

Visualización de Datos con Google Charts

SISTEMAS DE BIG DATA

MARCOS GIL CARAVACA
BIG DATA E IA | IES LA MAR

ÍNDICE

INTRODUCCIÓN	2
LIMPIEZA DE DATOS CON PANDAS	3
CREACIÓN DE LA VISUALIZACIÓN DE DATOS CON GOOGLE CHARTS	5
CONCLUSIÓN.....	9
REFERENCIAS	9

INTRODUCCIÓN

En este programa vamos a proceder a la extracción de datos de la revisión del padrón de habitantes de los municipios de la provincia de Alicante, desde el año 2005 hasta 2021.

<https://datos.gob.es/es/catalogo/101010014-revision-del-padrón-de-habitantes-de-los-municipios-de-la-provincia-de-alicante>

En dicho enlace podremos encontrar la información en 4 formatos diferentes *XLSX*, *CSV*, *JSON* y *XML*.

Nosotros vamos a utilizar el fichero con extensión **JSON**. Una vez obtenido el fichero, utilizaremos la librería de Python (Pandas) para poder filtrar por los datos que nos interesen. Y después volveremos a exportar los datos filtrados a un nuevo fichero **JSON**.

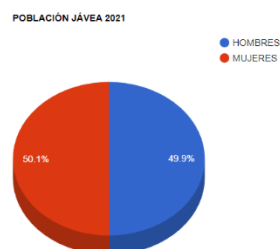
Una vez hecho el filtrado y limpieza de datos procederemos a la muestra de los datos utilizando la herramienta Google Charts, en este caso utilizaremos el tipo de [gráfico circular](#).

Por otro lado he utilizado la librería CharsJS para la creación de un gráfico de líneas del total de población por año, pero que en este documento no está reflejado, sí que en el [GitHub](#) está todo el código para que puedas consultarlo.

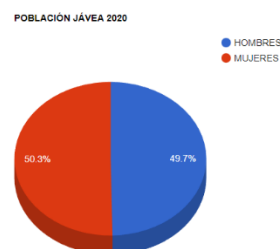
Una vez realizado todos los pasos anteriores, sin aplicar fuentes y estilos quedaría algo así similar.

POBLACIÓN EN JÁVEA

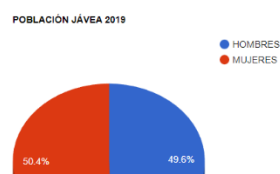
AÑO 2021



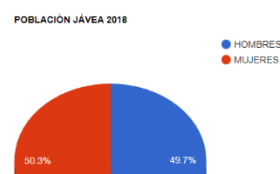
AÑO 2020



AÑO 2019



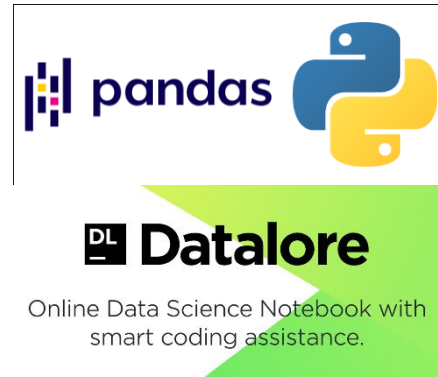
AÑO 2018



LIMPIEZA DE DATOS CON PANDAS

En este apartado veremos como extraer los datos referentes a la población de Jávea, haciendo uso de la librería de Python llamada Pandas. Y utilizaremos la herramienta DataLore para poder crear nuestro cuaderno de Jupiter y poder realizar nuestro código y extracción de datos.

Dicho esto, a continuación se muestran los pasos a seguir:



1. Leeremos el fichero descargado previamente descargado, de la página web.

```
[1] ▶ 0.1s
import json
with open('/data/notebook_files/revison-del-padron-de-habitantes-evolucion.json') as f:
    data = json.load(f)
```

2. Comprobamos que efectivamente se han registrado correctamente los datos en la variable **data**.

```
▶ 0.3s
data

{'RP_POBLACION': '28465'},
{'MU_CODINE': '03059',
 'MU_NOMBRE': 'Crevillent',
 'RP_EJERCICIO': '2016',
 'RP_HOMBRES': '14525',
 'RP_MUJERES': '14166',
 'RP_POBLACION': '28691'},
{'MU_CODINE': '03059',
 'MU_NOMBRE': 'Crevillent',
 'RP_EJERCICIO': '2017',
 'RP_HOMBRES': '14572',
 'RP_MUJERES': '14264',
 'RP_POBLACION': '28836'},
{'MU_CODINE': '03059',
 'MU_NOMBRE': 'Crevillent',
 'RP_EJERCICIO': '2018',
 'RP_HOMBRES': '14640',
 'RP_MUJERES': '14317',
 'RP_POBLACION': '28957'},
...}]
```

3. Ahora leemos cada registro:

```
▶ 0.05
for registro in data['registros']:
    print(registro)
```

```
+ Show all
{'MU_CODINE': '03903', 'MU_NOMBRE': 'Montesinos (Los)', 'RP_EJERCICIO': '2020', 'RP_HOMBRES': '2579', 'RP_MUJERES': '2482', 'RP_POBLACION': '25461'},
{'MU_CODINE': '03903', 'MU_NOMBRE': 'Montesinos (Los)', 'RP_EJERCICIO': '2021', 'RP_HOMBRES': '2624', 'RP_MUJERES': '2499', 'RP_POBLACION': '25461'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2021', 'RP_HOMBRES': '1117', 'RP_MUJERES': '1029', 'RP_POBLACION': '1094'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2020', 'RP_HOMBRES': '1094', 'RP_MUJERES': '1009', 'RP_POBLACION': '1094'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2019', 'RP_HOMBRES': '1024', 'RP_MUJERES': '962', 'RP_POBLACION': '962'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2018', 'RP_HOMBRES': '988', 'RP_MUJERES': '945', 'RP_POBLACION': '945'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2017', 'RP_HOMBRES': '983', 'RP_MUJERES': '937', 'RP_POBLACION': '937'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2016', 'RP_HOMBRES': '974', 'RP_MUJERES': '941', 'RP_POBLACION': '941'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2015', 'RP_HOMBRES': '961', 'RP_MUJERES': '913', 'RP_POBLACION': '913'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2014', 'RP_HOMBRES': '975', 'RP_MUJERES': '916', 'RP_POBLACION': '916'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2013', 'RP_HOMBRES': '980', 'RP_MUJERES': '921', 'RP_POBLACION': '921'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2012', 'RP_HOMBRES': '989', 'RP_MUJERES': '932', 'RP_POBLACION': '932'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2011', 'RP_HOMBRES': '999', 'RP_MUJERES': '935', 'RP_POBLACION': '935'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2010', 'RP_HOMBRES': '966', 'RP_MUJERES': '908', 'RP_POBLACION': '908'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2009', 'RP_HOMBRES': '937', 'RP_MUJERES': '869', 'RP_POBLACION': '869'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2008', 'RP_HOMBRES': '881', 'RP_MUJERES': '827', 'RP_POBLACION': '827'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2007', 'RP_HOMBRES': '806', 'RP_MUJERES': '755', 'RP_POBLACION': '755'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2006', 'RP_HOMBRES': '725', 'RP_MUJERES': '676', 'RP_POBLACION': '676'},
{'MU_CODINE': '03904', 'MU_NOMBRE': 'San Isidro', 'RP_EJERCICIO': '2005', 'RP_HOMBRES': '704', 'RP_MUJERES': '664', 'RP_POBLACION': '664'}
```

4. Ahora importamos la librería Pandas y creamos las columnas pertinentes para almacenar los diferentes registros.

```
1.4s
import pandas as pd
cols = ['MU_CODINE', 'MU_NOMBRE', 'RP_EJERCICIO', 'RP_HOMBRES', 'RP_MUJERES', 'RP_POBLACION']
df = pd.DataFrame(columns = cols)
```

5. Recorremos cada registro y creamos una variable booleana para comprobar si existe o no el valor “Jávea/Xàbia” en la clave ‘MU_NOMBRE’. Una vez comprobado insertaremos la fila al dataframe **df** creado el punto anterior. La impresión de los datos se realiza para saber que datos estamos cogiendo y verificar que son esos y no otros.

```
for registro in data['registros']:
    isJavea = "Jávea/Xàbia" == registro['MU_NOMBRE']
    if isJavea:
        print('MU_CODINE', registro['MU_CODINE'])
        print('MU_NOMBRE', registro['MU_NOMBRE'])
        print('RP_EJERCICIO', registro['RP_EJERCICIO'])
        print('RP_HOMBRES', registro['RP_HOMBRES'])
        print('RP_MUJERES', registro['RP_MUJERES'])
        print('RP_POBLACION', registro['RP_POBLACION'])
        print('='*30)
    #Añadiendo una fila al dataframe
    df = df.append(registro, ignore_index=True)
```

6. Una vez finalizado el punto anterior imprimimos los datos del dataframe, con la siguiente instrucción:

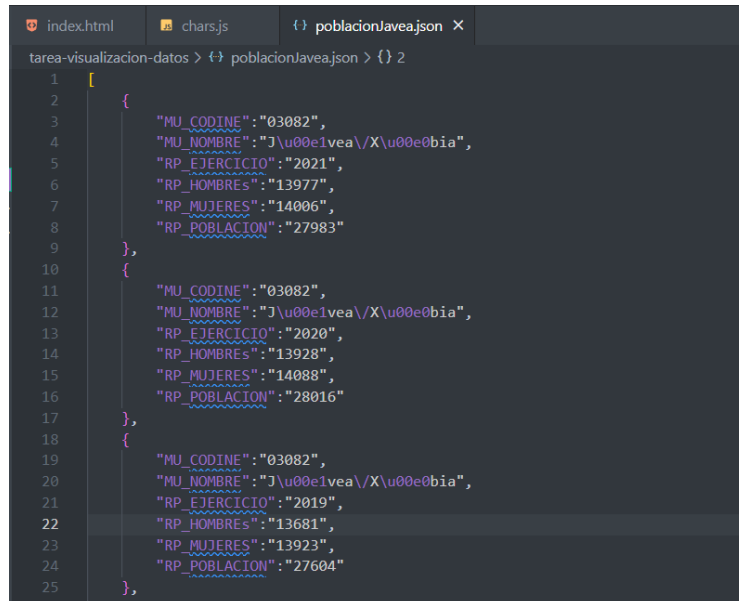
```
0.2s
df.head(30)
```

	MU_CODINE	MU_NOMBRE	RP_EJERCICIO	RP_HOMBRES	RP_MUJERES	RP_POBLACION
0	03082	Jávea/Xàbia	2021	13977	14006	27983
1	03082	Jávea/Xàbia	2020	13928	14088	28016
2	03082	Jávea/Xàbia	2019	13681	13923	27604
3	03082	Jávea/Xàbia	2018	13528	13696	27224
4	03082	Jávea/Xàbia	2017	13441	13619	27060
5	03082	Jávea/Xàbia	2016	13506	13719	27225
6	03082	Jávea/Xàbia	2015	13704	13977	27681
7	03082	Jávea/Xàbia	2014	14499	14568	29067
8	03082	Jávea/Xàbia	2013	16572	16577	33149
9	03082	Jávea/Xàbia	2012	16560	16423	32983
10	03082	Jávea/Xàbia	2011	16277	16192	32469
11	03082	Jávea/Xàbia	2010	15989	15920	31909
12	03082	Jávea/Xàbia	2009	15844	15749	31593
13	03082	Jávea/Xàbia	2008	15660	15480	31140
14	03082	Jávea/Xàbia	2007	15096	14827	29923
15	03082	Jávea/Xàbia	2006	14851	14428	29279
16	03082	Jávea/Xàbia	2005	14307	13935	28242

7. Por últimos solo tenemos que exportar los datos a un documento JSON para ello utilizaremos la siguiente instrucción.

```
df.to_json('poblacionJavea.json', orient='records')
```

8. Una vez hecho esto nos descargamos en el PC y nos deberá de salir algo parecido a lo que se muestra en la imagen, una vez lo pasemos al Javascript los caracteres como los acentos en este caso se visualizarán bien.



```
tarea-visualizacion-datos > { } poblacionJavea.json > { } 2
1  [
2    {
3      "MU_CODINE": "03082",
4      "MU_NOMBRE": "J\u00e1vea/X\u00e0bia",
5      "RP_EJERCICIO": "2021",
6      "RP_HOMBRES": "13977",
7      "RP_MUJERES": "14006",
8      "RP_POBLACION": "27983"
9    },
10   {
11     "MU_CODINE": "03082",
12     "MU_NOMBRE": "J\u00e1vea/X\u00e0bia",
13     "RP_EJERCICIO": "2020",
14     "RP_HOMBRES": "13928",
15     "RP_MUJERES": "14088",
16     "RP_POBLACION": "28016"
17   },
18   {
19     "MU_CODINE": "03082",
20     "MU_NOMBRE": "J\u00e1vea/X\u00e0bia",
21     "RP_EJERCICIO": "2019",
22     "RP_HOMBRES": "13681",
23     "RP_MUJERES": "13923",
24     "RP_POBLACION": "27604"
25   },
26 ]
```

NOTA: He hecho uso de la librería Pandas para poder visualizar mejor los datos y poder detectar posibles anomalías como datos nulos, por ejemplo. En este caso no se han detectado datos nulos ni nada similar.

CREACIÓN DE LA VISUALIZACIÓN DE DATOS CON GOOGLE CHARTS

A continuación nos falta mostrar el **JSON** filtrado en un página HTML5 haciendo uso de Google Charts en este caso haremos uso del tipo de gráfica CoreChart.

Para ello utilizaremos la librería de gráficos Google Charts, utilizando el siguiente código que deberemos de colocarlo en el header, de nuestro documento **index.html**

```
<!-- Carga la librería google charts de google para poder realizar el gráfico circular-->
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
```

En el resto del html colocaremos lo siguiente. En GitHub se han ido haciendo mejores en este archivo, las cuales no se ven reflejadas en la captura.

```
enpadronamientoJavea > index.html > html > body
1 <!DOCTYPE html>
2 <html lang="es">
3   <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7
8     <title>DATOS POBLACIÓN JÁVEA</title>
9
10    <!-- Carga la librería google charts de google para poder realizar el gráfico circular-->
11    <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
12
13    <!-- Compiled and minified CSS -->
14    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/materialize.min.css">
15
16    <!-- Compiled and minified JavaScript -->
17    <script src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js"></script>
18  </head>
19  <body>
20    <!-- EJECUCIÓN DE LOS CHARS -->
21    <script type="text/javascript" src="chars.js">
22    </script>
23    <div class="container">
24      <h1 style="text-align: center;"><b>POBLACIÓN EN JÁVEA</b></h1>
25      <div class="row" id="fechas" style="text-align: center;">
26        <script text="javascript">
27          let cont = 0;
28          const item = document.querySelector('#fechas');
29          for(let i = 2021;i>=2005;i--){
30            cont ++;
31            let fechaActual = i;
32            let tipoChar = "p"+cont+"Chart";
33            console.log(tipoChar);
34            item.innerHTML += `
35              <div class="col s12 m12 l6">
36                <h2>AÑO `+fechaActual+`</h2>
37                <div id="`+tipoChar+`" class="col s12"
38              </div>
39            `;
40          }
41        </script>
42      </div>
43    </div>
44  </body>
45 </html></body></html></body></html>
46
47
```

El siguiente archivo a crear es el chars.js que es el que va a leer el archivo json y va a crear los distintos gráficos circulares por cada año.

En el siguiente código podemos observar como cargamos los paquetes de google y también seleccionamos el tipo de gráfico a utilizar y leemos el archivo json.

```

1 // Le pasamos el paquete del tipo de gráfico que deseamos
2 google.charts.load("current", {packages: ["corechart"]});
3 // A continuación ejecutar la función drawChart
4 google.charts.setOnLoadCallback(drawCharts);
5
6 // Lectura fichero JSON
7 const xhttp = new XMLHttpRequest();
8 let datos;
9
10 xhttp.open('GET', 'poblacionJavea.json', true);
11
12 xhttp.send();
13
14 xhttp.onreadystatechange = function () {
15     if (xhttp.readyState == 4 && this.status == 200) {
16         // Mostrar Datos Cargados por consola
17         datos = JSON.parse(this.responseText);
18         console.log(datos);
19     }
20 }

```

A continuación creamos una función donde le pasaremos tres parámetros (fecha, numGrafico, TituloGrafica). La función de este método será crear un gráfico por cada año del (2021 al 2005).

```

// Función para la creación de la gráfica
function drawChartPoblacion(fecha, numGrafico, TituloGrafica) {
    console.log(fecha);

    var data = new google.visualization.DataTable();
    data.addColumn('string', 'TIPO');
    data.addColumn('number', 'CANTIDAD');

    for (let year of datos) {
        if (year['RP_EJERCICIO'] == fecha) {
            let poblacionHombres = parseInt(year['RP_HOMBRES']);
            let poblacionMujeres = parseInt(year['RP_MUJERES']);
            console.log(poblacionHombres);
            console.log(poblacionMujeres);
            data.addRow([
                [
                    'HOMBRES', poblacionHombres
                ],
                [
                    'MUJERES', poblacionMujeres
                ]
            ])
        }
    }

    var options = {
        title: TituloGrafica,
        is3D: true,
        width: 750,
        height: 630,
        colors: ['#1e88e5', '#005cb2']
    };

    var chart = new google.visualization.PieChart(document.getElementById(numGrafico));
    chart.draw(data, options);
}

```


A continuación crearemos una función la cual dibujará todos los gráficos. Llamando a la función anterior y pasándole los diferentes parámetros dibujamos los Charts.

```
1 function drawCharts() {
2   let cont = 0;
3   for(let i=2005; i<=2021; i++){
4     cont ++;
5     let tipoChar = "p" + cont + "Chart"; // Defiendo el nombre de char a seguir
6     drawChartPoblacion(i,tipoChar,"POBLACIÓN EN JÁVEA "+i);
7   }
8 }
```

RESULTADO FINAL:



CONCLUSIÓN

Me ha parecido un trabajo bastante divertido, al final estamos extrayendo datos, para después limpiarlos, seleccionar lo que deseemos extraerlo a otro fichero y mostrar los datos de manera gráfica y entendible, para la persona que lo vaya a consultar.

El trabajo está subido a GitHub. A continuación dejo el enlace del mismo:

<https://github.com/marcosgil1996/enpadronamientoJavea>

Y a continuación dejo el enlace del resultado final para que lo puedas consultar:

<https://marcosgil1996.github.io/enpadronamientoJavea/>

Puede ser que al ejecutar el enlace anterior en el Firefox veas que no cargue bien los gráficos. Prueba con Google Chrome, Brave, Edge por ejemplo. En Firefox hay veces que no carga bien los gráficos

REFERENCIAS

Video de referencia para la visualización de datos GoogleCharts:

<https://www.youtube.com/watch?v=afrDuYPfD10&t=932s>

Página oficial de GoogleCharts para el tema de diseño y personalizado del mismo Chart,

[https://developers-dot-devsite-v2
prod.appspot.com/chart/interactive/docs/gallery/piechart.html](https://developers-dot-devsite-v2.prod.appspot.com/chart/interactive/docs/gallery/piechart.html)

Librería Google Fonts para cambiar el tipo de fuente

<https://fonts.google.com/>

Chars JS para la creación del gráfico total de población por año:

<https://www.chartjs.org/>