

DATA SCIENCE AND MACHINE LEARNING APPLIED TO FINANCIAL MARKETS

Modulo III

CIENCIA DE DATOS

Profesor: Act. Fernando Ortega
Camargo

Objetivo

Los participantes examinarán las herramientas y las ideas esenciales de la ciencia de datos como análisis previo a aplicaciones más rigurosas. Se pondrán de relieve las aplicaciones de la ciencia de datos para explorar datos financieros desde una nueva perspectiva.



Temario

1 BASES DE DATOS Y SQL

1.1.Introducción a las bases de datos con MySQL

1.2.Relación entre tablas: llaves primarias y foráneas

1.3.Creación de una base de datos

1.4.Consultas en una base de datos

1.5.Python – SQL

1.5.1. SQLAlchemy

2 PYTHON CIENTÍFICO

2.1.Numpy: arreglos matriciales

2.2.Gráficas científicas con Matplotlib

2.3.Herramientas científicas con Scipy

Temario

3 INTRODUCCIÓN AL ANÁLISIS DE DATOS

3.1.DataFrames con Pandas

3.2.Manipulación de información

3.2.1. Desde una base de datos

3.2.2. CSV

3.2.3. JSON

3.3.Creación de nuevas variables

3.4.Limpieza y validación de información

3.5.Webscrapping

3.5.1. Lenguaje html, CSS y el DOM

3.5.2. Request, Get y Post

3.5.3. Acopio de información

3.5.4. Limpieza de información

3.5.5. Extracción de información de textos

3.5.6. Selenium

Temario

4 ANÁLISIS DE DATOS FINANCIEROS

4.1.Sentimiento de mercado

4.2.Análisis de tendencias en distintos subyacentes

5 ANÁLISIS Y CONSTRUCCIÓN DE PORTAFOLIOS

5.1.Medición del desempeño

5.2.Factores explicativos del desempeño

6 ANÁLISIS DEL LENGUAJE NATURAL

6.1.Sentimiento de mercado

6.1.1. Redes sociales

6.1.2. Noticias

6.1.3. Expresiones regulares (regex)

6.1.4. N-Grams

6.1.5. Naive Bayes

Temario

7 DETECCIÓN DE OPORTUNIDADES DE INVERSIÓN

7.1.Estrategias de valor relativo

7.1.1. Curvas de tasas de interés

7.1.2. Volatilidad

7.2.Robo-advisor

8 TRADING ALGORÍTMICO

8.1.Costos de transacción

8.2.Cobertura e impacto en liquidez

8.3.Ejecución de órdenes

8.4.Trading de pares

8.5.Reversión a la media

Temario

7 DETECCIÓN DE OPORTUNIDADES DE INVERSIÓN

7.1.Estrategias de valor relativo

7.1.1. Curvas de tasas de interés

7.1.2. Volatilidad

7.2.Robo-advisor

8 TRADING ALGORÍTMICO

8.1.Costos de transacción

8.2.Cobertura e impacto en liquidez

8.3.Ejecución de órdenes

8.4.Trading de pares

8.5.Reversión a la media

Tema 1

BASES DE DATOS Y SQL

¿Para qué sirven las bases de datos?

Las bases de datos son arquitecturas informáticas que sirven para almacenar información y para que los usuarios de esta puedan interactuar con ella, ya sea para recuperarla y/o actualizarla. Esta información puede ser de cualquier tipo y de cualquier tema mientras sea significativo para el individuo u organización que la requiera.



Las bases de datos son de vital importancia en casi todas las áreas donde la informática sea requerida como por ejemplo, negocios de correo electrónico, medicina, genética, educación, ciencia, finanzas, administración etc.

Ventajas

Compacta: Su almacenamiento no requiere tantos recursos físicos como es el papel.

Rapidez: La máquina puede recuperar y actualizar los datos más rápido que un humano.

Menos trabajo: Eliminan el proceso de administrar archivos manualmente(automatización)

Frecuencia: La información está disponible en cualquier momento.

Protección: Los datos pueden estar mejor protegidos contra la pérdida no intencional y algún problema de seguridad .



Desventajas

Alta complejidad: Administrar una base de datos conlleva un grupo de programas complejos los cuales deben ser comprendidos en su totalidad para una correcta funcionalidad.

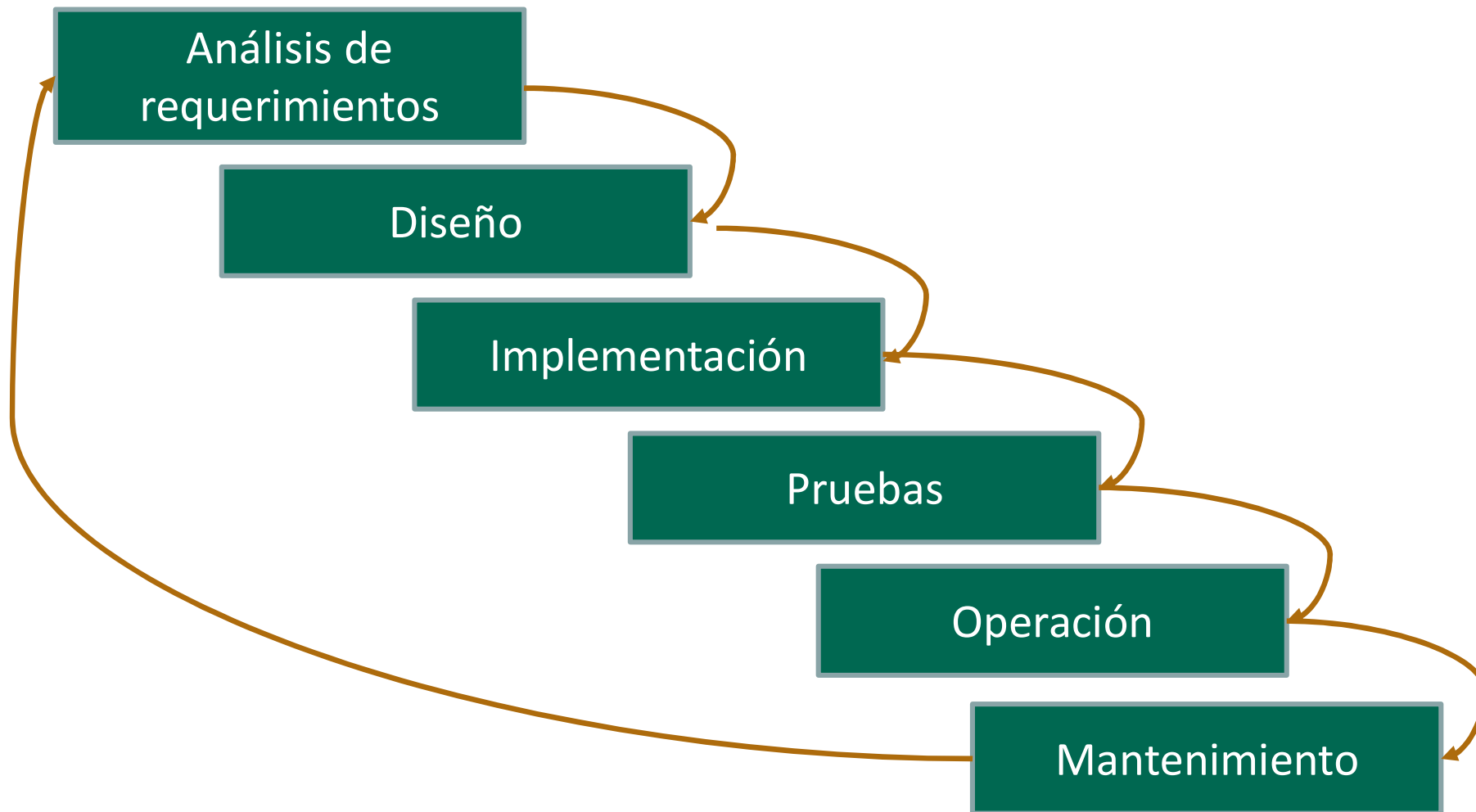
Gran tamaño: Se requiere infraestructura capaz de soportar una gran cantidad de espacio en disco y memoria para optimizar los procesos.

Costo: El costo de implementación normalmente depende del entorno y funcionalidad de la misma, incluyendo mantenimientos periódicos, con lo que los costos de implementación se pueden incrementar notoriamente.

Vulnerabilidad: el hecho de que todos los datos estén centralizados en una sola arquitectura hace que el sistema sea más vulnerable ante los fallos o actividades ilícitas.



Ciclo de vida de una base



Conceptos básicos

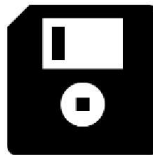
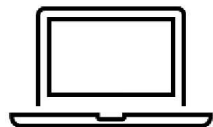
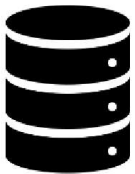
-Dato

-Información

-Sistema de información

-Base de datos

-Sistemas manejadores de bases de datos



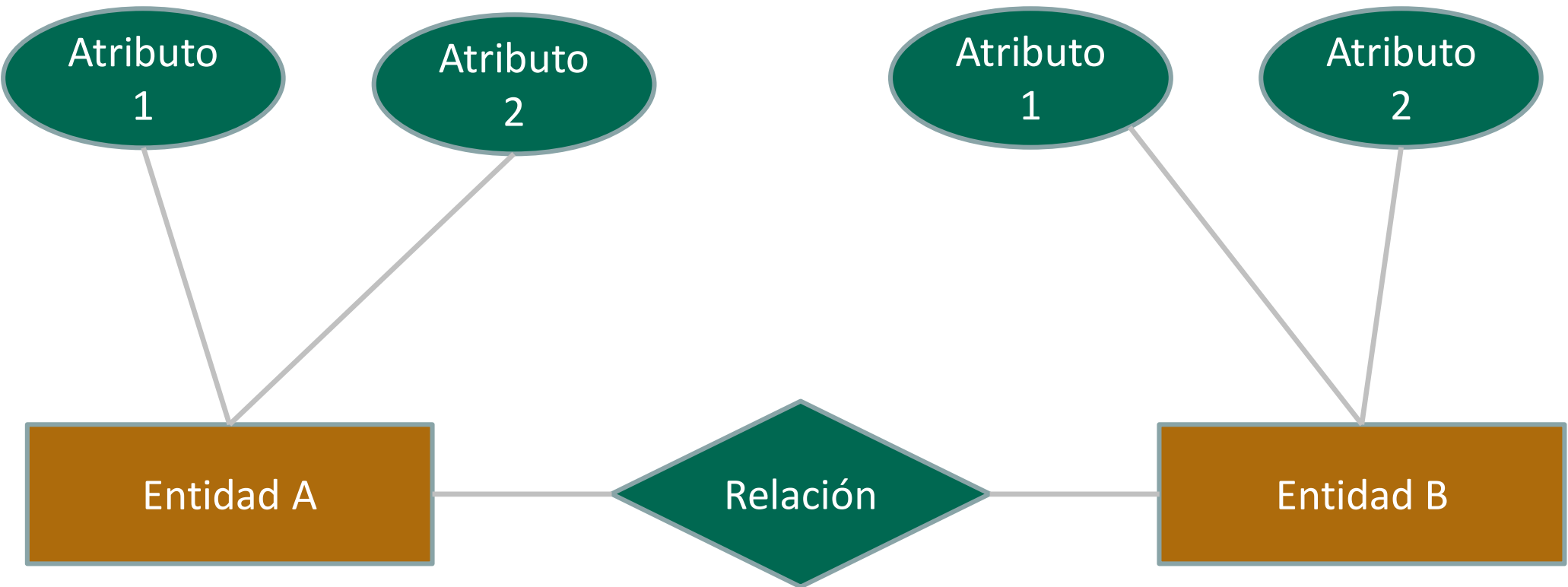
Entidad

Relación

Modelo entidad relación

Atributo

Modelo Entidad Relación



Mapeo y tuplas

Tabla Entidad

Er.tidad 1		
Atributo 1	Tipo de atributo1	Restricción A1
Atributo 2	Tipo de atributo2	Restricción A2
Atributo 3	Tipo de atributo3	Restricción A3
Atributo 4	Tipo de atributo4	Restricción A4
Atributo 5	Tipo de atributo5	Restricción A5

Tupla Entidad

1

Entidad 1			
Elemento	Atributo 1	Atributo 2	Atributo 3
1	Atributo 1 E1	Atributo 2 E1	Atributo 3 E1
2	Atributo 1 E2	Atributo 2 E2	Atributo 3 E2
3	Atributo 1 E3	Atributo 2 E3	Atributo 3 E3
4	Atributo 1 E4	Atributo 2 E4	Atributo 3 E4
5	Atributo 1 E5	Atributo 2 E5	Atributo 3 E5

GROUP BY

CREATE
DATABASE

UPDATE

Lenguaje de definición y manipulación

WHERE

CREATE TABLE

INSERT
INTO

SELECT
FROM

DELETE

Lenguaje de Definición de Datos (DDL) básico

**Creación de base
de datos**

```
CREATE DATABASE ejemplo1;
```

**Creación de
tablas**

```
CREATE TABLE mi_primera_tabla(  
  primera_columna numeric,  
  segunda_columna text);
```

Lenguaje de Definición de Datos (DDL) integridad

**Llaves
primarias**

```
CREATE TABLE tabla1(  
  columna1 integer,  
  columna2 varchar(50),  
  PRIMARY KEY (columna1));
```

Lenguaje de Definición de Datos (DDL)

integridad

Llaves foraneas

```
CREATE TABLE tabla2(  
  columna3 integer,  
  columna4 integer,  
  columna5 varchar(15),  
  FOREIGN KEY(columna4) REFERENCES tabla_1(columna1));
```

Dominio

```
CREATE TABLE Cliente(  
  ID_cliente integer,  
  Apellidos varchar(255),  
  Nombre varchar(255),  
  Genero char(1),  
  CHECK(Genero IN ('M', 'F')));
```

Existen otros casos de condición como: (A = B), (C <> D), (E NOT IN ('E', 'F', ..., 'X')), (X < 0), etc

Lenguaje de Definición de Datos (DDL)

integridad

No nulidad

```
CREATE TABLE Cliente1(  
  ID_Cliente INTEGER NOT NULL,  
  Apellidos VARCHAR(255) NOT NULL,  
  Nombre VARCHAR(255) NOT NULL);
```

Modificación de
restricciones

```
ALTER TABLE nombre_tabla  
DROP CONSTRAINT nombre_de_la_cláusula;  
ALTER TABLE nombre_tabla  
ADD CHECK (nombre_columna IN ('M', 'F'));
```

Lenguaje de manipulación de Datos (DML)

**Inserción de
datos**

```
INSERT INTO nombre_tabla (columna1, columna2, ..., columnaN)  
VALUES (valor1, valor2, ..., valorN);
```

**Consulta
general**

```
SELECT nombre_columna  
FROM nombre_tabla  
WHERE condicion;
```

Actualización

```
UPDATE nombre_tabla  
SET columna_1 = nuevo_valor  
WHERE condición;
```

Actualización

```
DELETE FROM nombre_tabla  
WHERE condicion;
```

Lenguaje de manipulación de Datos (DML) avanzado

Consultas agrupadas

```
SELECT nombre_columna1, nombre_columna2, ...  
FROM nombre_tabla  
WHERE condición GROUP BY atributo_agrupador  
HAVING condición_de_agrupación;
```

Funciones de agregación

```
SELECT Fecha, SUM(Total)  
FROM Orden  
GROUP BY Fecha;
```

Operadores y funciones útiles

BETWEEN Compara si un valor se encuentra dentro de los parámetros que el usuario puede definir

LIKE Compara si un valor cumple con un patrón que el usuario puede definir

SIMILAR TO Compara si un valor cumple con un patrón que el usuario puede definir de manera más general

IN Compara si un valor existe dentro de una lista de posibles valores definida por el usuario

NOT IN Compara si un valor no existe dentro de una lista de posibles valores definida por el usuario

And y Or Operadores lógicos

= Compara si dos valores son iguales

< Compara si el valor de la izquierda es menor que el de la derecha

> Compara si el valor de la izquierda es mayor que el de la derecha

< > Compara si dos valores son diferentes

Tipos de datos en SQL

Grupo	Tipo de dato	Intervalo	Almacenamiento
Numéricos exactos	bigint	De -2^{63} (-9.223.372.036.854.775.808) a $2^{63} - 1$ (9.223.372.036.854.775.807)	8 bytes
	int	De -2^{31} (-2.147.483.648) a $2^{31} - 1$ (2.147.483.647)	4 bytes
	smallint	De -2^{15} (-32.768) a $2^{15} - 1$ (32.767)	2 bytes
	tinyint	De 0 a 255	1 byte
	bit	Tipo de datos entero que puede aceptar los valores 1, 0 ó NULL	2 bytes
	decimal, numeric, decimal (p, s)	<ul style="list-style-type: none"> p (precisión): el número total máximo de dígitos decimales que se puede almacenar, tanto a la izquierda como a la derecha del separador decimal. La precisión debe ser un valor comprendido entre 1 y la precisión máxima de 38. La precisión predeterminada es 18. s (escala): el número máximo de dígitos decimales que se puede almacenar a la derecha del separador decimal. La escala debe ser un valor comprendido entre 0 y p. Sólo es posible especificar la escala si se ha especificado la precisión. La escala predeterminada es 0. <p>Con precisión máxima $10^{38} + 1$ y $10^{38} - 1$</p>	Precisión 1 - 9: 5 bytes
	money	Tipos de datos que representan valores monetarios o de moneda: de -922.337.203.685,4775808 a 922.337.203.685,4775807	8 bytes
	smallmoney	De - 214,7483648 a 214,7483647	4 bytes

Tipos de datos en SQL

Numéricos aproximados	float	De - 1,79E+308 a -2,23E-308, 0 y de 2,23E-308 a 1,79E+308	Depende del valor de n
	real	De - 3,40E + 38 a -1,18E - 38, 0 y de 1,18E - 38 a 3,40E + 38	4 Bytes
Fecha y hora	datetime	Del 1 de enero de 1753 hasta el 31 de diciembre de 9999	
	smalldatetime	Del 1 de enero de 1900 hasta el 6 de junio de 2079	
Cadenas de caracteres	char (n)	Caracteres no Unicode de longitud fija, con una longitud de n bytes. n debe ser un valor entre 1 y 8.000	n bytes
	varchar (n)	Caracteres no Unicode de longitud variable. n indica que el tamaño de almacenamiento máximo es de $2^{31} - 1$ bytes	n bytes (aprox.)
	text	En desuso, sustituido por <i>varchar</i> . Datos no Unicode de longitud variable con una longitud máxima de $2^{31} - 1$ (2.147.483.647) caracteres	max bytes (aprox.)
Cadenas de caracteres unicode	nchar (n)	Datos de carácter Unicode de longitud fija, con n caracteres. n debe estar comprendido entre 1 y 4.000	$2 * n$ bytes
	nvarchar (n)	Datos de carácter Unicode de longitud variable. n indica que el tamaño máximo de almacenamiento es $2^{31} - 1$ bytes	$2 * n$ bytes + 2 bytes
	ntext (n)	En desuso, sustituido por <i>nvarchar</i> . Datos Unicode de longitud variable con una longitud máxima de $2^{30} - 1$ (1.073.741.823) caracteres	$2 * n$ bytes

Tipos de datos en SQL

Cadenas binarias	binary (n)	Datos binarios de longitud fija con una longitud de n bytes, donde n es un valor que oscila entre 1 y 8.000	n bytes
	varbinary (n)	Datos binarios de longitud variable. n indica que el tamaño de almacenamiento máximo es de $2^{31} - 1$ bytes	n bytes
	image	En desuso, sustituido por <i>varbinary</i> . Datos binarios de longitud variable desde 0 hasta $2^{31} - 1$ (2.147.483.647) bytes	
Otros tipos de datos	cursor	Tipo de datos para las variables o para los parámetros de resultado de los procedimientos almacenados que contiene una referencia a un cursor. Las variables creadas con el tipo de datos <i>cursor</i> aceptan NULL	
	timestamp	Tipo de datos que expone números binarios únicos generados automáticamente en una base de datos. El tipo de datos <i>timestamp</i> es simplemente un número que se incrementa y no conserva una fecha o una hora	8 bytes
	sql_variant	Tipo de datos que almacena valores de varios tipos de datos aceptados en SQL Server, excepto <i>text</i> , <i>ntext</i> , <i>image</i> , <i>timestamp</i> y <i>sql_variant</i>	
	uniqueidentifier	Es un GUID (Globally Unique Identifier, Identificador Único Global)	16 bytes
	table	Es un tipo de datos especial que se puede utilizar para almacenar un conjunto de resultados para su procesamiento posterior. <i>table</i> se utiliza principalmente para el almacenamiento temporal de un conjunto de filas devuelto como el conjunto de resultados de una función con valores de tabla	
	xml	Almacena datos de XML. Puede almacenar instancias de xml en una columna o una variable de tipo xml	

Tema 2

PYTHON CIENTÍFICO

NumPy

NumPy es el paquete fundamental para la computación científica en Python. Es una biblioteca de Python que proporciona un objeto de matriz multidimensional, varios objetos derivados (como matrices y matrices enmascaradas) y una variedad de rutinas para operaciones rápidas en matrices, incluida la manipulación matemática, lógica, de formas, clasificación, selección, transformadas discretas de Fourier, álgebra lineal básica, operaciones estadísticas básicas, simulación aleatoria y mucho más.



Documentación Completa:
<https://numpy.org/>

Matplotlib

Matplotlib es una biblioteca completa para crear visualizaciones estáticas, animadas e interactivas en Python.

Entre sus posibilidades destacan:

- Creación de gráficos de calidad de publicación.
- figuras interactivas que puedan hacer zoom, desplazarse, actualizar.
- Personalización en diseño y el estilo visual.
- Posibilidad de exportación a muchos formatos de archivo.
- Compatibilidad con JupyterLab e interfaces gráficas de usuario.
- Amplia variedad de paquetes de terceros creados en Matplotlib.



Documentación Completa:

<https://matplotlib.org/>

SciPy

SciPy es una colección de algoritmos matemáticos y funciones de conveniencia creadas en la extensión NumPy de Python. Agrega un poder significativo a la sesión interactiva de Python al proporcionar al usuario comandos y clases de alto nivel para manipular y visualizar datos. Con SciPy, una sesión interactiva de Python se convierte en un entorno de procesamiento de datos y creación de prototipos de sistemas que rivaliza con los sistemas, como MATLAB, IDL, Octave, R-Lab y SciLab.



Documentación Completa:
<https://scipy.org/>

Tema 3

Introducción al análisis de datos

Pandas

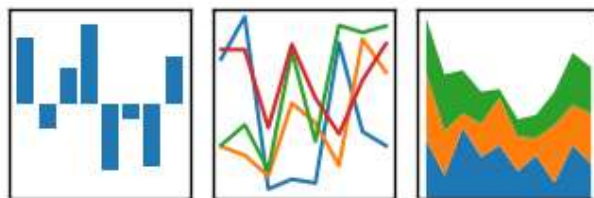
Pandas es una librería de Python especializada en el manejo y análisis de estructuras de datos multidimensionales.

Las principales características de pandas son:

- Define nuevas estructuras de datos basadas en los arrays de la librería NumPy (pandas, series).
- Permite leer y escribir ficheros en formato CSV, Excel y bases de datos SQL así como conexiones a las mismas.
- Permite acceder a los datos mediante índices o nombres para filas y columnas, parecido a como se hace en las hojas de calculo.
- Ofrece métodos para reordenar, dividir y combinar conjuntos de datos.
- Permite trabajar con series temporales.
- Realiza todas estas operaciones de manera muy eficiente.

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



Documentación Completa:
<https://pandas.pydata.org/>

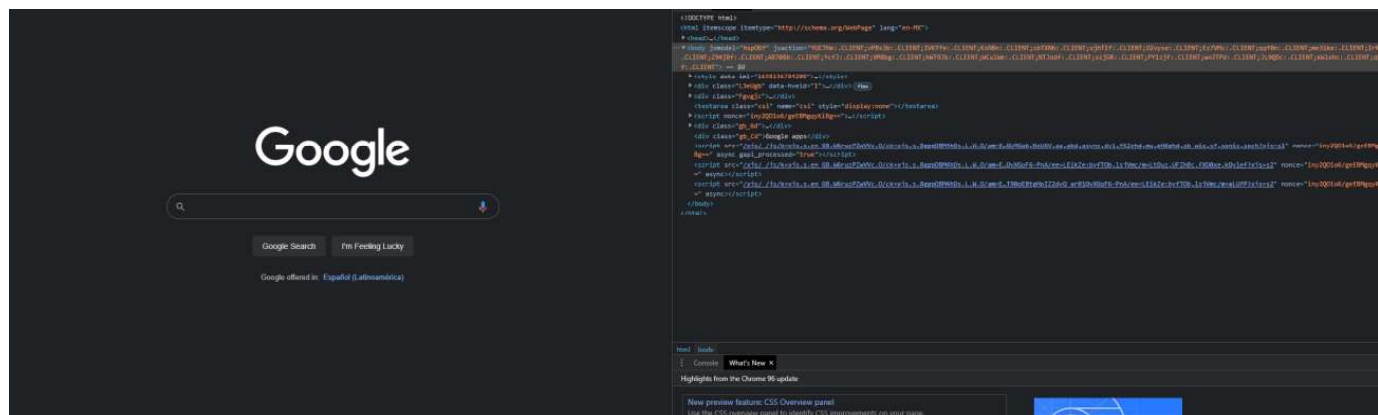
Tema 3.5

Webscrapping

Introducción a html

HTML (Lenguaje de Marcas de Hipertexto, HyperText Markup Language) es el componente más básico de la Web, es un lenguaje de marcado y es el más utilizado en todos los sitios web de la actualidad. Define el significado y la estructura del contenido web mediante etiquetas que servirán para la comunicación, formato y método de transferencia de información entre los distintos servidores.

Todas las paginas web que se conocen funcionan mediante un código html el cual puede ser accedido desde los navegadores predeterminados, por ejemplo , en el caso de Google Chrome, presionando click derecho y la opción de inspeccionar se puede analizar la estructura del html actual.



Introducción a html

Html funciona mediante etiquetas las cuales definirán la estructura de nuestra pag web, estas etiquetas harán referencia a títulos, párrafos, imágenes, hipervínculos dependiendo de la sintaxis que se utilice, de igual manera estas etiquetas definen el contenido.

El formato se dará mediante otra sintaxis que veremos mas adelante la cual tiene por nombre CSS (Cascading Style Sheets).

```
><style data-impl="1638136893753">...</style>
><div class="L3eUgb" data-hveid="1"> flex
  ><div class="o3j99 n1xJcf Ne6n5d">...</div> flex
... ><div class="o3j99 LLD4me yr19Zb LS80J">...</div> flex == $0
  ><div class="o3j99 ikrT4e om7nvf">...</div>
  ><div class="o3j99 qarstb">...</div>
  ><div class="o3j99 c93Gbe">...</div>
```

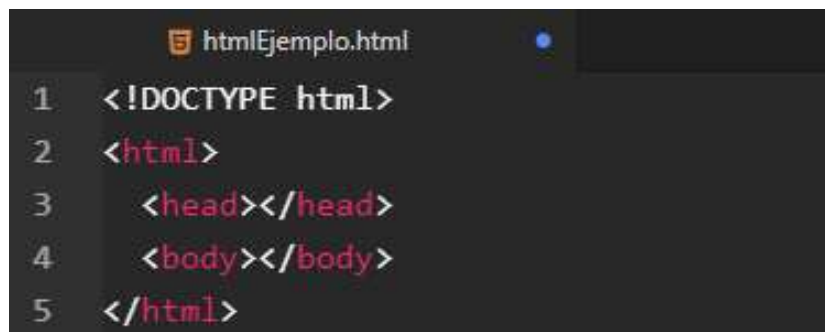
Por lo general la estructura básica de las etiquetas es la misma, mediante (< ó >) indicamos el inicio de una etiqueta, dentro de esto especificaremos distintos atributos, como pueden ser id's, clases, hipervínculos, etc), estos elementos estarán referenciados al css con lo que los elementos de una misma clase compartirán ciertos atributos de formato.

Introducción a html

Para empezar a escribir html, basta con escribir el código en cualquier editor de texto y simplemente guardarlo con la terminación .html , haciendo esto podemos empezar a crear una pagina web sencilla de manera local.

Es recomendable escribir el html en algún editor de código que tenga la sintaxis predefinida con lo que nos ahorrara trabajo de estar recordando todas las etiquetas posibles

Primeras etiquetas



```
1 <!DOCTYPE html>
2 <html>
3   <head></head>
4   <body></body>
5 </html>
```

El inicio de nuestros códigos html siempre tendrá este formato, iniciamos nuestro código con html con la etiqueta de apertura (<>) y cierre (</>)

Dentro de las etiquetas especificaremos el contenido de la misma, hay que notar que el contenido de la pagina web siempre estará en las etiquetas de body, pero esto no implica que en otras etiquetas como head, no podamos especificar algo, simplemente en ellas no se verá reflejado el contenido en la pagina web pero si realizarán ciertas funciones, por ejemplo, el apartado de head dará el titulo a nuestra pagina, título que no estará plasmado internamente en el contenido de la pag web final.

```
<!DOCTYPE html>
<html>
  <head> Titulo que no aparece en la pag</head>
  <body> Contenido que si aparece en la web </body>
</html>
```

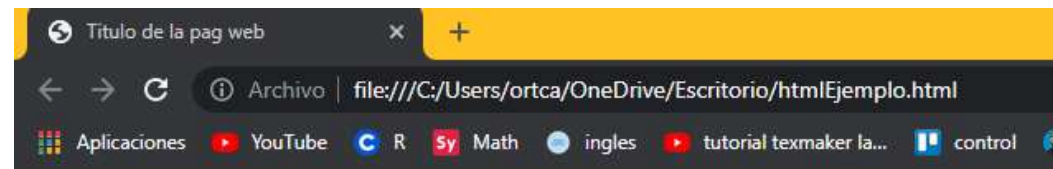
```
<!DOCTYPE html>
<html>
  <head> <title>Titulo que no aparece en la pag</title></head>
  <body> Contenido que si aparece en la web </body>
</html>
```

Títulos

Se especificarán mediante la etiqueta h y la numeración de orden

`<h1>Titulo 1</h1>`

```
<!DOCTYPE html>
<html>
  <head>
    <title>Titulo de la pag web</title>
  </head>
  <body>
    <h1>Titulo 1</h1>
    <h2>Titulo 2</h2>
    <h3>Titulo 3</h3>
  </body>
</html>
```



Titulo 1

Titulo 2

Titulo 3

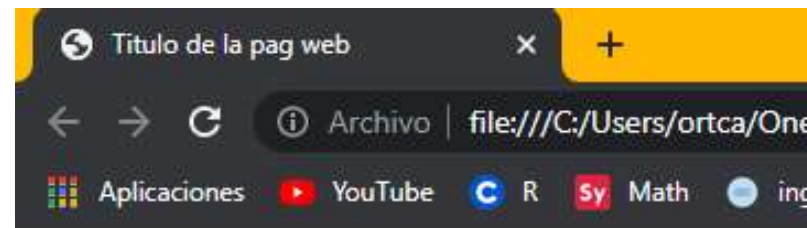
Párrafos

Se especificarán mediante la etiqueta p

<p>Párrafos </p>

```
<!DOCTYPE html>
<html>
  <head>
    <title>Titulo de la pag web</title>
  </head>
  <body>
    <p> Hola, soy un parrafo</p>
    <p>Hola, soy otro parrafo</p>
  </body>
</html>
```

Cada que escriba un párrafo diferente este hará un salto de línea, si yo quisiera un salto de línea dentro de un mismo párrafo utilizare la etiqueta
 </br>



Hola, soy un parrafo

Hola, soy otro parrafo

Comentarios

Se utilizara dentro de una etiqueta `<!-- comentario -->`

```
<html>
  <head>
    <title>Titulo de la pag web</title>
  </head>
  <body>
    <p> Hola, soy un parrafo</p>
    <p>Hola, soy otro parrafo</p>
    <!-- <p>Esto es un comentario </p> -->
  </body>
</html>
```

Teniendo este formato estándar en html, dentro de títulos o párrafos se pueden modificar algunos elementos de los textos, por ejemplo, tamaños, fuentes, etc.

Veremos a resumen estos elementos.

Formatos de fuentes

```
<!DOCTYPE html>
<html>
  <head>
    <title>Titulo de la pag web</title>
  </head>
  <body>
    <p> <b>Hola, soy un parrafo en negrita</b></p>
    <p><code>Hola, soy otro parrafo con formato código</code></p>
    <p><em>Hola soy un parrafo en cursiva</em></p>
    <p><s>Hola soy un parrafo tachado</s></p>
    <p><small>Hola soy un parrafo pequeño</small></p>
    <p><strong>Hola soy un parrafo en negrita</strong></p>
    <p><u>Hola soy un parrafo subrayado</u></p>
    <!-- <p>Esto es un comentario </p> -->
  </body>
</html>
```

Formatos de fuentes



Hola, soy un parrafo en negrita

Hola, soy otro parrafo con formato código

Hola soy un parrafo en cursiva

~~Hola soy un parrafo tachado~~

Hola soy un parrafo pequeño

Hola soy un parrafo en negrita

Hola soy un parrafo subrayado

Divisiones o secciones

Estas divisiones son como “cajas” las cuales estarán en un bloque que no servirá para dividir nuestro código html, su sintaxis es la siguiente:

`<div>Párrafos </div>`

Estas divisiones sirven para particionar en bloques nuestro html con el fin de darle mas orden cuando ciertos elementos comparten características similares.

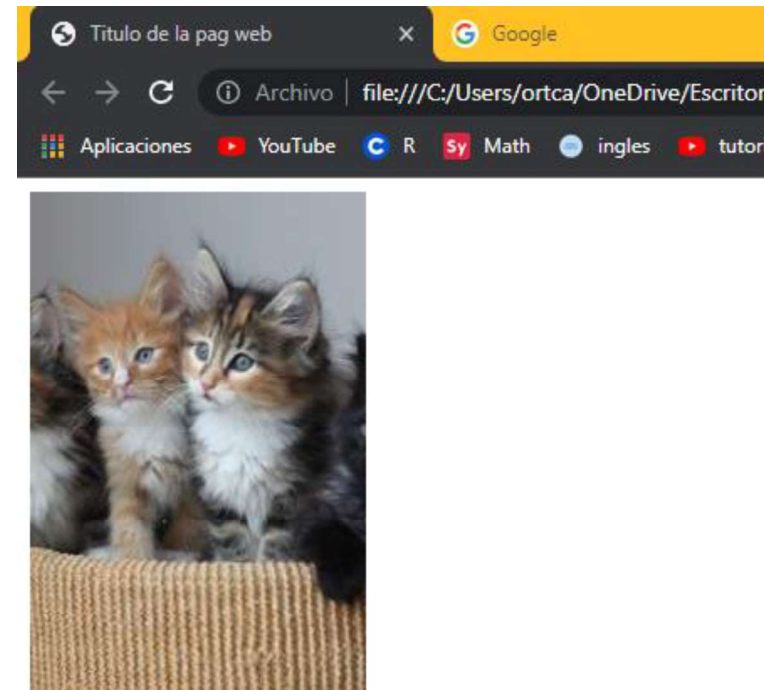
```
<!DOCTYPE html>
<html>
  <head>
    <title>Titulo de la pag web</title>
  </head>
  <body>
    <div>sección 1</div>
    <div>sección 2</div>
    <div>sección 3</div>

  </body>
</html>
```

Imágenes

Estas se insertarán mediante la etiqueta `img` y una url asociada a una imagen o recurso que se quiera insertar, para nuestro ejemplo visitaremos un sitio web que permite obtener imágenes aleatorias

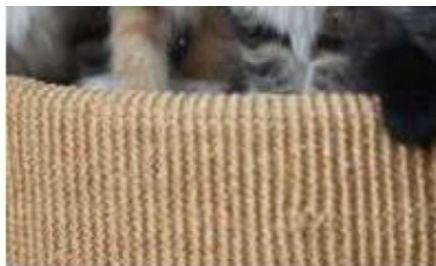
```
<html>
  <head>
    <title>Titulo de la pag web</title>
  </head>
  <body>
    
  </body>
</html>
```



link

Dentro de nuestros párrafos o imágenes podemos usar la etiqueta `a`, la cual nos redireccionara a un link que nosotros queramos.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Titulo de la pag web</title>
  </head>
  <body>
    
    <a href="https://www.google.com/"> Ir a google para mas info </a>
  </body>
</html>
```



[Ir a google para mas info](https://www.google.com/)

Introducción a css

CSS es un lenguaje que nos permitirá trabajar en conjunto con html para darle el formato deseado a nuestra pagina web, css es dependiente de html, contrario de este ultimo que puede existir por si solo.

Se denomina hojas de estilo en cascada por que puede tener varias hojas o “cajas” y cada una de ellas propiedades heredadas en forma de cascada de otras.

Con css se crean las reglas para decirle a nuestra pag como queremos mostrar nuestra información, y como vimos anteriormente mediante las clases o los ids podremos generalizar formatos a todos los elementos que compartan sus etiquetas.

Su sintaxis es muy parecida a los diccionarios en Python, donde dentro especificaremos, las etiquetas que queremos afectar y el atributo que queremos modificar. Se tiene que crear otro script y vincularlo en nuestro html de la siguiente forma.

Introducción a css

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ejemplo CSS</title>
    <link rel="stylesheet" href="main.css" type="text/css">
  </head>
  <body>
    <p>Parrafo</p>
    <spa>Esto es un span</spa>

  </body>
</html>
```

```
body {
  background-color: #fff000;
}
```

Parrafo

Esto es un span

Selectores Id y Class

Como vimos, parte importante del código html son las etiquetas , dentro de las cuales podemos especificar su id y clase, esto nos servirá para relacionar estos selectores a un apartado del css donde le daremos ciertas propiedades de formato a nuestras “cajas” dependiendo de si comparten su clase , o de manera individual por id.

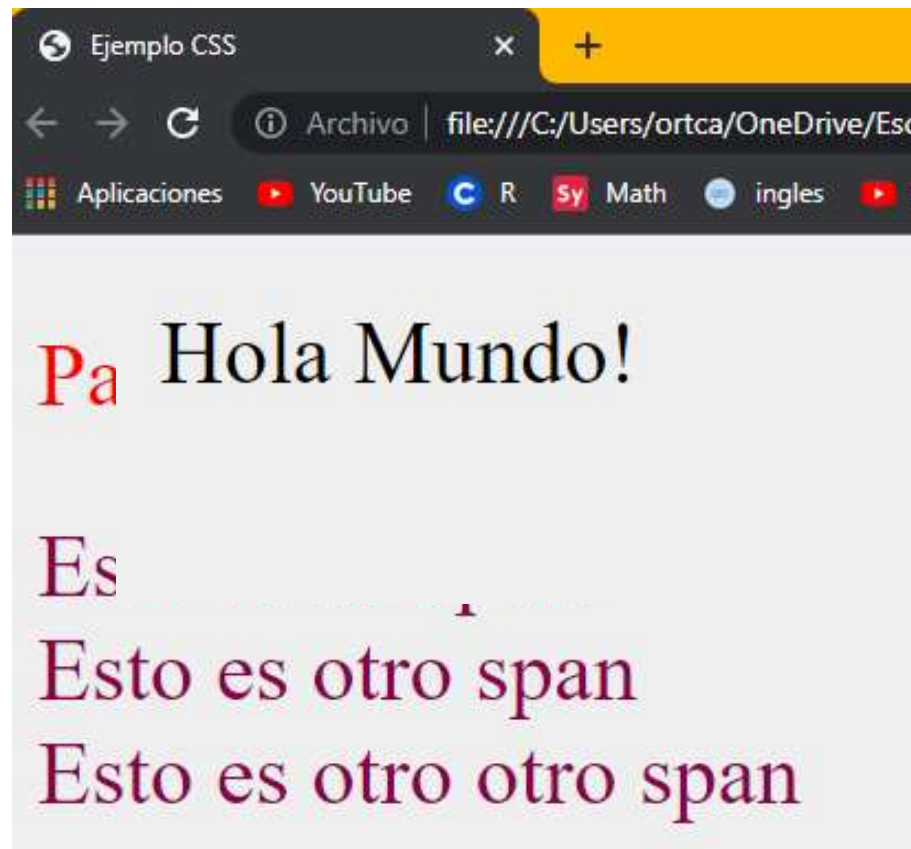
```
<!DOCTYPE html>
<html>
  <head>
    <title>Ejemplo CSS</title>
    <link rel="stylesheet" href="main.css" type="text/css">
  </head>
  <body>
    <p id="parrafo">Parrafo</p>
    <spa class="clase">Esto es un span</spa>
    <br/>
    <span class="clase">Esto es otro span</span>
    <br/>
    <span class="clase">Esto es otro otro span</span>
  </body>
</html>
```

```
body {
  background-color: #efefef;
}

#parrafo{
  color: #ff0000;
}

.clase{
  color: #8C004B;
}
```

Selectores Id y Class

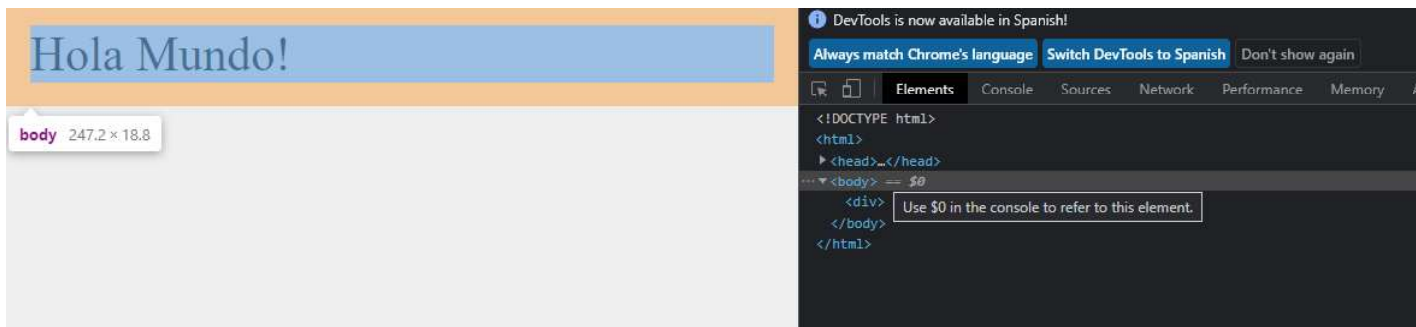


Modelo de caja

En html todas nuestras etiquetas son consideradas como “cajas” , bajo el concepto tradicional, hablar del modelo de caja, implica el cómo se modelarán los bloques de etiquetas en nuestra pagina.

Una manera fácil de ver este concepto es con el apartado inspeccionar como sigue

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ejemplo CSS</title>
    <link rel="stylesheet" href="main.css" type="text/css">
  </head>
  <body>
    <div>
      Hola Mundo!
    </div>
  </body>
</html>
```



Elementos

Márgenes

```
body {  
  background-color: #efefef;  
}  
  
.caja{  
  margin: 8px;  
}
```

Hola Mundo!

Hola Mundo!

Hola Mundo!

Elementos

Márgenes

```
.caja{  
  background-color: #00f;  
  margin: 8px;  
}
```



Hola Mundo!

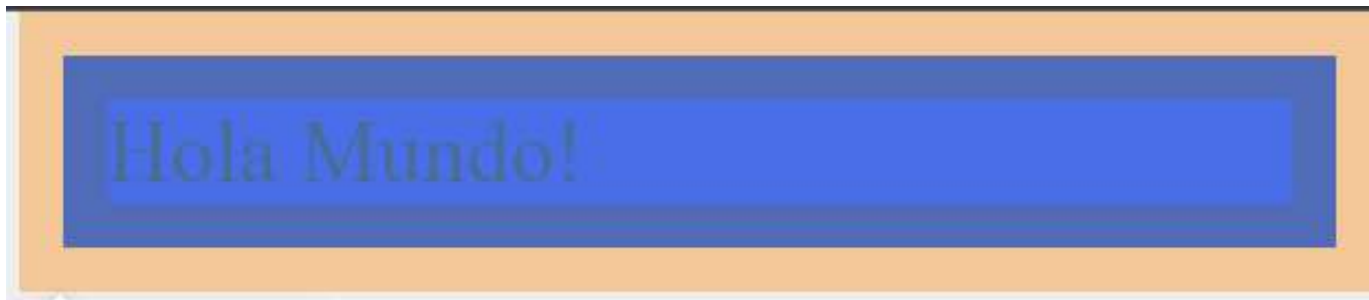


Hola Mundo!

Elementos

padding

```
.caja{  
  background-color: #00f;  
  margin: 8px;  
  padding: 8px;  
}
```



Elementos

Border(Entre margin y padding -- double,dashed,dotted,grooved,etc--)

```
.caja{  
  background-color: #00f;  
  margin: 8px;  
  padding: 8px;  
  border-style: solid;  
}
```



Hola Mundo!