Comparación de algoritmos de aprendizaje automático para la clasificación de datos

Jorge de la Calleja[1], Pilar Pérez Conde[2], Ignacio Huitzil[2], Eric Olmedo[2], Antonio Benitez[2], Ma. Auxilio Medina[1]
[1] *Ingeniería en Informática, Universidad Politécnica de Puebla*
{jdelacalleja,abenitez,mauxmedina}@uppuebla.edu.mx
[2] *Maestría en Ingeniería en Sistemas y Cómputo Inteligente, Universidad Politécnica de Puebla*

Resumen

En este artículo se presenta un estudio experimental sobre la clasificación de datos usando algoritmos de aprendizaje automático. Los algoritmos usados fueron redes neuronales artificiales, el clasificador Naive Bayes, support vector machines, árboles de decisión y métodos basados en instancias. Los conjuntos de datos fueron de diversos dominios y obtenidos del UCI Machine Learning Repository de la Universidad de California en Irvine. De acuerdo con los resultados experimentales, el algoritmo que mejor se desempeñó fue el de redes neuronales artificiales con un promedio de exactitud del 84.09%, mientras que support vector machines, usando un kernel de función de base radial, obtuvo en promedio los peores resultados con un 65.19% de exactitud.

1. Introducción

El aprendizaje automático es una disciplina que integra a la inteligencia artificial cuyo principal objetivo es desarrollar programas de computadora que puedan "aprender" a realizar alguna automáticamente y que por medio de la experiencia mejoren su desempeño. Por ejemplo, suponga que la tarea por aprender es que una computadora conduzca un vehículo de manera autónoma por una carretera, entonces se esperaría que fuera capaz de, no solo guiar al vehículo correctamente, sino que además incorporara la información obtenida por medio de algún sensor y de esta manera mejore su conducción.

Algunas aplicaciones exitosas que han involucrado al aprendizaje automático son: el reconocimiento de rostros, el reconocimiento de voz, algunas detecciones de enfermedades, detección de fraudes de tarjetas de crédito, clasificación de objetos astronómicos, reconocimiento de secuencias de ADN, reconocimiento de patrones, análisis del mercado de valores, aplicaciones en videojuegos, en robótica, entre muchas otras más [5].

En el presente trabajo se muestra una comparación del desempeño de algunos de los algoritmos más utilizados en el área de aprendizaje automatico, de acuerdo con la literatura, para hacer clasificación de datos. Para ejecutar los experimentos se usó la herramienta de Weka, que implementa una gran diversidad de algoritmos de aprendizaje automatico, tanto para hacer clasificación como para realizar regresión.

El resto del artículo está organizado de la siguiente manera: en la sección 2 se presentan conceptos básicos sobre aprendizaje automático, así como de los algoritmos usados en este trabajo. En la sección 3 se presenta la metodología seguida para evaluar a los algoritmos, describiendo cada una de las etapas. En la sección 4 los resultados experimentales son mostrados y analizados. Finalmente algunas conclusiones derivadas de este estudio experimental son presentadas en la sección 5.

2. Aprendizaje automático

Aprender, es probablemente el rasgo más distintivo que tenemos los seres humanos. Esto incluye, la adquisición de información para transformarla en conocimientos, el desarrollo de habilidades, el descubrimiento de nuevos hechos, entre muchos otros procesos [5]. El aprendizaje automático (*Machine*

learning) intenta imitar estos procesos al estudiarlos y modelarlos de manera computacional [1].

El aprendizaje automático no es una disciplina aislada, sino que se relaciona con una gran diversidad de áreas como son la teoría de control, la teoría de información, la neurobiología, la estadística, la teoría de la complejidad computacional, entre otras; con el propósito de comprender mejor los procesos del aprendizaje humano, y así desarrollar algoritmos que puedan realizar diversas tareas [1].

Generalmente, el aprendizaje automático se divide en dos grupos: el supervisado y el no supervisado [5]. Para el primer tipo, a los algoritmos se les proporciona un conjunto de datos con la clase de la tarea por aprender. Mientras que en el segundo grupo éstas clases no son proporcionadas y se deja que el algoritmo identifique patrones para agrupar a los datos con características comunes.

Para este trabajo de experimentación se utilizó el aprendizaje supervisado, y a continuación se describen las etapas básicas que éste involucra (Figura 1):

1. Definir la tarea u objetivo por realizar. Para este ejemplo la tarea será reconocer a la letra B.
2. Seleccionar un conjunto de datos que será utilizado para entrenar al algoritmo de aprendizaje automático y éste pueda "aprenderse" la tarea. A este conjunto de datos se le llama conjunto de ejemplos de entrenamiento.
3. Seleccionar un conjunto de datos (diferente al de entrenamiento) que servirá para probar el desempeño del algoritmo de aprendizaje automático sobre la tarea aprendida. A este conjunto de datos se le conoce como conjunto de ejemplos de prueba.
4. Al final se tendrá un algoritmo entrenado que pueda clasificar conjuntos de datos nuevos sobre la tarea especificada.

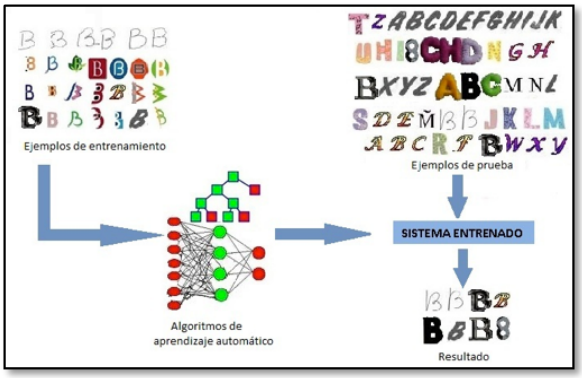

Figura 1. Proceso para aprender la tarea de reconocer la letra B.

Idealmente se espera que los algoritmos de aprendizaje automático puedan desempeñar la tarea para la cual fue entrenado con una exactitud del 100%. Sin embargo, en la práctica es poco probable que esto ocurra debido a muchos factores, que por el momento no serán tratados en este artículo.

2.1 Algoritmos de aprendizaje automático

En esta sección se da una muy breve descripción de los métodos usados, por lo que se recomienda al lector revise las referencias.

2.1.1 Redes neuronales artificiales

Las redes neuronales artificiales (*artificial neural networks*) han sido inspiradas por la observación de los sistemas biológicos neuronales, los cuales están formados por conjuntos de unidades llamadas neuronas que se encuentran densamente interconectadas.

Computacionalmente hablando, existen varias topologías para crear redes neuronales, sin embargo, las más comunes son las redes *feed forward* y las *recurrent networks*. El primer tipo tiene los nodos (neuronas) organizados en una serie de capas llamadas de entrada, ocultas y de salida. En el segundo tipo de red se pueden formar topologías arbitrarias.

Una red neuronal funciona de la siguiente manera: cada nodo produce un valor de salida, que resulta de combinar los valores de entrada de los nodos que le anteceden, de esta forma se alimenta a los siguientes nodos de las capas siguientes hasta producir un valor final en la capa de salida.

Independientemente de la topología de una red neuronal, el primer paso es su entrenamiento, donde los pesos (valores) de los nodos son determinados. Uno de los algoritmos más utilizados para realizar esta etapa de entrenamiento es el llamado algoritmo de retro-propagación (*backpropagation*) [5].

2.1.2 Métodos basados en instancias

Este tipo de métodos simplemente almacenan todos los datos de entrenamiento disponibles, y cuando se requiere clasificar un dato nuevo, se buscan los datos más parecidos a éste para asignarle el valor (clase) correspondiente.

Entre los algoritmos más usados en este tipo de aprendizaje, se encuentran *k*-vecinos más cercanos (*k*-

nearest neighbours) y regresión lineal localmente ponderada (*locally weigthed linear regression*).

2.1.3 Clasificador de Bayes

El clasificador de *Bayes* es un algoritmo probabilístico que se basa en la suposición de que todos los valores de los atributos del conjunto de datos son condicionalmente independientes para los valores objetivo. El clasificador de *Bayes* se utiliza en tareas de aprendizaje en donde cada dato puede ser descrito como una *tupla* de valores-atributo y la función objetivo puede tomar cualquier valor de un conjunto finito de valores. Para clasificar un dato nuevo, este algoritmo le asigna la probabilidad más alta de acuerdo con la función objetivo [5].

2.1.4 Árboles de decisión

Un árbol de decisión se puede definir como una estructura de datos con la información necesaria para tomar decisiones de alguna tarea u objetivo específico. Este algoritmo es uno de los más simples y utilizados por ser prácticos y fáciles de comprender. Algunos algoritmos que forman parte de la familia de árboles de decisión son ID3, ASSISTAN y C4.5 [1].

Los árboles de decisión están compuestos por nodos internos que son las pruebas o preguntas para cada valor de los atributos, las ramas son los posibles valores que pueden tomar los atributos, mientras que los nodos hoja son el valor final de la prueba realizada desde la raíz.

2.1.5 Máquinas de vectores de soporte

Las máquinas de vectores de soporte (*support vector machines*) pertenecen a la familia de métodos basados en funciones de *kernel*, que se encargan de mapear un conjunto de datos a un espacio de alta dimensionalidad, donde cada coordenada corresponde a una caracterísca, transformándolo a un conjunto de puntos en un espacio ecuclidiano.

Uno de los métodos más utilizados es precisamente el de *support vector machines*, que calculan el hiperplano óptimo que separa a un conjunto de datos, resolviendo un problema restringido de optimización cuadrático, cuya solución es obtenida en términos de un subconjunto de patrones de entrenamiento que recaen en la línea de separación. A estos patrones se les llama *vectores de soporte*, y llevan información relevante sobre el problema de clasificación.

2.2 Weka

Weka (*Waikato Environment for Knowledge Analysis*) es un paquete de software de distribución libre desarrollado por la universidad de Waikato Hamilton, Nueva Zelanda [4]. Este paquete contiene herramientas para hacer pre-procesamiento, clasificación, regresión, agrupamiento y visualiación de datos; todas éstas se encuentran agrupadas en cuatro módulos (Figura 2): *simple CLI*, *Explorer*, *Experimenter* y *KnowledgeFlow*.

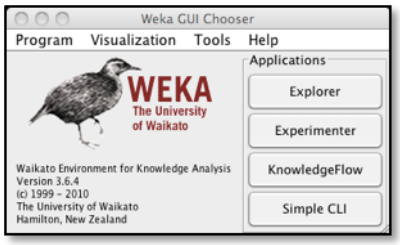

Figura 2. Módulos que integran a Weka.

Para este trabajo se utilizó el módulo de *Explorer* (Figura 3), que permite abrir un archivo, experimentar con diversos algoritmos de aprendizaje, visualizar la distribución del conjunto de datos, entre otras funcionalidades.

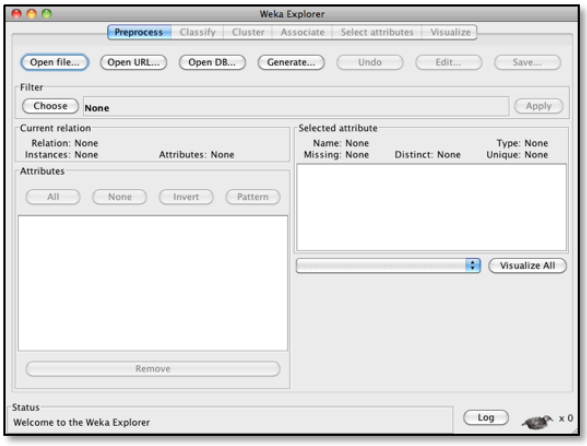

Figura 3. El módulo de Explorer, que incorpora varias funcionalidades para experimentar con datos.

Weka puede leer distintos formatos para los conjuntos de datos, sin embargo, el que usa por omisión es el *.arff* (*atribute-relation file format*). Este tipo de archivo utiliza texto ASCII que describe una lista de ejemplos con sus respectivos atributos.

3. Metodología

Para realizar la clasificación de los diferentes conjuntos de datos, se realizaron tres etapas: 1) Selección del conjunto de datos, 2) Creación de archivos *.arff* para los conjuntos, y 3) Clasificación de datos con Weka. A continuación se describe con más detalles cada una de las etapas mencionadas.

3.1 Selección de datos

Los conjuntos de datos se obtuvieron del *UCI Machine Learning Repository* de la Universidad de California en Irvine, y disponibles en http://archive.ics.uci.edu.ml. Estos datos se encuentran organizados en dos grupos: datos para clasificación y datos para regresión. En este trabajo se usaron aquellos de la sección de clasificación.

Los conjuntos se eligieron al azar, sin embargo, se prefirieron aquellos que no les faltaran atributos. Así, en la Tabla 1 se muestran los conjuntos elegidos y se dan algunas características de los mismos.

Nombre del conjunto	No. de ejemplos	No. de atributos	No. de clases
Car	1728	6	4
Heart	270	14	2
Glass	214	10	7
CPU	209	8	8
Lenses	24	4	3
Mammographic	961	6	2
Diabetes	768	8	2
Tic tac toe	958	9	2
Blood	748	5	2
Vowel	528	10	10
Credit	690	15	2
Statlog	946	19	4
Hayes-Roth	160	5	3
Breast cancer Wisconsin	699	10	2
Haberman's Survival	306	3	2
Breast Tissue	106	10	6
Acute inflammations	120	6	2
Balance scale	625	7	3
Teaching assistant evaluation	151	8	3
Ionosphere	351	7	2

Tabla 1. Conjuntos de datos usados en los experimentos.

3.2 Creación de archivos *.arff*

Una vez obtenidos los conjuntos de datos, éstos se deben convertir al formato que reconoce Weka para poder procesarlos [2]. En la figura 4 se muestran las partes que componen un archivo *.arff*. La primera de ellas es el nombre del conjunto de datos (`@relation`), para nuestro ejemplo se llama `heart-disease-simplified`. Enseguida se listan todos los atributos que tiene el cojunto de datos (`@attribute`), así como sus tipos; en este caso se tienen cinco, y uno más que corresponde a la clase. Por último se listan todos los ejemplos del conjunto de datos (`@data`), donde cada renglón corresponde a cada uno de ellos, considerando el orden de los atributos descritos anterioremente. Por ejemplo, el primer renglón estaría describiendo a un ejemplo con las carácterísticas de `age=62`, `sex=male`, `chest_pain_type=typ_angina`, `cholesterol=233` y `exercise_induced_angina=no`. Y su clasificación para este ejemplo sería `not_present`.

```
@relation heart-disease-simplified

@attribute age numeric
@attribute sex {female, male}
@attribute chest_pain_type {typ_angina, sympt, non_anginal,
atyp_angina}
@attribute cholesterol numeric
@attribute exercise_induced_angina { no, yes}
@attribute class { present, not_present}

@data
63,male,typ_angina,233,no,not_present
67,male,asympt,286,yes,present
67,male,asympt,229,yes,present
38,female,non_anginal,?,no,not_present
...
```

Figura 4. Estructra de un archivo *.arff*.

3.1 Clasificación de datos con Weka

Como anteriormente se ha mencionado, el módulo de *Explorer* es el que servirá para ejecutar la clasificación de los datos. Para ello, los siguientes pasos deben llevarse a cabo:

1. Elegir la pestaña de *Preprocess* y enseguida seleccionar *Open file…* para abrir el archivo que contiene el conjunto de datos por clasificar.
2. Elegir la pestaña de *Classify* y luego la opción de *Choose* para seleccionar el algoritmo de aprendizaje automático. Dependiendo del algoritmo seleccionado, varios parámetros pueden modificarse.
3. Por último, ejecutar el algoritmo sobre el conjunto de datos para clasificarlos seleccionando *Start*.

Como resultado de la ejecución del algoritmo se mostrará información sobre el número de ejemplos usados, el porcentaje de ejemplos correctamente clasificados, el porcentaje de ejemplos incorrectamente clasificados, una matriz de confusión, y otras métricas de evaluación (Ver Figura 5).

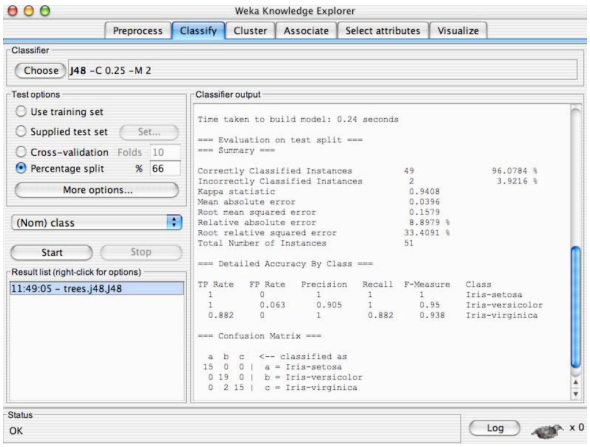

Figura 5. Una vez ejecutado el algoritmo de clasificación, se mostrarán los resultados obtenidos.

4. Resultados

Para realizar los experimentos se usó la técnica de validación cruzada (*cross-validation*), en particular se usó *10-fold cross-validation*. Esta técnica consiste en dividir el conjunto de datos original en diez partes iguales, de las cuales nueve partes son tomadas como conjunto de entrenamiento y una parte como conjunto de prueba. Así, cada experimento se repite diez veces de tal manera que todo el conjunto de datos es utilizado como entrenamiento y prueba. Para que Weka ejecuté esta técnica de experimentación se le debe indicar en la opción correspondiente dentro de la pestaña de *Classify*.

Los resultados que a continuación se presentan corresponden al promedio de realizar diez ejecuciones de *10-fold cross-validation* para cada algoritmo, teniendo cuidado de generar diferentes semillas para cada experimento.

Las implementaciones de los algoritmos usados tienen los siguientes nombres en Weka: *multilayerPerceptron* (redes neuronales), *J48* (árboles de decisión), *IBL/LWL* (métodos basados en instancias), *naive Bayes* (clasificador de *Bayes*) y *SMO* (máquinas de vectores de soporte).

Algunos parámetros fueron modificados para los algoritmos. Para el caso de las máquinas de vectores de soporte se usaron kernels polinomiales de grado 2 y 3, así como una función de base radial. Para los métodos basados en instancias, en el caso de vecinos más cercanos (*IBL*) se usaron 3 y 5 vecinos más cercanos; mientras que para regresión lineal (*LWL*) se consideraron todos los puntos para hacer la aproximación, y en ambos casos se consideró la distancia Euclidiana sin ponderación. Para los métodos de redes neuronales, árboles de decisión y Bayes, se dejaron los parámetros que se presentaban por omisión.

La Tabla 2 muestra la exactitud obtenida por cada algoritmo, la cual esta simplemente definida como el número de ejemplos correctamente clasificados entre el total de los mismos. Cabe mencionar que existen otras métricas que proporcionan una mejor información sobre el desempeño de los clasificadores, sin embargo, por el momento es suficiente con los resultados mostrados por ésta métrica para tener una idea de cómo se desempeñan los algoritmos utilizados.

Como se puede observar, los algoritmos que mejores resultados obtuvieron fueron redes neuronales y *support vector machines* para polinomios de grado 2 y 3; con 84.09%, 82.88% y 80.66% en promedio de exactitud, respectivamente. Sin embargo, los árboles de decisión obtienen en promedio resultados por encima del 80%, esto es, muy cercanos a los mejores. En contraste, el algoritmo que peor se desempeñó fue *support vector machines* usando una función de base radial como kernel, obteniendo en promedio una exactitud del 65.19%; y además obteniendo el peor resultado de todos los algoritmos con un 23.70% para el conjunto de *Breast cancer Wisconsin*.

Analizando los resultados por conjunto de datos, se puede observar que *Acute inflammations* fue el mejor clasificado con un promedio de 97.62%, y obteniendo hasta un 100%. Por otro lado el conjunto de *Teaching assistant evaluation* fue en promedio el peor clasificado con un 49.43%. En el caso del conjunto de *Vowel*, que tiene más clases, se obtuvo un promedio de clasificación del 76.58%, aunque *support vector machines* con un polinomio de tercer grado logró clasificarlo en un 99.28%. En cuanto al número de atributos, *Statlog* fue el que más tenía con 19, y en promedio fue clasificado con un 59.77% de exactitud.

Datos	Algoritmos									
	J48	Multilayer Perceptron	IBL		LWL	SMO			Naive Bayes	Promedio
			3 vecinos	5 vecinos		Polinomo grado 2	Polinomio grado 3	RBF		
Car	92.46	99.32	93.24	93.08	70.00	99.24	**99.94**	84.36	85.42	90.78
Heart	78.00	79.92	78.66	79.80	70.74	79.64	75.06	82.38	**83.56**	78.64
Glass	**97.60**	95.62	90.94	88.42	80.80	91.42	94.00	35.50	82.98	84.14
CPU	89.76	86.02	88.70	**90.04**	73.30	68.98	68.04	64.60	81.36	78.97
Lenses	**83.30**	77.52	77.50	75.00	71.68	77.52	69.16	62.50	74.16	74.26
Mammographic	82.22	79.72	77.28	79.96	81.80	79.56	79.22	77.94	**83.16**	80.09
Diabetes	74.08	75.52	73.24	73.90	71.86	**77.24**	77.20	65.10	75.32	73.71
Tic tac toe	85.64	97.26	98.72	98.72	69.90	99.80	**99.82**	75.42	70.12	88.37
Blood	77.86	**78.58**	74.60	77.02	76.20	76.54	76.52	76.20	75.22	76.52
Vowel	80.06	92.10	97.12	93.14	35.72	97.42	**99.28**	31.54	62.88	76.58
Credit	85.56	82.44	84.96	86.74	85.50	84.92	82.14	**87.90**	83.10	84.80
Statlog	69.36	69.98	58.94	61.26	57.64	**74.24**	68.48	26.38	51.70	59.77
Hayes-Roth	71.38	**84.08**	60.46	60.00	80.92	83.48	81.22	51.08	79.54	72.46
Breast cancer Wisconsin	64.90	64.70	65.90	67.20	42.50	58.80	53.40	23.70	**68.70**	56.64
Haberman's Survival	71.46	74.14	69.82	70.08	71.96	73.20	73.32	73.50	**74.48**	72.44
Breast Tissue	94.78	95.28	96.76	**97.04**	91.74	96.36	95.72	96.06	96.12	95.54
Acute inflammations	**100**	**100**	**100**	**100**	**100**	**100**	**100**	83.00	95.66	**97.62**
Balance scale	64.64	**98.48**	84.90	85.28	79.58	94.14	73.78	91.40	91.22	84.82
Teaching assistant evaluation	50.44	60.40	33.26	33.26	**63.18**	54.30	59.62	39.34	51.12	49.43
Ionosphere	89.30	90.78	54.88	53.44	86.60	**90.82**	87.46	76.02	82.40	79.07
Promedio	80.14	**84.09**	77.99	78.16	73.08	82.88	80.66	65.19	77.41	

Tabla 2. Exactitud obtenida por los algoritmos sobre los diferentes conjuntos de datos. Se muestra también el promedio de la exactitud obtenida por algoritmo y por conjunto de datos.

5. Conclusiones

Se ha presentado un estudio experimental sobre el desempeño de cinco algoritmos de aprendizaje automático para clasificar veinte conjuntos distintos de datos. De acuerdo con los resultados obtenidos se puede concluir que el algoritmo de redes neuronales artificiales fue el que mejor desempeño mostró. Por el contrario, el algoritmo de support vector machines, usando una función de base radial como kernel, fue el que peores resultados obtuvo. Tres conjuntos de datos fueron clasificados en promedio por arriba del 90%, Car, Breast Tissue y Acute inflammations. Se puede también comentar que el algoritmo de k-vecinos más cercanos obtiene resultados no muy lejanos a los obtenidos por support vector machines, y prácticamente es un algoritmo mucho más sencillo de comprender e implementar.

Agradecimientos

La experimentación para este trabajo fue realizada por alumnos de la maestría de Ingeniería en Sistemas y Cómputo Inteligente de esta Universidad durante el curso de aprendizaje automático.

Referencias

[1] E. Alpaydin, Introduction to Machine Learning. Cambridge, Massachuset: The MIT Press, 2004.

[2] E. F. Remco R. Bouckaert, R. K. Mark Hall, A. S. Peter Reutemann, and D. Scuse, WEKA Manual for version 3-7-3. University of Waikato, Hamilton, New Zealand, December 2010.

[3] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the smo algorithm," in Proceedings of the twenty first international conference on Machine learning, ICML '04, (New York, NY, USA), ACM, 2004.

[4] Herramienta de software Weka. Universidad de Waikato, Nueva Zelanda. 2011. http://www.cs.waikato.ac.nz/ml/weka/

[5] Mitchell, T. Machine Learning. McGraw Hill, 1997.

[6] *The UCI Machine Learning Repository*. Universidad de California en Irvine. 2011. http://archive.ics.uci.edu/ml/