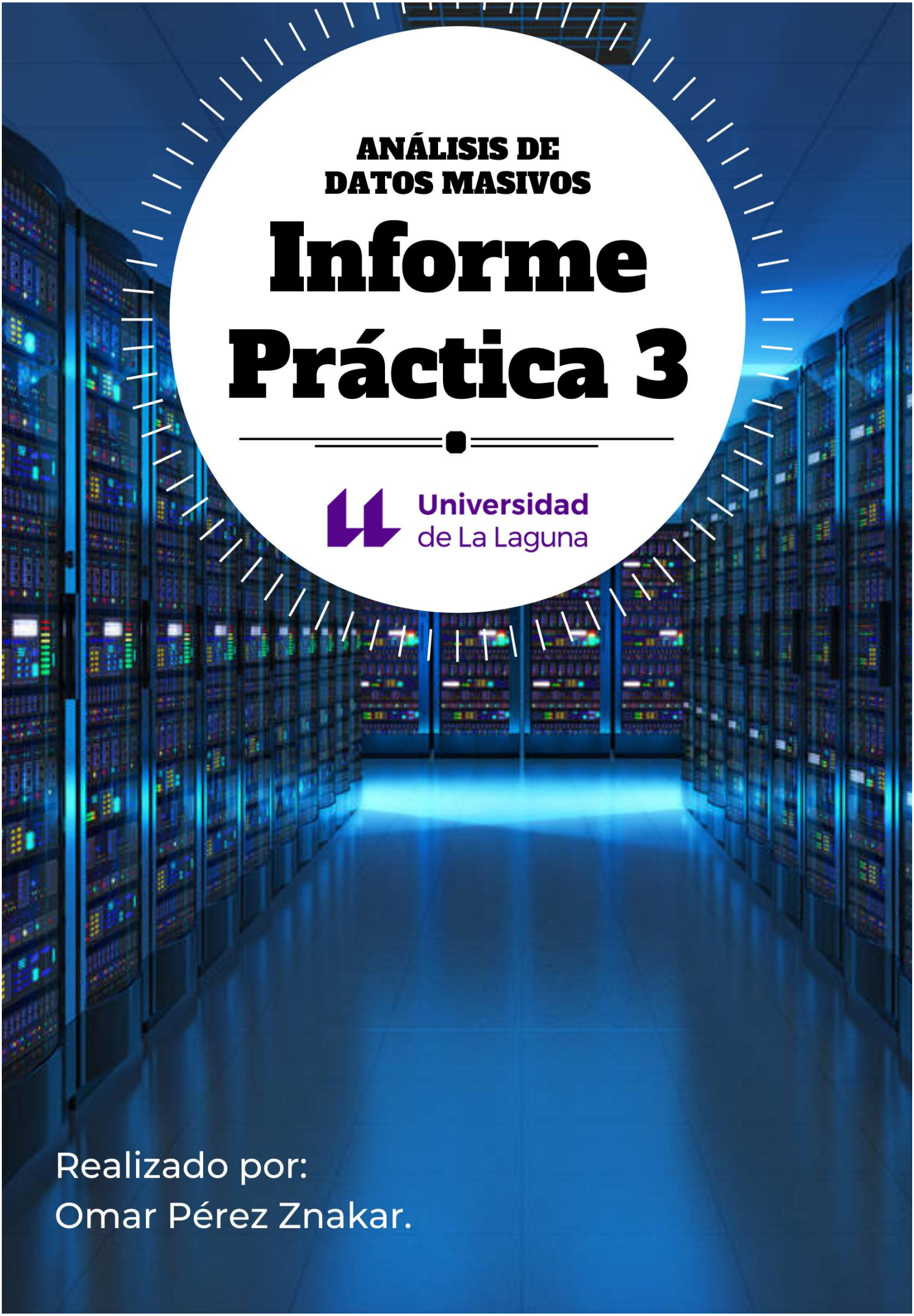


ANÁLISIS DE DATOS MASIVOS

Informe Práctica 3

Universidad de La Laguna

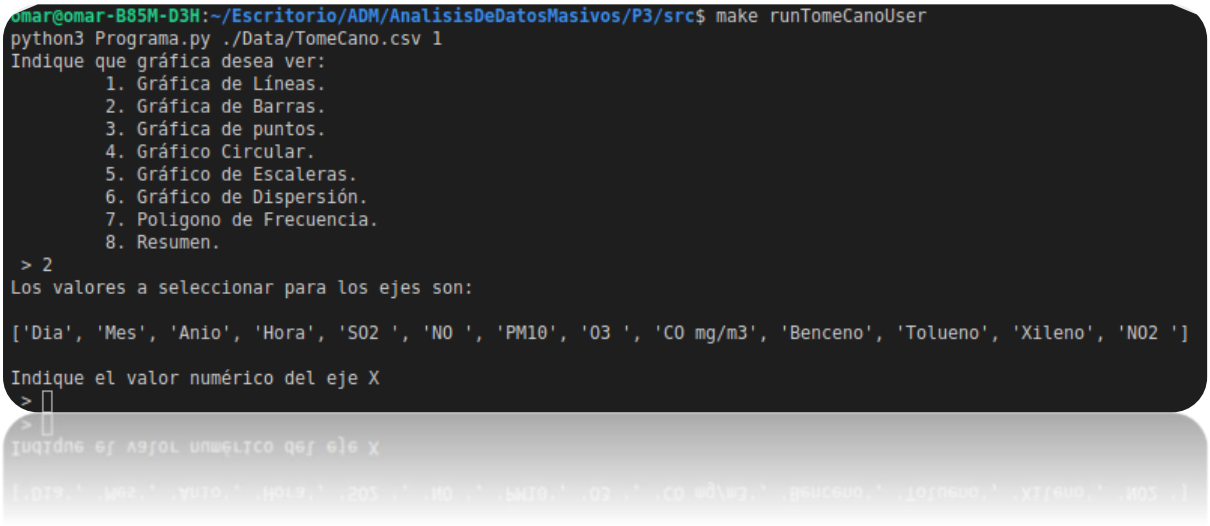
Realizado por:
Omar Pérez Znakar.

INDICE

1.- Mejoras.

1.1 Mostrar las opciones elegibles.

Tras la revisión de la semana pasada, el profesor me propuso poder ver las posibles opciones para facilitar la selección al usuario. Para ello, se ha realizado la siguiente aportación:



1.2 Gestión de grupos.

Esta aportación consiste en la posibilidad de agrupar por un elemento de la tabla y poder realizar diversas acciones de interés. Un ejemplo de esto, es la posibilidad de agrupar por sexo (dentro de los datos del coronavirus) o por mes (en el diagrama de los datos de la calidad del aire). Sin embargo, a la hora de agrupar existen diversas posibilidades, es decir, ¿Qué hacemos a la hora de agrupar? Por ello, se ha propuesto las siguientes acciones:

- **Suma**: sumar los elementos que coincidan. Por ejemplo, en caso de agrupar por sexo (en unos datos de muertes del coronavirus) sumaremos todas las muertes de las mujeres y del hombre por separado y, realizaremos el gráfico que el usuario nos ha pedido.
- **Máximo:** coger el valor más grande. Por ejemplo, en los datos de la calidad del aire si agrupamos los datos por día, podremos coger el valor máximo del día y, así podremos dictaminar si en ese día tenemos una buena o mala calidad del aire (según la OMS se tiene que coger el valor más alto y comprobar si está dentro de unos valores predeterminado).
- **Mínimo**: coger el valor más pequeño. Por ejemplo, en los datos de la calidad del aire si agrupamos los datos por días, podremos determinar a qué horas tenemos la mejor calidad del aire y, así podremos dictaminar la mejor franja horaria para hacer deporte (por ejemplo).

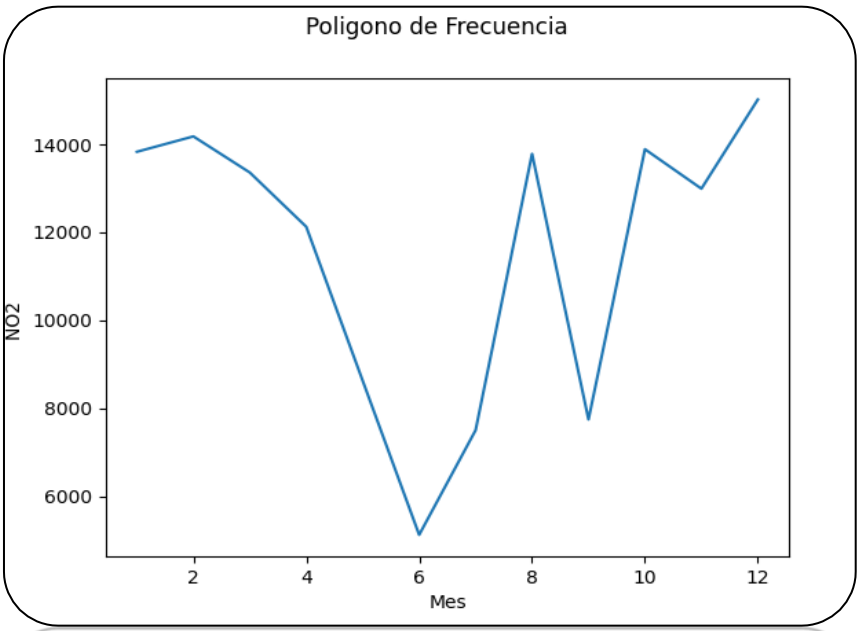
Esto lo podremos ver a continuación:

Código

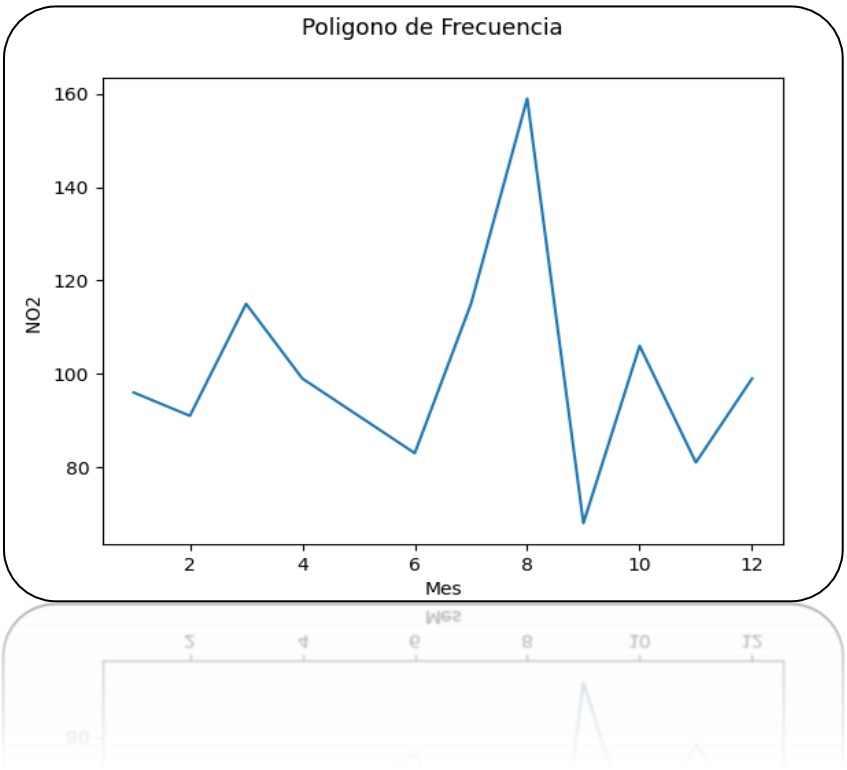
```
Indique que gráfica desea ver:
        1. Gráfica de Líneas.
        2. Gráfica de Barras.
        3. Gráfica de puntos.
        4. Gráfico Circular.
        5. Gráfico de Escaleras.
        6. Gráfico de Dispersión.
        7. Poligono de Frecuencia.
        8. Resumen.
 > 2
Los valores a seleccionar para los ejes son:

['Dia', 'Mes', 'Anio', 'Hora', 'SO2 ', 'NO ', 'PM10', 'O3 ', 'CO mg/m3', 'Benceno', 'Tolueno', 'Xileno', 'NO2 ']

Indique el valor numérico del eje X
 > 1
Indique el valor numérico del eje Y
 > 12
Indique el elemento por el que desea agrupar. Si no desea agrupar clicke enter
 > 1
En caso de colisión de datos similares al agrupar ¿Que desea hacer?:
        1. Suma.
        2. Máximo.
        3. Mínimo.
        4. Ninguno.
```

Gráfica (datos meteorológicos TomeCano)

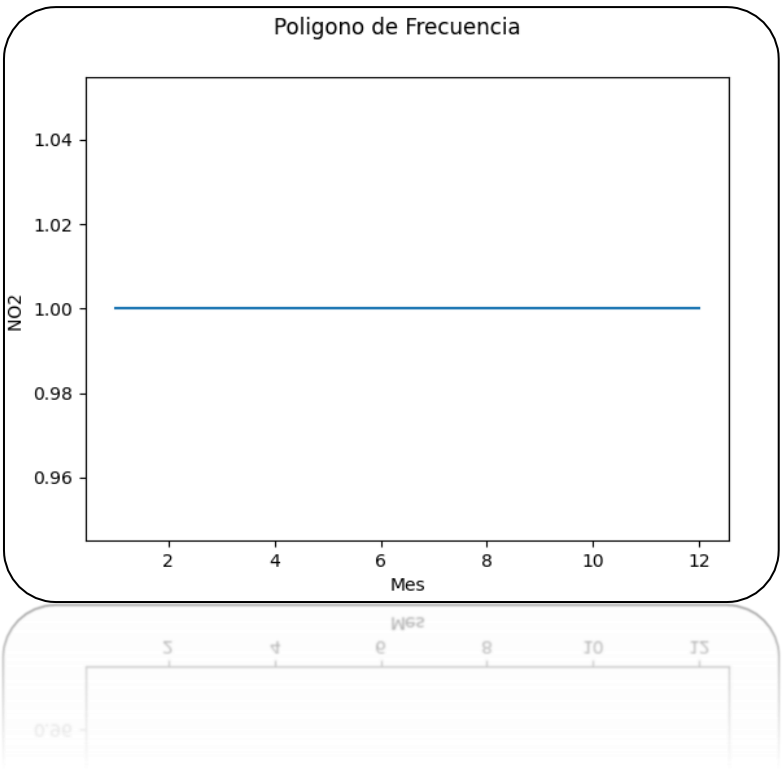
Agrupando por Suma



Agrupando por Máximo



Agrupando por Mínimo

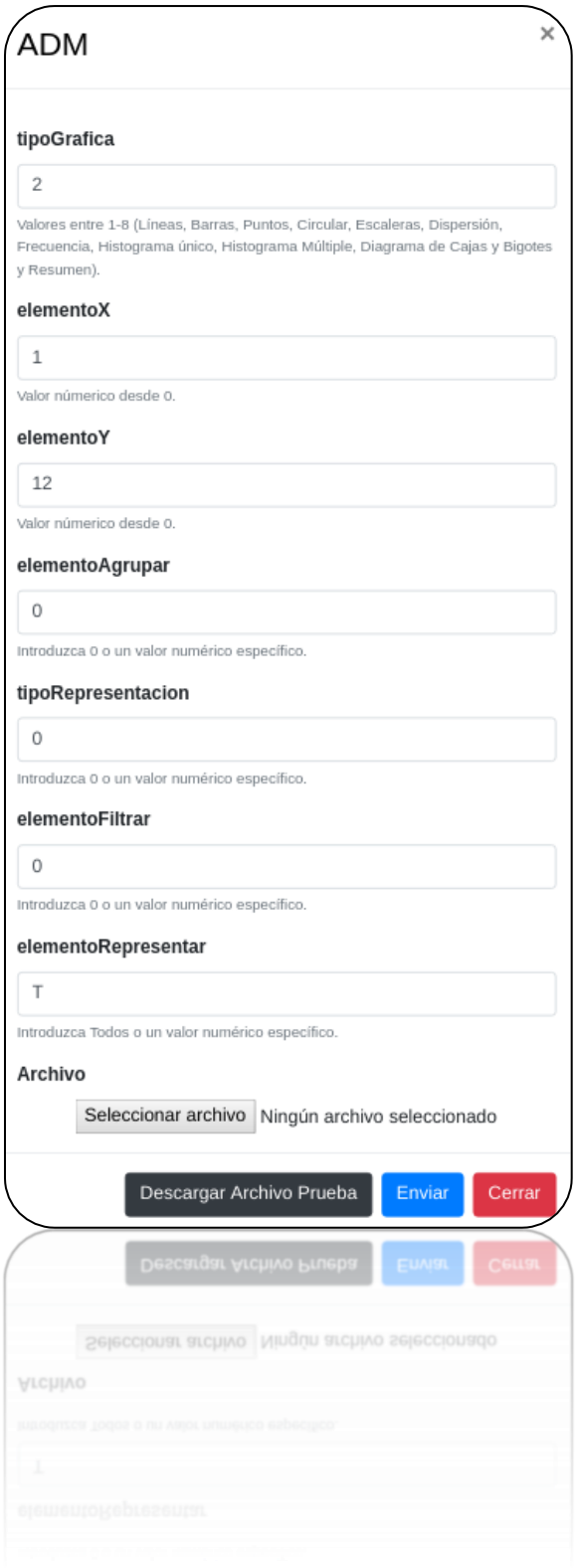


1.3 Ejecución de ejemplo para los algoritmos.

Para este apartado, se ha incorporado la posibilidad de crear un ejemplo para el algoritmo dado en el proyecto de Computación en la Nube. Para ello, se ha modificado el fichero JSON añadiendo el nombre del fichero de ejemplo y los valores para cada campo. Esto lo podremos ver a continuación:

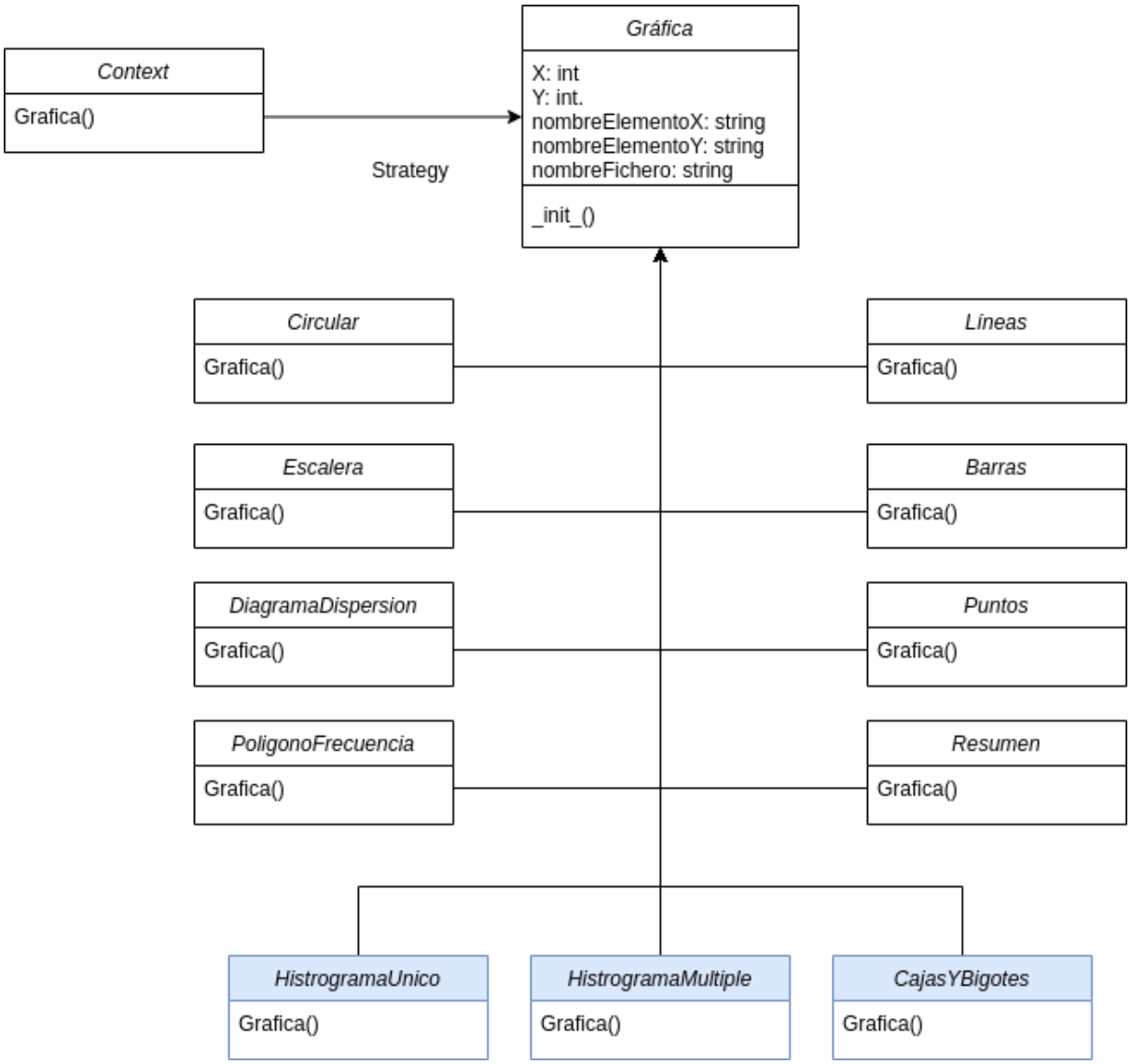

```
{
  "Name":"ADM",
  "file":"TomeCano.csv",
  "Description":"Método que permite recoger una gráfica a través de un fichero '.csv' dado con anteriornidad.",
  "Elements":[
    {
      "Name":"tipoGrafica",
      "value":"2",
      "Description":"Valores entre 1-8 (Líneas, Barras, Puntos, Circular, Escaleras, Dispersión, Frecuencia, H
    },
    {
      "Name":"elementoX",
      "value":"1",
      "Description":"Valor número desde 0."
    },
    {
      "Name":"elementoY",
      "value":"12",
      "Description":"Valor número desde 0."
    },
    {
      "Name":"elementoAgrupar",
      "value":"0",
      "Description":"Introduzca 0 o un valor numérico específico."
    },
    {
      "Name":"tipoRepresentacion",
      "value":"0",
      "Description":"Introduzca 0 o un valor numérico específico."
    },
    {
      "Name":"elementoFiltrar",
      "value":"0",
      "Description":"Introduzca 0 o un valor numérico específico."
    },
    {
      "Name":"elementoRepresentar",
      "value":"T",
      "Description":"Introduzca Todos o un valor numérico específico."
    }
  ]
}
```

Posteriormente, se ha modificado el Frontend, recogiendo los valores impuesto por el propietario del algoritmo de forma predeterminada. A continuación, se ha añadido un botón que nos permite descargar el archivo de ejemplo para añadirlo a la ejecución. Un ejemplo de esto lo podremos ver seguidamente:



1.4 Nuevas Gráficas.

A continuación, veremos las nuevas gráficas implementadas. Debido a esto, el diagrama de clases ha cambiado añadiendo estas nuevas funcionalidades (marcadas en azul). Este lo veremos a continuación:



1.4.1 Histograma.

Para este apartado se ha creado un nuevo tipo de diagrama. Este es el denominado como “Histograma”. Un histograma es la representación gráfica en forma de barras, que simboliza la distribución de un conjunto de datos. Sirven para obtener una "primera vista" general, o panorama, de la distribución de la población, o de la muestra, respecto a una característica, cuantitativa y continua.
En cuanto al programa generado se optado por realizar dos tipos de Histograma. Estos los podremos ver en los puntos siguientes.

1.4.1.1 Histograma único.

Esta representación está formada por un único histograma con todos los datos del archivo. Para realizar esto, se han seguido los siguientes pasos:

- Añadir este diagrama a la lista que se le pide al usuario.
- Crear una nueva clase.
- Llamar a la función de este diagrama con el valor del eje Y pedido con anterioridad.

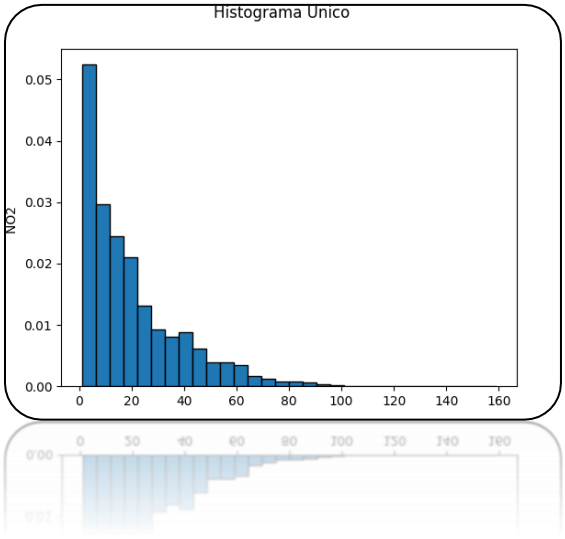
Podremos ver lo realizado con anterioridad a continuación:

Código

```
class HistogramaUnico(Grafica):
  def grafica(self, df):
    fig = plt.figure()
    fig.suptitle('Histograma Unico')
    plt.ylabel(self.nombreElementoY)
    plt.hist(self.Y, density=True, bins=30, alpha=1, edgecolor = 'black',  linewidth=1)
    if self.nombreFichero:
      plt.savefig(self.nombreFichero)
    else:
      plt.show()
```

Gráfica

Se ha creado un Histograma a partir de los datos de Tomé Cano. Para ello, se ha escogido el valor NO2.



1.4.1.2 Histograma múltiple.

Esta representación está formada por varios Histogramas para cada elemento seleccionado que se unen en un único. Esta representación, es muy útil para comparar gráficos. Para realizar esto, se han seguido los siguientes pasos:

- Añadir este diagrama a la lista que se le pide al usuario.
- Crear una nueva clase.
- Crear un vector único para cada elemento del eje x. Estos vectores se introducirán dentro de otro general, creando una estructura de vector de vectores.
- Llamar a la función de este diagrama con el valor calculado con anterioridad.

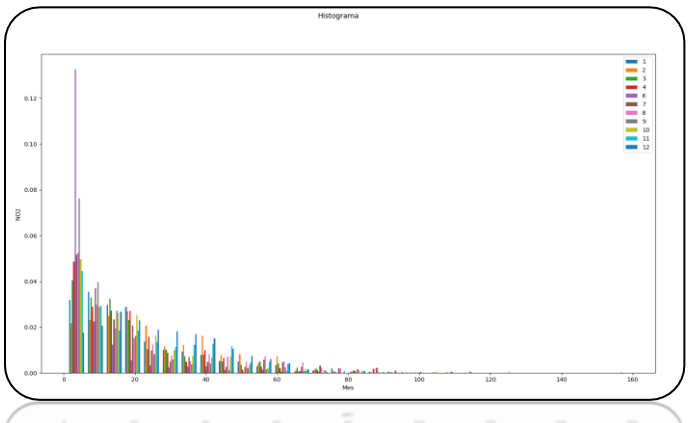
Podremos ver lo realizado con anterioridad a continuación:

Código

```
class HistogramaMultiple(Grafica):
  def grafica(self, df):
    fig = plt.figure()
    fig.suptitle('Histograma')
    plt.xlabel(self.nombreElementoX)
    plt.ylabel(self.nombreElementoY)
    listFinal = list()
    listProvisional = list()
    elements = dict.fromkeys(self.X).keys()
    for element in elements:
      listProvisional.append(element)
      elementosEjeX = df[self.nombreElementoX].isin(listProvisional)
      listFinal.append(df[elementosEjeX].loc[:,self.nombreElementoY])
      listProvisional = list()
    plt.hist(listFinal, density=True, bins=30, label=elements)
    plt.legend()
    if self.nombreFichero:
      plt.savefig(self.nombreFichero)
    else:
      plt.show()
```

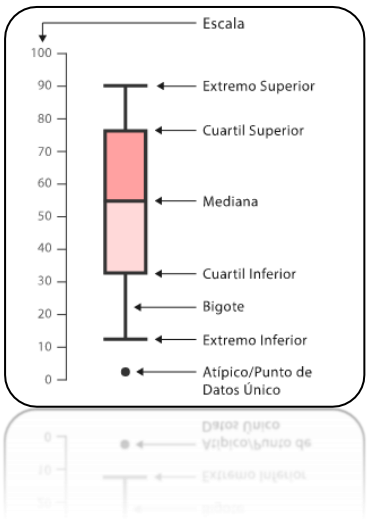
Gráfica

Se ha creado un gráfico que dispone de varios Histogramas cogiendo los datos de Tomé Cano. Para ello, se ha escogido el mes (como eje x) y NO2 (eje y).



1.4.2 Cajas y Bigotes.

Para este apartado, se ha creado un nuevo tipo de diagrama, denominado diagrama de cajas y bigotes. Un diagrama de cajas y bigotes es una manera conveniente de mostrar visualmente grupos de datos numéricos a través de sus cuartiles. Esto lo podremos ver a continuación:



Para poder realizar esta representación dentro de nuestro programa, se han seguido los siguientes pasos:

- Añadir este diagrama a la lista que se le pide al usuario.
- Crear una nueva clase.
- Crear un vector único para cada elemento del eje x. Estos vectores se introducirán dentro de otro general, creando una estructura de vector de vectores.
- Llamar a la función de este diagrama con los valores calculados.

Podremos ver lo realizado con anterioridad a continuación:

Código

```
class Cajas(Grafica):
  def grafica(self, df):
    fig = plt.figure()
    fig.suptitle('Cajas y Bigotes')
    plt.xlabel(self.nombreElementoX)
    plt.ylabel(self.nombreElementoY)
    listFinal = list()
    listProvisional = list()
    elements = dict.fromkeys(self.X).keys()
    for element in elements:
      listProvisional.append(element)
      elementosEjeX = df[self.nombreElementoX].isin(listProvisional)
      listFinal.append(df[elementosEjeX].loc[:,self.nombreElementoY])
      listProvisional = list()
    plt.boxplot(listFinal, notch=True, sym="o", labels=elements)
    if self.nombreFichero:
      plt.savefig(self.nombreFichero)
    else:
      plt.show()
```

Ejemplo

Se ha creado un diagrama de cajas y bigotes usando la fuente de datos de los datos meteorológicos de la zona de Tomé Cano. De igual forma, se ha escogido el mes (eje x) y “NO2” (eje y).