Parte I
Conceptos

1. ¿Qué es Minería de Datos?

En [12] tenemos, "Minería de datos es el proceso de descubrir nuevas correlaciones y patrones, examinando cuidadosamente a travez de grandes cantidades de datos almacenados en repositorios, utilizando tecnologías de reconocimiento de patrones como también técnicas estadíasticas y matemáticas", entre varias otras definiciones.

Sin embargo, esta actividad es un paso más dentro del proceso "Descubrimiento de Conocimiento en Datos" el cual engloba las actividades desde la obtención de los datos, su limpieza, la construcción de los modelos y su presentación final. En la actualidad el estandard CRISP-DM (Cross-Industry Standard Process for Data Mining) es el más utilizado [10] y detalla los siguientes pasos [12]:

Business Understanding Phase También llamado "Research Understanding Phase". Enuncia los objetivos claramente en términos del negocio/investigación. Traduce estos objectivos en términos de Minería de Datos.

Data Undestanding Phase Colecta los datos. Se familiariza con los datos utilizando Exploratory Data Analysis. Evalúa la calidad de los datos y selecciona los subconjuntos de interes según el paso anterior.

Data Preparation Phase Prepara los datos para ser procesados por pasos siguientes, es el paso mas laborioso. Selecciona y, de ser necesario, transforma las variables (atributos) que se consideran apropiadas para el paso siguiente.

Modeling Phase Alimenta los modelos con los datos preparados y los calibra para óptimo rendimiento. De ser necesario, se puede volver a pasos anteriores a preparar datos para un modelo en particular.

Evaluation Phase Determina si los modelos finales satisfacen los objectivos del primer paso.

Deployment Phase Hacer uso de los modelos, la construcción no es suficiente. Pueder ser desde un simple reporte realizado por un experto hasta la restructuración de la empresa según las recomendaciones del modelo.

En [12] se detallan 5 casos de estudio del CRISP-DM y en [5] podemos encontrar una copia. Debido a que CRISP-DM fue publicado inicialmente en 1999 y no ha presentado revisiones significativas[1] IBM ha presentado una mejora sobre CRISP-DM, "Analytics Solutions Unified Method" o "ASUM-DM"[7].

[1] La pagina oficial *crisp-dm.org* no esta activa en la actualidad, mayo 2016.

Panorama	Temperatura	Humedad	Viento	Tiempo
Soleado	26.7	85	BAJA	65
Nublado	21.4	?	ALTA	15
Soleado	30.1	70	MEDIA	?

Figura 1: Conjunto de datos falso

2. Conjunto de Datos, Instancias y Atributos

2.1. Instancias y atributos

Cada instancia es la minima unidad con la que trabajan los modelos. Cada instancia posee una serie de atributos que son las mediciones repectivas y pueden ser **numericos** o **nominales**. Los atributos numericos estan caracterizado por un dominio continuo, como por ejemplo 2,718281 o -50, y los atributos nominales por un dominio discreto, como por ejemplo *BAJO*, *MEDIO*, *ALTO*.

Una instancia es una fila de la figura 1 y un atributo es un campo. Se puede dar el caso de que el valor de un atributo de una instancia en particular se haya perdido o haya podido ser obtenido, en ese caso se considera el valor especial **perdido** y no todos los modelos son capaces de procesar este valor.

2.2. Conjunto de Datos

Usualmente los modelos se alimentan de conjuntos de datos, en inglés *dataset*, que son conjuntos[2] de instancias donde una de ellas es definida como el atrinuto a predecir o clasificar, solemos llamar a este atributo **clase**.

2.3. Aprendizaje supervisado y no supervisado

Cuando un modelo dispone de instancias con valores no perdidos del atributo clase se denomina **aprendizaje supervisado**, por ejemplo regresion lineal. Cuando no se dispone de esta información es **aprendizaje no supervisado**, por ejemplo clustering. Un ejemplo del primer tipo de instancia es la primera fila de la figura 1 y un ejemplo del segundo tipo es la tercera fila.

Parte II
Modelos

3. Modelos Lineales

Si disponemos de atributos numéricos y buscamos predecir un atributo también numérico, entonces es natural explorar los modelos lineales. Estos modelos asumen que el atributo a predecir $y^{(k)}$ responde a la forma $w_0 + \sum_i w_i x_i^{(k)}$, es decir a un hiperplano. Como se espera la presencia de errores dentro de las mediciones o simplemente se busca aproximar la forma "real" de y se minimiza

[2]En el estricto sentido matemático no son conjuntos ya que nada impide que la misma muestra aparezca varias veces.

alguna métrica de distancia entre $y^{(k)}$ y $y^{(k)'} = w_o + \sum_i w_i x_i$ como pueden ser según [14]:

- Minima Suma de Cuadrados

$$\min \sum_i (y^{(i)} - y^{(i)'})^2$$

- Minimos Valores Absolutos

$$\min \sum_i |y^{(i)} - y^{(i)'}|$$

- M, L y S Estimadores
- Minima Suma Podada de Cuadrados (Least Trimmed Squares)

$$\min \sum_j (y^{(j)} - y^{(j)'})^2, \quad \text{donde} \quad \{y^{(j)} - y^{(j)'}\} \subset \{y^{(i)} - y^{(i)'}\}$$

- Minima Media de Cuadrados

$$\min \text{med}_i (y^{(i)} - y^{(i)'})^2$$

La mayor característica de los métodos lineales es, precisamente, su linealidad. Considerando que es raro encontrar un fenómeno que se comporte linealmente, la linealidad es una desventaja. Sin embargo, esto también permite que el modelo presente un menor sesgo y se verá mas adelante que esto se suele aprovechar (ver pagina 5).

Segun [15], tenemos que LinearRegression() utiliza la minimización de la suma de los cuadrados sin embargo dispone de un método de selección de atributos basado en el Criterio de Información de Akaike (ver página 7) que simplifica el modelo resultante y está activado por defecto.

4. Ejemplos

```
// Cargamos los datos de alguna manera
Instances dataset = ...
// Preparamos una instancia de prueba
Instance testInstance = ...

// Creamos una instancia del modelo
Classifier model = new SimpleLinearRegression();
// Entrenamos con los datos de 'dataset'
model.buildClassifier(dataset);

// Podemos imprimir los detalles del modelo
System.out.println(model);
// Tambien podemos predecir una instancia
double prediction = model.classifyInstance(testInstance);
```

También podemos utilizar LinearRegression(), MultilayerPerceptron()[3] y con algunas modificaciones Logistic().

5. Árboles

Una de las maneras usuales de atacar problemas es mediante la estrategia "divide y conquistarás", los arboles, con su estructura recursiva, represetan fielmente esta idea. Existen multiud de variedades de árboles utilizados en minería de datos, sin embargo la idea predominante es dividir el conjunto de muestras en dos o mas subconjuntos utilizando un criterio de decisión, usualmente el valor de un atributo, y repetir esto hasta que se alcance la máxima complejidad aceptable o repetir hasta que se alcanze una predicción lo suficientemente buena.

En el caso de atributos nominales, la división es trivial y única. Para los atributos numéricos, se dispone de multitud de métodos listados en [11] y [6]. En WEKA disponemos de weka.filters.unsupervised.attribute.NumericToNominal que simplemente crea una biyección entre cada valor numérico y una clase, weka.filters.unsupervised.attribute.Discretize divide el atributo en clases con la misma anchura (equal-width binning) con opciones de optimización o en clases con una misma frecuencia (equal-frecuency binning) y finalmente weka.filters.supervised.attribute.Discretize el cual utiliza el método MDL de Fayyad e Irani, ver [9], que es el método por defecto utilizado en los modelos.

5.1. J48()

El modelo weka.classifiers.trees.J48 es el mas conocido entre los árboles presentes en WEKA, este modelo es la versión libre del algoritmo C4.5. En este modelo se construye un árbol de forma voráz seleccionando el atributo que "mejor" divide a las muestras. El criterio de selección está basado en la ganancia de información midiendo la diferencia de entropía entre el conjunto de datos original y el mismo conjunto de datos divido por el atributo.

$$\text{Entropía} = H = \sum_i -p_i \log_2(p_i)$$

Veamos este ejemplo. Las muestras estan originalmente clasificados en $(0,0,0,0,1,1,1)$ y luego de la división segun un atributo A se llega right $= (0,0,0,1)$ y left $= (0,1,1)$. Medimos la entropía de las muestras antes de la división.

$$H_0 = -\frac{4}{7}\log\frac{4}{7} - \frac{3}{7}\log\frac{3}{7} \approx 0{,}985$$

Medimos la entropía en cada subconjunto.

$$H_{right} = -\frac{3}{4}\log\frac{3}{4} - \frac{1}{4}\log\frac{1}{4} \approx 0{,}811$$

$$H_{left} = -\frac{1}{3}\log\frac{1}{3} - \frac{2}{3}\log\frac{2}{3} \approx 0{,}918$$

Obtenemos la entropía al final de la división haciendo una suma ponderada según el tamaño relativo de los subconjuntos.

$$H_1 = \frac{4}{7}H_{right} + \frac{3}{7}H_{left} = \frac{6}{7} \approx 0{,}857$$

[3]Si activamos la opción -G (GUI) podremos modificar la topología de la red.

Finalmente la Ganancia de Información, en inglés Information Gain, es la diferencia de entropías.

$$IG = H_0 - H_1 \approx 0{,}128$$

Luego de calcular la IG para cada atributo, se selecciona el mayor valor. Y se aplica recursivamente sobre los hijos. El J48() aplica otros procedimientos para simplificar aún mas el árbol resultante.

5.2. RandomTree() y RandomForest()

El RandomTree() utiliza una estrategia simple, seleccionar aleatoriamente k atributos y modelar un árbol utilizando los, finalmente no hay poda. El RandomForest () construye un conjunto de RandomTree().

5.3. Model Tree

Con weka. classifiers . trees . M5P podemos construir un árbol donde sus hojas son Modelos Lineales. Es la implementación del Algoritmo M5 de J. R. Quinlan presentado en [13], y utiliza la capacidad de "simplificar" de los árboles para terminar utilizando regresión.

6. Clustering

Cuando disponemos de datos sin clasificación, tenemos un problema de clustering. En clustering se busca construir una forma de clasificar los datos según su "similaridad". En la práctica la "similaridad" entre dos muestras se mide con alguna métrica [16] como:

- Distancia de Minkowski. Para $n = 1$ se reduce a D. de Manhattan, para $n = 2$ se reduce a la Euclidiana.

$$D(x, y) = \left(\sum_{i=1}^{d} (x_i - y_i)^{1/n} \right)^n$$

$$\lim_{n \to \infty} D(x, y) = \max_{1 \leq i \leq d} (x_i - y_i)$$

- Similaridad del Coseno.

$$S(i, j) = \cos \alpha = \frac{\boldsymbol{x}_i^T \boldsymbol{x}_j}{|\boldsymbol{x}_i||\boldsymbol{x}_j|}$$

El problema de clustering es NP-dificil[1] por ello se vuelve facilmente imprácticco encontrar el óptimo global determinisiticamente. Se suele utilizar heuristicas y distintas ejecuciones para evitar óptimos locales.

6.1. k-means

El algoritmo de k-means es el algoritmo de clustering mas conocido y se han propuesto multiples modificaciones para mejorar su desempeño. En k-means se inicia creando en ubicaciones aleatorias k puntos los cuales serán los representantes de cada clase. Luego se itera, separando las muetras en grupos tales

que una muestra pertenece a la clase representada por el punto representante mas cercano y al finalizar esto se reubican los puntos representantes en el centroide de las muestras de su clase. El algoritmo converge cuando no ocurren reasignaciones, mide la similaridad con la distancia euclidiana y busca el óptimo minimizando la suma de los cuadrados de las distancias entre las muestras y su respectivo centroide. Es muy sensible a las condiciones iniciales, es decir a las posiciones iniciales de los centroides, por ello se suele correr varias veces. La elección de la k es arbitraria, puede considerarse como un parámetro mas del algoritmo y probar distintos valores o puede ser proveído por el problema, como puede ser la opinión de un experto.

Esencialmente, k-means es un algoritmo de optimización donde se le han definido conceptos del problema de clustering. Por ello, clustering puede ser atacado con todos los recursos ya existentes enfocados a problemas de optimización.

Una de las mejoras es k-means++, el cual modifica el procedimiento de inicialización de los centroides. Provee una mayor carga en la inicialización, sin embargo el tiempo total se reduce y reduce considerablemente los errores finales. Sea $D(x)$ la distancia entre la muestra x y el punto representante mas cercano, entonces el algoritmo k-means++ es el siguiente:

1. Seleccionar como el primer punto representante una de las n muestras con una probabilidad uniforme. Es decir, cada muestra tiene la misma probabilidad de ser selecionada.

2. De las $n-1$ muestras, seleccionar el siguiente punto representante con una probabilidad igual a $\frac{D(x_i)^2}{\sum_j D(x_j)^2}$. Así se busca penalizar la selección de puntos cercanos a puntos representantes ya seleccionados.

3. Se repite el paso anterior hasta tener k puntos representantes.

4. Continuamos con el algoritmo usual k-means, utilizando los puntos anteriormente seleccionados.

Notamos que los centroides inicialmente estan en las mismas posiciones que algunas muestras. Los fundamentos teoricos y datos experimentales en [2]. Existe una mejora sobre k-means++ llamado k-means||o Scalable k-means++, que ofrece las mismas mejoras que k-means pero reduce el costo de inicialización de los centroides [3].

6.2. Fuzzy Clustering

Otra variación es la función de pertenencia de una instancia a un grupo. El algoritmo mas conocido es Fuzzy C-means, el cual es muy similar al K-means[4]. Los puntos mas interesantes son:

La función de pertenencia Un punto x pertence a un grupo k según la función $w_k(x) \in [0,1]$, donde $w_k(x) = 0$ sería el equivalente a no pertenecer al grupo en k-means y $w_k(x) = 1$ sería el equivalente a pertencer al grupo en k-means.

La inicialización Los valores iniciales de $w_k(x) \quad \forall x$ son inicializados aleatoriamente

La ubicacion de los centroides Los centroides son la media de todos los puntos ponderados según su pertenencia a dicho grupo.

$$c_k = \frac{\sum_x w_k^m(x)x}{\sum_x w_k^m(x)}$$

donde $m \geq 1$ es el "difusificador" (fuzzifier). $m = 1$ hace que $w_k(x)$ convergan a 0 o 1 y a mayores valores de m menores valores de $w_k(x)^m$ y el grupo se vuelve mas difuso.

Parte III
Evaluacion

Parte IV
Selección de Modelos

7. Principio de Parsimonia

Tambien llamado Navaja de Ockham o, en inglés, Ockham's Razor. Es un principio según el cual "entre varias explicaciones equivalentes, se prefiere la mas simple".

Cabe destacar que fueron muchas las críticas contra este principio por "ser imprudente". La *anti navaja* de A. Einstein: "Simple, pero no más simple.".

8. Maximum Log-Likelihood

En español, maximización de verosimilitud. Dado un modelo con parametros θ y un conjunto de muestras X independientes e identicamente distribuidos, la verosimilitud de los parametros dado el conjunto de muestras $\mathcal{L}(\theta|X)$ es igual a la probabilidad de obtener las muestras dado los parametros $P(X|\theta)$. Es decir, cual es la "probabilidad" de que los parametros θ "expliquen" las muestras X.

Luego tenemos que si maximizamos $\mathcal{L}(\theta|X)$ tendremos un conjunto de parametros $\hat{\theta}$ tal que sean, o sean lo suficientemente cercanos a, los parametros "reales"[4]. Entonces tendremos que $\mathcal{L}(\theta|X) = P(X|\theta) = \prod_i P(x_i|\theta)$, sin embargo maximizar esto en la práctica es dificil por ello se busca maximizar el logaritmo de el, $\text{máx}_\theta\, \mathcal{L}(\theta|X) = \text{máx}_\theta \log \mathcal{L}(\theta|X) = \log \mathcal{L}(\hat{\theta}|X) = \sum_i P(x_i|\theta)$.

9. Akaike Information Criterion

El AIC se define como $AIC = 2K - 2\ln \mathcal{L}(\hat{\theta}|X)$, donde K es la cantidad de parametros que se utilizaron en el modelo mas uno y $\mathcal{L}(\hat{\theta}|X)$ es el maximo de la función de verosimilitud. Este criterio indica que el modelo es "mejor" si posee un **menor** valor y es un criterio de comparación entre modelos **bajo los mismos**

[4]Suponiendo que el modelo "real" sea el mismo que el utilizado.

datos, es decir que el valor en si mismo no es significativo sino si el valor es menor o no al AIC de otro modelo con el cual comparamos. La interpretación es la siguiente: el término $2K$ es la penalización, por tener signo positivo y estamos buscando un menor valor, por la cantidad de parámetros que el modelo posee ya que a mayor cantidad de parámetros **mayor varianza** y mayor posibilidad de "overfitting", por otra parte el término $-2\ln\mathcal{L}(\hat{\theta}|X)$ favorece a una mejor calificación, por el tener un signo negativo, y está relacionado a verosimilitud del modelo para esos parámetros y esas muestras y también la un **menor sesgo**[8].

Parte V
Importación y Exportación

10. Exportación

Para exportar un modelo hacemos uso de weka.core.SerializationHelper() disponible desde la versión 3.5.5. Destacamos que el método write también trabaja con flujos y que es necesario capturar las excepciones que arroja.

```
// Disponemos del modelo a exportar
Classifier model = ...

// Utilizamos el metodo .write
String outputFilename = ...
SerializationHelper.write(outputFilename, model);
```

11. Importación

Para importar un modelo utilizamos weka.core.SerializationHelper() disponible desde la versión 3.5.5. Destacamos que el método read también trabaja con flujos y que es necesario capturar las excepciones que arroja.

```
Classifier model = (Classifier) SerializationHelper.read(
    inputFilename);
```

12. Trabajado en MATLAB

Para trabajar con las herramientas que provee WEKA utilizamos javaaddpath para agregar la ubicacion de la biblioteca[5].

Referencias

[1] Daniel Aloise y col. «NP-hardness of Euclidean sum-of-squares clustering». En: *Machine learning* 75.2 (2009), págs. 245-248.

[5] El archivo que usualmente llamado "weka.jar".

[2] David Arthur y Sergei Vassilvitskii. «k-means++: The advantages of careful seeding». En: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial y Applied Mathematics. 2007, págs. 1027-1035.

[3] Bahman Bahmani y col. «Scalable k-means++». En: *Proceedings of the VLDB Endowment* 5.7 (2012), págs. 622-633.

[4] James C Bezdek, Robert Ehrlich y William Full. «FCM: The fuzzy c-means clustering algorithm». En: *Computers & Geosciences* 10.2 (1984), págs. 191-203.

[5] *CRISP-DM 1.0 Step-by-step data mining guide*. `https://www.the-modeling-agency.com/crisp-dm.pdf`. Accedido: 2016-mayo.

[6] James Dougherty, Ron Kohavi, Mehran Sahami y col. «Supervised and unsupervised discretization of continuous features». En: *Machine learning: proceedings of the twelfth international conference*. Vol. 12. 1995, págs. 194-202.

[7] *Have you seen ASUM-DM?* `https://developer.ibm.com/predictiveanalytics/2015/10/16/have-you-seen-asum-dm/`. Accedido: 2016-mayo.

[8] Shuhua Hu. *Estimation error*. 2007.

[9] Keki B Irani. «Multi-interval discretization of continuous-valued attributes for classification learning». En: (1993).

[10] *KD Nuggets What main methodology are you using for your analytics, data mining, or data science projects?* `http://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-methodology.html`. Accedido: 2016-mayo.

[11] Sotiris Kotsiantis y Dimitris Kanellopoulos. «Discretization techniques: A recent survey». En: *GESTS International Transactions on Computer Science and Engineering* 32.1 (2006), págs. 47-58.

[12] Daniel T Larose. *Discovering knowledge in data: an introduction to data mining*. John Wiley & Sons, 2014.

[13] John R Quinlan y col. «Learning with continuous classes». En: *5th Australian joint conference on artificial intelligence*. Vol. 92. Singapore. 1992, págs. 343-348.

[14] David Ruppert y Raymond J Carroll. «Trimmed least squares estimation in the linear model». En: *Journal of the American Statistical Association* 75.372 (1980), págs. 828-838.

[15] I.H. Witten, E. Frank y M.A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2011. ISBN: 9780080890364. URL: `https://books.google.com.py/books?id=bDtLM8CODsQC`.

[16] Rui Xu, Donald Wunsch y col. «Survey of clustering algorithms». En: *Neural Networks, IEEE Transactions on* 16.3 (2005), págs. 645-678.