Curso de Especialización de
Inteligencia Artificial y Big Data (IABD)

Sistemas de Big Data

UD04. Análisis y Búsqueda de Respuestas en Datos.
Resumen.

JUAN ANTONIO GARCIA MUELAS

Lógica algorítmica.

La base para **comprender cómo se definen las soluciones a problemas.**

Un algoritmo es un **conjunto de instrucciones** que realizadas en orden permiten **solucion**ar un problema.

Un algoritmo debe ser:

- ✓ **Preciso:** cada paso debe indicar qué hacer sin ambigüedades.
- ✓ **Finito:** con un número limitado de pasos.
- ✓ **Definido:** debe producir siempre el mismo resultado para los mismos valores de entrada.

Es **posible definirlos con cierta independencia del lenguaje** en el que finalmente se vayan a implementar, **mediante diagramas de flujo o** mediante **pseudocódigo** (a medio camino entre el lenguaje natural y lenguaje de programación).

Deben tener un **único nodo de inicio y al menos uno** de **final**.

Un diagrama de flujo debe tener un único nodo de inicio y al menos uno de final.

Combinatoria y explosión combinacional.

Los algoritmos se utilizan para resolver problemas de todo tipo, para los que suele existir un amplísimo **espacio de soluciones** que por lo general está **formado por combinaciones** de posibles acciones o valores.
Permutaciones: Podemos querer **disponer N elementos en un orden determinado**. Eso es una permutación.
Variaciones: Podemos querer **disponer N elementos de entre un total de M en un orden determinado**. Eso es una variación.
Combinaciones: Las combinaciones son como las **variaciones** (tomamos N elementos de un conjunto de N) pero **sin importar** en qué **orden**.

Aparece una **explosión combinatoria** cuando el **número** de posibles soluciones **crece muy rápido** a medida que aumentamos determinados **valores de configuración** del propio problema.

Complejidad computacional.

El **estudio** de la **complejidad** de los problemas en función de los **recursos en tiempo o memoria** necesarios **para resolver**los con el mejor algoritmo posible.

Usamos algoritmos que pueden presentar una **explosión combinacional** y eso produce una determinada **complejidad computacional**.

Hay dos tipos:

- ✓ **Complejidad en tiempo de ejecución:** cuánto tardaremos en resolverlo.
- ✓ **Complejidad en memoria:** cuánta memoria necesitaremos para resolverlo.

La **complejidad** de un problema viene indicada por el **tiempo y/o memoria** que necesite **emplear** el **mejor algoritmo** posible.

Con ello, un problema será considerado ***tratable*** si se puede encontrar un **algoritmo capaz** de resolverlo **en tiempo polinomial**, e ***intratable*** si **no existe** tal **algoritmo** y es necesario emplear un **tiempo exponencial**.

Algoritmo de tiempo polinomial: En el peor de los casos permite solucionar cierto problema en un tiempo determinado por un polinomio en función del tamaño de la entrada. Ejemplo, un

algoritmo que para ordenar una lista de N elementos pueda llegar a necesitar $3N^3+N+8$ segundos en el peor caso, sería polinomial.
Algoritmo de tiempo exponencial: En el peor de los casos permite solucionar cierto problema en un tiempo determinado por una función exponencial con el tamaño de la entrada. Ejemplo, un algoritmo que para ordenar una lista de N elementos pueda llegar a necesitar 2^N segundos.

Un algoritmo de tiempo **polinomial** siempre termina siendo **más rápido** que uno de tiempo exponencial si el **tamaño de la entrada es** lo suficientemente **grande**.

Analítica y búsqueda de respuestas.

Principales metodologías en la Minería de Datos:

- ✓ **Niveles de Analítica de Datos** (según complejidad):
  - ➢ **Análisis Descriptivo:** ¿qué ha ocurrido? Intenta describirlo a través de un cuadro de mando estático con consultas operacionales (beneficio, tendencia, resultados).
  - ➢ **Análisis Diagnóstico:** ¿por qué ha ocurrido? Intenta determinar la causa de un fenómeno con información relacionada con él. Se muestra con herramientas interactivas para identificar tendencias y patrones.
  - ➢ **Análisis Predictivo:** ¿qué va a ocurrir? Utiliza modelos predictivos de ML para riesgos y oportunidades.
  - ➢ **Análisis Prescriptivo:** ¿qué hacer para que algo ocurra?
    El Análisis Prescriptivo **se apoya en los resultados que es capaz de producir el Análisis Predictivo**.

    El **análisis de datos** consiste en **encontrar hechos, relaciones, patrones o tendencias** con la intención de poder **tomar decisiones**.

    La **analítica de datos** es un concepto amplio, que **incluye** dentro al propio **análisis de datos, así como** a otras actividades alrededor del ciclo de vida del dato como:
    - **Obtención/recolección** de los datos desde diversas fuentes.
    - **Limpieza**.
    - **Integración (**desde las diversas fuentes a su representación unificada).
    - **Gobierno de Datos**, mediante el cual se gestiona:
      - Disponibilidad (datos disponibles cuando van a usarse).
      - Usabilidad (datos válidos para lo que se quiere hacer con ellos).
      - Integridad (datos correctos).
      - Seguridad (de modo que no puedan ser accedidos ni desde fuera de la organización ni por quienes no tienen los apropiados permisos dentro de la propia organización).
    - Análisis de Datos (ya mencionado).

    Algunos usos comunes de la **analítica de datos** son:

    - En el mundo de los **negocios**, para disminuir los costes operacionales y facilitar la toma estratégica de decisiones.
    - En el mundo **científico**, para entender por qué ocurre un determinado fenómeno y así mejorar el modo de abordarlo.
    - En el mundo de los **servicios**, para disminuir costes y mejorar su calidad.

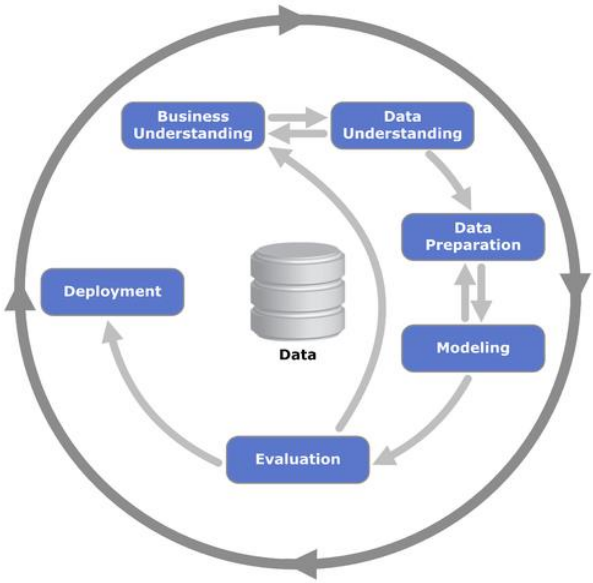
- ✓ **Principales metodologías en Minería de Datos:**
  - ➢ **SEMMA:** Sample (seleccionar datos para el modelado de muestreo con el tamaño suficiente para que sea eficiente), Explore (entender los datos y encontrar relaciones o anormalidades), Modify (Seleccionar, crear y transformar variables para preparar el modelado), Model (aplicar técnicas de minería para crear modelos), Assess (evaluación de los modelos para comprobar su utilidad).

    Es el acrónimo de una lista de pasos que se emplean en muchas ocasiones a modo de metodología para la minería de datos.
  - ➢ **CRISP-DM:** Cross-industry standard process for data mining. Muy utilizada y goza de cierta oficialidad.

    El *CRISP-DM* divide el proceso de minería de datos en 6 fases:
    - Comprensión del negocio.
    - Comprensión de los datos.
    - Preparación de los datos.
    - Modelado.
    - Evaluación.
    - Despliegue.

    La clave es que **no se trata de una secuencia sino de un proceso cíclico**, ya que **propone continuar dando vueltas** por las fases (**aplicando las lecciones aprendidas** durante cada iteración en el ciclo para la siguiente).

Diagrama del proceso CRISP-DM



R para analizar datos.

R es un lenguaje de programación interpretado de código abierto **creado específicamente para facilitar el análisis de datos**.

Es un lenguaje **interpretado, de alto nivel e imperativo** (ejecución línea a línea).

Puede trabajar tanto con valores individuales como con objetos unidimensionales como los vectores (operador `:` crea los valores del vector).

El índice de los vectores comienza en **1**.

Facilita ayuda con el operador `?` delante (`?sum` o `?"if"`).

Asigna variables con los operadores `<- , =` o en vectores con `[1] 7`.

Mostrar resultados: `x <- 3 + 4`

Los Dataframes y las listas de R pueden almacenar distintos tipos de datos.

Los arrays en R son multidimensionales y rectangulares (todas las filas tienen la misma longitud). Las **matrices** son arrays limitados a **2 dimensiones**.

Si trabajamos con R contamos con gran cantidad de conjuntos de datos dentro de paquetes a los cuales podemos acceder si están instalados.

Hay intérpretes online como:

- ✓ myCompiler - Online R Compiler o rdrr.io

Python para analizar datos.

Python es un lenguaje de programación de **alto nivel, interpretado y de propósito general**, pero a diferencia de R, no ha sido creado para el análisis de datos, para lo cual utiliza:

- ✓ **NumPy:** Añade **soporte** para **arrays (multidimensionales y rectangulares) y matrices (bidimensionales**), junto a una gran cantidad de operaciones optimizadas para trabajar sobre esas estructuras de datos.
- ✓ **Pandas:** Escrito **sobre NumPy**, ofrece **estructuras** específicas y así como gran cantidad de **funcionalidades** específicas para **análisis de datos**.
  En un DataFrame de Pandas **podemos añadir y eliminar columnas una vez está creado.**
  Los DataFrame de Pandas son el equivalente a un array bidimensional al que podemos asociar etiquetas tanto para columnas como para filas.
- ✓ **Matplotlib:** Aporta **capacidades de visualización** de datos.