UNIVERSIDAD POLITÉCNICA DE PUEBLA

Programa Académico de Ingeniería en Informática

"BALANCEO DE DATOS PARA LA CLASIFICACIÓN DE IMÁGENES DE GALAXIAS"

Reporte Técnico NUMERO

Gladis Huerta Rueda

Juan C. Bonilla, Puebla. Abril 2010

UNIVERSIDAD POLITÉCNICA DE PUEBLA

Programa Académico de Ingeniería en Informática

"BALANCEO DE DATOS PARA LA CLASIFICACIÓN DE IMÁGENES DE GALAXIAS"

Gladis Huerta Rueda

Reporte Técnico PII-32-04-10

Comité Evaluador

Dr. Jorge de la Calleja Mora
Asesor

Dr. Antonio Benitez Ruiz
Sinodal

Dra. María Auxilio Medina Nieto
Sinodal

Juan C. Bonilla, Puebla. Abril 2010

Índice

Resumen

El siguiente proyecto tiene como objetivo el comparar el desempeño de tres algoritmos de balanceo de datos para clasificar imágenes de galaxias usando la herramienta WEKA. Los algoritmos usados para balancear de datos fueron SMOTE, Resample, SpreadSubSample y los algoritmos de clasificación de datos fueron BayesNet, NaiveBayes, NaiveBayes Simple, SMO, LWL, J48, J48graft y Random Forest con un conjunto de datos de 293 y 310 imágenes de galaxias, para este conjunto de datos se consideran tres clases con 18 galaxias elípticas, 281 galaxias espirales y 11 galaxias irregulares, para cinco clases con 18 galaxias elípticas, 19 galaxias espirales tipo So, 104 galaxias espirales tipo Sa+Sb, 141 galaxias espirales tipo Sc+Sd y 11 galaxias irregulares y para siete clases con 18 galaxias elípticas, 19 galaxias espirales tipo So, 26 galaxias espirales tipo Sa, 78 galaxias espirales tipo Sb, 133 galaxias espirales tipo Sc, 8 galaxias espirales tipo Sd y 11 irregulares. Cada algoritmo de clasificación se ejecutó diez veces con semillas aleatorias diferentes e igualmente se hizo para cada uno de los algoritmos de balanceo de datos. Los resultados obtenidos fueron muy favorables aplicando el algoritmo Resample al 500 % con el algoritmo RandomForest.

Capítulo 1

Planteamiento del problema de investigación

1.1. Introducción

La inteligencia artificial es un área de las ciencias computacionales cuyo propósito es imitar las capacidades de inteligencia del ser humano utilizando, entre otros dispositivos, a la computadora. El aprendizaje automático es una disciplina que forma parte de la inteligencia artificial, cuyo objetivo es construir programas que aprendan por medio de la experiencia para realizar alguna actividad .

De manera similar a como el ser humano aprende, a una computadora se le deben proporcionar ejemplos de los cuales pueda inferir la respuesta correcta a una situación determinada. Idealmente, el conjunto de datos que contiene los ejemplos para aprender debería estar balanceado, esto es, la cantidad de instancias para las diferentes situaciones que se están aprendiendo serían iguales. Sin embargo, es muy frecuente que problemas del mundo real presenten la problemática de datos no balanceados.

El problema de datos no balanceados se refiere a tener conjuntos de datos de alguna

clase, que exceden significativamente a otros de distintas clases. Es muy frecuente que los algoritmos de aprendizaje automático obtengan clasificaciones poco precisas sobre alguna clase de conjuntos de datos minoritarios. Por ejemplo, si se tienen dos tipos de datos: 99 % y 1 %, respectivamente; el algoritmo de clasificación podría clasificar a todos los datos de acuerdo a la clase mayoritaria, se podría estimar que será 99 % preciso. Sin embargo, no estaría clasificando correctamente ninguno de los datos de la clase minoritaria. Es decir, con datos no balanceados, la hipótesis obtenida es generalmente aquella que clasifica a casi todos los datos como pertenecientes a la clase mayoritaria. Por lo tanto, se tendrán clasificadores que obtengan alta exactitud para los datos de la clase mayoritaria y muy baja para los datos de la clase minoritaria, que es este trabajo la de mayor interés.

Algunas aplicaciones en donde se tienen datos no balanceados son clasificación de texto [30] ,[5], detección de cáncer [18], detección de llamadas telefónicas fraudulentas [26], entre otras. Por ejemplo, considere la clasificación de los pixeles en las imágenes de mamografías como posiblemente cancerosos [14]. Un conjunto de datos de mamografías pueden contener 98 % de pixeles normales y el 2 % de pixeles anormales. Una estrategia simple predeterminada de suponer de la clase mayoritaria daría un valor predictivo del 98 %. Así, se desea tener algoritmos de clasificación que obtengan predicciones exactas para los datos minoritarios, aunque se tengan índices de error ligeramente altos para los casos mayoritarios.

En Astronomía, el problema de los conjuntos de datos no balanceados es muy frecuente y este trabajo de investigación se abordará en este dominio.

Básicamente este proyecto de investigación se enfocará en la comparación del de-

sempeño de tres algoritmos de balanceo de datos para clasificar imágenes de galaxias. Para ello se emplearán algoritmos existentes de la herramienta WEKA.

1.2. Objetivo general

Comparar el desempeño de tres algoritmos de balanceo de datos para clasificar morfológicamente imágenes de galaxias.

1.3. Objetivos específicos

- Mejorar la clasificación de galaxias de tipo irregular.
- Identificar al algoritmo con mejor desempeño de balanceo para clasificar imágenes de galaxias.
- Probar 9 algoritmos de aprendizaje automático para clasificar.

1.4. Justificación

La Astronomía es una ciencia inmensamente rica en información, que presenta de manera inherente el problema de datos no balanceados, particularmente en la clasificación de objetos astronómicos. Esto se debe a que el primer paso en la formulación de nuevas teorías que expliquen el origen y evolución del Universo, es la clasificación de los objetos que lo conforman. Aunado a esto, el avance de la tecnología ha permitido la captación de grandes cantidades de imágenes, lo que hace imposible su estudio de forma manual y origina la automatización de esta actividad [22].

Recientemente ha habido un gran interés por parte de los astrónomos en aplicar algoritmos de aprendizaje automático y análisis de imágenes para resolver algunos problemas. Ejemplos de estos trabajos incluyen Clasificación de galaxias [2], [22], de estrellas [3], galaxias/estrellas [23],entre otros; en donde se han utilizado algoritmos de aprendizaje automático como arboles de decisión, métodos basados en instancias, redes neuronales, máquinas de vectores de soporte, ensambles de clasificadores, entre otros.

Sin embargo, en ninguno de estos trabajos se ha tratado el problema de tener conjuntos no balanceados. En consecuencia, surge nuestro interés en investigar y proponer soluciones para este problema.

1.5. Cronograma de actividades

Tabla 1.1: Cronograma de actividades por semana

Actividades	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Investigación del estado del arte	*	*	*	*										
Planteamiento del problema		*	*	*	*	*								
Primer entrega del reporte						*								
Investigar temas de Aprendizaje Automático					*	*	*	*	*	*	*	*	*	
Investigar Algoritmos de balanceo de datos					*	*	*	*	*	*	*	*	*	
Investigar Algoritmos de clasificación de datos					*	*	*	*	*	*	*	*	*	
Investigar conceptos de astronomía					*	*	*	*	*	*	*	*	*	
Aprender el uso de Weka					*	*	*	*	*	*	*	*	*	
Segunda entrega del reporte												*		
Elaboración delmarco Teórico					*	*	*	*	*	*	*	*	*	*
Elaborar el protocolo	*	*	*	*	*	*	*	*	*	*	*	*	*	*

Tabla 1.2: Cronograma de actividades por semana

Actividades	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Experimento para el conjunto de datos del algoritmo SMOTE	*	*	*	*										
Experimento para el conjunto de datos del algoritmo Resample			*	*	*	*	*	*						
Experimeto para el conjunto de datos del algoritmo SpreadSubSample			*	*	*	*	*	*						
Experimento para la clasificación de datos			*	*	*	*	*	*	*	*	*	*		
Resultados														
Elaboración del reporte técnico	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1.6. Recursos de hardware y software

Para el desarrollo de este proyecto, se requieren los recursos siguientes:

1.6.1. Recursos de hardware

- Computadora con sistema operativo Windows XP,
- Procesador Pentium Dual Core
- 4 GB en RAM

1.6.2. Recursos de software

- Weka 3.7.0
- Matlab 7.0

1.7. Alcances y limitaciones

- Sólo se usarán los algoritmos implementados en Weka.

- Sólo se trabajará con clasificación de galaxias.
- Sólo se usará PCA para caracterizar a las imágenes.

Capítulo 2

Marco teórico

2.1. Aprendizaje automático

El aprendizaje automático es una rama de la Inteligencia Artificial cuyo objetivo es desarrollar programas que permitan a las computadoras aprender. De forma más concreta, se trata de crear programas capaces de generalizar comportamientos a partir de una información no estructurada suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento. Estudia cómo construir programas que mejoren el desempeño a través de la experiencia. Dentro de las aplicaciones de aprendizaje automático están: motores de búsqueda, diagnósticos médicos, detección de fraude en el uso de tarjetas de crédito, análisis del mercado de valores, clasificación de secuencias de ADN, reconocimiento del habla y del lenguaje escrito, juegos y robótica [1].

De tal manera Decimos que un programa de computadoras aprende de la experiencia E con respecto a una clase de tareas R y a una medida de desempeño, si su desempeño en las tareas T, evaluado mediante P, se incrementa con E [27]. Dada la definición anterior, para poder modelar y resolver un problema mediante Aprendizaje Automático se deben especificar sus tres características: la clase de tareas T, la medida de desempeño a ser

mejorada P y la fuente de experiencia E. Por ejemplo, un programa de computadora que aprende a jugar rompecabezas podría mejorar su desempeño a través de la experiencia obtenida por los juegos contra los expertos humanos o la propia.

El aprendizaje automático está relacionado con otras disciplinas como inteligencia artificial, probabilidad y estadística, biología, teoría de la información entre otros. Los algoritmos más utilizados son:

- Aprendizaje basado en árboles de decisión
- Aprendizaje basado en redes neuronales artificiales
- Aprendizaje probabilístico y bayesiano
- Aprendizaje basado en instancias
- Aprendizaje evolutivo
- Aprendizaje lógico inductivo
- Aprendizaje por refuerzo

2.1.1. Aprendizaje supervisado

El aprendizaje supervisado es una técnica de aprendizaje automático para deducir una función de datos de entrenamiento. Los datos de entrenamiento consisten de pares de objetos de entrada (normalmente vectores) y los resultados deseados. La salida de la función puede ser un valor constante (llamado de regresión), o puede predecir una etiqueta de clase del objeto de entrada (llamada clasificación), es decir, Encontrar una función que relacione un conjunto de entradas con un conjunto de salidas.

Supongamos que se quiere desarrollar un programa que, cuando se da una imágen de una persona, pueda determinar si la persona es hombre o mujer. Este programa se

llama clasificador, ya que asigna una clase (hombre ó mujer) a un objeto(fotografía). La tarea del aprendizaje supervisado es el de construir un clasificador dado un conjunto de ejemplos de entrenamiento clasificados en este caso, las fotografías, junto con las clases correctas. El desafío clave para el aprendizaje supervisado es el problema de generalización: Después de analizar sólo una muestra de fotografía, el sistema de aprendizaje es la salida de un clasificador que funciona bien con todas las fotografías posibles [7], [1]

.

Existen muchos algoritmos de aprendizaje diferentes que se han desarrollado para la clasificación supervisada y la regresión, éstos pueden ser agrupados de acuerdo con el formalismo que emplean para representar el clasificador aprendido o predictor: Árboles de decisión, reglas de decisión, redes neuronales, funciones lineales discriminantes, redes bayesianas, máquinas de soporte vectorial y métodos del vecino más cercano, entre otros.

2.1.2. Aprendizaje no supervisado

El término de aprendizaje no supervisado se emplea para describir una amplia gama de diferentes tareas de aprendizaje. Como su nombre lo indica, estas tareas analizan un conjunto de datos que no tienen adjunto las etiquetas de clases. Este tipo de aprendizaje es utilizado en los modelos de Estimación de Medidas (EM), *K-means*, mapas auto-organizativos de *Kohonen*.

2.1.3. Algoritmos de aprendizaje automático

Existen varios algoritmos de aprendizaje automático, pero sólo se mencionarán los utilizados para este trabajo.

[1] http://web.engr.oregonstate.edu/ tgd/publications/nature-ecs-machine-learning.pdf

2.1.4. Redes neuronales

Según citan [11] al profundizar en los principios de las redes neuronales artificiales y observar continuamente el término neurona, no es de extrañar que se piense en el cerebro humano, este hecho quizás se deba a que las redes neuronales artificiales están basadas en la inspiración biológica.

El hombre posee cerca de 10 000 000 000 de neuronas masivamente interconectadas, la neurona es una célula especializada que puede propagar una señal electroquímica. Las neuronas tienen una estructura ramificada de entrada, las dendritas y una estructura ramificada de salida, los axones. El cuerpo de la neurona o soma contiene el núcleo, que se encarga de todas las actividades metabólicas de la neurona y recibe la información de otras neuronas vecinas a través de las conexiones sinápticas. Esta estructura se muestra en la Figura 2.1 Los axones de una célula se conectan con las dendritas de otra, por vía de la sinapsis, la neurona se activa y excita una señal electroquímica a lo largo del axón. Esta señal transfiere la sinapsis a otras neuronas, las que a su vez pueden excitarse. Las neuronas se excitan sólo sí la señal total recibida en el cuerpo de las células por conducto de las dendritas, es superior a cierto nivel, conocido como umbral de excitación. Las redes neuronales artificiales tratan de imitar este principio de funcionamiento cerebral.

Las redes neuronales artificiales, denominadas también como RNA, son un paradigma de aprendizaje y procesamiento automático, inspiradas en la forma en que funciona el cerebro. Se trata de un sistema de interconexión de neuronas en una red que colabora para producir una salida [20].

Una red neuronal, según [13], es un sistema de neuronas paralelas conectadas entre

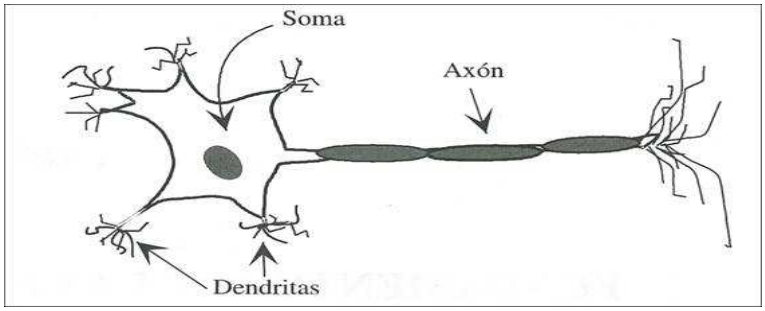

Figura 2.1: Estructura de una neurona biologica.

sí, donde cada neurona de la red se representa como un nodo. Estas conexiones establecen una estructura jerárquica que tratando de emular la fisiología del cerebro, busca nuevos modelos de procesamiento para solucionar problemas concretos del mundo real. Lo importante en el desarrollo de la técnica de las RNA es su útil comportamiento al aprender, reconocer y aplicar relaciones entre objetos y tramas de objetos propios del mundo real [20].

En [13], se destaca que las propiedades que hacen especialmente atractivas a las redes neuronales para ser usadas en una gran cantidad de problemas prácticos son las siguientes:

- *Simulación de sistemas distribuidos no lineales.* Una neurona es un elemento no lineal por lo que una interconexión de ellas también será un dispositivo no lineal.
- *Son sistemas tolerantes a fallos.* Una red neuronal, al ser un sistema distribuido, permite el fallo de algunos elementos individuales sin alterar significativamente la respuesta total del sistema.
- *Adaptabilidad.* Una red neuronal tiene la capacidad de modificar los parámetros de los que depende su funcionamiento de acuerdo con los cambios que se produzcan

en su entorno de trabajo.

Estructura

Segun [17], las RNA imitan la estructura del sistema nervioso, con la intención de construir sistemas de procesamiento de información que puedan presentar un comportamiento inteligente.

[11], describen que las redes neuronales artificiales están formadas por una gran cantidad de neuronas, éstas no suelen denominarse neuronas artificiales sino nodos o unidades de salida. Un nodo o neurona cuenta con una cantidad variable de entradas que provienen del exterior (X_1, X_2, ..., X_m). A su vez dispone de una sola salida (X_j) que transmitirá la información al exterior o hacia otras neuronas. Cada X_j o señal de salida tiene asociada una magnitud llamada peso, éste se calcula en función de las entradas, por lo cual cada una de ellas es afectada por un determinado peso ($W_{jo}...W_{jq+m}$). Los pesos corresponden a la intensidad de los enlaces sinápticos entre neuronas y varían libremente en función del tiempo en cada una de las neuronas que forman parte de la red. La Figura 2.2, ilustra la estructura de una neurona, según lo citado por [11].

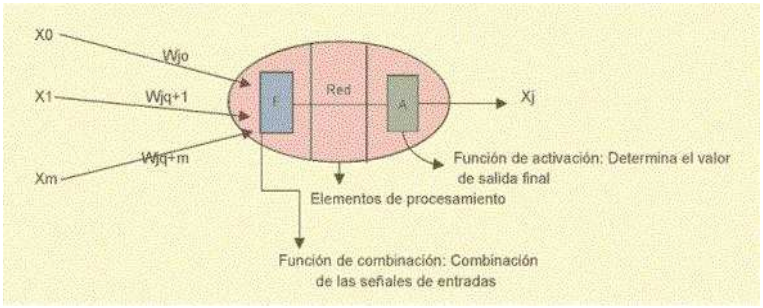

Figura 2.2: Esquema de una neurona

La Figura 2.3 muestra las partes en que está dividida una red neuronal. Como [17] explica, cada neurona realiza una función matemática. Las neuronas se agrupan en ca-

pas,las neuronas están diseñadas y entrenadas para llevar a cabo una labor específica. Finalmente, una o varias redes, más las interfaces con el entorno, conforman un sistema neuronal.

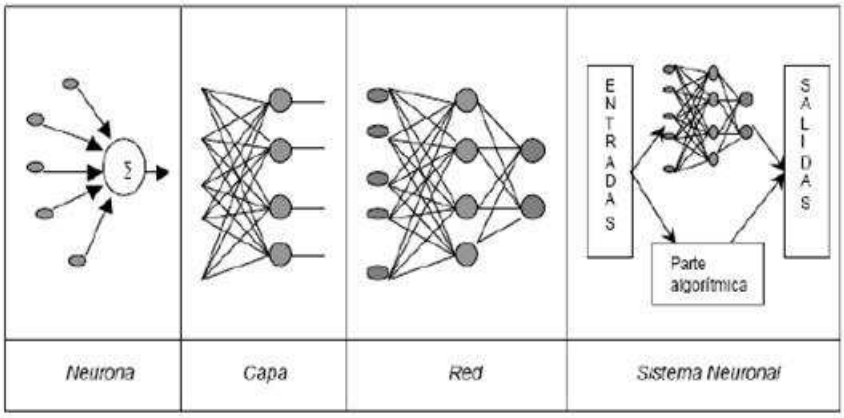

Figura 2.3: Estructura jerárquica de un sistema basado en RNAs

2.1.5. Métodos basados en instancias

En el método basado en instancias, se almacenan los ejemplos de entrenamiento y cuando se quiere clasificar un nuevo objeto, se extraen los objetos más parecidos a este método y se usa su clase para clasificarlo. También se conoce como *lazy learning* ó *memory based learning* donde los datos de entrenamiento se procesan sólo hasta que se requiere (cuando se requiere clasificar datos nuevos) y la relevancia de los datos se mide en función de una medida de distancia [2] .

En el aprendizaje basado en instancias, cada nueva instancia se compara con las existentes usando una métrica de distancia, y la instancia más próxima se usa para asignar su clase a la instancia nueva. La variante más sencilla de éste método de clasificación

[2] http://ccc.inaoep.mx/ jagonzalez/ML/c-ibs.pdf

es conocido como el vecino más cercano *(Nearest-neighbor)*. Otra variante conocida como el método de los k vecinos más cercanos (*k- Nearest-neighbor*), usa k vecinos más cercanos, en cuyo caso la clase mayoritaria de estos k vecinos se asigna a la nueva instancia.

En la Figura 2.4 se observa que a la nueva instancia N se le ha de asignar la clase *a* ya que, entre los vecinos más próximos (marcados en un círculo), hay más instancias de la clase *a*.

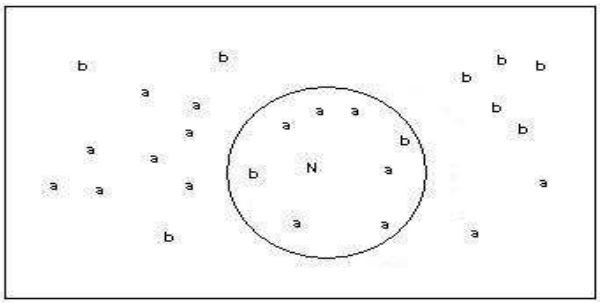

Figura 2.4: Ejemplo de Método Basado en Instancias

2.1.6. Datos no balanceados

Un conjunto de datos no es balanceado si las clases no son aproximadamente igual representadas, esto es alguna de las clases contienen un número mucho menor de ejemplos que las otras clases.(Ver Figura 2.5)[8].

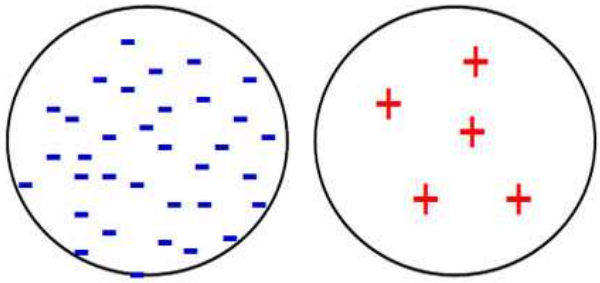

Figura 2.5: Conjunto de datos no balanceados

2.1.7. Algoritmos para manejar datos no balanceados *SMOTE*

SMOTE (*Syntetic Minority Over-sampling Technique*)[18] es un algoritmo que genera instancias sintéticas ó artificiales para equilibrar la muestra de datos basado en la regla del vecino más cercano. La generación se realiza extrapolando nuevas instancias en lugar de duplicarlas como hace el algoritmo de *re-Sampling*. Para cada una de las instancias minoritarias se buscan las instancias minoritarias vecinas (más cercanas) y se crean N instancias entre la línea que une la instancia original y cada una de las vecinas. El valor de N depende del tamaño de *Over-sampling* deseado. Para un caso del 200 % por cada instancia de la clase minoritaria deben crearse dos nuevas instancias genéricas

El algoritmo de SMOTE realiza los siguientes pasos [12],[28]:

- Recibe como parámetro el porcentaje de ejemplos a sobre-muestrear.
- Calcula el número de ejemplos que tiene que generar.
- Calcula los k vecinos más cercanos de los ejemplos de la clase minoritaria.
- Genera los ejemplos siguiendo este proceso:
    - Para cada ejemplo de la clase minoritaria, elige aleatoriamente el vecino a utilizar para crear el nuevo ejemplo.

- Para cada atributo del ejemplo a sobre-muestrear, calcula la diferencia entre el vector de atributos muestra y el vecino elegido.
    - Multiplica esta diferencia por un número aleatorio entre 0 y 1.
    - Suma este último valor al valor original de la muestra.
    - Devuelve el conjunto de ejemplos sintéticos.

Algoritmo SMOTE(T, N, k)

Código

Input:Número de clases minoritarias T; Cantidad de SMOTE N %; Número de vecinos mas cercanos k

Output: (N/100)* T Muestras de clases minoritarias sinteticas 1.(*Si N es menor que 100 %,las muestras de las clases minoritarias aleatorias son solo un porcentaje de ellos que sería SMOTEd.*)

2. if N < 100

3. Entonces T es la muestras de clases minoritariass.

4. T =(N/100)*T

5. N = 100

6. endif

7. N = (int)(N/100)(*La cantidad de SMOTE es simulado para múltiples integrantes de 100*)

8. k = Numero de vecinos mas cercanos

9. numattrs = Numero de atributos

10. Sample[ ][ ]: Arreglo para muestras de clases minoritarias originales

11. newindex: Se queda con una cantidad d enumero de muestas sinteticas generadas,y se inicia con 0.

12. Synthetic[ ][ ]: Arreglo para muestras sinteticas(*Calcula los k vecinos mas cercanos slo para cada muestra de clases minoritaria*)

13. for i $\longleftarrow$ 1 a T

14 Calcula los k vecinos mas cercanos para i, y guarda los indice en nnarray.

15. Populate(N, i, nnarray)

16. endfor

Populate(N, i, nnarray)(*Function to generate the synthetic samples*)

17. while $N \neq 0$

18. Elige un numero aleatorio entre 1 y k, llama nn. Este paso elige uno de los k vecinosmas cercanos de i.

19. for attr $\longleftarrow$ 1 a numattrs

20. Calcula: dif = Sample[nnarray[nn]][attr].Sample[i][attr]

21. Compute: gap = random number between 0 and 1

22. Synthetic[newindex][attr] = Sample[i][attr] + gap $*$ dif

23. endfor

24. newindex++

25. N = N - 1

26. endwhile

27.return (*End of Populate*)

Fin del Pseudocodigo

2.1.8. Resample

El método de *resampling* incluye *under-sampling* y *over-sampling*. El *under-sampling* elimina ejemplos de las clases mayoritarias mientras que el *over-sampling* incrementa ejemplos de clases minoritarias. Se utiliza normalmente en grandes conjuntos de datos

en los que hay suficientes datos redundantes que deben eliminarse.

Este método hace un *over-sampling* aleaotorio de un conjunto de datos usando cualquier muestreo con remplazo o sin remplazo.

Muestreo con reemplazo:

Considere la posibilidad de una población de sacos de papas, cada uno de los cuales tiene ya sea 12, 13, 14, 15, 16, 17, o 18 papas, y todos los valores son igualmente probables. Supongamos que, en esta población, se tiene un saco para cada número. Así que toda la población tiene siete sacos. Si muestra dos con reemplazo, entonces el primero en elegir uno (14). Se tiene un 1/7 de probabilidad de ser elegido uno. Entonces que se sustituya y que se elija otro. Cada uno de ellos aún tiene 1/7 de probabilidad de ser elegido. Hay exactamente 49 posibilidades diferentes, éstas son: (12,12), (12,13), (12, 14), (12,15), (12,16), (12,17), (12,18), (13,12), (13,13), (13,14), etc.

Muestreo sin reemplazo:

Considere la posibilidad de la misma población de sacos de papas, cada uno de los cuales tiene ya sea 12, 13, 14, 15, 16, 17, o 18 papas, y todos los valores son igualmente probables. Supongamos que, en esta población, se tiene un saco para cada número. Así que toda la población tiene siete sacos. Si me muestra dos sin reemplazo, entonces el primero en elegir uno (14). Se tiene un 1/7 de probabilidad de ser elegido uno. Entonces se elije otro. En este punto, sólo hay seis posibilidades: 12, 13, 15, 16, 17 y 18, así que sólo hay 42 posibilidades diferentes, éstas son: (12,13), (12,14), (12,15), (12,16), (12,17), (12,18), (13,12), (13,14), (13,15), etc.

Diferencia

Cuando es muestreo con reemplazo, los dos valores de la muestra son independientes. En la práctica, esto significa que lo que se tiene en el primer experimento no afecta a lo que se tiene en el segundo. Matemáticamente, esto significa que la covarianza entre los dos es igual a cero.

En el muestreo sin reemplazo, los dos valores de la muestra no son independientes. En la práctica, esto significa que lo que se tiene en el primer afecta a lo se puede conseguir para el segundo. Matemáticamente, esto significa que la covarianza entre los dos no es igual a cero.

2.1.9. Definición de Algoritmos

BayesNet: Es un modelo probabilístico multivariado que relaciona un conjunto de variables aleatorias mediante un grafo dirigido que indica explícitamente influencia causal. Las redes Bayesianas son gráficos acíclicos dirigidos cuyos nodos representan variables y los arcos que los unen codifican dependencias condicionales entre las variables. Los nodos pueden representar cualquier tipo de variable, ya sea un parámetro medible (o medido), una variable latente o una hipótesis. [3]

Naive Bayes (NB): es una de las técnicas de clasificación más ampliamente usada, debido a su proceso computacionalmente simple incluso para un conjunto de datos de entrenamiento grande. Está basado en el teorema de Bayes, que puede predecir la probabilidad de que un caso dado pertenezca a una clase determinada. Su simplicidad computacional se debe a la suposición conocida como independencia condicional de clase (supone que el efecto de un valor de atributo sobre una clase dada es independiente de los valores de los otros atributos) y en este sentido es considerado ingenuo[10].

NaiveBayes Simple: Clase para la creación y el uso de un simple clasificador Naive Bayes. Los atributos numéricos se toman por una distribución normal.

RBF Network: Las redes neuronales basadas en funciones de base radial (RBFNs) son un modelo inspirado de las redes neuronales artificiales de una única capa oculta, además de las capas de entrada y de salida convencionales, su especifidad reside en el

[3] http://es.wikipedia.org/wiki/Red_bayesiana

modo en el que las neuronas de la dicha capa oculta tratan los vectores de entrada.[4]

SMO: (*Sequential Minimal Optimization*) Algoritmo de entrenamiento para el clasificador de máquinas con vectores de soporte, es una implementacion de WEKA del algoritmo SWM (*Support Vector Machine*)[5] .

LWL: La Regresión Localmente Ponderada, pertenece a la familia de los algoritmos basados en instancias, utiliza las cercanías de los ejemplos de entrenamiento para formar una aproximación local a la función objetivo *f*. La idea es almacenar todos los datos de entrenamiento disponibles con el objetivo de poder clasificar nuevas instancias [6] .

J48: Genera árboles de decisión para atributos discretos y continuos, utiliza la razón de ganancia para seleccionar el atributo de cada nodo y aplica estrategias de poda para reducir el ruido de los datos de entrenamiento [7] .

J48graft: Es una extension de J4 que realiza la poda.

Random Forest: Este algoritmo construye muchos árboles de decisión. Cada árbol se construye escogiendo, entre los atributos que describen a las instancias, un pequeño subconjunto aleatorio. Para clasificar un nuevo objeto, se le da el vector que lo describe a cada árbol, los cuales hacen su clasificación independientemente. Cada clasificación es un voto. El algoritmo selecciona la clasificación que más votos obtenga [15].

2.1.10. Medidas para evaluación

Matriz de Confusión

Una matriz de confusión contiene información acerca de las clasificaciones actuales y previstas realizado por un sistema de clasificación. Da una información muy útil porque no sólo refleja los errores producidos sino también informa del tipo de éstos. La siguiente

[4] http://cedi2005.ugr.es/2005/pdf/16/970.pdf
[5] http://www.cs.waikato.ac.nz/ml/weka/
[6] http://ccc.inaoep.mx/ villasen/index$_a rchivos$/tesis/TesisMaestria-AaronPancardo.pdf
[7] http://wiki.pentaho.com/display/DATAMINING/J48graft

tabla muestra la matriz de confusión para un clasificador de la clase dos [8] [9] .

Las entradas de la matriz de confusión tienen el siguiente significado:

- TN es el número de predicciones correctas que una instancia es negativa,
- FP es el número de predicciones incorrectas que una instancia es positiva,
- FN es el número de predicciones incorrectas de que una instancia sea negativa, y
- TP es el número de predicciones correctas de que una instancia sea positiva.

Tabla 2.1: ===**Matriz de confusión**===

clase	Positivos Predecidos	Negativos Predecidos
Negativo	FP	TN
Positivo	TP	FN

Precisión

Precisión (P) es la proporción de casos positivos que predijeron que eran correctas las muestras de las clases, según lo calculado mediante la fórmula: [10]

$$Precision = valorpredictivopositivo = \frac{TP}{TP + FP} \tag{2.1}$$

Recall

Recall es la proporción de casos positivos que fueron identificados correctamente, según lo calculado mediante la fórmula:

$$Recall = TPrate = \frac{TP}{TP + FN} \tag{2.2}$$

F-Measure

[8] http://www2.cs.uregina.ca/ dbd/cs831/notes/confusion$_m atrix$/confusion$_m atrix$.html
[9] http://www.metaemotion.com/diego.garcia.morate/download/weka.pdf
[10] http://www.stanford.edu/ maureenh/quals/html/ml/node130.html

La media armónica de precisión y recall. Da un valor, una especie de alternativa al área bajo la curva ROC. [11]

$$F = \frac{2 * precision * recall}{precision + recall} \tag{2.3}$$

ROC-Area

Características operativas del receptor (ROC), ilustran el compromiso entre la sensibilidad y especificidad. Las curvas ROC grafican la tasa de Verdaderos Positivos con los de la tasa de Falso Positivos, en diferentes puntos de la curva [12] .

La sensibilidad es la probabilidad de clasificar correctamente a un individuo cuyo estado real sea el definido como positivo respecto a la condición que estudia la prueba, razón por la que también es denominada fracción de verdaderos positivos (FVP). La especificidad es la probabilidad de clasificar correctamente a un individuo cuyo estado real sea el definido como negativo. Es igual al resultado de restar a uno la fracción de falsos positivos (FFP) [13] .

2.2. Comprensión de datos

La compresión de datos es la codificación de un conjunto de datos D en un cuerpo de datos más pequeño D'[9]. Para comprimir los datos, los métodos de compresión examinan los datos, buscan redundancia en ellos, e intentan removerla. Una parte central en la compresión es la redundancia en los datos. Sólo los datos con redundancia pueden comprimirse aplicando un método o algoritmo de compresión que elimine o remueva de alguna forma dicha redundancia. La redundancia depende del tipo de datos (texto, imágenes, sonido, etc.), por tanto, no existe un método de compresión universal que

[11] http://www.stanford.edu/ maureenh/quals/html/ml/node131.html
[12] http://www.stanford.edu/ maureenh/quals/html/ml/node129.html
[13] http://www.fisterra.com/mbe/investiga/curvas$_r$oc/curvas$_r$oc.htm

pueda ser óptimo para todos los tipos de datos [6]. El desempeño de los métodos de compresión se mide en base a dos criterios: la razón de compresión y el factor de compresión, siendo el segundo el inverso del primero. Las relaciones para determinar estas medidas están dadas por las igualdades de las fórmulas 2.1 y 2.2 .

Entre mayor redundancia exista en los datos, mejor razón *(factor)* de compresión será obtenido.

$$Razondecompresion = \frac{Nobytesarchivocomprimido}{Nobytesarchivooriginal} \tag{2.4}$$

$$factordecompresion = \frac{Nobytesarchivooriginal}{Nobytesarchivocomprimido} \tag{2.5}$$

La compresión de datos puede dividirse en dos tipos principales: compresión con pérdida y compresión sin pérdida. En compresión sin pérdida, es posible reconstruir exactamente los datos originales D dado D'. Al proceso de reconstrucción se le denomina descompresión. Por otra parte, en compresión con pérdida, la descompresión produce solamente una aproximación D* a los datos originales D. La compresión de datos sin pérdida es comúnmente usada en aplicaciones como compresión de texto, donde la pérdida de un solo bit de información es inaceptable. La compresión con pérdida es usada a menudo para aplicaciones de compresión de imágenes y audio (video) destinadas al entretenimiento. No es el caso en aplicaciones de compresión de imágenes médicas (identificación de tumores o anomalías) [6]. Los métodos de compresión con pérdida logran mejores razones de compresión. El reto que siempre se persigue es conseguir un método de compresión que emplee el tiempo más corto, que sea posible comprimir la mayor cantidad de información y que no exista pérdida de datos.

2.2.1. Análisis de componentes principales (PCA)

El Análisis de Componentes Principales (PCA) pertenece a un grupo de técnicas estadísticas multivariantes, eminentemente descriptivas. PCA transforma un conjunto de variables en un número menor de variables (llamadas dimensiones, componentes principales o componentes), no correlacionadas entre sí, que contienen la mayor parte de la información (varianza) del conjunto inicial. PCA busca guardar la información de un gran número de variables en un pequeño número de componentes no correlacionados, con la mínima pérdida de información. Las nuevas variables(componentes principales) son obtenidas como combinaciones lineales de las variables originales. Los componentes se ordenan en función del porcentaje de varianza de cada uno de ellos. En este sentido, el primer componente será el más importante por ser el que explica mayor porcentaje de la varianza de los datos [24].

Cálculo de los componentes principales

Se considera una serie de variables $(x_1, x_2, ..., x_p)$ sobre un grupo de objetos o individuos y se trata de calcular, a partir de ellas, un nuevo conjunto de variables y_1, y_2, ..., y_p, en correlación entre sí, cuyas varianzas vayan decreciendo progresivamente. Cada y_j (donde j = 1, . . . , p) es una combinación lineal de las x_1, x_2, ..., xp originales, es decir:

$$y_j = x_1 + aj_2\ x_2 + ... + a_{jp}\ x_p =$$
$$= a'_j x$$

siendo a'$_j$ = $(a_{1j}\ ,\ a_{2j}\ ,\ ...,\ a_{pj})$ un vector de constantes,y

$$\mathrm{X}=[X_1 ... X_p]$$

Si lo que que se desea es maximizar la varianza, como se muestra a continuación, una forma simple podría ser aumentar los coeficientes a_{ij} . Por ello, para mantener la

ortogonalidad de la transformación se impone que el módulo del vector a_{0j} =(a_{1j} , a_{2j} ,..., a_{pj}), es decir:

$$a'_j a_j = \sum_{k=1}^{P} a_{kj}^2 = 1 \tag{2.6}$$

El primer componente se calcula eligiendo a_1 de modo que y_1 tenga la mayor varianza posible, sujeta a la restricción de que $a'_1 a_1 = 1$. El segundo componente principal se calcula obteniendo a_2 de modo que la variable obtenida, y_2 esté en correlación con y_1. Del mismo modo se eligen y_1, y_2, , y_p, en correlación entre sí, de manera que las variables aleatorias obtenidas vayan teniendo cada vez menor varianza [25].

2.3. Conceptos de astronomía

Astronomía (Del griego astron: astro y nomos: Ley). Ciencia que tiene por objeto el estudio del universo, de los cuerpos que lo constituyen, de las posiciones relativas que éstos ocupan, de las leyes que gobiernan sus movimientos y de la evolución que experimentan a lo largo del tiempo. Esta disciplina comprende tres ramas principales: la astronomía de posición y la mecánica celeste, que se encargan de determinar las coordenadas de los astros y estudian la magnitud de su variación natural; la astrofísica, en sus aspectos aplicado y teórico, que estudia las leyes físicas que rigen su comportamiento, y la cosmología, que estudia las leyes generales de la estructura, el origen y la evolución del universo como un todo.

Universo: Conjunto de todo lo existente, éste contiene galaxias, cúmulos de galaxias y estructuras de mayor tamaño, llamadas supercúmulos, amén de materia intergaláctica. La edad del universo se estima en unos 15000 millones de años.

Galaxia: Conjunto de estrellas y de materia interestelar, ligadas a interacciones gravitatorias, que presenta las mismas características que la Galaxia (Vía Láctea) a la

que pertenece nuestro sistema solar. Tipos principales:

Galaxias elípticas

Las galaxias elípticas, contienen una gran población de estrellas viejas, normalmente poco gas y polvo, y algunas estrellas de nueva formación. Las galaxias elípticas tienen gran variedad de tamaños, desde gigantes a enanas. Hubble simbolizó las galaxias elípticas con la letra E y las subdividió en ocho clases, desde la E0, prácticamente esféricas, hasta la E7, usiformes. En las galaxias elípticas la concentración de estrellas va disminuyendo desde el núcleo, que es pequeño y muy brillante, hacia sus bordes (Ver figura 2.6)[14] .

Figura 2.6: Galaxia elíptica gigante

Galaxias espirales o espirales normales

Las galaxias espirales son discos achatados que contienen no sólo algunas estrellas viejas sino también una gran población de estrellas jóvenes, bastante gas y polvo, y nubes moleculares que son el lugar de nacimiento de las estrellas. Generalmente, un halo de débiles estrellas viejas rodea el disco, y suele existir una protuberancia nuclear

[14] http://www.astromia.com/universo/clasegalaxias.htm

más pequeña que emite dos chorros de materia energética en direcciones opuestas. Las galaxias espirales se designan con la letra S. Dependiendo del menor o mayor desarrollo que posea cada brazo, se le asigna una letra a, b o c (Sa, Sb, Sc, SBa, SBb,SBc) (Ver Figura 2.7).

Figura 2.7: Galaxia espiral

Galaxias irregulares

Las galaxias irregulares se simbolizan con la letra I o IR, aunque suelen ser enanas o poco comunes. Se engloban en este grupo aquellas galaxias que no tienen estructura y simetría bien definidas. Se clasifican en irregulares de tipo 1 o magallánico, que contienen gran cantidad de estrellas jóvenes y materia interestelar y galaxias irregulares de tipo 2, menos frecuentes y cuyo contenido es difícil de identificar. Las galaxias irregulares se sitúan generalmente próximas a galaxias más grandes, y suelen contener grandes cantidades de estrellas jóvenes, gas y polvo cósmico. (Ver figura 2.8)

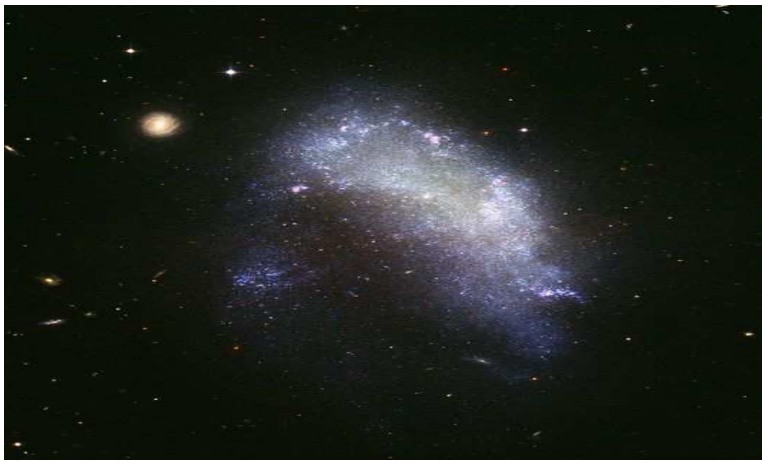

Figura 2.8: Galaxia irregular

Vía Láctea: Banda luminosa, formada por múltiples estrellas, nubes de polvo y gas de nuestra galaxia, vista de la posición que ocupa la Tierra en el sistema solar. Es una galaxia espiral.

Sistema Solar: Agrupación formada por una estrella (el Sol) y los planetas y demás cuerpos que orbitan a su alrededor. Está formado por un cuerpo central (el Sol, que supone un 99,85 % de la masa total) y diversos cuerpos que giran a su alrededor (los planetas y sus satélites, los asteroides, los cometas, etc.)

Estrella: Las estrellas son cuerpos gaseosos cuya existencia se debe al equilibrio entre la presión de radiación y las fuerzas gravitatorias.

Nebulosa: Acumulación de gas y polvo interestelares. Los diversos tipos de nebulosas representan los diferentes estadíos que permiten al universo reutilizar la materia que se ha dispersado con anterioridad, durante los fenómenos violentos que tienen lugar en él.

Planeta: Cuerpo celeste que carece de luz propia y describe una órbita, generalmente elíptica y de poca excentricidad, alrededor del Sol u otra estrella cualquiera.(los nueve cuerpos principales del sistema solar son: Mercurio, Venus, Tierra, Marte, Júpiter,

Saturno, Urano, Neptuno y Plutón).

2.4. Método de clasificación

La parte del método de clasificación está dividido en tres partes: (La estandarización de la imagen, compresion de datos y la clasificación).

El método funciona de la siguiente manera:

En primer lugar, las imágenes son rotadas, centradas,cortadas, ampliadas y redimensionadas de forma totalmente automática. Entonces, estas imágenes estandarizadas son reducidas en dimensión y se encuentran sus conjuntos de características(componentes principales). La proyección de imágenes en los componentes principales serán los parámetros de entrada para realizar la última tarea: la clasificación [21].

2.4.1. Estandarización de la imagen

El objetivo de este paso es crear imágenes invariantes a color, posición, orientación y tamaño, porque las imágenes de galaxias en general son diferentes en tamaño y formatos de color y la mayoría de las veces la galaxia contenida en la imagen no está en el centro.

Primero, se encuentra la galaxia contenida en la imagen aplicando un umbral. Entonces se obtiene la fila del centro y de la columna de la imagen de la galaxia. Después se obtiene la matriz de covarianza de los puntos en la imagen de galaxia. El eje principal de la galaxia está dado por el primer eigenvector (el eigenvector con el mayor eigenvalor correspondiente) de la matriz de covarianza. Por lo tanto, se hace girar la imagen de modo que el eje principal sea horizontal. Luego, se centra la galaxia y se recorta la imagen, la eliminación de las columnas que contienen sólo el fondo (negro) píxeles. Por último, se amplía y se cambia el tamaño de las imágenes a un tamaño de 128*128 [21].

2.4.2. Compresión de datos

Una idea general para la clasificación de galaxias es extraer la información relevante en una imagen de la galaxia, codificar de la manera más eficaz posible, y comparar una galaxia decodificada con una base de datos de imágenes de manera similar codificada. Se utiliza el análisis de componentes principales (PCA) para encontrar esta información relevante.

La idea básica del PCA es encontrar los eigenvectores de la matriz de covarianza del conjunto de imágenes de galaxias, a fin de que explique la máxima cantidad de varianza posible. Estos eigenvectores propios puede ser pensado como un conjunto de características que juntos caracterizan la variación entre las galaxias.

Las imágenes estandarizadas obtenidas en el paso anterior, de tamaño 128 * 128 pixels, describen un vector de dimensión de 16.384. Por lo tanto, se tiene una matriz de tamaño de 16.384 por el número de imágenes de la galaxia. Sin embargo, mediante la reducción de los datos mediante PCA, se obtiene una matriz más pequeña de los PC de tamaño por el número de imágenes de la galaxia [21].

2.4.3. Clasificación

Una vez que se han reducido la dimensionalidad de los datos, y se encontró el parámetro de entrada (componentes principales), se puede realizar la tarea de clasificación.

En primer lugar, los algoritmos están entrenados y un subconjunto de las imágenes de galaxias son proporcionanados al algoritmo de aprendizaje entrenado como conjunto de prueba para llevar a cabo la clasificación de galaxias. Finalmente se tendrá la clasificacón para ese subconjunto. [21]

2.4.4. Manejo de datos no balanceados

Es esta sección se utilizarán los algoritmos de balanceo de datos para obtener mejores resultados para la clasificación; los cuáles son: SMOTE, resampling, spreadsubsample.

2.5. Herramienta Weka

WEKA[15] es una herramienta de aprendizaje automático y de minería de datos, escrita en lenguaje Java, gratuita y desarrollada en la Universidad de Waikato (WEKA = *Waikato Environment for Knowledge Analysis*).

WEKA es una recopilación de algoritmos para aprendizaje automático y herramientas de pre-procesamiento de datos. Además proporciona soporte para todo el proceso experimental: evaluación, preparación y visualización de datos y resultados. Contiene métodos de clasificación, regresión, agrupamiento y reglas de asociación.

[16] .

WEKA se puede utilizar de 3 formas distintas, desde la línea de comandos, desde una de las interfaces de usuario o creando un programa de java:

A: Desde la línea de comandos

Cada uno de los algoritmos incluidos en WEKA se puede invocar desde la línea de comandos de MS-DOS como programas individuales. Los resultados se muestran únicamente en modo texto.

B: Desde una de las interfaces de usuario

WEKA dispone de 4 interfaces de usuario distintos, que se pueden elegir después de lanzar la aplicación completa. Las interfaces son:

- Interfaz 1. Simple CLI (*command line interface*): interfaz en modo texto.

[15] http://www.cs.waikato.ac.nz/ml/weka/
[16] www.metaemotion.com/diego.garcia.morate/download/weka.pdf

Esta interfaz proporciona una consola para poder introducir comandos. A pesar de ser en apariencia muy simple es extremadamente potente porque permite realizar cualquier operación soportada por Weka de forma directa; no obstante, es muy complicada de manejar ya que es necesario un conocimiento completo de la aplicación. Su utilidad es pequeña desde que se fue recubriendo WEKA con interfaces mas gráficos. Actualmente ya prácticamente sólo es útil como una herramienta de ayuda a la fase de pruebas.

Los comandos soportados son los siguientes:

java <nombre-de-la-clase> <args> :Ejecuta el método main de la clase dada con los argumentos especificados al ejecutar este comando (en el caso de que realmente se haya proporcionado alguno). Cada comando es atendido en un hilo independiente por lo que es posible ejecutar varias tareas concurrentemente.

*break:*Detiene la tarea actual

kill: Mata la tarea actual. Este comando no es aconsejable, sólo debe usarse en el caso de que la tarea no pueda ser detenida con el comando break.

(Clear Screen): Limpia el contenido de la consola (cls).

exit: Sale de la aplicación.

help <comando>: Proporciona una breve descripción de cada comando.

- Interfaz 2. Explorer: interfaz gráfico básico.

  El modo Explorador es el modo más usado y más descriptivo. Éste permite realizar operaciones sobre un sólo archivo de datos.

  El explorador permite tareas de:

1. Preprocesado de los datos y aplicación de filtros.
2. Clasificación.
3. Agrupamiento.
4. Búsqueda de asociaciones.
5. Selección de atributos.
6. Visualización de datos.

- Interfaz 3. Experimenter:

  El modo experimentador (*Experimenter*) es un modo muy útil para aplicar uno o varios métodos de clasificación sobre un gran conjunto de datos y, luego poder realizar contrastes estadísticos entre ellos y obtener otros índices estadísticos

  Esta interfaz tiene posibilidad de comparar el funcionamiento de diversos algoritmos de aprendizaje.

- KnowledgeFlow:interfaz gráfico que permite interconectar distintos algoritmos de aprendizaje en cascada, creando una red.

C: Creando un programa Java

La tercera forma en la que se puede utilizar el programa WEKA es mediante la creación de un programa Java que llame a las funciones que se desee. El código fuente de WEKA está disponible, con lo que se puede utilizar para crear un programa propio.

2.6. Trabajo Relacionado

Algunos trabajos que se relacionan con el conjunto de datos no balanceados son:

Ling y Li (1998)[4] combinaron las técnicas de *Over-Sampling* y *Under-Sampling*. Ellos usaron un análisis aleatorio en lugar de medir con exactitud una ejecución clasificada. Una curva de elevación es similar a una curva ROC (Características operativas del receptor) , pero es más adaptado para el problema de análisis del mercadeo.

En el 2000, Japkowicz [19] evaluó las técnicas *Over-Sampling* y *Under-Sampling*, concluyó que ambas son efectivas. El *re-Sampling* aleatorio consiste de re-muestrear la clase más pequeña aleatoriamente hasta que conste de tantas muestras como la clase mayoritaria.

Chawla et al. [18] en 2002 ideó un método llamado(*Synthetic Minority Over-Sampling Technique SMOTE*), Técnicas de sobremuestreo de datos minoritarios . Esta técnica crea nuevos ejemplos sintéticos para la clase minoritaria. Para cada clase positiva (clase minoritaria), el vecino más cercano positivo es identificado y es creado utlizando SMOTE para una nueva instancia positiva y se coloca aleatoriamente en la instancia entre uno de sus vecinos.

En 2005, Hui Han [29] presentaron dos nuevos métodos minoritarios de *Over-Sampling*: *borderline-SMOTE1* y *borderline-SMOTE2*, en los cuales sólo los ejemplos minoritarios acerca del *borderline* son *Over-Sampling*. Estos experimentos muestran que estos métodos logran mejor coeficiente TP y valor F que SMOTE y los métodos aleatorios de *over-sampling*.

Kubat y Matwin [16] presentarón una heurística de un método sobre muestreo para balanceo de datos, eliminando ejemplos con ruido y redundantes en una clase mayoritaria.

Capítulo 3

Diseño de Experimentos

El objetivo del trabajo es comparar el desempeño de tres algoritmos de balanceo de datos para clasificar imágenes de galaxias. Para lograrlo, son necesarias tres etapas: estandarización de la imagenes, compresión de datos, y su clasificación.

Las dos primeras partes ya fueron realizadas en trabajos previos [21], y para esta investigación se continuará con la tercera etapa incluyendo métodos de balanceo de datos para su clasificación .

El conjunto de datos original consta de 293 y 310 imágenes de galaxias, así también el número de componentes principales que se ha considerado es el que representa el 80 % de la información en el conjunto de datos, es decir 13 componentes principales. Para este conjunto de datos se consideran tres clases con 18 elípticas, 281 espirales y 11 irregulares, para cinco clases con 18 elípticas, 19 So,104 Sa+Sb, 141 Sc+Sd y 11 irregulares y para siete clases con 18 elípticas, 19 So, 26 Sa, 78 Sb, Sc 133, Sd 8 y 11 irregulares.

Los algoritmos utilizados para los experimentos son: Bayesnet, NaiveBayes, NaiveBayesSimple, SMO, LWL, J48, J48graft y RandomForest, usando la implementación de Weka. Cada algoritmo de clasificación se ejecutó diez veces con semillas aleatorias diferentes e igualmente se hizo para cada uno de los algoritmos de balanceo de datos.

Se usa una técnica llamada 10-fold cross-validation para llevar a cabo todos los experimentos, es decir, se dividieron aleatoriamente los datos en diez subconjuntos de igual tamaño, usando en cada experimento uno de los subconjuntos para la prueba y los otros nueve para el entrenamiento. Los resultados que se muestran son obtenidos por el promedio de las 10 ejecuciones de 10-fold cross-validation. El algoritmo de SMOTE utiliza 5 vecinos más cercanos aleatoriamente, está a un 100 % , además de tener una semilla aleatoria de 1. El Resample también está al 100 % y tiene una semilla aleatoria de 1. Finalmente el SpreadSubSample tiene una semilla aleatoria de 1.

Capítulo 4

Resultados

En la Tabla 4.1 se muestran los resultados de los algoritmos sin hacer sobremuestreo para clasificar 3 tipos de galaxias. los mejores fueron NaiveBayes en Precision con 0.8782, RBF Network en F-Measure con 0.8911 y ROC Area con 0.7167 y finalmente Random Forest en Recall con 0.9113. Los peores fueron SMO y LWL en Precision con 0.822,el algoritmo NaiveBayesSimple clasificó mal en Recall con 0.8069 y en F-Measure con 0.8323 y SMO tambien clasificó mal en ROC-Area con 0.5.

Tabla 4.1: **Resultados para 3 clases sin hacer sobremuestreo**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.8215	0.9036	0.8607	0.6249
Naive Bayes	**0.8782**	0.8229	0.8429	0.6964
NaiveBayes Simple	0.8755	0.8069	0.8323	0.6946
RBFNetwork	0.8776	0.9075	**0.8911**	**0.7167**
SMO	0.822	0.906	0.862	0.5
LWL	0.822	0.906	0.862	0.6444
J48	0.8562	0.883	0.8692	0.5562
J48graft	0.8601	0.8909	0.8748	0.5638
Random Forest	0.8695	**0.9113**	0.8864	0.6979

En la Tabla 4.2 se muestran los resultados de los algoritmos sin hacer sobremuestreo para clasificar 5 tipos de galaxias,los mejores fueron RBF Network en Precision con 0.4433, Recall con 0.4823 y F-Measure con 0.4549 y NaiveBayes Simple en ROC Area

con 0.5815. Los peores son SMO en Precision con 0.2614, F-Measure con 0.3137 y en ROC Area con 0.4997, NaiveBayes Simple en Recall con 0.4265.

Tabla 4.2: **Resultados para 5 clases sin hacer sobremuestreo**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.3128	0.4755	0.3262	0.5387
Naive Bayes	0.4188	0.4303	0.4135	0.5807
NaiveBayes Simple	0.4224	0.4265	0.4119	**0.5815**
RBFNetwork	**0.4433**	**0.4823**	**0.4549**	0.5651
SMO	0.2614	0.4715	0.3137	0.4997
LWL	0.3149	0.4669	0.3336	0.5713
J48	0.4278	0.4512	0.4312	0.544
J48graft	0.4335	0.4557	0.4356	0.5496
Random Forest	0.4317	0.4487	0.4347	0.5738

En la Tabla 4.3 se muestran los resultados de los algoritmos sin hacer sobremuestreo para clasificar 7 tipos de galaxias,los mejores fueron RBF Network en Precision con 0.357, BayesNet y SMO en Recall con 0.454, RBF Network en F-Measure con 0.3752 y Random Forest en ROC Area con 0.589, los peores algoritmos en clasificar son BayesNet y SMO con 0.206, ademas también clasificaron mal en F-Measure con 0.283, el J48 clasifió mal en Recall con 0.3441 y el SMO tambien clasificó mal en ROC Area con 0.5041.

Tabla 4.3: **Resultados para 7 clases sin hacer sobremuestreo**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.206	**0.454**	0.283	0.5323
Naive Bayes	0.3431	0.3849	0.3502	0.5774
NaiveBayes Simple	0.3378	0.3798	0.3447	0.5809
RBFNetwork	**0.357**	0.4226	**0.3752**	0.5761
SMO	0.206	**0.454**	0.283	0.5041
LWL	0.2061	0.4534	0.2831	0.5838
J48	0.3397	0.3441	0.34	0.539
J48graft	0.3373	0.3475	0.3398	0.5339
Random Forest	0.3541	0.4053	0.3742	**0.589**

En la Tabla 4.4 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 3 clases con SMOTE, el algortimo RandomForest clasificó mejor en Precision con 0.8825,Recall con 0.8919 y ROC Area con 0.8078, RBFNetwork clasificó mejor en F-Measure con 0.8632. Los peores resultados son para el algoritmo SMO en Precision con 0.766 y en ROC Area con 0.522, el NaiveBayes Simple en Recall con 0.8118 y bayesNet en F-Measure con 0.8132.

Tabla 4.4: **3 Clases: SMOTE 100 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.767	0.866	0.8132	0.6728
Naive Bayes	0.8678	0.8189	0.8333	0.7764
NaiveBayes Simple	0.8634	0.8118	0.8278	0.7817
RBFNetwork	0.8632	0.8834	**0.8632**	0.7771
SMO	0.766	0.875	0.817	0.522
LWL	0.773	0.8742	0.8173	0.7274
J48	0.831	0.8358	0.8323	0.616
J48graft	0.8384	0.8519	0.8421	0.6145
Random Forest	**0.8825**	**0.8919**	0.8615	**0.8078**

En la Tabla 4.5 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 5 clases con SMOTE,el mejor en Precision con 0.4719 y ROC Area con 0.6205 es el RandomForest pero el algoritmo RBFNetwork tiene mejor promedio en Recall con 0.4914 y F-Measure con 0.4686. Los peores promedios están en los algoritmos de SMO en Precision con 0.2772, F-Measure con 0.2997 y en ROC Area con 0.5249 y J48 en Recall con 0.4207.

En la Tabla 4.6 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 7 clases con SMOTE, el mejor en Precision con .3739, F-Measure con 0.3846 es el RBFNetwork pero el algoritmo BayesNet tiene mejor promedio en Recall con 0.4416 y el RandomForest tiene mejor promedio en ROC Area con 0.605. Los peores promedios son para el algoritmo SMO en Precision con 0.195 y junto con el BayesNet

Tabla 4.5: **5 Clases: SMOTE 100 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.2978	0.4528	0.3063	0.556
Naive Bayes	0.4481	0.4588	0.442	0.6175
NaiveBayes Simple	0.4451	0.4551	0.4374	0.6149
RBFNetwork	0.4667	**0.4914**	**0.4686**	0.6025
SMO	0.2772	0.4574	0.2997	0.5249
LWL	0.3086	0.4609	0.3047	0.6091
J48	0.4034	0.4207	0.4045	0.5558
J48graft	0.408	0.428	0.4107	0.5628
Random Forest	**0.4719**	0.4684	0.459	**0.6205**

en F-Measure con 0.271, SMO en ROC Area con 0.5132 y el J48 en Recall con 0.3455.

Tabla 4.6: **7 Clases: SMOTE 100 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.1952	**0.4413**	0.271	0.5177
Naive Bayes	0.3407	0.376	0.3446	0.5818
NaiveBayes Simple	0.3406	0.3778	0.3433	0.5862
RBFNetwork	**0.3739**	0.43	**0.3846**	0.5885
SMO	0.195	0.442	0.271	0.5132
LWL	0.2223	0.4343	0.2741	0.575
J48	0.3369	0.3455	0.3393	0.5424
J48graft	0.3395	0.3505	0.3422	0.5417
Random Forest	0.3496	0.3943	0.3641	**0.605**

En la Tabla 4.7 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 3 clases con Resample, el mejor algoritmo que tuvo buenos resultados fue Random Forest en Precision con 0.9689, Recall 0.9682, F-Measure con 0.9659 y ROC Area con 0.9211 y los peores promedios los obtuvo SMO en Precision con 0.798 en F-Measure con 0.843 y ROC Area con 0.5 y NaiveBayes Simple en Recall con 0.8633.

En la Tabla 4.8 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 5 clases con Resample, el mejor algoritmo que tuvo buenos resultados fue Random Forest en Precision con 0.7786, Recall 0.7744, F-Measure con 0.7744 y ROC

Tabla 4.7: **3 Clases: Resample**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.9055	0.9361	0.9204	0.8444
Naive Bayes	0.8879	0.8777	0.879	0.8256
NaiveBayes Simple	0.8843	0.8633	0.8697	0.8266
RBFNetwork	0.9272	0.928	0.9257	0.8575
SMO	0.798	0.894	0.843	0.5
LWL	0.8401	0.9012	0.8593	0.8328
J48	0.9336	0.9345	0.933	0.8792
J48graft	0.9468	0.9468	0.944	0.8787
Random Forest	**0.9689**	**0.9682**	**0.9659**	**0.9211**

Area con 0.9072 y los peores promedios los obtuvo SMO en Precision con 0.3913 en F-Measure con 0.4352 y ROC Area con 0.618 y NaiveBayes en Recall con 0.4453

Tabla 4.8: **5 Clases: Resample**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.4979	0.5079	0.4828	0.6805
Naive Bayes	0.4659	0.4453	0.4466	0.6737
NaiveBayes Simple	0.4669	0.4454	0.4465	0.6739
RBFNetwork	0.6005	0.6005	0.5954	0.7415
SMO	0.3913	0.499	0.4352	0.618
LWL	0.4485	0.4815	0.4382	0.6863
J48	0.7163	0.7149	0.7143	0.8045
J48graft	0.7146	0.7133	0.7123	0.8013
Random Forest	**0.7786**	**0.7744**	**0.7744**	**0.9072**

En la Tabla 4.9 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 7 clases con Resample, el mejor algoritmo que tuvo buenos resultados fue RandomForest en Precision con 0.7582, Recall 0.754, F-Measure con 0.7488 y ROC Area con 0.9093 y los peores promedios los obtuvo SMO en Precision con 0.2631, en Recall con 0.3936 F-Measure con 0.2522 y ROC Area con 0.5449.

En la Tabla 4.10 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 3 clases con SpreadSubSample, el mejor algoritmo que tuvo

Tabla 4.9: **7 Clases: Resample**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.3243	0.3989	0.3196	0.5932
Naive Bayes	0.4141	0.3977	0.3875	0.6766
NaiveBayes Simple	0.4124	0.3976	0.386	0.6741
RBFNetwork	0.5564	0.5569	0.5474	0.7585
SMO	0.2631	0.3936	0.2522	0.5449
LWL	0.3224	0.4202	0.3024	0.6638
J48	0.6715	0.6708	0.6686	0.7989
J48graft	0.6657	0.6634	0.6617	0.7951
Random Forest	**0.7582**	**0.754**	**0.7488**	**0.9093**

buenos resultados RBFNetwork en Precision con 0.8785, F-Measure con 0.892 y ROC Area con 0.7258 y Random Forest en Recall con 0.912. Los peores promedios los obtuvo SMO y LWL en Precision con 0.822, NaiveBayes Simple en Recall con 0.8081y en F-Measure con 0.8331 y SMO en ROC Area con 0.5.

Tabla 4.10: **3 Clases: SpreadSubSample**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.8331	0.903	0.863	0.6273
Naive Bayes	0.8762	0.8157	0.8379	0.6854
NaiveBayes Simple	0.8752	0.8081	0.8331	0.6865
RBFNetwork	**0.8785**	0.9078	**0.892**	**0.7258**
SMO	0.822	0.906	0.862	0.5
LWL	0.822	0.906	0.862	0.631
J48	0.8596	0.8848	0.8715	0.556
J48graft	0.8564	0.8931	0.8736	0.5486
Random Forest	0.8707	**0.912**	0.8876	0.6588

En la Tabla 4.11 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 5 clases con SpreadSubSample, los algoritmos que obtuvieron buenos resultados son el RBFNetwork en Precision con 0.4555 y en Recall con 0.4935 y en F-Measure con 0.465 y RandomForest en ROC Area con 0.5961. Los peores promedios los obtuvo SMO en Precision con 0.3002, en F-Measure con 0.3198 y en ROC Area

con 0.5045, NaiveBayes Simple en Recall con 0.4329.

Tabla 4.11: **5 Clases: SpreadSubSample**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.3493	0.4758	0.3272	0.5344
Naive Bayes	0.4308	0.4382	0.4232	0.5877
NaiveBayes Simple	0.4306	0.4329	0.4185	0.5865
RBFNetwork	**0.4555**	**0.4935**	**0.465**	0.5761
SMO	0.3002	0.474	0.3198	0.5045
LWL	0.3254	0.4745	0.339	0.5679
J48	0.4178	0.4399	0.4215	0.5477
J48graft	0.4161	0.442	0.4219	0.5461
Random Forest	0.4452	0.4656	0.4519	**0.5961**

En la Tabla 4.12 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 7 clases con SpreadSubSample, los algoritmos que obtuvieron buenos resultados son el J48graft en Precision con 0.3533, LWL en Recall con 0.4541, RBFNetwork en F-Measure con 0.3787 y Random Forest en ROC Area con 0.5931. Los peores promedios son del algoritmo SMO y BayesNet en Precision con 0.206,y en F-Measure con 0.283, J48 en Recall con 0.3664, y SMO en ROC Area con 0.504.

Tabla 4.12: 7 Clases: SpreadSubSample

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.206	0.454	0.283	0.5366
Naive Bayes	0.3438	0.3832	0.3509	0.5823
NaiveBayes Simple	0.3385	0.3789	0.345	0.5835
RBFNetwork	0.362	0.4201	**0.3787**	0.5894
SMO	0.206	0.454	0.283	0.504
LWL	0.2092	**0.4541**	0.2832	0.5862
J48	0.3526	0.3664	0.3583	0.543
J48graft	**0.3533**	0.3709	0.3606	0.5436
Random Forest	0.3525	0.401	0.373	**0.5931**

En la Tabla 4.13 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 3 clases con SMOTE al 200 %, los algoritmos que tuvieron

buenos resultados son el NaiveBayes en Precision con 0.8493, RBFNetwork en Recall con 0.8645 y en F-Measure con 0.8485 y RandomForest en ROC Area con 0.7982. Los peores resultados son de los algoritmos SMO y LWL en Precision con 0.716, NaiveBayes Simple en Recall 0.7947, BayesNet en F-Measure con 0.7741 y SMO en ROC Area con 0.5002.

Tabla 4.13: **3 Clases: SMOTE 200 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.7449	0.8256	0.7741	0.7058
Naive Bayes	**0.8493**	0.7995	0.8154	0.7877
NaiveBayes Simple	0.8432	0.7947	0.8094	0.7854
RBFNetwork	0.8464	**0.8645**	**0.8485**	0.7707
SMO	0.716	0.846	0.776	0.5002
LWL	0.716	0.846	0.776	0.6989
J48	0.8137	0.8166	0.8145	0.7028
J48graft	0.8218	0.8364	0.8272	0.6935
Random Forest	0.8462	0.8622	0.8303	**0.7982**

En la Tabla 4.14 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 3 clases con Resample al 200 % , el mejor algoritmo que tuvo buenos resultados es el RandomForest en Precision con 0.9888, en Recall con 0.9888, en F-Measure con 0.9885 y en ROC Area con 0.9745.Los peores resultados son de los algoritmos SMO en Precision con 0.784, en F-Measure con 0.832 y en ROC Area con 0.5043, NaiveBayes Simple en Recall con 0.8548.

En la Tabla 4.15 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 5 clases con SOMOTE al 200 %, los algoritmos que tuvieron buenos resultados son el RBFNetwork en Precision con 0.4586, en Recall con 0.4813, en F-Measure con 0.4599, Random Forest en ROC Area con 0.6422, los peores resultados son de los algoritmos LWL en Precision con 0.218, F-Measure con 0.2805, J48 en Recall con 0.4144 y SMO en ROC Area con 0.5113.

Tabla 4.14: **3 Clases: Resample 200 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.9598	0.9592	0.9523	0.944
Naive Bayes	0.8927	0.8589	0.8691	0.8753
NaiveBayes Simple	0.8908	0.8548	0.8664	0.8755
RBFNetwork	0.932	0.9296	0.9275	0.8961
SMO	0.784	0.885	0.832	0.5043
LWL	0.792	0.8852	0.8324	0.7993
J48	0.9724	0.9715	0.9716	0.9679
J48graft	0.982	0.982	0.9818	0.9717
Random Forest	**0.9888**	**0.9888**	**0.9885**	**0.9745**

Tabla 4.15: **5 Clases: SMOTE 200 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.2928	0.4398	0.294	0.5631
Naive Bayes	0.4248	0.4334	0.4171	0.6157
NaiveBayes Simple	0.4277	0.4361	0.4186	0.6154
RBFNetwork	**0.4586**	**0.4813**	**0.4599**	0.619
SMO	0.2731	0.4414	0.2824	0.5113
LWL	0.218	0.4456	0.2805	0.6229
J48	0.3947	0.4144	0.3996	0.5597
J48graft	0.3954	0.4159	0.4008	0.5605
Random Forest	0.4537	0.4507	0.4439	**0.6422**

En la Tabla 4.16 se muestran los resultados de los algoritmos aplicando sobre-muestreo para clasificar 5 clases con Resample al 200 %, el mejor algoritmo que tuvo buenos resultados es el RandomForest Precision 0.9132, Recall 0.9123, F-Measure con 0.9122 y ROC Area con 0.9791. Los peores resultados son de los algoritmos LWL en Precision con 0.3546, en F-Measure con 0.3779 NaiveBayes Simple en Recall con 0.4856 y SMO en ROC Area con 0.5918.

En la Tabla 4.17 se muestran los resultados de los algoritmos aplicando sobre-muestreo para clasificar 7 clases con SMOTE al 200 %, los algoritmos que tuvieron buenos resultados son el RandomForest en Precision con 0.38 y en ROC Area con

Tabla 4.16: **5 Clases: Resample 200 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.6308	0.6332	0.6274	0.7807
Naive Bayes	0.5115	0.4875	0.4892	0.67
NaiveBayes Simple	0.5107	0.4856	0.4875	0.6702
RBFNetwork	0.5991	0.602	0.5902	0.7365
SMO	0.4142	0.5196	0.4263	0.5918
LWL	0.3546	0.4895	0.3779	0.6676
J48	0.8699	0.8692	0.8694	0.9225
J48graft	0.8666	0.8662	0.866	0.9204
Random Forest	**0.9132**	**0.9123**	**0.9122**	**0.9791**

0.6169, BayesNet y SMO en Recall con 0.43, RBFNetwork con 0.388. Los peores resultados son para los algoritmos SMO y BayesNet en Precision con 0.185 y en F-Measure con 0.259, J48 en Recall con 0.3478 y BayesNet con 0.5163.

Tabla 4.17: **7 Clases: SMOTE 200 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.185	**0.43**	0.259	0.5163
Naive Bayes	0.3525	0.3829	0.3544	0.604
NaiveBayes Simple	0.353	0.3837	0.3539	0.6055
RBFNetwork	0.3769	0.4254	**0.388**	0.6118
SMO	0.185	0.43	0.259	0.5306
LWL	0.1965	0.425	0.2598	0.587
J48	0.3424	0.3478	0.3429	0.5589
J48graft	0.3413	0.3508	0.3434	0.5573
Random Forest	**0.38**	0.4093	0.386	**0.6169**

En la Tabla 4.18 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 7 clases con Resample al 200 %, el algoritmo que tuvo buenos resultados es el RandomForest en Precision con 0.9166, Recall con 0.9148, F-Measure con 0.9145, ROC Area con 0.9775. Los peores resultados obtenidos son del algoritmo SMO en Precision con 0.195, Recall con 0.4415, F-Measure con 0.2707 y ROC Area con 0.5336.

Tabla 4.18: **7 Clases: Resample 200 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.5571	0.5592	0.5386	0.7844
Naive Bayes	0.482	0.4854	0.465	0.6845
NaiveBayes Simple	0.4804	0.4834	0.4614	0.6839
RBFNetwork	0.559	0.57	0.5433	0.7639
SMO	0.195	0.4415	0.2707	0.5336
LWL	0.2711	0.4422	0.2723	0.6711
J48	0.8272	0.8247	0.8248	0.9251
J48graft	0.8285	0.8263	0.8262	0.9256
Random Forest	**0.9166**	**0.9148**	**0.9145**	**0.9775**

En la Tabla 4.19 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 3 clases con SMOTE al 500 %, el algoritmo que tuvo buenos resultados es el RandomForest en Precision con 0.8684, Recall con 0.8706, F-Measure con 0.8559, ROC Area con 0.8903. Los peores resultados obtenidos son de los algoritmos SMO en Precision con 0.593, en F-Measure con 0.67 y ROC Area con 0.5002 y NaiveBayes en Recall con 0.7616.

Tabla 4.19: **3 Clases: SMOTE 500 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.7813	0.8028	0.7858	0.8125
Naive Bayes	0.8001	0.7616	0.7719	0.8005
NaiveBayes Simple	0.796	0.7553	0.7657	0.7975
RBFNetwork	0.8076	0.8191	0.809	0.8375
SMO	0.593	0.77	0.67	0.5002
LWL	0.6566	0.7703	0.6725	0.6915
J48	0.789	0.7937	0.7908	0.7205
J48graft	0.789	0.7937	0.7908	0.7205
Random Forest	**0.8684**	**0.8706**	**0.8559**	**0.8903**

En la Tabla 4.20 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 3 clases con Resample al 500 %, el algoritmo que tuvo buenos resultados es el RandomForest en Precision con 0.9997, Recall con 0.9997, F-Measure

con 0.9997, ROC Area con 1. Los peores resultados obtenidos son de los algoritmos SMO en Precision con 0.818, en ROC Area con 0.5148, NaiveBayes Simple en Recall con 0.8057, en F-Measure con 0.8371.

Tabla 4.20: **3 Clases: Resample500 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.9885	0.9882	0.9881	0.9943
Naive Bayes	0.8955	0.806	0.8373	0.8441
NaiveBayes Simple	0.8953	0.8057	0.8371	0.8436
RBFNetwork	0.9186	0.9271	0.9189	0.8768
SMO	0.818	0.905	0.859	0.5148
LWL	0.8886	0.9164	0.8828	0.8605
J48	0.9945	0.9944	0.9944	0.9966
J48graft	0.9953	0.9952	0.9952	0.997
Random Forest	**0.9997**	**0.9997**	**0.9997**	**1**

En la Tabla 4.21 se muestran los resultados de los algoritmos aplicando sobre-muestreo para clasificar 5 clases con SMOTE al 500 %, el algoritmo que tuvo buenos resultados es el RandomForest en Precision con 0.5236, Recall con 0.5208, F-Measure con 0.5181, ROC Area con 0.7157. Los peores resultados obtenidos son de los algoritmos LWL en Precision con 0.3321, en Recall con 0.4165, SMO en F-Measure con 0.2892 y en ROC Area con 0.5608.

Tabla 4.21: **5 Clases: SMOTE 500 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.3927	0.4315	0.3886	0.6415
Naive Bayes	0.4565	0.457	0.4449	0.6686
NaiveBayes Simple	0.4576	0.4563	0.4438	0.6671
RBFNetwork	0.4808	0.4949	0.4788	0.6738
SMO	0.3597	0.4178	0.2892	0.5608
LWL	0.3321	0.4165	0.3275	0.6189
J48	0.4188	0.4354	0.4233	0.6096
J48graft	0.4197	0.4382	0.4244	0.6123
Random Forest	**0.5236**	**0.5208**	**0.5181**	**0.7157**

En la Tabla 4.22 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 5 clases con Resample al 500 %, el algoritmo que tuvo buenos resultados es el RandomForest en Precision con 0.9926, Recall con 0.9925, F-Measure con 0.9925, ROC Area con 0.9992. Los peores resultados obtenidos son de los algoritmos LWL en Precision con 0.3733, en Recall con 0.4741, en F-Measure con 0.3077 y SMO en ROC Area con 0.612.

Tabla 4.22: **5 Clases:Resample 500 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.9513	0.9507	0.9506	0.9913
Naive Bayes	0.5504	0.5305	0.5276	0.6997
NaiveBayes Simple	0.5503	0.5289	0.5264	0.6987
RBFNetwork	0.6043	0.6098	0.597	0.7472
SMO	0.4372	0.529	0.4698	0.612
LWL	0.3733	0.4741	0.3077	0.6855
J48	0.9758	0.9757	0.9757	0.9937
J48graft	0.9759	0.9758	0.9758	0.9937
Random Forest	**0.9926**	**0.9925**	**0.9925**	**0.9992**

En la Tabla 4.23 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 7 clases con SMOTE al 500 %, los algoritmo que tuvieron buenos resultados son RBFNetwork en Precision con 0.4231, en Recall con 0.4632, en F-Measure con 0.4335 y Random Forest en ROC Area con 0.6727. Los peores resultados son de los algoritmos SMO en Precision con 0.1635, J48 con 0.3642, BayesNet en F-Measure con 0.2283 y en ROC Area con 0.5272.

En la Tabla 4.24 se muestran los resultados de los algoritmos aplicando sobremuestreo para clasificar 7 clases con Resample al 500 %,los algoritmo que obtuvieron buenos resultados son RandomForest en Precision con 0.9941, Recall con 0.9939, F-Measure con 0.9939 y ROC Area con 0.9995. Los peores resultados son de los algoritmos LWL en Precision con 0.197, en Recall con 0.444, en F-Measure con 0.273, SMO

Tabla 4.23: **7 Clases: SMOTE 500 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.167	0.3948	0.2283	0.5272
Naive Bayes	0.3503	0.3831	0.355	0.6322
NaiveBayes Simple	0.352	0.3831	0.3547	0.6321
RBFNetwork	**0.4231**	**0.4632**	**0.4335**	0.6707
SMO	0.1635	0.3966	0.2287	0.5302
LWL	0.2615	0.4017	0.254	0.604
J48	0.3598	0.3642	0.3611	0.5824
J48graft	0.3623	0.3693	0.3652	0.5859
Random Forest	0.4198	0.4548	0.4326	**0.6727**

en ROC Area con 0.575.

Tabla 4.24: **7 Clases: Resample 500 %**

Algoritmo	Precision	Recall	F-Measure	ROC Area
BayesNet	0.972	0.9717	0.9716	0.98
Naive Bayes	0.4804	0.4875	0.4656	0.7043
NaiveBayes Simple	0.477	0.483	0.4617	0.7031
RBFNetwork	0.5828	0.5924	0.5652	0.7887
SMO	0.3853	0.4705	0.338	0.575
LWL	0.197	0.444	0.273	0.7347
J48	0.9729	0.9726	0.9724	0.9951
J48graft	0.9729	0.9727	0.9725	0.995
Random Forest	**0.9941**	**0.9939**	**0.9939**	**0.9995**

A continuación se muestran las mejores matrices de Confusión para 3, 5 y 7 clases al 100 %, 200 % y 500 %

En la Tabla 4.25 se muestran los resultados de la matriz de confusión de tres clases con el algoritmo SMOTE al 100 %.

En la Tabla 4.26 se muestran los resultados de la matriz de confusión de cinco clases con el algoritmo Resample al 100 %.

En la Tabla 4.27 se muestran los resultados de la matriz de confusión de siete clases con el algoritmo Resample al 100 %.

Tabla 4.25: **===Matriz de confusión===**

a	b	c	classified as
4	14	0	a=E
13	258	10	b=S
0	13	9	c=Irr

Tabla 4.26: **===Matriz de confusión===**

a	b	c	d	e	classified as
18	0	2	1	0	a=E
0	16	3	3	0	b=So
2	1	88	16	0	c=Sa+Sb
0	1	25	99	1	d=Sc+Sd
0	0	3	2	12	e=Irr

Tabla 4.27: **===Matriz de confusión===**

a	b	c	d	e	f	g	classified as
19	0	0	2	0	0	0	a=E
0	14	1	0	6	1	0	b=So
4	1	7	3	6	0	0	c=Sa
0	0	4	65	16	0	1	d=Sb
0	2	3	13	100	0	0	e=Sc
0	0	0	1	3	4	0	f=Sd
0	1	0	2	1	0	13	g=Irr

En la Tabla 4.28 se muestran los resultados de la matriz de confusión de tres clases con el algoritmo Resample al 200 %.

Tabla 4.28: **===Matriz de confusión===**

a	b	c	classified as
48	1	0	a=E
1	548	0	b=S
0	3	19	c=Irr

En la Tabla 4.29 se muestran los resultados de la matriz de confusión de cinco clases con el algoritmo Resample al 200 %.

Tabla 4.29: **===Matriz de confusión===**

a	b	c	d	e	classified as
38	0	1	0	0	a=E
0	33	3	1	0	b=So
0	1	187	21	0	c=Sa+Sb
1	0	16	259	0	d=Sc+Sd
0	0	0	3	22	e=Irr

En la Tabla 4.30 se muestran los resultados de la matriz de confusión de siete clases con el algoritmo Resample al 200 %.

Tabla 4.30: **===Matriz de confusión===**

a	b	c	d	e	f	g	classified as
38	1	0	0	0	0	0	a=E
1	34	0	1	1	0	0	b=So
2	0	44	1	5	0	0	c=Sa
0	0	2	145	10	0	0	d=Sb
0	1	0	11	245	2	0	e=Sc
0	0	0	1	2	14	0	f=Sd
0	0	0	1	2	0	22	g=Irr

En la Tabla 4.31 se muestran los resultados de la matriz de confusión de tres clases con el algoritmo Resample al 500 %.

Tabla 4.31: **===Matriz de confusión===**

a	b	c	classified as
91	0	0	a=E
0	1402	0	b=S
0	0	57	c=Irr

En la Tabla 4.32 se muestran los resultados de la matriz de confusión de cinco clases con el algoritmo Resample al 500 %.

Tabla 4.32: **===Matriz de confusión===**

a	b	c	d	e	classified as
94	0	0	0	0	a=E
0	84	3	0	0	b=So
1	1	538	1	0	c=Sa+Sb
0	0	0	694	0	d=Sc+Sd
0	0	0	0	49	e=Irr

En la Tabla 4.33 se muestran los resultados de la matriz de confusión de siete clases con el algoritmo Resample al 500 %.

Tabla 4.33: **===Matriz de confusión===**

a	b	c	d	e	f	g	classified as
94	0	0	0	0	0	0	a=E
0	84	1	2	0	0	0	b=So
0	0	127	1	0	0	0	c=Sa
0	0	0	410	3	0	0	d=Sb
0	1	0	1	648	0	0	e=Sc
0	0	0	0	0	44	0	f=Sd
0	0	0	0	0	0	49	g=Irr

Capítulo 5

Conclusiones

Este trabajo ha presentado un estudio experimental para clasificar imagénes de galaxias, considerando el problema de datos no balanceados.

De acuerdo a los resultados se tienen las siguientes conclusiones.

- Para la clasificación de tres tipos de galaxias el algoritmo con la mejor Precision fue RandomForest con 0.9997, Recall con 0.9997, F-Measure con 0.9997 y ROC-Area con 1.

- Para la clasificación de cinco tipos de galaxias el algoritmo con la mejor Precision fue RandomForest con 0.9926, Recall con 0.9925, F-Measure con 0.9925 y ROC-Area con 0.9992.

- Para la clasificación de siete tipos de galaxias el algoritmo con la mejor Precision fue RandomForest con 0.9941, Recall con 0.9939, F-Measure con 0.9939 y ROC-Area con 0.9995.

Por último, se puede destacar que tanto el objetivo general, como los particulares se cumplieron de manera satisfactoria.

Como trabajo futuro se experimentará con otros algoritmos para manejar datos no balanceados, como BordelineSMOTE, SVMSMOTE, entre otros. Asi también aplicarlo

entre otros dominios como bioinformática y visión por computadora.

Bibliografía

[1] Basilio Sierra Araujo. *Aprendizaje Automático: Conceptos Básicos Y Avanzados.* Pearson Educación, 2006.

[2] Bazell and Aha. Ensembles of classifiers for morphological galaxy classification. *The Astrophysical Journal*, pages 219–223, 2001.

[3] Eyer C. and Blake. Automated classification of variable stars for all-sky automated survey 12 data. *Monthly Notices of the Royal Astronomical Society*, 2005.

[4] Ling C. and Li. Data mining for direct marketing problems and solutions. *Fourth International Conference on Knowledge Discovery and Data Mining*, 1998.

[5] Mladenic D. and Grobelnik. *M.Feature Selection for Unbalanced Class Distribution and Naive Bayes in Proceedings of the 16th International Conference on Machine Learnig.* PhD thesis, J Stefan Institute, Jornova 39,1000,Ljubljana Slovenia, 1999.

[6] Salomón D. A guide to data compression methods. *New York, Springer-Verlag*, 2002.

[7] Thomas G. and Dietterich. *Machine Learning.* PhD thesis, Oregon State University, Corvallis, OR 97331, 1994.

[8] Francisco Herrera. Curso: M6: Nuevas tendencias en minería de datos. *Dpto. Ciencias de la Computación e I.A. Universidad de Granada.España*, 2008.

[9] Fowler J. and Yagel. Lossless compression of volume data. *Symposium on Volume Visualization*, pages 45–50, 1994.

[10] Han J. and Kamber M. Data mining: Concepts and techniques (2nd. ed.). *San Francisco: Morgan Kaufmann.*, 2006.

[11] Hilera J. and Martínez V. *Redes neuronales artificiales: fundamentos, modelos y aplicaciones*. RA-MA, 1995.

[12] Moreno J., Rodríguez., MA Sicilia., Riquelme JC., and Ruiz R. Smote-i: mejora del algoritmo smote para balanceo de clases minoritarias. *Departamento de ciencias de la Computación. Escuela Técnica Superior de Ingeniería Informática. Escuela Politécnica Superior*, 2009.

[13] Freeman JA. and Skapura DM. *Redes Neuronales, Algoritmos, aplicaciones y técnicas de programación*. México,Addison-Wesley, 1993.

[14] Woods K., Doss C., Bowyer K., Solka J., Priebe C., and Kegelmeyer P. Comparative evaluation of pattern recognition techniques for detection of microcalcifications in mammography. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(6):1417–1436, 1993.

[15] Breiman L. Random forests. machine learning. 2001.

[16] Kubat M. and Matwin. S. addressing the curse of imbalanced training sets. *One Sided Selection, in Proceedings of the Fourteenth International Conference on Machine Learning*, pages 79–186, 1997.

[17] Villanueva M. Las redes neuronales artificiales y su importancia como herramienta en la toma de decisiones. *Trabajo de Investigación - Universidad Nacional Mayor de San Marcos. Facultad de Ciencias Matemáticas.*, 2002.

[18] Chawla N., Bowyer K., Hall L., and Kegelmeyer. P.smote: Synthectic minority over-sampling technique. *Journal of Artificial Intelligence*, pages 321–357, 2002.

[19] Japkowicks N. The class imbalance problem: Significance and strategies in proceedings of the 2000. *International Conference on Artificial Intelligence*, pages 111–117, 1997.

[20] Nilsson N. *Inteligencia Artificial,una nueva síntesis.* McGraw-Hill, 2001.

[21] Fuentes O. and de la Calleja. Machine learning for morphological galaxy classification. Technical report, Computer Science Department, National Institute of Astrophysics, Optics and Electronics, Tonantzintla 72840, Puebla, México, October 2003.

[22] Fuentes O. and de la Calleja. Machine learning and image analysis for morphological galaxy classification. *Monthly Notices of the Royal Astronomical Society*, 349:87–93, 2004.

[23] Fuentes O. and de la Calleja. Automated star galaxy discrimination in multispectral wide-field images. *Proceedings of the 2nd International Conference on Computer Vision Theory and Applications*, pages 155–160, 2007.

[24] Jonathon Shlens. *A Tutorial on Principal Component Analysis.* Institute for Nonlinear Science, University of California, San Diego, La Jolla, CA 92093-0402, 2 edition, December 10 2005.

[25] Smith-2002. *A Tutorial on Principal Component Analysis.* John Wiley and Sons Inc, 2005.

[26] Fawcett T. and Provost F. *Combining Data Mining and Machine Learning for Effective User Profile*. PhD thesis, NYNEXS, 400 Westchester Avenue White Plains, New York 10604, 1996.

[27] Mitchell T. *Machine Learning*. McGraw-Hill, 1997.

[28] Nitesh V., Chawla, Lawrence O. Kevin W. Bowyer, Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, page 321357, 2002.

[29] Han Wang. and Mao. *Bordeline-SMOTE: A new over-sampling method in imbalanced data sets learning*. Springer Berlin Heidelberg, Septiembre 2005.

[30] Zhen Z., Wu X., and Srihari. *Feature Selection for Text Categorization on Imbalanced Data*. PhD thesis, University at Buffalo, Amherst, NY 14260, 2004.