Universidad Internacional de La Rioja (UNIR)

Escuela de Ingeniería

Grado en Ingeniería Informática

Red neuronal para diagnóstico de hipertensión arterial

Ubicación del código fuente:
https://github.com/monteroy/TFG/

Trabajo Fin de Grado

presentado por: García Montero Yolanda

Director/a: Soltero Domingo Francisco José

Ciudad: Madrid
Fecha: Julio 2018

Resumen

Este trabajo tiene por objetivo predecir una enfermedad aplicando distintos tipos de clasificadores dando mayor peso a los métodos basados en redes neuronales. Se utilizaran distintos técnicas intentando aunar los conocimientos adquiridos durante los estudios.

Palabras Clave: Clasificadores, Perceptrón Multicapa, árbol de decisión C4.5.

Abstract

This work aims to predict a disease by applying different types of classifiers giving greater weight to the methods based on neural networks. Different techniques will be used trying to combine the knowledge acquired during the studies.

Keywords: Classifiers, Multilayer Perceptron, decision tree C4.5

MOTIVACIÓN 1
OBJETIVO 1
1 Introducción 1
2 Contexto y estudio preliminar 1
2.1 Estudios existentes sobre hipertensión arterial y técnicas de IA 1
2.1.1 Técnicas de IA y ayuda a toma de decisiones 2
2.2 Marco teórico 2
2.2.1 Aprendizaje automático 2
2.2.2 Preprocesado de los datos 3
2.2.3 Árbol de clasificación 3
2.2.4 Datos faltantes. 3
2.2.4.1 Decisión Tree 4
2.2.4.2 CHAID 4
2.2.4.3 Decisión Stump 4
2.2.4.4 Gradient Boosted Trees 5
2.2.4.5 Random Tree 5
2.2.4.6 RandomForest 5
2.2.4.7 Naive Bayes 5
2.2.4.8 Moda 5
2.2.4.9 Deep Learning 5
2.2.4.10 Knn 5
2.2.5 Redes neuronales 6
2.2.5.1 Clasificación RNAs 7
Naturaleza de entrada 7
Propiedades 7
Topología 7
Algoritmo de aprendizaje 7
Arquitectura 7
2.2.5.2 Perceptrón. 7

2.2.5.3 Red neuronal multicapa 8
Retropropagación en una red con una capa oculta 8
Retropropagación en una red con varias capas ocultas 8
2.2.6 Aprendizaje supervisado. Combinación secuencial de clasificadores con base diferente 9
2.2.6.1 Ensamblado de clasificadores....10
2.2.7 Evaluación de modelos clasificadores. Métricas de evaluación. 10
2.2.7.1 Verdaderos positivos TP – True Positives....10
2.2.7.2 Verdaderos negativos TN – True Negatives....10
2.2.7.3 Falsos positivos (FP – False Positives)....11
2.2.7.4 Falsos negativos (FN – False Negatives)....11
2.2.7.5 Matriz de confusión....11
2.2.7.6 Tasa de verdaderos positivos (TP Rate), recall, alcance o sensibilidad (sensitivity). 11
2.2.7.7 Tasa de verdaderos negativos (TN Rate), especificidad o specificity. 11
2.2.7.8 Exactitud (accuracy, correctly classified instances)....12
2.2.7.9 Tasa de error (error rate) 12
2.2.7.10 Precisión....12
2.2.7.11 F-measure 12
2.2.7.12 Kappa statistic....12
2.2.7.13 Error absoluto medio y Raíz del error cuadrático medio....13
2.2.7.14 Error absoluto relativo y la raíz del error cuadrático relativo....13
2.2.7.15 MCC 14
2.2.7.16 Curva ROC 14
3 Selección y preparación del conjunto de datos 15
3.1 Selección datos en la web....15
4 Herramientas 20
4.1 Herramientas utilizadas en la extracción de datos....20
4.1.1 Node.js....20
request 20

sanitize-html 20
js.....20
4.1.2 Visual Studio Code Insiders 21
Regex Previewer 21
4.2 Herramientas utilizadas para convertir los datos a formato arff 21
4.2.1 Dev-C++ 21
4.3 Herramientas utilizadas en clasificación, imputación de datos faltantes y balanceo de datos. 23
4.3.1 Weka.....23
4.3.2 RapidMiner.....23
4.4 Herramientas utilizadas en la predicción de nuevos casos utilizando el modelo generado. 24
4.4.1 NetBeans 24
4.4.2 Launch4j-3.12 24
4.5 GitHub.....24
4.6 Typora.....24
5 Descripción del proyecto 25
5.1 Extracción de datos web. 25
5.2 Conversión de datos a formato arff. 27
5.3 Mejor validación 31
5.3.1 Use training set 32
5.3.2 Supplied test set 32
5.3.3 Cross-validation 32
5.3.4 Percentage Split.....32
5.4 Imputación de datos.....33
5.4.1 Datos desbalanceados.....36
5.4.1.1 Reducción de datos 38
Sample.....38
Muestreo estratificado.....38
5.5 Selección de atributos.....42

5.6 Red Neuronal con Weka 42
5.6.1 Resultados de la clasificación con MLP 42
5.7 Ejecución de ensambladores 46
5.7.1 Bagging....47
5.7.2 RandomSubSpace....47
5.7.3 AdaBoos 48
5.7.4 Voting....48
5.7.5 Stacking....48
5.7.6 Resultados del ensamblado de clasificadores....49
6 Evaluación 51
6.1 Resultados....51
6.2 Uso del modelo generado 53
7 Conclusiones y trabajo futuro....56
Trabajos citados....58
Anexos....73
Variables y frecuencias 75
Diccionario de acrónimos....78

Índice de Figuras

Figura 1 (Palma M. J. T., 2008). Modelo neuronal de McCulloch y Pitts.....6
Figura 2 (Pérez Águila, 2012) Red de perceptrones de m capas ocultas con m ≥ 2.....9
Figura 3 (Pérez Águila, 2012) Propagación de la entrada en una red de multicapas ocultas..9
Figura 4 Ejemplo de curva ROC (Cerda J., 2012). 14
Figura 5 salud castilla y león. (s.f.). 16
Figura 2 salud castilla y león. (s.f) 16
Figura 3 salud castilla y león. (s.f.). 17
Figura 4 salud castilla y león. (s.f.). 17
Figura 5 salud castilla y león. (s.f.). 18
Figura 6 salud castilla y león. (s.f.). 18
Figura 11 Características de distintos dataset.....19
Figura 12 Comparativa de clasificación entre J48 y MLP 19
Figura 13 Encabezado con los comentarios en el formato arff 22
Figura 14 Encabezado con el nombre de la relación y los atributos 22
Figura 15 Ejemplo de los datos de entrada 23
Figura 16 Ejemplo de expresión regular.....25
Figura 17 Función de escritura en JavaScript.....25
Figura 18 Elementos seleccionados por la aplicación descargaDatos_1.js 26
Figura 19 Parte del fichero de salida de la aplicación descargaDatos_1.js.....26
Figura 20 Vista general del software de extracción de datos.....27
Figura 21 Almacenamiento cantidad de enfermos y sanos que son hombres 29
Figura 22 Evaluación con condición para estudios en menores de dieciséis años 30
Figura 23 Evaluación con condición para trabajo en menores de dieciséis años y mayores de sesenta y cinco 30
Figura 24 Marcado de datos ausentes en la salida.....31
Figura 25 Muestra de la matriz obtenida importada con la herramienta RapidMiner.....31
Figura 26 Filtros en Weka 33
Figura 27 Imputación de valores ausentes mediante la herramienta RapidMiner 34
Figura 28 Histogramas comparados en resultados de imputación de datos 34
Figura 29 Porcentajes de la clasificación hecha.....36
Figura 30 Filtro Sample para reducción de instancias 38
Figura 31 Árbol generado por Weka en la clasificación de sample por probabilidad 5% 41
Figura 32 Red neuronal de dos capas ocultas con cuatro neuronas por capa.....45
Figura 33 Error en la época 3.....45
Figura 34 Error en la época 100.....45
Figura 35 Acciones llevadas a cabo por cada icono.....46

Figura 36 Baggin.....47
Figura 37 RandomSubSpace.....47
Figura 38 AdaBoos.....48
Figura 39 Voting.....48
Figura 40 Stacking.....49
Figura 41 Resultados clasificación MLP.....51
Figura 42 Falsos negativos.....52
Figura 43 Fichero arff con los datos de un usuario.....53
Figura 44 Weka en el entorno NetBeans.....53
Figura 45 Ejemplo de uso de Launch4j.....54
Figura 46 Salida de la predicción.....55

Índice de tablas

Tabla 1 11
Tabla 2 27
Tabla 3 32
Tabla 4 35
Tabla 5 37
Tabla 6 39
Tabla 7 39
Tabla 8 44
Tabla 9 49
Tabla 10 52
Tabla 11 74
Tabla 12 75
Tabla 13 76
Tabla 14 77

MOTIVACIÓN

Tras una etapa apasionante dedicada a la obra civil tuve la necesidad de replantear mi futuro. En ese momento es cuando veo que siempre he sido una apasionada de la informática, a pesar de ser una gran desconocedora del área y consciente de la revolución que estamos viviendo decido comenzar el grado en informática en la UNIR.

Cuatro años después constato que ha sido una muy buena decisión, me ha permitido descubrir temáticas que me entusiasman y entre ellas destaca los sistemas de ayuda a la toma de decisiones.

OBJETIVO

El objetivo de este trabajo es que el facultativo pueda predecir con una probabilidad alta una enfermedad como es la hipertensión, evitando hacer pruebas diagnósticas a pacientes sanos.

1 Introducción

Según la organización mundial de la salud (OMS, s.f.) la hipertensión, o tensión arterial alta, es un trastorno en el que los vasos sanguíneos tienen una tensión habitual alta, pudiendo dañarlos. Cuando el corazón late, bombea sangre a los vasos y estos llevan la sangre a todas las partes del cuerpo. La tensión arterial es la fuerza que ejerce la sangre contra las paredes de los vasos (arterias) al ser bombeada por el corazón. Cuanto más alta es la tensión, más esfuerzo tiene que realizar el corazón para bombear. La mayoría de las personas con hipertensión no muestra ningún síntoma. A veces, la hipertensión causa síntomas como dolor de cabeza, dificultad respiratoria, vértigos, dolor torácico, palpitaciones del corazón y hemorragias nasales, pero no siempre. Si no se controla, la hipertensión puede provocar un infarto de miocardio, un ensanchamiento del corazón y, a la larga, una insuficiencia cardiaca.

Las pruebas necesarias a este tipo de pacientes pueden ser de alto coste económico según se dice en (Saez M., 2012) en promedio, un hipertenso costaría el doble que lo que costaría un individuo normotenso. Ejemplo de estas pruebas son; los electrocardiogramas de esfuerzo (del Río A., 2002),, las ecocardiografías (Torres Macho J., 2012), los exámenes ecográficos de los vasos de la pierna y del cuello (Berardi H., 2015), sin mencionar las pruebas necesarias para las graves enfermedades derivadas según l según la (OMS, 2013) de esta dolencia

como son; la cardiopatía (Esteban Fernández A., 2014), los accidentes cerebrovasculares (Álvarez-Aliaga, 2006), o la insuficiencia renal (Marín R., 2004).

Este trabajo comienza con un primer apartado en el que se definen los conceptos que serán objeto de estudio, un segundo apartado identifica los requisitos y explica las distintas metodologías que se utilizaran para conseguir el objetivo, continuando por el apartado que describe las herramientas utilizadas. El siguiente capítulo describe el proyecto, que es lo que aporta y que funcionalidad tiene. Terminando con la evaluación y las conclusiones.

2 Contexto y estudio preliminar

Los sistemas de ayuda a la toma de decisiones se encuentran muy extendidos actualmente en distintas campos, ejemplo de ello es la predicción de **quiebra bancarias** (Cinca C., 1993), el cálculo de la **inflación** (Aristizábal M., 2006), aplicadas a la ingeniería **eléctrica** (Flores Fernández J.M., 2011), enfocados al **lenguaje natural** (de Luna-Ortega C. A., 2014) o en la detección de **daño en vigas** (Villalba J.D, 2012).

En el campo de la medicina existen múltiples estudios en los que se usan técnicas de inteligencia artificial (IA) para el diagnóstico de enfermedades, entre los que se pueden citar el Sistema Experto para diagnóstico de **colesterol** (Mamami Vargas G. F., 2017) que emplea lógica difusa, otros implementan redes neuronales para la detección de enfermedades del **corazón** (Avellaneda González J.A., 2010) que recurre al aprendizaje supervisado (implementando un perceptrón multicapa) y no supervisado (implementando una red ARP), distintos estudios emplean árboles de decisión para la identificación de posibles interacciones entre factores de riesgo de **diabetes** tipo 2 (Ramezankhani A., 2016), o clasifican medidas de glucemia para detectar **diabetes gestacional** (Caballero Ruiz E., 2012).

2.1 Estudios existentes sobre hipertensión arterial y técnicas de IA

Existen diversos estudios en éste ámbito, cabe destacar los realizados por (Cuadrado Rodríguez S., 2011) “Sistema experto basado en casos para el diagnóstico de la hipertensión arterial” en el cual los individuos son clasificados en **normotensos** (personas con presión arterial normal), **prehipertensos** (personas en riesgo de padecer HTA) e **hipertensos** utilizando variables extraídas de la **historia clínica,** también a partir del historial médico en el estudio de (Chávez M. C., 2009) llamado “Uso De Redes Bayesianas Obtenidas Mediante Optimización de enjambre de partículas para diagnóstico de hipertensión arterial” los pacientes se clasifican en **normotensos**, **hiperreactivos** e **hipertensos** siendo equivalente la clasificación entre ambos estudios.

En (Yzquierdo R., 2005) las fuentes fueron los expertos en materia de HTA con los que se siguió un proceso de ingeniería del conocimiento.

En (Vanderlei Filho D., 2005) se aplican técnicas de IA para diagnostico o ayuda a toma de decisiones.

2.1.1 Técnicas de IA y ayuda a toma de decisiones

Existe una amplia gama de estudios referentes a este apartado; el **razonamiento basado en casos** para enseñanza a distancia es propuesta por (Shen R., 2003), las **redes Bayesianas** como herramientas de modelado en psicología (López Puga, 2007), una sola **red Neuronal** y la escala de Framingham (Pérez Díaz A., 2012) que es un test que usan los cardiólogos para saber el riesgo aproximado que tiene un paciente de sufrir un infarto de miocardio en los próximos años, el **perceptrón multicapa** aplicados a tecnología Android (Ñustes S.A., 2013), los **árboles de decisiones** (Solarte Martínez G., 2011), que usa el algoritmo ID3 con el objeto de proporcionar medicación a pacientes con síntomas de enfermedad clasificados según propiedades que indican la enfermedad o **sistemas basados en reglas** para diseñar transmisiones por tornillo sinfín (Moya-Rodríguez J.L., 2012).

2.2 Marco teórico

En nuestro trabajo nos basaremos en distintas técnicas, conceptos y algoritmos ampliamente corroborados pero creemos necesario hacer un pequeño repaso de los mismos para mejor comprensión de los resultados.

2.2.1 Aprendizaje automático

El aprendizaje automático es una rama de la inteligencia artificial cuyo objetivo es desarrollar técnicas que permitan a las computadoras aprender (Mitchell T., 1997) queremos crear un modelo que sea capaz de generalizar comportamientos a partir de información suministrada en forma de ejemplos, En nuestro caso estamos ante un aprendizaje supervisado cuyo objetivo es crear una función capaz de predecir el valor correspondiente a cualquier objeto de entrada después de haber visto una serie de ejemplos.

Existen múltiples técnicas para crear clasificadores (Aluja T., 2001) como son; el análisis factorial descriptivo, el análisis de la cesta de la compra, técnicas de clustering, series temporales, redes bayesianas, modelos lineales, redes neuronales, árboles de decisión y algoritmos genéticos entre otros.

En un primer momento se puede pensar que un clasificador estadístico puede ser apropiado para este conjunto de datos pero hay estudios en que se decantan por otro tipo de clasificadores como las redes neuronales alegando distintos motivos, como ejemplo en (González Padilla A., 2013) donde seleccionan las redes neuronales por las siguientes ventajas; por ser máquinas independientes de los modelos, por ser capaces de ajustarse a cualquier salida deseada, por ser de menos costo computacional y porque la velocidad de

respuesta en tiempo real es mayor que empleando una clasificación estadística. En el artículo (Buendía Rodríguez E., 2002) se cita que el 71 % de los estudios publicados sobre modelos de RNA predicen mejor que las técnicas estadísticas tradicionales, que el 12 % de los estudios de RNA predicen aproximadamente igual que las técnicas estadísticas tradicionales, y que sólo el 17 % de los estudios logran que las técnicas estadísticas tradicionales sean mejores que los modelos de RNA.

2.2.2 Preprocesado de los datos

La fase de preprocesado y selección de atributos es tan relevante en (García Gutiérrez J.A., 2016) se justifica que puede llevar incluso la eliminación de alguno de ellos para así lograr una mejor clasificación, basado en esto haremos una evaluación de los atributos con distintas técnicas.

2.2.3 Árbol de clasificación

A partir de este momento vamos a utilizar distintos árboles de clasificación, debido a su fácil interpretación los clasificadores de árbol son muy usados (Valcárcel Asencios V., 2004). El algoritmo C4.5 es un método de inducción de árboles de decisión basado en ID3. Propuesto por (Quinlan J., 1993) ID3 se aplica a atributos con valores discretos, C4.5 (Quinlan J. R., 1996) se puede aplicar también a valores continuos, puede trabajar con datos ausentes, poda tras la generación del árbol (pospoda (Medina-Merino, 2017)) con el fin de mejorar la generalización del modelo (mapea a reglas para hacer la poda). Pasos; se seleccionan los atributos con la medida de proporción de ganancia (Alonso Jiménez J.A., 2000), con estos se genera el árbol de decisión a partir de los datos de entrenamiento, convierte el árbol en un conjunto de reglas (el camino raíz hoja es una regla (una serie de condiciones enlazadas con el operador AND), poda cada regla (se eliminan las condiciones en el antecedente que suponen mejorar la precisión de la clasificación) y ordena las reglas en función de la precisión estimada.

C4.5 realiza una poda pesimista ya que la precisión la calcula con los propios datos de entrenamiento (Mitchell T., 1997).

2.2.4 Datos faltantes.

En la mayoría de los estudios muéstrales y/o censales hay espacios vacíos, que producen problemas en el análisis posterior, desde hace tiempo se estudia la forma de “llenar” estos espacios vacíos, con el fin de obtener un conjunto de datos completos. La ausencia de datos es un gran problema, si el conjunto de datos es grande y hay poca pérdida, puede ignorarse

la ausencia de datos, pero esto no es conveniente cuando se trata de pocas observaciones o de altas proporciones de pérdida (Useche Castro L. M., 2006).

Existen múltiples algoritmos que versan sobre el tratamiento de valores faltantes para datos nominales podemos usar entre otros árboles de decisión con técnicas como CHAID (Kass V., 1980) que es un operador de árbol de decisión utilizando un criterio basado en ji cuadrado en lugar de los criterios de ganancia de información (Solarte Martínez G., 2011) o relación de ganancia que se usan en otros árboles de decisiones.

Existen técnicas para afrontar este problema como son el análisis de casos completos y las imputaciones. En nuestro caso no podemos analizar casos completos por lo que la solución pasa por realizar imputaciones.

Hemos imputado los datos con las siguientes técnicas:

2.2.4.1 Decisión Tree

Es un árbol de decisión (Quinlan, 1986) los nodos hoja se etiquetan con una de las posibles clases y cada nodo interno hace referencia a un atributo de tal manera que cada nodo hijo corresponde a un valor del atributo. El camino hasta un nodo hoja corresponde con una conjunción de los valores de los atributos de los nodos visitados.

2.2.4.2 CHAID

CHAID (Wilkinson L.) funciona como el operador de árbol de decisión con una excepción, utiliza un criterio basado en ji cuadrado de Pearson (Spiegel M. R., 1991), utiliza una distribución de probabilidad continua en lugar de los criterios de ganancia de información (Quintero-Méndez M. A., 2008). La ganancia de información utiliza la entropía para medir la efectividad de un atributo para clasificar ejemplos. Específicamente mide la reducción de entropía cuando se distribuyen los ejemplos de acuerdo a un atributo concreto. La entropía es una medida de como está organizado un conjunto de datos en un sistema cerrado y trabaja con la proporción de entradas que pertenece a cada clase dado un atributo (Rényi A., 1961). La entropía caracteriza la heterogeneidad de un conjunto de ejemplos.

2.2.4.3 Decisión Stump

Es un modelo de aprendizaje automático (Yongheng Z., 2008), se usa para generar un árbol de decisión con una sola división.

2.2.4.4 Gradient Boosted Trees

Es un modelo con gradiente mejorado (De'ath G., 2007), conjunto de modelos de árbol de regresión o de clasificación, son métodos conjuntos de aprendizaje progresivo que obtienen resultados predictivos a través de estimaciones mejoradas gradualmente. El refuerzo es un procedimiento de regresión no lineal flexible que ayuda a mejorar la precisión de los árboles.

2.2.4.5 Random Tree

Funciona como el operador Decision Tree (Le Gall J.F., 1992) con una excepción, para cada división solo está disponible un subconjunto aleatorio de atributos.

2.2.4.6 RandomForest

Es un conjunto de un cierto número de árboles aleatorios (Liaw A., 2001), se especifican el parámetro de número de árboles (Breiman L., Random Forests, 2001).

2.2.4.7 Naive Bayes

Clasificador probabilístico (Jonh G. H., 1995) fundamentado en el teorema de Bayes (Nieves Hurtado A., 2010)

2.2.4.8 Moda

Medida de tendencia central para atributos nominales (Montero Lorenzo J.M., 2007), (Spiegel M. R., 1991).

2.2.4.9 Deep Learning

Se basa en una red neuronal artificial de alimentación de múltiples capas que está entrenada con un descenso de gradiente estocástico, no determinístico, mediante retro-propagación (Schmidhuber J., 2014). El descenso de gradiente es un algoritmo de optimización iterativa de primer orden para encontrar el mínimo de una función (Mederos Brú M.V., 2004).

2.2.4.10 Knn

Está englobado en los métodos kernel (Peterson L. E., 2009), los métodos kernel están basados en una función kernel que refleja una relación de similitud entre dos elementos arbitrarios del conjunto de entrada. El caso más común en el que utilizar un método de kernel es ventajoso es cuando los ejemplos no son linealmente separables en el espacio original (lo cual es una característica común en la mayoría de los ejemplos reales) pero sí en el espacio kernelizado.

El algoritmo k-Nearest Neighbor se basa en la comparación de un ejemplo desconocido con los k ejemplos de entrenamiento que son los vecinos más cercanos del ejemplo desconocido.

Primero calcula la distancia entre el ejemplo desconocido y los ejemplos, pueden usar diferentes medidas, como la distancia euclidiana, después clasifica el ejemplo desconocido por el voto mayoritario de los vecinos encontrados.

2.2.5 Redes neuronales

Una Red de Neuronas Artificiales o RNA es un tipo de procesamiento de información inspirado en el modo en el que lo hace el cerebro (Palma M. J. T., 2008). Las redes neuronales actuales se basan en el modelo matemático de neurona propuesto por McCulloch y Pitts en 1943 en el que cada neurona recibe un conjunto de entradas {x1, x2, . . . , xD} y devuelve una única salida y. Dentro de la neurona existen numerosas conexiones de mayor o menor intensidad llamados pesos sinápticos. Para obtener la salida y de la neurona se la activa calculando la suma ponderada de las entradas afectadas de sus pesos sinápticos y un umbral o sesgo w_0 para compensar la diferencia entre el valor medio de las entradas (2-1) gráficamente lo podemos ver en la figura 1.

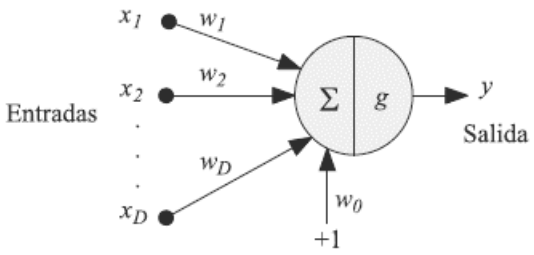


Figura 1 (Palma M. J. T., 2008). Modelo neuronal de McCulloch y Pitts

$$y = \sum_{i=1}^{D} w_i x_i + w_0 \quad (2\text{-}1)$$

A partir de la entrada neta a se obtiene la salida aplicando la función de activación o de transferencia g(α), que se encarga de decidir cuándo activar la neurona y cuando no.

$$y = g(\alpha) = g\left(\sum_{i=1}^{D} w_i x_i + w_0\right) \quad (2\text{-}2)$$

Esta expresión matemática dice como procesa la información la neurona artificial.

La función de activación g(α), función escalón, toma los valores uno si $\alpha > 0$ o valor cero si $\alpha < 0$. En otros modelos (Gironés J., 2017) son otras funciones como la lineal, sigmoidea, de tangente hiperbólica o gaussiana.

2.2.5.1 Clasificación RNAs

No es fácil hacer una clasificación de estas redes ya que en su evolución aparecen múltiples modelos híbridos (Sánchez E., 2011) hace distintas clasificaciones de las redes neuronales (RNAs) según;

Naturaleza de entrada

Continua o binaria (discreta)

Propiedades

Topología

Feedforward, Backforward o Redes competitivas (Martín Valdivia M.T., 2002), redes de Kohonen

Algoritmo de aprendizaje

Aprendizaje supervisado, aprendizaje no supervisado u auto organizado. La combinación del aprendizaje supervisado y no supervisado producen redes hibridas entre ambas se encuentra el aprendizaje reforzado.

Arquitectura

Monocapa (Bradley D.A., 2000); Perceptrón y Adeline, Multicapa (Shashi Sathyanarayana P., 2014); perceptrón multicapa y Redes recurrentes (Unadkat S. B., 2001) como las de Elman y Hopfiel.

2.2.5.2 Perceptrón.

Dentro de las neuronas artificiales está el Perceptrón (Gironés J., 2017) propuesta en 1958 por el psicólogo Frank Rosenblatt cuyo mecanismo de aprendizaje está basado en un entrenamiento supervisado (Basogain Olabe X.).

La estructura del Perceptrón se basa en la estructura fundamental de una célula nerviosa, tiene varias entradas cada una asociada a un peso y se tiene una única salida la cual puede ser direccionada a otras neuronas.

En el libro de (T. Hagan M.) se explica que lo que hace realmente una neurona perceptrón es dividir el espacio de entrada para clasificar los patrones mediante una frontera de decisión que puede ser modificada con los pesos sinápticos y el umbral.

2.2.5.3 Red neuronal multicapa

El uso de redes de una sola capa es limitado, para aumentar el grado de precisión se crearon las redes de múltiples capas, (Palma M. J. T., 2008) surge así el MLP que incluye una o varias capas intermedias llamadas capas ocultas. Cuando se añaden neuronas y capas a una red se aumenta, generalmente, su poder de predicción, la calidad de dicha predicción y la capacidad de separación, aunque también se aumenta su tendencia a la sobreespecialización y se aumenta el coste computacional y temporal de entrenamiento (Gironés J., 2017).

Hay muchos métodos y algoritmos para el aprendizaje de redes neuronales uno de ellos es la retropropagación del error (Basogain Olabe X.), tienen capas en las que todas las neuronas de una capa se conectan con todas las neuronas de las capas anterior y posterior, se basa en el principio del ajuste gradual de pesos (Rumelhart D. E., 1986).

Retropropagación en una red con una capa oculta

Las neuronas que forman parte de una red estarán interconectadas y agrupadas en la capa de entrada (reciben información del exterior), la de salida (neuronas que proporcionan el resultado esperado) y la capa oculta (está entre las capas anteriores). El número de neuronas de la capa oculta está relacionado con la capacidad de procesamiento y clasificación, pero un número demasiado grande de neuronas en esta capa aumenta el riesgo de sobreespecialización en el proceso de entrenamiento (Gironés J., 2017) el número de neuronas de la capa de salida depende del problema en casos de clasificación en n grupos se suele emplear n neuronas.

Retropropagación en una red con varias capas ocultas

Basa el aprendizaje de sus pesos en una regla de ajuste de error utilizando el método del descenso del gradiente, son redes unidireccionales de alimentación hacia adelante.

Los errores producidos por la red, y los ajustes de pesos aplicados, son distribuidos entre todas las neuronas en las capas ocultas y de salida. El objetivo es minimizar el error, podemos ver las conexiones en la figura 2.

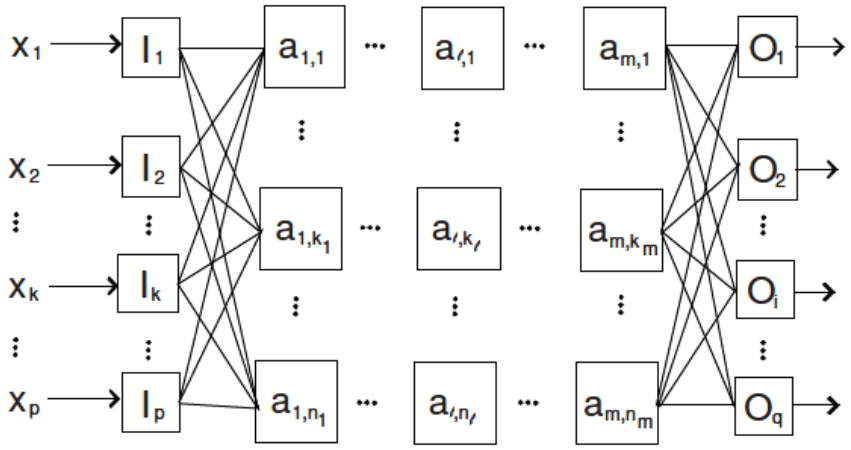


Figura 2 (Pérez Águila, 2012) Red de perceptrones de m capas ocultas con m ≥ 2

La red recibirá (Pérez Aguila, 2012) un vector de entrada en R^p y las neuronas de la capa de entrada propagan sus componentes a todas las neuronas en la primera capa oculta teniendo como salida el vector en R^{n1} gráficamente lo vemos a continuación.

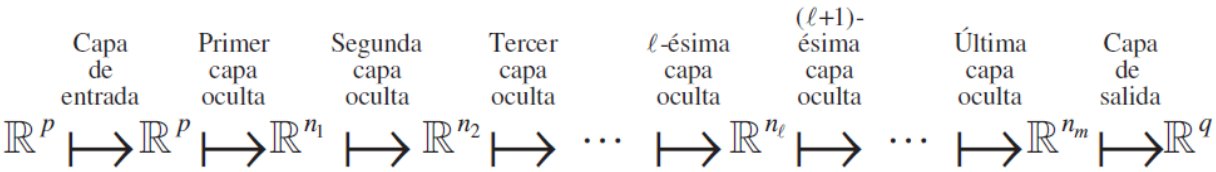


Figura 3 (Pérez Águila, 2012) Propagación de la entrada en una red de multicapas ocultas

2.2.6 Aprendizaje supervisado. Combinación secuencial de clasificadores con base diferente

Como se menciona en (Gironés J., 2017) combinando clasificadores podemos mejorar la predicción de un modelos propone dos técnicas;

- Stacking construye diferentes clasificadores que generan decisiones parciales, se construye un nuevo clasificador usando como entrada las predicciones parciales no los datos de entrada. Estos clasificadores no suelen generalizar bien ante nuevos datos
- Cascading no utiliza solo los datos generados por los clasificadores bases, además de los datos originales pueden usarse datos que han surgido durante la toma de decisiones.

Otros trabajos (Breiman L., Arcing classifiers, 1998) han demostrado que la combinación de múltiples versiones de clasificadores inestables como árboles o redes neuronales ofrecen un error reducido.

En (Tavarez D., 2013) se propone un algoritmo de fusión de dos clasificadores basados en la precisión y la cobertura utilizando sus matrices de confusión.

2.2.6.1 Ensamblado de clasificadores

Existen múltiples algoritmos de aprendizaje automático entre ellos se encuentran;

Bagging (Breiman L.,1996) Puede hacer clasificación y regresión según el aprendizaje deseado.

RandomSubSpace (Ho T. K., 1998) Este método construye un clasificador basado en árbol de decisión que mantiene la mayor precisión en los datos de entrenamiento y mejora la precisión de la generalización a medida que crece en complejidad.

AdaBoos (Yoav Freund R., 1996) es un algoritmo perteneciente a los métodos Boosting, que es un meta algoritmo de aprendizaje automático que reduce el sesgo y varianza dentro de un contexto de aprendizaje supervisado.

Voting (Kuncheva L.I., 2004) y (Kittler J., 1998) combina clasificadores de estimaciones de probabilidad.

Stacking (Wolpert D.H., 1992) Combina varios clasificadores utilizando el método de apilamiento. Puede hacer clasificación o regresión.

2.2.7 Evaluación de modelos clasificadores. Métricas de evaluación.

Se valorarán los clasificadores mediante las siguientes métricas (Powers D., 2007) y (Ron Kohavi F.P., 1998);

2.2.7.1 Verdaderos positivos TP – True Positives

Instancias positivas (de la clase hipertenso) que fueron correctamente clasificadas como hipertenso.

2.2.7.2 Verdaderos negativos TN – True Negatives

Instancias negativas (de la clase normotenso) que fueron correctamente clasificadas como normotenso.

2.2.7.3 Falsos positivos (FP – False Positives)

Instancias negativas (de la clase normotenso) que fueron incorrectamente clasificadas como hipertenso. Pacientes para los cuales se predice que sí tienen la enfermedad y no la tienen.

2.2.7.4 Falsos negativos (FN – False Negatives)

Instancias positivas (de la clase hipertenso) que incorrectamente son clasificadas como normotenso. Pacientes para los cuales no se predice que tienen la enfermedad y sí la tienen, las consecuencias de un FN son más graves como se detalla en (Petticrew M.P., 2000)

2.2.7.5 Matriz de confusión

Se construye con los cuatro términos descritos antes, es una tabla tal que las columnas se refieren a la clase asignada por parte del clasificador mientras que las filas se refieren a la clase real de la instancia. Lo ideal es que todos los elementos de la matriz sean igual a cero excepto la diagonal que refleja las instancias correctamente clasificadas.

		Clase predicha	
		hipertenso	normotenso
Clase real	hipertenso	Verdadero Positivo (TP)	Falso Negativo (FN)
	normotenso	Falso Positivo (FP)	Verdadero Negativo (TN)

Tabla 1

2.2.7.6 Tasa de verdaderos positivos (TP Rate), recall, alcance o sensibilidad (sensitivity).

Es la proporción de instancias positivas correctamente clasificadas. Si TP es el número de instancias positivas correctamente clasificadas y P el total de instancias positivas será:

$$\text{TP rate} = \text{Recall} = \text{Sensibilidad} = \frac{\text{TP}}{\text{P}} \tag{2-3}$$

2.2.7.7 Tasa de verdaderos negativos (TN Rate), especificidad o specificity.

Instancias negativas correctamente clasificadas como negativas. Si TN es el número de instancias negativas correctamente clasificadas y N el total de instancias negativas será:

$$\text{TN rate} = \text{Especificidad} = \text{Specificity} = \frac{\text{TN}}{\text{N}} \tag{2-4}$$

2.2.7.8 Exactitud (accuracy, correctly classified instances)

Porcentaje de instancias del conjunto de datos de prueba que son clasificadas correctamente por el clasificador.

$$\text{exactitud } = \text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}} \tag{2-5}$$

2.2.7.9 Tasa de error (error rate)

Instancias del conjunto de datos de prueba que son clasificadas incorrectamente por el clasificador.

$$\text{Tasa de error } = 1 - \text{accuracy} = \frac{\text{FP} + \text{FN}}{\text{P} + \text{N}} \tag{2-6}$$

2.2.7.10 Precisión

Mide el porcentaje de aquellas instancias que son positivas respecto al total predichas como positivas:

$$\text{Precisión } = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2-7}$$

Es la probabilidad de que para un sujeto enfermo se obtenga en la prueba un resultado positivo. Es, por lo tanto, la capacidad del test para detectar la enfermedad.

2.2.7.11 F-measure

Combina los resultados de recall y precisión.

$$\text{F} = \frac{2 \text{ x precision x recall}}{\text{precision} + \text{recall}} \tag{2-8}$$

2.2.7.12 Kappa statistic

Mide el nivel de acuerde entre la clasificación predicha por el modelo clasificador y la clasificación obtenida en los datos de prueba, corrigiendo el acuerdo que sucede por azar, es

decir mide la probabilidad de que el clasificador clasifique una instancia al azar. Si Pobservado es el acuerdo observado entre el modelo clasificador y la clasificación real, y Pesperado es la probabilidad de acuerdo por casualidad:

$$\text{kappa} = \frac{\text{Pobservado} - \text{P esperado}}{1 - \text{Pesperado}} \quad (2\text{-}9)$$

El valor máximo es de uno y el mínimo es cero (cuando clasifica totalmente al azar)

2.2.7.13 Error absoluto medio y Raíz del error cuadrático medio

Es un promedio de los errores de clasificación de cada una de las instancias. Si tenemos n instancias con unos valores predichos p1, p2, p3…pn, y unos valores reales x1, x2, x3… xn, el error absoluto medio se calcula según la siguiente expresión:

$$MAE = \frac{|p1 - x1| + |p2 - x2| + \cdots + |pn - xn|}{n} \quad (2\text{-}10)$$

$$RMSE = \sqrt{\frac{|p1 - x1|^2 + |p2 - x2|^2 + \cdots + |pn - xn|^2}{n}} \quad (2\text{-}11)$$

2.2.7.14 Error absoluto relativo y la raíz del error cuadrático relativo

Normalizan el resultado, obteniendo un error relativo al error que obtendría un modelo que predice siempre el valor de la media de los valores reales.

$$RAE = \frac{|p1 - x1| + |p2 - x2| + \cdots + |pn - xn|}{|x1 - \bar{x}| + |x2 - \bar{x}| + \cdots + |xn - \bar{x}|} \quad (2\text{-}12)$$

$$RRSE = \sqrt{\frac{|p1 - x1|^2 + |p2 - x2|^2 + \cdots + |pn - xn|^2}{|x1 - \bar{x}|^2 + |x2 - \bar{x}|^2 + \cdots + |xn - \bar{x}|^2}} \quad (2\text{-}13)$$

2.2.7.15 MCC

El coeficiente de correlación de Matthews se utiliza en el aprendizaje automático como una medida de la calidad de las clasificaciones binarias (de dos clases) MCC = 1 representa una predicción perfecta, y un MCC = -1 representa un completo desacuerdo entre la predicción y la observación (Pelaez J.I, 2016).

$$MCC = \frac{TPxTN - FPxFN}{\sqrt{(TP + FP)x(TP + FN)x(TN + FP)x(TN + FN)}} \quad (2\text{-}14)$$

2.2.7.16 Curva ROC

Es la representación gráfica de la sensibilidad frente a la especificidad (Cerda J., 2012) como se ve en la siguiente figura.

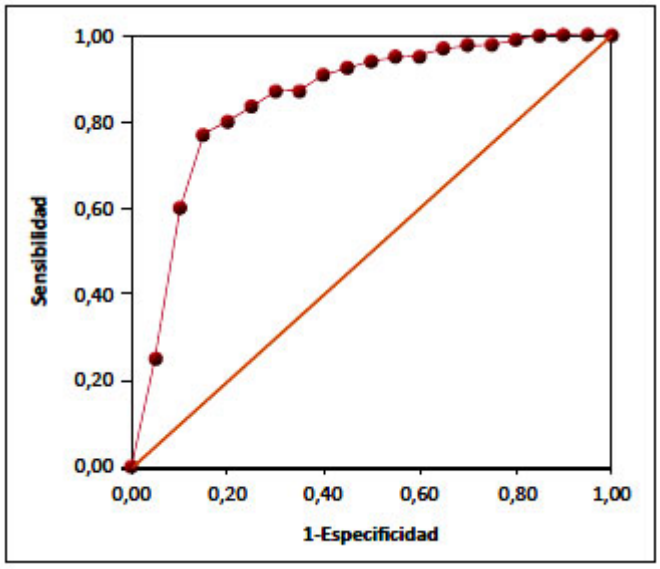


Figura 4 Ejemplo de curva ROC (Cerda J., 2012).

3 Selección y preparación del conjunto de datos

En los siguientes apartados vamos a definir los datos utilizados para realizar el estudio, junto con las distintas herramientas y metodologías.

3.1 Selección datos en la web

Existen múltiples lugares en la web para consultar datos relacionados con la enfermedad en Centers for Disease Control an Prevention (CDC, s.f.) se pueden acceder a numerosos informes estadísticos ((INE, s.f.), (OECD iLibrary, s.f.), (Eurostat, s.f.), (World Bank Open Data, s.f.) , (data Catalog, s.f.)) y recursos de datos relacionados con la presión arterial alta, sus factores de riesgo y condiciones relacionadas donde los usuarios pueden ver mapas de enfermedades cardíacas y accidentes cerebrovasculares a nivel del condado y sus factores de riesgo por grupo racial / étnico, junto con mapas de condiciones ambientales sociales y servicios de salud, para todo Estados Unidos o para un estado o territorio elegido. En el estudio de (Palmer P.A., 2000) se predice el consumo de éxtasis mediante el uso de una red neuronal artificial, del tipo backpropagation capaz de discriminar entre quién consume éxtasis y quién no. Utiliza una muestra de doscientos noventa y seis individuos, veinticinco variables predictoras (el estado civil, el nivel de estudios, la ocupación, con quién viven, relación con los padres, religiosidad, ocio, consumo opinión sobre el éxtasis y personalidad, como variable dependiente que determina la clase si es consumidor o no es consumidor). Es este enfoque lo que lleva a decidir que la base de este trabajo sean los resultados de la encuesta de salud de Castilla y León (Saludcastillayleon, s.f.) disponible en la web, donde se pueden consultar los resultados de distintos problemas o enfermedades crónicas o de larga evolución diagnosticadas por el médico (figura 5) en relación a **sexo** y grupo de **edad** (figura 6), por sexo del sujeto y nivel de **estudios** del cabeza de familia (figura 7), por sexo del sujeto y **clase social** del cabeza de familia (imagen 8), por sexo del sujeto y **situación laboral** del cabeza de familia (figura 9) y por sexo y **tamaño del municipio** de residencia (figura 10).

En la página (Saludcastillayleon, s.f.) se encuentran los datos obtenidos a través de la encuesta regional de salud de Castilla y León, tiene distintos marcadores, entre ellos se encuentra la hipertensión diagnosticada por el médico. Los resultados que se muestran son pacientes normotensos (Clínica Universidad de Navarra, s.f.) e hipertensos (hipertensión, s.f.) relacionados por los atributos de sexo, edad, estudios, clase social, trabajo y tamaño del municipio de residencia. Estos datos se muestran en las siguientes figuras.

La figura 5 muestra como podemos seleccionar distintas enfermedades diagnosticadas por el médico.

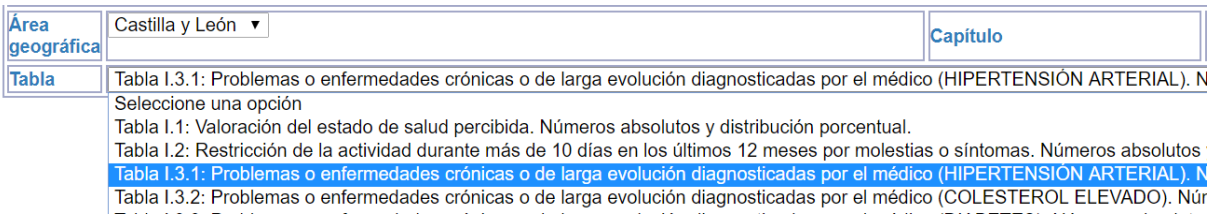


Figura 5 salud castilla y león. (s.f.).

Una vez seleccionada la hipertensión (hipertensión, s.f.) la página se dirige al resultado que muestra las tablas por; sexo y grupo de edad (figura 6), por sexo del sujeto y nivel de estudios del cabeza de familia (figura 7), por sexo del sujeto y clase social del cabeza de familia (figura 8), por sexo del sujeto y situación laboral del cabeza de familia (figura 9) y por sexo y tamaño del municipio de residencia (figura 10).

Tabla I.3.1: Problemas o enfermedades crónicas o de larga evolución diagnosticadas por el médico (HIPERTENSIÓN ARTERIAL). Números absolutos y distribución porcentual.

Por sexo y grupo de edad

		SI		NO		TOTAL	
		N	%	N	%	N	%
HOMBRES	DE 0 A 4 AÑOS	-	-	28.124	100,00	28.124	100,00
	DE 5 A 15 AÑOS	-	-	132.131	100,00	132.131	100,00
	DE 16 A 24 AÑOS	-	-	136.757	100,00	136.757	100,00
	DE 25 A 34 AÑOS	3.795	2,01	185.238	97,99	189.033	100,00
	DE 35 A 44 AÑOS	9.628	5,08	179.978	94,92	189.606	100,00
	DE 45 A 54 AÑOS	15.216	9,27	148.880	90,73	164.096	100,00
	DE 55 A 64 AÑOS	25.176	19,30	105.276	80,70	130.452	100,00
	DE 65 A 74 AÑOS	37.233	28,12	95.177	71,88	132.410	100,00
	DE 75 Y MÁS AÑOS	32.050	31,65	69.204	68,35	101.254	100,00
	SUBTOTAL	123.098	10,23	1.080.765	89,77	1.203.863	100,00
MUJERES	DE 0 A 4 AÑOS	-	-	30.382	100,00	30.382	100,00
	DE 5 A 15 AÑOS	-	-	121.337	100,00	121.337	100,00

Figura 6 salud castilla y león. (s.f)

Tabla I.3.1: Problemas o enfermedades crónicas o de larga evolución diagnosticadas por el médico (HIPERTENSIÓN ARTERIAL). Números absolutos y distribución porcentual.

Por sexo (del sujeto) y nivel de estudios del cabeza de familia

		SI		NO		TOTAL	
		N	%	N	%	N	%
HOMBRES	ANALFABETOS O SIN ESTUDIOS	20.253	15,96	106.626	84,04	126.879	100,00
	PRIMARIOS Y SECUNDARIOS DE PRIMER CICLO	83.153	11,47	641.829	88,53	724.982	100,00
	SECUNDARIOS DE SEGUNDO CICLO Y POST-SECUNDARIOS	10.389	5,10	193.270	94,90	203.659	100,00
	UNIVERSITARIOS	9.304	6,27	139.039	93,73	148.343	100,00
	SUBTOTAL	123.098	10,23	1.080.765	89,77	1.203.863	100,00
MUJERES	ANALFABETOS O SIN ESTUDIOS	36.294	27,65	94.955	72,35	131.248	100,00
	PRIMARIOS Y SECUNDARIOS DE PRIMER CICLO	128.611	18,41	569.921	81,59	698.532	100,00
	SECUNDARIOS DE SEGUNDO CICLO Y POST-SECUNDARIOS	13.872	5,95	219.421	94,05	233.294	100,00
	UNIVERSITARIOS	10.883	6,57	154.688	93,43	165.571	100,00
	SUBTOTAL	189.660	15,44	1.038.985	84,56	1.228.645	100,00
TOTAL	ANALFABETOS O SIN ESTUDIOS	56.546	21,91	201.581	78,09	258.128	100,00
	PRIMARIOS Y SECUNDARIOS DE PRIMER CICLO	211.764	14,88	1.211.750	85,12	1.423.514	100,00
	SECUNDARIOS DE SEGUNDO CICLO Y POST-SECUNDARIOS	24.261	5,55	412.692	94,45	436.953	100,00
	UNIVERSITARIOS	20.186	6,43	293.727	93,57	313.914	100,00
	SUBTOTAL	312.758	12,86	2.119.750	87,14	2.432.508	100,00

Figura 7 salud castilla y león. (s.f.).

Tabla I.3.1: Problemas o enfermedades crónicas o de larga evolución diagnosticadas por el médico (HIPERTENSIÓN ARTERIAL). Números absolutos y distribución porcentual.

Por sexo (del sujeto) y clase social del cabeza de familia

		SI		NO		TOTAL	
		N	%	N	%	N	%
HOMBRES	CLASE I: DIRECTIVOS DE LA ADMINISTRACIÓN PÚBLICA Y DE EMPRESAS DE 10 O MÁS ASALARIADOS. PROFESIONES ASOCIADAS A TITULACIONES DE 2º Y 3ER CICLO	5.163	6,30	76.753	93,70	81.915	100,00
	CLASE II: DIRECTIVOS DE EMPRESAS DE MENOS DE 10 ASALARIADOS. PROFESIONES ASOCIADAS A UNA TITULACIÓN DE 1ER CICLO UNIVERSITARIO. TÉCNICOS SUPERIORES. ARTISTAS Y DEPORTISTAS	5.878	5,53	100.377	94,47	106.255	100,00
	CLASE III: EMPLEADOS DE TIPO ADMINISTRATIVO Y PROFESIONALES DE APOYO A LA GESTIÓN ADMINISTRATIVA Y FINANCIERA. TRABAJADORES DE LOS SERVICIOS PERSONALES Y DE SEGURIDAD. TRABAJADORES POR CUENTA PROPIA. SUPERVISORES DE TRABAJADORES MANUALES	44.172	11,41	342.936	88,59	387.108	100,00
	CLASE IVa:TRABAJADORES MANUALES CUALIFICADOS	41.175	12,23	295.624	87,77	336.799	100,00
	CLASE IVb:TRABAJADORES MANUALES SEMICUALIFICADOS	12.067	8,65	127.500	91,35	139.567	100,00
	CLASE V:TRABAJADORES NO CUALIFICADOS	13.257	9,54	125.758	90,46	139.015	100,00
	CLASE VI: OTROS	1.090	10,41	9.383	89,59	10.473	100,00
	NO CONSTA	295	10,81	2.435	89,19	2.730	100,00
	SUBTOTAL	123.098	10,23	1.080.765	89,77	1.203.863	100,00
MUJERES	CLASE I: DIRECTIVOS DE LA ADMINISTRACIÓN PÚBLICA Y DE EMPRESAS DE 10 O MÁS ASALARIADOS. PROFESIONES ASOCIADAS A TITULACIONES DE 2º Y 3ER CICLO	8.929	10,15	79.051	89,85	87.979	100,00
	CLASE II: DIRECTIVOS DE EMPRESAS DE MENOS DE 10 ASALARIADOS. PROFESIONES ASOCIADAS A UNA TITULACIÓN DE 1ER CICLO UNIVERSITARIO. TÉCNICOS SUPERIORES. ARTISTAS Y DEPORTISTAS	5.806	5,55	98.828	94,45	104.634	100,00

Figura 8 salud castilla y león. (s.f.).

Tabla I.3.1: Problemas o enfermedades crónicas o de larga evolución diagnosticadas por el médico (HIPERTENSIÓN ARTERIAL). Números absolutos y distribución porcentual.
Por sexo (del sujeto) y situación laboral del cabeza de familia

		SI		NO		TOTAL	
		N	%	N	%	N	%
HOMBRES	OCUPADO	35.246	4,68	717.867	95,32	753.113	100,00
	PARADO	3.248	10,50	27.701	89,50	30.949	100,00
	INACTIVO	84.604	20,15	335.197	79,85	419.801	100,00
	SUBTOTAL	123.098	10,23	1.080.765	89,77	1.203.863	100,00
MUJERES	OCUPADO	40.896	5,55	696.156	94,45	737.051	100,00
	PARADO	6.693	22,74	22.744	77,26	29.437	100,00
	INACTIVO	142.071	30,74	320.086	69,26	462.157	100,00
	SUBTOTAL	189.660	15,44	1.038.985	84,56	1.228.645	100,00
TOTAL	OCUPADO	76.141	5,11	1.414.023	94,89	1.490.164	100,00
	PARADO	9.941	16,46	50.444	83,54	60.386	100,00
	INACTIVO	226.675	25,70	655.283	74,30	881.958	100,00
	SUBTOTAL	312.758	12,86	2.119.750	87,14	2.432.508	100,00

Figura 9 salud castilla y león. (s.f.).

Tabla I.3.1: Problemas o enfermedades crónicas o de larga evolución diagnosticadas por el médico (HIPERTENSIÓN ARTERIAL). Números absolutos y distribución porcentual.
Por sexo y tamaño del municipio de residencia

		SI		NO		TOTAL	
		N	%	N	%	N	%
HOMBRES	MENOR O IGUAL A 2.000 HABITANTES	42.775	11,51	328.952	88,49	371.727	100,00
	2.001 A 50.000 HABITANTES	29.408	9,21	289.967	90,79	319.375	100,00
	MÁS DE 50.000 HABITANTES	50.915	9,93	461.847	90,07	512.761	100,00
	SUBTOTAL	123.098	10,23	1.080.765	89,77	1.203.863	100,00
MUJERES	MENOR O IGUAL A 2.000 HABITANTES	66.986	20,32	262.725	79,68	329.711	100,00
	2.001 A 50.000 HABITANTES	50.830	15,18	283.927	84,82	334.758	100,00
	MÁS DE 50.000 HABITANTES	71.844	12,73	492.333	87,27	564.176	100,00
	SUBTOTAL	189.660	15,44	1.038.985	84,56	1.228.645	100,00
TOTAL	MENOR O IGUAL A 2.000 HABITANTES	109.761	15,65	591.678	84,35	701.438	100,00
	2.001 A 50.000 HABITANTES	80.238	12,27	573.894	87,73	654.132	100,00
	MÁS DE 50.000 HABITANTES	122.758	11,40	954.179	88,60	1.076.938	100,00
	SUBTOTAL	312.758	12,86	2.119.750	87,14	2.432.508	100,00

Figura 10 salud castilla y león. (s.f.).

De las anteriores figuras extraeremos los ejemplos que entrenaran a los clasificadores, tendremos 2.432.508 ejemplos con los atributos de edad, nivel de estudios, situación laboral,

clase social, tamaño del municipio de residencia, sexo y como variable dependiente si el individuo está enfermo o no. Podemos pensar que un mayor número de atributos nos puede llevar a una mejor clasificación, pero como se indica en (Blum, 1997) para la construcción de modelos de clasificación se desea utilizar la menor cantidad de atributos posibles, una cantidad excesiva de atributos retrasa significativamente el proceso de aprendizaje y puede producir sobre-ajustes. En (Arona R., 2012) se hace una comparativa entre clasificadores utilizando distintas bases de datos, con un número de atributos que van de cinco a diecinueve dependiendo de la base de datos, lo mostramos en la siguiente figura.

Datasets	Instances	Attributes	No. of Classes	Type
balance-scale	625	5	3	Numeric
diabetes	768	9	2	Numeric
glass	214	10	7	Numeric
lymphography	148	19	4	Nominal
vehicle	946	19	4	Numeric

Figura 11 Características de distintos dataset

En este estudio (Arona R., 2012) se obtienen los siguientes resultados (figura 12).

S.N.	Datasets	J48	MLP
1	balance-scale	76.64	90.72
2	diabetes	73.828	75.391
3	glass	66.822	67.757
4	lymphography	77.027	84.46
5	vehicle	72.459	81.679

Figura 12 Comparativa de clasificación entre J48 y MLP

Como conclusión, entre otros aspectos, destaca que la red tiene una mejor capacidad de aprendizaje, por lo tanto es adecuada para problemas de clasificación por lo que consideramos, en principio, que nuestros atributos pueden ser suficientes para el propósito perseguido.

4 Herramientas

4.1 Herramientas utilizadas en la extracción de datos

Todas están disponibles en la web.

4.1.1 Node.js

Lo utilizamos para establecer la conexión a la página (Saludcastillayleon, s.f.) donde se encuentran los datos que necesitamos los leemos y obtenemos un fichero de salida. Node.js, (Nodejs, s.f.) es un entorno de ejecución para JavaScript, de código abierto, construido con el motor de JavaScript V8 de Chrome, usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, proporciona la herramienta npm (Node Package Manager, (npm, s.f.)) con la que se pueden instalar los paquetes necesarios, que en nuestro caso son;

request. (Request, s.f.). Para establecer conexión, realizar llamadas http. Es compatible con HTTPS y sigue redirecciones por defecto. Puede transmitir un archivo a una solicitud PUT o POST (Hatem H., 2010). Este método también verificará la extensión del archivo contra una asignación de extensiones de archivos a tipos de contenido (en este caso, aplicación / json) y usará el tipo de contenido adecuado en la solicitud PUT (si los encabezados aún no lo proporcionan). Puede transmitir cualquier respuesta a una secuencia de archivos.

La solicitud también puede canalizarse a sí misma. Al hacerlo, el tipo de contenido y la longitud del contenido se conservan en los encabezados PUT.

sanitize-html (Sanitize-html, s.f.). Para mostrar solo el contenido seguro, que los datos procedentes de los usuarios no son seguros. Al contrario, hay muchos usuarios malintencionados que tratan de explotar las vulnerabilidades de seguridad de nuestra aplicación.

Jsonlint (Jsonlint.com, s.f.) Es la herramienta de validación y reformateo en línea para JSON, un formato liviano de intercambio de datos. Informará un error de sintaxis con detalles o imprimirá la fuente si es válida.

js. El módulo fs (File system, s.f.) proporciona una API para interactuar con el sistema de archivos, con esta API se imprime el fichero de salida (Salida código 1).

4.1.2 Visual Studio Code Insiders

Para escribir el código utilizamos Visual Studio Code Insiders, (VisualStudio, s.f.) que es un editor de código redefinido y optimizado para construir y depurar aplicaciones web y en la nube. El código fuente está disponible bajo el acuerdo de licencia de MIT (condiciones que solo requieren la preservación de los derechos de autor y avisos de licencia) y es apto para distintas plataformas: Linux, Mac OSX y Windows.

Regex Previewer, extensión VS que devuelve las coincidencias con una expresión regular definida. Una expresión regular (Aho A.V., 2008) describe a todos los lenguajes que pueden construirse a partir de unos operadores aplicados a los símbolos de cierto alfabeto. Se utilizan para eliminar la información que no es de interés.

4.2 Herramientas utilizadas para convertir los datos a formato arff

4.2.1 Dev-C++

Tanto Weka (weka, s.f.) como RapidMiner (RapidMiner, s.f.) leen archivos arff, Attribute-Relation File Format, este formato de archivo tiene; un encabezado con los comentarios, el nombre de la relación y una lista de los atributos y el cuerpo con los datos.

En la siguiente página mostramos la figura 13 donde se muestran los comentarios, estos son para la comprensión humana (Weka no necesita este aporte para nada), precedidos por el símbolo %.

En la figura 14 tenemos, precedido de @, los parámetros que necesita Weka para entender los datos que va a recibir, el nombre de la relación y los atributos, al estar entre corchetes entiende que son de tipo nominal (Rodríguez Tapia S., 2018)

En la figura 15 vemos el cuerpo precedido de @data, tenemos una matriz con los datos de cada atributo separados por espacios.

```
% ATRIBUTOS
% EDAD;
%       10= DE 0 A 4 AÑOS
%       11= DE 5 A 15 AÑOS
%       12= DE 16 A 24 AÑOS
%       13= DE 25 A 34 AÑOS
%       14= DE 35 A 44 AÑOS
%       15= DE 45 A 54 AÑOS
%       16= DE 55 A 64 AÑOS
%       17= DE 65 A 74 AÑOS
%       18= DE 75 Y MÁS AÑOS
% ESTUDIOS;
%       20= SIN ESTUDIOS
%       21= PRIMER CICLO
%       22= SEGUNDO CICLO
%       23= UNIV
% CLASE SOCIAL;
%       30= CLASE I
%       31= CLASE II
%       32= CLASE III
%       33= CLASE Iva
%       34= CLASEIvb
%       35= CLASE V
%       36= CLASE VI
%       37= NO CONSTA
% TRABAJO;
%       40= OCUPADO
%       41= PARADO
%       42= INACTIVO
% TAMAÑO POBLACION;
%       30= MENOR O IGUAL A 2.000 HABITANTES
%       31= 2.001 A 50.000 HABITANTES
%       32= MÁS DE 50.000 HABITANTES
%       32= MÁS DE 50.000 HABITANTES
% SEXO;
%       100= HOMBRE
%       101= MUJER
% CLASS
%       1=  HIPERTENSO
%       2=  NORMOTENSO
```

Figura 13 Encabezado con los comentarios en el formato arff

```
@relation hypertension
@attribute edad {10,11,12,13,14,15,16,17,18}
@attribute estudios {20,21,22,23}
@attribute claseSocial {30,31,32,33,34,35,36,37}
@attribute trabajo {40,41,42}
@attribute municipio {50,51,52}
@attribute sexo {100,101}
@attribute class {1,2}
```

Figura 14 Encabezado con el nombre de la relación y los atributos

```
@data
10 20 ? 42 ? 100 1
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
10 20 ? 42 ? 100 2
```

Figura 15 Ejemplo de los datos de entrada

Para conseguir este formato utilizamos *Dev-C++* (C++, s.f.) , disponible en la web, es un compilador (Aho A.V., 2008) y entorno de desarrollo para C/C++ (Castillo Sanz A., 2005) publicado bajo licencia libre GNU (gnu.org, s.f.).

4.3 Herramientas utilizadas en clasificación, imputación de datos faltantes y balanceo de datos.

Antes de enumerar las herramientas es necesario que tengamos claro el término imputar, imputar consiste en asignar valores a los datos ausentes ya sea con la media, prediciendo el valor ausente mediante modelos de regresión o haciendo imputaciones múltiples (Canizares M., 2004).

4.3.1 Weka

Weka (weka, s.f.), Waikato Environment for Knowledge Analysis, es una aplicación que contiene una colección de algoritmos de aprendizaje automático y herramientas para el preprocesamiento de datos, clasificación, regresión, clustering, reglas de asociación y visualización. Está implementado en el lenguaje Java que es lenguaje de programación de propósito general, concurrente, orientado a objetos (Allende, 2005).

4.3.2 RapidMiner

RapidMiner (RapidMiner, s.f.) es una plataforma de software para equipos de ciencias de datos que combina la preparación de datos, el aprendizaje automático y la implementación predictiva del modelo. Utilizado en múltiples estudios como por ejemplo en (Jungermann F., 2009) o en (Antonelli D., 2013). Se distribuye bajo licencia AGPL, que es una licencia copyleft derivada de la Licencia Pública General de GNU diseñada específicamente para asegurar la cooperación con la comunidad en el caso de software que corra en servidores de red.

4.4 Herramientas utilizadas en la predicción de nuevos casos utilizando el modelo generado.

Además de DevC++ para solicitar los datos del usuario hemos utilizado Netbeans. (NetBeans, s.f.)

4.4.1 NetBeans

NetBeans es un entorno de desarrollo (un entorno de desarrollo es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas) integrado libre que soporta lenguajes como Java PHP C/C++ Groovy. Bajo licencia Open Source. Pertenece a Oracle.

Lo hemos utilizado para crear un archivo ejecutable de java, al ser abierto carga la aplicación Weka con todos sus componentes.

4.4.2 Launch4j-3.12

Launch4j es un contenedor ejecutable multiplataforma de Java para crear ejecutables nativos de Windows (sourceforge.net, s.f.). Lo hemos utilizado para crear otro ejecutable para aquellas personas que no dispongan de java y deseen usar la aplicación.

4.5 GitHub

GitHub (GitHub, s.f.) es un gestor de repositorios que sirve para controlar el código, sus versiones y otros componentes del proyecto, con la posibilidad de volver atrás en caso de error o si queremos ver como hemos ido evolucionando.

4.6 Typora

Hemos utilizado el editor Typora (Typora, s.f.) que utiliza el lenguaje de marcado ligero markdown (Voegler J., 2014) para escribir las instrucciones y el readme que hay en GitHub.

5 Descripción del proyecto

5.1 Extracción de datos web.

Se desarrolla una pequeña aplicación cliente, disponible en el repositorio indicado al inicio del documento en la carpeta DescargaDatos, si queremos acceder al archivo el enlace es https://github.com/monteroy/TFG/blob/master/00DescargaDatos/descargarDatos_1.js

Utilizamos el lenguaje Javascript (Mozilla, s.f.) que puede ser aplicado a un documento HTML (w3schools, s.f.) y usarlo para crear interactividad dinámica en los sitios web.

La aplicación la hacemos con el editor Visual Studio Code Insider (VisualStudio, s.f.), lee los datos de la página (Saludcastillayleon, s.f.) y escribe un fichero de texto plano un array con solo los datos numéricos, ya que hemos eliminado los elementos que no queremos con expresiones regulares. Una expresión regular es un tipo de notación para la definición de lenguajes (Hopcroft J.E.).

```
datosJSON = datosJSON.replace(/\[/, '')
```

Figura 16 Ejemplo de expresión regular

En la anterior figura se muestra una de las expresiones regúlales utilizadas, con ella eliminamos el símbolo [cuando se encuentre en la lectura de la página, así encadenando sucesivas expresiones logramos nuestros datos.

En la siguiente figura se muestra la función de escritura descrita anteriormente.

```
fs.writeFileSync('salidaJson.json', datosJSON);
```

Figura 17 Función de escritura en JavaScript

Las tres siguientes figuras muestran; la 18 el formato que se puede ver al abrir la página (Saludcastillayleon, s.f.) con un navegador, subrayado el dato que se desea, la figura 19 muestra el dato coloreado en el array de salida y la 20 el software una vez ejecutado.

		SI		NO		TOTAL	
		N	%	N	%	N	%
HOMBRES	DE 0 A 4 AÑOS	-	-	28.124	100,00	28.124	100,00
	DE 5 A 15 AÑOS	-	-	132.131	100,00	132.131	100,00
	DE 16 A 24 AÑOS	-	-	136.757	100,00	136.757	100,00
	DE 25 A 34 AÑOS	3.795	2,01	185.238	97,99	189.033	100,00
	DE 35 A 44 AÑOS	9.628	5,08	179.978	94,92	189.606	100,00
	DE 45 A 54 AÑOS	15.216	9,27	148.880	90,73	164.096	100,00
	DE 55 A 64 AÑOS	25.176	19,30	105.276	80,70	130.452	100,00
	DE 65 A 74 AÑOS	37.233	28,12	95.177	71,88	132.410	100,00
	DE 75 Y MÁS AÑOS	32.050	31,65	69.204	68,35	101.254	100,00
	SUBTOTAL	123.098	10,23	1.080.765	89,77	1.203.863	100,00

Figura 18 Elementos seleccionados por la aplicación descargaDatos_1.js

```
0;28124;28124;0;132131;
132131;0;136757;136757;
3795;185238;189033;9628
;179978;189606;15216;14
8880;164096;25176;10527
6;130452;37233;95177;13
2410;32050;69204;101254
```

Figura 19 Parte del fichero de salida de la aplicación descargaDatos_1.js

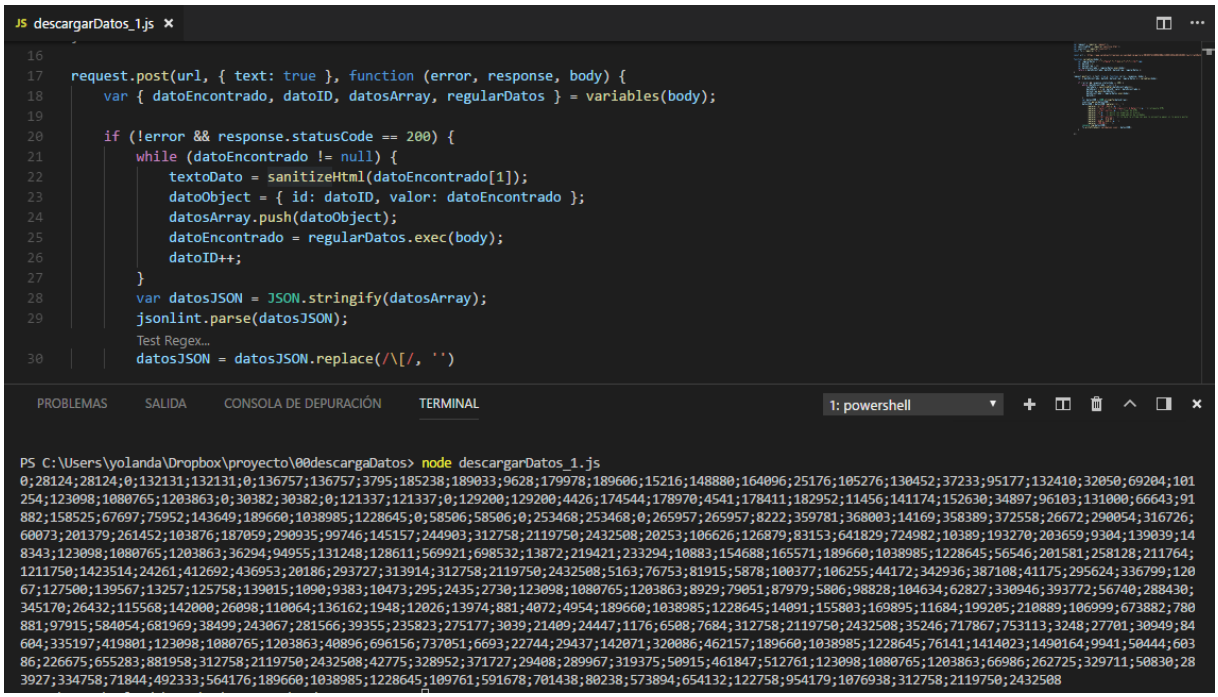


Figura 20 Vista general del software de extracción de datos

5.2 Conversión de datos a formato arff.

Tras el proceso anterior pasamos a utilizar el software Devp++ (C++, s.f.) y el lenguaje de programación C (Draft, 2007) creamos un código, que explicaremos a continuación, disponible en https://github.com/monteroy/TFG/tree/master/01TratamientoDatos/codigo/Datos.dev.

Utilizando el archivo de la salida anterior preparamos los datos para tener una matriz. Se comienza leyendo el archivo resultante del anterior código, si la lectura no produce error se recogen los datos en un array, dado que se desconoce el tamaño del fichero se utiliza asignación de memoria dinámica (Joyanes Aguilar L., 2005). Para asignar memoria dinámica se utilizan punteros que indican la dirección de memoria donde se encuentra el dato almacenado. Hemos escrito distintas funciones para ordenar y separar los ejemplos. Se usa la función de salida fwrite() que escribe un buffer de cualquier tipo de para tener la matriz resultante, esta matriz en cada columna contendrá los distintos atributos y la última columna tendrá la clases es decir si el paciente ha sido diagnosticado o no como hipertenso, cada fila será un ejemplo (un paciente). La siguiente tabla muestra el encabezado de nuestros datos.

Edad	Estudios	Clase Social	Trabajo	Municipio	Sexo	Clase

Tabla 2

El motivo de que la última columna contenga la clase es que beneficia la lectura de datos en Weka y en RapidMiner es indiferente puede leer el formato de entrada de Weka arff (waikato, s.f.).

El array obtenido contiene para cada atributo; la cantidad de enfermos que si sufren hipertensión, cantidad de enfermos que no la sufren y en la tercera posición la suma de estas cantidades, esta última cifra es irrelevante para el estudio y se deshecha con un bucle. Codificamos los atributos y sus posibles valores del siguiente modo:

EDAD
- 10= DE 0 A 4 AÑOS
- 11= DE 5 A 15 AÑOS
- 12= DE 16 A 24 AÑOS
- 13= DE 25 A 34 AÑOS
- 14= DE 35 A 44 AÑOS
- 15= DE 45 A 54 AÑOS
- 16= DE 55 A 64 AÑOS
- 17= DE 65 A 74 AÑOS
- 18= DE 75 Y MÁS AÑOS

ESTUDIOS
- 20= SIN ESTUDIOS
- 21= PRIMER CICLO
- 22= SEGUNDO CICLO
- 23= UNIVERSITARIOS

CLASE SOCIAL
- 30= CLASE I
- 31= CLASE II
- 32= CLASE III
- 33= CLASE Iva
- 34= CLASE Ivb
- 35= CLASE V
- 36= CLASE VI
- 37= NO CONSTA

TRABAJO
- 40= OCUPADO
- 41= PARADO
- 42= INACTIVO

TAMAÑO POBLACIÓN
- 50= MENOR O IGUAL A 2.000 HABITANTES
- 51= 2.001 A 50.000 HABITANTES
- 52= MÁS DE 50.000 HABITANTES

SEXO

100= HOMBRE
101= MUJER

CLASE

1= HIPERTENSO
2= NORMOTENSO

Veamos con más detalle el proceso, el array original se almacena en dos estructuras, separando pacientes enfermos y sanos ya que siguiendo el orden de lectura de los datos conocemos esta información y si se trata de hombres o mujeres ya que las tablas disponen de la suma de hombres y la suma de mujeres los distintos atributos son controlados con sucesivas estructuras repetitivas, mostramos a continuación un ejemplo grafico de estas estructuras.

```
while (sumaClaseNoEdadHombre != datos_init[auxClase+1] ||
datos_init[auxClase+1] == 0){
    if (auxClase%2 ==0 ){
        atributos[contadorWhile].Edad= edadHombreSi++;
        atributos[contadorWhile].Estudios=0;
        atributos[contadorWhile].Trabajo=0;
        atributos[contadorWhile].claseSocial=0;
        atributos[contadorWhile].Municipio=0;
        atributos[contadorWhile].sexo =100;
        atributos[contadorWhile].clase =1;
        atributos[contadorWhile].cantidad=datos_init[auxClase];
        sumaClaseSiEdadHombre += datos_init[auxClase];
        filas +=  datos_init[auxClase]/NUM_TABLAS;
        filasTotal+=datos_init[auxClase];
        contadorWhile++;
    }else if (auxClase%2 !=0 ){
        atributos[contadorWhile].Edad= edadHombreNo++;
        atributos[contadorWhile].Estudios=0;
        atributos[contadorWhile].Trabajo=0;
        atributos[contadorWhile].claseSocial=0;
        atributos[contadorWhile].Municipio=0;
        atributos[contadorWhile].sexo =100;
        atributos[contadorWhile].clase =2;
        atributos[contadorWhile].cantidad=datos_init[auxClase];
        sumaClaseNoEdadHombre += datos_init[auxClase];
        filas +=  datos_init[auxClase]/NUM_TABLAS;
        filasTotal+=datos_init[auxClase];
        contadorWhile++;
    }
    auxClase++;
}
```

Figura 21 Almacenamiento cantidad de enfermos y sanos que son hombres

Cuando la suma acumulada coincide con el total del array almacenado saltamos al siguiente atributo.

Finalmente se asigna para cada caso una fila con los atributos que le corresponden mediante estructuras de control, un bucle for, un while por cada atributo y dentro de estos sentencias if seguidas de cláusulas else para evaluar distintos casos . Para el atributo de nivel de estudios se puede inferir que los menores de dieciséis años no poseen estudios (BOE.), su muestra la asignación en la siguiente figura.

```
                case 1:
                    if (atributos[contadorStruct].Edad==10 ||
atributos[contadorStruct].Edad==11){

    matrizEntradaPerceptron[contadorFilasMatriz][contadorColumnasMatriz]=
20;//los menores de 16 no tienen estudios acabados
                    }else{

    matrizEntradaPerceptron[contadorFilasMatriz][contadorColumnasMatriz]=
atributos[contadorStruct].Estudios;
                    }

                    contadorColumnasMatriz++;
                    break;
```

Figura 22 Evaluación con condición para estudios en menores de dieciséis años

Para el atributo de trabajo los menores de dieciséis y mayores de sesenta y cinco contarán como inactivos (BOE, 2015), se muestra la asignación en la siguiente figura.

```
                case 3:
                    if (atributos[contadorStruct].Edad==10 ||
atributos[contadorStruct].Edad==11||atributos[contadorStruct].Edad==17
||atributos[contadorStruct].Edad==18){

    matrizEntradaPerceptron[contadorFilasMatriz][contadorColumnasMatriz]=
42;//los mayores de 65 estan inactivos y lo menores de 16 no pueden trabaja
                    }else{

    matrizEntradaPerceptron[contadorFilasMatriz][contadorColumnasMatriz]=
atributos[contadorStruct].Trabajo;
                    }

                    contadorColumnasMatriz++;
                    break;
```

Figura 23 Evaluación con condición para trabajo en menores de dieciséis años y mayores de sesenta y cinco

Desde esta estructura mediante estructuras de control se rellena la matriz, dada la cantidad de filas que tiene la matriz inicial también es necesario trabajar con memoria dinámica. El

principal rasgo de esta parte es que para datos distintos de cero se generan tantas filas como indique el atributo cantidad.

Por último se vuelca en un fichero la matriz para ello se usa la función de salida fwrite() que escribe un buffer de cualquier tipo de dato para tener un archivo con los datos preparados para que los lea Weka o RapidMiner. Los datos ausentes son identificados mediante el símbolo '?' como podemos ver en la figura 25 que muestra esa parte del código.

```
            if
(matrizEntradaPerceptron[contadorFilasMatriz][contadorColumnasMatriz] == 0)
                fprintf(matriz, "? ");
            }else{
                fprintf(matriz, "%d
",matrizEntradaPerceptron[contadorFilasMatriz][contadorColumnasMatriz] );

            }
        }
```

Figura 24 Marcado de datos ausentes en la salida

El resultado es una matriz de 2.432.508 filas y 7 columnas (los atributos), podemos verlo en la figura 25.

class	edad	estudios	claseSocial	trabajo	municipio	sexo
1	10	20	?	42	?	100
2	10	20	?	42	?	100
2	10	20	?	42	?	100
2	10	20	?	42	?	100
2	10	20	?	42	?	100
2	10	20	?	42	?	100
2	10	20	?	42	?	100
2	10	20	?	42	?	100
2	10	20	?	42	?	100

Figura 25 Muestra de la matriz obtenida importada con la herramienta RapidMiner

El código se encuentra dentro del proyecto Datos.dev, en el archivo de nombre main.c https://github.com/monteroy/TFG/tree/master/01TratamientoDatos/0%20codigo.

5.3 Mejor validación

Al aplicar un algoritmo de clasificación en Weka tenemos un conjunto de test para elegir;

5.3.1 Use training set

Weka entrenará con todos los datos y luego lo aplicará otra vez sobre los mismos datos la evaluación

5.3.2 Supplied test set

Clasifica con un fichero de datos y evalúa con otro fichero de datos, en nuestro caso no tenemos los datos separados por lo que no lo usamos hasta que lleguemos al proceso de predicción, que será cuando introduciremos los datos sin el valor de clase y tendremos como resultado la clase predicha.

5.3.3 Cross-validation

Es una validación cruzada estratificada del número de particiones dado (folds). Consiste en dado un número n se divide los datos en n partes y, por cada parte, se construye el clasificador con las n−1 partes restantes y se prueba con esa parte.

5.3.4 Percentage Split

Se dividen los datos en dos grupos, de acuerdo con el porcentaje indicado (%), el valor indicado es el porcentaje de instancias para construir el modelo, que seguidamente es evaluado sobre las que se han dejado aparte.

Hacemos una primera clasificación con la matriz de salida con los métodos descritos utilizando el clasificador basado en reglas ZeroR en (Wahbeh A.H., 2011) se indica que es un método rápido y hace unas buenas primeras aproximaciones.

Método de validación utilizado	Instancias correctamente clasificadas
Use training set	87.1428 %
Cross-validation	87.1428 %
Percentage Split	87.1626 %

Tabla 3

En nuestro caso parece que no es determinante el método que usemos para hacer la validación. Los resultados pueden verse en los archivos que se encuentran en la carpeta https://github.com/monteroy/TFG/tree/master/01TratamientoDatos/1%20Resultados%20distintas%20validaciones.

5.4 Imputación de datos

Es el momento de recordar que nuestras variables tienen la mayoría de atributos declarados como valores ausentes, en (Farhangfar A., 2008) se realiza estudio experimental que muestra que la imputación de valores faltantes mejora la precisión de la clasificación en comparación con la clasificación sin imputación. Algunos clasificadores como C4.5 y Naïve-Bayes pueden producir una clasificación precisa en presencia de datos faltantes, mientras que otros clasificadores como K-vecino más cercano (K-nearest-neighbor) (Peterson L. E., 2009), SVM y RIPPER se beneficia de la imputación.

En nuestro caso no podemos analizar casos completos por lo que la solución pasa por realizar imputaciones, para ellos trabajamos con Weka y RapidMiner, el primero por haber sido usado alguna vez durante los estudios y el segundo por ser una herramienta desconocida hasta este momento.

Weka ofrece tres filtros para el caso que nos ocupa, uno que reemplaza valores ausentes según una probabilidad establecida, otro que los reemplaza por una constante y otro que los reemplaza por la moda (Montero Lorenzo J.M., 2007) como nuestros datos son nominales este es el adecuado, el filtro se llama ReplaceMissingValues.

La salida se muestra en la siguiente figura:

Histograma datos originales | Histograma después del filtro en Weka

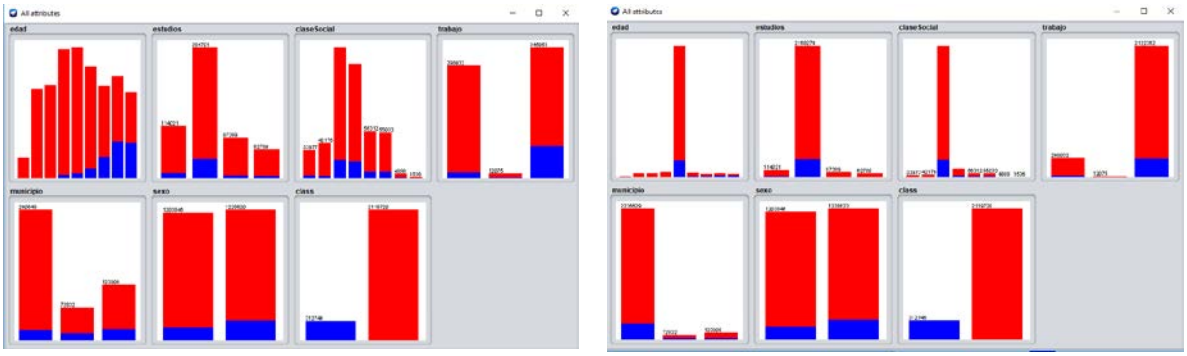


Figura 26 Filtros en Weka

Como se puede ver en la figura anterior al estimar los datos faltantes con la moda se produce un sesgo importante en ellos, para evitar esto se pueden recurrir a distintas técnicas en (P.J. & J.L., 2008) se señala que para las bases de datos reales existen numerosos datos incompletos, siendo una solución la imputación mediante el uso del algoritmo Knn (Peterson L. E., 2009) la clasificación de K-vecino más cercano se desarrolló a partir de la necesidad de

realizar análisis discriminante cuando las estimaciones paramétricas confiables de las densidades de probabilidad son desconocidas o difíciles de determinar.

En (Tibshirani R., 1999) se presentan tres métodos para realizar imputaciones, el primero de ellos es la técnica de aproximación de bajo rango mediante la descomposición de valores singulares (SVD, Singular Value Decomposition), el segundo está basado en K-neares-neighbor y el último está basado en sucesivas regresiones con el estándar EM, algoritmo esperanza-maximización (Fernández-Alonso, 2012). También es posible inferir mediante métodos bayesianos (Silva L.C., 2001).

La herramienta RapidMiner nos ofrece distintos algoritmos para poder reemplazar los valores ausentes mediante la función impute missing values, a continuación vemos, figura 27, como muestra esta aplicación la técnica de imputar valores.

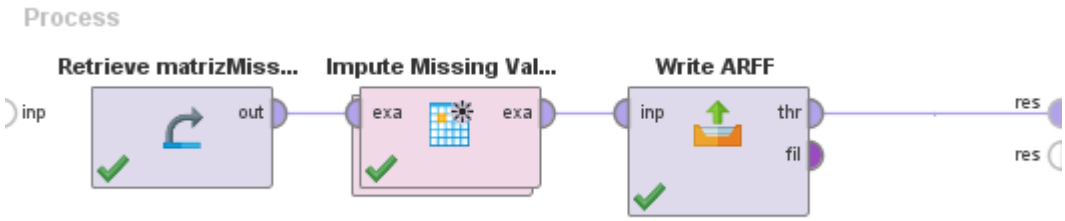


Figura 27 Imputación de valores ausentes mediante la herramienta RapidMiner

En la figura 28 vemos que existen diferencias en los histogramas entre imputar los datos mediante dos técnicas distintas.

Histograma después del filtro con la moda en Weka

Histograma con valores imputados con RapidMiner y Decision Tree

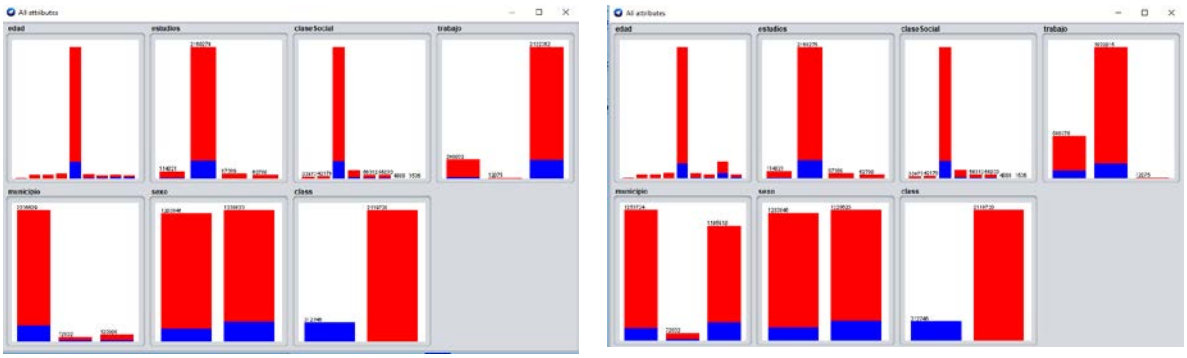


Figura 28 Histogramas comparados en resultados de imputación de datos

También es posible convertir nuestros datos nominales a numéricos, llamado proceso de numerización, inverso a la discretización (Pérez Lopez C.), para tratar de imputar los datos ausentes con otros métodos que utilizan datos numéricos.

Realizamos la imputación de valores con la distintas técnicas explicadas anteriormente para después evaluar los resultados el algoritmo J48 (que es la implementación del algoritmo j4.5 (Quinlan J., 1993)).

Una vez imputados los datos utilizamos el algoritmo j48 disponible en Weka con validación cruzada para ver una primera clasificación de las distintas técnicas aplicadas, los resultados se muestran en la siguiente tabla y se pueden consultar los modelos y resultados en https://github.com/monteroy/TFG/tree/master/01TratamientoDatos/2%20input%20data%20sin%20balancear%20datos. Esta tabla muestra las instancias clasificadas correctamente y las incorrectas separadas en Falsos Negativos y Falsos positivos, si no separásemos parecería que tenemos una tasa de clasificación aceptable, pero vamos a observar la tabla y extraer conclusiones.

Técnica	Instancias Correctamente Asignadas	Instancias Incorrectamente Asignadas. Falsos Negativos (hipertensos clasificados como normotensos)	Instancias Incorrectamente Asignadas. Falsos Positivos (normotensos clasificados como hipertensos)
CHAID	87.8032 %	12.1968 %	0.0000 %
Decision Strump	87.8032 %	12.1968 %	0.0000 %
Decisión Tree	87.8032 %	12.1968 %	0.0000 %
Gradient Boosted Trees	87.8032 %	12.1968 %	0.0000 %
Random Tree	87.8032 %	12.1968 %	0.0000 %
RandomForest	87.8032 %	12.1968 %	0.0000 %
Naive Bayes	87.8032 %	12.1968 %	0.0000 %
Moda	87.8032 %	12.1968 %	0.0000 %
Deep Learning	87.8032 %	12.1968 %	0.0000 %
Knn	87.8032 %	12.1968 %	0.0000 %

Tabla 4

De los resultados anteriores podemos deducir que **los algoritmos ignoran** totalmente una de las clases, la minoritaria, los datos tal y como están en este momento **no se pueden usar**. Gráficamente se refleja en la figura 29.

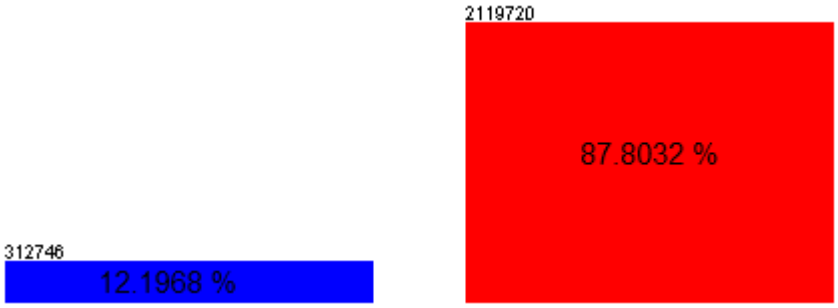


Figura 29 Porcentajes de la clasificación hecha

Un conjunto de dos clases se dice que es no balanceado si una de las clases (mayoritaria) está significativamente más representada que la otra (minoritaria) es lo que ocurre en nuestro caso, tenemos 312.746 hipertensos frente a 2.119.720 normotensos lo que supone algo más del 87 por ciento de peso hacia la clase mayoritaria, entonces es fácil pensar que la mala aproximación hasta el momento de la clasificación se deba a esto. En (García Jiménez V., 2010) se muestran distintas líneas de investigación para tratar el desbalance; técnicas de re-muestreo, métodos de clasificación en entornos no balanceados y combinación de estrategias. En (Barandela R., 2004) se indica que para tratar el problema, el caso de conjuntos con un alto ratio de desbalance, las técnicas de sobremuestreo son la mejor opción, existen algoritmos dirigidos a incrementar la talla de la clase minoritaria como son; SMOTE (Chawla N.V., 2002), Bordeline SMOTE (Han H., 2005), LLE SMOTE (Wang J., 2006) o la generación de nuevos datos a partir de algoritmos de clustering (Cohen G., 2006).

5.4.1 Datos desbalanceados

Weka dispone del filtro Weka.filters.supervised.instance.SMOTE que muestrea el conjunto de datos aplicando la técnica de sobremuestreo de minorías mediante la incorporación de muestras artificiales generadas por interpolación, utiliza el procedimiento de k-vecinos (Peterson L. E., 2009). El problema del sobreaprendizaje se evita al extender la frontera de la clase minoritaria hacia la región mayoritaria, a través de la creación de ejemplos de los cuales aprender. Hacemos igual que en el caso anterior una primera clasificación con el algoritmo J48 con validación cruzada para cada una de las técnicas empleadas, los resultados se muestran en la siguiente tabla en la que también hemos separado la tasa de errores, veamos que ocurre.

Técnica	Instancias Correctamente Asignadas	Instancias Incorrectamente Asignadas. Falsos Negativos (hipertensos clasificados como normotensos)	Instancias Incorrectamente Asignadas. Falsos Positivos (normotensos clasificados como hipertensos)
Decision Strump	67.5436 %	17.3759 %	15.0805 %
Decision Tree	67.1366 %	23.5851 %	9.2783 %
Gradient Boosted Trees	67.6773 %	23.0444 %	9.2783 %
Random Tree	67.5436 %	17.3759 %	15.0805 %
Random Forest	67.6773 %	23.0444 %	9.2783 %
Naive Bayes	**73.8261 %**	**9.4672 %**	**16.7067 %**
Moda	68.7038 %	10.8318 %	20.4644 %
Deep Learning	67.3740 %	23.3476 %	9.2784 %

Tabla 5

Tras aplicar el algoritmo el número de instancias ha aumentado hasta igualar las dos clases, como podemos observar ya no tenemos resultados desbalanceados y distintos algoritmos arrojan distintos resultados, siendo la mejor imputación la realizada con la técnica de Naive Bayes, ya no solo por el porcentaje de instancias correctamente clasificadas si no porque nos ofrece una tasa de falsos negativos bastante más baja que el resto. Los resultados están dentro de la carpeta llamada input data con datos balanceados, se pueden consultar en https://github.com/monteroy/TFG/tree/master/01TratamientoDatos/3%20input%20data%20con%20datos%20balanceados.

Al hacer una clasificación con estos datos y el MLP el equipo deja de responder, como solución a este problema recurrimos a la reducción de instancias. La eliminación de instancias no tiene porqué producir una degradación de los resultados (Herrera F., 2006), ya que se pueden estar reduciendo ejemplos repetidos o ruido con distintas técnicas, como son; los muestreos aleatorios, los muestreos estratificados, por agrupamiento, los muestreos sistemáticos, los muestreos dobles, los muestreos enlazados, los muestreos inversos o los muestreos progresivos.

Otros algoritmos se basan en los vecinos más cercanos, WIRS (Vallejo C.G., 2004). RapidMiner (RapidMiner, s.f.) ofrece varios filtros para volver a muestrear los datos y generar

un nuevo conjunto de datos reducido en tamaño, que tiene en cuenta la distribución de clase. Hasta ahora el mejor resultado lo tenemos con la asignación de datos faltantes con la técnica de Naive Bayes (Jonh G. H., 1995), utilizamos estos datos completos para generar otros muestreos y ver si obtenemos mejores clasificaciones. En la siguiente figura vemos como gráficamente RapidMiner nos muestra la técnica en la figura 30.

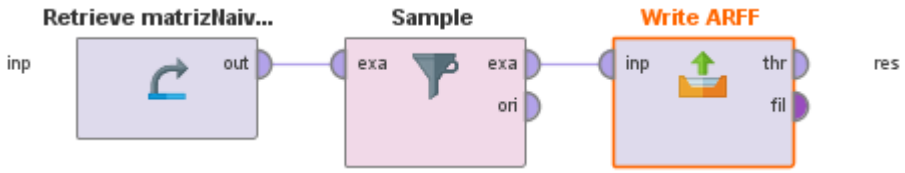


Figura 30 Filtro Sample para reducción de instancias

5.4.1.1 Reducción de datos

Realizamos la reducción de datos con distintas técnicas;

Sample

Crea una muestra de un conjunto de ejemplos seleccionando ejemplos aleatoriamente. El tamaño de una muestra se puede especificar en base absoluta, relativa y de probabilidad (cada ejemplo tiene la probabilidad señalada de ser seleccionado en los datos de salida).

Muestreo estratificado

Crea subconjuntos aleatorios y asegura que la distribución de clases en los subconjuntos sea la misma que en todo el conjunto de ejemplos. Este operador no se puede aplicar a conjuntos de datos sin una etiqueta o con una etiqueta numérica. El tamaño de la muestra puede especificarse en términos absolutos y relativos.

Hacemos nuevamente una primera clasificación con el algoritmo j48 para ver si de algún conjunto de datos se tiene mejor clasificación, los resultados reducidos tienen el mismo o similar número de instancias, se muestran en la siguiente tabla separando la tasa de errores como venimos haciendo hasta ahora.

Técnica	Instancias Correctamente Asignadas	Instancias Incorrectamente Asignadas. Falsos Negativos (hipertensos clasificados como normotensos)	Instancias Incorrectamente Asignadas. Falsos Positivos (normotensos clasificados como hipertensos)
Sample_Probabilidad	**73.8261 %**	**9.4672 %**	**16.7067 %**
Sample_Absoluto	73.8051 %	9.5021 %	16.6928 %
Sample_Relativo	73.8055 %	9.5017 %	16.6928 %
Estratificado_Absoluto	73.8383 %	9.4298 %	16.7319 %
Estratificado_Relativo	73.8031 %	9.4873 %	16.7095 %

Tabla 6

Tenemos el mejor resultado con la reducción de datos mediante sample por probabilidades, los resultados se encuentran en la carpeta input data datos balanceados, disponible en https://github.com/monteroy/TFG/tree/master/01TratamientoDatos/3%20input%20data%20con%20datos%20balanceados, veamos qué ocurre si aumentamos la probabilidad o disminuimos la probabilidad en el algoritmo, los resultados se muestran en la siguiente tabla.

Probabilidad	Instancias Correctamente Asignadas	Instancias Incorrectamente Asignadas. Falsos Negativos (hipertensos clasificados como normotensos)	Instancias Incorrectamente Asignadas. Falsos Positivos (normotensos clasificados como hipertensos)
1 %	72.4227 %	9.4126 %	18.1647 %
5 %	**73.8447 %**	**9.4078 %**	**16.7475 %**
10 %	73.8383 %	9.4298 %	16.7319 %
15 %	73.8327 %	9.4544 %	16.7129 %
20 %	73.8131 %	9.4642 %	16.7227 %

Tabla 7

El dataset generado con la técnica de balanceo sample por probabilidad al 5% será el utilizado para entrenar el MLP ya que es el que mejores resultados ha obtenido, este resultado contiene 211.972 ejemplos lo que además nos ayudará a reducir los tiempos de computo.

Los resultados se pueden consultar en la carpeta remplazo probabilidades. https://github.com/monteroy/TFG/tree/master/01TratamientoDatos/3%20input%20data%20con%20datos%20balanceados/bayesianos/imput%20naive%20bayes/remplazo%20probabilidades.

La figura 31 muestra uno de los árboles generados en estas clasificaciones, no se muestra totalmente extendido ya que no tendríamos una imagen global.

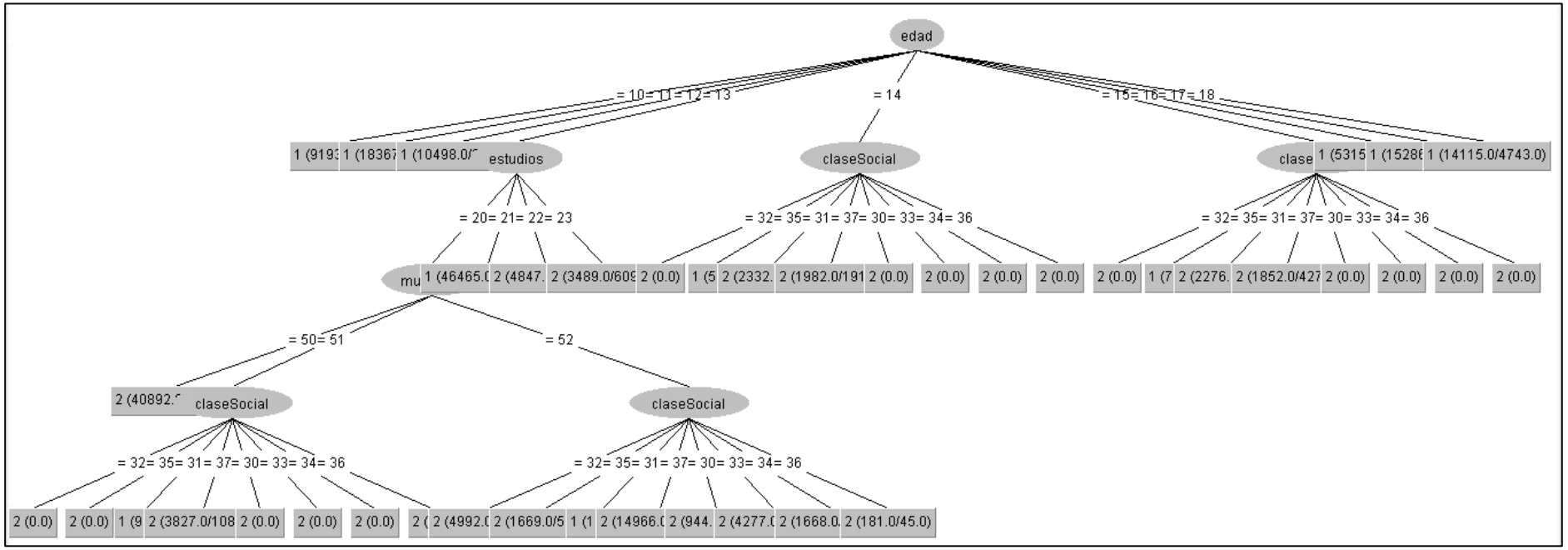


Figura 31 Árbol generado por Weka en la clasificación de sample por probabilidad 5%

5.5 Selección de atributos

Weka incluye numerosos filtros que se pueden utilizar antes de invocar a un clasificador para limpiar el conjunto de datos o modificarlo de alguna manera.

Aplicamos ClassifierAttributeEval con J48, MLP y bayes.NaiveBayes, ninguno de estos algoritmos descarta ninguno de nuestros atributos, los resultados se pueden consultar en https://github.com/monteroy/TFG/tree/master/01TratamientoDatos/Resultados%20seleccion%20atributos/ranker%2BclassifierAttributeEval.

5.6 Red Neuronal con Weka

Utilizaremos Weka que implementa el perceptrón multicapa con backpropagation para clasificar instancias con la clase MultilayerPerceptron. La red puede ser monitoreada y modificada durante el tiempo de entrenamiento. Los nodos en esta red son todos sigmoide (excepto cuando la clase que es numérica, en cuyo caso los nodos de salida se convierten en unidades lineales no restringidas). Tiene distintos parámetros que pueden ser modificados, son los siguientes;

Tasa de aprendizaje para el algoritmo de backpropagation. (El valor debe estar entre 0 - 1, Predeterminado = 0.3).

Velocidad de impulso para el algoritmo de retropropagación. (El valor debe estar entre 0 - 1, Predeterminado = 0.2).

Número de épocas a las cuales entrenar. (Predeterminado = 500).

Tamaño de porcentaje del conjunto de validación. (El valor debe estar entre 0 - 100, Predeterminado = 0).

El valor utilizado para inicializar el generador de números aleatorios.

Las capas ocultas por defecto pueden ser; ('a' = (attribs + clases) / 2, 'i' = attribs, 'o' = clases, 't' = attribs + clases), como valor predeterminado = a).

5.6.1 Resultados de la clasificación con MLP

El entrenamiento de redes neuronales se controla mediante diversos parámetros, como hemos detallado anteriormente, a estos parámetros les hemos asignado la siguiente nomenclatura;

- E es el número de épocas, iteraciones que realizamos con el algoritmo
- T.A la tasa de aprendizaje, controla el ajuste de las ponderaciones en las distintas épocas.
- C son las capas ocultas
- O el número de neuronas en las capas ocultas
- V la velocidad para el algoritmo de retropropagación
- I son las instancias correctamente clasificadas
- K es el estadístico kappa
- TP son los verdaderos positivos
- FP los falsos positivos
- P la precisión
- TN los verdaderos negativos
- FN los falsos negativos

En las casillas de TP, FP, TN y FN nos encontramos con dos medidas, el uno señala el porcentaje de la clase hipertensos clasificados por la red y el dos al porcentaje clasificados como normotensos.

Aunque el aumento de épocas de entrenamiento implique un incremento exponencial de tiempo y un mínimo incremento en la mejora de los resultados según el artículo de (Kala P., 2012) decidimos ir aumentando el número de épocas ya que en ese estudio el número de épocas es muy alto y el número de datos muy pequeño, 40 set input, nosotros en comparación tenemos una cantidad de datos mucho más alta. Además, para ver si nuestro algoritmo aprendía, también hemos modificado la tasa de aprendizaje, la velocidad del algoritmo de retropropagación y el número de capas ocultas.

Los distintos resultados y modelos de redes están disponibles para su consulta en https://github.com/monteroy/TFG/tree/master/02Resultados/Resultados%20MLP y la tabla siguiente muestra los resultados siguiendo la nomenclatura descrita más arriba.

E	T.A	C	O	V	I %	K	TP %	FP %	P %	TN %	FN %
1	0.3	0		0.2	70.8606	0.42	1= 69.0	1= 27.3	1= 71.8		1= 15.5
							2= 72.7	2= 31.0	2= 70.0	2= 13.6	
1	0.2	0		0.2	71.1618	0.42	1= 70.4	1= 28.0	1= 71.6		1=14.8
							2= 72.0	2= 29.6	2= 70.7	2=14.0	
1	0.1	0		0.2	71.6703	0.43	1= 72.3	1= 29.0	1= 71.5		1=14.5
							2= 71.0	2= 27.7	2=71.9	2=13.9	
1	0.01	0		0.2	73.4951	0.47	1=80.5	1=33.6	1=70.7		1=9.8
							2=66.4	2=19.5	2=77.3	2=16.7	
1	0.001	0		0.2	72.3591	0.45	1=78.3	1=33.6	1=70.1		1=10.9
							2= 66.4	2= 21.7	2= 75.3	2= 16.7	
1	0.01	0		0.1	73.4951	0.47	1=80.5	1=33.6	1=70.7		1=9.8
							2=66.4	2=19.5	2=77.3	2=16.7	
1	0.01	0		0.3	73.5196	0.47	1=80.6	1=33.6	1=70.7		1=9.7
							2= 66.4	2= 19.4	2= 77.3	2= 16.7	
100	0.01	0		0.2	73.5498	0.47	1=80.6	1=33.6	1=70.7		1=9.7
							2= 66.4	2= 19.4	2= 77.4	2= 16.7	
500	0.01	0		0.2	73.5498	0.47	1=80.6	1=33.6	1=70.7		1=9.7
							2= 66.4	2= 19.4	2= 77.4	2= 16.7	
100	0.01	1	2	0.2	73.6989	0.47	1=80.5	1=33.1	1=70.9		1=9.8
							2= 66.9	2= 19.5	2= 77.3	2= 16.5	
100	0.01	1	4	0.2	73.8437	0.47	1=81.2	1=33.6	1=70.8		1=9.4
							2= 66.4	2= 18.8	2= 77.9	2=16.7	
100	0.01	1	8	0.2	73.8428	0.48	1=81.2	1=33.6	1=70.8		1=9.4
							2= 66.4	2= 18.8	2= 77.9	2= 16.7	
100	0.01	1	15	0.2	73.8437	0.48	1=81.2	1=33.6	1=70.8		1=9.4
							2= 66.4	2= 18.8	2= 77.9	2=16.7	
100	0.01	2	4,2	0.2	73.8395	0.48	1=81.3	1=33.6	1=70.8		1=9.4
							2= 66.4	2= 18.7	2= 77.9	2= 16.8	
100	**0.01**	**2**	**4.4**	**0.2**	**73.8447**	**0.48**	**1=81.2**	**1=33.6**	**1=70.8**		**1=9.4**
							2= 66.4	**2=18.8**	**2= 77.9**	**2= 16.7**	
100	0.01	2	8,4	0.2	73.8447	0.48	1=81.2	1=33.6	1=70.8		1=9.4
							2= 66.4	2=18.8	2= 77.9	2= 16.7	

Tabla 8

Podemos ver que en los sucesivos entrenamientos hemos conseguido mejorar los resultados cambiando los parámetros y que llega un momento en que nuestra red no logra “aprender” más. Los mejores resultados los obtenemos con una red que realiza 100 iteraciones, con una tasa de aprendizaje de 0.01, dos capas ocultas de cuatro nodos cada y una velocidad de 0.2. Nuestra mejor red se muestra en la siguiente figura.

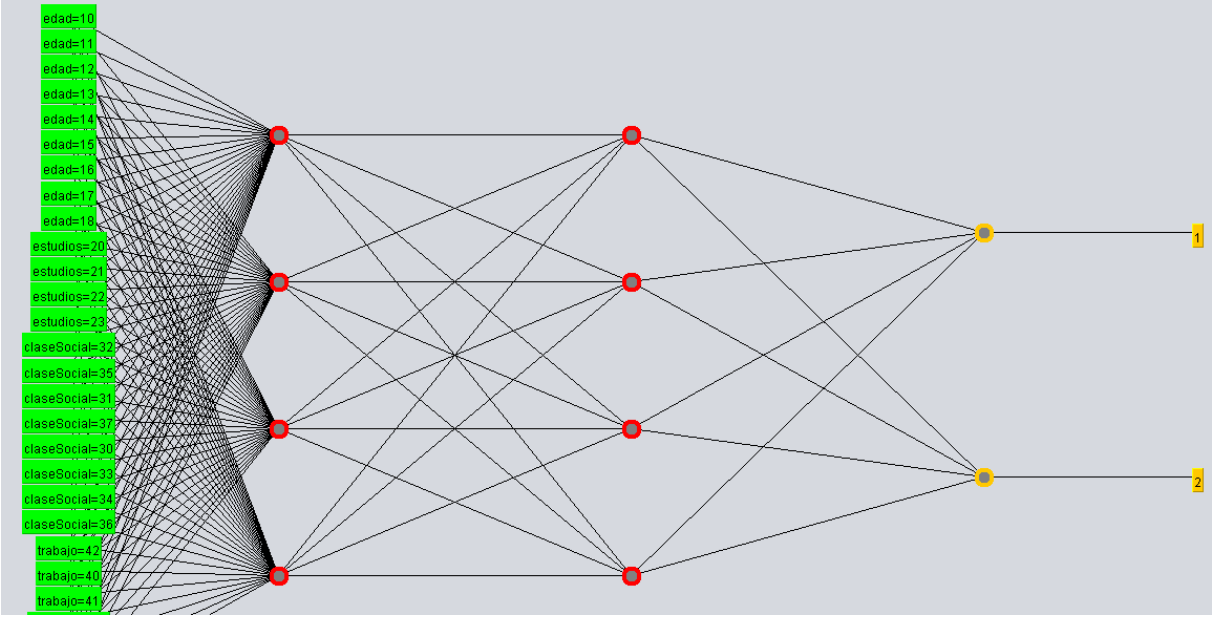


Figura 32 Red neuronal de dos capas ocultas con cuatro neuronas por capa

En la figura podemos ver como cada valor posible para cada atributo se conecta con la primera capa oculta de cuatro nodos (la función de activación y transferencia son transparentes en la figura), todos estos nodos se conectan a la siguiente capa oculta y estos últimos cuatro nodos a los dos de salida que son las clases, a medida que avanzan las épocas podemos ver como cambia el error, lo mostramos en las figuras 33 y 34.

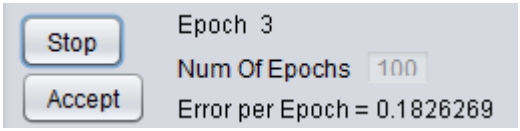


Figura 33 Error en la época 3

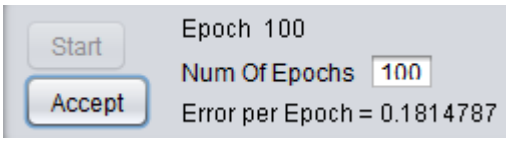


Figura 34 Error en la época 100

5.7 Ejecución de ensambladores

Weka también dispone de una interfaz gráfica con la cual lanzar los experimentos, que usamos para aplicar las técnicas que hemos comentado. La siguiente figura muestra que hace cada uno de los iconos.

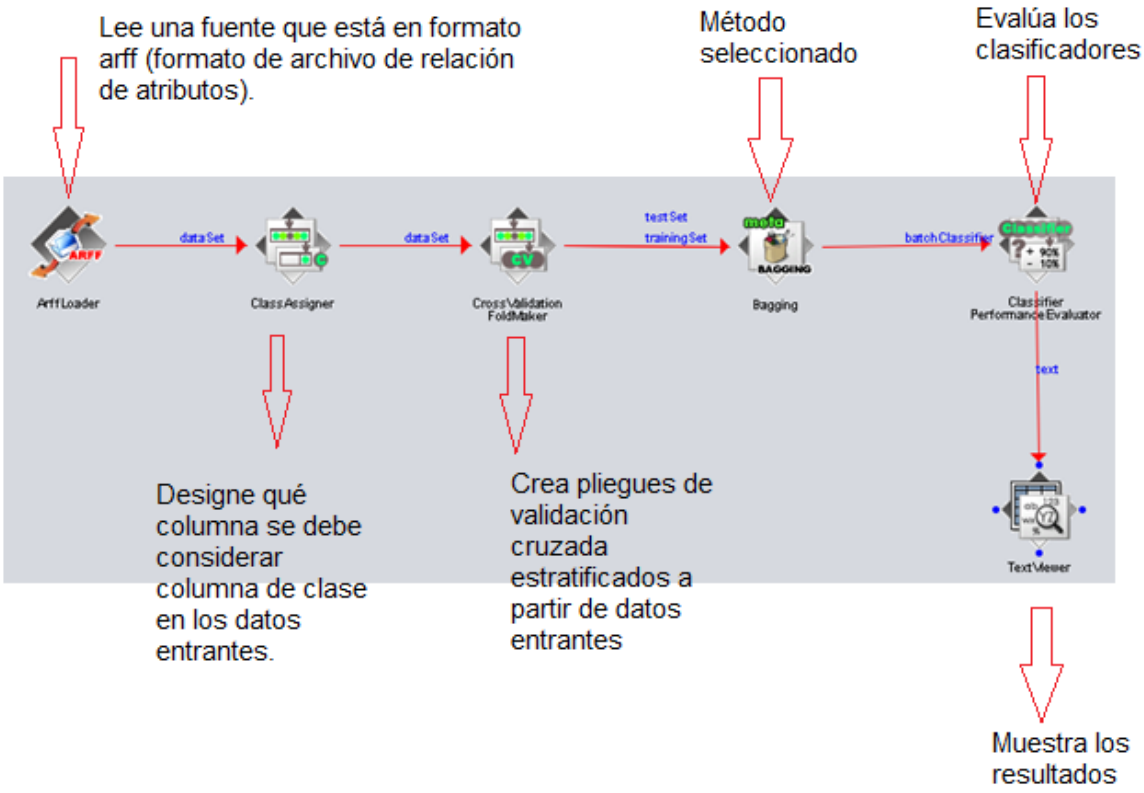


Figura 35 Acciones llevadas a cabo por cada icono

Mostramos los distintos ensamblados de las figuras 36 a 40 y explicamos las combinaciones que tienen.

5.7.1 Bagging

Utilizamos el modelo generado por el MLP junto con un árbol de decisión J48 que es la implementación del algoritmo J4.5 (Quinlan J., 1993).

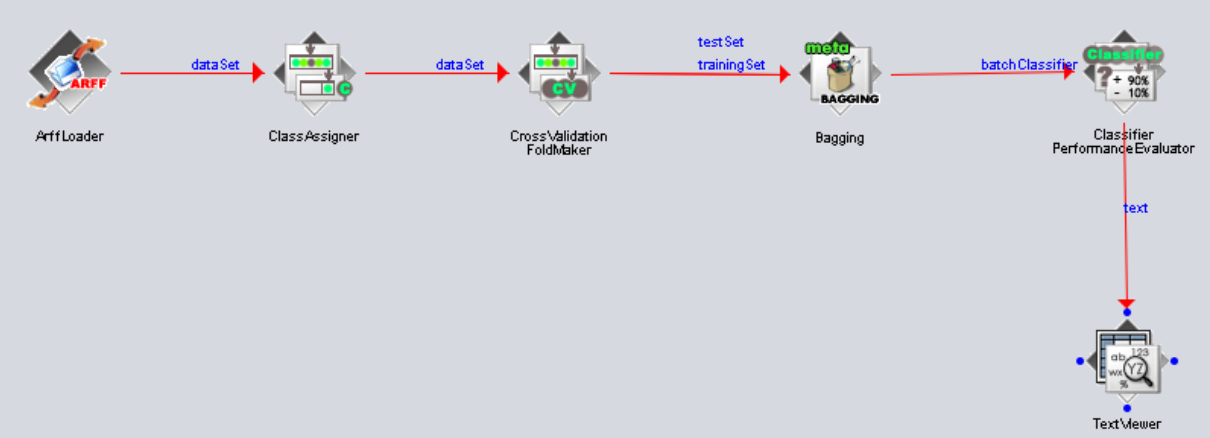


Figura 36 Baggin

5.7.2 RandomSubSpace

Utilizamos el modelo generado por el MLP junto con el Random Tree que es la clase que construye un árbol, considera K atributos elegidos al azar en cada nodo. No realiza ninguna poda También tiene una opción para permitir la estimación de las probabilidades de clase (o la media del objetivo en el caso de regresión) en función de un conjunto de retención (retroadaptación (Ingeniería, s.f.)).

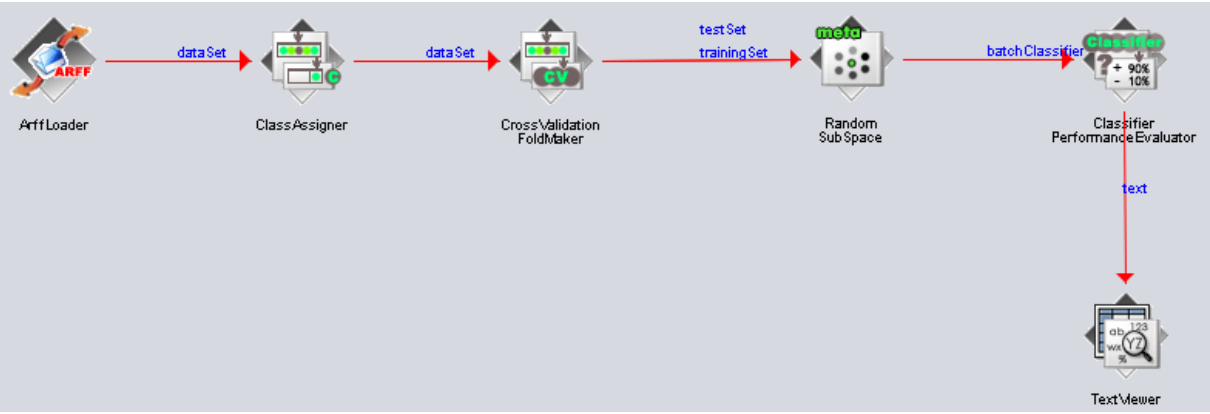


Figura 37 RandomSubSpace

5.7.3 AdaBoos

Utilizamos el modelo generado por el MLP junto con un árbol de decisión Strump que hace la clasificación mediante el cálculo del error cuadrático medio y la entropía de la información. La entropía es una medida de como está organizado un conjunto de datos en un sistema cerrado y trabaja con la proporción de entradas que pertenece a cada clase dado un atributo (Rényi A., 1961). La entropía caracteriza la heterogeneidad de un conjunto de ejemplos.

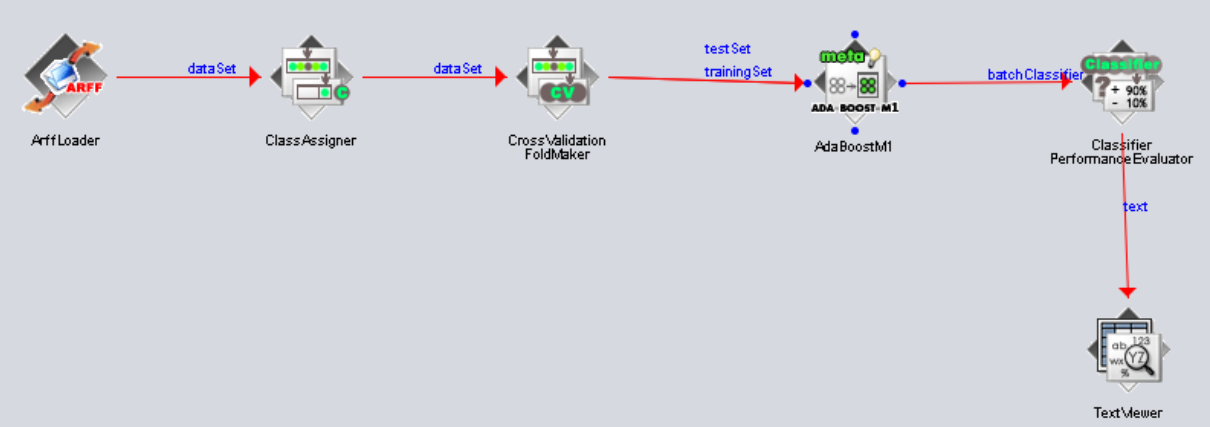


Figura 38 AdaBoos

5.7.4 Voting

Utilizamos el modelo generado por el MLP junto con el algoritmo de reglas de decisión JRip que hace el uso reglas proposicionales con poda incremental repetida para producir reducción de errores (Cohen W. W., 1995).

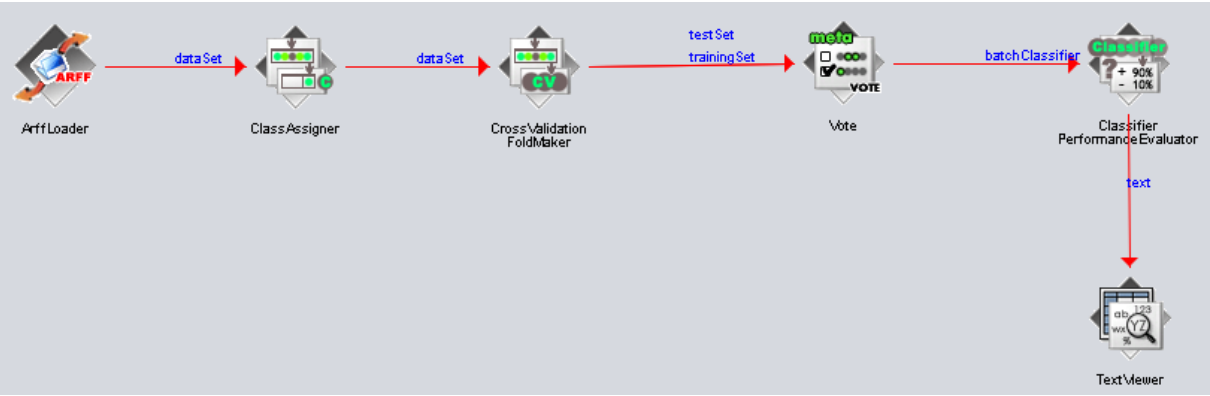


Figura 39 Voting

5.7.5 Stacking

Utilizamos el modelo generado por el MLP también con el algoritmo de reglas de decisión JRip (Cohen W. W., 1995).

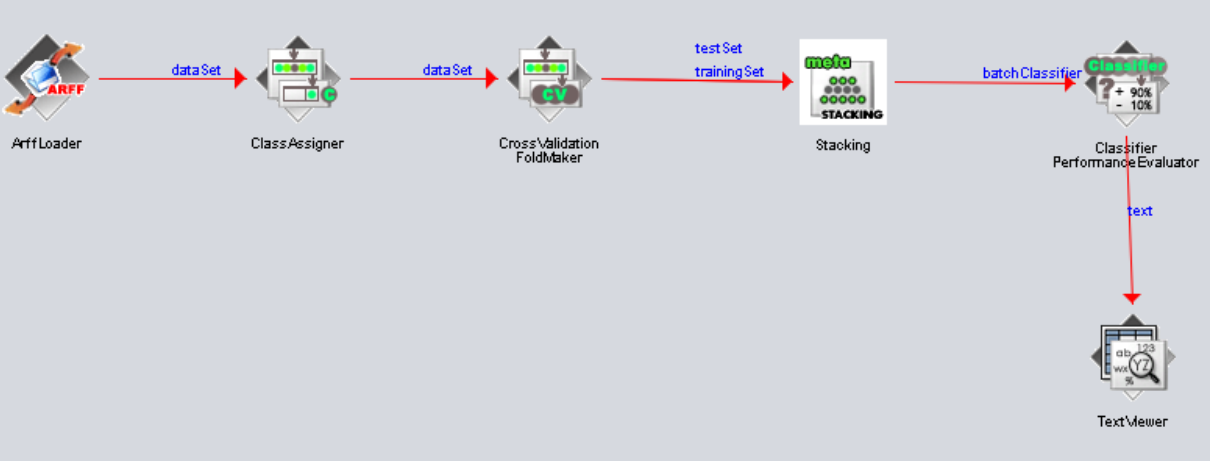


Figura 40 Stacking

5.7.6 Resultados del ensamblado de clasificadores

Hemos ejecutado los distintos algoritmos, los resultados y modelos generados están disponibles para su consulta en la carpeta de ensamblado de nuestro repositorio https://github.com/monteroy/TFG/tree/master/02Resultados/Resultados%20Ensamblado.

La siguiente tabla los detalla, primero figura la técnica empleada, el porcentaje correcto de instancias clasificadas y el porcentaje incorrecto de instancias clasificadas, separados en falsos negativos y falsos positivos como venimos haciendo hasta ahora.

Técnica	Instancias Correctamente Asignadas (Verdaderos positivos y verdaderos negativos)	Instancias Incorrectamente Asignadas. Falsos Negativos (hipertensos clasificados como normotensos)	Instancias Incorrectamente Asignadas. Falsos Positivos (normotensos clasificados como hipertensos)
Bagging	**73.8447 %**	**9,4078 %**	**16,7475 %**
RandomSubSpace	73.7192 %	9,3522 %	16,9287 %
AdaBoos	72.2312 %	11,0213 %	16,7475 %
Voting	73.7838 %	9,4003 %	16,8159 %
Stacking	73.8074 %	9,4451 %	16,7475 %

Tabla 9

Como podemos ver la técnica de AdaBoos, que hemos aplicado junto con un árbol de decisión Strump, tiene un resultado significativamente peor que las demás técnicas y este error de casi un dos por ciento repercute directamente sobre las personas que realmente son enfermos.

El resto de los resultados son bastante similares entre sí, vemos que la mejor clasificación se hace con la técnica de Baggin, aplicada junto con el árbol de decisión J48 y que esta iguala su resultado con el resultado del MLP.

Como conclusión en este apartado podemos decir que **ninguna** de las técnicas empleadas **supera** el nivel de clasificación del MLP.

6 Evaluación

6.1 Resultados

Nuestros mejores resultados, los obtenidos con la clasificación del MLP, se reflejan en la figura 41, aquí podemos ver lo evaluadores que hemos descrito anteriormente, instancias correcta e incorrectamente clasificadas, el estadístico kappa, el error absoluto medio, etc.

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

Correctly Classified Instances      156530               73.8447 %
Incorrectly Classified Instances     55442               26.1553 %
Kappa statistic                          0.4767
Mean absolute error                      0.3617
Root mean squared error                  0.4249
Relative absolute error                 72.3435 %
Root relative squared error             84.9799 %
Total Number of Instances           211972

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,812    0,336    0,708      0,812    0,757      0,482    0,794     0,795     1
                 0,664    0,188    0,779      0,664    0,717      0,482    0,794     0,782     2
Weighted Avg.    0,738    0,262    0,744      0,738    0,737      0,482    0,794     0,788

=== Confusion Matrix ===

     a     b   <-- classified as
 86278 19942 |     a = 1
 35500 70252 |     b = 2
```

Figura 41 Resultados clasificación MLP

Como podemos ver el mejor resultado arroja un 73.8 % de instancias bien clasificadas y un 24.2 % de instancias mal clasificadas.

Evaluamos el resto de resultados para tener un mejor entendimiento de la solución, para ello comenzamos analizando el estadístico kappa (2-9), la siguiente tabla corresponde a la escala utilizada con frecuencia para expresar cualitativamente la fuerza de la concordancia, según (Richard Landis J., 1977).

Coeficiente Kappa	Fuerza de la concordancia (acuerdo)
< 0.00	Pobre (Poor)
0.00-0.20	Leve (Slight)
0.21-0.40	Aceptable (Fair)
0.41-0.60	Moderada (Moderate)
0.61-0.80	Considerable (Substantial)
0.81-1.00	Casi perfecta (Almost perfect)

Tabla 10

Nuestro resultados es de 0.47 por lo que el índice de acuerdo es moderado.

Error medio absoluto es de (2-10) 0.3617, Weka asigna un cero si predice la clase correcta y un uno en caso de que sea predicha la clase incorrecta su raíz (2-11) tiene el valor de 0.4249.

Las siguientes dos métricas que analizamos son el error absoluto relativo (2-12) y la raíz del error cuadrático relativo (2-13) Weka calcula la media asumiendo valores numéricos para las clases «si» y «no», los valores obtenidos resultan descompensados por el hecho de que no estamos trabajando con valores numéricos y estas métricas se deben utilizar para predicciones numéricas.

La tasa de verdaderos positivos (2-3) es del 73.8 %, de verdaderos negativos (2-4) 26.2 % una precisión (2-5) del 74.4 %.

De la matriz de confusión podemos observar que las instancias incorrectamente asignadas, falsos negativos (hipertensos clasificados como normotensos) ascienden al 9,4% y los falsos positivos al 16,7%, mostramos la matriz de confusión que se muestra tras la clasificación en la siguiente figura.

```
    a     b   <-- classified as
86278 19942 |     a = 1
35500 70252 |     b = 2
```

Figura 42 Falsos negativos

El MCC (2-14) es de 0.428 para ambas clases, está compensado recordando que su valor está entre -1 y 1 el nivel de acuerdo es bastante alto y el área de la curva ROC se eleva hasta el 79.4%.

6.2 Uso del modelo generado

Podemos usar nuestro modelo predictor directamente desde el repositorio, para ello hemos creado un código disponible en la carpeta Weka exe del repositorio (llamado Weka.exe) https://github.com/monteroy/TFG/tree/master/05%20Weka%20exe. Este código solicita las variables al usuario y genera un fichero de formato arff, el tipo de clase, por defecto, es desconocido, se muestra la salida que genera a continuación (figura 43).

```
@relation hypertension
@attribute edad {10,11,12,13,14,15,16,17,18}
@attribute estudios {20,21,22,23}
@attribute claseSocial {30,31,32,33,34,35,36,37}
@attribute trabajo {40,41,42}
@attribute municipio {50,51,52}
@attribute sexo {100,101}
@attribute class {1,2}
@data
16 23 31 40 52 101 ?
```

Clase desconocida

Figura 43 Fichero arff con los datos de un usuario

Hemos cargado Weka en NetBeans con la ayuda del archivo Weka-src.jar, con el entorno de desarrollo integrado NetBeans hemos generado un ejecutable llamado Weka.jar, el código se puede consultar en la carpeta Weka java, también hemos creado un ejecutable .exe con la ayuda de Launch4j para las personas que quieran usar el software sin java, ambos ejecutables se encuentran en la carpeta Weka exe. La figura 44 muestra el aspecto de Weka en el entorno NetBeans.

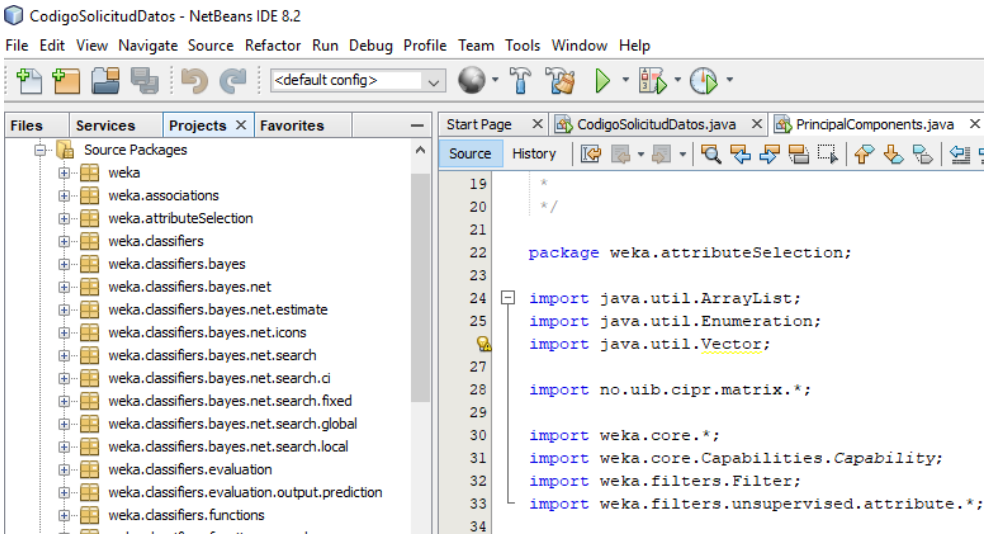


Figura 44 Weka en el entorno NetBeans

Mostramos como se interactúa con el software Launch4j para tener un ejecutable con la extensión exe en la figura 45, donde indicamos el origen, el destino, el manifiesto (colección de datos que describen como se relacionan entre sí los elementos del ensamblado (microsoft, s.f.)), y un icono que ilustra la salida.

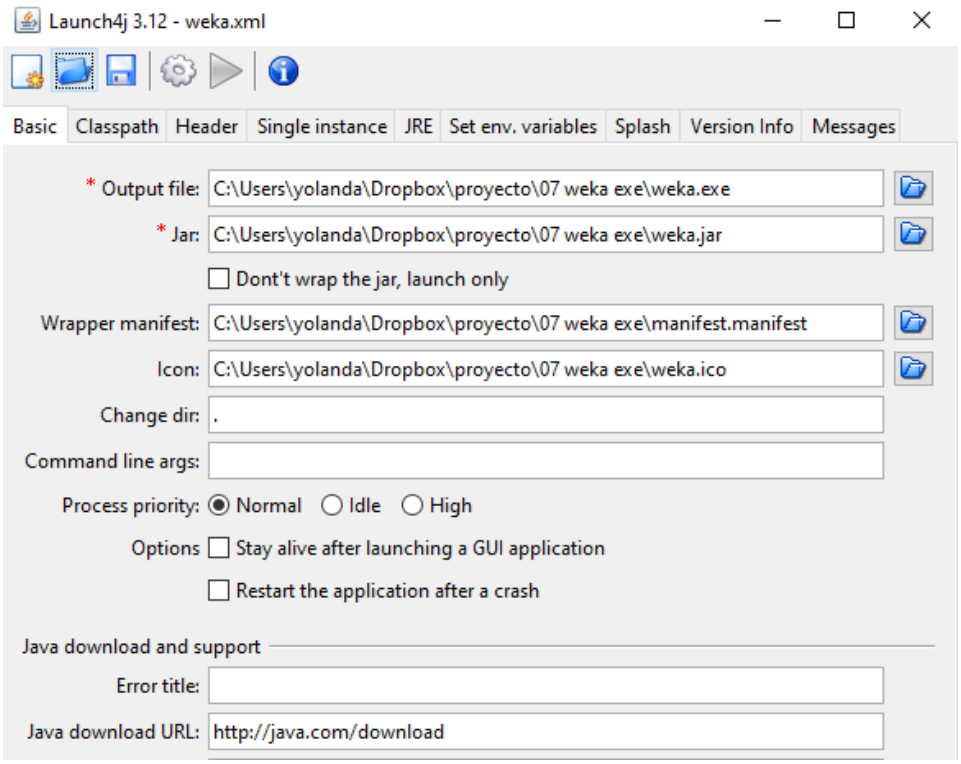
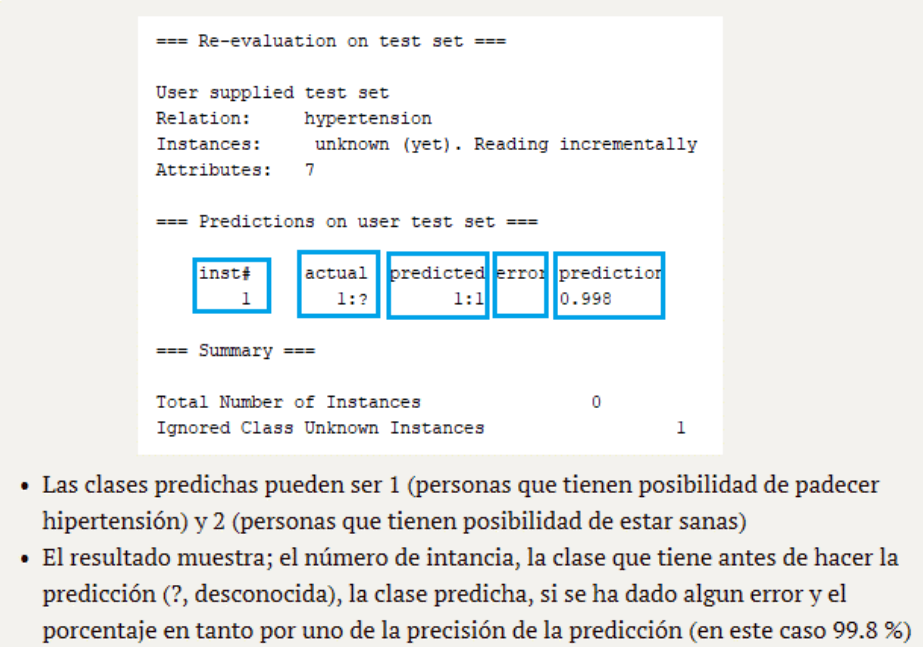


Figura 45 Ejemplo de uso de Launch4j

Finalmente redactamos las instrucciones para poder usar el modelo generado con el MLP, estas instrucciones se pueden consultar en el repositorio en el apartado https://github.com/monteroy/TFG/blob/master/06%20instrucciones/instrucciones.pdf.

Finalizan con la explicación de la predicción de la clase, que ha sido generada a partir de los datos introducidos por el usuario, en la figura 46 vemos como se muestra.



```
=== Re-evaluation on test set ===

User supplied test set
Relation:     hypertension
Instances:     unknown (yet). Reading incrementally
Attributes:   7

=== Predictions on user test set ===

 inst#     actual  predicted error prediction
    1        1:?        1:1       0.998

=== Summary ===

Total Number of Instances                0
Ignored Class Unknown Instances                  1
```

- Las clases predichas pueden ser 1 (personas que tienen posibilidad de padecer hipertensión) y 2 (personas que tienen posibilidad de estar sanas)
- El resultado muestra; el número de intancia, la clase que tiene antes de hacer la predicción (?, desconocida), la clase predicha, si se ha dado algun error y el porcentaje en tanto por uno de la precisión de la predicción (en este caso 99.8 %)

Figura 46 Salida de la predicción

7 Conclusiones y trabajo futuro

Citando (Pita Fernández S., 2010) “Resulta obvio que lo ideal sería trabajar con pruebas diagnósticas de alta sensibilidad y especificidad, pero esto no siempre es posible. En general, las pruebas de screening deben ser de alta sensibilidad para poder captar a todos los enfermos. Una prueba muy sensible será especialmente adecuada en aquellos casos en los que el **no diagnosticar** la enfermedad puede resultar **fatal** para los enfermos, como ocurre con enfermedades peligrosas pero tratables, como los linfomas o la tuberculosis, o en enfermedades en las que un falso positivo no produzca serios trastornos psicológicos o económicos para el paciente (por ejemplo, la realización de mamografía en el cáncer de mama).” En nuestro experimento la sensibilidad alcanzada para la clase 1 es alta, un 81.2 %, recordemos que son los enfermos que serían diagnosticados como enfermos y para la clase 2 desciende al 66.4 %, personas sanas a las que se les recomendaría más pruebas. Creemos que no hemos alcanzado un mayor porcentaje de instancias bien clasificadas debido a la gran cantidad de atributos ausentes que teníamos inicialmente; para la edad, clase social y tamaño del municipio de residencia eran de un 80%, estudios 77%, trabajo 73%, sexo y clase 0%.

Desde el inicio del trabajo hemos pretendido enfrentarnos a nuevos retos como han sido utilizar distintas técnicas y algoritmos para manejar la ausencia de datos y tratar el desbalanceo de los mismos, recordemos que hemos imputado valores con distintas técnicas y aplicado muestreo para reducir ejemplos, ya que con el algoritmo SMOTE, usado para balancear las clases, se llegaban casi a los cuatro millones y medio de filas y el equipo con el que estábamos trabajando no era capaz de operar con ese número de ejemplos.

Otro de los objetivos era el tratar de adquirir conocimiento a partir de datos que a priori parecen poco o nada relevantes, como son los atributos de; sexo, clase social, edad, tamaño del municipio de residencia o nivel de estudios, para tener una conclusión que si es relevante, que es si una persona está o no enferma.

Hemos desarrollado el software necesario para, desde una página HTML, obtener los datos globales de una enfermedad y otro software para preparar el dataset que necesitan los programas que hemos utilizado para hacer la clasificación. Para esta parte primero hemos tenido que estudiar en que forma estaba estructurada la información para detectar problemas en los datos (como así ha sido, ya que al hacer las sumas alguna no se correspondían con los totales mostrados). Sin esta parte no hubiésemos podido realizar la clasificación.

Weka lo hemos usado en varias asignaturas durante los estudios pero no de una manera tan profunda y detallada como ahora, el ensamblado de algoritmos ha sido uno de los nuevos aprendizajes. Dentro del amplio abanico que existe en las herramientas destinadas a la minería de datos elegimos RapidMiner (era una herramienta totalmente desconocida) muy intuitiva en el manejo y con numerosos tutoriales en la web.

Antes de iniciar las clasificaciones con la MLP hemos profundizado en su conocimiento y mostrado un resumen al lector, para después ejecutar distintas rondas con distintas configuraciones, con esto hemos comprobado que nuestra red aprendía.

Posteriormente hemos ejecutados diversos algoritmos de aprendizaje automático para intentar mejorar los resultados obtenidos con el MLP sin que haya sido así.

Como ya hemos visto a lo largo del trabajo existen numerosos estudios que utilizan datos médicos para predecir diversas enfermedades con un alto grado de precisión, si nuestro objetivo hubiese sido este las bases de datos las podríamos encontrar en distintos sitios, por ejemplo en (https://www.data.gov/, s.f.) que tiene a disposición catorce bases de datos relacionadas con la hipertensión, estas relaciones tienen un número de ejemplos que va de los veintisiete de la más pequeña a algo más de ochenta y cinco mil la mayor, ninguna tiene datos ausentes, pero nos limitaríamos a hacer un ejercicio de clasificación y no de adquisición de conocimiento que es uno de los propósitos de este trabajo. Dada las muy buenas clasificaciones que se encuentran en los estudios ya mencionados no contemplamos la posibilidad de realizar este trabajo nuevamente con datos médicos.

Trabajos citados

1. *Eurostat*. (s.f.). Obtenido de http://ec.europa.eu/eurostat/search?p_auth=hZUjSz3n&p_p_id=estatsearchportlet_WAR_estatsearchportlet&p_p_lifecycle=1&p_p_state=maximized&p_p_mode=view&_estatsearchportlet_WAR_estatsearchportlet_action=search&text=hypertension

2. *World Bank Open Data*. (s.f.). Obtenido de https://datacatalog.worldbank.org/search?search_api_views_fulltext_op=AND&query=hypertension&nid=&sort_by=search_api_relevance&sort_by=search_api_relevance

3. Aho A.V., L. M. (2008). *Compiladores principios, técnicas y herramientas, 2da Ed.* México: Pearson Educación.

4. Allende, F. M.-T. (2005). *Java 2: iniciación y referencia (2a. ed.).*

5. Alonso Jiménez J.A., G. N. (2000). *cs.us.es.* Obtenido de Razonamiento Automático: https://www.cs.us.es/~jalonso/cursos/ra-00/temas/tema-12.pdf

6. Aluja T. (2001). La minería de datos, entre la estadística y la inteligencia artificial. *Questiio*, 479-498. Obtenido de https://www.raco.cat/index.php/Questiio/article/viewFile/27009/26843

7. Álvarez-Aliaga, A. R.-B.-V. (2006). Factores de riesgo de la enfermedad cerebrovascular aguda hipertensiva. *Revista Cubana de Medicina*, vol.45, n.4. Obtenido de http://scielo.sld.cu/scielo.php?script=sci_isoref&pid=S0034-75232006000400006&lng=es&tlng=es

8. Antonelli D., B. E. (2013). Analysis of diabetic patients through their examination history. doi:https://doi.org/10.1016/j.eswa.2013.02.006

9. Aristizábal M. (2006). Evaluación asimétrica de una red neuronal artificial: aplicación al caso de la inflación en Colombia. *Lecturas de Economía, (65)*, 73-116. Obtenido de http://www.redalyc.org/comocitar.oa?id=155213357003

10. Arona R., S. (2012). Comparative Analysis of Classification Algorithms on Diferent Datasets using Weka. *International Journal of Computer Applications*, 21-25. Obtenido de

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.258.9202&rep=rep1&type=pdf

11. Avellaneda González J.A., O. R. (2010). Implementación redes neuronales para la detección de enfermedades del corazón. *Redes de ingeniería*, 38-46. Obtenido de https://revistas.udistrital.edu.co/ojs/index.php/REDES/article/viewFile/7159/8816

12. Barandela R., V. R. (2004). Imbalanced Training Sample Problem: under or over Sampling. 806-814. Obtenido de https://pdfs.semanticscholar.org/7c29/7b36f8ad53a1e387613336886a466e3f0d01.pdf

13. Basogain Olabe X. (s.f.). *Redes neuronales artificiales y sus aplicaciones.* Escuela superior de ingenieria de Bilbao. Obtenido de https://ocw.ehu.eus/file.php/102/redes_neuro/contenidos/pdf/libro-del-curso.pdf

14. Berardi H., C. A. (2015). Examen Doppler de la insuficiencia venosa de miembros inferiores: consenso entre especialistas. *Revista Argentina de Radiología, vol. 79, núm. 2*, 72-79. Obtenido de http://www.redalyc.org/pdf/3825/382539300003.pdf

15. Blum, A. P. (1997). Selection of relevant features and examples in machine learnning. *Artificial Intelligence*, 245-271. Obtenido de https://core.ac.uk/download/pdf/82009906.pdf

16. BOE. (2015). Real Decreto Legislativo 2/2015, de 23 de octubre, por el que se aprueba el texto refundido de la Ley del Estatuto de los Trabajadores. España. Obtenido de https://www.boe.es/buscar/act.php?id=BOE-A-2015-11430&p=20170513&tn=1#a6

17. BOE. (s.f.). Real Decreto 1105/2014, de 26 de diciembre, por el que se establece el. Obtenido de https://www.boe.es/boe/dias/2015/01/03/pdfs/BOE-A-2015-37.pdf

18. Bradley D.A., S. D. (2000). *Mechatronics and the Design of Intelligent Machines and Systems.* CRC Press. Obtenido de https://books.google.es/books?id=jHg9SfB68hgC&pg=PA150&dq=Perceptr%C3%B3n+Adeline&hl=es&sa=X&ved=0ahUKEwjV6Ze4t4DaAhWG8RQKHXHhCqEQ6AEIKjAA#v=onepage&q=Perceptr%C3%B3n%20Adeline&f=false

19. Breiman L. (1996). Bagging Predictors. *Machine Learning*, 123-140. doi:https://doi.org/10.1023/A:1018054314350

20. Breiman L. (1998). Arcing classifiers. *The Annals of statistics*, 801-849. Obtenido de https://projecteuclid.org/download/pdf_1/euclid.aos/1024691079

21. Breiman L. (2001). Random Forests. *Machine Learning*, 5-32. doi:https://doi.org/10.1023/A:1010933404324

22. Buendía Rodríguez E., V. P. (2002). Aplicación de redes neuronales artificiales y técnicas sig para la predicción de coberturas forestales. *Chapingo. Serie Ciencias Forestales y del Ambiente*, 31-37. Obtenido de http://www.redalyc.org/html/629/62980104/

23. C++, D. (s.f.). *bloodshed*. Obtenido de http://www.bloodshed.net/index.html

24. Caballero Ruiz E., G.-S. G. (2012). *Clasificación de medidas de glucemia en función de ingestas en diabetes gestacional.* Obtenido de http://oa.upm.es/39109/9/tmp_11891-Diabetes1052324520.pdf

25. Canizares M., B. I. (2004). Datos incompletos: una mirada crítica para su manejo en estudios sanitarios. *Gac Sanit [online*, 58-63. Obtenido de http://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S0213-91112004000100010

26. Castillo Sanz A., S. G. (2005). *C algoritmos, programación y estructuras de datos.* Madrid: McGraw-Hill España. Obtenido de https://intprog.files.wordpress.com/2015/08/programacion-en-c-metodologia-algoritmos-y-estructura-de-datos-editorial-mcgraw-hill.pdf

27. *CDC*. (s.f.). Obtenido de CDC: https://www.cdc.gov/bloodpressure/maps_data.htm

28. Cerda J., C. L. (2012). Uso de curvas ROC en investigación clínica. Aspectos teórico-prácticos. *Revista chilena de infectología*, 138-141. doi:http://dx.doi.org/10.4067/S0716-10182012000200003

29. Chávez M. C., C. G. (2009). Uso De Redes Bayesianas Obtenidas Mediante Optimización de enjambre de partículas para diagnóstico de hipertensión arterial. *INVESTIGACIÓN OPERACIONAL*, 52-60. Obtenido de https://biblat.unam.mx/es/revista/investigacion-operacional/articulo/uso-de-redes-bayesianas-obtenidas-mediante-optimizacion-del-enjambre-de-particulas-para-el-diagnostico-de-la-hipertension-arterial

30. Chawla N.V., B. K. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 321-357. doi:https://doi.org/10.1613/jair.953

31. Cinca C., D. B. (1993). Prediccion de la quiebra bancaria mediante el empleo de redes artificiales. *Revista Española De Financiación Y Contabilidad, 22(74*, 153-176. Obtenido de http://www.jstor.org/stable/42781034

32. Clínica Universidad de Navarra. (s.f.). Obtenido de https://www.cun.es/diccionario-medico/terminos/normotenso

33. Cohen G., H. M. (2006). Learning from imbalanced data in surveillance of nosocomial infection. *Artificial Intelligence in Medicine, 37*, 7-18. doi:https://doi.org/10.1016/j.artmed.2005.03.002

34. Cohen W. W. (1995). Fast Effective Rule Induction. (S. R. Armand Prieditis, Ed.) *Machine Learning Proceedings*, 115-123. doi:https://doi.org/10.1016/B978-1-55860-377-6.50023-2

35. Cuadrado Rodríguez S., G. R. (2011). Sistema experto basado en casos para el diagnóstico de la hipertensión arterial. *Revista Facultad de Ingeniería Universidad de Antioquia*, 202-213. Obtenido de http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0120-62302011000400020&lng=en&tlng=es.

36. *data Catalog*. (s.f.). Obtenido de https://datacatalog.worldbank.org/?search_api_views_fulltext_op=AND&query=HYPERTENSION&sort_by=changed

37. de Luna-Ortega C. A., M.-G. M.-R.-R.-M. (2014). Reconocimiento del habla mediante el uso de la correlación cruzada y una perceptrón multicapa. *Nova Scientia.*, 108-124. Obtenido de https://doaj.org/article/2f4803d08fa24eed8dab4b56cea0e7c4

38. De'ath G. (2007). Boosted Trees for ecological modeling and prediction. *Ecology*, 243-251. doi:https://doi.org/10.1890/0012-9658(2007)88[243:BTFEMA]2.0.CO;2

39. del Río A., F. M. (2002). Las pruebas de esfuerzo. *elsevier*, 41-50. Obtenido de http://www.elsevier.es/es-revista-clinica-e-investigacion-arteriosclerosis-15-articulo-las-pruebas-esfuerzo-S0214916802788232

40. Draft, C. (2007). *ISO/IEC 9899:TC3.* Obtenido de http://www.open-std.org/JTC1/SC22/WG14/www/docs/n1256.pdf

41. Esteban Fernández A., S. G. (2014). Aproximación diagnóstica a la cardiopatía hipertensiva. *Cardiocore*, 28-30. Obtenido de http://www.redalyc.org/pdf/2770/277031274009.pdf

42. Farhangfar A., z. L. (2008). Impact of imputation of missing values on classification error for discrete data. *Science Direct*, 3692-3705. doi:https://doi.org/10.1016/j.patcog.2008.05.019

43. Fernández-Alonso, R. S.-Á. (2012). Imputación de datos perdidos en las evaluaciones diagnósticas educativas. *Psicothema*, 166-175. Obtenido de http://www.redalyc.org/html/727/72723431026/

44. *File system*. (s.f.). Obtenido de https://nodejs.org/api/fs.html

45. Flores Fernández J.M. (2011). *Diseño de una herramienta auxiliar en el diagnóstico de cáncer de pulmón mediante la cuantificación de proteínas biomarcadoras utilizando redes neuronales artificiales.* Obtenido de https://ciatej.repositorioinstitucional.mx/jspui/handle/1023/369

46. García Gutiérrez J.A. (2016). *Comenzando con Weka: Filtrado y selección de subconjuntos de atributos basada en su relevancia descriptiva para la clase.* Obtenido de https://www.researchgate.net/publication/308141950_Comenzando_con_Weka_Filtrado_y_seleccion_de_subconjuntos_de_atributos_basada_en_su_relevancia_descriptiva_para_la_clase?enrichId=rgreq-f64cf84f17069f83b211d91d86724231-XXX&enrichSource=Y292ZXJQYWdlOzMwODE

47. García Jiménez V. (2010). *Distribuciones de Clases No Balanceadas: Métricas, Análisis de Complejidad y Algoritmos de Aprendizaje.* Universitat Jaume I. Departament de Llenguatges i Sistemes Informàtics . Obtenido de https://www.tdx.cat/handle/10803/10491

48. Gironés J., C. J. (2017). *Minería de datos: modelos y algoritmos.* Barcelona: UOC.

49. *GitHub*. (s.f.). Obtenido de https://github.com/github

50. gnu.org. (s.f.). *gnu.org*. Obtenido de https://www.gnu.org/licenses/licenses.es.html

51. González Padilla A., B. Q. (2013). Clasificación del clutter marino utilizando redes neuronales artificiales. *Ingeniería Electrónica, Automática y Comunicaciones*, 1-11. Obtenido de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59282013000100001&lng=es&tlng=en

52. Han H., H. W. (2005). Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. *ICIC*, 878-887. Obtenido de http://sci2s.ugr.es/keel/keel-dataset/pdfs/2005-Han-LNCS.pdf

53. Hatem H., M. S. (2010). Performance Evaluation of RESTful Web Services for Mobile Devices. *International Arab Journal of e-Technology*, 72-78. Obtenido de http://www.gregbulla.com/TechStuff/Docs/ws-restful-pdf.pdf

54. Herrera F., C. J. (2006). Técnicas de reducción de datos en KDD. El uso de Algoritmos Evolutivos para la Selección de Instancias. *Actas del I Seminario Sobre Sistemas Inteligentes*, 165-181. Obtenido de http://sci2s.ugr.es/sites/default/files/ficherosPublicaciones/0678_Herrera-Cano-ssi06.pdf

55. hipertensión. (s.f.). *Clínica Universidad de Navarra*. Obtenido de https://www.cun.es/material-audiovisual/infografia/hipertension-arterial

56. Ho T. K. (1998). The Random Subspace Method for Constructing Decision Forests. 832-844. Obtenido de http://citeseer.ist.psu.edu/ho98random.html

57. Hopcroft J.E., M. R. (s.f.). *Teoría de autómatas, lenguajes y computación.* Pearson.

58. *https://www.data.gov/*. (s.f.). Obtenido de https://www.data.gov/: https://catalog.data.gov/dataset?q=hypertension&sort=score+desc%2C+name+asc&as_sfid=AAAAAAX8T4VViPLLEwdK-jZhb35C8gZp3gGY7KounfJ3Bvj4vYARS10MMSuwXnBv3Gh3kug8sndVi_GsNO_nUwFHGgeoMzE0AZEPVY61-ruQzHdXbg3ge9yFI8Wr91p0Mz701_k%3D&as_fid=640bc06d0eb1a5266ad2ddaeb

59. *INE*. (s.f.). Obtenido de http://www.ine.es/buscar/searchResults.do?searchString=hipertension&Menu_botonBuscador=Buscar&searchType=DEF_SEARCH&startat=0&L=0

60. Ingeniería, R. A. (s.f.). *Diccionario Español de Ingeniería*. Recuperado el 2018, de http://diccionario.raing.es

61. Jonh G. H., L. P. (1995). Estimating Continuous Distributions in Bayesian Classifiers. *Eleventh Conference on Uncertainty in Artificial Intelligence*, 338-345. Obtenido de http://web.cs.iastate.edu/~honavar/bayes-continuous.pdf

62. Joyanes Aguilar L., Z. M. (2005). *Programación en C: metodología, algortimos y estructura de datos (2a. ed.).* Madrid: McGraw-Hill España.

63. *Jsonlint.com*. (s.f.). Obtenido de https://jsonlint.com/

64. Jungermann F. (2009). *Information Extraction with RapidMiner.* Duisburg . Obtenido de http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.184.1826&rep=rep1&type=pdf#page=59

65. Kala P., S. R. (2012). Desing of rectangular partch antenna using MLP neural network. *Journal of Global Research in Computer Science*, 11-14. Obtenido de http://www.rroij.com/open-access/design-of-rectangular-patch-antenna-using-mlp-artificial-neural-network-11-14.pdf

66. Kass V. (1980). An Exploratory Technique for Investigating Large Quantities of Categorical Data. *Appl. Statist*, 119-127. Obtenido de https://www.jstor.org/stable/2986296?seq=1#page_scan_tab_contents

67. Kittler J., H. M. (1998). On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 226-239. Obtenido de https://dspace.cvut.cz/bitstream/handle/10467/9443/1998-On-combining-classifiers.pdf?sequence=1

68. Kuncheva L.I. (2004). *Combining Pattern Classifiers: Methods and Algorithms.* doi:DOI:10.1002/0471660264

69. Le Gall J.F. (1992). The uniform random tree in a Brownian excursion. *Probability Theory and Related Fields*, 369–383. Obtenido de https://link.springer.com/article/10.1007/BF01292678#citeas

70. Liaw A., W. M. (2001). Classification and regression by randomForest. 18-22. Obtenido de

https://www.researchgate.net/profile/Andy_Liaw/publication/228451484_Classification_and_Regression_by_RandomForest/links/53fb24cc0cf20a45497047ab/Classification-and-Regression-by-RandomForest.pdf

71. López Puga, J. G. (2007). Las redes bayesianas como herramientas de modelado en psicología. *Anales de Psicología*, 1695-2294. Obtenido de http://www.redalyc.org/html/167/16723218/

72. Mamami Vargas G. F. (2017). *Sistema Experto para diagnóstico de colesterol.* Obtenido de http://repositorio.umsa.bo/xmlui/handle/123456789/12331

73. Marín R., G. M. (2004). Nefropatia vascular. Concepto y significado. Epidemiologia. Hipertension arterial esencial e insuficiencia renal. *revistanefrologia*, 73-83. Obtenido de http://www.revistanefrologia.com/es-publicacion-nefrologia-articulo-nefropatia-vascular-concepto-significado-epidemiologia-hipertension-arterial-esencial-e-insuficiencia-X0211699504030623

74. Martín Valdivia M.T., G. V. (2002). Resolución de la ambigüedad mediante redes neuronales. *Procesamiento del lenguaje natural, nº29*, 39-45. Obtenido de http://www.sepln.org/revistaSEPLN/revista/29/29-Pag39.pdf

75. Mederos Brú M.V., L. F. (2004). Una comparación de dos metodos de gradiente en el escalamiento multidimensional. *Ciencias Matemáticas*, 44. Obtenido de http://go.galegroup.com/ps/anonymous?id=GALE%7CA146221886&sid=googleScholar&v=2.1&it=r&linkaccess=abs&issn=02565374&p=AONE&sw=w

76. Medina-Merino, R. Ñ.-C. (2017). Bosques aleatorios como extensión de los árboles de clasificación con los programas R y Python. *Interfases*, 165-189. doi:http://dx.doi.org/10.26439/interfases2017.n10.1775

77. microsoft. (s.f.). *Manifiesto del ensamblado*. Obtenido de https://docs.microsoft.com/es-es/dotnet/framework/app-domains/assembly-manifest

78. Mitchell T. (1997). *Machine Learning.* McGraw-Hill Science. Obtenido de https://www.cs.ubbcluj.ro/~gabis/ml/ml-books/McGrawHill%20-%20Machine%20Learning%20-Tom%20Mitchell.pdf

79. Montero Lorenzo J.M. (2007). *Estadística descriptiva.* Madrid: Paraninfo. Obtenido de https://books.google.es/books?hl=es&lr=&id=D6sj2d0xTgUC&oi=fnd&pg=PR4&dq=m

edidas+de+posici%C3%B3n+central+moda&ots=4nLjQRsAzL&sig=fLTLNtvdk70Q_PUlQRma-U4U5Cc#v=onepage&q=medidas%20de%20posici%C3%B3n%20central%20moda&f=false

80. Moya-Rodríguez J.L., B.-F. A.-M. (2012). Utilización de Sistemas Basados en Reglas y en Casos para diseñar transmisiones por tornillo sinfín. *Ingeniería Mecánica vol.15 no.1* . Obtenido de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59442012000100001

81. Mozilla. (s.f.). *developer.mozilla.org*. Obtenido de https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/JavaScript_basics

82. NetBeans. (s.f.). *NetBeans*. Obtenido de https://netbeans.org/features/ide/index.html

83. Nieves Hurtado A., D. S. (2010). *Probabilidad y estadística para ingeniería un enfoque moderno.* México: McGraw-Hill.

84. *Nodejs*. (s.f.). Obtenido de https://nodejs.org/es/

85. *npm*. (s.f.). Obtenido de https://www.npmjs.com/

86. Ñustes S.A., H. J. (2013). Introducción al desarrollo de redes neuronales perceptrón multicapa aplicadas en tecnología Android. *Amazonia Investiga*. Obtenido de http://www.udla.edu.co/revistas/index.php/amazonia-investiga/article/view/17

87. *OECD iLibrary*. (s.f.). Obtenido de https://www.oecd-ilibrary.org/search?value1=hypertension&option1=quicksearch&facetOptions=51&facetNames=pub_igoId_facet&operator51=AND&option51=pub_igoId_facet&value51=%27igo%2Foecd%27

88. *OMS*. (s.f.). Obtenido de Organización mundial de la salud : http://www.who.int/topics/hypertension/es/

89. OMS. (2013). *Información general sobre la HIPERTENSIÓN en el mundo.* Suiza. Obtenido de http://apps.who.int/iris/bitstream/handle/10665/87679/WHO_DCO_WHD_2013.2_spa.pdf;jsessionid=29D9428985B09BCA34000AEEA02442B4?sequence=1

90. P.J., G. L., & J.L., S. G. (2008). Regresión Local por Mínimos Cuadrados para Estimación Eficiente de Datos Incompletos. *Jornadas de Introducción a la Investigación de la UPCT*, (págs. 16-18). Cartagena. Obtenido de http://repositorio.upct.es/bitstream/handle/10317/2581/1.4.pdf?sequence=1&isAllowed=y

91. Palma M. J. T., M. M. (2008). *Inteligencia artificial: métodos, técnicas y aplicaciones.* Obtenido de https://bv.unir.net:2056

92. Palmer P.A., M. M. (2000). Predicción del consumo de éxtasis a partir de redes neuronales artificiales. *IREFREA*, 29-41. Obtenido de http://www.irefrea.eu/uploads/PDF/Palmer%20et%20al_2000_Prediccion%20consumo%20extasis.pdf

93. Pelaez J.I, V. G. (2016). Un Sistemas para Detección de Contaminación por Hidrocarburos: Aplicación al Oriente Ecuatoriano. *Memorias de la Décima Quinta Conferencia Iberoamericana en Sistemas, Cibernética e Informática*, (págs. 180-185). Obtenido de http://www.iiis.org/CDs2016/CD2016Summer/papers/CA048UI.pdf

94. Pérez Aguila, R. (2012). *Una introducción al cómputo neuronal artificial.* El Cid Editor. Obtenido de https://es.scribd.com/document/361598049/Una-Introduccion-Al-Computo-Neuronal-Artificial-Ricardo-Perez-Aguila

95. Pérez Díaz A. (2012). Aplicación de la red de probabilidad neuronal y escala de framingham para la predicción de hipertensión arterial. *Memorias Convención Internacional de Salud Pública.* Cuba. Obtenido de http://www.convencionsalud2012.sld.cu/index.php/convencionsalud/2012/paper/view/303

96. Pérez Lopez C., S. G. (s.f.). *Minería de datos. Técnicas y herramientas.* Paraninfo.

97. Peterson L. E. (2009). *scholarpedia*. doi:10.4249/scholarpedia.1883

98. Petticrew M.P., S. A.-S. (2000). False-negative results in screening programmes: systematic review of impact and implications. *pubmed*. Obtenido de https://www.ncbi.nlm.nih.gov/pubmed/10859208

99. Pita Fernández S., P. D. (2010). Pruebas diagnósticas: Sensibilidad y especificidad. *Cad Aten Primaria 2003*, 120-124. Obtenido de www.hsfq.gob.ec/multimedia/Pruebas_diag.docx

100. Powers D. (2007). *Evaluation: From Precision, Recall and F-Factor.* School of Informatics and Engineering , Adelaide • Australia. Obtenido de http://www.flinders.edu.au/science_engineering/fms/School-CSEM/publications/tech_reps-research_artfcts/TRRA_2007.pdf

101. Quinlan J. (1993). Programs form machine learning. *Machine Learning*, 235-240. Obtenido de https://link.springer.com/content/pdf/10.1007%2FBF00993309.pdf

102. Quinlan J. R. (1996). Improved Use of Continuous Attributes in C4.5. *Journal of Artificial Intelligence Research*, 77-90. doi:doi:10.1613/jair.279

103. Quinlan, J. (1986). Induction of decision trees. *Mach Learn*, 81-106. doi:https://doi.org/10.1007/BF00116251

104. Quintero-Méndez M. A., D.-N. M. (2008). Aplicación de dos pruebas estadísticas de bondad de ajuste en muestras complejas: un caso práctico en el campo forestal. *Agrociencia*, 287-297. Obtenido de http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-31952008000300004&lng=es&tlng=es

105. Ramezankhani A., H. E. (2016). *Modelo de decisión basado en árbol para la identificación de posibles interacciones entre factores de riesgo de diabetes tipo 2*. Obtenido de http://bmjopen.bmj.com/content/bmjopen/6/12/e013336.full.pdf

106. *RapidMiner*. (s.f.). Obtenido de https://rapidminer.com/

107. Rényi A. (1961). On measures of entropy and information. *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability.* California. Obtenido de www.dtic.mil/get-tr-doc/pdf?AD=AD1027761

108. *Request*. (s.f.). Obtenido de https://www.npmjs.com/package/request

109. Richard Landis J., K. G. (1977). The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 159-174. Obtenido de https://www.jstor.org/stable/2529310?origin=JSTOR-pdf&seq=1#page_scan_tab_contents

110. Rodríguez Tapia S., C. C. (2018). AUTOMÁTICO SUPERVISADO EN LA CLASIFICACIÓN TEXTUAL SEGÚN EL GRADO DE ESPECIALIZACIÓN, LOS MÉTODOS DE APRENDIZAJE. *Revista de estudios Filológicos*. Obtenido de http://www.tonosdigital.es/ojs/index.php/tonos/article/view/2009/1018

111. Rojas R. (1996). *Neural Networks.* Berlin. Obtenido de https://page.mi.fu-berlin.de/rojas/neural/chapter/K7.pdf

112. Ron Kohavi F.P. (1998). Glossary of Terms. *Machine Learning, 30*, 271-274. Obtenido de http://ai.stanford.edu/~ronnyk/glossary.html

113. Rumelhart D. E., H. G. (1986). Learning representations by back-propagating errors. *Nature*, 533–536. doi:10.1038/323533a0

114. Saez M., B. M. (2012). Coste de la hipertensión arterial en España. *Hipertensión y Riesgo Vascular*, 145-151. Obtenido de https://www.sciencedirect.com/science/article/pii/S1889183712000645

115. *Saludcastillayleon*. (s.f.). Obtenido de https://www.saludcastillayleon.es/sanidad/cm/gallery/ENCUESTA%20REGIONAL%20DE%20SALUD%202003/Marco.html

116. Sánchez E., A. A. (2011). Redes Neuronales Artificiales: Una revisión del estado del arte, aplicaciones y tendencias futuras. *Investigación y Desarrollo en TIC, Vol 2 N 1*, 18-27. Obtenido de http://publicaciones.unisimonbolivar.edu.co/rdigital/inovacioning/index.php/identic/article/viewFile/21/29

117. *Sanitize-html*. (s.f.). Obtenido de https://www.npmjs.com/package/sanitize-html

118. Schmidhuber J. (2014). Deep learning allows computational models that are composed of multiple processing. *Neural Networks*, 85-117. Obtenido de https://www.sciencedirect.com/science/article/pii/S0893608014002135

119. Shashi Sathyanarayana P. (2014). *A Gentle Introduction to Backpropagation.* Obtenido de http://numericinsight.com/uploads/A_Gentle_Introduction_to_Backpropagation.pdf

120. Shen R., H. P. (2003). Data Mining and Case-based Reasoning for Distance Learning. *International Journal of Distance Education*, 46-58. Obtenido de https://pdfs.semanticscholar.org/5360/235efce9001c0ec3a3307dee76f05387cfbd.pdf

121. Silva L.C., B. A. (2001). El enfoque bayesiano: otra manera de inferir. *Gaceta Sanitaria*, 341-346. doi:https://doi.org/10.1016/S0213-9111(01)71578-6

122. Solarte Martínez G., S. M. (2011). Arboles de decisiones en el diagnóstico de enfermedades cardiovasculares. *Scientia Et Technica*, 104-109. Obtenido de http://revistas.utp.edu.co/index.php/revistaciencia/article/view/1487

123. sourceforge.net. (s.f.). *sourceforge.net*. Obtenido de https://sourceforge.net/projects/launch4j/files/launch4j-3/3.12/

124. Spiegel M. R. (1991). *Estadística.* Madrid: McGraw-Hill Interamericana. Obtenido de https://clea.edu.mx/biblioteca/Spiegel%20Murray%20-%20Probabilidad%20Y%20Estadistica.pdf

125. T. Hagan M., B. D. (s.f.). *Neural Network Design 2nd Edition.* Obtenido de http://hagan.okstate.edu/NNDesign.pdf

126. Tavarez D., N. E. (2013). Nueva técnica de fusión de clasificadores aplicada a la mejora de la segmentación de audio. *Procesamiento del lenguaje natural*, 161-168. Obtenido de https://pdfs.semanticscholar.org/1d67/de0594dbdd3feb50ab52a970a7694932608c.pdf

127. Tibshirani R., S. G. (1999). Imputing Missing Data for Gene Expression Arrays Trevor Hastie. *Technical Report, Division of Biostatistics, Stanford University*. Obtenido de http://ww.web.stanford.edu/~hastie/Papers/missing.pdf

128. Torres Macho J., G. d. (2012). Ecocardiografía clínica básica en Medicina Interna. *Revista Clínica Española*, 141-146. Obtenido de http://www.revclinesp.es/es/ecocardiografia-clinica-basica-medicina-interna/articulo/S0014256511005169/

129. *Typora*. (s.f.). Obtenido de https://typora.io/

130. Unadkat S. B., C. M. (2001). *Recurrent neural network.* CRC Press. Obtenido de

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.375.5562&rep=rep1&type=pdf

131. Useche Castro L. M., M. Á. (2006). Una introducción a la Imputación de Valores Perdidos. *Terra Nueva Etapa 2006 XXII*, 127-152. Obtenido de http://www.redalyc.org/html/721/72103106/

132. Valcárcel Asencios V. (2004). Data Mining y el descubrimiento del conocimiento. *Revista de la Facultad de Ingeniería Industrial*. Obtenido de http://www.redalyc.org/html/816/81670213/

133. Vallejo C.G., T. J. (2004). *WIRS. Un Algoritmo de Reducción de Instancias Basado en Ranking.* Sevilla: Departamento de Lenguajes y Sistemas Informaticos Universidad de Sevilla. Obtenido de https://www.researchgate.net/profile/F_Javier_Ortega/publication/229150262_WIRS_un_algoritmo_de_reduccion_de_instancias_basado_en_ranking/links/0deec5320412896d0b000000/WIRS-un-algoritmo-de-reduccion-de-instancias-basado-en-ranking.pdf

134. Vanderlei Filho D. (2005). Sistemas de apoio à decisão no diagnóstico médico da hipertensão arterial e das arritmias cardíacas. *post-graduado UNIVERSIDADE FEDERAL DE PERNAMBUCO*. Obtenido de https://repositorio.ufpe.br/bitstream/handle/123456789/5087/arquivo7051_1.pdf?sequence=1&isAllowed=y

135. Villalba J.D, G. I. (2012). Detección de daño en vigas utilizando redes neuronales artificiales y parámetros dinámicos. *Facultad de Ingeniería Universidad de Antioquia.*, 141-153. Obtenido de https://doaj.org/article/92d3a3da19734b37910663501953c487

136. *VisualStudio*. (s.f.). Obtenido de https://code.visualstudio.com/insiders/

137. Voegler J., B. J. (2014). A Simple Syntax for Transcription of Accessible Study Materials. *International Conference on Computers for Handicapped Persons*, (págs. 545-548). Obtenido de https://link.springer.com/chapter/10.1007/978-3-319-08596-8_85#citeas

138. w3schools. (s.f.). *w3schools.com*. Obtenido de https://www.w3schools.com/html/html_intro.asp

139. Wahbeh A.H., A.-R. Q.-K. (2011). A comparison study between data mining tools over some classification methods. *International Journal of Advanced Computer Science and Applications*, 18-26. Obtenido de https://pdfs.semanticscholar.org/199e/2a48f36b56f011ba4542721dc47e1b9078aa.pdf

140. waikato. (s.f.). *waikato*. Obtenido de https://www.cs.waikato.ac.nz/ml/weka/arff.html

141. Wang J., X. M. (2006). Classification of Imbalanced Data by Using the SMOTE Algorithm and Locally Linear Embedding. *ICSP2006 Proceedings*, 16-20. Obtenido de http://sci2s.ugr.es/keel/pdf/specific/congreso/04129201.pdf

142. *weka*. (s.f.). Obtenido de https://www.cs.waikato.ac.nz/ml/weka/

143. Wilkinson L. (s.f.). *Tree Structured Data Analysis: AID, CHAID and CART.* Chicago. Obtenido de https://datamining.bus.utk.edu/Documents/Tree-Structured-Data-Analysis-(SPSS).pdf

144. Wolpert D.H. (1992). Stacked generalization. *Neural Networks.*, 241-259. doi:https://doi.org/10.1016/S0893-6080(05)80023-1

145. Yoav Freund R. (1996). Experiments with a new boosting algorithm. *Machine Learning*, 148-156. Obtenido de https://cseweb.ucsd.edu/~yfreund/papers/boostingexperiments.pdf

146. Yongheng Z., Y. Z. (2008). Comparison of decision tree methods for finding active objects. *Advances in Space Research*, 1955-1959. doi:https://doi.org/10.1016/j.asr.2007.07.020

147. Yzquierdo R., P. P. (2005). *Sistema para la ayuda a la toma de decisiones en el diagnóstico, evaluación, contron y tratamiento de los pacientes con hipertensión arterial (HYPERWEB) .* doi:10.13140/RG.2.1.4476.6163

Anexos

TFG		Fecha inicio	Duración Subtarea	Duración Tarea	Duración Total	Fecha Final
Plan de trabajo		**06/03/2018**			16	**27/03/2018**
	Propuesta-conceptualización	06/03/2018		8		15/03/2018
	Selección datos web	15/03/2018		2		16/06/2018
	Selección de metodología de extracción datos	19/03/2018		3		21/03/2018
	Selección algoritmo de clasificación	22/03/2018		3		26/03/2018
	Selección red neuronal	27/03/2018		1		27/03/2018
Arquitectura, prototipo tecnológico y estructura de memoria		**28/03/2018**			27	**03/05/2018**
	Descripción de alto nivel fase extracción de datos	28/03/2018		7		05/04/2018
	Descripción de alto nivel del algoritmo seleccionado	06/04/2018		6		13/04/2018
	Descripción de alto nivel de la red neuronal seleccionada	16/04/2018		6		23/04/2018
	Estructura de la memoria	24/04/2018		7		02/05/2018
	Índice	24/04/2018	1			24/04/2018
	Motivación y objetivo del TFG	25/04/2018	1			25/04/2018
	Metodología	26/04/2018	1			26/04/2018
	Código	27/04/2018	1			27/04/2018
	Resultados obtenidos	28/04/2018	1			30/04/2018

	Conclusiones y trabajos futuros	01/05/2018	1		01/05/2018
	Referencias bibliográficas	02/05/2018	1		02/05/2018
Prototipo y borrador de la memoria		**07/05/2018**		24	**07/06/2018**
	Código extracción de datos	07/05/2018		7	15/05/2018
	Código algoritmo	15/05/2018		7	23/05/2018
	Entrenamiento red	23/05/2018		3	25/05/2018
	Borrador memoria	28/05/2018		8	06/06/2018
Depósito de Trabajo Fin de Grado		**07/06/2018**			**19/07/2018**
	Corrección de errores	07/06/2018		11	21/06/2018
	Corrección de errores	03/07/2018		13	19/07/2018

Tabla 11

Variables y frecuencias

CLASE	DE 0 A 4 AÑOS	DE 5 A 15 AÑOS	DE 16 A 24 AÑOS	DE 25 A 34 AÑOS	DE 35 A 44 AÑOS	DE 45 A 54 AÑOS	DE 55 A 64 AÑOS	DE 65 A 74 AÑOS	DE 75 Y MÁS AÑOS	SIN ESTUDIOS	PRIMER CICLO
123.098	0 0,00	0 0,00	0 0,00	3.795 0,01	9.628 0,02	15.216 0,02	25.176 0,04	37.233 0,06	32.050 0,05	20.253 0,03	83.153 0,14
Hombres Si	**0**	**0**	**0**	**759**	**1925,6**	**3043,2**	**5035,2**	**7446,6**	**6410**	**4050,6**	**16630,6**
1.080.765	28.124 0,01	132.131 0,02	136.757 0,03	185.238 0,03	179.978 0,03	148.880 0,03	105.276 0,02	95.177 0,02	69.204 0,01	106.626 0,02	641.829 0,12
Hombres No	**5624,8**	**26426,2**	**27351,4**	**37047,6**	**35995,6**	**29776**	**21055,2**	**19035,4**	**13840,8**	**21325,2**	**128365,8**
189.660	0 0,00	0 0,00	0 0,00	4.426 0,00	4.541 0,00	11.456 0,01	34.897 0,04	66.643 0,07	67.697 0,07	36.294 0,04	128.611 0,14
Mujeres Si	**0**	**0**	**0**	**885,2**	**908,2**	**2291,2**	**6979,4**	**13328,6**	**13539,4**	**7258,8**	**25722,2**
1.038.985	30.382 0,01	121.337 0,02	129.200 0,02	174.544 0,03	178.411 0,03	141.174 0,03	96.103 0,02	91.882 0,02	75.952 0,01	94.955 0,02	569.921 0,11
Mujeres No	**6076,4**	**24267,4**	**25840**	**34908,8**	**35682,2**	**28234,8**	**19220,6**	**18376,4**	**15190,4**	**18991**	**113984,2**

Tabla 12

CLASE	SEGUNDO CICLO	UNIV,	CLASE I	CLASE II	CLASE III	CLASE Iva	Ivb	CLASE V	CLASE VI	NO CONSTA
123.098	10.389	9.304	5.163	5.878	44.172	41.175	12.067	13.257	1.090	295
	0,02	0,02	0,01	0,01	0,07	0,07	0,02	0,02	0,00	0,00
Hombres Si	**2077,8**	**1860,8**	**1032,6**	**1175,6**	**8834,4**	**8235**	**2413,4**	**2651,4**	**218**	**59**
1.080.765	193.270	139.039	76.753	100.377	342.936	295.624	127.500	125.758	9.383	2.435
	0,04	0,03	0,01	0,02	0,06	0,05	0,02	0,02	0,00	0,00
Hombres No	**38654**	**27807,8**	**15350,6**	**20075,4**	**68587,2**	**59124,8**	**25500**	**25151,6**	**1876,6**	**487**
189.660	13.872	10.883	8.929	5.806	62.827	56.740	26.432	26.098	1.948	881
	0,01	0,01	0,01	0,01	0,07	0,06	0,03	0,03	0,00	0,00
Mujeres Si	**2774,4**	**2176,6**	**1785,8**	**1161,2**	**12565,4**	**11348**	**5286,4**	**5219,6**	**389,6**	**176,2**
1.038.985	219.421	154.688	79.051	98.828	330.946	288.430	115.568	110.064	12.026	4.072
	0,04	0,03	0,02	0,02	0,06	0,06	0,02	0,02	0,00	0,00
Mujeres No	**43884,2**	**30937,6**	**15810,2**	**19765,6**	**66189,2**	**57686**	**23113,6**	**22012,8**	**2405,2**	**814,4**

Tabla 13

CLASE	OCUPADO	PARADO	INACTIVO	MENOR O IGUAL A 2.000 HABITANTES	2.001 A 50.000 HABITANTES	MÁS DE 50.000 HABITANTES	SUMA
123.098	35.246	3.248	84.604	42.775	29.408	50.915	
	0,06	0,01	0,14	0,07	0,05	0,08	
Hombres Si	**7049,2**	**649,6**	**16920,8**	**8555**	**5881,6**	**10183**	**123098**
1.080.765	717.867	27.701	335.197	328.952	289.967	461.847	
	0,13	0,01	0,06	0,06	0,05	0,09	
Hombres No	**143573,4**	**5540,2**	**67039,4**	**65790,4**	**57993,4**	**92369,4**	**1080765**
189.660	40.896	6.693	142.071	66.986	50.830	71.844	
	0,04	0,01	0,15	0,07	0,05	0,08	
Mujeres Si	**8179,2**	**1338,6**	**28414,2**	**13397,2**	**10166**	**14368,8**	**189660**
1.038.985	696.156	22.744	320.086	262.725	283.927	492.333	
	0,13	0,00	0,06	0,05	0,05	0,09	
Mujeres No	**139231,2**	**4548,8**	**64017,2**	**52545**	**56785,4**	**98466,6**	**1038985**

Tabla 14

Diccionario de acrónimos

I.A. Inteligencia Artificial

TFG. Trabajo fin de grado.

RNAs. Redes neuronales artificiales

HTA. Hipertensión arterial.

MLP Multilayer Perceptron, perceptrón multicapa.

ARP. Adaptive resonance theory.

OMS. Organización Mundial de la Salud

SMOTE. Synthetic Minority Oversampling Tecnique.

SVM. Support Vector Machines

RIPPER Repeated Incremental Pruning Produce Error Reduction

CHAID Chi-Squared Automatic Interaction Detector

MCC coeficiente de correlación de Matthews

ROC Receiver Operating Characteristic,

SMOTE Synthetic Minority Over-sampling Technique

npm Node Package Manager