BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

LICENCIATURA EN CIENCIAS
DE LA COMPUTACIÓN

"CLASIFICACIÓN DE PORTADAS DE LIBROS USANDO APRENDIZAJE AUTOMÁTICO"

PROYECTO DE TESINA

QUE PARA OBTENER EL GRADO DE LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:
RAYMUNDO GABRIEL MARTÍNEZ CRUZ

ASESOR 1:
DR. IVÁN OLMOS PINEDA

ASESOR 2:
DR. JORGE DE LA CALLEJA MORA

PUEBLA, PUE. MARZO, 2013

ÍNDICE

1 PLANTEAMIENTO DEL PROBLEMA DE INVESTIGACIÓN

1.1 Introducción

Las Ciencias Computacionales (CC) se dedican principalmente al estudio del almacenamiento, transformación y transferencia de información en las computadoras. Estas funciones son realizadas desde perspectivas teóricas y prácticas; en la parte teórica por medio del diseño, eficiencia y aplicación de algoritmos (secuencia de acciones para resolver un problema o ejecutar una tarea) y en la parte práctica, involucra la implementación de dichos algoritmos en hardware y software de computadoras.

Las CC tienen muchas especialidades o campos de estudio. Algunos de ellos son: estructuras de datos, metodología y lenguajes de programación, ingeniería del software, inteligencia artificial, teoría de autómatas, sistemas de base de datos, computación paralela, computación gráfica y sistemas operativos entre otros.

El campo de la inteligencia artificial intenta, no solo comprender, sino que también se esfuerza en construir entidades inteligentes. La inteligencia artificial abarca, en la actualidad, una gran cantidad de problemas que van desde áreas de propósito general, como el aprendizaje y la percepción, a otras más específicas como el ajedrez, la demostración de teoremas matemáticos, la escritura de poesía y el diagnóstico de enfermedades (Stuart J. Russell, 2004).

El campo del Machine Learning (ML) o Aprendizaje Automático (AA) trata de construir programas que automáticamente mejoren con la experiencia (Mitchell, 1997). El AA se apoya en conceptos y resultados de muchas áreas como lo es la estadística, filosofía, teoría de la información, biología, ciencia cognitiva, complejidad computacional y teoría de control (Mitchell, 1997).

Algunas aplicaciones exitosas que han involucrado el aprendizaje automático son: el reconocimiento de voz (Shin J., 2010), el reconocimiento de rostros (Jones, 2004), algunas detecciones de enfermedades (Peña L., 2006), detección de fraudes con tarjetas de crédito (Stolfo, 1998), clasificación de objetos astronómicos (Ball N., 2006), reconocimientos de patrones (Drummond, 2006), reconocimientos de secuencias del

ADN (H., 2003), análisis del mercado de valores (Riezler, 2002), aplicaciones de videojuegos (Stanley, 2005), aplicaciones de robótica (Stone, 2004), entre otras más.

En el presente trabajo se propone usar algoritmos de aprendizaje automático para clasificar portadas de libros. Los algoritmos usados fueron support vector machine, neuronal network y bagging. Así también se emplea la técnica de análisis de componentes principales para caracterizar las imágenes. Los resultados obtenidos muestran que en todos los casos Bagging con random forest fué el que mejor clasifico con 57.52% de instancias correctamente clasificadas y una precisión del 58% en el mejor de los casos con los eigenverctores correspondientes a componentes principales del 90% de dos conjuntos de imágenes de 100x100 pixeles.

1.2 Objetivo General

- Diseño de un modelo para la clasificación de imágenes de portadas de libros a partir de algoritmos de aprendizaje automático.

1.3 Objetivos Específicos

- Caracterizar las imágenes usando la técnica de análisis de componentes principales.
- Evaluar tres algoritmos de aprendizaje automático para la clasificación de las imágenes.
- Diseñar un método que permita utilizar ensambles de clasificadores que alimente algoritmos de categorización de imágenes.

1.4 Justificación

Dado que trabajar con imágenes requiere cierta cantidad de recursos y tiempo de procesamiento se abordó la clasificación de componentes principales asociadas a las imágenes de portadas de libros.

Se realizó el presente trabajo con el fin de observar si las portadas de libros se pueden caracterizar por áreas de estudio y clasificarlas de manera automática.

1.5 Contribuciones Esperadas

- La caracterización de las imágenes de manera automática.
- Un análisis comparativo de los algoritmos usados.

1.6 Cronograma de actividades

Las actividades que se llevarán a cabo para el presente proyecto de investigación se muestran en la tabla 1.

DURACIÓN DEL PROYECTO																
Actividad	Semanas															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Revisión del estado del arte	*	*	*	*												
Revisión del algoritmo de componentes principales			*	*												
Revisión de los algoritmos de clasificación					*	*	*									
Revisión de WEKA								*	*	*						
Obtención de imágenes										*						
Cálculo de los componentes principales de cada subconjunto de imágenes											*					
Clasificación en WEKA											*	*				
Análisis de los parámetros obtenidos en WEKA													*	*	*	
Elaboración del documento	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

Tabla 1*. Cronograma de actividades a lo largo de un cuatrimestre dividido en semanas.*

1.7 Recursos de hardware

- Equipo de cómputo con procesador a 1.5Ghz, Memoria RAM de 2Gb y disco duro de 80Gb.

1.8 Recursos de Software

- MatLab R2010a
- Weka 3.7.4

1.9 Alcances y limitaciones

- Solo se clasificarán las imágenes de portadas de libros.
- Únicamente se probará con una técnica de caracterización de imágenes.
- El conjunto de imágenes es relativamente pequeño.
- Solo se usarán algoritmos de aprendizaje supervisado.

2 MARCO TEÓRICO

2.1 Antecedentes históricos

Desde la aparición de la primera computadora, el hombre se ha enfocado en tratar hacer que las la computadoras aprendan.

En 1940, Turing y un equipo de científicos construyeron el primer computador operacional de carácter electromecánico, llamado Healt Robinson en honor a un caricaturista muy famoso, con el propósito de descifrar mensajes alemanes. En 1943, desarrollaron Colossus que era una máquina de propósito general basada en válvulas de vacio (Metropolis, 1985).

El primer computador operacional programable fue el Z-3, inventado por Konrad Zuse en Alemania, en 1941. Zuse también invento los primeros coma flotante y el primer lenguaje de alto nivel, Plankalkül (Stuart J. Russell, 2004).

El primer computador electrónico, el ABC, fue creado entre 1940 y 1942 por John Atanasoff y Clifford Berry de la universidad de Iowa. Posteriormente apareció ENIAC, creado por John Muchly y John Eckert en la universidad de Pensilvania. A esta última se le conoce como el precursor de los computadores modernos.

En 1943 Warren McCulloch y Walter Pitts presentaron su modelo de neuronas artificiales, el cual se considera el primer trabajo del campo de inteligencia artificial, aun cuando todavía no existía el término.

En 1950, Turing sentó un precedente para lo que hoy conocemos como inteligencia artificial. El realizó una prueba que consistía en un juego en el cual una computadora debería de contestar preguntas lanzadas por una persona. La computadora, por un lado y por otro un concursante contestaban preguntas con el fin de que el interrogador adivinara quien era la máquina. A tal experimento lo llama el juego de la imitación. El trata de dar una respuesta a una prueba bastante compleja como lo es: ¿Puede pensar una máquina? Turing, señaló que una máquina podría fracasar y aún ser inteligente. Aun así creía que las máquinas podrían superar la prueba a finales del siglo XX (Turing, 1950).

En 1951 William Shockley inventa el transistor de unión. El invento hizo posible una nueva generación de computadoras mucho más rápidas y pequeñas.

En 1956 se dio el término "inteligencia artificial" en Dartmouth durante una conferencia convocada por McCarthy, a la cual asistieron, entre otros, Minsky, Newell y Simon. En esta conferencia se hicieron previsiones triunfalistas a diez años que jamás se cumplieron, lo que provocó el abandono casi total de las investigaciones durante quince años.

En 1958, surge el lenguaje de Programación Lisp desarrollado por McCarthy.

En 1980 la historia se repitió con el desafío japonés de la quinta generación, que dio lugar al auge de los sistemas expertos pero que no alcanzó muchos de sus objetivos, por lo que este campo sufrió una nueva interrupción en los años noventa.

En 1987 Martin Fischles y Oscar Firschein (Oscar, 1987) describieron los atributos de un agente inteligente. Dichos atributos del agente inteligente son:

- Tiene actitudes mentales tales como creencias e intenciones.
- Tiene la capacidad de obtener conocimiento, es decir, aprender.
- Puede resolver problemas, incluso particionando problemas complejos en otros más simples.
- Entiende. Posee la capacidad de crearle sentido, si es posible, a ideas ambiguas o contradictorias.
- Planifica, predice consecuencias, evalúa alternativas (como en los juegos de ajedrez)
- Conoce los límites de sus propias habilidades y conocimientos.
- Puede distinguir a pesar de la similitud de las situaciones.
- Puede ser original, creando incluso nuevos conceptos o ideas, y hasta utilizando analogías.
- Puede generalizar.
- Puede percibir y modelar el mundo exterior.
- Puede entender y utilizar el lenguaje y sus símbolos.

Podemos entonces decir que la IA posee características humanas tales como el aprendizaje, la adaptación, el razonamiento, la autocorrección, el mejoramiento implícito, y la percepción modular del mundo. Así, podemos hablar ya no sólo de un objetivo, sino de muchos, dependiendo del punto de vista o utilidad que pueda encontrarse a la IA.

A finales del siglo XX, Mitchell (Mitchell, 1997) menciona que todavía no se conoce como hacer que las computadoras aprendan tan bien como los humanos, sin embargo los algoritmos ya inventados son efectivos en cierto tipos de tareas.

Algunos de estos logros se aprecian en los reconocimientos de voz, prevención de enfermedades, vehículos autónomos, jugadores de ajedrez, etc.

Durante años la investigación en aprendizaje automático se ha realizado con distinto grado de intensidad, utilizando diferentes técnicas y haciendo énfasis en distintos aspectos y objetivos.

2.2 Aprendizaje Automático

El aprendizaje automático es un campo de la Inteligencia Artificial cuyo objetivo es desarrollar programas que permitan a las computadoras aprender. Por lo que, se trata de crear programas capaces de generalizar comportamientos a partir de información no estructurada suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento (Stuart J. Russell, 2004).

El aprendizaje automático estudia cómo construir programas que mejoren su desempeño a través de la experiencia, de tal manera que: "Un programa de computadora se dice que aprende de la experiencia E con respecto a una clase de tarea T y a una medida de desempeño P, es decir, si el desempeño en la tarea T se evalúa con una medida de desempeño P entonces mejora con la experiencia E" (Mitchell, 1997).

Para ello es importante indicarle de ¿Dónde debe de aprender?, ¿Cuál es el objetivo a cumplir?, y ¿Qué tipo de resultados esperamos obtener?.

Para que un programa pueda tomar una buena decisión –acertada o no- sobre la clasificación que va a asignar, es necesario indicarle las características que se han observado en el caso que se esté estudiando (Araujo, 2006).

2.3 Paradigmas del Aprendizaje Automático

Para automatizar el proceso de clasificación, existen diferentes aproximaciones:

1) Un experto humano capaz de diseñar un sistema clasificador.
2) El sistema clasificador aprenda de manera automática.
3) Una combinación de 1) y 2).

Al sistema clasificador que aprende de alguna manera por si mismo, se le conoce como aprendizaje automático, que consiste en aprender el modelo clasificatorio a utilizar en base a la información que se halla contenida en una base de datos histórica en la que se guardan valores de la variables predictoras de casos tratados anteriormente, junto con la clase real a la que pertenecen los mismos. A este histórico se le conoce como base de datos de entrenamiento.

2.3.1 Clasificación

2.3.1.1 Reconocimiento de formas

- Aproximaciones paramétricas: Se tienen un conocimiento a priori acerca de la forma funcional de las distribuciones de probabilidad de cada clase sobre el espacio de representación.
- Aproximaciones no paramétricas: No supone ninguna forma de las distribuciones de probabilidad sobre el espacio de representación, de modo que el único conocimiento a priori será el correspondiente a la información inducida a partir del conjunto de muestras (Eli, 2008).

2.3.1.2 Reconocimiento de patrones

- Clasificación supervisada: Parte de un conjunto de objetos descritos por un vector de características y la clase a la que pertenece cada uno de ellos; a este conjunto de objetos de los que conocemos la clase a la que pertenecen se los denomina “conjunto de entrenamiento” o “conjunto de aprendizaje”.

- Clasificación no supervisada: Enfoca la clasificación como el descubrimiento de clases del problema. Los objetos únicamente vienen descritos por un vector de características (Eli, 2008).

2.3.1.3 Situaciones dentro del problema clasificatorio

- Clases que definen el problema son separables: Cuando todos los objetos con las mismas características pertenecen a la misma clase.
- Clases que definen el problema no son separables: Cuando dos o más objetos con las mismas características pertenecen a diferentes clases (Eli, 2008).

2.3.1.4. Evaluación de clasificadores

- Tasa de error: nos da una idea de porcentaje de objetos nuevos, de los cuales no sabemos su clase.
- Rapidez: con la que el clasificador construye el modelo o con la que clasifica objetos nuevos.
- Interpretabilidad del modelo: Cuan fácil resulta el entender el modelo construido.
- Simplicidad del modelo: Construcción de modelos eficientes, sin complejidades (Eli, 2008).

2.3.1.5 Construcción de modelos de clasificación

- Modelización hacia adelante (forward): Empieza desde el modelo más simple posible, aumentando paso a paso, la complejidad del modelo hasta el cumplimiento de algún criterio preestablecido.
- Modelización hacia atrás (backward): Empieza desde el modelo más complejo posible, disminuyendo, paso a paso, la complejidad del modelo hasta el cumplimiento de algún criterio preestablecido.
- Modelización paso a paso (stepwise): Empieza desde el modelo más simple o complejo posible; planteando en cada paso tanto el aumento como disminución de la complejidad del modelo (Eli, 2008).

2.3.2 Métodos de validación

La validación de un clasificador sirve para medir su capacidad de predicción sobre nuevas instancias que le lleguen en el fututo para que las clasifique. La validación se realiza habitualmente basándose en la tasa de error del clasificador como representante más habitual de la medida de éxito de éste (Araujo, 2006).

2.3.2.1 Método H (Holdout). Particiona el conjunto de casos en dos grupos:

- Entrenamiento: está conformado por las dos terceras partes y es usado para inducir un modelo clasificatorio.
- Prueba: es conformado por la última tercer parte y se le utiliza para estimar la tasa de error verdadera.

2.3.2.2 Método de remuestreo (random subsampling). Es una variante del método H, y se fundamenta en aplicar el método H múltiples veces (variando el criterio de selección del grupo de entrenamiento y el de prueba), y se calcula el error en base a la media de las tasas de error obtenidas.

2.3.2.3 Método de validación cruzada (cross-validation). Se basa en la partición de la muestra en K subconjuntos, aproximadamente del mismo tamaño, donde K – 1 subconjuntos constituyen el grupo de entrenamiento y el restante el grupo de prueba.

2.3.2.4 Método de Bootstraping: En un conjunto de casos de cardinalidad N, se escoge una muestra aleatoria con reemplazamiento del mismo tamaño como grupo de entrenamiento, dejando los casos no seleccionados como grupo de testeo. (El muestreo con reemplazamiento consiste en extraer elementos de una población de forma que, tras cada extracción, el elemento extraído se vuelve a introducir y puede volver a ser seleccionado.)

2.3.3 Búsqueda en el espacio de modelos

En el caso de la inducción (Imagen 1) de reglas de clasificación, los árboles de clasificación y las redes Bayesianas, la construcción del modelo clasificatorio se efectúa habitualmente de una manera voraz y progresiva. Es decir, empieza a partir de cierta estructura (sea lo más simple o lo más compleja posible – criterio de

construcción de modelos de clasificación) construyéndose el modelo paso a paso, tomando decisiones acerca del aumento o disminución de complejidad del modelo, hasta que se cumpla un criterio específico que la haga detenerse.

Por lo tanto, en vez de construir un modelo paso a paso, se trata de realizar una búsqueda inteligente en el espacio de posibles modelos, siendo cada individuo de nuestra búsqueda, un modelo clasificatorio que trate de resolver el problema que se está estudiando.

Una vez que se plantea el problema de la inducción del modelo en términos de una búsqueda, se deben definir:

- El punto de inicio de la búsqueda.
- La organización de la búsqueda.
- La función de la evaluación.
- El criterio para la búsqueda.

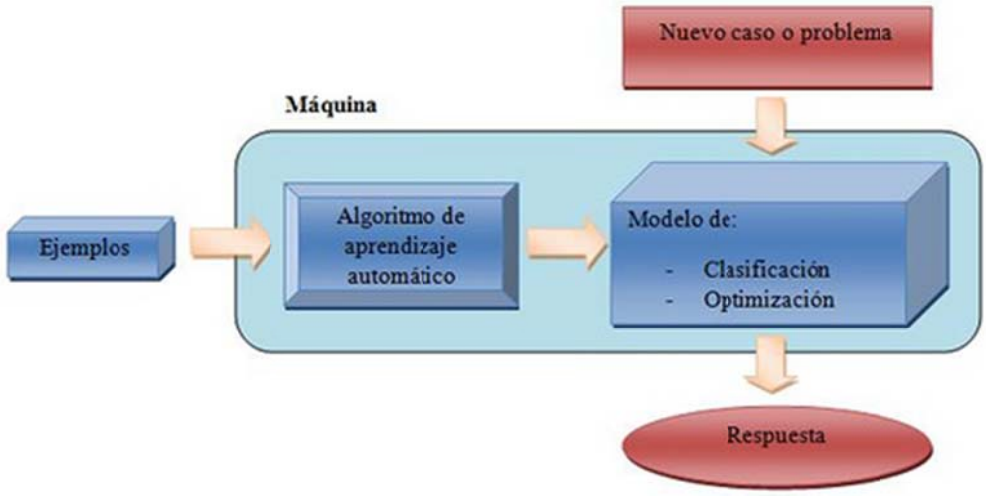


Imagen 1. *Proceso de inducción y deducción* (Eli, 2008)

2.4 Algoritmos de clasificación supervisada

Dentro de las técnicas de clasificación supervisada se encuentran:

- Algoritmos de clasificación por vecindad
- Árboles de clasificación
- Aprendizaje de reglas de decision
- Redes bayesianas
- Redes neuronales
- Modelos ocultos de Markov
- Métodos kernel y máquinas de vectores soporte

2.4.1 Árboles de Clasificación

Un árbol de decisión es un conjunto de condiciones o reglas organizadas en una estructura jerárquica, de tal manera que la decisión final se puede determinar siguiendo las condiciones que se cumplen desde la raíz hasta alguna de sus hojas.

Un árbol de decisión tiene unas entradas las cuales pueden ser un objeto o una situación descrita por medio de un conjunto de atributos y a partir de esto devuelve una respuesta la cual en últimas es una decisión que es tomada a partir de las entradas.

Los valores que pueden tomar las entradas y las salidas pueden ser valores discretos o continuos. Se utilizan más los valores discretos por simplicidad. Cuando se utilizan valores discretos en las funciones de una aplicación se denomina clasificación y cuando se utilizan los continuos se denomina regresión.

Los árboles de clasificación entran dentro de los métodos de clasificación supervisada, es decir, donde se tiene una variable dependiente o clase, y el objetivo del clasificador va a ser averiguar dicha clase para casos nuevos. La construcción del árbol de clasificación se realiza mediante un proceso de inducción, de ahí que también sean denominados como Top-Down-Induction-Decision-Trees (Araujo, 2006).

2.4.2 Redes Neuronales

Referidas habitualmente de forma más sencilla como redes de neuronas, las redes de neuronas artificiales (Imagen 2) son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales.

Consiste en simular las propiedades observadas en los sistemas neuronales biológicos a través de modelos matemáticos recreados mediante mecanismos artificiales (como un circuito integrado, un ordenador o un conjunto de válvulas). El objetivo es conseguir que las máquinas den respuestas similares a las que es capaz el cerebro que se caracterizan por su generalización y su robustez.

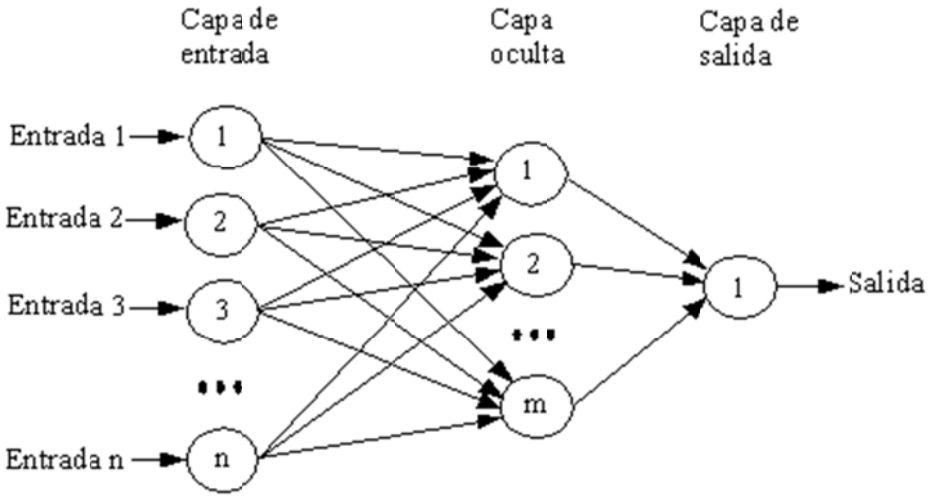


***Imagen 2**. Ejemplo de Red Neuronal Artificial de tipo Perceptrón simple con n neuronas de entrada, m neuronas en su capa oculta y una neurona de salida.*

Las Redes Neuronales pueden ser aplicadas a la construcción de generalizaciones que caractericen grandes grupos de datos ya que adquieren conocimiento integrador de información durante el proceso de ajuste de los pesos de las conexiones entre neuronas que las integran, tienen la ventaja de ser tolerantes al ruido y la capacidad de extender la generalización al momento de necesitar manipular datos nuevos (Knight, 1994).

2.4.3 Máquinas de vectores de soporte

Las Máquinas de Soporte Vectorial (*Support Vector Machines*, SVM) es un algoritmo de aprendizaje supervisado que trata de encontrar un hiperplano óptimo (frontera) que sea capaz de separar un conjunto de otro. Para ello se extraen las muestras más cercanas a la frontera, a las que se les conoce como vectores de soporte (Imagen 3).

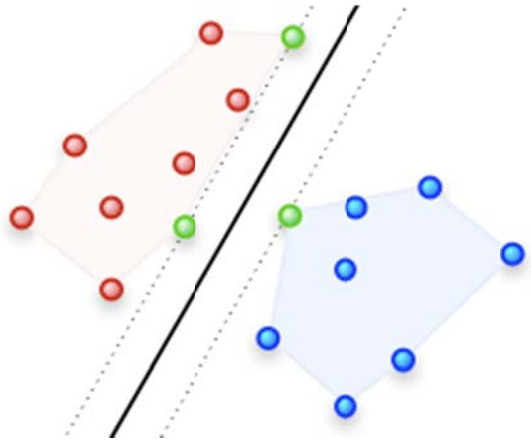


Imagen 3. *Los puntos rojos y azules pertenecen a clases diferentes y los vectores de soporte utilizados para trazar el plano se encuentran de color verde.*

El hiperplano óptimo es aquel que maximiza el margen o distancia entre la frontera y dichos vectores de soporte.

Las Máquinas de Vectores Soporte fueron diseñadas originalmente para resolver problemas de clasificación binaria. Para abordar el problema de la clasificación en k clases existen dos aproximaciones básicas:

- Uno-contra-todos (*one-versus-all*), donde se entrenan k SVM y cada uno separa una clase del resto.
- Uno-contra-uno (*one-against-one*), donde se han de entrenar $k(k-1)/2$ modelos y cada modelo discrimina entre un par de clases.

Es importante notar que como uno-contra-uno trabaja con menos muestras, tiene mayor libertad para encontrar una frontera que separe ambas clases. Respecto al

coste de entrenamiento, es preferible el uso de uno-contra-todos puesto que sólo ha de entrenar k SVM (Tomás, 2005) .

La complejidad de prueba de ambas estrategias es similar: uno-contra-todos necesita k evaluaciones y uno-contra-uno $k - 1$.

2.5 Bagging

El término Bagging procede de la contracción de las palabras bootstrap aggregating. El método consiste en agregar, mediante una votación simple, los resultados de varios clasificadores obtenidos de un mismo conjunto de entrenamiento mediante bootstrap (Araujo, 2006).

Leo Breiman (Breiman, 1996) describió esta técnica a principios de los noventa del siglo pasado y desde entonces se han utilizado ampliamente para intentar mejorar los resultados de clasificadores, en especial los árboles de decisión.

En el Bagging cada modelo individual se crea a partir de un conjunto con el mismo número de elementos que el inicial, pero obtenido mediante extracción aleatoria con reemplazo.

2.6 Análisis de componentes principales

El análisis de componentes principales (PCA, por sus siglas en inglés), consiste en encontrar transformaciones ortogonales de las variables originales para conseguir un nuevo conjunto de variables incorreladas, denominadas Componentes Principales, que se obtienen en orden decreciente de importancia.

Los componentes son combinaciones lineales de las variables originales y se espera que, solo unas pocas (las primeras) recojan la mayor parte de la variabilidad de los datos, obteniéndose una reducción de la dimensión en los mismos. Luego el propósito fundamental de la técnica consiste en la reducción de la dimensión de los datos con el fin de simplificar el problema en estudio.

El PCA puede entenderse también como la búsqueda del subespacio de mejor ajuste.

2.6.1 Obtención de componentes principales

La obtención de componentes principales puede realizarse por varios métodos alternativos:

1. Buscando aquella combinación lineal de las variables que maximiza la variabilidad. (Hottelling)
2. Buscando el subespacio de mejor ajuste por el método de los mínimos cuadrados. Esto es, minimizando la suma de cuadrados de las distancias de cada punto al subespacio.(Pearson)
3. Minimizando la discrepancia entre las distancias euclideas y los puntos calculados en el espacio original y en el subespacio de baja dimensión. (Coordenadas principales, Gower)
4. Mediante regresiones alternadas. (Métodos Biplot)

Probablemente el método más ampliamente usado y mejor conocido de los métodos multivariados es en inventado por Pearson (1901) y Hottelling (1933).

En la Imagen 4 se puede observar de manera gráfica la idea para la obtención de componentes principales.

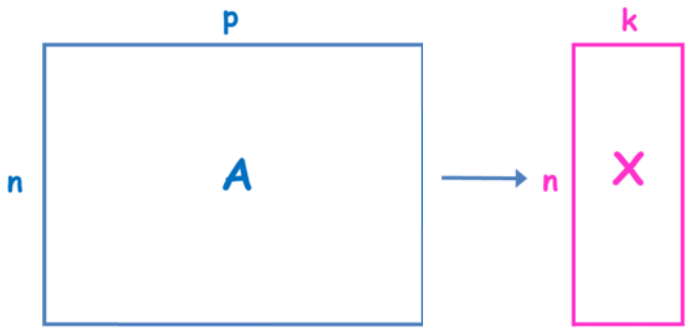


***Imagen 4**. Reducción de un conjunto de n objetos con p datos a uno de k eigenvectores.*

La idea básica en PCA es encontrar los componentes (eigenvectores) de la matriz de covarianza de un conjunto de objetos, tales que estos puedan representar la máxima varianza de *n* componentes linealmente transformados. Por tanto, ese conjunto de eigenvectores pueden ser un conjunto que caracterízan la variación del objeto.

La asociación de eigenvalores permite ordenar a los eigenvectores de acuerdo a su utilidad en la caracterización de la variación entre los objetos (Imagen 4).

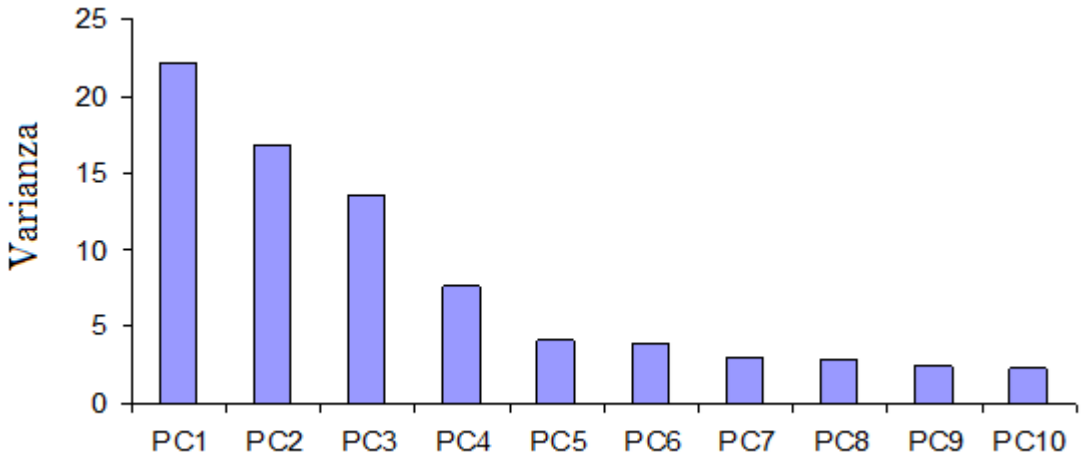


Imagen 5. *El primer componente principal acumula la mayor cantidad de varianza.*

2.6.2 PCA aplicado a imágenes

Matthew Turk y Alex Pentland del Grupo de Visión y Modelado en el laboratorio de Medios en el MIT[1], aplicaron el PCA para el reconocimiento de rostros en tiempo real (Pentland, 1991). Ellos se enfocaron en el problema de reconocimiento de rostros en dos dimensiones ya que es complejo para trabajar la geometría de las cabezas en tres dimensiones.

A manera de ejemplo (Imagen 6), dado un conjunto de *n* imágenes de *nxn* pixeles, el funcionamiento del algoritmo consiste en:

[1] MIT. Massachusetts Institute of Technology

1. Dada una imagen de *nxn* pixeles, se toma la segunda fila y se colocan los *n* elementos como columnas de la primera fila y así sucesivamente hasta obtener una imagen de $1*n^2$.
2. Se realiza el procedimiento del punto 1 para las *n-1* imágenes restantes.
3. Cada imagen de dimensión $1*n^2$ forman las filas de la nueva imagen. De tal manera que el conjunto de imágenes lo podemos representar con una matriz de $numeroDeImagenes*n^2$.
4. Luego, se ejecuta el algoritmo que obtiene los componentes principales dando como resultado un conjunto de eigenvectores asociados a dicha imagen.
5. Una vez obtenido el conjunto de eigenvectores se toman como entrada para clasificar en weka.

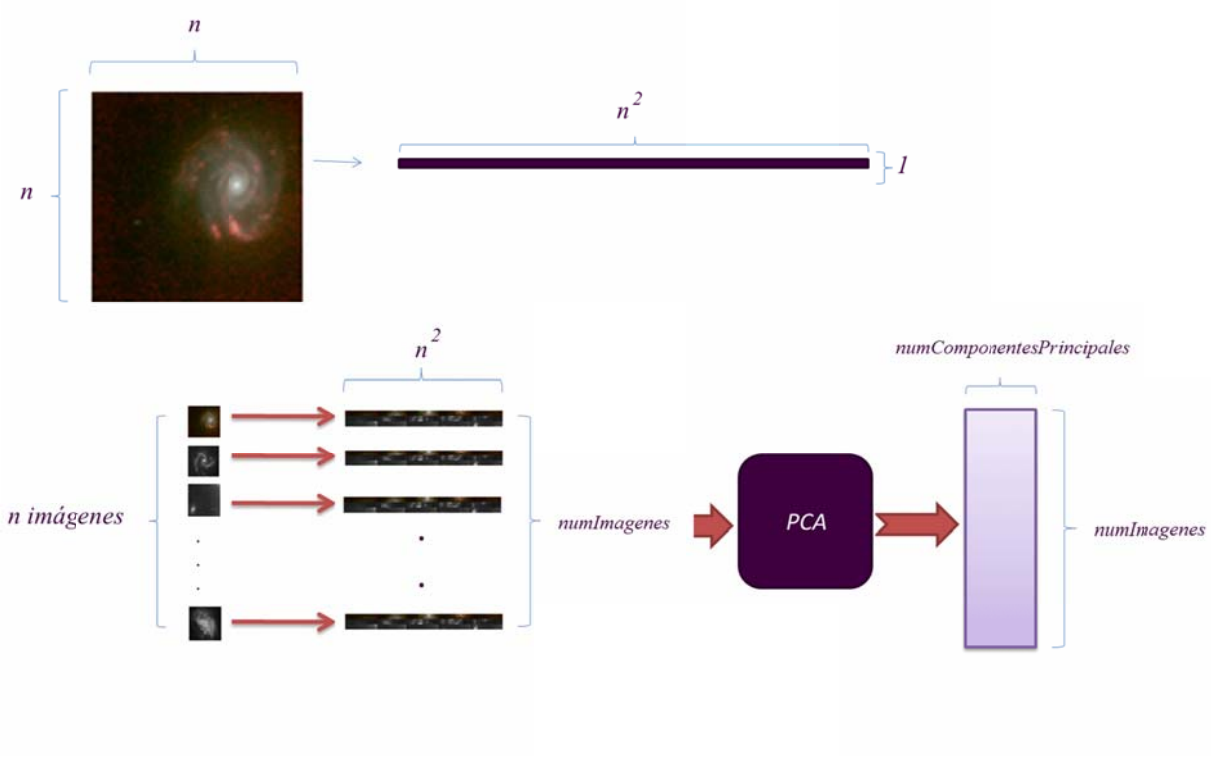


Imagen 6. *PCA aplicado a imágenes.*

2.7 Weka

WEKA, acrónimo de Waikato Environment for Knowledge Analysis, es un entorno para experimentación de análisis del conocimiento que permite aplicar, analizar y evaluar las técnicas más relevantes del aprendizaje automático sobre cualquier conjunto de datos los cuales deben de estar en formato ARFF (Attribute-Relation File Format). (Ramírez, 2006).

Weka contiene herramientas para el pre procesamiento, clasificación, regresión, agrupamiento, reglas de asociación y visualización de datos (Ian H. Witten, 2011).

La licencia de Weka es GPL[2], lo que significa que este programa es de libre distribución y difusión.

Además, ya que Weka está programado en Java, es independiente de la arquitectura, ya que funciona en cualquier plataforma sobre la que haya una máquina virtual Java disponible.

[2] ***G****NU* ***P****ublic* ***L****icense*. http://www.gnu.org/copyleft/gpl.html

3 METODOLOGIA

3.1 Conjunto de datos.

Se usaron conjuntos de imágenes de portadas de libros[3] en formato jpg con resoluciones de 50x50px (Imagen 7) y 100x100px (Imagen 8).

Imagen 7. *Ejemplo de Portadas de libros de diferentes áreas en formato jpg de 50x50px.*

Imagen 8. *Ejemplo de Portadas de libros de diferentes áreas en formato jpg de 100x100px.*

Las imágenes se agruparon en conjuntos de la siguiente manera (Tabla 2):

Grupo	No. de conjuntos	No. de imágenes
A	2	186
B	3	293
C	4	399
D	5	538
E	6	804

Tabla 2. *Conjuntos agrupados que serán clasificados. Donde Grupo es la etiqueta asociada al número de conjuntos (No. De conjuntos) agrupados.*

[3] Las portadas de libros se clasificaron y agruparon de acuerdo a diferentes áreas de conocimiento (administración, biotecnología, electrónica, física, informática y literatura)

3.2 Obtención de los componentes principales

Se usó la implementación de Matthew Dailey (Pentland, 1991), escrito en Matlab que obtiene un conjunto de componentes principales de un conjunto de imágenes. Se calcularon los valores propios que cubran una varianza del 80% y 90%, por otra parte también se tomó en cuenta los valores arrojados de un componente principal(1PC).

Por tanto, los componentes principales usados para la clasificación se muestran en la tabla 3 y tabla 4 para imágenes de 50x50px y 100x100px respectivamente.

G	NC	NI	PC's -> 80%	PC's -> 90%
A	2	186	40	70
B	3	293	45	90
C	4	399	45	95
D	5	538	50	105
E	6	804	55	125

Tabla 3. *Número de componentes principales (PC's) para obtener una varianza del 80% y 90% del número de conjuntos (NC) contenidos en grupos (G) con un número de imágenes que lo forman (NI) de 50x50px.*

G	NC	NI	PC's -> 80%	PC's -> 90%
A	2	186	60	90
B	3	293	70	120
C	4	399	70	140
D	5	538	80	160
E	6	804	100	220

Tabla 4. *Número de componentes principales (PC's) para obtener una varianza del 80% y 90% del número de conjuntos (NC) contenidos en grupos (G) con un número de imágenes que lo forman (NI) de 100x100px.*

Una vez obtenidos los componentes principales, las imágenes se clasifican usando algoritmos de aprenzaje automatico que vienen implementados en weka.

3.3 Algoritmos de clasificación

Los algoritmos usados fueron support vector machines, redes neuronales y bagging con random forest, cuyas implementaciones en Weka son SMO Polykernel, MultilayerPerceptron y RandomForest.

3.4 Resultados

Se realizará un conjunto de 5 corridas usando el método de validación cruzada usando 10 folds y se promediarán para las métricas de:

- Accuracy = CORRECT / TOTAL
- Precision = TP / (TP + FP)
- Recall = TP / (TP + FN)
- F-measure = 2*TP / (2*TP) + FP + FN

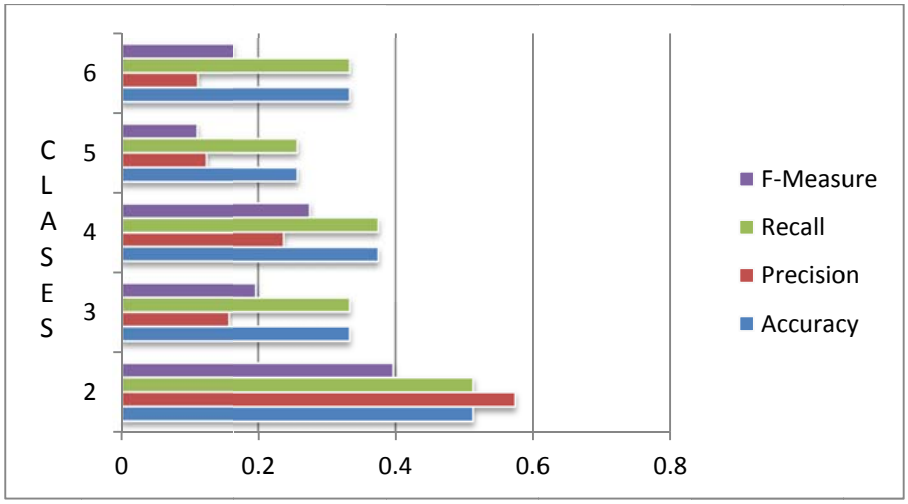
Donde:

- TP: Verdaderos positivos
- FP: Falsos positivos
- FN: Falsos Negativos

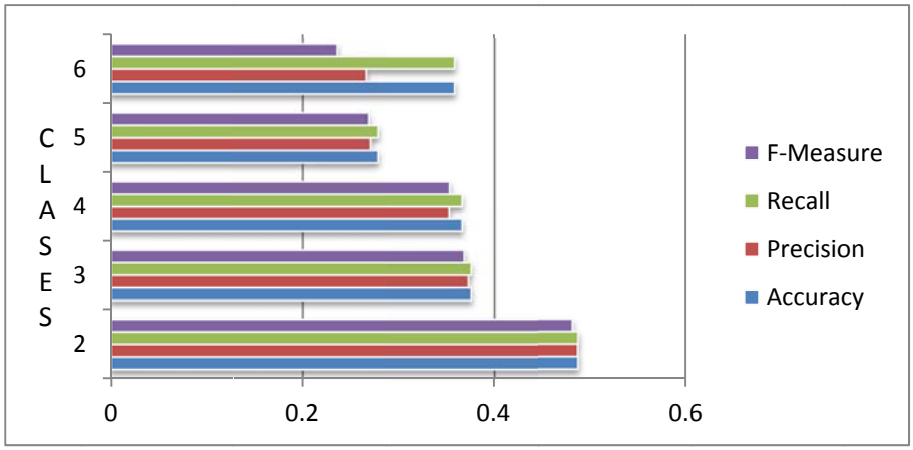
4 RESULTADOS

En las Gráficas 1-9 se muestra la precisión con la cual clasificó cada uno de los algoritmos para grupos de 2, 3, 4, 5 y 6 clases de imágenes de tamaño de 50px.

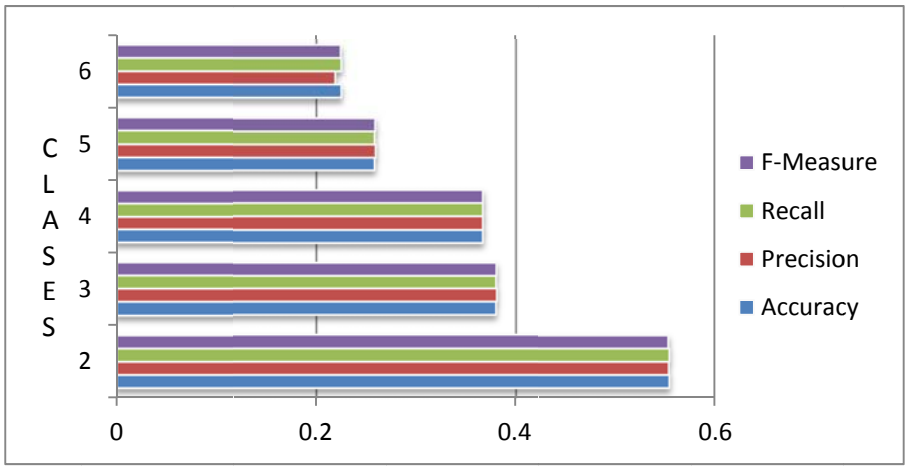
En el Gráfico 1 se observa como clasificó support vector machine para una imagen de 50 pixeles con 1 PC. Del mismo modo en el Gráfico 2 y 3 para los algoritmos de red neuronal y bagging con random forest 10 respectivamente.



Gráfica 1. *Gráfica correspondientes support vector machine pk2 obtenida en weka para imágenes de 50x50px con 1PC*

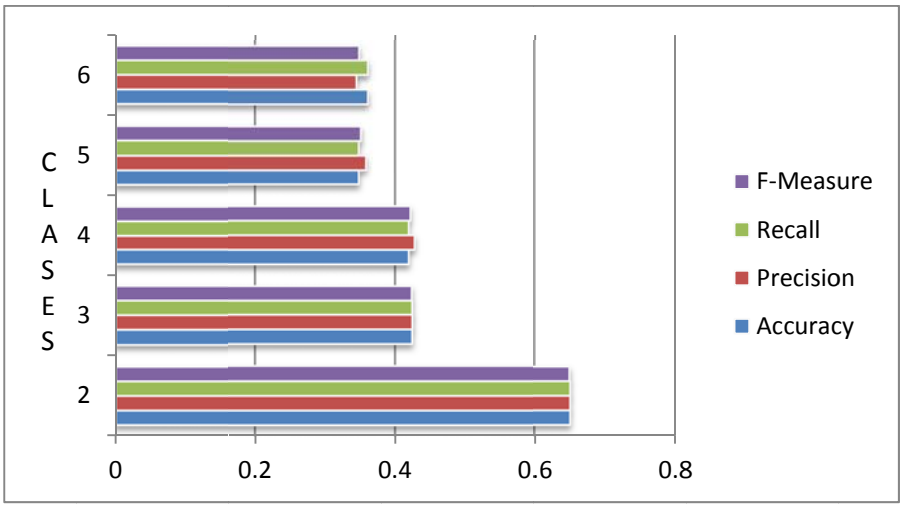


Gráfica 2. *Gráfica correspondientes neuronal network obtenida en weka para imágenes de 50x50px con 1PC*

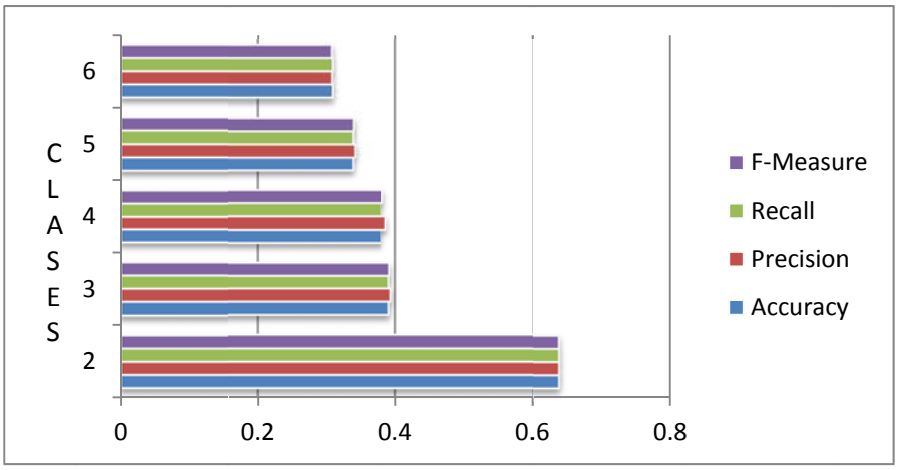


Gráfica 3. *Gráfica correspondientes a Bagging con random forest de 10 árboles obtenida en weka para imágenes de 50x50px con 1PC*

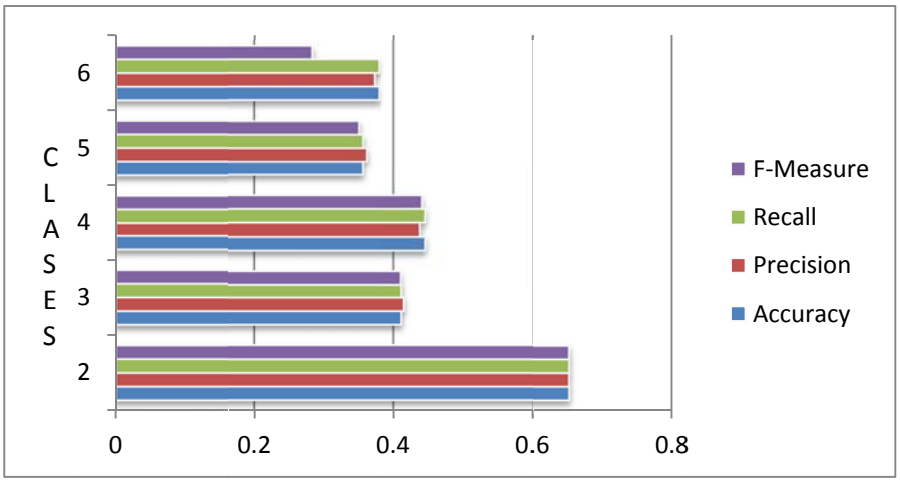
Como podemos observar el método de clasificación que mejor Accuracy tuvo fue el Bagging con 10 árboles de clasificación para imágenes de 50x50px con 1PC. Por otra parte se observa que las mejores métricas de clasificación se obtienen para 2 conjuntos de imágenes de ahí reducen sus valores cuando el número de conjuntos aumenta. Otro detalle es que con redes neuronales las métricas de clasificación sigue el patrón descrito en el párrafo anterior, sin embargo es menor que en los otros dos métodos. Otra observación es que con support vector machines las métricas de clasificación son más inestables, es decir, no se aprecia algún tipo de comportamiento o patrón



Gráfica 4. *Gráfica correspondientes support vector machine pk2 obtenida en weka para imágenes de 50x50px con 80%*

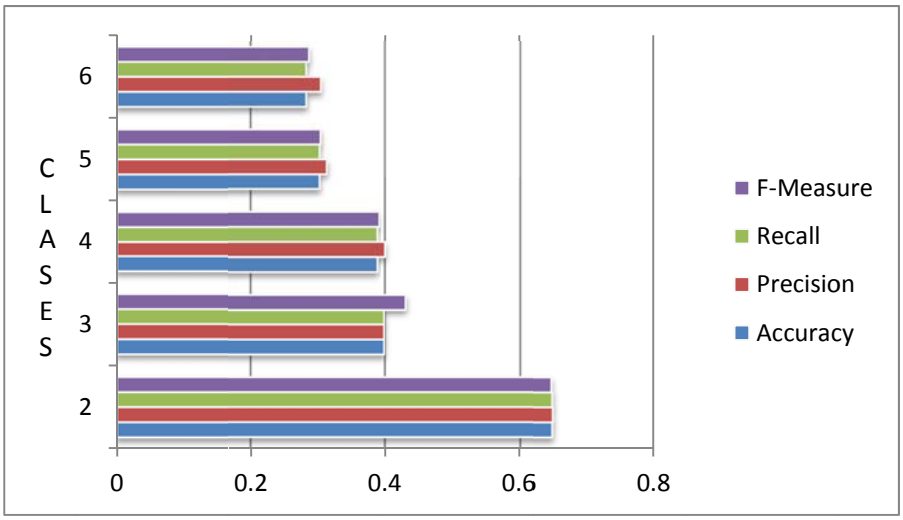


Gráfica 5. *Gráfica correspondientes neuronal network obtenida en weka para imágenes de 50x50px con 80%*

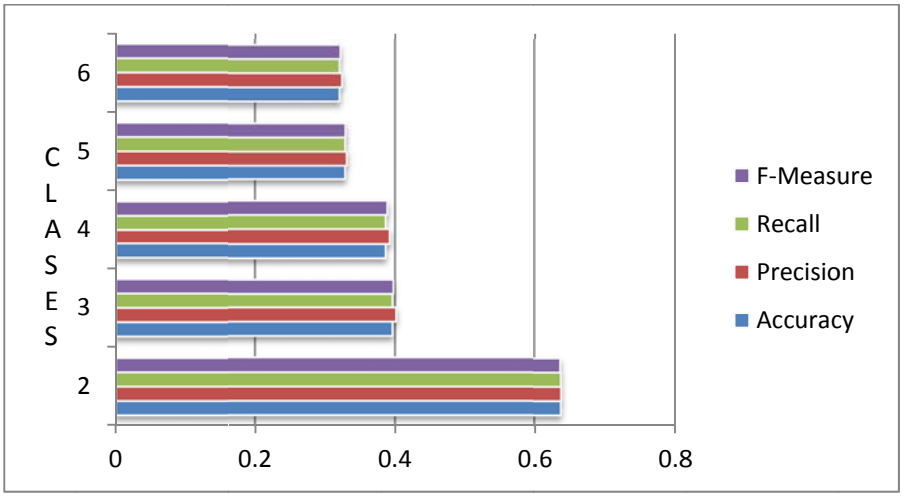


Gráfica 6. *Gráfica correspondientes a Bagging con random forest de 10 árboles obtenida en weka para imágenes de 50x50px con 80%*

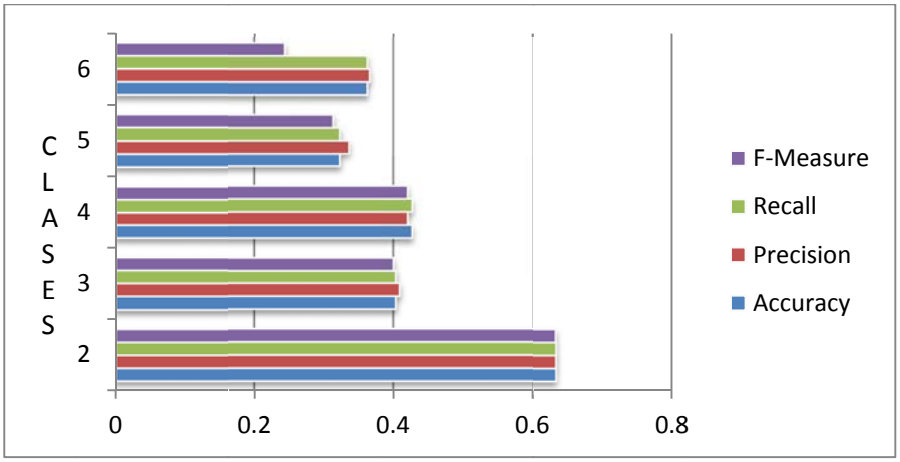
Como podemos observar en las gráficas 4, 5 y 6, el método de clasificación que mejor Accuracy tuvo fue el Bagging con 10 árboles de clasificación aunque los otros 2 métodos no estuvieron tan alejados para imágenes de 50x50px con componentes principales que aproximan a un 80% la varianza. Por otra parte se observa que las mejores métricas de clasificación son para 2 conjuntos de imágenes de ahí reducen sus valores cuando el número de conjuntos aumenta.



Gráfica 7. *Gráfica correspondientes support vector machine pk2 obtenida en weka para imágenes de 50x50px con 90%*



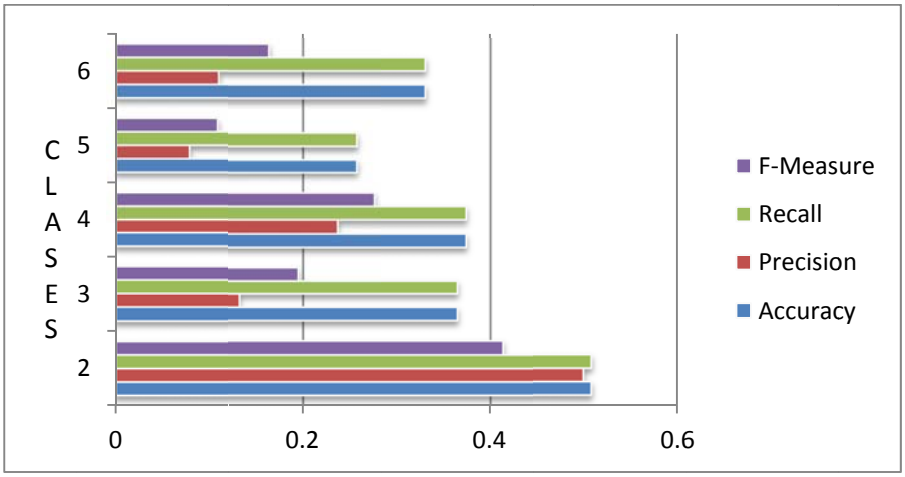
Gráfica 8. *Gráfica correspondientes neuronal network obtenida en weka para imágenes de 50x50px con 90%*



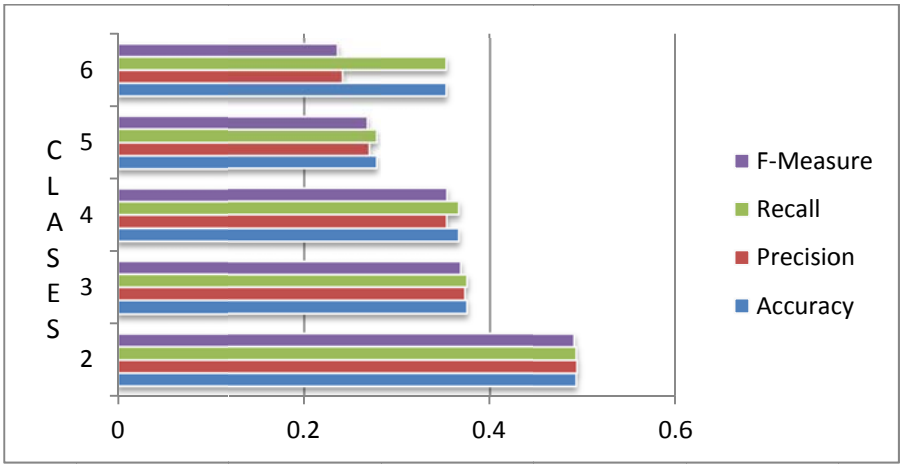
Gráfica 9. *Gráfica correspondientes a Bagging con random forest de 10 árboles obtenida en weka para imágenes de 50x50px con 90%*

Como podemos observar en las gráficas 7, 8 y 9, los métodos de clasificación no variaron mucho en sus métricas de clasificación para imágenes de 50x50px con componentes principales que aproximan a un 90% la varianza. Observamos que el que obtiene mejores métricas de clasificación para todos los grupos de conjuntos de componentes principales es Bagging con 10 árboles de clasificación. Se aprecia que no varía mucho las métricas de clasificación para los tres clasificadores, por otra parte se logra distinguir que las mejore métricas de clasificación se logra para 2 conjuntos de imágenes de ahí reducen sus valores cuando el número de conjuntos aumenta.

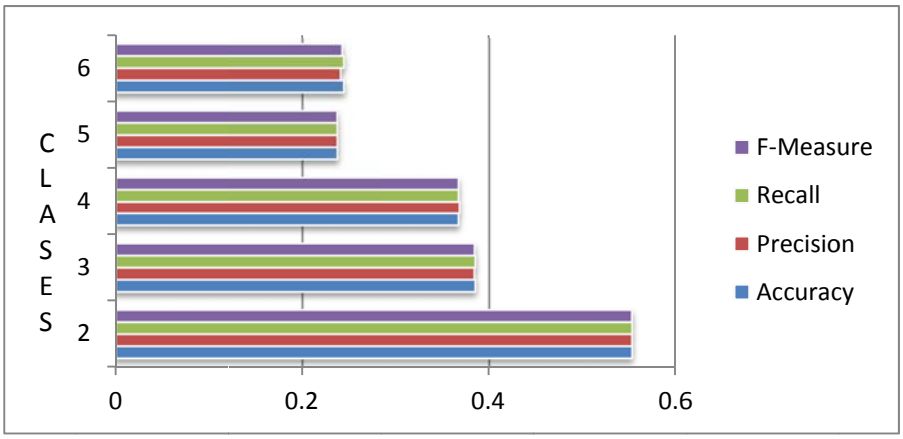
En las Gráficas 10-18 muestran la precisión con la cual clasifico cada uno de los algoritmos para grupos de 2, 3, 4, 5 y 6 clases de imágenes de tamaño de 100px.



Gráfica 10. *Gráfica correspondientes support vector machine pk2 obtenida en weka para imágenes de 100x100px con 1PC*

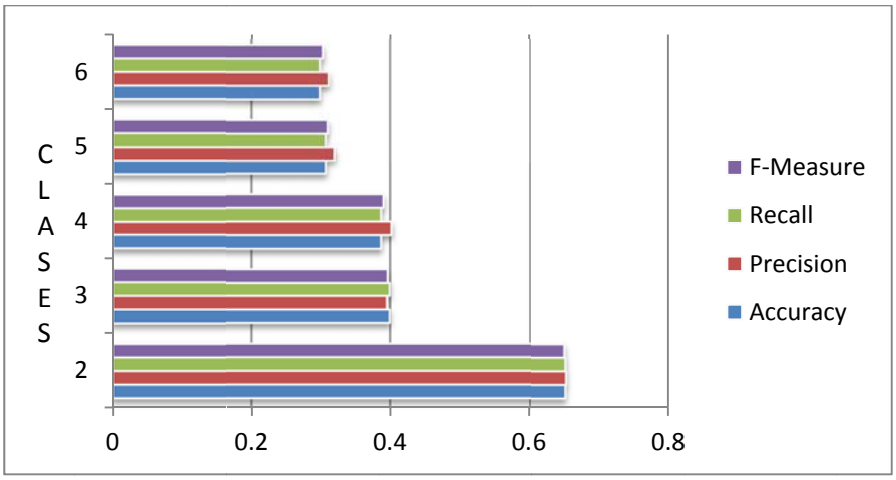


Gráfica 11. *Gráfica correspondientes neuronal network obtenida en weka para imágenes de 100x100px con 1PC*

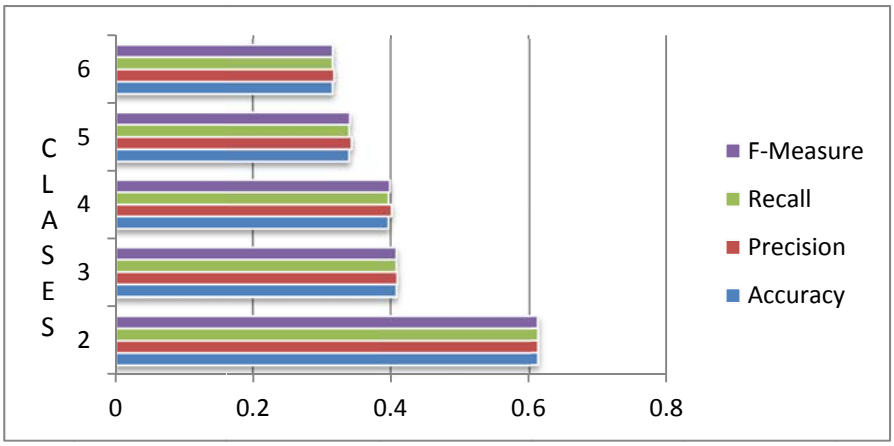


Gráfica 12. *Gráfica correspondientes a Bagging con random forest de 10 árboles obtenida en weka para imágenes de 100x100px con 1PC*

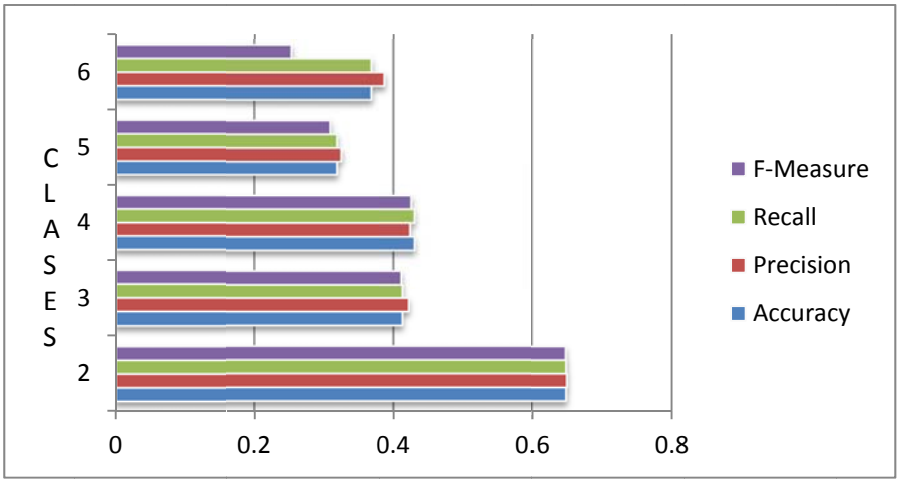
Si comparamos las Gráficas 1, 2 y 3 con 10, 11 y 12 en el cual el tamaño de la imagen trabajada duplica su tamaño, podemos apreciar que son muy parecido su comportamientos en cuanto incrementa el número de conjuntos, sin embargo la precisión incrementa al menos un poco más.



Gráfica 13. *Gráfica correspondientes support vector machine pk2 obtenida en weka para imágenes de 100x100px con 80%*

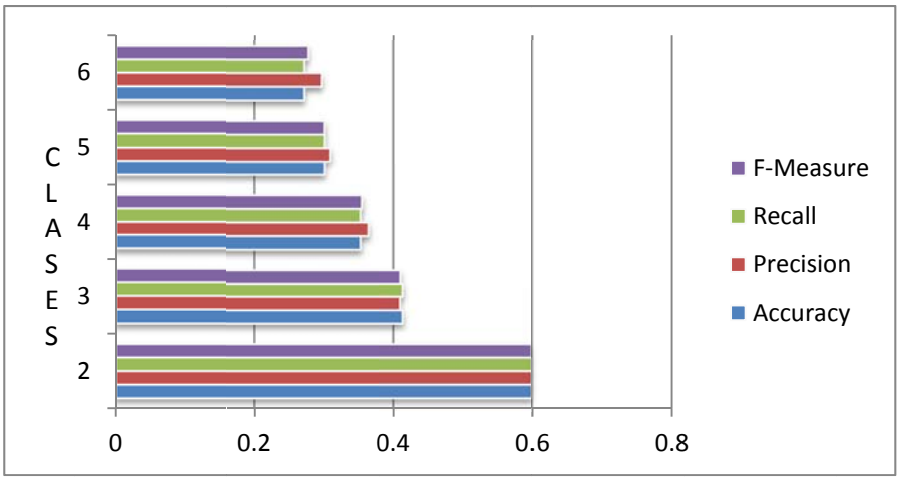


Gráfica 14. *Gráfica correspondientos neuronal network obtenida en weka para imágenes de 100x100px con 80%*

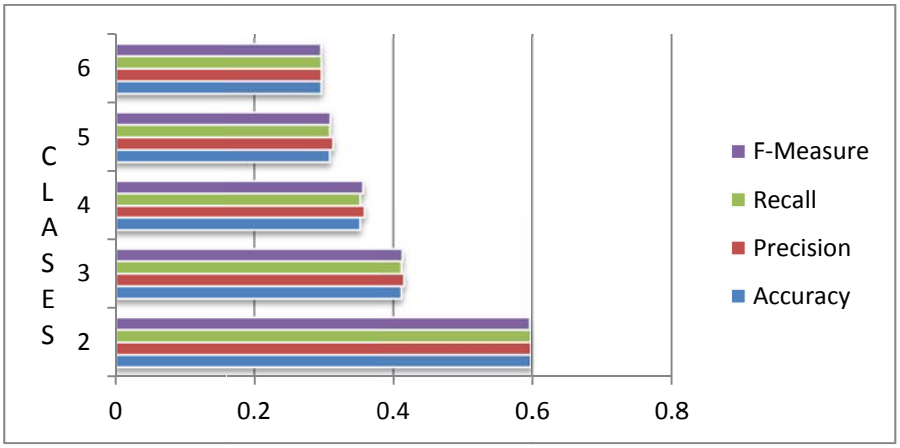


Gráfica 15. *Gráfica correspondientes a Bagging con random forest de 10 árboles obtenida en weka para imágenes de 100x100px con 80%*

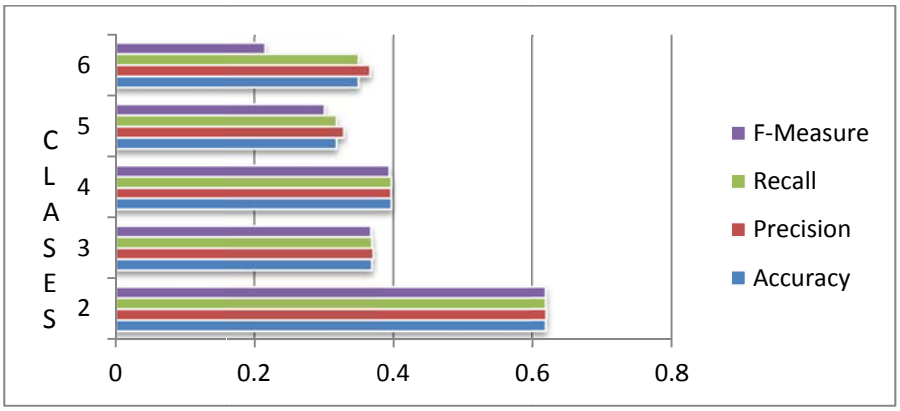
Si comparamos las Gráficas 4, 5 y 6 con 13, 14 y 15 en el cual el tamaño de la imagen trabajada duplica su tamaño, podemos apreciar que son muy parecido su comportamientos en cuanto incrementa el número de conjuntos, sin embargo la precisión incrementa al menos un poco más.



Grafica 16. *Grafica correspondientes support vector machine pk2 obtenida en weka para imágenes de 100x100px con 90%*



Grafica 17. *Grafica correspondientes neuronal network obtenida en weka para imágenes de 100x100px con 90%*



Grafica 18. *Grafica correspondientes a Bagging con random forest de 10 árboles obtenida en weka para imágenes de 100x100px con90%*

Si comparamos las Gráficas 7, 8 y 9 con 16, 17 y 18 en el cual el tamaño de la imagen trabajada duplica su tamaño, podemos apreciar que son muy parecido su comportamientos en cuanto incrementa el número de conjuntos, sin embargo la precisión incrementa al menos un poco más.

Finalmente, observamos que el que mejor clasificó fue Random Forest con 2 clases, por lo que en la Imagen 9 se muestra que se tiene un error del 40% y una precisión del 60%.

```
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

Random forest of 10 trees, each constructed while considering 7 random features.
Out of bag error: 0.4731



Time taken to build model: 0.17 seconds

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

Correctly Classified Instances         107               57.5269 %
Incorrectly Classified Instances        79               42.4731 %
Kappa statistic                          0.1535
Mean absolute error                      0.4704
Root mean squared error                  0.502
Relative absolute error                 94.116  %
Root relative squared error            100.4057 %
Coverage of cases (0.95 level)          99.4624 %
Mean rel. region size (0.95 level)      99.4624 %
Total Number of Instances              186

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.659     0.505      0.556     0.659     0.603      0.6      ELECTRONICA
                 0.495     0.341      0.603     0.495     0.543      0.6      BIOTECNOLOGIA
Weighted Avg.    0.575     0.421      0.58      0.575     0.573      0.6

=== Confusion Matrix ===

  a  b   <-- classified as
 60 31 |  a = ELECTRONICA
 48 47 |  b = BIOTECNOLOGIA
```

Imagen 9. *Ejemplo de la clasificación de los eigenvectores (90%PC) de dos conjunto de portadas de libros de electrónica y biotecnología de 100x100 pixeles*

4.1 Discusión

Para todos los resultados independientemente de la técnica de clasificación la mejor categorización fue del 64%, esto nos hace suponer que los datos de entrada traían cierta cantidad de ruido o que las imágenes eran demasiado pequeñas y por consecuencia se haya perdido información relevante para la caracterización de las mismas. Por lo que, se propone buscar algunas técnicas para el pre procesamiento de las imágenes como binarización, búsqueda de bordes o reconocimiento de caracteres con el fin de mejorar las métricas de clasificación.

5 CONCLUSIONES

En este trabajo se presentó una metodología para clasificar imágenes de portadas de libros usando algoritmos de aprendizaje automático y análisis de componentes principales.

De los resultados obtenidos, se puede destacar lo siguiente:

- Entre más grande sean las imágenes el número de eigenvectores incrementa.
- Entre mayor sea la cantidad de eigenvectores mejora la precisión del clasificador.
- Uno esperaría que al duplicar la dimensión de las imágenes, se incrementara de manera considerable la precisión, sin embargo la precisión realmente no incrementó de manera significativa. Por lo que en este punto da casi lo mismo trabajar con imágenes de poca dimensión por los costos computacionales.
- En estas experimentaciones realmente no fue muy bueno el desempeño de los clasificadores, debido a que el problema de clasificar portadas por área de trabajo no muestran de alguna manera clara de patrones que los diferencien de manera determinantes. Por lo que al incrementar el número de conjuntos de imágenes el desempeño disminuía.
- En todos los casos Bagging con random forest fue el que mejor clasificó.
- Y por último, la clasificación entre 2 conjuntos es mejor que si se hace entre más. A partir de 3 empiezan a reducir mucho su capacidad de clasificar de manera correcta.

6 BIBLIOGRAFÍA

Araujo, B. S. (2006). *Aprendizaje Automático: Conceptos Básicos y Avanzados.* Pearson.

Ball N., R. J. (2006). Robust Machine Learning Applied to Astronomical Data Sets. I. Star-Galaxy Classification of the Sloan Digital Sky Survey DR3 Using Decision Trees. *The Astrophysical Journal*, ApJ 650 497 .

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 123-140.

De la Calleja, J. a. (6). Comparación de algoritmos de aprendizaje automático para la clasificación de datos. *Visón Politécnica*, 11-16.

Drummond, E. R. (2006). Machine Learning for High-Speed Corner Detection . *Lecture Notes in Computer Science*, 430-443.

Eli, N. a. (08 de 05 de 2008). *Advanced Tech Computing Group UTPL*. Recuperado el 05 de 05 de 2012, de ATGC Grupo de Tecnologías Avanzadas en Computación : http://advancedtech.wordpress.com/2008/05/08/clasificacion-en-el-entorno-del-aprendizaje-automatico/

H., C. S. (2003). Machine learning in DNA microarray analysis for cancer classification. *Proceedings of the First Asia-Pacific bioinformatics conference on Bioinformatics* , 189-198 .

Ian H. Witten, E. F. (2011). *Data Mining: Practical Machine Learning Tools and Techniques.* Waikako: The Morgan Kaufmann Series in Data Management Systems.

Jones, P. V. (2004). Robust Real-Time Face Detection. *International Journal of Computer Vision, 57*(2), 137-154.

Knight, E. R. (1994). *Inteligencia artificial.* Madrid: McGraw-Hill.

Metropolis, N. (1985). *History of Computing in the Twentieth Century.* New York: Academic Press, INC.

Mitchell, T. M. (1997). *Machine Learning.* McGraw-Hill Science/Engineering/Math.

Oscar, M. F. (1987). *Intelligence: The Eye, the Brain, and the Computer.*

Pentland, M. T. (1991). Eigenfaces for Recognition. *Journal of Cognitive Neuroscience, vol.3, no. 1*, 71-86.

Peña L., G. J. (2006). Clasificación Automática Sensible al costo para la Detección de Neuropatías Periféricas Focales. *SISOFT*, 39-46.

Ramírez, J. H. (Marzo de 2006). *Universidad Politécnica de Valencia*. Recuperado el 01 de Enero de 2012, de Curso de Doctorado. Extracción Automática de conocimiento en bases de datos e ingenieria del software: http://users.dsic.upv.es/~cferri/weka/CursDoctorat-weka.pdf

Riezler, S. (2002). Parsing the wall street journal using a Lexical-Functional Grammar and discriminative estimation techniques. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 271-278.

Shin J., C. J. (2010). Voice activity detection based on statistical models and machinelearning approaches. *Computer Speech & Language, 24*(3), 515-530.

Stanley, K. (2005). Evolving Neural Network Agents in the NERO Video Game. *Symposium on Computational Intelligence and Games*, 653 - 668.

Stolfo, C. P. (1998). Toward Scalable Learning with Non-uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection. *American Association for Artificial Intelligence (KDD-98)*, 166-168.

Stone, N. K. (2004). Machine Learning for Fast Quadrupedal Locomotion. 211-216.

Stuart J. Russell, P. N. (2004). *Inteligencia Artificial Un Enfoque Moderno* (2 ed.). Prentice Hall.

Tomás, D. (2005). Una aproximación multilingüe a la clasificación de preguntas basada en aprendizaje automático. *Procesamiento del Lenguaje Natural*, 391-398.

Turing, A. M. (1950). Computing Machinary and Intelligence. *Mind New Series*, 433-460.