

Curso Análisis de Datos para Financieros

Verónica Ruiz Méndez
vruiz@afi.es

9 de Mayo de 2023
Afi Escuela de Finanzas

Índice

1. ¿Qué es Python?
2. Proceso de Extracción del Conocimiento
3. Preprocesamiento de datos
4. APIs
5. Ejemplos APIs financieras e indicadores



¿Qué es Python?

¿Qué es Python?

Historia de Python

- Creado en 1990 por Guido van Rossum (actualmente en Microsoft).
- El nombre está basado en los humoristas británicos Monty Python.
- A partir del año 2001, pasa a ser administrado por Python Software Foundation, una compañía sin ánimo de lucro con un funcionamiento similar al de Apache Software Foundation.



¿Qué es Python?

- Es un lenguaje de programación **de alto nivel**.
- Es un lenguaje de programación **de propósito general**.
- Es un lenguaje de programación **open source**.
- Es un lenguaje de programación **orientado a objetos**.
- Es un lenguaje de programación **dinámicamente tipado y fuertemente tipado**.
- Es un lenguaje de programación **conciso**.
- Es un lenguaje de programación **con una comunidad muy activa**.
- Es un lenguaje de programación **con infinidad de módulos orientado a muy diferentes dominios** (tratamiento de imágenes, videojuegos, bases de datos, **análisis de datos**, etc.).




¿Qué es Python?

Ventajas

- Python es un **lenguaje de alto nivel multipropósito**. También se utiliza en otros campos más allá del análisis de datos: desarrollo web, scripting ...
- Es un lenguaje más rápido en ejecución (respecto a otros más basados en estadística, por ejemplo R).
- Es **muy fácil de aprender para los principiantes**: curva de aprendizaje menos dura.
- La sintaxis del lenguaje te ayuda a ser un mejor programados: código más condensado y legible.
- Más rápido en el manejo de grandes conjuntos de datos y puede cargar los archivos con facilidad.

¿Qué es Python?

¿Cómo usar Python?

Distribución	Descripción	url_descarga
	<ul style="list-style-type: none">- Core de Python.- Incluye únicamente los paquetes básicos y el intérprete de la consola de comandos.	https://www.python.org/
	<ul style="list-style-type: none">- Distribución más extendida y reconocida de las existentes.- Incluye más de 300 modulos preinstalados desde análisis de datos, hasta desarrollo web pasando por librerías matemáticas.- Incluye una consola gráfica para Python, iPython, Jupyter Notebooks, Spyder y VisualStudioCode.	https://www.anaconda.com/products/individual
	<ul style="list-style-type: none">- Distribución más orientada al análisis científico, con paquetes matemáticos mucho más específicos.- También muy centrada en la visualización de datos, incluyendo paquetes de visualización de datos avanzados.	https://assets.enthought.com/downloads/

¿Qué es Python?

¿Cómo usar Python?

- Existen muchos IDEs (*Integrated Development Environment*) para Python.
- Cada uno de ellos está diseñado para dar soporte a una forma de trabajo en función del dominio (análisis de datos, desarrollo general, programación reproducible...) al que se orienten.
- Se pueden encontrar desde consolas básicas (tipo R o Matlab) hasta entornos completos de desarrollo y despliegue de aplicaciones y servicios (tipo Visual Studio, Eclipse, NetBeans, etc.).



¿Qué es Python?

Instalación de paquetes

PIP

Administrador estándar de paquetes de Python. El instalador de Python instala pip automáticamente. Para instalar paquetes con pip basta con escribir en la consola “pip install nombre_paquete”. Los paquetes se descargan del repositorio oficial de paquetes de Python, llamado PyPi.

CONDA

Gestor de paquetes y entornos de Python. Conda está incluido al instalar la distribución Anaconda. Para instalar paquetes, debemos escribir “conda install nombre_paquete”. Los paquetes se descargan del repositorio de Anaconda.

POETRY

Herramienta para gestionar proyectos en Python. Permite definir los paquetes necesarios en nuestro proyecto y los instala/actualiza.

Ejemplo en Python



AnalisisDatosFinancieros2023_BasicsPython.ipynb

¿Qué es Python?

Librerías - NUMPY

- [NumPy](#) es un módulo de Python, abreviatura de Numerical Python.
- Ofrece estructuras de datos y funciones matemáticas complejas de replicar en el core de Python (computación matricial, operaciones matemáticas sobre grandes conjuntos de información, etc.).
- Dicen que...

Python + NumPy + SciPy + Matplotlib -> MATLAB

*Scipy es una librería de Python que se compone de herramientas y algoritmos matemáticos.

*Matplotlib es una librería de Python para generar gráficos.

[Documentación Numpy](#)

¿Qué es Python?

Librerías - NUMPY

Capacidades principales:

- Estructura de datos para almacenamiento en forma de matrices n-dimensionales: ndarray.
- Funciones matemáticas estándar con rendimiento optimizado para ser aplicadas a matrices complejas sin necesidad de usar bucles.
- Herramientas para la lectura y escritura de información matricial a disco.
- Funciones para aplicación de álgebra lineal.
- Herramientas para integrar código escrito en C, C++ o Fortran.

Ejemplo en Python



AnalisisDatosFinancieros2023_Numpy.ipynb

¿Qué es Python?

Librerías - PANDAS

- [Pandas](#) es un módulo de Python orientado al análisis de datos.
- Creado por Wes McKinney . La primera versión se publicó en 2008.
- Una de las librerías con más evolución y seguimiento por parte de la comunidad (más de 200 contribuidores).

[Documentación Pandas](#)

¿Qué es Python?

Librerías - PANDAS

Características:

- Capacidad de almacenamiento y procesamiento de diferentes estructuras de datos.
- Facilidad para la carga de información desde diferentes fuentes: ficheros CSV, bases de datos relacionales...
- Capacidad para el tratamiento de *missing values*.
- Utilidad tanto para carga y tratamiento de datos, como para el análisis estadístico, exploratorio y modelado.
- Integración con otras librerías.

Ejemplo en Python



AnalisisDatosFinancieros2023_Pandas.ipnyb

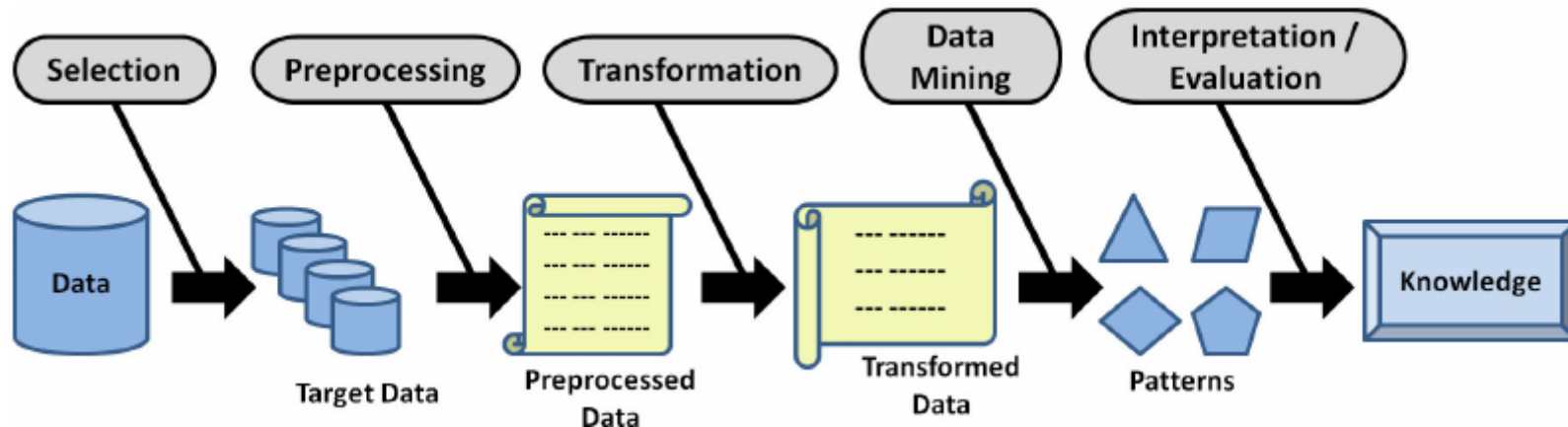


Proceso de Extracción del conocimiento

Proceso de Extracción del Conocimiento

El **Proceso de Extracción del Conocimiento** (Knowledge Discovery in Databases, KDD) es:

- El proceso no trivial de identificar patrones válidos, novedosos, potencialmente útiles e inteligibles en datos.
- Un análisis exploratorio más o menos automático de bases de datos grandes.

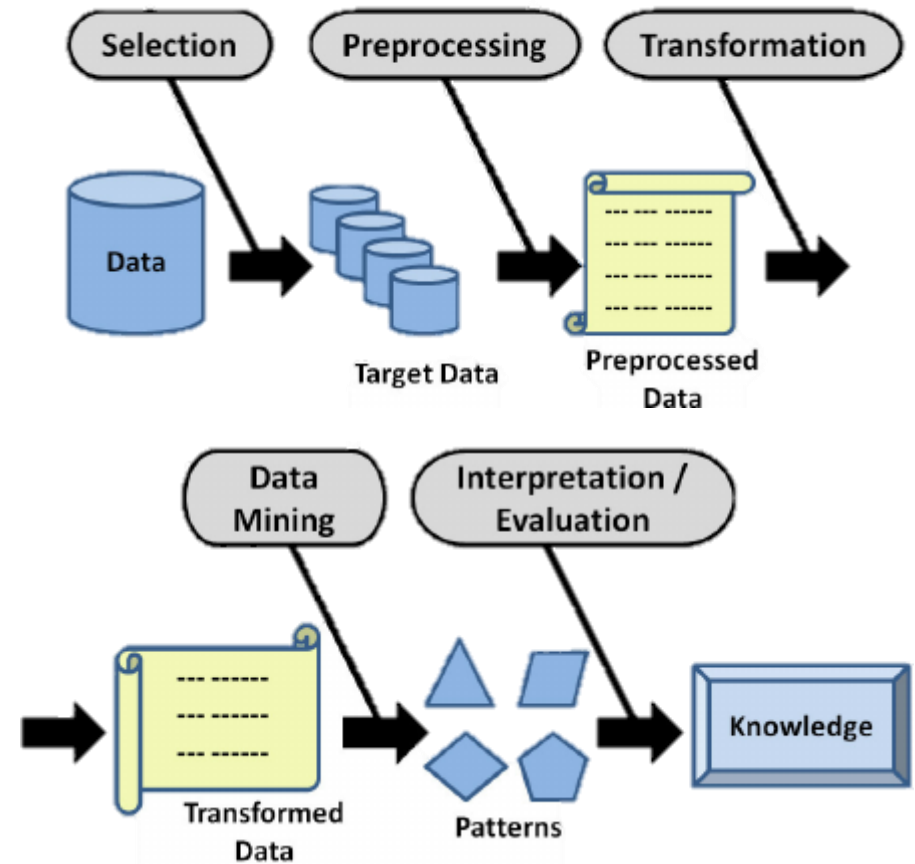


Proceso de Extracción del Conocimiento

Paso 1.

COMPRENDER EL PROBLEMA Y ESTABLECER OBJETIVOS.

Es fundamental tener claros los límites y objetivos que pretendemos. En este paso, se reúne toda la información más importante a utilizar.

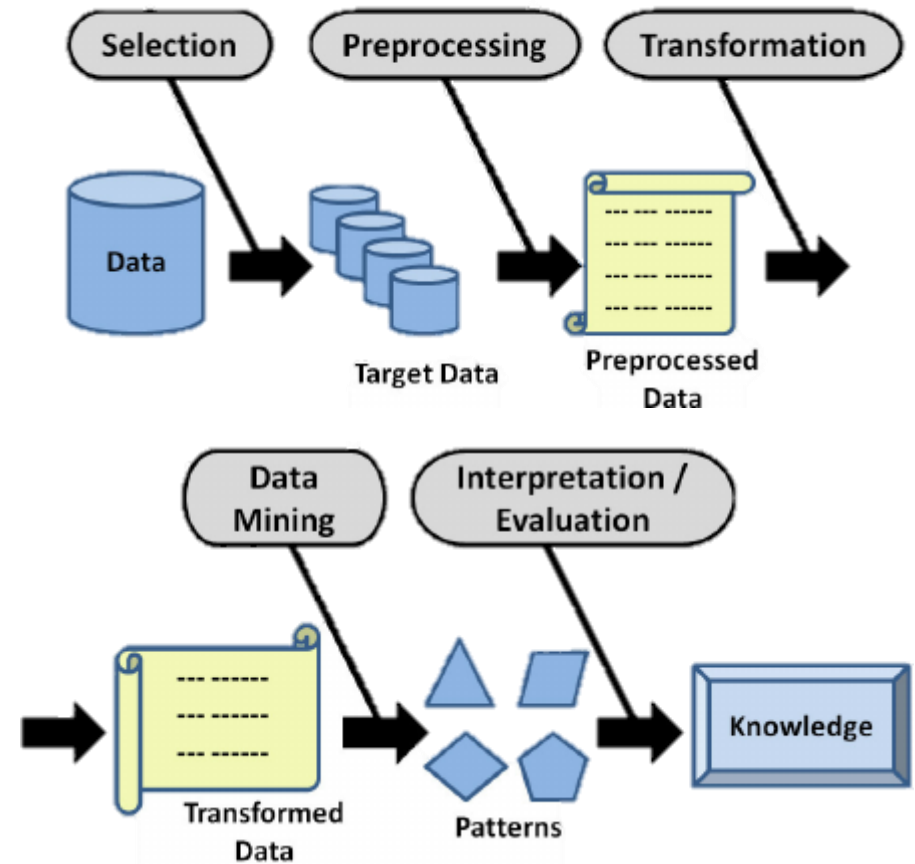


Proceso de Extracción del Conocimiento

Paso 2.

CREAR UN SET DE DATOS OBJETIVO.

Una vez establecido el problema, debemos determinar cuál es la variable objetivo. Los datos pueden existir en bases de datos, documentos, imágenes, etc. Debemos homogeneizar los formatos para poder procesarlos y analizarlos.



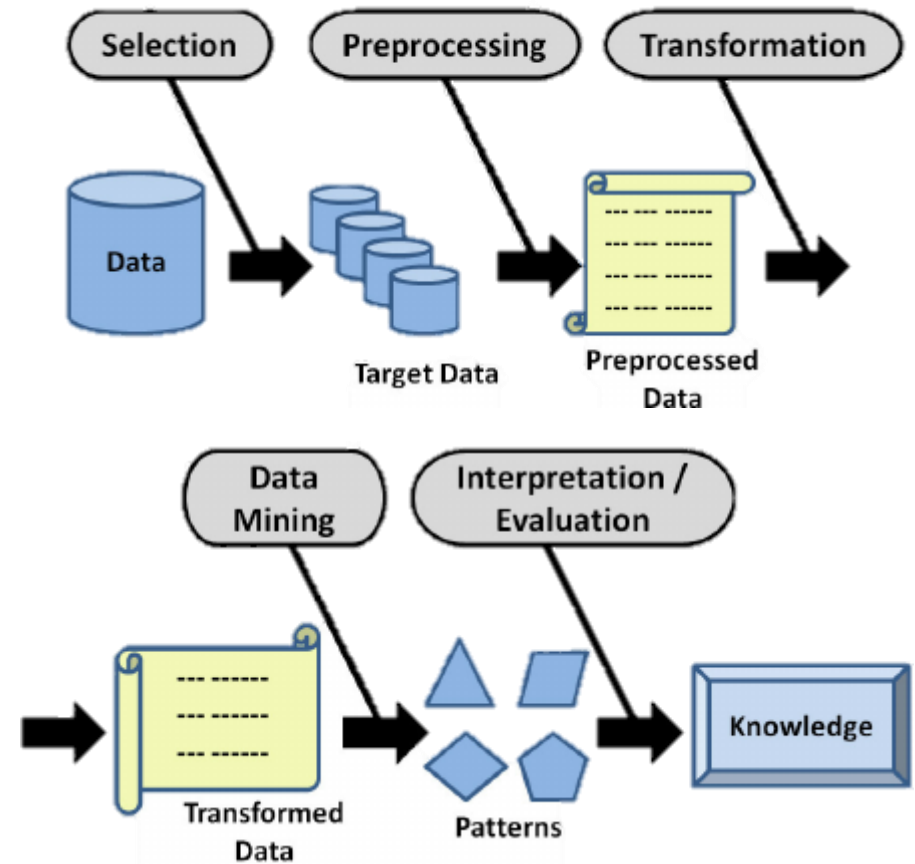
Proceso de Extracción del Conocimiento

Paso 3.

PREPROCESAMIENTO Y LIMPIEZA DE DATOS.

- Eliminación de ruido y datos aislados o *outliers*.
- Eliminar inconsistencias y duplicados.
- Imputar información faltante

El preprocesamiento y limpieza tiene como objetivo mejorar la calidad de los datos y los resultados de la minería.



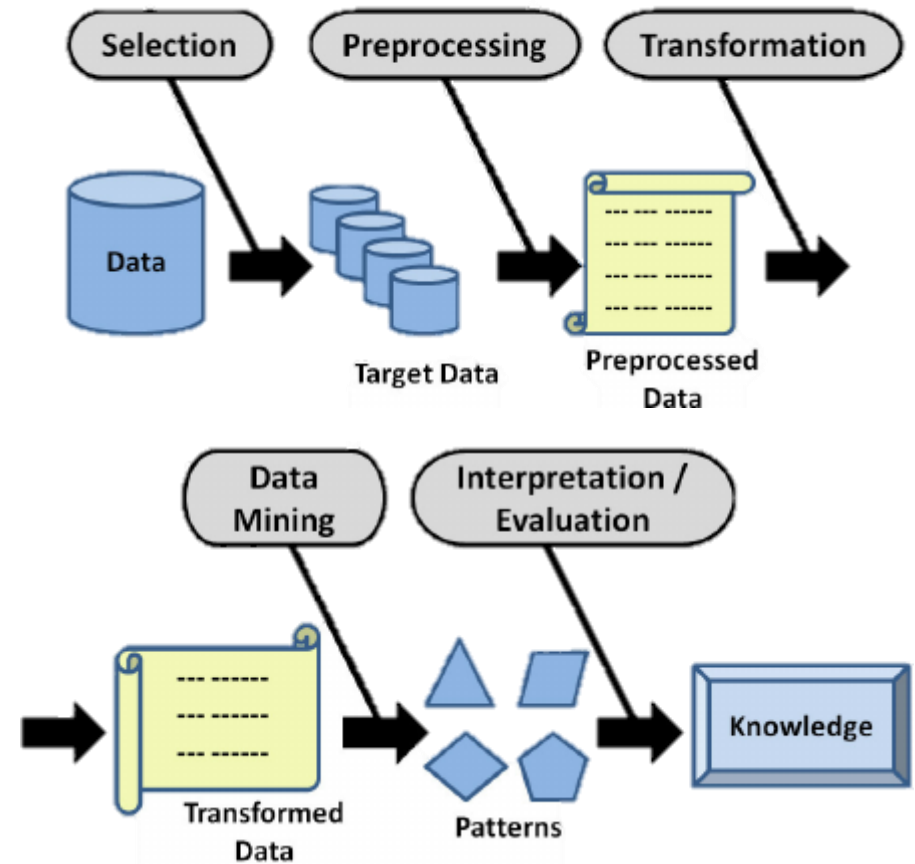
Proceso de Extracción del Conocimiento

Paso 4.

MINERÍA DE DATOS

Haciendo uso de algoritmos, extraemos nueva información de nuestro set de datos. Pasos de la minería de datos:

- Seleccionar la tarea
- Seleccionar el algoritmo/algoritmos a utilizar
- Usar nuestros algoritmos

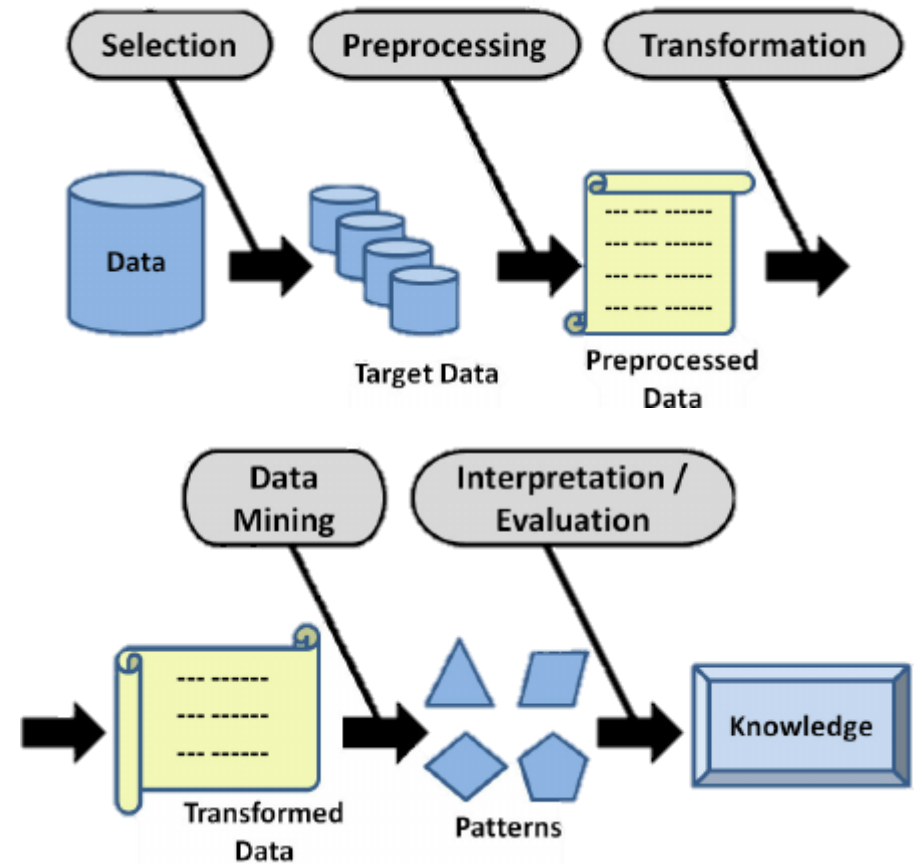


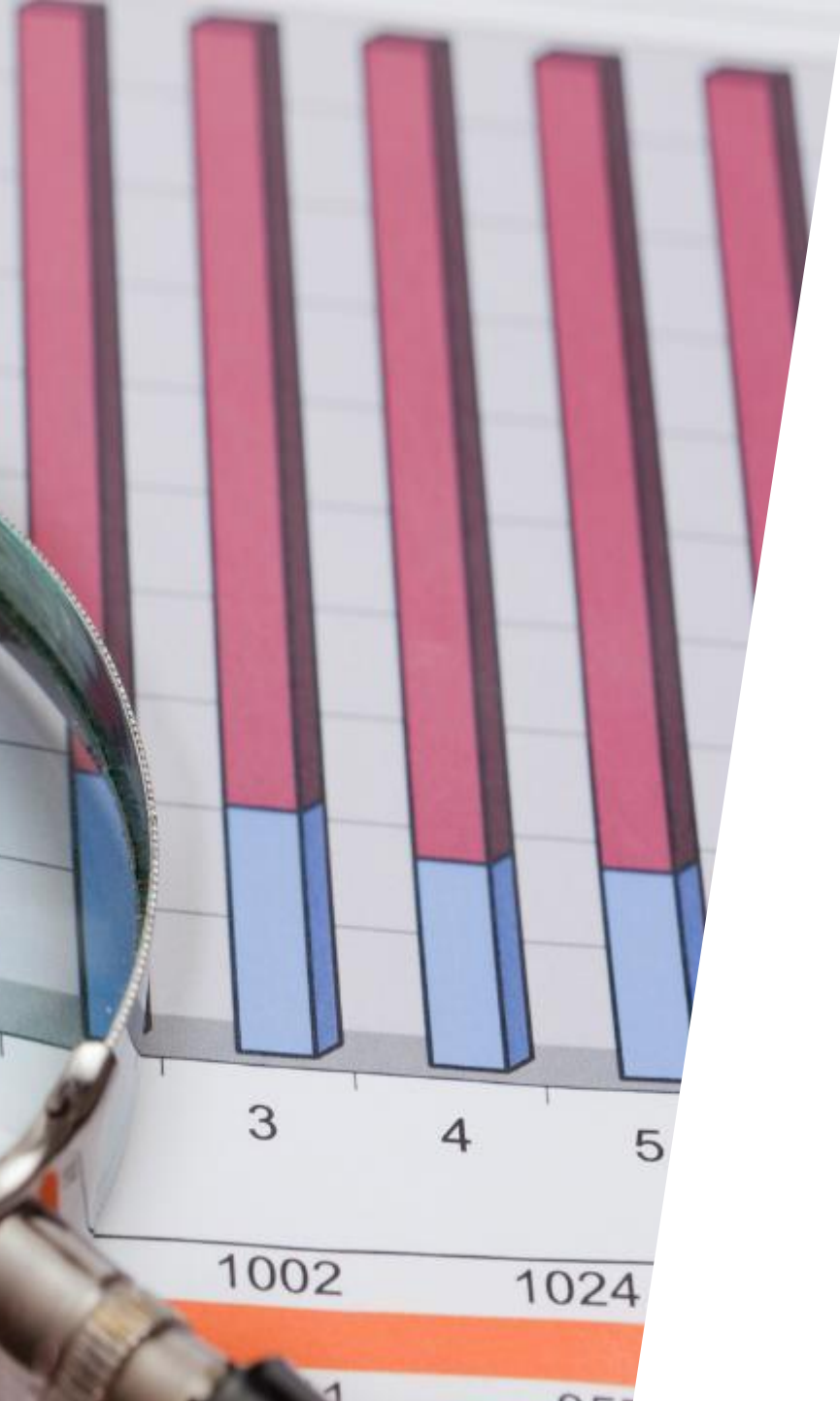
Proceso de Extracción del Conocimiento

Paso 5.

INTERPRETACIÓN DE LOS PATRONES

Esta etapa consiste en la interpretación de los resultados obtenidos; así como la exposición de manera clara de éstos para que sean entendibles. Es muy habitual el uso de técnicas de visualización.





Preprocesamiento de datos

Preprocesamiento de datos

Problemas que pueden presentar los datos

- **Incompletos:** atributos que carecen de valores, atributos sin interés... (*missing values*).
- **Ruidosos:** contienen errores o “*outliers*”.
- **Inconsistentes:** discrepancias en códigos o nombres.
- **Tamaño excesivo:** filas y/o columnas.

Sin datos de calidad, no hay calidad en los resultados

Preprocesamiento de datos

Limpieza de datos

La **limpieza de datos** es el conjunto de operaciones que:

- Corrigen datos erróneos
- Filtran datos incorrectos.
- Reducen un innecesario nivel de detalle.
- Detectan y resuelven discrepancias.

El proceso de limpieza de datos intenta completar los valores perdidos, suavizar el ruido al identificar valores atípicos y corregir inconsistencias en los datos.

Preprocesamiento de datos

Limpieza de datos

La **limpieza de datos** es el conjunto de operaciones que:

- Corrigen datos erróneos
- Filtran datos incorrectos.
- Reducen un innecesario nivel de detalle.
- Detectan y resuelven discrepancias.

El proceso de limpieza de datos incluye la validación y corrección de datos, para obtener datos de calidad.

La **calidad de los datos** se consigue cuando se cumplen:

- **Integridad:** deben cumplir requisitos de entereza y validez.
- **Consistencia:** corrección de contradicciones.
- **Uniformidad:** relacionado con las irregularidades.
- **Densidad:** valores omitidos sobre el número de valores totales.
- **Unicidad:** no tener duplicados ni registros inconsistentes.



Preprocesamiento de datos

Transformación de datos

La **discretización** es el proceso por el cual, se convierten variables continuas en variables categóricas.

Al realizar esta transformación, se pierde información. Sin embargo, hay **algoritmos** (por ejemplo, algoritmos de clasificación) que **necesitan** que los datos de entrada sean **variables categóricas**.

Esto hace que **los datos sean más fáciles de estudiar y analizar**, y **mejora la eficiencia** de las tareas que queramos realizar con ellos.

Preprocesamiento de datos

Normalización de datos

La **normalización** trata de conseguir que todas las variables estén expresadas en una escala similar, para que todas ellas tengan un peso comparable.

Algunos ejemplos habituales:

Normalización Z-score

$$z = \frac{x - \bar{x}}{s}$$

Los datos serán estandarizados siguiendo una distribución Normal(0,1), donde

- media = 0
- desviación típica = 1

Normalización Min-Max

$$x = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Los datos serán escalados en un rango fijo, normalmente entre 0-1.

Preprocesamiento de datos

Missing values

Un **missing value** (valor faltante, perdido o desconocido) es un atributo que no está almacenado. En la librería Pandas de Python, se representan como None y Nan (acrónimo de Not a Number, valor especial de punto flotante).

Tratamiento de *missing values*

- Eliminar filas, cuando la mayoría de los atributos de cierta observación son missing values.
- Eliminar columnas, cuando el atributo no representa valores para la mayoría de las observaciones.
- Imputar su valor, en situaciones intermedias, cuando sea necesario.

Lo más habitual a la hora de imputar *missing values* es rellenarlo según alguna medida de resumen (media, moda, mediana...), lo que puede presentar problemas:

- Reduce la dispersión del atributo.
- Se utiliza menos información de la disponible.

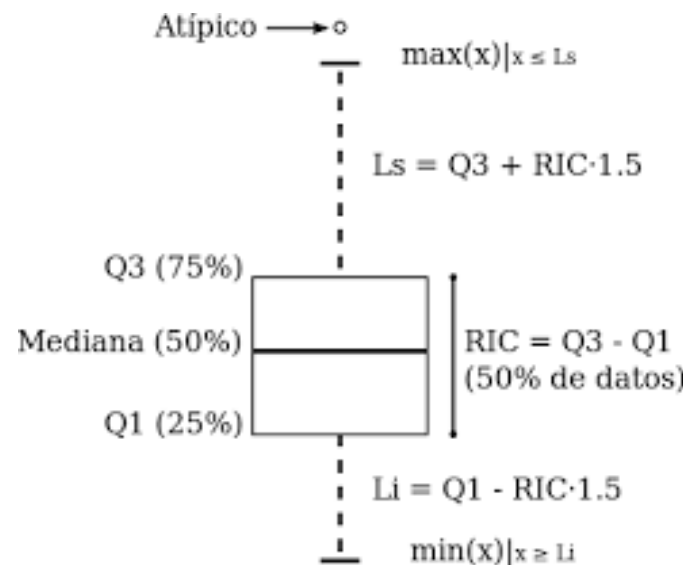
Preprocesamiento de datos

Outliers

Barnet y Lewis (1994) definen **outlier** o valor atípico en un conjunto de datos como una observación (o conjunto de observaciones) que parecen ser inconsistentes con ese conjunto de datos.

Outliers no es igual a error. Los valores atípicos deben ser detectados. Su inclusión o no en el análisis depende del estadístico.

Gráficamente, podemos usar el **diagrama de cajas y bigotes de Tuckey**, conocido como *box-plot*.



Preprocesamiento de datos

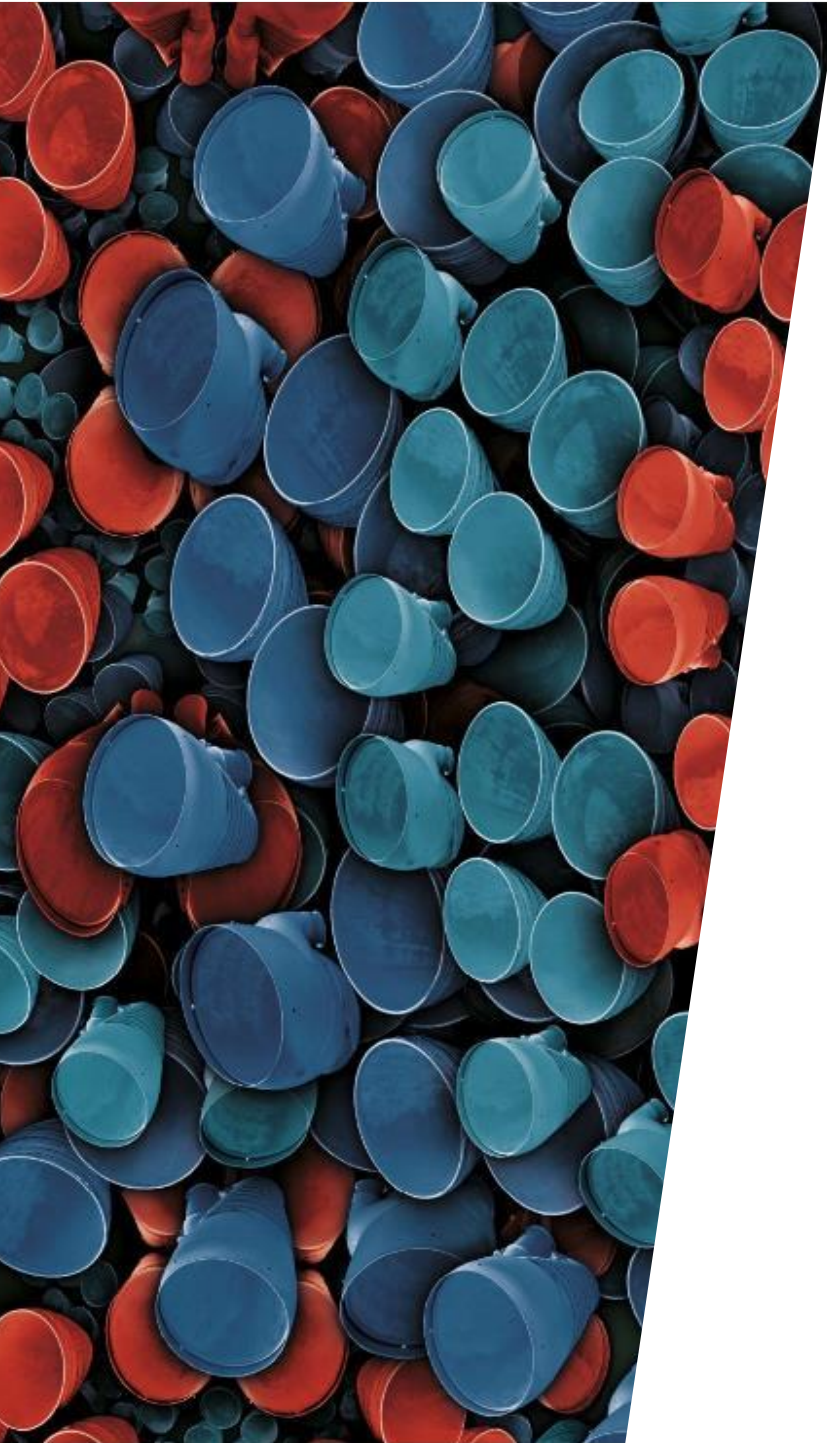
Acciones sobre outliers

- **Ignorar:** algunos algoritmos son robustos a *outliers*.
- **Filtrar, eliminar o reemplazar la columna.**
- **Filtrar la fila:** sesga los datos.
- **Reemplazar el valor:** se debe analizar cuál es el método más adecuado, pudiendo reemplazar por un valor “nulo”, máximo, mínimo, media... e incluso, se puede predecir utilizando alguna técnica de Machine Learning.
- **Discretizar.**

Ejemplo en Python



AnalisisDatosFinancieros_PreprocesadoInformacion.ipynb



APIs

APIs

¿Qué es una API?

- Una API (**Application Programming Interface**) es un conjunto de funciones (o métodos) y procedimientos que ofrece un componente software, mostrando tanto las entradas y salidas que emplea como las operaciones que realiza.
- Hablamos de **APIs web** cuando aplicamos este concepto a la web. Es decir, la interfaz de programación es expuesta al exterior para que pueda ser empleada mediante peticiones y respuestas web (peticiones HTTP como las que hace un navegador web).
- Este intercambio de información se hace normalmente en formato **XML** o **JSON**.
- Las APIs web pueden servir para:
 - Realizar una determinada operación o cálculo.
 - Exponer un set de datos al mundo.

APIs

Elementos y partes de una web API.

- **URL base:** es la dirección (URL) donde se encuentra desplegada la API. Normalmente tienen la siguiente forma: <https://api.twitter.com>, <https://api.github.com>
- **Endpoint:** cada función o recurso de la API tiene una URL diferente, que empieza siempre por la URL base. Cada una de estas funciones o recursos se le conoce con el nombre de *endpoint*. La documentación oficial de la API tiene información sobre los diferentes *endpoints*.
- **API reference:** es el nombre que recibe la documentación oficial de la API. Está compuesta por la descripción de los servicios así como por ejemplos de *requests* and *responses*. También suele contar con una consola para simular peticiones a la API.
- **Requests:** peticiones/llamadas/invocaciones a la API. Contiene los parámetros, cabeceras, método usado, URL del *endpoint*, etc.
- **Responses:** información devuelta por el servidor. Incluye los datos, cabeceras, código de retorno, etc.

APIs

Elementos y partes de una web API.

- **Códigos de retorno (*Status code*):** contienen información sobre el resultado de la petición y nos indican si la petición fue satisfactoria. Algunos ejemplos: 200 (OK), 400 (*Badrequest*). 401 (*Unauthorized*), 404 (*NotFound*), 405 (*Methodnotallowed*), 500 (*Internalserver error*).
- **Cabeceras (*Headers*):** parámetros especiales para configurar las *requests* y *responses*. Algunos ejemplos: Accept (tipo de contenido que acepta el cliente), Content-type (tipo de contenido que devolverá el servidor), Authentication (contiene las credenciales).
- **HTTP *methods*:** también llamados verbos, especifican la acción que se desea realizar. Algunos ejemplos: POST (crear), GET (leer), PUT (actualizar), DELETE (borrar). Normalmente usaremos POST y GET para obtener los datos.
- **Query *parameters*:** colección de parámetros necesarios para realizar la acción deseada y que se pasan al *endpoint*.

APIs

Elementos y partes de una web API.

- **Autenticación (*Authentication*):** sistema de identificación y de acceso a la API. Normalmente sirve de control de que se tiene permisos para usar la API. Hay diferentes métodos: API Key, OAuth, OAuth2.
- **Paginación:** en muchas ocasiones las API devuelven gran cantidad de datos. En estas ocasiones se parten los datos en porciones más pequeñas de tal modo que el servidor los puede servir de manera óptima. Se usan parámetros para establecer: resultados por página, número de página y número total de resultados.
- **Ratelimit:** límite de peticiones (*requests*) que puede realizar un usuario en un periodo de tiempo. Por ejemplo: peticiones por segundo, peticiones al mes, etc. De este modo se controla el uso de la API, impidiendo a los usuarios hacer un mal uso de la misma.

APIs

Formato XML

- XML (**eXtensible Markup Language**) es un lenguaje de marcas empleado para almacenar datos de forma legible. Tiene una estructura jerárquica.
- Similar a HTML.
- Sirve para intercambio de información.
- Partes del documento XML:
 - **Declaración:** donde se establece la versión y la codificación del documento. También es posible declarar el tipo de documento (DTD y XML Schema).
 - **Tags:** aparecen entre los símbolos < y >.
 - *start-tags*: <section>
 - *end-tags*: </section>
 - *empty-element-tags*: <section/>
- **Elementos:** componentes del documento XML. Comienzan por un *start-tag* y terminan por un *end-tag*. Entre los tags aparece el contenido del elemento (puede contener otros elementos a su vez).
- **Atributos:** características o propiedades de los elementos (pares clave/valor). Los valores van entrecomillados (con comillas dobles).

APIs

Ejemplo formato XML

```
1 <persona>
2   <nombres>Elsa</nombres>
3   <apellidos>Zambrano</apellidos>
4   <fecha-de-nacimiento>
5     <día>18</día>
6     <mes>6</mes>
7     <año>1996</año>
8   </fecha-de-nacimiento>
9   <ciudad>Pamplona</ciudad>
10</persona>
```


APIs

Formato JSON

JSON es un lenguaje para **representar datos**, con la sintaxis propia de los literales de objetos Javascript. Sus usos son varios, pero principalmente:

- Compartir y transferir información entre distintos sistemas de software.
- Almacenar información en sistemas de persistencia.

Los archivos JSON pueden tener cualquier nivel de **anidación** que sea necesario, ya que podemos colocar unos objetos dentro de otros, creando un árbol de datos de cualquier profundidad.

Todos los archivos de JSON representa una de estas dos **estructuras**:

- Lista de pares clave/valor.
- Colección de elementos, que se conoce como array o arreglo.

APIs

Ejemplo JSON

```
{ "empinfo" :  
  {  
    "employees" : [  
      {  
        "name" : "Scott Philip",  
        "salary" : f44k,  
        "age" : 27,  
      },  
      {  
        "name" : "Tim Henn",  
        "salary" : f40k,  
        "age" : 27,  
      },  
      {  
        "name" : "Long Yong",  
        "salary" : f40k,  
        "age" : 28,  
      }  
    ]  
  }  
}
```

APIs

Ejemplos de APIs



The New York Times



yahoo!
finance

Ejemplo en Python



AnalisisDatosFinancieros2023_API_YahooFinance.ipnyb



Afi Escuela

© 2023 Afi Escuela. Todos los derechos reservados.