozar de salud y vida.

A los Ingenieros: Guido Friofrío y Luis Chamba por su colaboración como miembros del jurado de tesis, que supieron guiarme en la investigación con sus observaciones y sugerencias, así como por su tiempo y dedicación.

A mis profesores, que despejaron mis inquietudes con sus conocimientos, durante mi crecimiento académico.

ÍNDICE DE CONTENIDOS

INDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE GRÁFICOS

RESUMEN

En el presente trabajo de titulación se evaluaron dos técnicas de aprendizaje automático (AA): Redes Neuronales y Redes Bayesianas, para ello se utilizaron dos dataset y dos herramientas AA, con mensajes reales extraídos de la plataforma MOODLE de la Universidad Técnica Particular de Loja. Específicamente de un seminario con contenido de Desarrollo Web, y de una clase de Estructura de Datos, ambos de la modalidad de estudios a distancia.

Para la elaboración de las pruebas y el cumplimiento de los objetivos, se usó “Weka” en vista de que permite utilizar un mayor número de algoritmos y una mejor visualización de los resultados. Se aplicaron dos tipos de validación que son Percentage Split y Cross-validation en cada uno de los algoritmos de dichas técnicas. De esta manera se comparó los resultados; por lo que se seleccionó Redes Bayesianas por obtener mayor porcentaje en las instancias correctamente clasificadas, mayor número de precisión y cobertura. El algoritmo seleccionado fue “Multinominal Naive Bayes” y validación por “Percentage Split” por obtener 88,97% de instancias correctamente clasificadas y 0,891 de precisión con el dataset en español, también obtuvo 80% de instancias correctamente clasificadas y 0,8 de precisión con el dataset en inglés.

Aplicando Redes Bayesianas en la experimentación con los mensajes de la plataforma MOODLE, como resultado de la clasificación por polaridad se obtuvo: en el curso de Estructura de Datos de la modalidad a distancia un total de 48 elementos clasificados positivos y 21 elementos clasificados negativos. Por lo que podemos decir que 48 mensajes reflejan sentimientos positivos y 21 reflejan sentimientos negativos.

En cuanto a la clasificación por sentimiento en el seminario de la modalidad a distancia tenemos 44 mensajes clasificados en la clase “Simpatía”, 22 mensajes clasificados en la clase “Angustia” y 22 mensajes clasificados en la clase “Frustración”.

ABSTRACT

The present work aims to try some of the machine learning techniques; focused on the use of neural networks and Bayesian networks, to obtain the most accurate algorithm with which the text classification can be perform. Some datasets and tools were implement to apply the algorithms with actual MOODLE messages, previously its necessary to made the natural language processing through labor "Adaptation of a processing tool for natural language labeling feelings and analysis of language used in Spanish "focused on feelings. Now days there are many machine learning techniques which help us to implement programs that can improve their performance through experience at the time we are getting results.

A comparison of the results of the tools and techniques were performed; so Bayesian Networks was selected to obtain better result by testing in both instances correctly and accurately classified. The algorithm selected was "Multinomial Naive Bayes" and validation by "Percentage Split" for best results. We worked with the "Weka" tool that allows us to use as many of these techniques algorithm and better visualization of results; and the tests were performed with different dataset to end with real-world messages of MOODLE.

INTRODUCCIÓN

En el presente trabajo se evaluaron algunas de las técnicas de aprendizaje automático enfocado en la utilización de Redes Neuronales (RN) y Redes Bayesianas (RB), esto para obtener el algoritmo más preciso con el cual se pueda realizar la clasificación de texto.

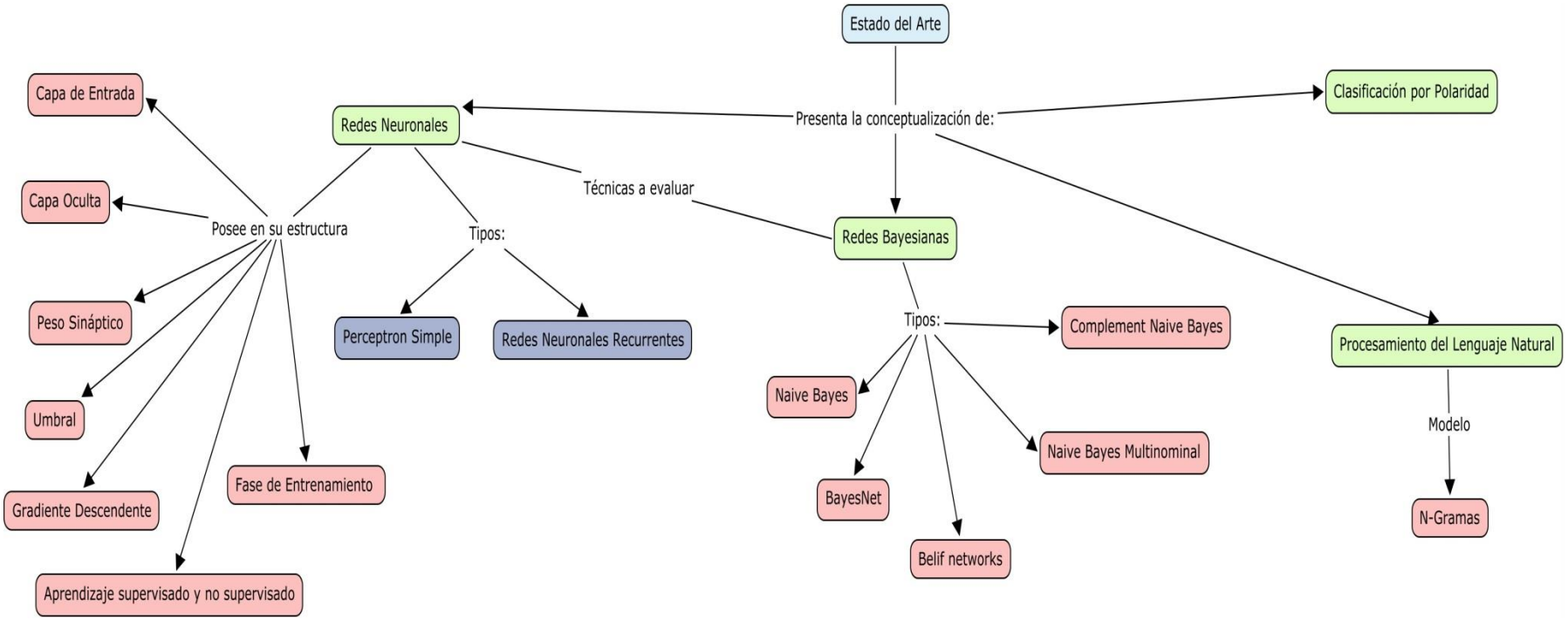
La clasificación consiste en determinar la polaridad de un texto de acuerdo con la opinión y/o el sentimiento del escritor, como positivo o negativo. Para esto aplicaremos estas técnicas a fin de poder construir sistemas que nos ayuden a procesar la información que puedan presentar un cierto comportamiento inteligente. Para realizar estas pruebas se tomó en cuenta herramientas en las cuales podamos aplicar algoritmos para la clasificación.

A continuación se detallan los objetivos que se han considerado para el desarrollo de este proyecto:

- Búsqueda de algoritmos de Redes Neuronales y Redes Bayesianas, se desarrolla en el Capítulo 1. (Estado del Arte), en este se describe cada uno de los algoritmos y otros conceptos necesarios para efectuar el presente trabajo.
- Búsqueda y Análisis de Herramientas. Se realizó en el Capítulo 2 (Análisis y Selección de Técnicas, Algoritmos y Pruebas de Herramientas) presentando el análisis de las herramientas, la descripción de los dataset que se utilizarán.
- Determinar y probar un algoritmo de aprendizaje automático para el análisis de polaridad y de sentimiento en un curso virtual de la plataforma MOODLE de la UTPL. Para cumplir con este objetivo se realizó en el Capítulo 2, las pruebas y resultados de la experimentación con cada una de las técnicas y sus algoritmos, utilizando dos tipos de validación. Luego de estas pruebas se seleccionó la técnica para el análisis de polaridad y de sentimientos.
- Clasificación de polaridad de texto, Clasificación de sentimientos de texto y Análisis de resultados del algoritmo en la plataforma MOODLE se presentan en el Capítulo 3 (Experimentación con Mensajes de un Curso de la Plataforma MOODLE.)

1. ESTADO DEL ARTE

1.1. Mapa Conceptual del Estado del Arte



En el presente Capitulo se detallan los conceptos de las técnicas de Redes Neuronales y Redes Bayesianas con sus respectivos algoritmos, así como también, se describe la estructura que corresponden a dichas técnicas y algunos conceptos que nos permiten clarificar el presente trabajo. Se realiza la descripción de algunos trabajos relacionados que mantienen similitud con este proyecto, con el fin de tener una referencia para la selección de la técnica a utilizar.

1.2. Redes neuronales

El proceso de una red neuronal se limita generalmente en la estructura de una sola capa oculta, debido a las estrategias de formación desfavorables de la red neuronal con múltiples capas ocultas y pesos temporales complejos en el proceso. El aprendizaje profundo ha surgido como un método de pre-entrenamiento eficaz para la red neuronal con múltiples capas ocultas; una red neuronal puede ser entrenada para clasificar los mensajes; las entradas son una serie de valores diferentes con un procedimiento mientras que la salida es un valor estático. Una neurona biológica posee la estructura similar a una neurona de una red neural artificial.

Las redes neuronales artificiales imitan una estructura del sistema nervioso, para así poder construir sistemas que nos ayuden a procesar la información que puedan presentar un cierto comportamiento inteligente. (Bertona 2005)

La idea es construir una red neuronal artificial que funcione y resuelva eficientemente problemas que el cerebro lo puede hacer, para esto resulta convenientemente, construir un sistema que copie la estructura de las redes neuronales bilógicas, para así poder tener una función similar.

Un modelo de red neuronal es generalmente útil si está definida para ayudarnos a entender más al mundo, y sí este nos ayuda a realizar unas predicciones útiles acerca de cómo se comportará el mundo. Las redes neuronales son modelos de procesamiento de información y distribución parecidas a las redes neuronales biológicas, éstas se utilizan más para la clasificación ya que adquiere conocimiento a través de la experimentación.

Una red neuronal tiene por lo menos dos componentes en su estructura física, los elementos de procesamiento y las conexiones entre ellos. Los elementos de la estructura de procesamiento se llaman neuronas, y las conexiones que se encuentran entre dichas neuronas se conocen como enlaces. (Zhang, Kuldip et al. 2003)

Las redes neuronales están distribuidas por capas y en cada una existen neuronas, como se puede observar en la Figura 1, las cuales reciben información en forma de entrada, la procesan y

envían información a otra capa. Las redes neuronales pueden tener cualquier número de capas, y cualquier número de nodos por capa.

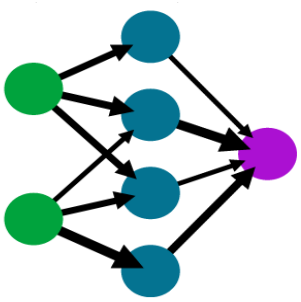


Figura 1. Red Neuronal
Fuente: (Muab'Dib 2008)

Se denomina capa, al conjunto de neuronas que poseen una entrada y una salida. Cada neurona recibe un estímulo de las neuronas vecinas conectados a ella, procesa la información, y produce una salida. En la capa de entrada se recibe la información del exterior y las neuronas que reciben estímulos de fuera de la red se llaman neuronas de entrada. Las conexiones entre cada neurona tiene un peso sináptico que aprende con la modificación de dicho peso. (Zhang, Kuldip et al. 2003)

La capa donde se realizan las operaciones de activación o que no tienen conexión con el mundo exterior ya sea de entrada o de salida se denomina capa oculta, y, las neuronas que se encuentran en esta capa reciben y producen estímulos para otras neuronas, se las conocen como neuronas ocultas (Figura 2). La capa final que es la de la salida de la información posee neuronas cuyas salidas se utilizan externamente y se denominad neuronas de salida. (Zhang, Kuldip et al. 2003)

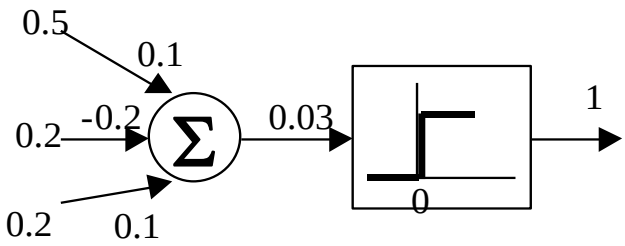


Figura 2. *Estructura de una Red Neuronal*
Fuente: El Autor.

1.2.1. Capa de entradas.

Esta se comunica con una o más capas ocultas, donde se encuentra los nodos o neuronas conectadas al exterior, y que son quienes reciben la información y se las denomina Xn; ellas reciben un único valor en su entrada y duplican el valor de sus múltiples salidas.

1.2.2. Peso sináptico.

El peso sináptico viene a ser la importancia que se le da a cada una de las entradas, se conoce como la fuerza de la entrada con la conexión. Estos pesos son randómicos por lo que pueden ser modificados dependiendo de las respuestas de entrenamiento, calculando el ajuste de pesos.

1.2.3. Umbral.

Umbral se conoce a los valores de salida en que la red neuronal puede tener para ser activada.

1.2.4. Capa oculta.

Los valores que entran en un nodo oculto se multiplican por pesos, un conjunto de números predeterminados almacenados. La parte fundamental de una red neuronal es la llamada neurona que es donde se realizara el cálculo de las entradas por los pesos. Esta capa posee las funciones de activación que está compuesta con una **función de suma** y función de activación que puede ser una **función línea o sigmoidal**.

1.2.4.1. Función suma.

Esta función realiza la suma de las entradas por los pesos asignados. La fórmula viene dada por la siguiente función:

$$\sum(Xn*Wi)$$

1.2.4.2. Funciones de activación.

Función lineal base, es la función que realiza la combinación básica de las entradas con el peso, en la cual se define un límite inferior y otro superior conocido como el umbral. Si la suma de las

señales de entrada es menor que el umbral se define la activación como 0 y si la suma es mayor se define como 1.

$$y=F(x)=x$$

1.2.4.3. Función sigmoidea.

Los valores de salida que produce esta función, están dentro de un rango que va de 0 a 1. La función de activación sigmoidea es más útil para datos de entrenamiento que es también entre 0 y 1.

1.2.4.4. Función tangente hiperbólica.

Los valores de salida de la función tangente hiperbólica están comprendidos dentro de un rango que va de -1 a 1.

1.2.5. Fase de entrenamiento.

La fase de entrenamiento busca un conjunto de pesos que permitan a la red realizar correctamente una determinada tarea. Durante la fase de entrenamiento se va mejorando la solución hasta alcanzar un nivel de resultado suficientemente bueno.

Se propone una función que mide el rendimiento actual de la neurona en función de los pesos. El objetivo de la fase de entrenamiento es encontrar el conjunto de pesos sinápticos que mejore la función. El método de optimización propone una regla de mejora de los pesos que conjuntamente con las entradas modifican los pesos hasta alcanzar el punto óptimo de la red neuronal (Bertona, 2005)

Lo que se desea obtener, es que una aplicación determinada con un conjunto de entradas, produzca el conjunto de salidas deseadas o consideradas aceptables.

1.2.6. Aprendizaje supervisado y no supervisado.

Los algoritmos supervisados tienen un conjunto de datos de los cuales ya conocemos la salida correcta de nuestro sistema, si conocemos las entradas dado el conjunto de entrenamiento de la red podríamos saber cuál es la salida y cuál es la diferencia con el resultado obtenido. Entre

mayor información de la red y el conjunto de entrenamiento tengamos, mejor podremos aprender (Figura 3).

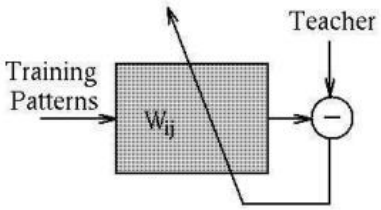


Figura 3. Reglas de entrenamiento Supervisado.
Fuente: (Marín 2011)

Los algoritmos no supervisados muestran un mayor interés por los datos de entrada. Poseen poca información para trabajar y son muy difíciles de construir y realizan fuertes supuestos de lo que están tratando de hacer (Figura 4).

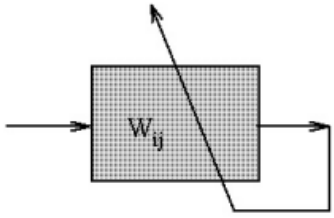


Figura 4. Reglas de entrenamiento No Supervisado.
Fuente: (Marín 2011)

1.2.7. Gradiente descendente.

La técnica de gradiente descendente es un procedimiento iterativo que busca minimizar una función moviéndose en la dirección opuesta al gradiente de dicha función.

1.2.8. Tipos redes neuronales.

1.2.8.1. Percepton simple.

Es el modelo matemático más simple de una neurona, se considera una red de aprendizaje supervisada por lo que necesita los valores de entrada, es unidireccional. El perceptron simple consta de una solo capa de entrada con muchos nodos y una capa de salida con un solo nodo (Figura 5). La función de la primera capa de un perceptron es hacer de entrada, por ella llegan las

señales a la red. La segunda capa realiza todo el procesamiento. Para realizar la explicación de un perceptron simple se la puede entender como el concepto de una red neuronal básica y para esto utilizaremos una función matemáticas que está dada por:

$$f(x) = \begin{cases} 1 \quad si \quad w_1x_1 + w_1x_1 + \ldots + w_ix_1 \geq 0 \\ -1 \quad si \quad w_1x_1 + w_1x_1 + \ldots + w_ix_1 \leq 0 \end{cases}$$

Donde los parámetros w, se llaman pesos sinápticos y se los conoce como los pesos con los que se ponderan los valores de entrada x, la suma ponderada u = $w_1x_1 + w_1x_1 + \ldots + w_ix_1$ se llama potencial sináptico y el parámetro θ se llama umbral o sesgo.

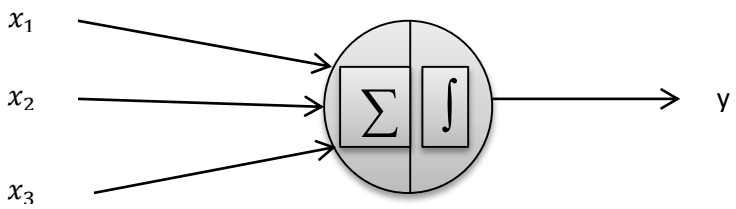


Figura 5. *Estructura Perceptron Simple*
Fuente: El Autor.

Cuando la salida de la unidad de proceso es igual a 1, se dice que dicha unidad está activada o encendida y presenta el estado 1, mientras que si su salida es igual a cero se dice que está desactivada o apagada, presentando el estado 0 (Muñoz 2010).

1.2.8.2. Redes neuronales recurrentes.

Son redes neuronales que presentan una o más interconexiones entre las diferentes neuronas de las capas en cualquier sentido y esto permite tener un mayor número de parámetros de aprendizaje (Figura 6), las salidas de cada neurona dependen no solo de la capa anterior sino también de la capa a la que se encuentran conectadas o a esta misma. Existen tres tipos de conexiones recurrentes entre las neuronas. Estas son de una neurona con ella misma, de una neurona con otra de su misma capa y de una neurona con otra de una capa distinta.

La función de activación no depende solo de las salidas de activación de la neurona anterior, también dependen de la activación de las neuronas conectadas a ésta. En cada paso de tiempo, la RNN recibe una entrada, actualiza su estado oculto, y hace una predicción.

A estas conexiones se le suma lo que se llama como variable tiempo.

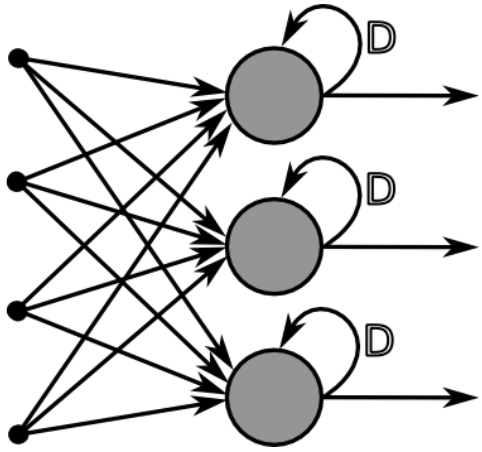


Figura 6. Red Neuronal Recurrente
Fuente: (Cazala 2014)

1.2.8.3. Redes parcialmente recurrente.

Está formada por una capa entrada, dos capas intermedias (una capa oculta y otra capa de contexto) y una capa de salida (Figura 7) (Muñoz 2010).

Estas redes poseen algunas conexiones recurrentes para así recordar el estado o nivel de activación anterior de algunas de las neuronas conectadas, en la capa de entrada de una red parcialmente recurrente se puede encontrar dos tipos de neuronas, las neuronas de activación y las neuronas de contexto. Las neuronas de activación, son las que reciben la señal del exterior y las neuronas de contexto, son las que reciben la señal de las conexiones recurrentes.

Las neuronas de salida reciben la señal de salida de las neuronas de la capa oculta, que está conformada por los correspondientes pesos y se suelen utilizar como función de transferencia. Las neuronas de contexto se utilizan para memorizar las salidas de las neuronas ocultas en la capa anterior, de manera que cada neurona de contexto tiene como salida, la salida de la unidad oculta correspondiente en la capa anterior.

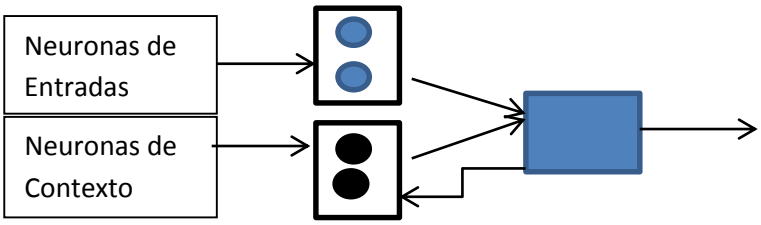


Figura 7. Red Neuronal Recurrente.
Fuente: El Autor.

1.2.8.4. Redes totalmente recurrentes.

Redes totalmente recurrentes son distintas a las redes recurrentes simples, éstas se diferencian en que las redes totalmente recurrentes utilizan actualizaciones concurrentes y envían derivados de error hacia atrás en el tiempo. En una red totalmente recurrente, todos los grupos actualizan primero sus entradas y luego, asimismo actualizan sus salidas. Por lo tanto, la información puede propagar a través de un solo juego de conexiones por garrapata. La red se denomina completamente recurrente, porque la salida de todas las neuronas, están conectadas de forma recurrente a todas las neuronas en la red.

1.3. Redes bayesianas.

Una red bayesiana, se representa como un gráfico probabilístico en el cual los nodos vienen a ser variables aleatorias que se encuentran conectadas a sus dependencias, y los nodos que no se encuentran conectados se los considera como nodos independientes. Estos nodos pueden representar una entidad del mundo real. Todos los estados posibles del modelo representan a todos los mundos posibles que pueden existir, es decir, todas las posibles formas en que las partes o estados pueden ser configurados. En las redes bayesianas las uniones de las neuronas pueden formar bucles, pero no pueden formar ciclos. (Norsys. 2015)

Las Redes Bayesianas, toman en cuenta cada neurona que esta probabilísticamente relacionada por algún tipo de dependencia causal. No se almacenan todas las configuraciones de los estados, todo lo que se necesita para almacenar y trabajar con todas las combinaciones posibles es de los nodos relacionados y estados entre los grupos de padres e hijos.

Las Redes Bayesianas, son redes de relación; se denominan "Bayes" por Thomas Bayes, un teólogo y matemático británico, que escribió una ley básica de la probabilidad que ahora se llama la regla de Bayes. Donde se lee p (A) como la probabilidad de A, y p (A | B) como la probabilidad de A dado que B ha ocurrido.

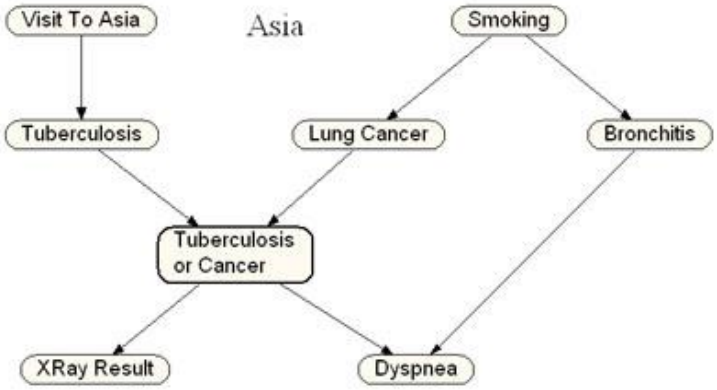


Figura 8. Ejemplo
Fuente: (Norsys. 2015)

Como cita Norsys Software Copr, la red de la Figura 8, podría ser utilizada para diagnosticar pacientes que llegan a una clínica. Cada neurona de la red tiene que ver con alguna condición del paciente. "Visita a Asia" indica si el paciente visitó recientemente Asia. Las flechas indican si existen relaciones de existencia entre los estados de los dos nodos. Por lo tanto, el tabaquismo aumenta las probabilidades de contraer cáncer de pulmón y de contraer bronquitis. Tanto el cáncer de pulmón y bronquitis aumentan las posibilidades de conseguir la disnea. Tanto el cáncer de pulmón y la tuberculosis, pero no suele ser bronquitis, pueden causar un pulmón de rayos x anormal. Etcétera. (“Tutorial on Bayesian Networks with Netica,” n.d.)

Las Redes Bayesianas, poseen diferentes aplicaciones para diagnóstico, clasificación y decisión que brinde información importante en cuanto a cómo se relacionan las variables, las cuales pueden ser interpretadas como relaciones de causa-efecto. Existen distintas aplicaciones de interés para las redes bayesianas: sistematización, diagnostico, predicción y clasificación entre otras. El mecanismo de inferencia sobre redes bayesianas permite utilizarlas para construir clasificadores (Rivera 2011).

1.3.1. Belief networks.

Belief Networks o Redes de creencias profundas, utilizan una representación gráfica para representar la estructura probabilística. Se compone de un conjunto de neuronas interconectadas (Figura 9), donde cada neurona representa una variable, las conexiones representan las relaciones causales entre estas variables. Consta de nodos y flechas. Conecta el borde de un nodo padre a otro nodo hijo. En una red de creencia cada nodo se utiliza para representar una variable aleatoria, y cada conexión representa una dependencia inmediata o influencia directa (Storkey 2005).

Las dos características más relevantes de las redes de creencias profundas son:

- El procedimiento es eficaz por cada capa para el aprendizaje de los pesos, se realiza de arriba hacia abajo determinando cómo las variables en una capa dependen de las variables en la capa de superior.

- Después de aprender, los valores de las variables en cada capa pueden ser modificados por un pase de abajo hacia arriba, que empieza con un vector de datos observados en la capa inferior y utiliza los pesos generativos en la dirección inversa. (Hinton 2009).

Redes de creencias son muy utilizadas para realizar aplicaciones basadas en el conocimiento. Se emplean técnicas de red de creencias para ofrecer sistemas basados en el conocimiento avanzado para resolver problemas del mundo real. Son útiles para aplicaciones de diagnóstico y se han utilizado en muchos sistemas funcionales, como la función de ayuda en el producto Microsoft Office que emplea tecnología de red creencia bayesiana. (Harrison 1997)

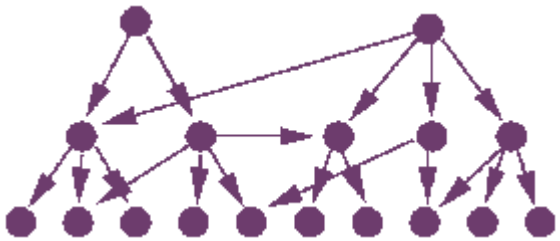


Figura 9. Red de Creencia
Fuente: (Storkey 2005)

1.3.2. BayesNet.

El algoritmo de aprendizaje BayesNet, utiliza algunos tipos de búsqueda y medidas de calidad. Es la clase básica para un clasificador de Bayes y es un clasificador lineal que es conocido por ser simple pero muy eficaz. El modelo probabilístico de los clasificadores de BayesNet, se basa en el teorema de Bayes. (Stanford-University,2011). BayesNet, refleja los estados de una parte de un mundo que se está modelando y describe cómo esos estados están relacionados por probabilidades.

1.3.3. Naive Bayes

Naive Bayes, es un algoritmo de aprendizaje el cual se suele utilizar con frecuencia para resolver algunos problemas de clasificación de texto. Es computacionalmente muy eficiente y fácil de implementar (Kibriya, Frank et al. 2005)

Al momento de realizar investigaciones, se ha estudiado la clasificación de texto, y, el de Naive Bayes, es uno de los métodos más conocidos para ello. (Komiya, Sato et al. 2011).

1.3.4. Complement Naive Bayes (CNB)

Es una modificación del clasificador NaiveBayes. Nos ayuda a trabajar de una mejor manera y tener mayor precisión en la clasificación utilizando todas las categorías, sin tomar en cuenta la categoría en la que se está trabajando.

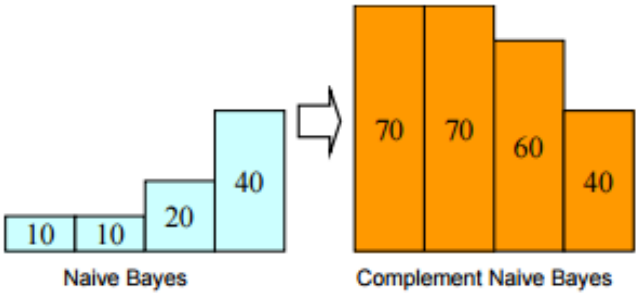


Figura 10. Complement Naive Bayes
Fuente: (Komiya, Sato et al. 2011)

La Figura 10, muestra el entrenamiento del clasificador NaiveBayes y el clasificador CNB. El clasificador NB realiza cálculos de una categoría con los datos de la categoría en la que se centra. Como se muestra en la figura, existen 4 categorías de: 10, 10, 20, 40 datos de entrenamiento. La categoría con el mayor número de datos tiene cuatro veces más datos que la categoría con menos datos. En el caso del clasificador CNB, éste realiza cálculos de parámetros de una categoría, mediante los datos de todas, sin tomar en cuenta la categoría que se centra. Por lo que la categoría con menos datos es 40 y la que mayor datos contiene es de 70. La brecha de la cantidad de los datos de entrenamiento es menor que el clasificador NB. Este clasificador también se utiliza como una línea de base (Kibriya, Frank et al. 2005).

1.3.5. NaiveBayes Multinomial.

El clasificador multinomial bayesiano ingenuo, es adecuado para la clasificación con características discretas. La distribución multinomial, normalmente requiere recuento de característica, es una versión especializada de Naive Bayes, que está diseñado más para documentos de texto.

El modelo multinomial, captura la información de frecuencia de palabras en los documentos. En el modelo multinomial, un documento es una secuencia ordenada de eventos de palabras, extraídas del mismo vocabulario, es decir, unigramas con recuentos de palabras enteras.

McCallum y Nigam (1998), postulan un modelo bayesiano multinomial para la clasificación de texto y muestran un rendimiento mejorado, con un vocabulario más extenso en comparación con otros modelos, esto debido a la incorporación de la información de la frecuencia. Es esta versión multinomial, que llamamos "Multinomial Naive Bayes" (MNB) (McCallum and Nigam 1998).

1.4. Clasificación por polaridad (Sentimiento).

Dentro del amplio campo de la clasificación de texto, existe la tarea más conocida que es realizar la clasificación de polaridad del sentimiento. El análisis de la polaridad tiene como objetivo, determinar la actitud de un escritor con respecto a algún tema, o la polaridad de un documento. La tarea de esta pretende clasificar texto, en positivo o negativo dependiendo de su significado.

La clasificación de la polaridad obtiene un resultado que identifique si el texto expresa una opinión positiva o negativa, esto se puede obtener mediante un rango, el cual si va dentro del rango de 0 indicaría neutral, 1 positiva y -1 negativa.

Para determinar la polaridad se puede seguir dos estrategias: la supervisada y la no supervisada. La supervisada utiliza como la presencia o no de los términos para el cálculo de la polaridad. Los métodos no supervisados, se fundamentan en la detección de identificadores de opinión en los textos, para después calcular la polaridad empleando alguna función basada en los indicadores encontrados. Para la identificación, se utilizan conjuntos de vocablos etiquetados por su polaridad. Los métodos basados en corpus, se basan en las características propias del corpus para la ampliación del lexicón de palabras.

1.5. Procesamiento del Lenguaje Natural.

1.5.1. N-Grama

Un modelo n-grama, es un tipo de modelo probabilístico que permite hacer una predicción del próximo elemento de cierta secuencia de elementos. Los modelos de lenguaje que poseen el orden de las palabras, se llaman modelos de lenguaje n-grama, n representa cualquier número entero mayor que cero.

Modelo N-grama, se puede separar u ocultar algunas palabras sobre una frase o un texto, para que sólo n palabras sean visibles al mismo tiempo. El modelo de n-grama más simple es el unigram. En éste modelo solo podemos fijarnos en una palabra a vez.

La sentencia hola mundo Ecuador Loja, por ejemplo, contiene tres unigrams: "hola", "mundo", "Ecuador", "Loja".

Unigrams:

Hola,

Mundo,

Ecuador,

Loja

Un **Bigrama,** puede separar dos palabras sobre una frase o un texto, para que sólo dos palabras sean visibles al mismo tiempo.

Bigramas:

Hola mundo,

mundo Ecuador,

Ecuador Loja.

Podríamos decir, que todos los gramas de una oración se pueden encontrar separando o las dos primeras palabras, y al irnos moviendo por la frase de forma escalonada. Se repite esto hasta las dos últimas palabras de una frase. De manera similar se puede hacer lo mismo para unigrams y N-grams con n mayor que dos (Lin 2009).

1.6. Trabajos relacionados

1.6.1. Un Estudio sobre redes neuronales recursiva basada en la clasificación de sentimientos de Sina Weibo (A Study on Recursive Neural Network Based Sentiment Classification of Sina Weibo).

En este trabajo se utiliza una red recursiva, la cual se la entrenó para realizar la clasificación de sentimientos de mensajes en la red social "Sina Weibo", teniendo en cuenta el significado sintáctico y semántico de la oración. Según Chen Fu, se realizó extensos experimentos en gran conjunto de datos de Sina Weibo y estos demuestran que este modelo, supera consistentemente los modelos existentes de la clasificación de sentimiento al identificar el sentimiento oculto o implícito.

Esto se realiza con el fin de detectar los eventos en las poblaciones, el seguimiento de posibles epidemias, las encuestas de opinión de modelo o incluso inferir resultados de las elecciones. Es crucial entender el contenido generado por usuarios en las redes sociales, blogs o comentario. Detectar sentimiento en estos datos, es una tarea difícil que ha generado recientemente una gran cantidad de interés. El modelo se basa en el algoritmo autoencoder recursivo y Word2vec para analizar el sentimiento de condenas en Sina Weibo (Fu, Xue et al. 2014)

1.6.2. Análisis de Sentimientos sobre un Corpus en Español: Experimentación con un Caso de Estudio.

Este trabajo realiza la clasificación de textos en idioma español, utilizando métodos de aprendizaje supervisado y no supervisado y se presenta resultados basados en experiencias sobre la efectividad de algoritmos de clasificación, los que ya han sido utilizados con éxito en trabajos previos para el idioma inglés.

Trabaja con los algoritmos de Naıve Bayes, Modelos de Máxima Entropía, Decisión Trees, Support Vector Machines y una adaptación del algoritmo no supervisado de Turney para el idioma español, con el objetivo de comparar y evaluar performance en función de distintos pre

procesamientos de textos propuestos, tipos de atributos extraídos y tamaños de corpus que van desde 500 hasta 22000 documentos.

Se utilizó corpus de datos de comentarios junto con el puntaje asignado por el usuario y se asignaron etiquetas a los documentos utilizando siguiente criterio: se asignó la etiqueta "POSITIVO", a aquellos comentarios cuya suma de puntos sea 10 o superior; se asignó la etiqueta "NEGATIVO", a aquellos comentarios cuyo puntaje en la categoría "comida" (identificada como la más relevante para el dominio estudiado) sea 1 (malo/regular) o bien sea 2 (bueno) y que el resto de las categorías tengan ambas puntaje 1 (malo/regular).

Para la implementación del algoritmo Naive Bayes, utilizaron la herramienta NLTK. El sitio con el que trabaja es una página web de crítica gastronómica www.guiaoleo.com, ya que los usuarios dan varias opiniones. El trabajo como resultado muestra que con Naıve Bayes, se obtienen los mejores resultados para corpus pequeños, pero su performance decrece levemente para los tamaños de corpus más grandes. Y que si se utilizan bigramas como atributos Naıve Bayes presenta los mejores resultados (Dubiau and Ale 2013)

1.6.3. Redes neuronales recurrentes para clasificaciones de series de tiempo (Recurrent Neural Networks for Time Series Classification)

En este trabajo se utiliza Redes Neuronales Recurrentes para realizar una clasificación en diferentes clases especificadas. La predicción ayuda al desarrollo de una estructura adecuada, que representa las principales características de la entrada de una red neuronal, para resolver el problema de clasificación. Por lo tanto, la velocidad y el éxito de la formación, así como, la capacidad de generalización de la RNN entrenados se mejoran significativamente. El RNN entrenado proporciona un buen rendimiento de clasificación y permite al usuario evaluar eficientemente el grado de fiabilidad del resultado de clasificación (Husken and Stagge 2003).

1.6.4. Redes Neuronales Recurrentes para la clasificación de texto robusto del mundo-real (Recurrent Neural Networks for Robust Real-World Text Classification)

Este trabajo explora la aplicación de redes neuronales recurrentes para la tarea de clasificación robusta de texto en un corpus de evaluación comparativa en el mundo real. Hay muchos enfoques bien establecidos, que se utilizan para la clasificación de texto, pero no abordan el reto de un punto de vista más multidisciplinar, como el procesamiento del lenguaje natural y la inteligencia artificial. Los resultados demuestran que estas redes neuronales recurrentes pueden ser una

adición viable a las muchas técnicas utilizadas en la inteligencia Web para tareas como contexto de clasificación de correo electrónico sensible y la indexación del sitio web (Arevian 2007).

1.6.5. Acercamiento al Análisis de Sentimientos mediante aprendizaje semi-supervisado de Clasificadores multidimensionales (Approaching Sentiment Analysis by Using Semi-supervised Learning of Multidimensional Classifiers)

En este trabajo, se resuelve un problema del mundo real multidimensional, problema que consiste en caracterizar la actitud de un cliente cuando escribe un post sobre un tema en particular en un foro específico, a través de tres dimensiones de manera diferente relacionados: Voluntad de influir, Polaridad sentimiento y subjetividad.

Debido al hecho de que tiene tres variables diferentes, las técnicas de clasificación tienen un estado de Ofthe-art (unidimensionales) para obtener soluciones subóptimas. Por esa razón, se propone el uso de clasificadores de red bayesiana multidimensionales, como una nueva herramienta metodológica para tomar ventaja de estas relaciones existentes entre estas variables objetivo. Además, con el fin de evitar la ardua y lenta tarea de ejemplos de etiquetado en este campo, se presentan estas técnicas multidimensionales en ambos marcos de aprendizaje supervisado y semi-supervisados (Ortigosa, Rodrıgueza et al. 2011)

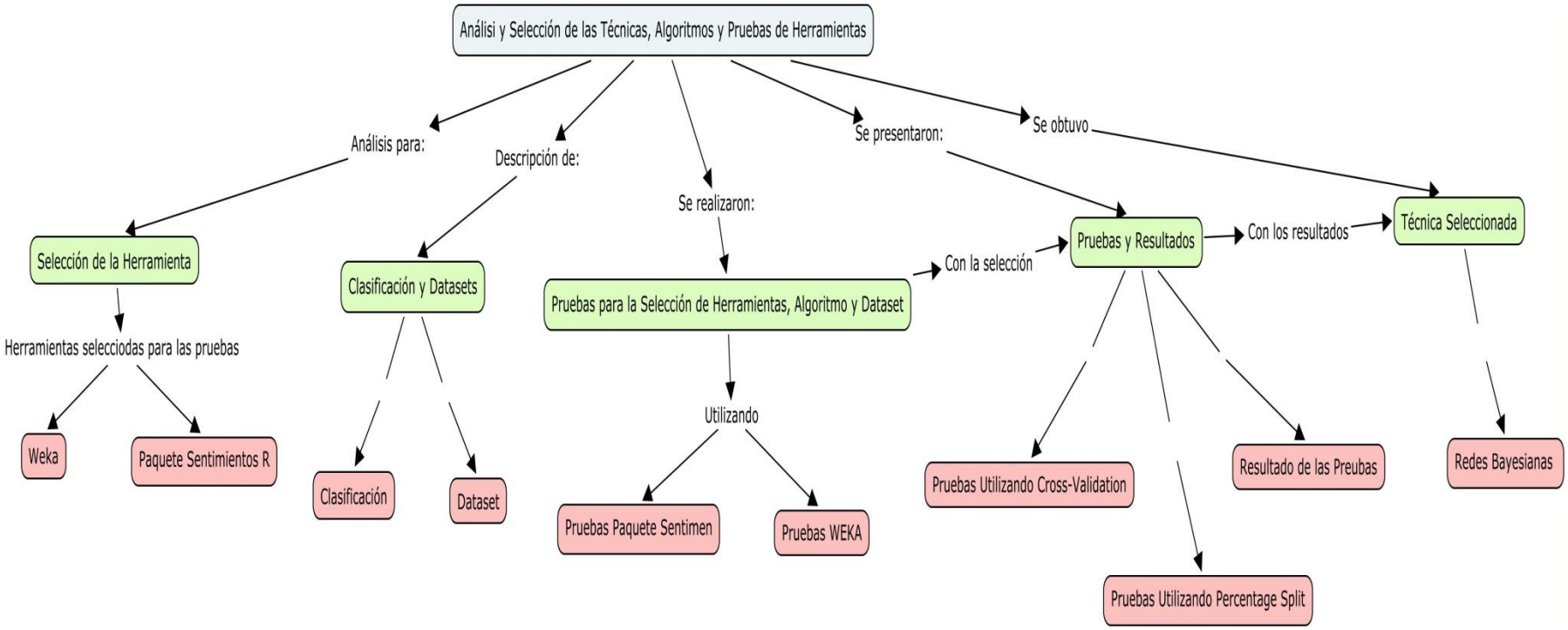
1.6.6. Un estudio empírico de aprendizaje masivo paralelo de redes bayesianas para la extracción de sentimiento de texto no estructurados. (An Empirical Study of Massively Parallel Bayesian Networks Learning for Sentiment Extraction from Unstructured Text).

En este trabajo, presentamos un algoritmo paralelo para BN (Redes Bayesianas) estructura apoyada desde dateset a gran escala mediante el uso de un clúster MapReduce.

Luego, aplicamos este algoritmo de aprendizaje BN paralelo para capturar las dependencias entre palabras, y, al mismo tiempo, encuentra un vocabulario que es eficaz para el propósito de sentimientos de extracción. Se discuten los beneficios del uso de MapReduce para BN estructura de aprendizaje. El rendimiento de la utilización de BN para extraer sentimientos se demuestra mediante su aplicación a los datos de blog web reales. Los resultados experimentales en el conjunto demostración de datos web que nuestro algoritmo es capaz de seleccionar un conjunto de características parsimoniosa con substancialmente menos variables predictoras que en el conjunto de datos completo y conduce a mejores predicciones sobre orientaciones sentimiento que varios métodos utilizados habitualmente (Chen, Zong et al. 2011).

2. ANÁLISIS Y SELECCIÓN DE LAS TÉCNICAS, ALGORITMOS Y PRUEBAS DE HERRAMIENTAS

2.1. Mapa Conceptual



En el presente capítulo se realiza la descripción de la herramienta WEKA, el paquete sentiment de R y los tipos de clasificación. Se presentaron dos dataset, el primero en inglés y el segundo en español. Se realizaron las pruebas para la selección de la herramienta, la técnica y de los algoritmos. Para esto cada uno de los dataset se clasifico con los distintos algoritmos utilizando dos tipos de validación que son: cross-validation y percentage Split. Una vez obtenido el resultado se procedió a comparar las instancias correctamente clasificadas, la precisión y la cobertura. De esta manera se presentan los resultados y la técnica seleccionada para realizar la experimentación con mensajes de un curso de la plataforma MOODLE.

2.2. Selección de la Herramienta.

2.2.1. Weka.

Weka, es una colección de algoritmos desarrollados en JAVA para la extracción y análisis de conocimientos. Los algoritmos se pueden aplicar directamente a un conjunto de datos o ser llamados desde su propio código Java. Weka contiene herramientas para realizar operaciones de transformación sobre los datos, tareas de clasificación, regresión, clustering, asociación y visualización.

2.2.1.1. Formas de leer los datos desde el programa.

Se lo puede realizar desde:

- DataSet, conjunto de archivos txt
- Archivos .CSV atributos separados por comas
- Archivos arff

2.2.2. Paquete sentimientos R.

R es un entorno y lenguaje de programación mediante el cual podemos tratar datos y graficar. Está orientado a las estadísticas, proporciona algunas herramientas que nos pueden ser útiles para el cálculo numérico y la minería de datos.

Una de las herramientas para realizar el análisis de sentimientos es el paquete de R “sentiment” por Timothy Jurka (2012). Este paquete está actualmente desarrollado para trabajar con texto en inglés, y nos proporciona dos funciones:

2.2.2.1. Clasificación de la emoción (classify_emotion).

Esta función nos ayuda a analizar un texto y clasificarlo en diferentes tipos de emociones: ira, asco, miedo, alegría, tristeza y sorpresa.

2.2.2.2. Clasificación de la polaridad (classify_polarity).

En contraste con la clasificación de las emociones, la función classify_polarity nos permite clasificar un texto como positivo o negativo.

Tabla 1. Comparación entre las herramientas.

Herramienta	Ventajas	Desventajas
Weka	• Desarrollado en Java • clasificación (redes neuronales, reglas y árboles de decisión, aprendizaje Bayesiona) • Trabajar con un conjunto de frases	• Resultados que esperan ser interpretados • Escasa documentación orientada al usuario • Previa clasificación en algunos Casos
Paquetes Sentiment	• Código abierto. • Métodos Establecidos para el análisis de sentimientos y su polaridad	• Restricción en cuanto al algoritmo de Bayes • Escasa documentación orientada al programador • Trabaja con una frase a la ves

Elaboración: El Autor.

2.3. Clasificación y Datasets

2.3.1. Clasificación

Para realizar la clasificación se tiene como entrada el documento y un conjunto de clases fijo, y como salida se obtiene una clase de predicción; se puede realizar dos tipos de clasificación que son:

- Hand-coded rules(Reglas de codificación manual)
    - Reglas basadas en combinaciones de palabras u otras características las cuales poseen una lista de normas realizada por expertos.
    - La exactitud del resultado puede ser alto.
    - Pero la construcción y el mantenimiento de las normas puede ser tedioso.
- Supervised Machine Learning
    - Como entrada se realiza un documento con todas las instancias.
    - Conjunto fijo de las clases

- Conjunto de entrenamiento de los documentos etiquetados a mano
- Como salida se obtiene un clasificador de aprendizaje.

2.3.2. Dataset

Para las pruebas en las cuales se determina el algoritmo que se aplicará, se utilizó dos dataset, el uno está formado por 2000 archivos de críticas de películas en inglés de la empresa IMDb.com, Inc (Internet Movie Database) y creado por el Departamento de Ciencias de la Computación de la Universidad de Cornell.

El segundo dataset está formado por 400 archivos de críticas de carros, hoteles, máquinas, libros, celulares, música, computadoras, y películas en español realizado por Maite Taboada; profesora de la Simon Fraser University.

2.4. Pruebas para la selección de herramientas, algoritmo y dataset.

A continuación se realizaron las pruebas utilizando la herramientas WEKA y el paquete sentiment de R, se aplicaron las técnicas de redes bayesianas y de redes neuronales, en el caso del paquete sentiment sólo se aplicará la técnica de redes bayesianas por las limitación de este paquete como se mencionó en la Tabla 1. Se utilizarán algunos algoritmos de estas técnicas para así poder obtener el mejor resultado en la clasificación.

2.4.1. Pruebas paquete sentiment.

Instalar el paquete “sentiment”, el cual ahora se encuentra fuera de la página de CRAN (The Comprehensive R Archive Network), que es una red de servidores web de todo el mundo la que almacenan versiones de código y documentación para R. Por lo tanto se tiene que instalar el paquete mediante líneas de código.

Si se utiliza la plataforma R, es más extenso el proceso de cargado de los paquetes necesarios para el análisis de sentimientos, por lo que utilizamos Rstudio, el cual nos da una interfaz y nos permite cargar el paquete sentiment y automáticamente se agregan los demás paquetes necesarios.

Una vez cargadas las librerías, se puede utilizar el paquete “sentimet” directamente colocando las siguientes líneas de código en la pantalla de comandos de Rstudio

```
# LOAD LIBRARY

library(sentiment)

 # DEFINE DOCUMENTS

documents <- c("I am very happy, excited, and optimistic.")

# CLASSIFY EMOTIONS

classify_emotion(documents,algorithm="bayes",verbose=TRUE)
```

Las cuales nos permiten obtener la clasificación de una frase con un algoritmo básico de bayes.

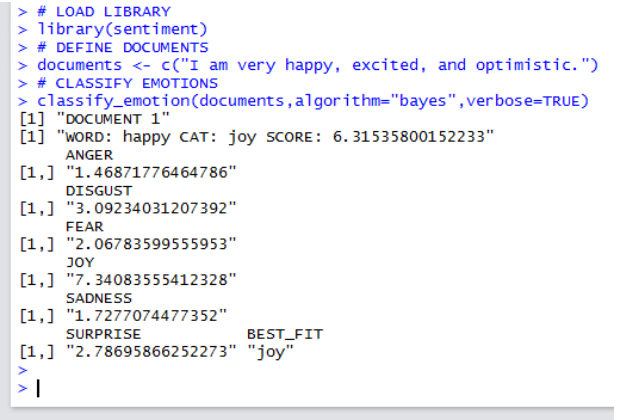


Figura 11. Resultado clasificación con algoritmo en paquete sentiment

Como podemos observar en la Figura 11, en el resultado obtenemos una puntuación general de la frase, seguidos por las distintas puntaciones de los sentimientos y al final la mejor puntuación en la clasificación de la frase.

2.4.2. Pruebas Weka.

Para trabajar Weka utilizaremos un conjunto de texto con particiones en dos subcarpetas positivas y negativas.

En Weka podemos importar de una manera simple el conjunto de datos de texto, mediante el TextDirectoryLoader para conjuntos de textos en diferentes archivos o con la opción de abrir archivos para archivos .arff o .csv (Figura 12).

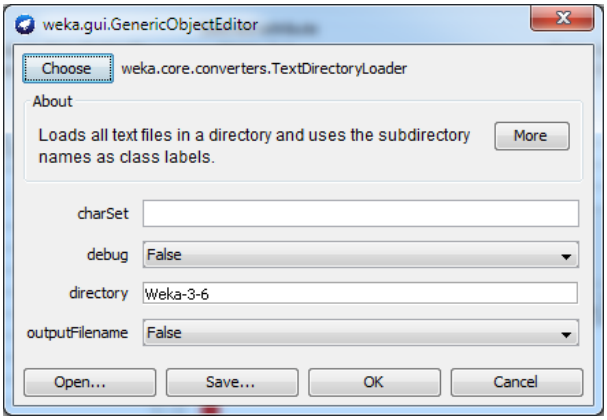


Figura 12. *Importación de datos*
Elaboración: *El Autor.*

Esto nos permite crear dos atributos como se puede observar en la Figura 13, el primero posee todo el texto y el segundo el de clase que posee los dos atributivos que se encuentran definidas por las sub-carpetas.

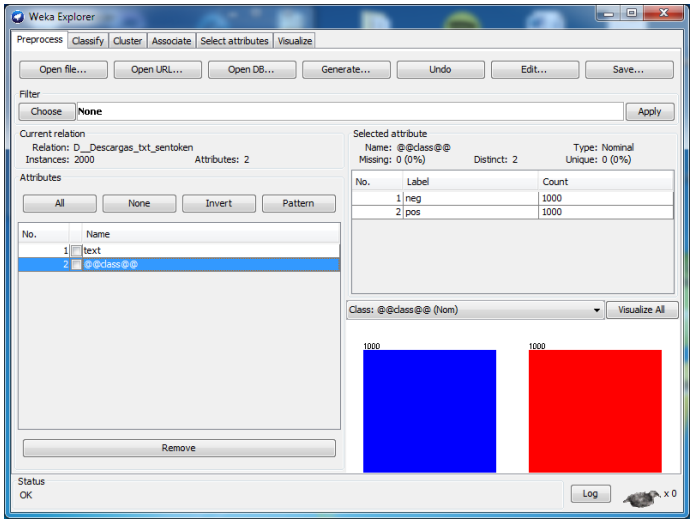


Figura 13. Creación de atributos
Elaboración: El Autor

Se tiene que realizar un pre-procesamiento en el cual se transforma el texto, para esto se realizarán operaciones con la finalidad de preparar los datos, con el objetivo de adaptarlos para aplicar la técnica de clasificación.

Para realizar el pre-procesamiento en WEKA, se puede usar un filtro no supervisado que es el StringToWordVector. Este filtro nos ayuda a convertir las cadenas en un conjunto de atributos representados por la concurrencia de las palabras, cada documento es representado por la presencia (o frecuencia) de algunos de los términos "importantes".

Cambiaremos el atributo “stopwordsHandler” de este filtro para aplicar stopwords, que son las palabras sin significado como: artículos, pronombres, preposiciones, etc. que son filtradas antes o después del procesamiento de datos en lenguaje natural. Para esto utilizaremos “RegExpFromFIle” que utiliza las expresiones almacenas en un archivo para determinar si una palabra es una palabra vacía. Para esto tenemos dos archivos de stopwords el primero con palabras vacías en español proporcionado por el proyecto de tesis “Adaptación de una herramienta de procesamiento de lenguaje natural para el etiquetado de sentimientos y el análisis de lenguaje en español”. El segundo con palabras vacías en inglés de la empresa IMDb.com, Inc (Internet Movie Database) y creado por el Departamento de Ciencias de la Computación de la Universidad de Cornell.

Cada frase es representada mediante un vector booleano n-dimensional, donde n es el tamaño del vocabulario, y cada valor por la presencia o la ausencia de un término en la frase.

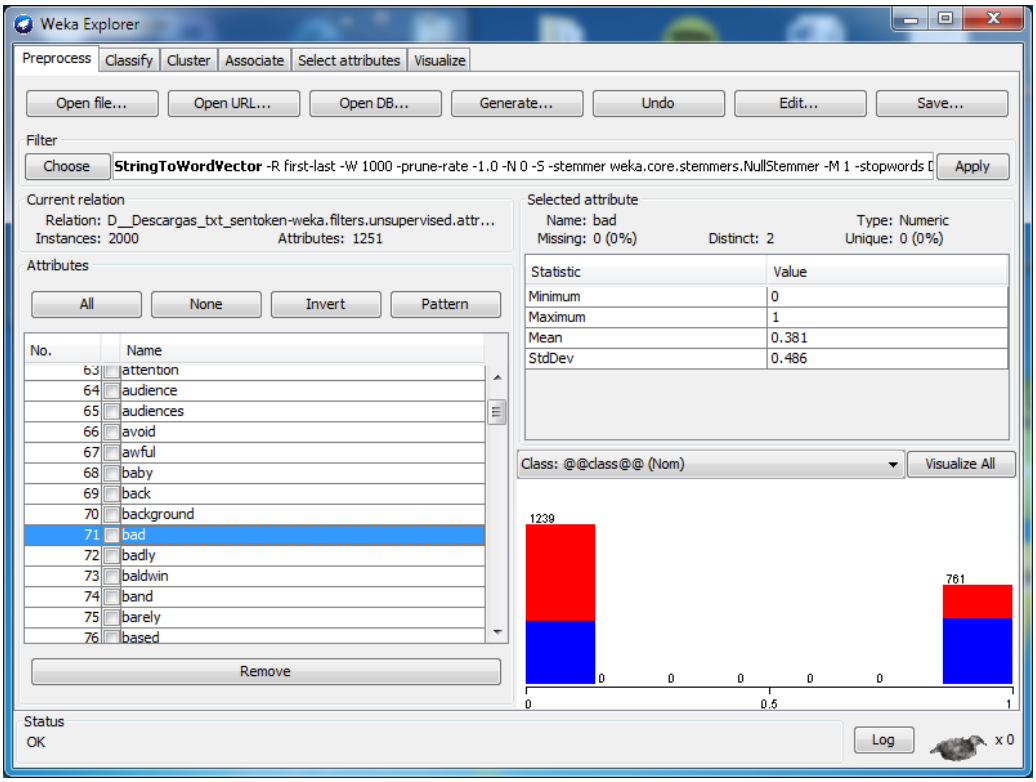


Figura 14. Creación de atributos
Elaboración: El Autor

Ahora tenemos más atributos que son las diferentes palabras, como podemos observar en la Figura 14 la palabra bad (malo) aparece en el valor 1 en más críticas negativas que son las que están representadas con el color azul.

Luego utilizaremos un filtro supervisado que es AttributeSelection (Figura 15) el cual nos ayudara a trabajar con los atributos que tienen mayor correlación con la clase y eliminar los que se encuentran mal caracterizados, de esta manera podremos tener una mejor clasificación.

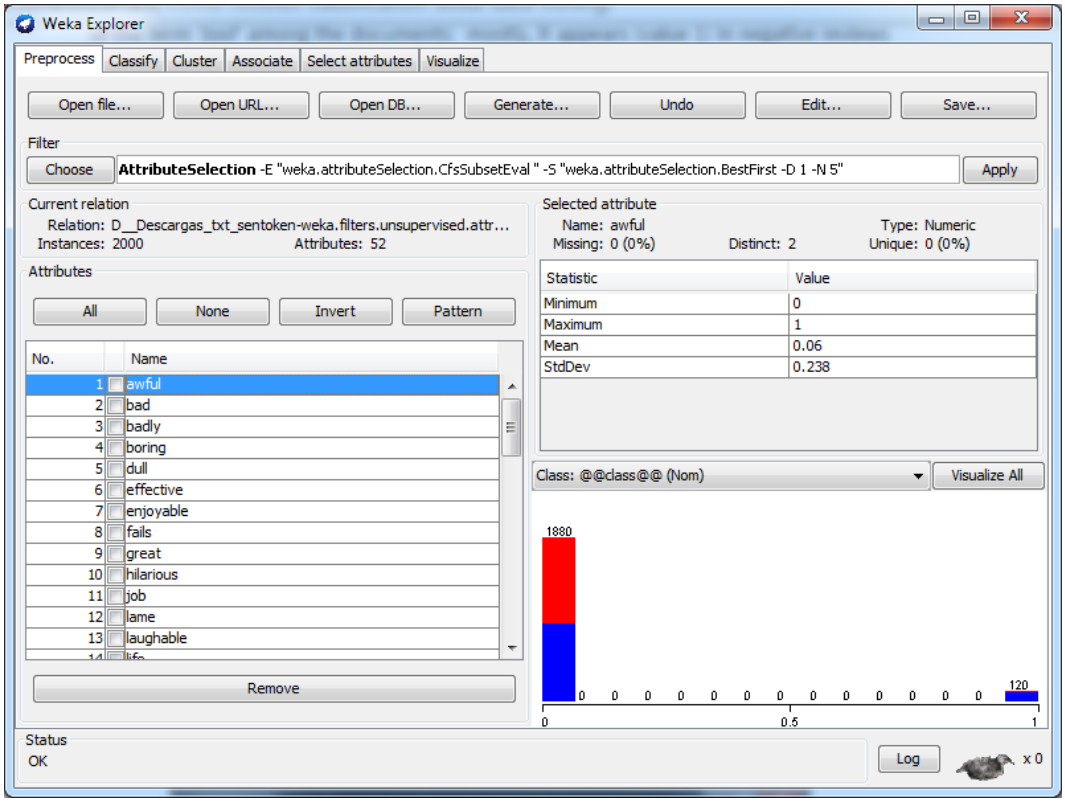


Figura 15. Creación de atributos
Elaboración: El Autor

En el caso de trabajar con un archivo .arff se realiza un reordenamiento de los datos con el fin de colocar el atributo de clase al final de todos los datos de esta manera no tener problema al momento de realizar la clasificación, para lo cual utilizaremos el filtro Reorder.

Luego podremos proceder a la clasificación de los textos, la cual es una tarea de aprendizaje supervisado en el cual se lleva a cabo la asignación de una etiqueta de clase a una lista ordenada de elementos no clasificados, esto se realiza de acuerdo a un conjunto de instancias previamente clasificadas las cuales se utilizan como el conjunto de entrenamiento.

2.4.2.1. Para analizar los resultados.

Los resultados que se tomó en cuenta para la realización del análisis del resultado de las pruebas para determinar el algoritmo con más precisión fueron:

2.4.2.1.1. Matriz de confusión.

Una matriz de confusión permite observar los errores cometidos por un clasificador (Tabla 2).

Tabla 2. Matriz de confusión.

	Clase Predicha	
Clase Real	**Jugar**	**No Jugar**
Jugar	Verdaderos Positivos	Falsos Negativos
No Jugar	Falsos Positivos	Verdaderos Negativos

Elaboración: El Autor.

Según (Corso 2009).

- VP (Verdaderos positivos): instancias correctamente reconocidas por el sistema.
- FN (Falsos negativos): instancias que son positivas y que el sistema dice que no lo son.
- FP (Falsos positivos): instancias que son negativas pero el sistema dice que no lo es.
- VN (Verdaderos negativos): instancias que son negativas y correctamente reconocidas como tales.

2.4.2.1.2. Instancias Correctamente Clasificadas (Correctly Classified Instances) e Instancias Incorrectamente Clasificadas (Incorrectly Classified Instances).

El resultado de instancias correctamente clasificadas se obtiene por la suma de la diagonal de verdaderos de la matriz de confusión y la diagonal de falsos nos presenta las instancias incorrectamente clasificadas.

- Correctly Classified Instances= VP+VN
- Incorrectly Classified Instances= FN+FP

2.4.2.1.3. Cobertura (Recall).

La cobertura mide la proporción de términos correctamente reconocidos respecto al total de términos reales, dicho de otro modo, mide en qué grado están todos los resultados (Corso 2009).

Cobertura = VP/VP+VN

2.4.2.1.4. Precisión:

La precisión, mide el número de términos correctamente reconocidos respecto al total de términos predichos, sean estos verdaderos o falsos términos. En este caso, la precisión está midiendo la pureza o el grado en que son todos los que están (Corso 2009).

Precisión = VP/VP+FP

2.5. Pruebas y Resultados.

Como se mencionó en la sección 2.1.1.1. El funcionamiento del clasificador otorga una clase a un documento, esta operación weka la realiza al momento de clasificar con los distintos algoritmos.

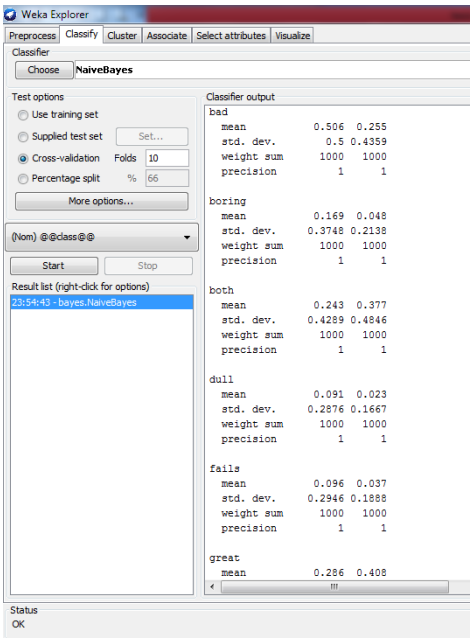


Figura 16. Creación de atributos
Elaboración: El Autor

Como podemos observar en la Figura 16 cada una de las palabras tiene ya su puntaje con respecto a cada clase, por lo que la primera columna pertenece a negativos y la segunda a positivos, con esto podemos observar que la clase que se asigna a la palabra, es la que tiene mayor puntaje como el caso de la palabra bad(malo) que es superior en la clase negativa, por lo que el programa le asigna dicha clase al momento de realizar la clasificación o la palabra great(estupendo) que posee mayor puntaje en la clase positivo. Esto sería igual a calcular las probabilidades y elegir la clase como se realizó en el apartado 2.1.1.1.

2.5.1. Pruebas utilizando cross-validation.

Se realizó el mismo procedimiento ya especificado en el apartado 2.5.1, con las opciones de test y fase de validación, en la que se utilizó Cross-validation que divide la muestra original en un numero de 10 submuestras, de estas, una se retiene como los datos de validación para probar el modelo, y la restante se utilizan como datos de entrenamiento.

Tabla 3. Resultado clasificación correcta e incorrecta del dataset en inglés utilizando Redes Bayesianas.

	NaiveBayes		**Bayesian Logistic Regression**		**Complement Naive Bayes**		**Multinomial Naive Bayes**		**BayesNet**	
	N	%	N	%	N	%	N	%	N	%
Correctly Classified Instances	1590	**79.5**	1605	**80.25**	1604	**80.2**	1604	**80.2**	1570	**78.5**
Incorrectly Classified Instances	410	**20.5**	395	**19.75**	396	**19.8**	396	**19.8**	430	**21.5**

Elaboración: El Autor

Tabla 4. Matrices de confusión dataset en inglés utilizando Redes Bayesianas.

	NaiveBayes		**Bayesian Logistic Regression**		**Complement Naive Bayes**		**Multinomial Naive Bayes**		**BayesNet**	
	A	B	A	B	A	B	A	B	A	B
A=Negativas	764	236	797	203	781	219	781	219	757	243
B=Positivas	174	826	192	808	177	823	177	823	187	813

Elaboración: El Autor

Tabla 5. Resultados Cobertura (Recall) y Precisión del dataset en inglés utilizando Redes Bayesianas.

	NaiveBayes	Bayesian Logistic Regression	Complement Naive Bayes	Multinomial Naive Bayes	BayesNet
Recall	0,795	0,803	0,802	0,802	0,785
Precisión	0,796	0,803	0,803	0,803	0,786

Elaboración: El Autor

En la primera prueba que se realizó con el dataset en inglés, como se puede observar en las Tablas 3, se obtuvo mayor número de instancias correctamente clasificadas con los algoritmos de Bayesian Logistic Regression, Complement Naive Bayes, Multinomial Naive Bayes. Esto se demuestra en la matriz de confusión (Tabla 4), en la que se muestra los verdaderos y falsos positivos, con esto podemos obtener el Recall (Tabla 5) que presenta la proporción de términos correctamente reconocidos respecto al total de términos reales; con un 0,815 del algoritmo Multinomial Naive Bayes podemos decir que se obtuvo mejor resultado en esta prueba.

Tabla 6. Resultado de la clasificación correcta e incorrecta del dataset en inglés utilizando Redes Neuronales.

	Multilayer Perceptron		Neural Network	
	N	%	N	%
Correctly Classified Instances	1503	**75.15**	1595	**79.75**
Incorrectly Classified Instances	497	**24.85**	405	**20.25**

Elaboración: El Autor

Tabla 7. Matrices de confusión dataset en inglés utilizando Redes Neuronales.

	Multilayer Perceptron		Neural Network	
	A	B	A	B
A=Negativas	787	213	822	178
B=Positivas	284	716	227	773

Elaboración: El Autor

Tabla 8. Resultados Cobertura (Recall) y Precisión del dataset en inglés utilizando Redes Neuronales.

	Multilayer Perceptron	**Neural Network**
Recall	0,752	0,798
Precisión	0,753	0,798

Elaboración: El Autor

En la prueba realizada con el dataset en inglés, utilizando redes neuronales se obtuvo mayor número de instancias correctamente clasificadas (Tabla 6) y mayor Precisión (Tabla 8) con el algoritmo de "Neurla Network". También se puede observar un resultado un poco inferior al obtenido en las pruebas con Redes Bayesianas, esto se puede observar en la matriz de confusión (Tabla 7) en la que se muestran los verdaderos y falsos positivos.

Tabla 9. Resultado clasificación de las instancias correctas e incorrectas del dataset en español utilizando Redes Bayesianas.

	NaiveBayes		**Bayesian Logistic Regression**		**Complement Naive Bayes**		**Multinomial Naive Bayes**		**BayesNet**	
	N	%	N	%	N	%	N	%	N	%
Correctly Classified Instances	318	**79.5**	334	**83.5**	345	**86.25**	345	**86.25**	214	**53.5**
Incorrectly Classified Instances	82	**20.5**	66	**16.5**	55	**13.75**	55	**13.75**	186	**46.5**

Elaboración: El Autor

Tabla 10. Matrices de confusión del dataset en español utilizando Redes Bayesianas.

	NaiveBayes		**Bayesian Logistic Regression**		**Complement Naive Bayes**		**Multinomial Naive Bayes**		**BayesNet**	
	A	B	A	B	A	B	A	B	A	B
A=Negativas	160	40	174	26	175	25	175	25	182	18
B=Positivas	42	158	40	160	30	170	30	170	168	32

Elaboración: El Autor

Tabla 11. Resultados Cobertura (Recall) y Precisión del dataset en español utilizando Redes Bayesianas.

	NaiveBayes	Bayesian Logistic Regression	Complement Naive Bayes	Multinomial Naive Bayes	BayesNet
Recall	0.795	0.835	0.863	0.863	0.535
Precision	0.795	0.837	0.863	0.863	0.58

Elaboración: El Autor

El resultado en las pruebas con el dataset en español, se obtuvo mayor instancias correctamente clasificadas (Tabla 9) con los algoritmos Complement Naive Bayes, Multinomial Naive Bayes con redes bayesianas. Podemos observar una mejoría en el algoritmo NaiveBayes en cuanto a la Cobertura (Recall) (Tabla 11).

Tabla 12. Resultado clasificación correcta e incorrecta del dataset en español utilizando Redes Neuronales.

	Multilayer Perceptron		Neural Network	
	N	%	N	%
Correctly Classified Instances	314	**78.5**	326	**81.5**
Incorrectly Classified Instances	86	**21.5**	74	**18.5**

Elaboración: El Autor

Tabla 13. Matrices de confusión del dataset en español utilizando Redes Neuronales.

	Multilayer Perceptron		Neural Network	
	A	B	A	B
A=Negativas	156	44	168	32
B=Positivas	42	158	42	158

Elaboración: El Autor

Tabla 14. Resultados Cobertura (Recall) y Precisión del dataset en español utilizando Redes Neuronales.

	Multilayer Perceptron	**Neural Network**
Recall	0,785	0,815
Precisión	0,785	0,816

Elaboración: El Autor

Con el dataset en español utilizando redes neuronales, se obtuvo mayor número de instancias correctamente clasificadas que con el algoritmo de "Neurla Network" como se muestra en la Tabla 12. De igual manera con Redes Bayesianas se ve una mejora en la clasificación. También se puede observar en la Tabla 14 que la precisión aún es inferior a la obtenida en las pruebas con Redes Bayesianas.

2.5.2. Pruebas utilizando percentage split

Se realizó el mismo procedimiento con los dataset en inglés y en español, con la diferencia que para la fase de validación se utilizó Percentage Split, con un porcentaje de 66%, que se encuentra ya por defecto en el programa; con el que aprende el modelo y la evaluación se realiza con los datos restantes.

Tabla 15. Resultado clasificación correcta e incorrecta del dataset en inglés utilizando Percentage Split con Redes Bayesianas.

	NaiveBayes		**Bayesian Logistic Regression**		**Complement Naive Bayes**		**Multinomial Naive Bayes**		**BayesNet**	
	N	%	N	%	N	%	N	%	N	%
Correctly Classified Instances	550	**80.9**	546	**80.3**	544	**80**	544	**80**	512	**75.3**
Incorrectly Classified Instances	130	**19.1**	134	**19.7**	136	**20**	136	**20**	165	**24.7**

Elaboración: El Autor

Tabla 16. Matrices de confusión dataset en inglés utilizando Percentage Split con Redes Bayesianas.

	NaiveBayes		Bayesian Logistic Regression		Complement Naive Bayes		Multinomial Naive Bayes		BayesNet	
	A	B	A	B	A	B	A	B	A	B
A=Negativas	271	68	273	66	265	74	265	74	261	78
B=Positivas	62	279	66	273	62	279	62	279	90	251

Elaboración: El Autor

Tabla 17. Resultados Cobertura (Recall) y Precisión dataset en inglés utilizando Percentage Split con Redes Bayesianas.

	NaiveBayes	Bayesian Logistic Regression	Complement Naive Bayes	Multinomial Naive Bayes	BayesNet
Recall	0,809	0,803	0,8	0,8	0,753
Precision	0,809	0,803	0,8	0,8	0,753

Elaboración: El Autor

Podemos observar que el número de instancias se ha reducido, esto se debe a que sólo se trabajó con 680 es decir un 44% de las 2000 que se tenía anteriormente, ya que como se mencionó el 66% fue utilizado por la opción Percentage Split, para que aprenda el modelo. Lo mismo va a suceder con las instancias en español.

Se obtuvo mayor número de instancias correctamente clasificadas (Tabla 15), con el algoritmo de NaiveBayes. Al parecer cuando es menor el número de instancias se aplica de una mejor manera este algoritmo, ya que aumenta su Cobertura (Recall) (Tabla 17).

Tabla 18. Resultado clasificación correcta e incorrecta del dataset en inglés utilizando Percentage Split con Redes Neuronales.

	Multilayer Perceptron		Neural Network	
	N	%	N	%
Correctly Classified Instances	503	**73.97**	541	**79.56**
Incorrectly Classified Instances	177	**26.03**	139	**20.44**

Elaboración: El Autor

Tabla 19. Matrices de confusión dataset en inglés utilizando Percentage Split con Redes Neuronales.

	Multilayer Perceptron		Neural Network	
	A	B	A	B
A=Negativas	265	74	265	74
B=Positivas	103	238	65	276

Elaboración: El Autor

Tabla 20. Resultados Cobertura (Recall) y Precisión dataset en inglés utilizando Percentage Split con Redes Neuronales.

	Multilayer Perceptron	Neural Network
Recall	0,740	0,796
Precisión	0,742	0,796

Elaboración: El Autor

Se obtuvo mayor número de instancias correctamente clasificadas (Tabla 18), y mayor Precisión (Tabla 20) con el algoritmo de Neural Network utilizando Percentage Split con 66% de instancias que fue utilizado para que aprenda el modelo. Neural Network sigue siendo el mejor en redes neuronales, pero sigue siendo inferior en instancias correctamente clasificadas comparando con algunas redes bayesianas.

Tabla 21. Resultado clasificación correcta e incorrecta del dataset en español utilizando Percentage Split con Redes Bayesianas.

	NaiveBayes		Bayesian Logistic Regression		Complement Naive Bayes		Multinomial Naive Bayes		BayesNet	
	N	%	N	%	N	%	N	%	N	%
Correctly Classified Instances	107	**78.7**	109	**80.25**	121	**88.97**	121	**88.97**	67	**49.3**
Incorrectly Classified Instances	29	**21.3**	27	**19.85**	15	**11.03**	15	**11.03**	69	**50.7**

Elaboración: El Autor

Tabla 22. Matrices de confusión dataset en español utilizando Percentage Split con Redes Bayesianas.

	NaiveBayes		Bayesian Logistic Regression		Complement Naive Bayes		Multinomial Naive Bayes		BayesNet	
	A	B	A	B	A	B	A	B	A	B
A=Negativas	57	10	59	8	61	6	61	6	67	0
B=Positivas	19	50	19	50	9	60	9	60	69	0

Elaboración: El Autor

Tabla 23. Resultados Cobertura (Recall) y Precisión dataset en español utilizando Percentage Split con Redes Bayesianas.

	NaiveBayes	Bayesian Logistic Regression	Complement Naive Bayes	Multinomial Naive Bayes	BayesNet
Recall	0,787	0,801	0,89	0,89	0,493
Precisión	0,792	0,81	0,891	0,891	0,243

Elaboración: El Autor

Se obtuvo mayor número de instancias correctamente clasificadas (Tabla 21) y mayor Precisión (Tabla 23) con los algoritmos Complement Naive Bayes y Multinomial Naive Bayes.

Tabla 24. Resultado clasificación correcta e incorrecta del dataset en español utilizando Percentage Split con Redes Neuronales.

	Multilayer Perceptron		Neural Network	
	N	%	N	%
Correctly Classified Instances	107	**78.68**	107	**78.68**
Incorrectly Classified Instances	29	**21.32**	29	**21.32**

Elaboración: El Autor

Tabla 25. Matrices de confusión dataset en español utilizando Percentage Split con Redes Neuronales.

	Multilayer Perceptron		Neural Network	
	A	B	A	B
A=Negativas	60	7	60	7
B=Positivas	22	47	22	47

Elaboración: El Autor

Tabla 26. Resultados Cobertura (Recall) y Precisión dataset en español utilizando Percentage Split con Redes Neuronales.

	Multilayer Perceptron	Neural Network
Recall	0,787	0,787
Precisión	0,802	0,802

Elaboración: El Autor

En la prueba con dataset en español utilizando Percentage Splitn en redes neuronales, se obtuvo el mismo resultado en Multilayer Perceptron y Neural Network, tanto en instancias correctamente clasificadas y Precisión como se observa en la Tabla 24 y 26, pero el resultado es más bajo que utilizando redes bayesianas.

2.5.3. Resultados Pruebas

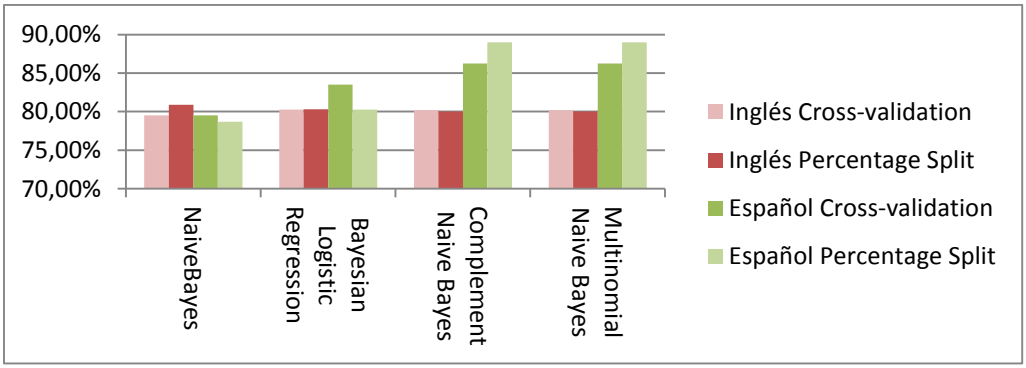


Gráfico 1. Resultado de las instancias correctamente clasificadas con cada uno de los algoritmos utilizando Redes Bayesianas
Elaboración: El Autor.

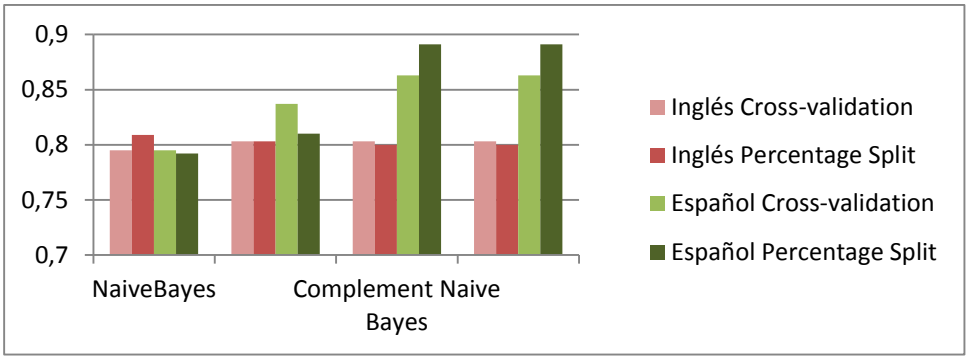


Gráfico 2. Resultado obtenido en la Cobertura (Recall) con cada uno de los algoritmos utilizando Redes Bayesianas.
Elaboración: El Autor.

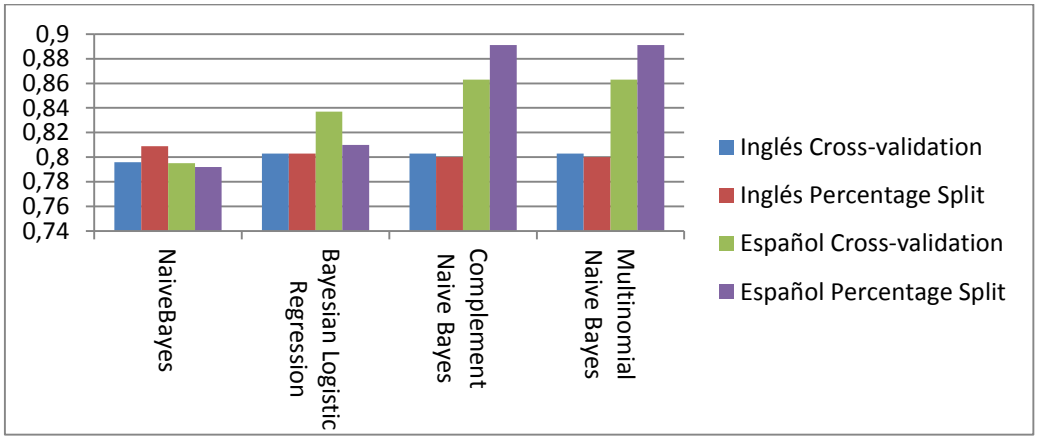


Gráfico 3. Resultado obtenido en la Precisión con cada uno de los algoritmos utilizando Redes Bayesianas.
Elaboración: **El Autor.**

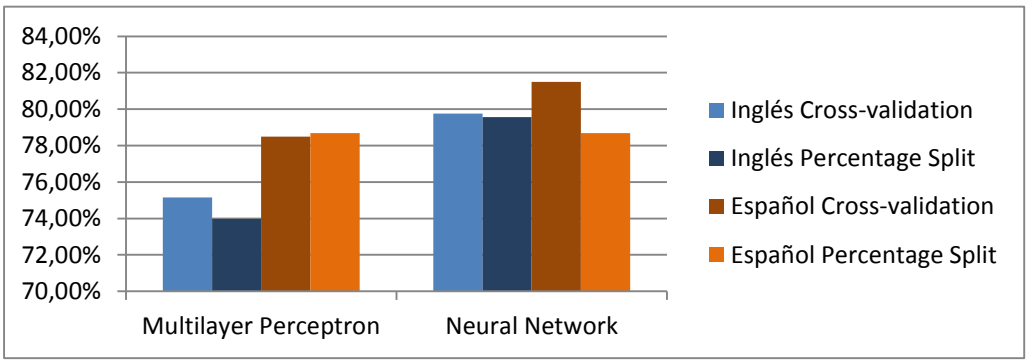


Gráfico 4. Resultado de las instancias correctamente clasificadas con cada uno de los algoritmos utilizando Redes Neuronales.
Elaboración: El Autor.

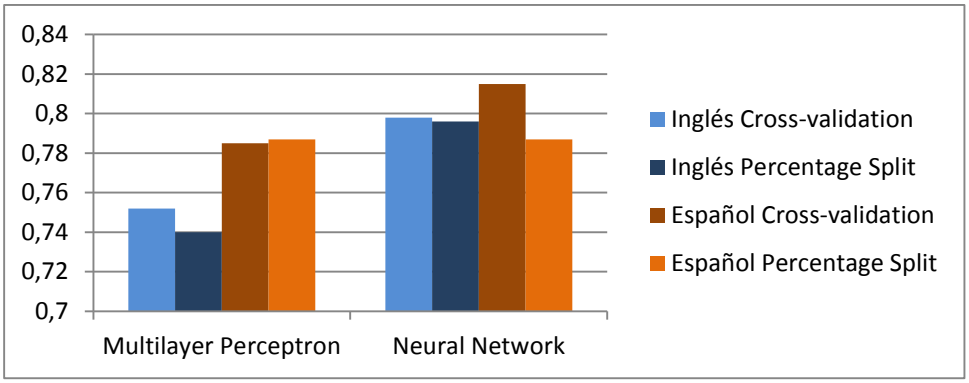


Gráfico 5. Resultado obtenidos en la Cobertura (Recall) con cada uno de los algoritmos utilizando Redes Neuronales.
Elaboración: El Autor.

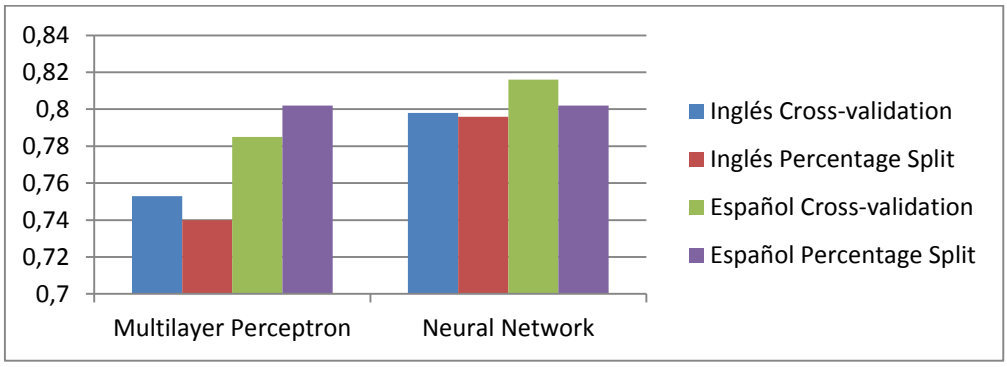


Gráfico 6. Resultado obtenidos en la Precisión con cada uno de los algoritmos utilizando Redes Neuronales.
Elaboración: El Autor

Utilizando Redes Bayesianas podemos determinar a Multinomilan Naive Bayes como el algoritmo que obtuvo mayor número de instancias correctamente clasificadas como, se observa en el Grafico 1, así como, un alto número de Cobertura (Recall) y Precisión que se presentan en el Grafico 2 y en el Grafico 3, este resultado se presenta de igual manera con el algoritmo Complement Naive Bayes. Se seleccionó el algoritmo Multinomial Naive Bayes para realizar las clasificaciones utilizando la validación por Percentage Split.

Utilizando Redes neuronales, seleccionamos el algoritmo Neural Network por ser el que mayores instancias correctamente clasificadas obtuvo, como se puede observar en el Grafico 4, la validación con mayor puntaje fue Cross-validation. De igual manera obtuvo mayor Precisión (Grafico 5) y mayor Cobertura (Grafico 6)

Una vez obtenido estos resultados, se comparó entre las técnicas, los algoritmos y la validación y se seleccionó como técnica a utilizar a Redes Bayesianas, con validación

Percentage Split y el algoritmo Multinominal Naive Bayes por ser los que mayor puntaje obtuvieron en las diferentes pruebas realizadas tanto en el dataset en inglés y en español.

2.6. Técnica seleccionada.

2.6.1. Redes bayesanias.

Se encontró algunos trabajos relacionados para la clasificación de texto con Redes Bayesianas y herramientas para la clasificación de palabras. Las redes bayesianas son consideradas como clasificadores bayesianos que proporcionan una función que clasifica un dato, en una o diferentes clases predeterminadas. Los clasificadores bayesianos son ampliamente utilizados debido a que presentan ciertas ventajas:

- Generalmente, son fáciles de construir y entender.
- Las respuestas de estos clasificadores son extremadamente rápidas, requiriendo solo un paso para hacerlo.
- Es muy robusto considerando atributos irrelevantes.
- Toma evidencia de muchos atributos para realizar la predicción final. (Sucar,2006)

Esta técnica es conocida por la creación de modelos sencillos, pero que realizan bien su trabajo sobre todo en los campos de la clasificación de documentos; así como: en predicción, diagnóstico, etc. También podemos mencionar a la presentación de Stanford sobre la clasificación de texto con Redes Bayesianas, en las cuales se utiliza el método de red bayesiana multinominal con el objetivo de encontrar la mejor clase para la clasificación de texto.

2.6.1.1. Funcionamiento

- **Probabilidad de una clase:**

$$\hat{P}(c) = \frac{N_c}{N}$$

Es igual al número de documentos con esa clase sobre el número completo de documentos.

- **Probabilidad de una palabra dada una clase:**

$$\hat{P}(w \mid c) = \frac{count(w,c) + 1}{count(c) + |V|}$$

Es igual al conteo de las veces que aparece la palabra en los documentos de la clase más uno, dividido para el conteo de todas las palabras en los documentos de la clase más el tamaño del vocabulario.

- Ejemplo

Tabla 27. Ejemplo funcionamiento de la Red Bayesiana

	Docs.	Palabras	Clases: F(Feliz)/T(Tristeza)
Training	1	Feliz Aburrimiento Feliz	F
	2	Feliz Feliz Ira	F
	3	Feliz Sorpresa	F
	4	Miedo Tristeza Feliz	T
Tests	5	Feliz Feliz Feliz Miedo Tristeza	?

Elaboración: El Autor

Como podemos observar en la Tabla 27, tenemos 4 documentos o instancias que se encuentras previamente clasificadas en las clases F (Feliz) o T (Tristeza), estos documentos los utilizaremos como entrenamientos para determinar la clase del último documento, para esto realizaremos las siguientes operaciones tomando en cuenta la teoría ya mencionada:

- **Probabilidad de una clase:**
  - Es igual al número de veces que aparece la clase f (feliz) o t (tristeza) sobre el número completo de documentos que utilizaremos como entrenamiento, como podemos ver f aparece 3 veces en los 4 documentos y t solo 1.

$$P(f) = \frac{3}{4}$$

$$P(t) = \frac{1}{4}$$

- **Probabilidad de una palabra dada una clase:**
  - Esta operación se realiza contando las veces que aparece una palabra en los documentos de la clase más uno, por ejemplo en la primera operación se contaron

5 veces la palabra feliz en los documentos de la clase f. Esto es dividido para el conteo de todas las palabras en los documentos de la clase, en el caso de la primera operación son 8 palabas que se encuentran en los documentos de la clase f, más el tamaño de entrenamiento que serían 6.

$$P(Feliz|f) = \frac{5+1}{8+6} = \frac{6}{14} = \frac{3}{7}$$

$$P(Miedo|f) = \frac{0+1}{8+6} = \frac{1}{14}$$

$$P(Tristeza|f) = \frac{0+1}{8+6} = \frac{1}{14}$$

$$P(Feliz|t) = \frac{1+1}{3+6} = \frac{2}{9}$$

$$P(Miedo|t) = \frac{1+1}{3+6} = \frac{2}{9}$$

$$P(Tristeza|t) = \frac{1+1}{3+6} = \frac{2}{9}$$

- Escoger una clase

Para escoger la clases se multiplica la ***probabilidad de una clase*** que se obtuvo anteriormente, en el caso de la clases f es $\frac{3}{4}$ y en el t es $\frac{1}{4}$, esto multiplicado por la ***probabilidad de una palabra dada una clase,*** en el caso de la clase f tenemos $\frac{3}{7}, \frac{1}{4}, \frac{1}{4}$, de las palabras Feliz, Miedo y Tristeza correspondientemente, esto elevado al número de veces que aparece en el documento al cual se desea clasificar (documento 5) en el caso de la palabra feliz aparece 3 veces.

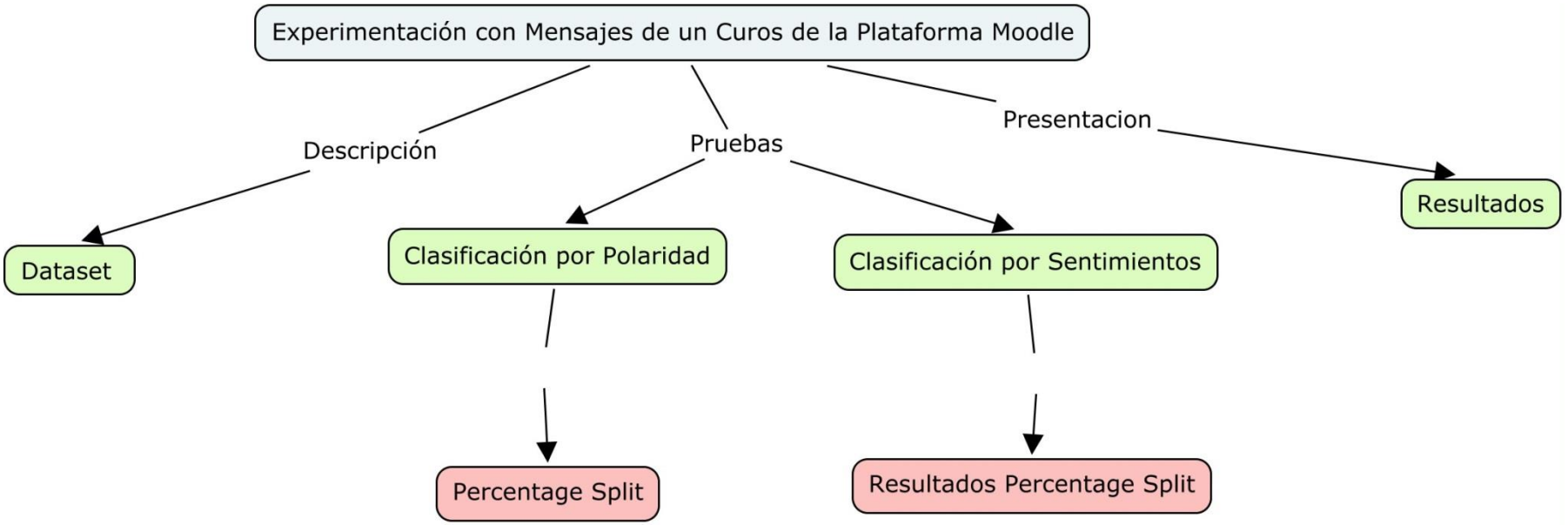
$$P(f|d5) = \frac{3}{4} * \frac{3}{7}^{3} * \frac{1}{4} * \frac{1}{4} = 0{,}0003$$

$$P(t|d5) = \frac{1}{4} * \frac{2}{9}^{3} * \frac{2}{9} * \frac{2}{9} = 0{,}0001$$

Una vez realizada esta operación podemos observar que la mayor probabilidad de clase es 0,0003 y equivale a la clase feliz por lo tanto el documento 5 es clasificado como feliz por obtener mayor resultado sobre las demás clases.

3. Experimentación con mensajes de un curso de la plataforma MOODLE

3.1. Mapa Conceptual



En este capítulo se realizan las pruebas para la clasificación de mensajes de un curso de la plataforma MOODLE, utilizando mensajes de dos cursos de la modalidad a distancia, los cuales se detallan posteriormente, los mismo que se clasifican por polaridad, obteniendo mensajes tanto negativos como positivos, y por sentimientos que son: simpatía, angustia, frustración, aburrimiento, ansiedad y confusión. Para este propósito se utilizó la herramienta, la técnica, el algoritmo y la validación previamente seleccionados en el Capítulo 2.

3.2. DataSet y Algoritmos

Para realizar las pruebas se utilizó los datos de la plataforma MOODLE de la Universidad Técnica Particular de Loja, se obtuvo dos tipos de dataset, el primero un archivo .arff de un seminario de la modalidad a distancia proporcionado por el proyecto de tesis "Adaptación de una herramienta de procesamiento de lenguaje natural para el etiquetado de sentimientos y el análisis de lenguaje en español". El cual realiza el procesamiento del lenguaje natural para la obtención del etiquetado de los sentimientos (simpatía, angustia, frustración, aburrimiento, ansiedad, confusión) con el que se realizó la clasificación previamente seleccionada la herramienta, técnica y algoritmos. De esta manera se pudo obtener el sentimiento que representa este grupo de mensajes. El archivo posee dos atributos, el primero el atributo "texto" que contiene todas las palabras y el segundo el atributo clase que posee los sentimientos de el atributo texto con un total de 230 instancias.

El segundo dataset se realizó de una clase de Estructura de Datos de la modalidad a distancia de la Universidad Técnica Particular de Loja, se trabajó con la clasificación Hand-coded rules (Reglas de codificación manual) etiquetado según su significado en trabajos anteriormente realizados por expertos, para la clasificación por polaridad se utilizó un total de 202 mensajes distribuidos en las clases positivas y negativas y se realizó los procedimientos del capítulo 2. Este dataset posee la misma estructura del primero, con la diferencia que el atributo "texto" posee lo mensajes completos y no solo las palabras que poseen un sentimiento.

3.3. Clasificación por Polaridad

Con todas las pruebas realizadas y teniendo en cuenta que se seleccionó como técnica a utilizar a Redes Bayesianas, con validación Percentage Split y el algoritmo Multinominal Naive Bayes por ser los que mayor puntaje obtuvieron en las diferentes pruebas

realizadas. Se procedió a realizar la clasificación por polaridad, que se refiere a las frases negativas o positivas utilizando el dataset del curso de la plataforma MOODLE descrito en el apartado 3.1.

3.3.1. Percentage Split

Tabla 28. Resultado clasificación utilizando Percentage Split correcta e incorrecta de las pruebas reales.

	Multinomial Naive Bayes	
	N	%
Correctly Classified Instances	50	**72,5**
Incorrectly Classified Instances	19	**27.5**

Elaboración: El Autor

Tabla 29. Matriz de Confusión de las pruebas reales utilizando Percentage Split.

	Multinomial Naive Bayes	
	A	B
A=Negativas	2	19
B=Positivas	0	48

Elaboración: El Autor

Tabla 30. Resultados Recall y Precisión de las pruebas reales utilizando Percentage Split

	Multinomial Naive Bayes
Recall	0,725
Precisión	0,803

Elaboración: El Autor

3.3. Clasificación por Sentimientos

Se realizó la clasificación por sentimientos en el que se utilizó el dataset con los sentimientos de (simpatía, angustia, frustración, aburrimiento, ansiedad, confusión).

3.3.1. Percentage Split

Tabla 31. Resultado de la clasificación correcta e incorrecta de sentimientos utilizando Percentage Split en las pruebas reales.

	Multinomial Naive Bayes	
	N	%
Correctly Classified Instances	55	**70.5128**
Incorrectly Classified Instances	23	**29.4872**

Elaboración: El Autor

Tabla 32. Matriz de Confusión de las pruebas de clasificación de sentimientos utilizando Cross-Validation.

	Multinomial Naive Bayes					
	A	B	C	D	E	F
A=simpatía	44	0	0	0	0	0
B=angustia	15	7	0	0	0	0
C=frustración	2	0	4	0	0	0
D=aburrimiento	1	0	0	0	0	0
E=ansiedad	3	0	0	0	0	0
F=confusión	2	0	0	0	0	0

Elaboración: El Autor

Tabla 33. Resultados Recall y Precisión de las pruebas reales utilizando Percentage Split

	Multinomial Naive Bayes
Recall	0,705
Precisión	0,729

Elaboración: El Autor

3.4. Resultados

- Como resultado de clasificación por polaridad (Tabla 28), podemos observar un total de 72,5% de instancias correctamente clasificadas, y en la matriz de confusión (Tabla 29) tenemos 2 verdaderos positivos de la clase “negativas” mientras que tenemos 48 verdaderos negativos en la clase “Positivas”.

  Hay 21 elementos clasificados en la clase “Negativas”:

  - 2 de estos elementos están correctamente clasificados en la clase “Negativas”.
  - 19 de estos elementos están incorrectamente clasificados en la clase “Negativas”.

  Hay 48 elementos clasificados en la clase “Positivas”:

  - 0 elementos están incorrectamente clasificados en la clase “Positivas”.
  - 48 elementos están correctamente clasificados en la clase “Positivas”.

- En el resultado de la clasificación por sentimientos (Tabla 31) tenemos un total de 70,5% de instancias correctamente clasificadas, y en la matriz de confusión (Tabla 32) tenemos 44 verdaderos positivos de la clase “Simpatía” que es el número más alto de elementos correctamente clasificados.

  Hay 44 elementos clasificados en la clase “Simpatía”:

  - 44 de estos elementos están correctamente clasificados en la clase “Simpatía”.
  - 0 de estos elementos están incorrectamente clasificados en la clase “Simpatía”.

  Hay 22 elementos clasificados en la clase “Angustia”:

- 7 elementos están correctamente clasificados en la clase “Angustia”.
- 15 elementos están incorrectamente clasificados en la clase “Angustia”.

Hay 22 elementos clasificados en la clase “Frustración”:

- 4 elementos están correctamente clasificados en la clase “Frustración”.
- 2 elementos están incorrectamente clasificados en la clase “Frustración”.

Como se puede observar en las clases “Aburrimiento”, “Ansiedad” y “Confusión”, no se obtuvieron valores en las diagonal de verdaderos teniendo 0 elementos correctamente clasificados.

En el curso de Estructura de Datos de la modalidad a distancia tenemos un total de 48 elementos clasificados positivos (reflejan sentimientos positivos) y 21 elementos clasificados negativos (reflejan sentimientos negativos).

En cuanto a la clasificación por sentimiento en el seminario de la modalidad a distancia tenemos 44 mensajes clasificados en la clase “Simpatía”, 22 mensajes clasificados en la clase “Angustia” y 22 mensajes clasificados en la clase “Frustración”.

CONCLUSIONES

Al terminar el presente trabajo podemos llegar a las siguientes conclusiones:

- En el Capítulo 1 se buscó algoritmos de Redes Neuronales y Redes Bayesianas, se describe cada uno de los algoritmos y otros conceptos necesarios para realizar el presente trabajo.
- En el Capítulo 2 se realizaron las pruebas de los algoritmos en las cuales se determinó Multinomial Naive Bayes**,** se utilizó dos dataset, el uno formado por 2000 archivos y el segundo dataset formado por 400 archivos en español.
- Se seleccionó la herramienta weka para las pruebas, debido a que está desarrollado en Java y permite realizar la clasificación utilizando Redes Neuronales reglas y Redes Bayesiona con muchos tipos de dataset.
- Se probaron técnicas, herramientas y algoritmos para la clasificación de texto, de las cuales se seleccionó como técnica a utilizar Redes Bayesianas, con validación Percentage Split y el algoritmo Multinominal Naive Bayes, por ser los que mayor puntaje obtuvieron en las diferentes pruebas realizadas tanto en el dataset en inglés y en español.
- Se obtuvo en las pruebas sobre clasificación de polaridad un 72,5% de instancias correctamente clasificadas, y en la matriz de confusión 2 verdaderos positivos de la clase “negativas” mientras que tenemos 48 verdaderos negativos en la clase “Positivas”.
- Se realizó las pruebas sobre clasificación de sentimientos, obteniendo un total de 70,5% de instancias correctamente clasificadas, y en la matriz de confusión 44 verdaderos positivos de la clase “Simpatía”, que es el número más alto de elementos correctamente clasificados.
- En el Capítulo 3, se realizaron las pruebas con mensajes de la plataforma MOODLE aplicando la herramienta, algoritmo y técnicas seleccionadas. Cuyo resultado en el curso de Estructura de Datos de la modalidad a distancia es de 48 mensajes clasificados positivos y 21 clasificados negativos. En el seminario de la modalidad a distancia tenemos 44 mensajes clasificados en la clase “Simpatía”, 22 mensajes clasificados en la clase “Angustia” y 22 mensajes clasificados en la clase “Frustración”.

RECOMENDACIONES

- Considerando que el tema de clasificación de texto es muy extenso se debe realizar un estudio muy a fondo de este tema con ayuda de un experto en psicología y lingüística.
- Basarse en trabajos previamente realizados ya que estos poseen una clara explicación de los métodos usados.
- Realizar pruebas con distintos dataset de diferentes tamaños ya que algunos resultados varían.
- Realizar un investigación más afondo en cuanto a técnicas y herramientas de aprendizaje automático para poder clasificar un texto obteniendo un 100% de precisión

Bibliografía

Arevian, G. (2007). Recurrent neural networks for robust real-world text classification. Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, IEEE Computer Society.

Bertona, L. (2005). Entrenamiento de redes neuronales basado en algoritmos evolutivos Facultad de Ingenieria. Argentina, Universidad de Buenos Aires. **Ingeniero en Informatica:** 245.

Cazala, J. (2014). Neural Networks 101.

Corso, C. (2009). "Aplicación de algoritmos de clasificación supervisada usando Weka." Córdoba: Universidad Tecnológica Nacional, Facultad Regional Córdoba.

Chen, W., et al. (2011). "An Empirical Study of Massively Parallel Bayesian Networks Learning for Sentiment Extraction from Unstructured Text."

Dubiau, L. and J. Ale (2013). Análisis de Sentimientos sobre un Corpus en Español: Experimentación con un Caso de Estudio. 14th Argentine Symposium on Artificial Intelligence, ASAI 2013.

Fu, C., et al. (2014). A Study on Recursive Neural Network Based Sentiment Classification of Sina Weibo. Trust, Security and Privacy in Computing and Communications (TrustCom), 2014 IEEE 13th International Conference on, IEEE.

Harrison, I. (1997). "Belief networks." Artificial Intelligence Applications Institute.

Hernández, J. and C. Ferri, Eds. (2006). Curso de Doctorado Extracción Automática de Conocimiento en Bases de Datos e Ingeniería del Software.

Hinton, G. (2009). Deep belief networks. Scholarpedia. Canada.

Husken, M. and P. Stagge (2003). "Recurrent Neural Networks for Time Series Classification." Neurocomputing 50 (C).

Kibriya, A., et al. (2005). Multinomial naive bayes for text categorization revisited. AI 2004: Advances in Artificial Intelligence, Springer**:** 488-499.

Komiya, K., et al. (2011). Negation Naive Bayes for Categorization of Product Pages on the Web. RANLP.

Lin, J. (2009). N-Gram Language Models. Estados Unidos, University of Maryland.

Marín, J. (2011). Introducción a las redes neuronales aplicadas.

McCallum, A. and K. Nigam (1998). A comparison of event models for naive bayes text classification. AAAI-98 workshop on learning for text categorization, Citeseer.

Muab'Dib, E., Ed. (2008). Inteligencia Artificial: Redes Neuronales.

Muñoz, J. (2010). Modelos Computacionales. España.

Norsys., C. S. (2015). "Introduction to Bayes Nets." from http://www.norsys.com/tutorials/netica/secA/tut_A1.htm.

Ortigosa, J., et al. (2011). "Approaching Sentiment Analysis by Using Semi-supervised Learning of Multi-dimensional Classifiers."

Rivera, M. (2011). "El papel de las redes bayesianas en la toma de decisiones." La Simulación al servicio de la academia **2**.

Stanford-University. Text Classification and Naïve Bayes.

Storkey, A. (2005). "Introduction to bielef networks."

Sucar, L. "Redes Bayesianas." 28.

Zhang, Q., et al. (2003). "Artificial Neural Networks for RF and Microwave Design- From Theory to Practice." IEEE.

ANEXOS

Anexo 1.- Tablas de Resultados de todas las pruebas

Tabla 34. Instancias correctamente clasificadas con cada uno de los algoritmos de Redes Bayesianas en las pruebas.

	NaiveBayes	Bayesian Logistic Regression	Complement Naive Bayes	Multinomial Naive Bayes
Inglés Cross-validation	79,50%	80,25%	80,20%	80,20%
Inglés Percentage Split	80,90%	80,30%	80%	80%
Español Cross-validation	79,50%	83,50%	86,25%	86,25%
Español Percentage Split	78,70%	80,25%	88,97%	88,97%

Elaboración: El Autor

Tabla 35. Cobertura (Recall) de cada uno de los algoritmos de Redes Bayesianas en las pruebas.

	NaiveBayes	Bayesian Logistic Regression	Complement Naive Bayes	Multinomial Naive Bayes
Inglés Cross-validation	0,795	0,803	0,803	0,803
Inglés Percentage Split	0,809	0,803	0,8	0,8
Español Cross-validation	0,795	0,837	0,863	0,863
Español Percentage Split	0,792	0,81	0,891	0,891

Elaboración: El Autor

Tabla 36. Instancias correctamente clasificadas con cada uno de los algoritmos de Redes Neuronales en las pruebas.

	Multilayer Perceptron	Neural Network
Inglés Cross-validation	75,15%	79,75%
Inglés Percentage Split	73,97%	79,56%
Español Cross-validation	78,50%	81,50%
Español Percentage Split	78,68%	78,68%

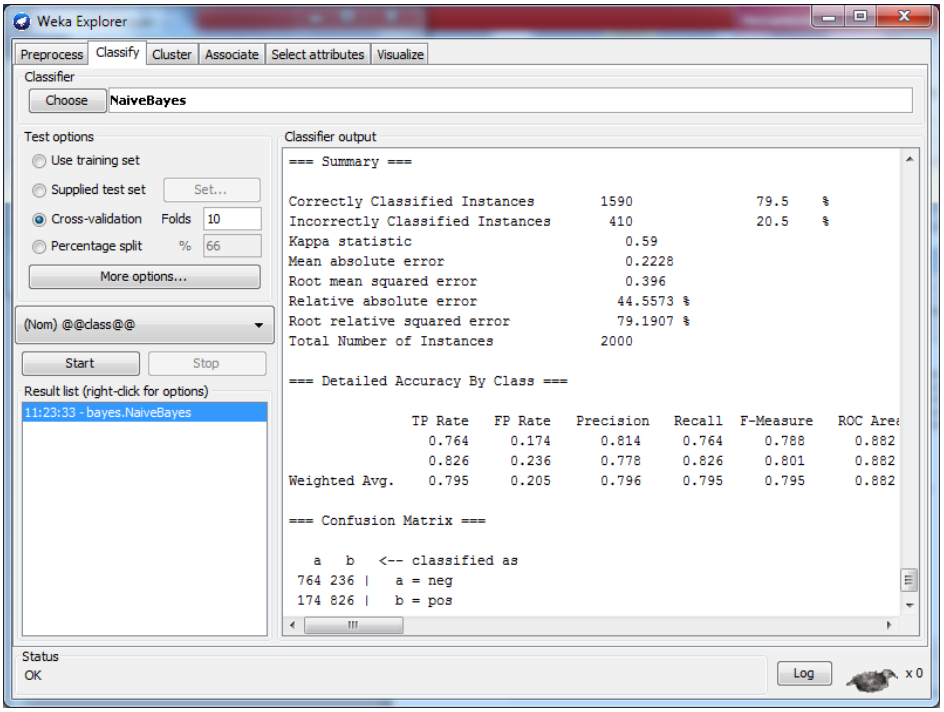
Elaboración: El Autor

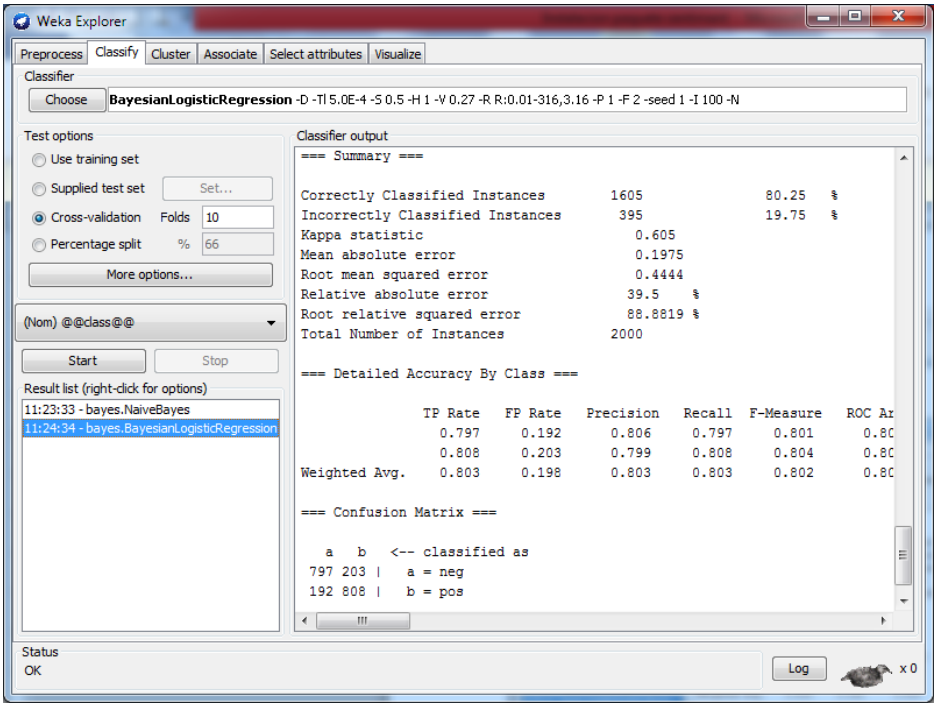
Tabla 37. Cobertura (Recall) de cada uno de los algoritmos de Redes Bayesianas en las pruebas.

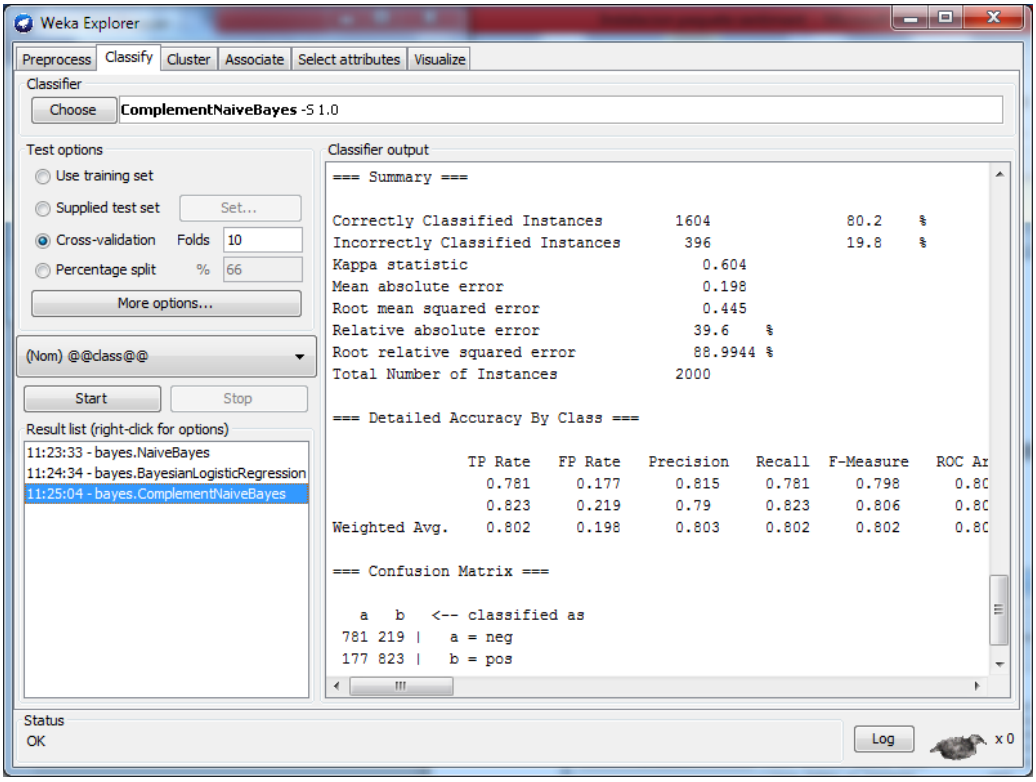
	Multilayer Perceptron	Neural Network
Inglés Cross-validation	0,752	0,798
Inglés Percentage Split	0,74	0,796
Español Cross-validation	0,785	0,815
Español Percentage Split	0,787	0,787

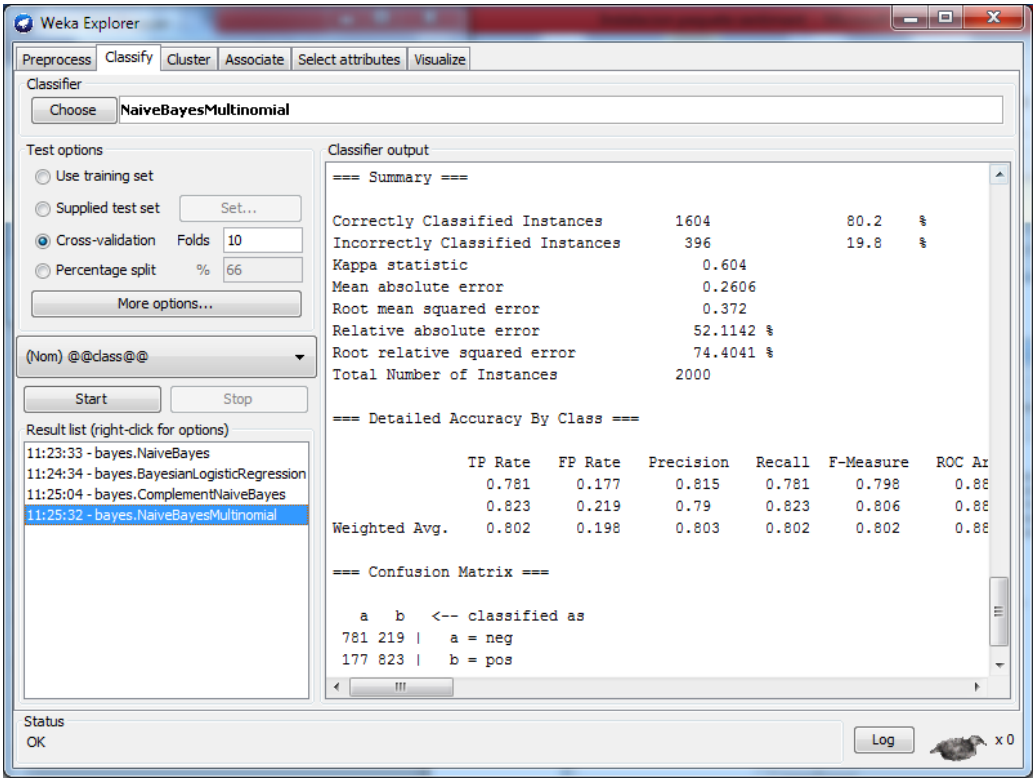
Elaboración: El Autor

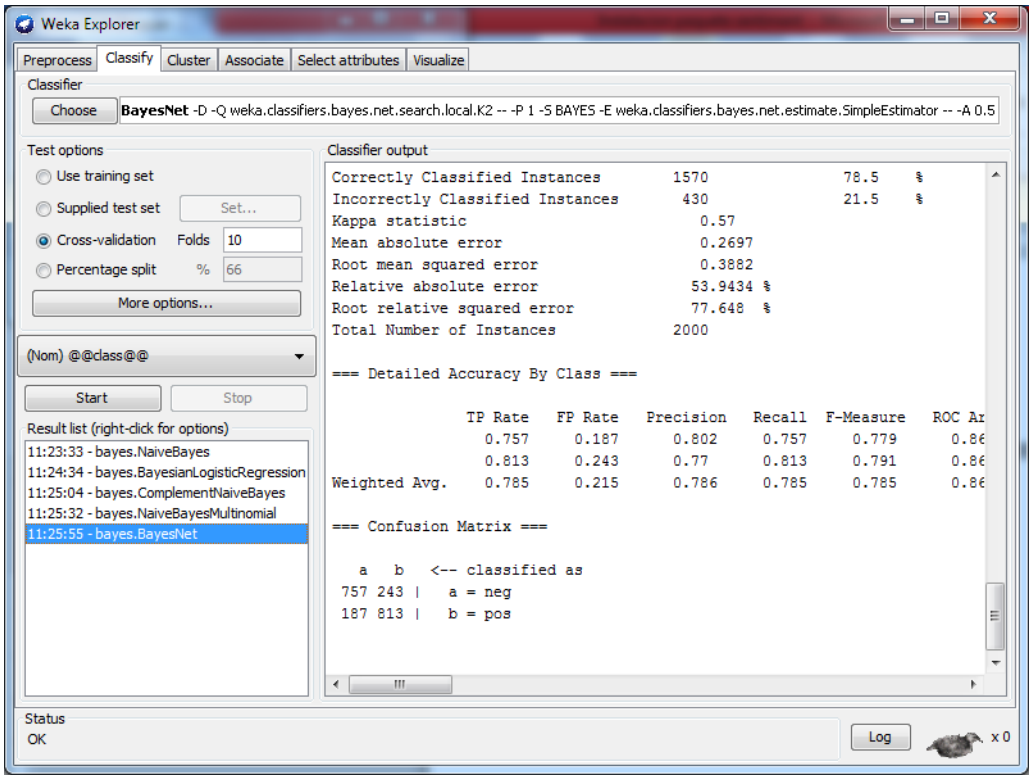
Anexo 2.- Pruebas Utilizando Dataset en Inglés con archivos separados y Cross-validation



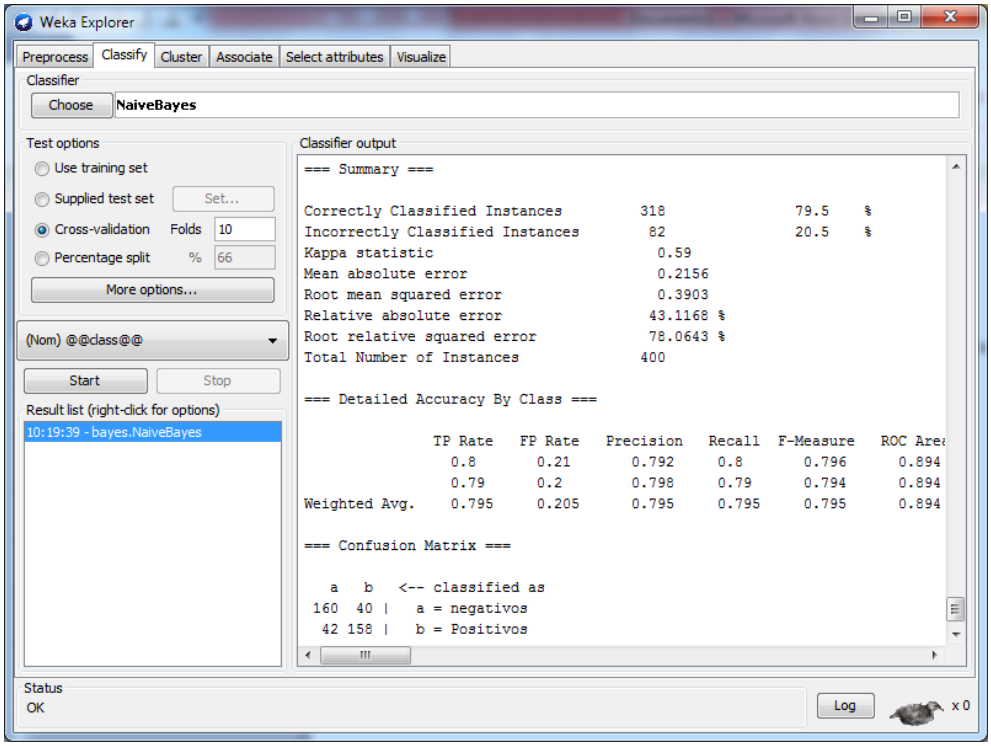

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
BayesianLogisticRegression -D -Tl 5.0E-4 -S 0.5 -H 1 -V 0.27 -R R:0.01-316,3.16 -P 1 -F 2 -seed 1 -I 100 -N
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
11:23:33 - bayes.NaiveBayes
11:24:34 - bayes.BayesianLogisticRegression
Classifier output
=== Summary ===
Correctly Classified Instances 1605 80.25 %
Incorrectly Classified Instances 395 19.75 %
Kappa statistic 0.605
Mean absolute error 0.1975
Root mean squared error 0.4444
Relative absolute error 39.5 %
Root relative squared error 88.8819 %
Total Number of Instances 2000
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.797 0.192 0.806 0.797 0.801 0.80
0.808 0.203 0.799 0.808 0.804 0.80
Weighted Avg. 0.803 0.198 0.803 0.803 0.802 0.80
=== Confusion Matrix ===
a b <-- classified as
797 203 | a = neg
192 808 | b = pos
Status
OK
Log
x 0

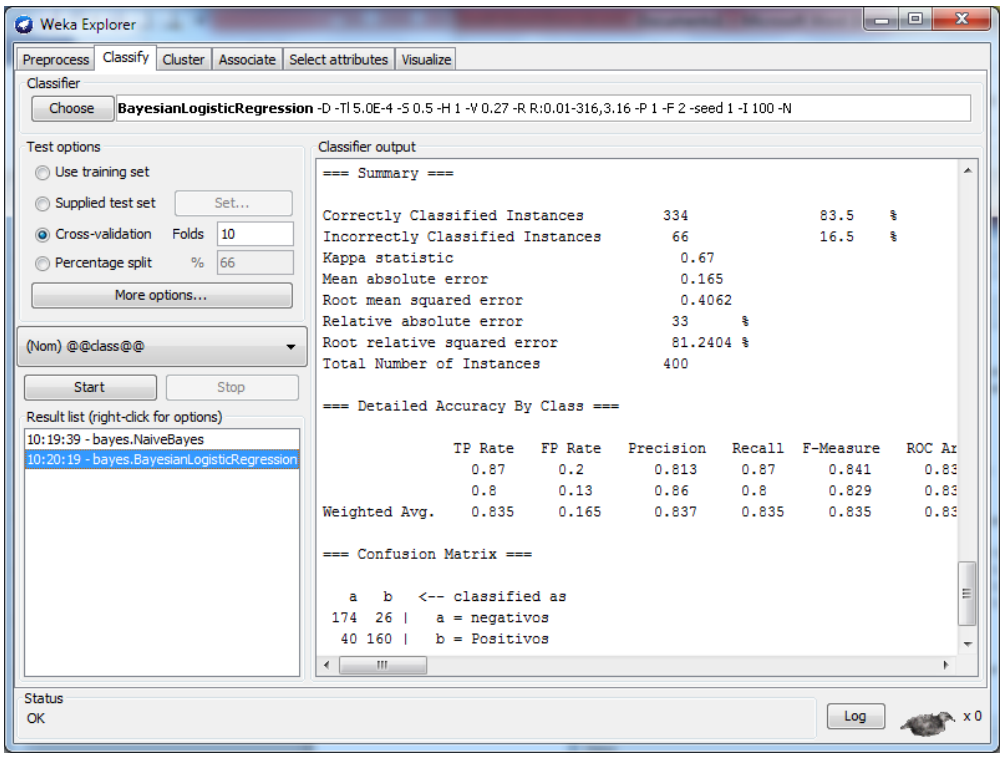

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
ComplementNaiveBayes -S 1.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
11:23:33 - bayes.NaiveBayes
11:24:34 - bayes.BayesianLogisticRegression
11:25:04 - bayes.ComplementNaiveBayes
Classifier output
=== Summary ===
Correctly Classified Instances 1604 80.2 %
Incorrectly Classified Instances 396 19.8 %
Kappa statistic 0.604
Mean absolute error 0.198
Root mean squared error 0.445
Relative absolute error 39.6 %
Root relative squared error 88.9944 %
Total Number of Instances 2000
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.781 0.177 0.815 0.781 0.798 0.80
0.823 0.219 0.79 0.823 0.806 0.80
Weighted Avg. 0.802 0.198 0.803 0.802 0.802 0.80
=== Confusion Matrix ===
a b <-- classified as
781 219 | a = neg
177 823 | b = pos
Status
OK
Log
x 0

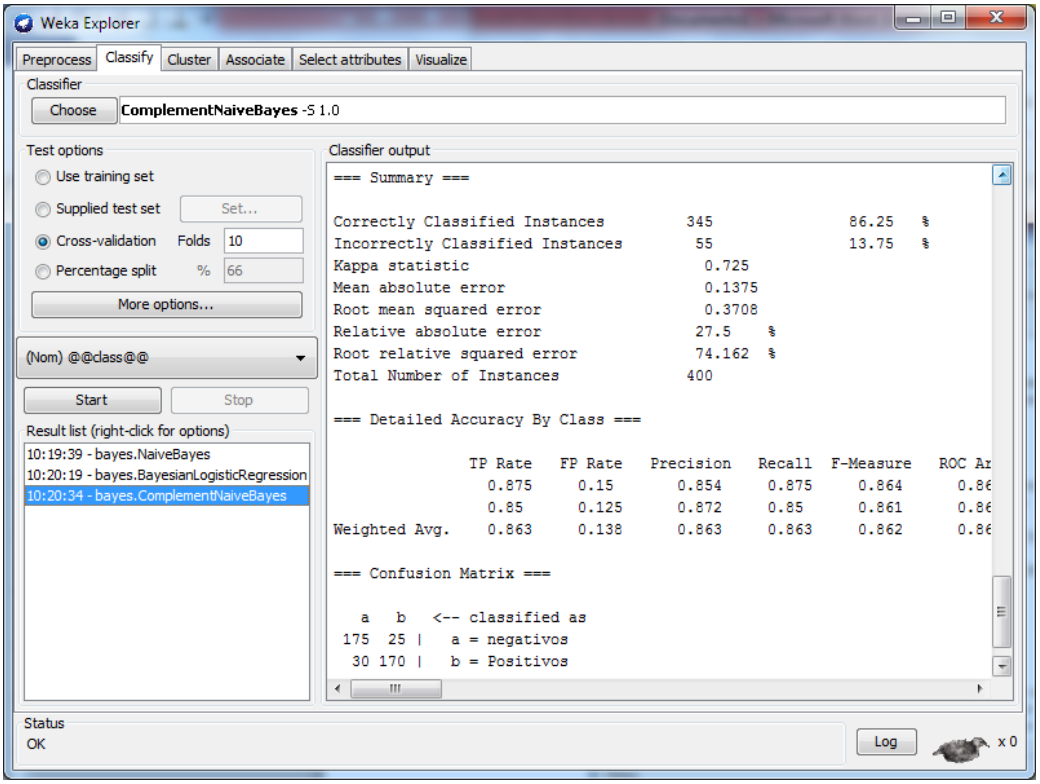

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayesMultinomial
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
11:23:33 - bayes.NaiveBayes
11:24:34 - bayes.BayesianLogisticRegression
11:25:04 - bayes.ComplementNaiveBayes
11:25:32 - bayes.NaiveBayesMultinomial
Classifier output
=== Summary ===
Correctly Classified Instances 1604 80.2 %
Incorrectly Classified Instances 396 19.8 %
Kappa statistic 0.604
Mean absolute error 0.2606
Root mean squared error 0.372
Relative absolute error 52.1142 %
Root relative squared error 74.4041 %
Total Number of Instances 2000
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.781 0.177 0.815 0.781 0.798 0.88
0.823 0.219 0.79 0.823 0.806 0.88
Weighted Avg. 0.802 0.198 0.803 0.802 0.802 0.88
=== Confusion Matrix ===
a b <-- classified as
781 219 | a = neg
177 823 | b = pos
Status
OK
Log
x 0

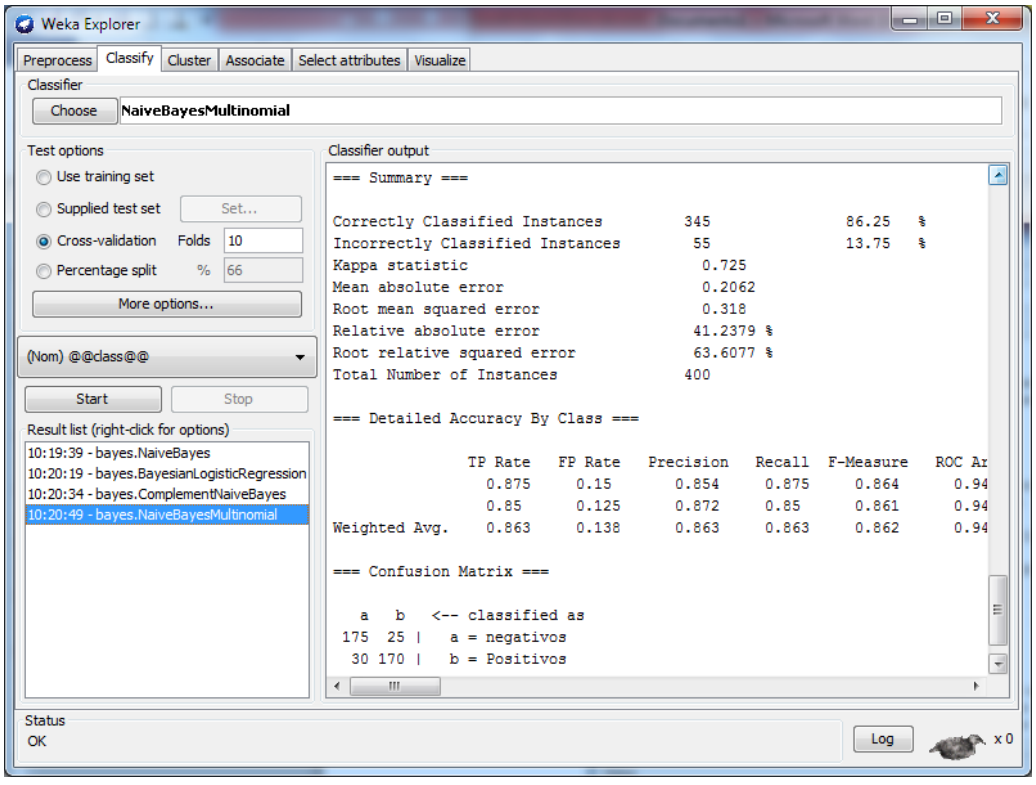

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
BayesNet -D -Q weka.classifiers.bayes.net.search.local.K2 -- -P 1 -S BAYES -E weka.classifiers.bayes.net.estimate.SimpleEstimator -- -A 0.5
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
11:23:33 - bayes.NaiveBayes
11:24:34 - bayes.BayesianLogisticRegression
11:25:04 - bayes.ComplementNaiveBayes
11:25:32 - bayes.NaiveBayesMultinomial
11:25:55 - bayes.BayesNet
Classifier output
Correctly Classified Instances 1570 78.5 %
Incorrectly Classified Instances 430 21.5 %
Kappa statistic 0.57
Mean absolute error 0.2697
Root mean squared error 0.3882
Relative absolute error 53.9434 %
Root relative squared error 77.648 %
Total Number of Instances 2000
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.757 0.187 0.802 0.757 0.779 0.86
0.813 0.243 0.77 0.813 0.791 0.86
Weighted Avg. 0.785 0.215 0.786 0.785 0.785 0.86
=== Confusion Matrix ===
a b <-- classified as
757 243 | a = neg
187 813 | b = pos
Status
OK
Log
x 0

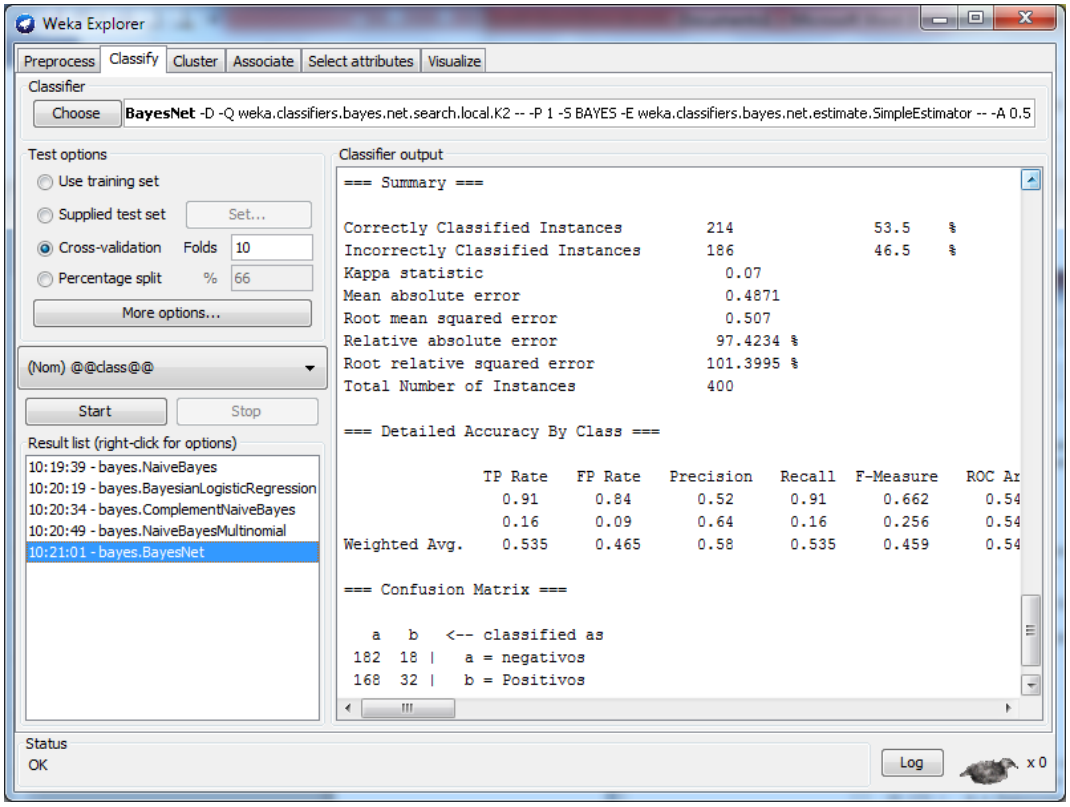
Anexo 2.- Pruebas Utilizando Dataset en español con archivos separados y Cross-validation



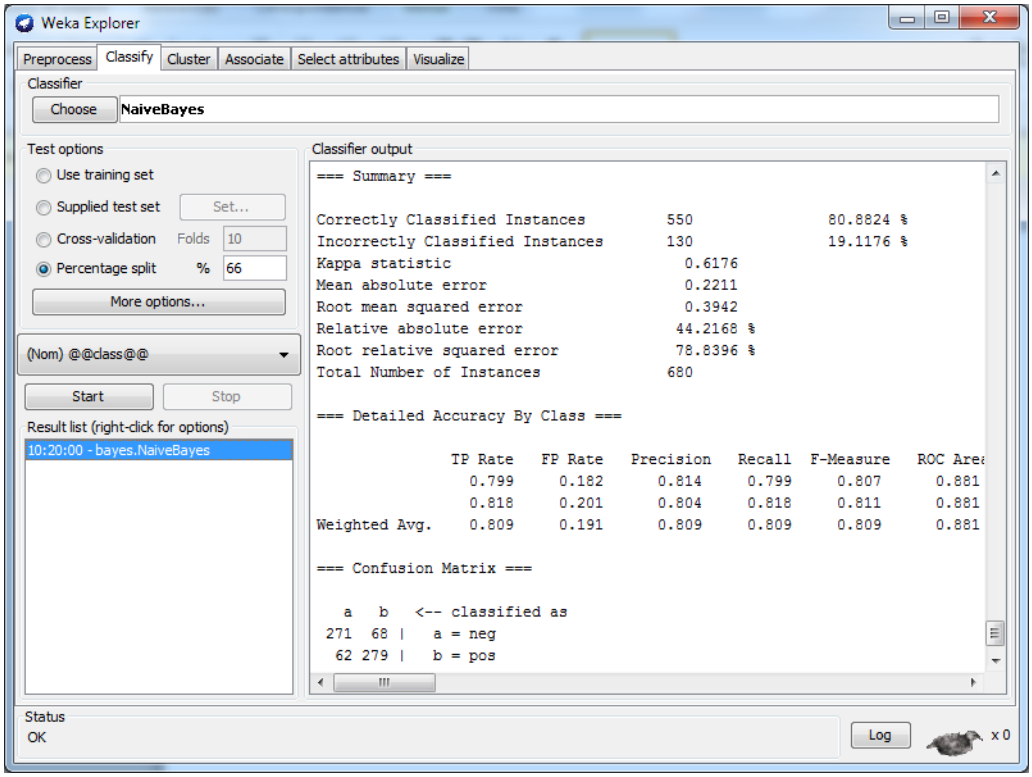


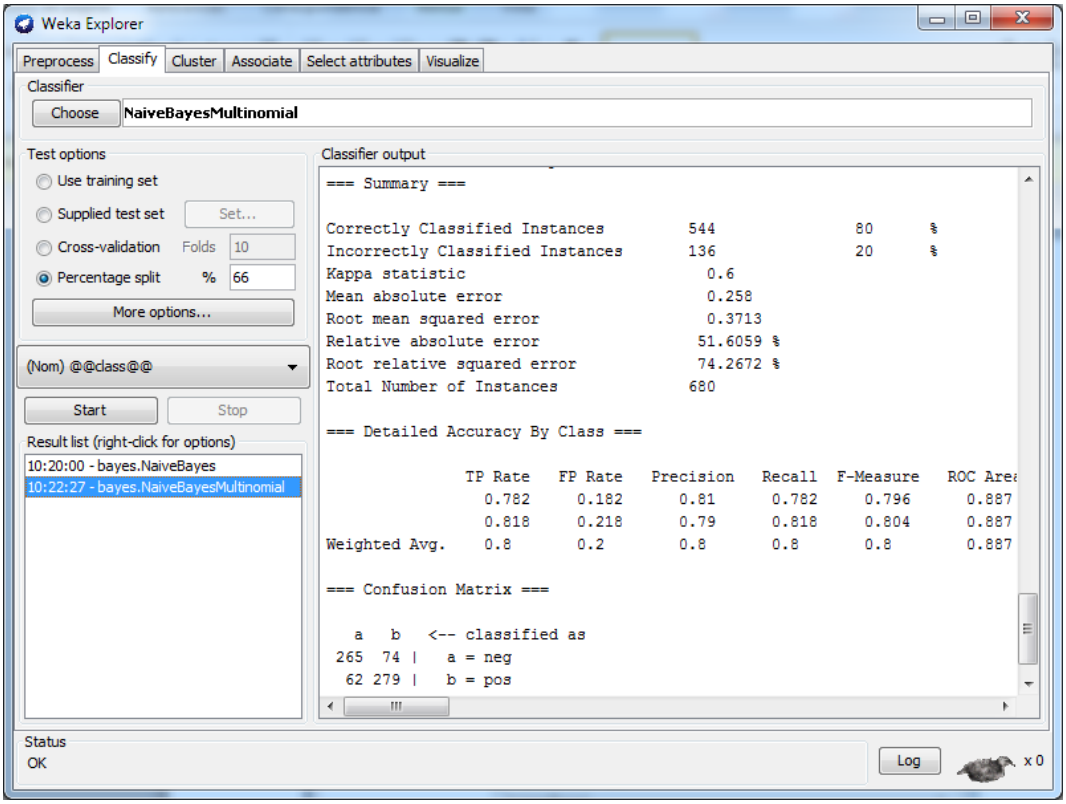

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
ComplementNaiveBayes -S 1.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
10:19:39 - bayes.NaiveBayes
10:20:19 - bayes.BayesianLogisticRegression
10:20:34 - bayes.ComplementNaiveBayes
Classifier output
=== Summary ===
Correctly Classified Instances 345 86.25 %
Incorrectly Classified Instances 55 13.75 %
Kappa statistic 0.725
Mean absolute error 0.1375
Root mean squared error 0.3708
Relative absolute error 27.5 %
Root relative squared error 74.162 %
Total Number of Instances 400
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.875 0.15 0.854 0.875 0.864 0.86
0.85 0.125 0.872 0.85 0.861 0.86
Weighted Avg. 0.863 0.138 0.863 0.863 0.862 0.86
=== Confusion Matrix ===
a b <-- classified as
175 25 | a = negativos
30 170 | b = Positivos
Status
OK
Log
x 0

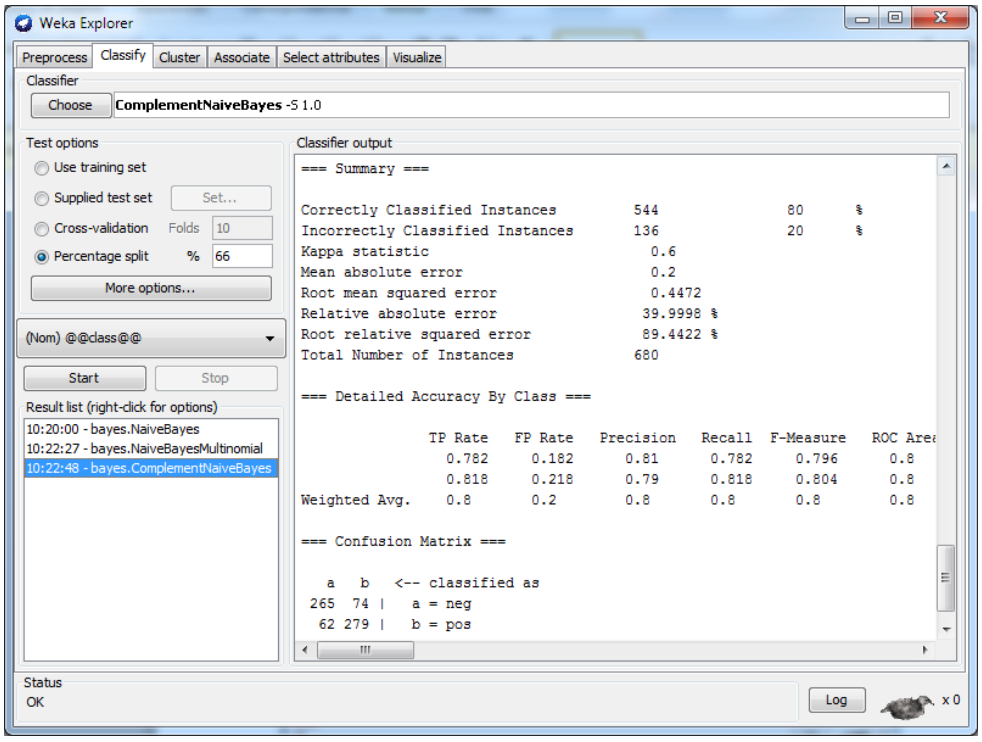

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayesMultinomial
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
10:19:39 - bayes.NaiveBayes
10:20:19 - bayes.BayesianLogisticRegression
10:20:34 - bayes.ComplementNaiveBayes
10:20:49 - bayes.NaiveBayesMultinomial
Classifier output
=== Summary ===
Correctly Classified Instances 345 86.25 %
Incorrectly Classified Instances 55 13.75 %
Kappa statistic 0.725
Mean absolute error 0.2062
Root mean squared error 0.318
Relative absolute error 41.2379 %
Root relative squared error 63.6077 %
Total Number of Instances 400
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.875 0.15 0.854 0.875 0.864 0.94
0.85 0.125 0.872 0.85 0.861 0.94
Weighted Avg. 0.863 0.138 0.863 0.863 0.862 0.94
=== Confusion Matrix ===
a b <-- classified as
175 25 | a = negativos
30 170 | b = Positivos
Status
OK
Log
x 0

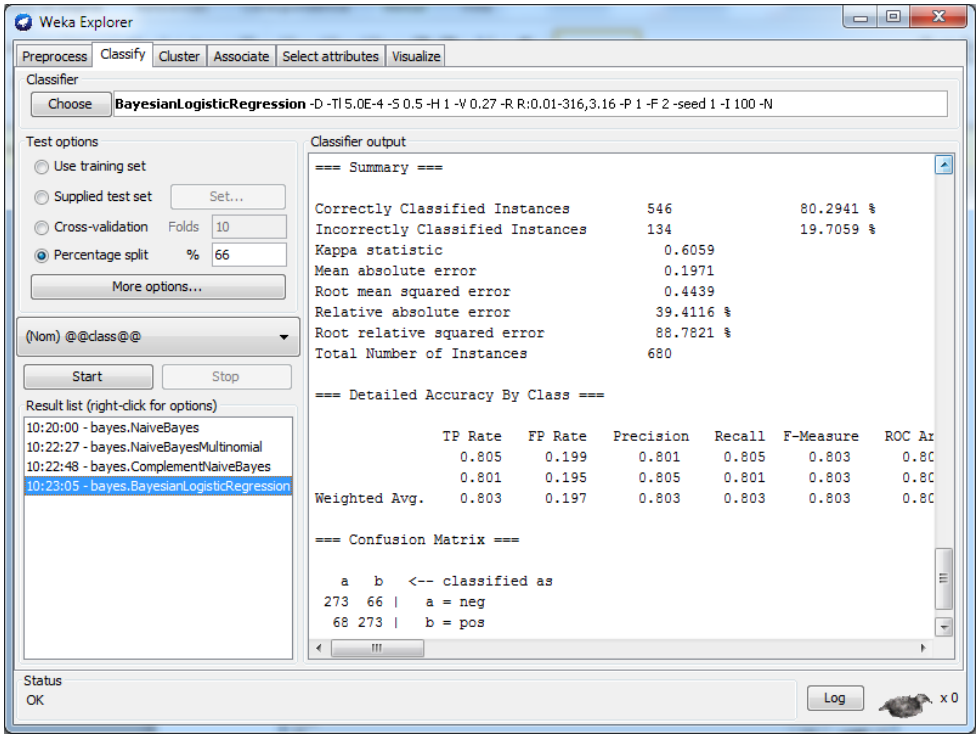


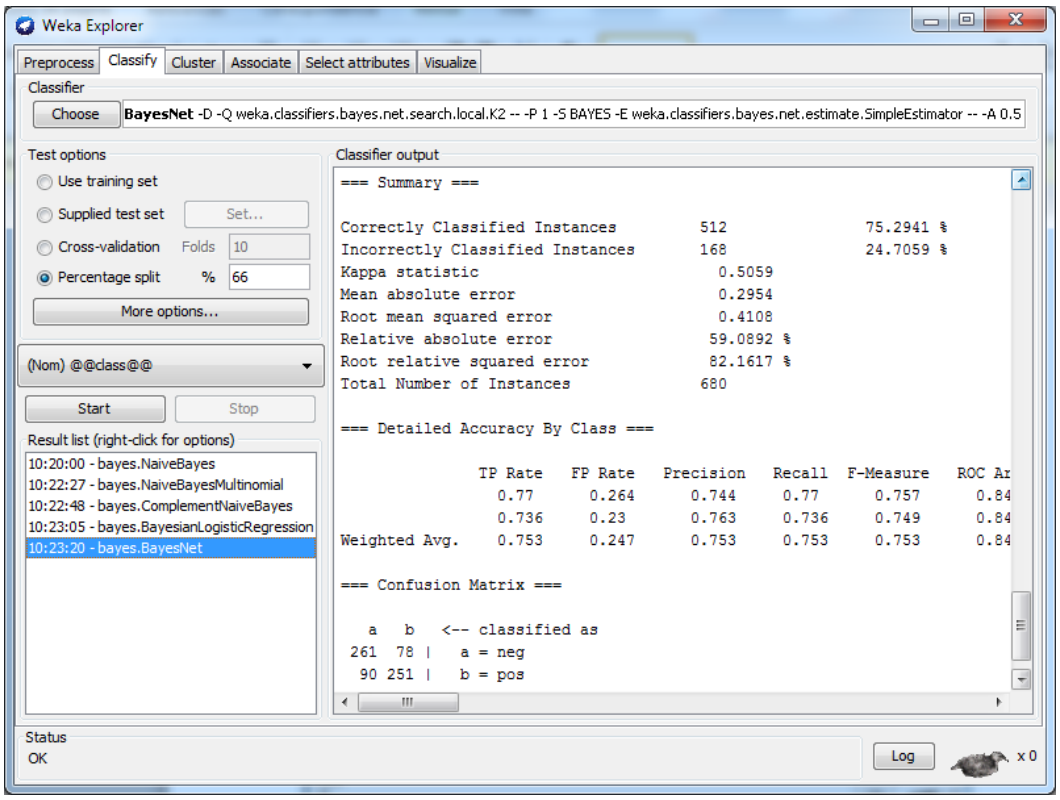
Anexo 3.- Pruebas Utilizando Dataset en Inglés con archivos separados y Percentage split.



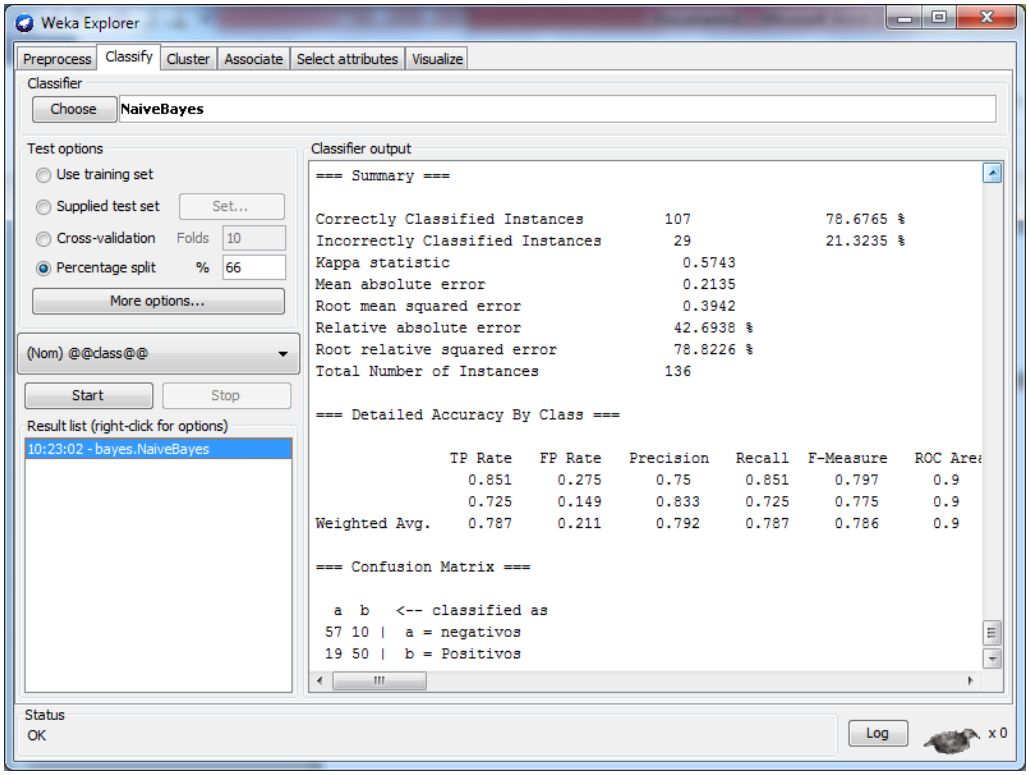

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayesMultinomial
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
10:20:00 - bayes.NaiveBayes
10:22:27 - bayes.NaiveBayesMultinomial
Classifier output
=== Summary ===
Correctly Classified Instances 544 80 %
Incorrectly Classified Instances 136 20 %
Kappa statistic 0.6
Mean absolute error 0.258
Root mean squared error 0.3713
Relative absolute error 51.6059 %
Root relative squared error 74.2672 %
Total Number of Instances 680
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area
0.782 0.182 0.81 0.782 0.796 0.887
0.818 0.218 0.79 0.818 0.804 0.887
Weighted Avg. 0.8 0.2 0.8 0.8 0.8 0.887
=== Confusion Matrix ===
a b <-- classified as
265 74 | a = neg
62 279 | b = pos
Status
OK
Log
x 0

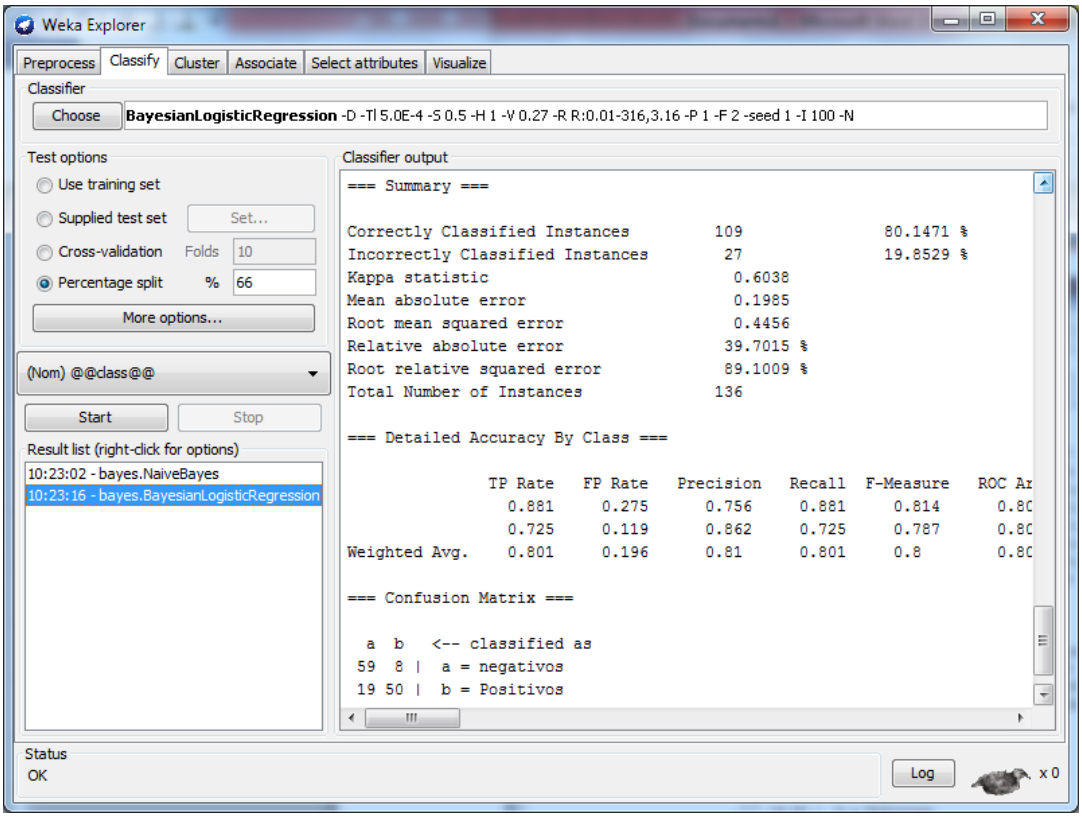

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
ComplementNaiveBayes -S 1.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
10:20:00 - bayes.NaiveBayes
10:22:27 - bayes.NaiveBayesMultinomial
10:22:48 - bayes.ComplementNaiveBayes
Classifier output
=== Summary ===
Correctly Classified Instances 544 80 %
Incorrectly Classified Instances 136 20 %
Kappa statistic 0.6
Mean absolute error 0.2
Root mean squared error 0.4472
Relative absolute error 39.9998 %
Root relative squared error 89.4422 %
Total Number of Instances 680
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area
0.782 0.182 0.81 0.782 0.796 0.8
0.818 0.218 0.79 0.818 0.804 0.8
Weighted Avg. 0.8 0.2 0.8 0.8 0.8 0.8
=== Confusion Matrix ===
a b <-- classified as
265 74 | a = neg
62 279 | b = pos
Status
OK
Log
x 0

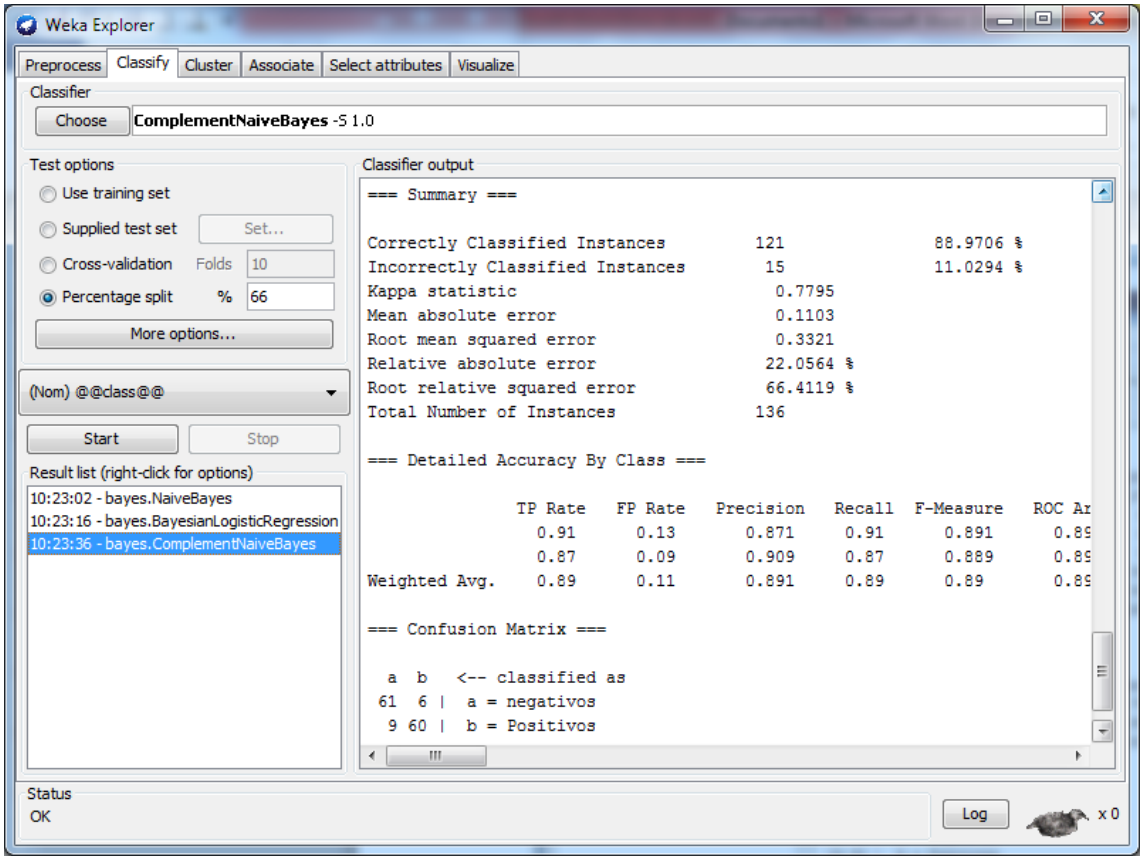

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
BayesianLogisticRegression -D -Tl 5.0E-4 -S 0.5 -H 1 -V 0.27 -R R:0.01-316,3.16 -P 1 -F 2 -seed 1 -I 100 -N
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
10:20:00 - bayes.NaiveBayes
10:22:27 - bayes.NaiveBayesMultinomial
10:22:48 - bayes.ComplementNaiveBayes
10:23:05 - bayes.BayesianLogisticRegression
Classifier output
=== Summary ===
Correctly Classified Instances 546 80.2941 %
Incorrectly Classified Instances 134 19.7059 %
Kappa statistic 0.6059
Mean absolute error 0.1971
Root mean squared error 0.4439
Relative absolute error 39.4116 %
Root relative squared error 88.7821 %
Total Number of Instances 680
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.805 0.199 0.801 0.805 0.803 0.80
0.801 0.195 0.805 0.801 0.803 0.80
Weighted Avg. 0.803 0.197 0.803 0.803 0.803 0.80
=== Confusion Matrix ===
a b <-- classified as
273 66 | a = neg
68 273 | b = pos
Status
OK
Log
x 0

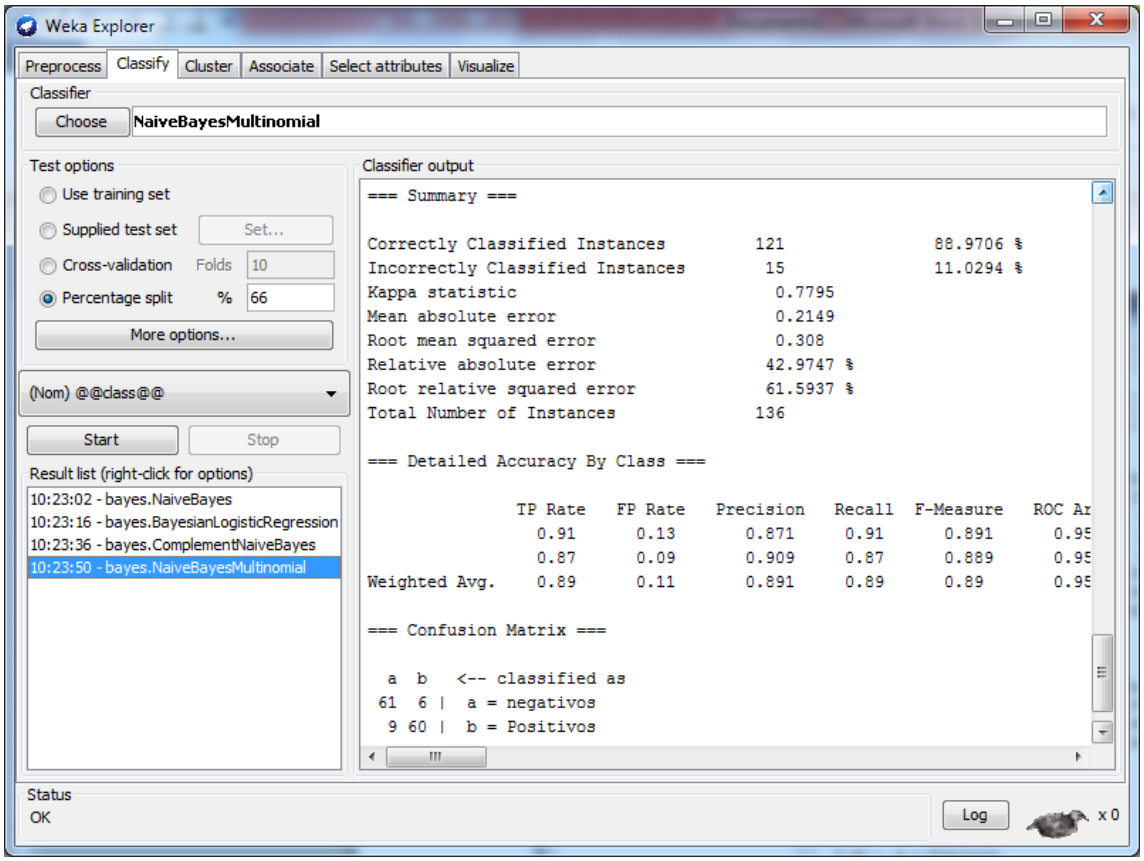

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
BayesNet -D -Q weka.classifiers.bayes.net.search.local.K2 -- -P 1 -S BAYES -E weka.classifiers.bayes.net.estimate.SimpleEstimator -- -A 0.5
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
10:20:00 - bayes.NaiveBayes
10:22:27 - bayes.NaiveBayesMultinomial
10:22:48 - bayes.ComplementNaiveBayes
10:23:05 - bayes.BayesianLogisticRegression
10:23:20 - bayes.BayesNet
Classifier output
=== Summary ===
Correctly Classified Instances 512 75.2941 %
Incorrectly Classified Instances 168 24.7059 %
Kappa statistic 0.5059
Mean absolute error 0.2954
Root mean squared error 0.4108
Relative absolute error 59.0892 %
Root relative squared error 82.1617 %
Total Number of Instances 680
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.77 0.264 0.744 0.77 0.757 0.84
0.736 0.23 0.763 0.736 0.749 0.84
Weighted Avg. 0.753 0.247 0.753 0.753 0.753 0.84
=== Confusion Matrix ===
a b <-- classified as
261 78 | a = neg
90 251 | b = pos
Status
OK
Log
x 0

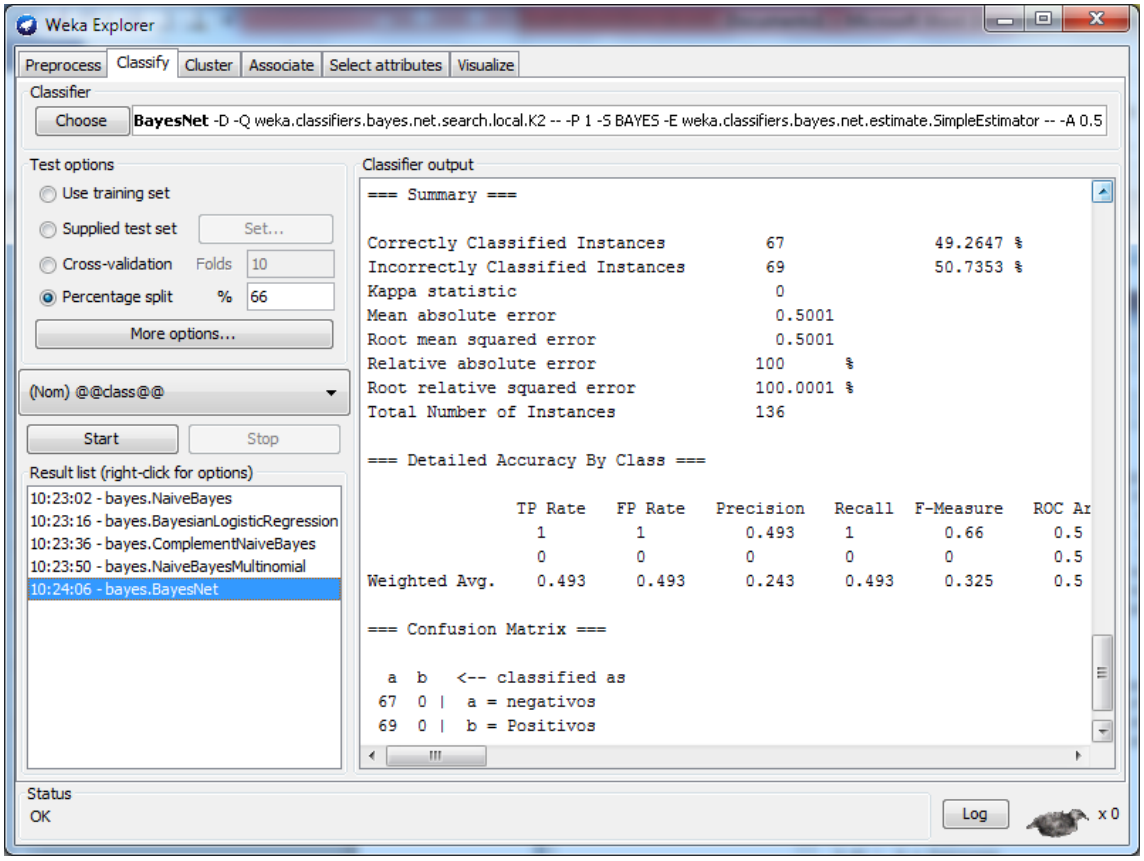
Anexo 4.- Pruebas Utilizando Dataset en español con archivos separados y Percentage split.



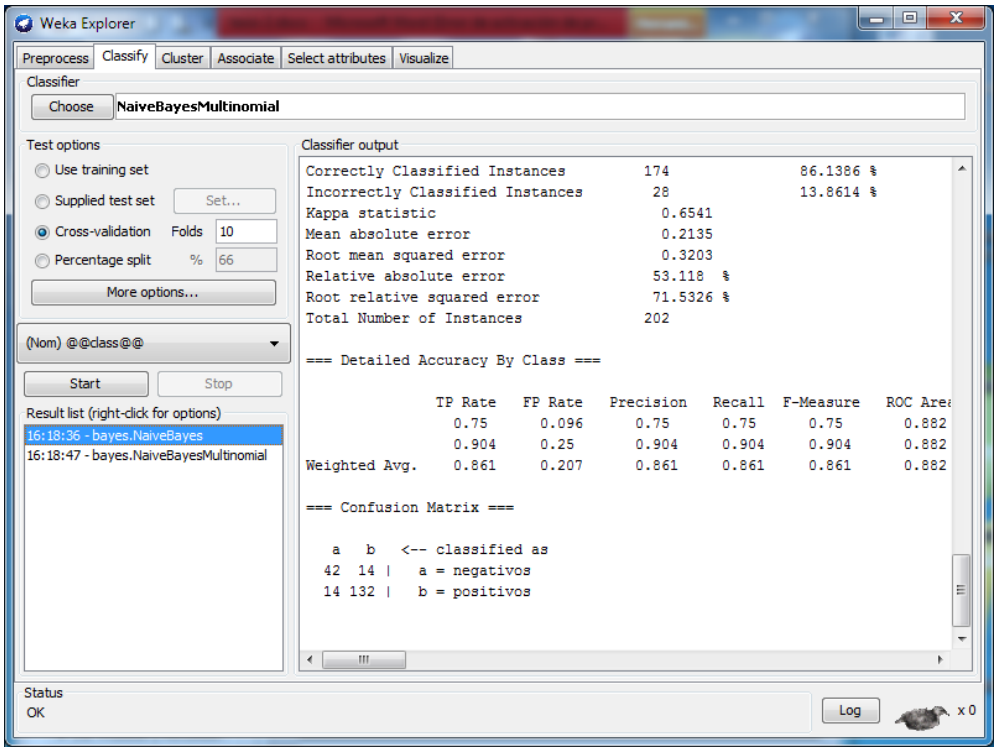


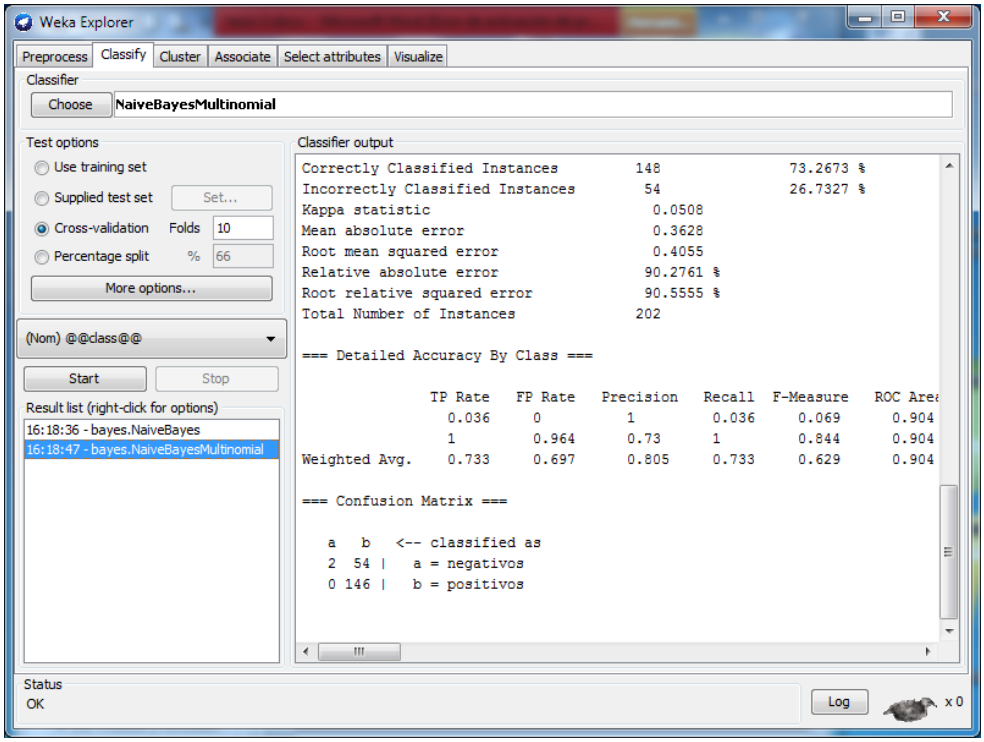

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
ComplementNaiveBayes -S 1.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
10:23:02 - bayes.NaiveBayes
10:23:16 - bayes.BayesianLogisticRegression
10:23:36 - bayes.ComplementNaiveBayes
Classifier output
=== Summary ===
Correctly Classified Instances 121 88.9706 %
Incorrectly Classified Instances 15 11.0294 %
Kappa statistic 0.7795
Mean absolute error 0.1103
Root mean squared error 0.3321
Relative absolute error 22.0564 %
Root relative squared error 66.4119 %
Total Number of Instances 136
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.91 0.13 0.871 0.91 0.891 0.89
0.87 0.09 0.909 0.87 0.889 0.89
Weighted Avg. 0.89 0.11 0.891 0.89 0.89 0.89
=== Confusion Matrix ===
a b <-- classified as
61 6 | a = negativos
9 60 | b = Positivos
Status
OK
Log
x 0

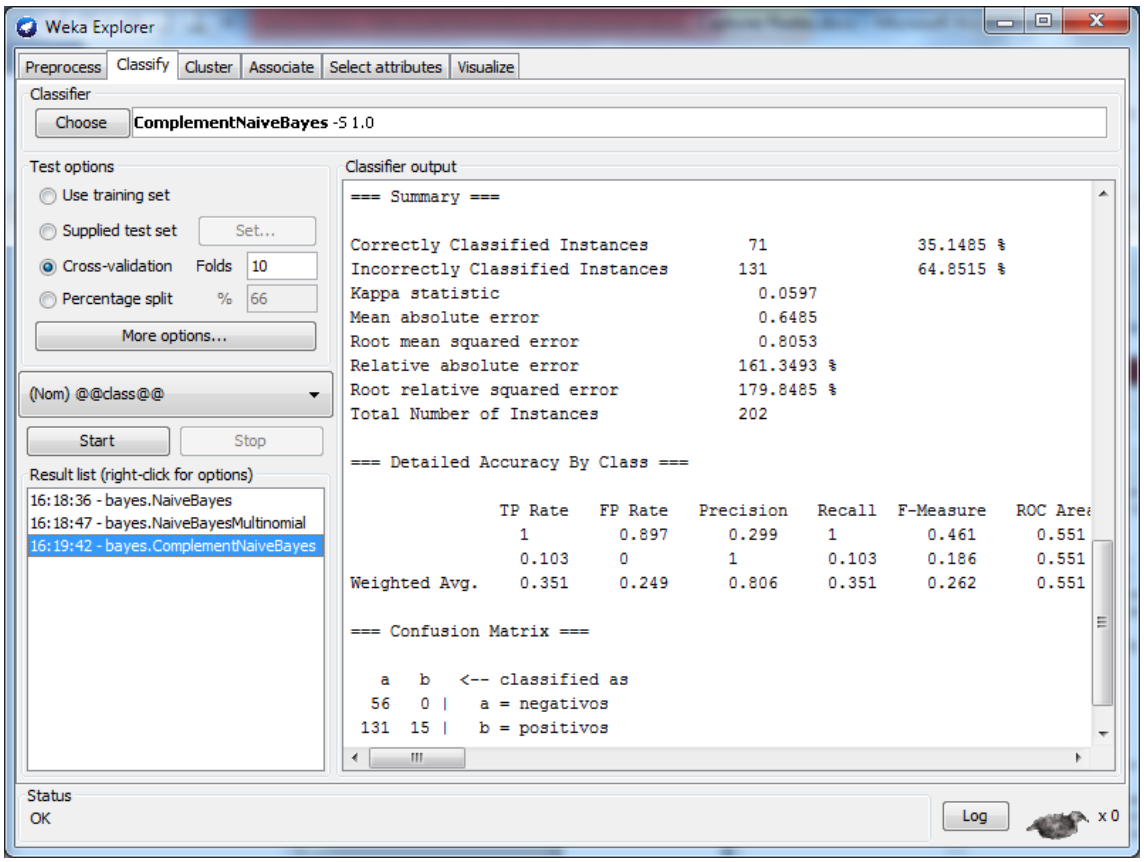

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayesMultinomial
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
10:23:02 - bayes.NaiveBayes
10:23:16 - bayes.BayesianLogisticRegression
10:23:36 - bayes.ComplementNaiveBayes
10:23:50 - bayes.NaiveBayesMultinomial
Classifier output
=== Summary ===
Correctly Classified Instances 121 88.9706 %
Incorrectly Classified Instances 15 11.0294 %
Kappa statistic 0.7795
Mean absolute error 0.2149
Root mean squared error 0.308
Relative absolute error 42.9747 %
Root relative squared error 61.5937 %
Total Number of Instances 136
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.91 0.13 0.871 0.91 0.891 0.95
0.87 0.09 0.909 0.87 0.889 0.95
Weighted Avg. 0.89 0.11 0.891 0.89 0.89 0.95
=== Confusion Matrix ===
a b <-- classified as
61 6 | a = negativos
9 60 | b = Positivos
Status
OK
Log
x 0

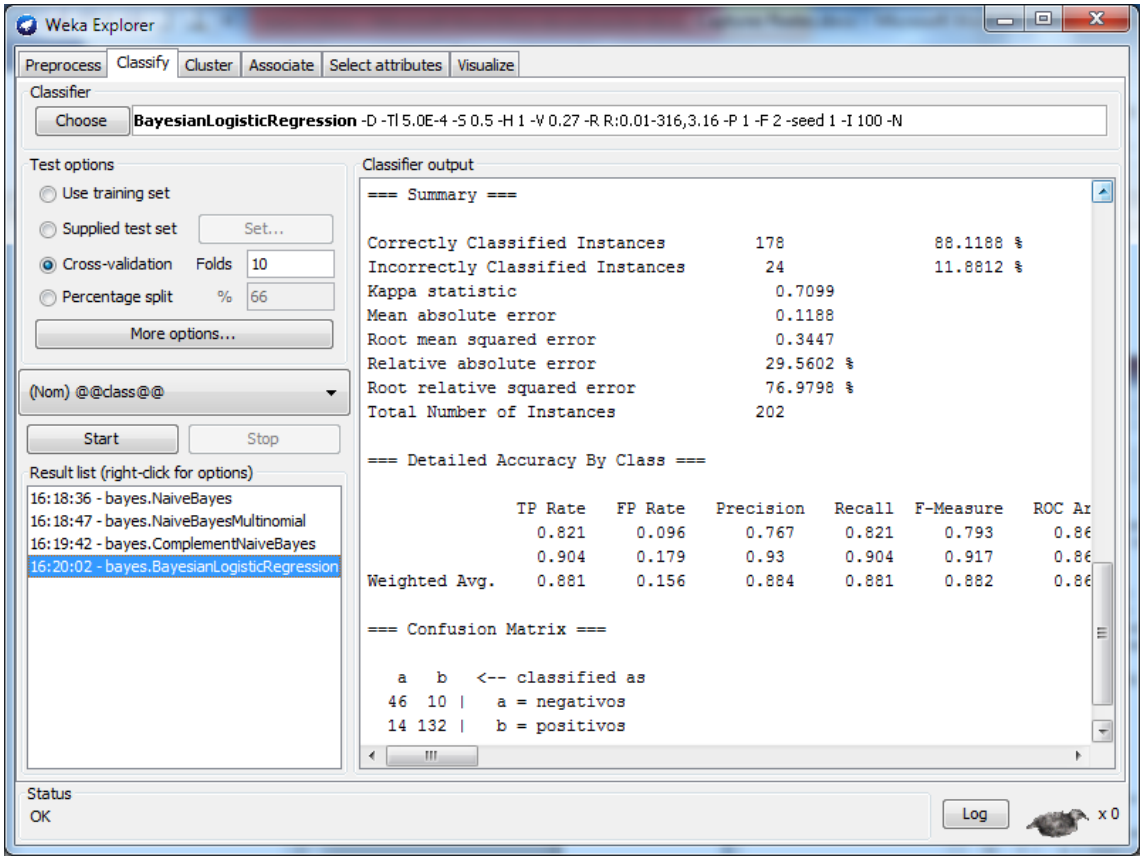


Anexo 5.- Pruebas de polaridad con datos reales de la plataforma MOODLE utilizando Cross-Validation.

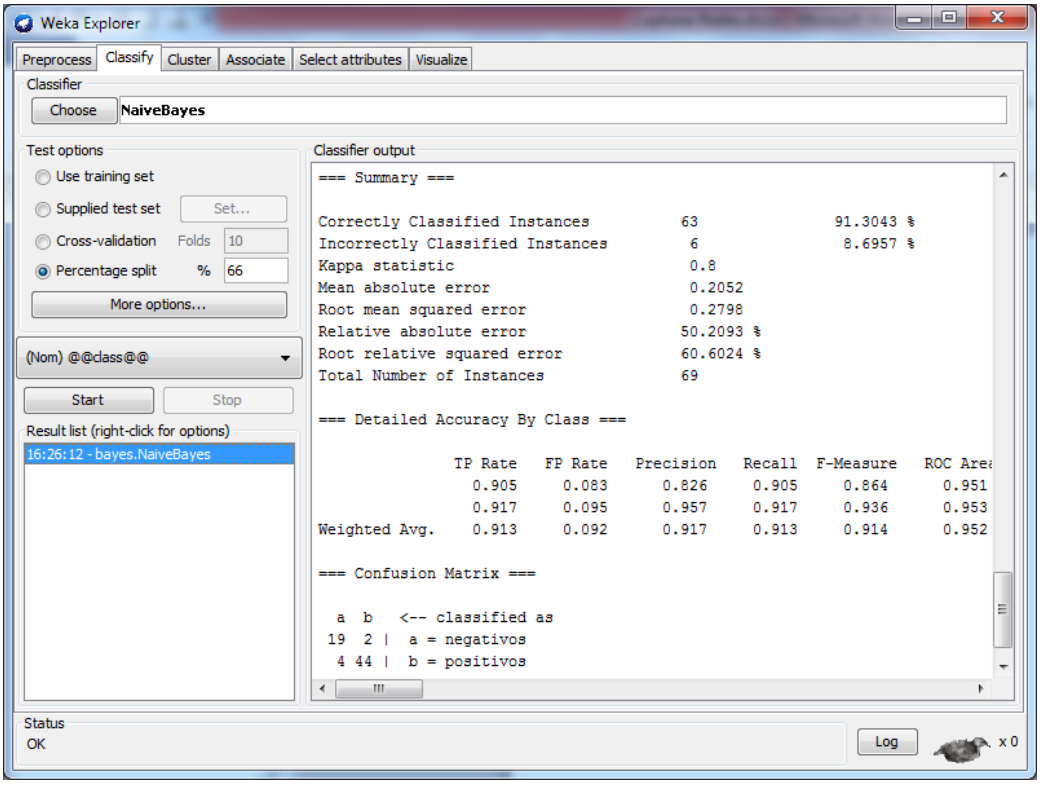


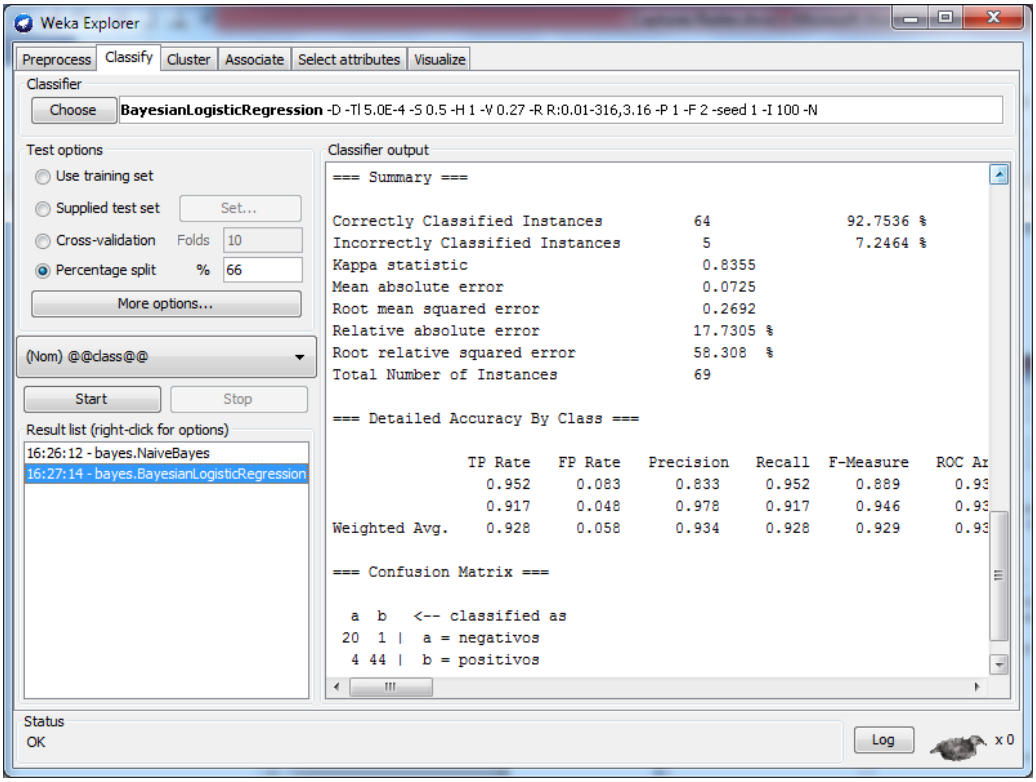

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayesMultinomial
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
16:18:36 - bayes.NaiveBayes
16:18:47 - bayes.NaiveBayesMultinomial
Classifier output
Correctly Classified Instances 148 73.2673 %
Incorrectly Classified Instances 54 26.7327 %
Kappa statistic 0.0508
Mean absolute error 0.3628
Root mean squared error 0.4055
Relative absolute error 90.2761 %
Root relative squared error 90.5555 %
Total Number of Instances 202
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area
0.036 0 1 0.036 0.069 0.904
1 0.964 0.73 1 0.844 0.904
Weighted Avg. 0.733 0.697 0.805 0.733 0.629 0.904
=== Confusion Matrix ===
a b <-- classified as
2 54 | a = negativos
0 146 | b = positivos
Status
OK
Log
x 0

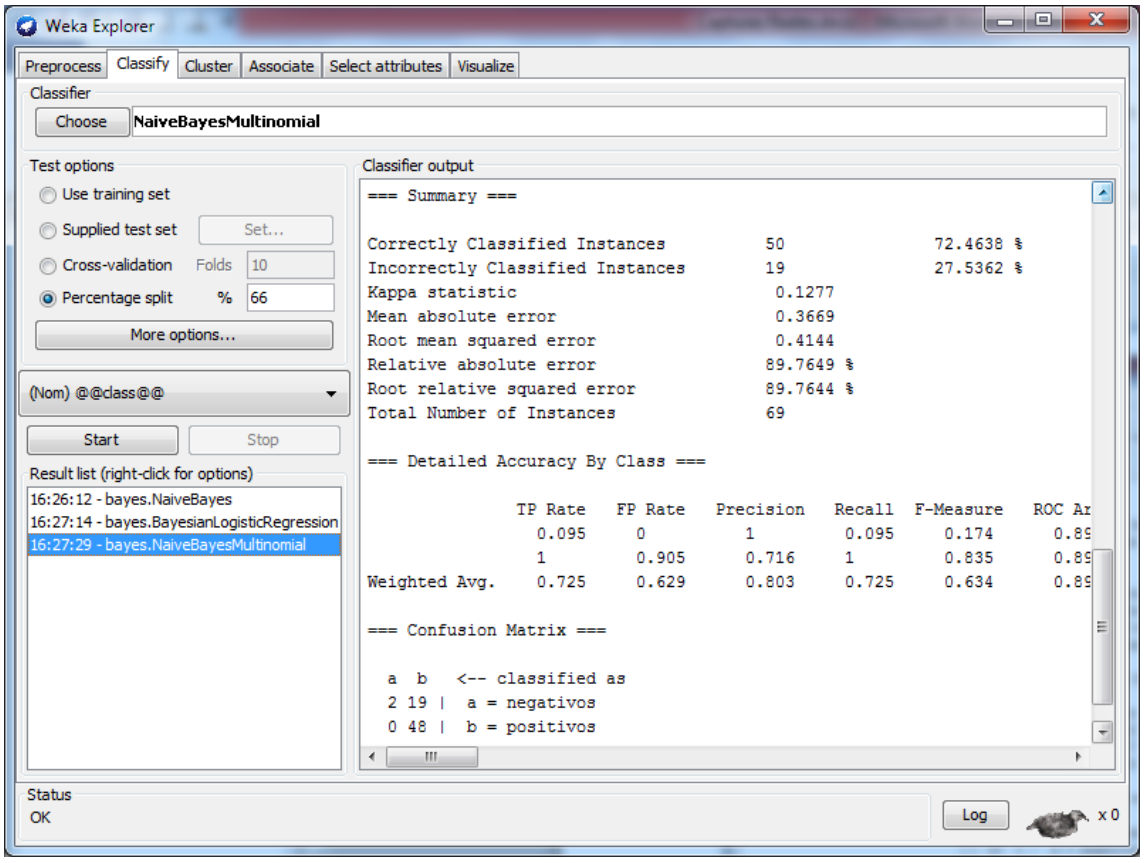

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
ComplementNaiveBayes -S 1.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
16:18:36 - bayes.NaiveBayes
16:18:47 - bayes.NaiveBayesMultinomial
16:19:42 - bayes.ComplementNaiveBayes
Classifier output
=== Summary ===
Correctly Classified Instances 71 35.1485 %
Incorrectly Classified Instances 131 64.8515 %
Kappa statistic 0.0597
Mean absolute error 0.6485
Root mean squared error 0.8053
Relative absolute error 161.3493 %
Root relative squared error 179.8485 %
Total Number of Instances 202
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area
1 0.897 0.299 1 0.461 0.551
0.103 0 1 0.103 0.186 0.551
Weighted Avg. 0.351 0.249 0.806 0.351 0.262 0.551
=== Confusion Matrix ===
a b <-- classified as
56 0 | a = negativos
131 15 | b = positivos
Status
OK
Log
x 0

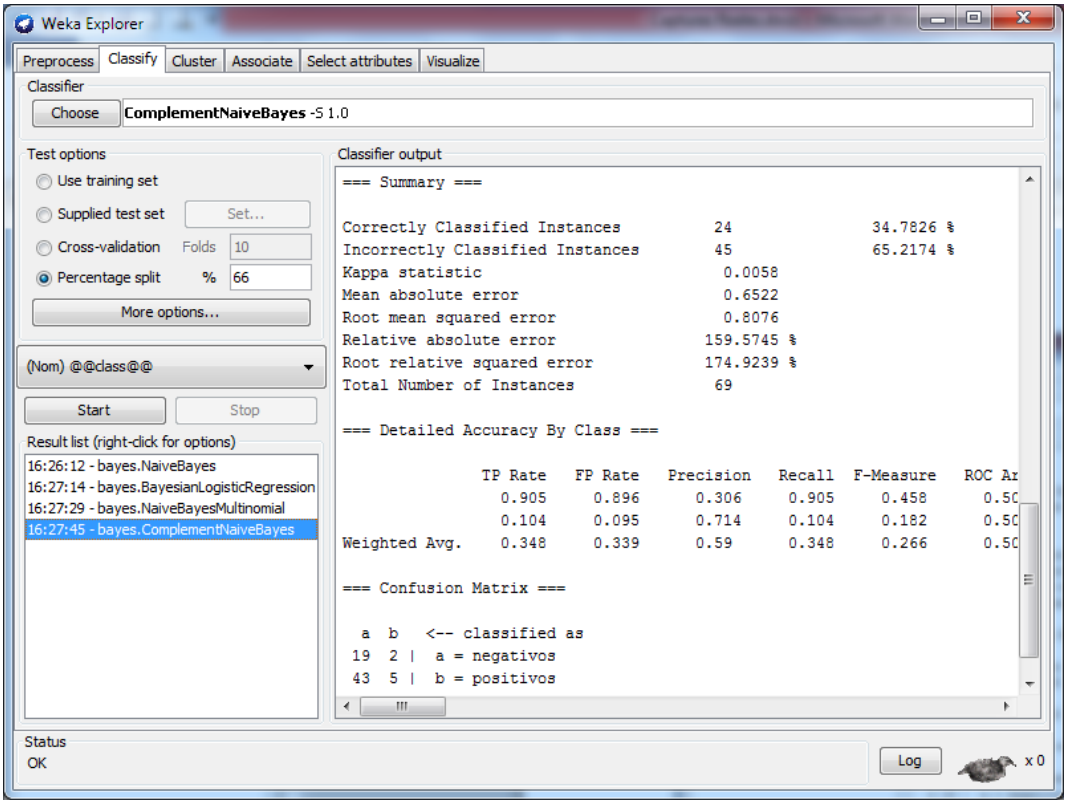


Anexo 6.- **Pruebas de polaridad con datos reales de la plataforma MOODLE utilizando Percentage Split.**

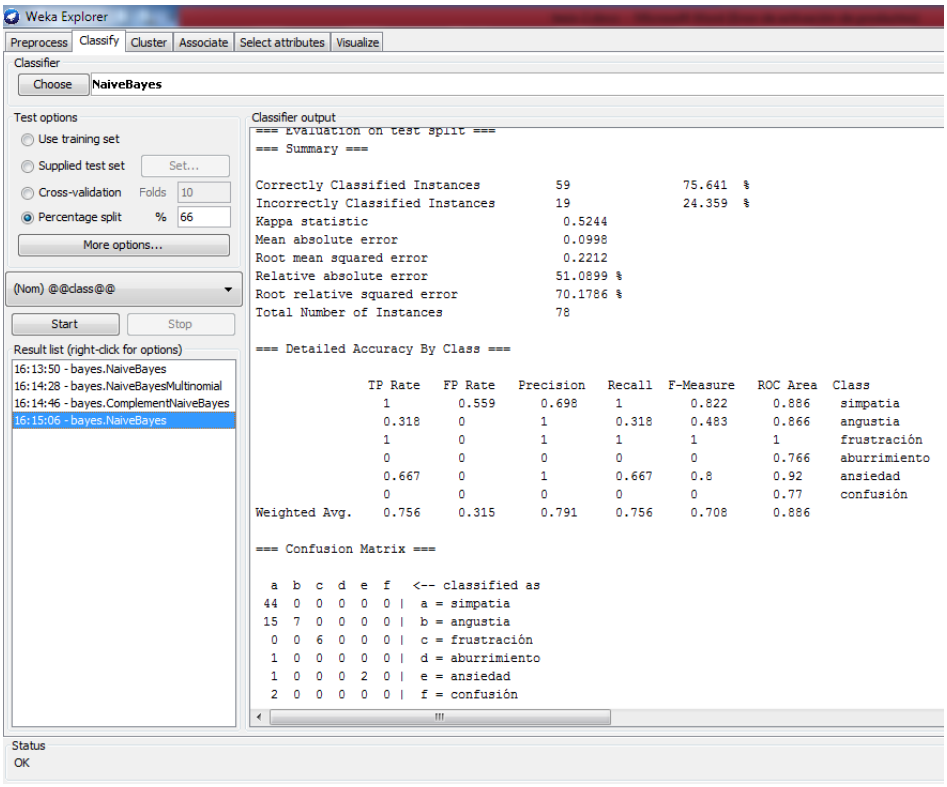


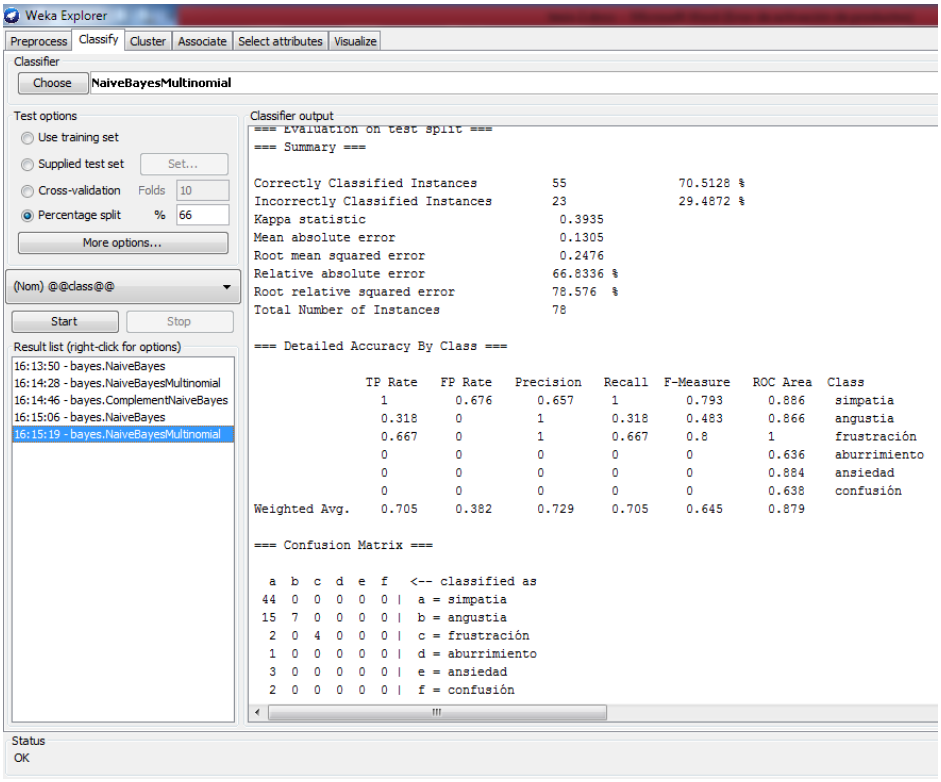

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
BayesianLogisticRegression -D -Tl 5.0E-4 -S 0.5 -H 1 -V 0.27 -R R:0.01-316,3.16 -P 1 -F 2 -seed 1 -I 100 -N
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
16:26:12 - bayes.NaiveBayes
16:27:14 - bayes.BayesianLogisticRegression
Classifier output
=== Summary ===
Correctly Classified Instances 64 92.7536 %
Incorrectly Classified Instances 5 7.2464 %
Kappa statistic 0.8355
Mean absolute error 0.0725
Root mean squared error 0.2692
Relative absolute error 17.7305 %
Root relative squared error 58.308 %
Total Number of Instances 69
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.952 0.083 0.833 0.952 0.889 0.93
0.917 0.048 0.978 0.917 0.946 0.93
Weighted Avg. 0.928 0.058 0.934 0.928 0.929 0.93
=== Confusion Matrix ===
a b <-- classified as
20 1 | a = negativos
4 44 | b = positivos
Status
OK
Log
x 0

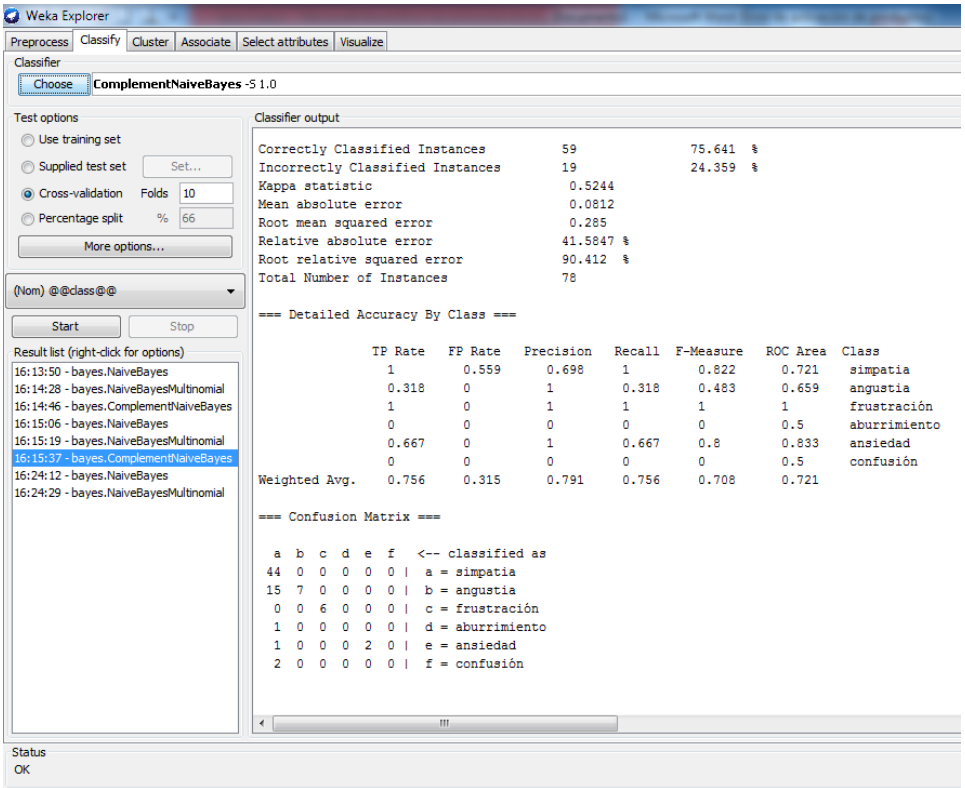

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayesMultinomial
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
16:26:12 - bayes.NaiveBayes
16:27:14 - bayes.BayesianLogisticRegression
16:27:29 - bayes.NaiveBayesMultinomial
Classifier output
=== Summary ===
Correctly Classified Instances 50 72.4638 %
Incorrectly Classified Instances 19 27.5362 %
Kappa statistic 0.1277
Mean absolute error 0.3669
Root mean squared error 0.4144
Relative absolute error 89.7649 %
Root relative squared error 89.7644 %
Total Number of Instances 69
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Ar
0.095 0 1 0.095 0.174 0.89
1 0.905 0.716 1 0.835 0.89
Weighted Avg. 0.725 0.629 0.803 0.725 0.634 0.89
=== Confusion Matrix ===
a b <-- classified as
2 19 | a = negativos
0 48 | b = positivos
Status
OK
Log
x 0



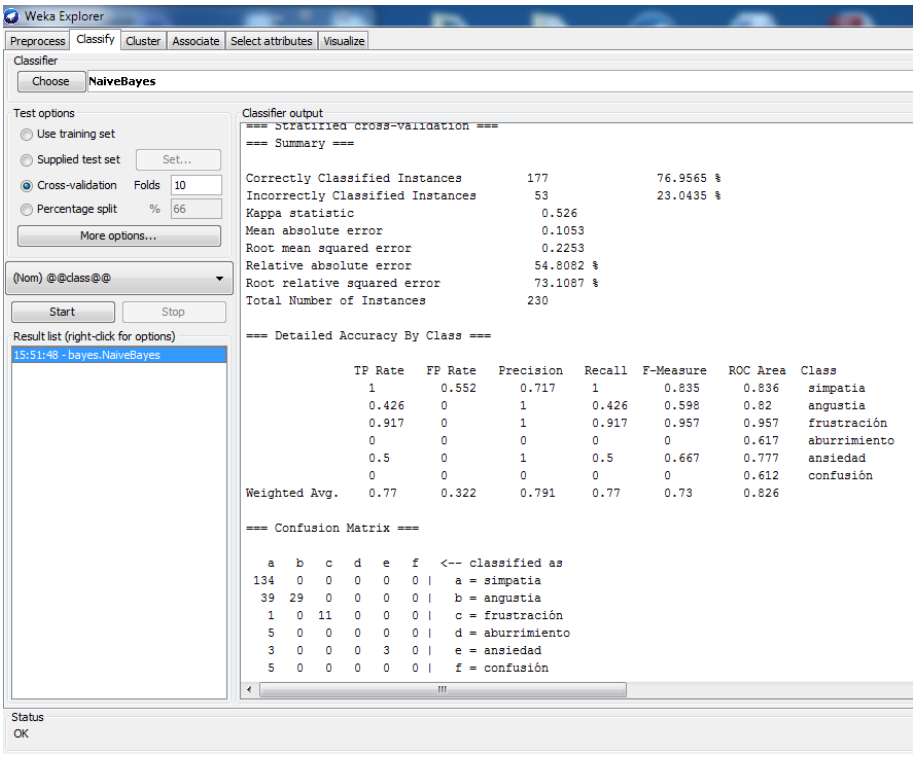
Anexo 7.- Pruebas de clasificación de sentimientos con datos reales de la plataforma MOODLE utilizando Percentage Split.

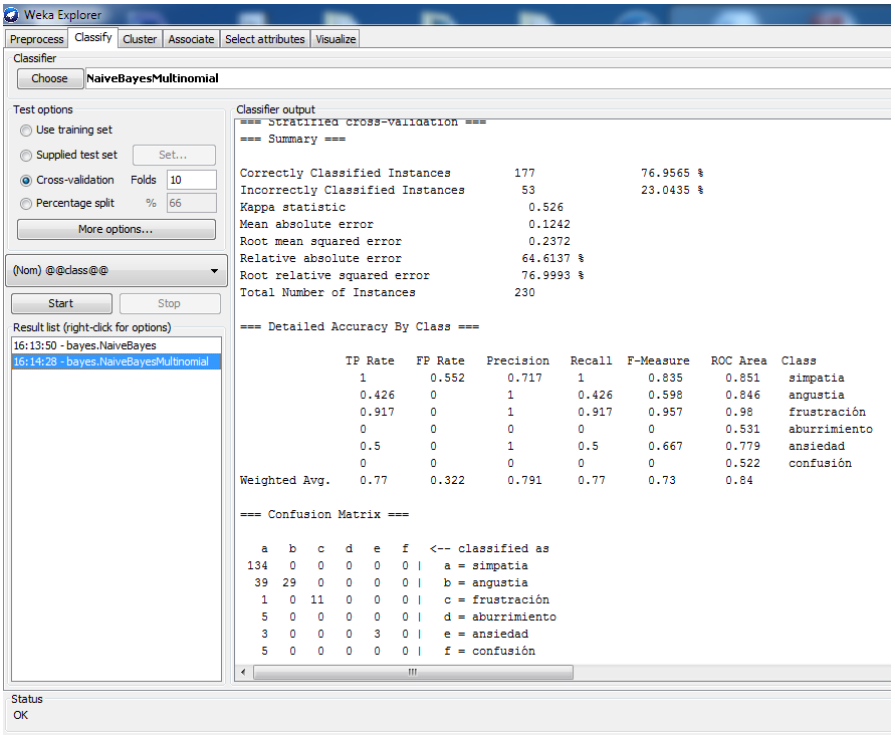


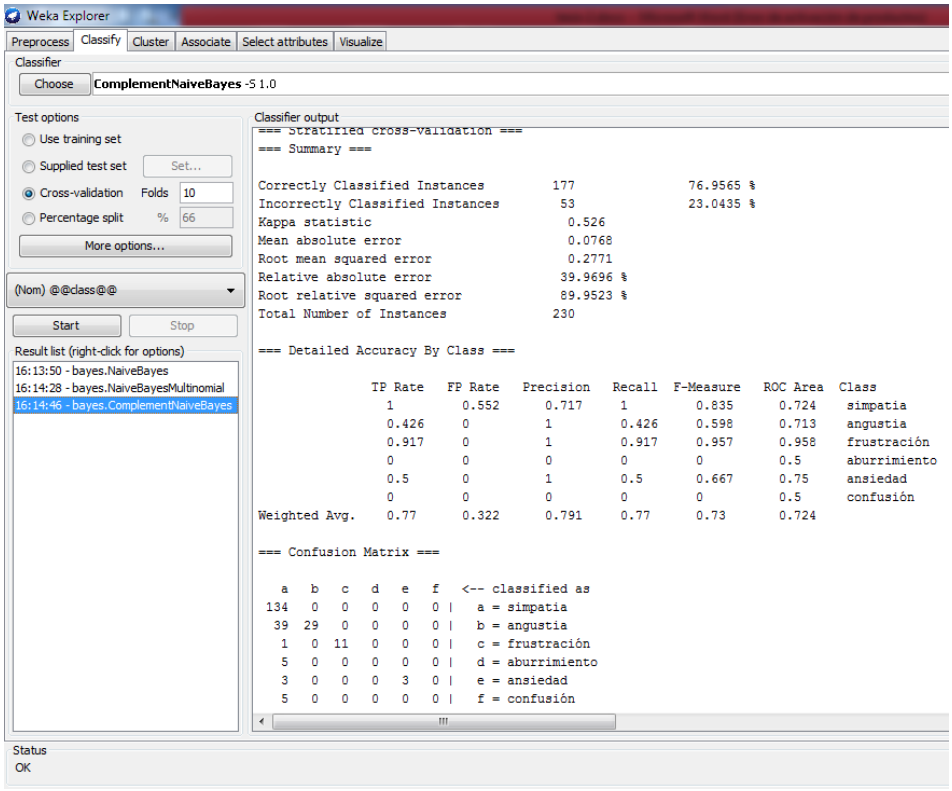

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
NaiveBayesMultinomial
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
16:13:50 - bayes.NaiveBayes
16:14:28 - bayes.NaiveBayesMultinomial
16:14:46 - bayes.ComplementNaiveBayes
16:15:06 - bayes.NaiveBayes
16:15:19 - bayes.NaiveBayesMultinomial
Classifier output
=== Evaluation on test split ===
=== Summary ===
Correctly Classified Instances 55 70.5128 %
Incorrectly Classified Instances 23 29.4872 %
Kappa statistic 0.3935
Mean absolute error 0.1305
Root mean squared error 0.2476
Relative absolute error 66.8336 %
Root relative squared error 78.576 %
Total Number of Instances 78
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
1 0.676 0.657 1 0.793 0.886 simpatia
0.318 0 1 0.318 0.483 0.866 angustia
0.667 0 1 0.667 0.8 1 frustración
0 0 0 0 0 0.636 aburrimiento
0 0 0 0 0 0.884 ansiedad
0 0 0 0 0 0.638 confusión
Weighted Avg. 0.705 0.382 0.729 0.705 0.645 0.879
=== Confusion Matrix ===
a b c d e f <-- classified as
44 0 0 0 0 0 | a = simpatia
15 7 0 0 0 0 | b = angustia
2 0 4 0 0 0 | c = frustración
1 0 0 0 0 0 | d = aburrimiento
3 0 0 0 0 0 | e = ansiedad
2 0 0 0 0 0 | f = confusión
Status
OK


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
ComplementNaiveBayes -S 1.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
16:13:50 - bayes.NaiveBayes
16:14:28 - bayes.NaiveBayesMultinomial
16:14:46 - bayes.ComplementNaiveBayes
16:15:06 - bayes.NaiveBayes
16:15:19 - bayes.NaiveBayesMultinomial
16:15:37 - bayes.ComplementNaiveBayes
16:24:12 - bayes.NaiveBayes
16:24:29 - bayes.NaiveBayesMultinomial
Classifier output
Correctly Classified Instances 59 75.641 %
Incorrectly Classified Instances 19 24.359 %
Kappa statistic 0.5244
Mean absolute error 0.0812
Root mean squared error 0.285
Relative absolute error 41.5847 %
Root relative squared error 90.412 %
Total Number of Instances 78
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
1 0.559 0.698 1 0.822 0.721 simpatia
0.318 0 1 0.318 0.483 0.659 angustia
1 0 1 1 1 1 frustración
0 0 0 0 0 0.5 aburrimiento
0.667 0 1 0.667 0.8 0.833 ansiedad
0 0 0 0 0 0.5 confusión
Weighted Avg. 0.756 0.315 0.791 0.756 0.708 0.721
=== Confusion Matrix ===
a b c d e f <-- classified as
44 0 0 0 0 0 | a = simpatia
15 7 0 0 0 0 | b = angustia
0 0 6 0 0 0 | c = frustración
1 0 0 0 0 0 | d = aburrimiento
1 0 0 0 2 0 | e = ansiedad
2 0 0 0 0 0 | f = confusión
Status
OK

Anexo 8.- Pruebas de clasificación de sentimientos con datos reales de la plataforma MOODLE utilizando Percentage Split.






Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
ComplementNaiveBayes -S 1.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) @@class@@
Start
Stop
Result list (right-click for options)
16:13:50 - bayes.NaiveBayes
16:14:28 - bayes.NaiveBayesMultinomial
16:14:46 - bayes.ComplementNaiveBayes
Classifier output
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 177 76.9565 %
Incorrectly Classified Instances 53 23.0435 %
Kappa statistic 0.526
Mean absolute error 0.0768
Root mean squared error 0.2771
Relative absolute error 39.9696 %
Root relative squared error 89.9523 %
Total Number of Instances 230
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
1 0.552 0.717 1 0.835 0.724 simpatia
0.426 0 1 0.426 0.598 0.713 angustia
0.917 0 1 0.917 0.957 0.958 frustración
0 0 0 0 0 0.5 aburrimiento
0.5 0 1 0.5 0.667 0.75 ansiedad
0 0 0 0 0 0.5 confusión
Weighted Avg. 0.77 0.322 0.791 0.77 0.73 0.724
=== Confusion Matrix ===
a b c d e f <-- classified as
134 0 0 0 0 0 | a = simpatia
39 29 0 0 0 0 | b = angustia
1 0 11 0 0 0 | c = frustración
5 0 0 0 0 0 | d = aburrimiento
3 0 0 0 3 0 | e = ansiedad
5 0 0 0 0 0 | f = confusión
Status
OK