Análisis de Datos y Big Data

Sesión 3 : Visualización y comunicación de datos

Presentan:

Dr. Ulises Olivares Pinto

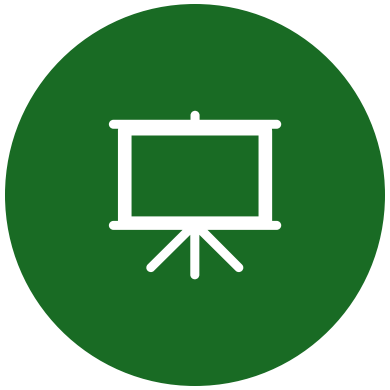
Joshelyn Yanori Mendoza Alfaro

René Delgado Servín

BLOQUE
Innovación, Tecnología y Creatividad.

Contenido

1. REPASO DE PANDAS



2. VISUALIZACIÓN DE DATOS

3. ANÁLISIS EXPLORATORIO DE DATOS

1. Repaso breve con Pandas

Series

	apples
0	3
1	2
2	0
3	1

+

Series

	oranges
0	0
1	3
2	7
3	2

=

DataFrame

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

Método *groupby* para Data Frames

Usando el método **"group by"** podemos:

- Dividir los datos en grupos en función de algunos criterios
- Calcular estadísticas (o aplicar una función) a cada grupo
- Similar a la función dplyr() en R

```
In [ ]: #Agrupando los datos usando rank
        df_rank = df.groupby(['rank'])
```

```
In [ ]: #Calcular la media de cada
        columna numérica por cada grupo
        df_rank.mean()
```

rank	phd	service	salary
AssocProf	15.076923	11.307692	91786.230769
AsstProf	5.052632	2.210526	81362.789474
Prof	27.065217	21.413043	123624.804348

Método *groupby* para Data Frames

Una vez creado el objeto groupby podemos calcular varias estadísticas para cada grupo:

```
In [ ]: #Calculate mean salary for each professor rank:
        df.groupby('rank')[['salary']].mean()
```

	salary
rank	
AssocProf	91786.230769
AsstProf	81362.789474
Prof	123624.804348

Nota: Si se utilizan corchetes individuales para especificar la columna (por ejemplo, salario), la salida es el objeto Pandas Series. Cuando se utilizan corchetes dobles, la salida es un data frame

Método *groupby* para Data Frames

- ***groupby*** *notas de rendimiento:*
  - *No se produce ninguna agrupación/división hasta que sea necesario.*
  - *La creación del objeto groupby solo comprueba que ha pasado una asignación válida*
  - *Por defecto las claves de grupo se ordenan durante la operación groupby. Se puede que desee pasar sort-False para una posible aceleración:*

```
In [ ]: #Calculate mean salary for each professor rank:
        df.groupby(['rank'], sort=False)[['salary']].mean()
```

Data Frame: filtering

Para crear un subconjunto de los datos, podemos aplicar la indexación booleana. Esta indexación se conoce comúnmente como filtro. Por ejemplo, si queremos crear un subconjunto de las filas en las que el valor salarial es mayor que $120K:

In [ ]:
```
#Calcular el salario promedio para cada rango de profesor:
df_sub = df[ df['salary'] > 120000 ]
```

Cualquier operador booleano se puede utilizar para crear un subconjunto de los datos:

In [ ]:
```
#Select only those rows that contain female professors:
df_f = df[ df['sex'] == 'Female' ]
```

Data Frames: Ordenamientos

Podemos ordenar los datos por un valor en la columna. De forma predeterminada, la ordenación se producirá en orden ascendente y se devolverá un nuevo marco de datos.

In [ ]:
```
# Cree un nuevo marco de datos a partir del original ordenado por la columna Salario
df_sorted = df.sort_values( by ='service')
df_sorted.head()
```

Out[ ]:

	rank	discipline	phd	service	sex	salary
55	AsstProf	A	2	0	Female	72500
23	AsstProf	A	2	0	Male	85000
43	AsstProf	B	5	0	Female	77000
17	AsstProf	B	4	0	Male	92000
12	AsstProf	B	1	0	Male	88000

Valores faltantes

Los valores que faltan se marcan como NaN

In [ ]:
```
# Leer un conjunto de datos con valores faltantes
vuleos = pd.read_csv(" https://raw.githubusercontent.com/ulises1229/INTRO-PYTHON-ENESJ/master/data/flights.csv ")
```

In [ ]:
```
# Seleccione las filas que tienen al menos un valor faltante
vuelos[vuelos.isnull().any(axis=1)].head()
```

Out[ ]:

	year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
330	2013	1	1	1807.0	29.0	2251.0	NaN	UA	N31412	1228	EWR	SAN	NaN	2425	18.0	7.0
403	2013	1	1	NaN	NaN	NaN	NaN	AA	N3EHAA	791	LGA	DFW	NaN	1389	NaN	NaN
404	2013	1	1	NaN	NaN	NaN	NaN	AA	N3EVAA	1925	LGA	MIA	NaN	1096	NaN	NaN
855	2013	1	2	2145.0	16.0	NaN	NaN	UA	N12221	1299	EWR	RSW	NaN	1068	21.0	45.0
858	2013	1	2	NaN	NaN	NaN	NaN	AA	NaN	133	JFK	LAX	NaN	2475	NaN	NaN

Valores faltantes

Hay una serie de métodos para tratar con los valores que faltan en el marco de datos:

df.method()	Descripción
dropna()	Retirar observaciones faltantes
dropna(how='all')	Elimina todas las celdas con NA
dropna(axis=1, how='all')	Elimina la columna si faltan todos los valores
dropna(thresh = 5)	Elimina las filas que contienen menos de 5 valores que no faltan
fillna(0)	Reemplaza los valores faltantes por ceros
isnull()	Devuelve True si falta el valor
notnull()	Devuelve True para los valores que no faltan


BIG DATA

Big Data

- **Big Data** se refiere a la gestión y análisis de grandes volúmenes de datos que son demasiado complejos para las herramientas tradicionales. Se caracteriza por las **cinco V's**:

Volumen: Grandes cantidades de datos generados de múltiples fuentes.

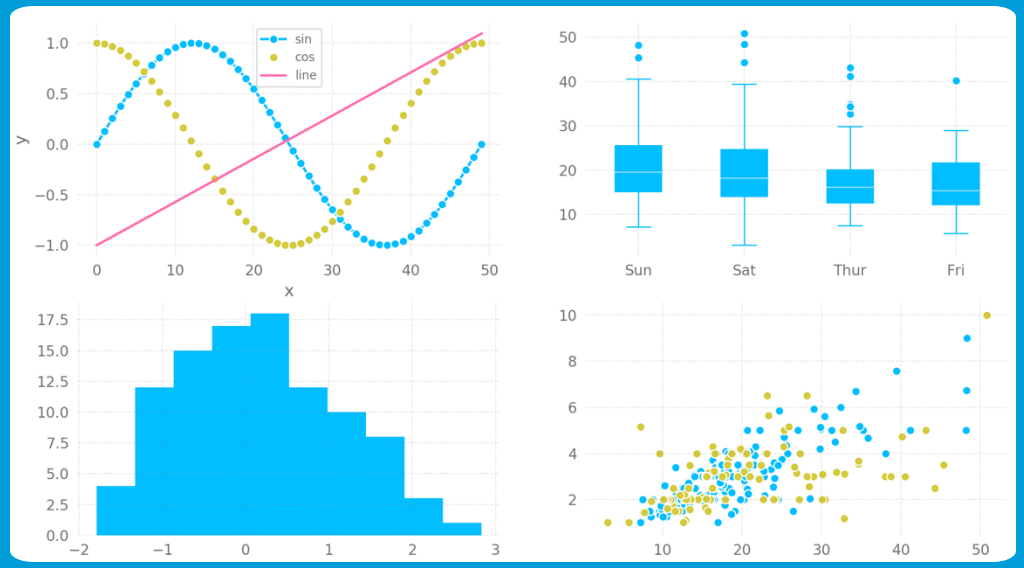
Velocidad: La rapidez con la que se crean y procesan los datos.

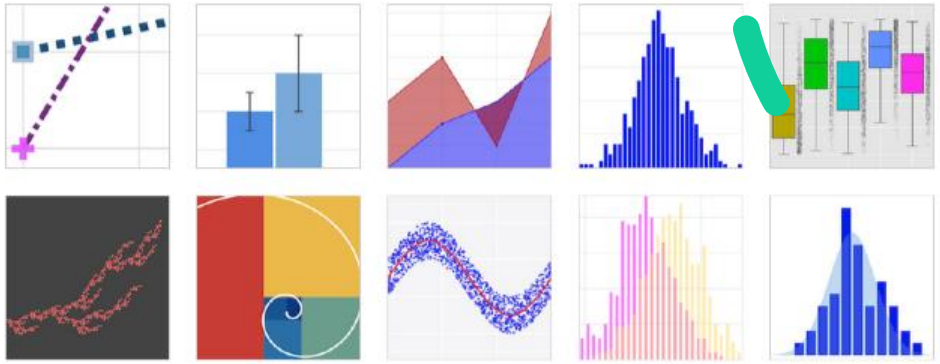
Variedad: Diversidad en tipos y formatos de datos (estructurados y no estructurados).

Veracidad: La importancia de la calidad y precisión de los datos.

Valor: La capacidad de extraer información útil para la toma de decisiones.

2. Visualización de datos con

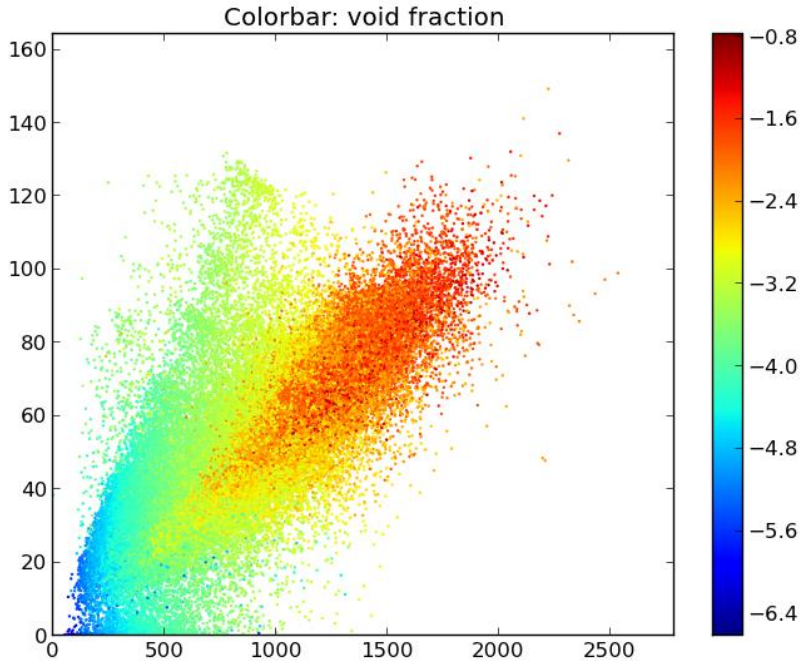




Tipos de gráficos

El tipo de gráfico es un aspecto muy importante, debido a que es el mensaje que se proyectará. Entonces, se deber seleccionar el gráfico de acuerdo a los datos con los que se cuente .

Gráficos de dispersión (scatter)



- Mostrar variables cuantitativas en un intervalo específico.
- Los puntos se deberán distinguir claramente.
- Si existe más de una categoría de deberá utilizar colores y formas para hacer las distinción entre ambas categorías.

Diagrama Dispersión

Utilizar dataframe salary.csv para generar un gráfico de dispersión.

```
# importar la librería de pandas como pd
import pandas as pd

#Leer el dataframe salaries.csv
df = pd.read_csv("https://raw.githubusercontent.com/ulises1229/INTRO-PYTHON-ENESJ/master/data/salaries.csv")
df.head()
```

Diagrama Dispersión

```
# Almacenar variables años y salario
x = df.service
y = df.salary

# Importar librería Matplotlib
import matplotlib.pyplot as plt

# Generar diagrama de dispersión
plt.scatter(x, y)

# Títulos: gráfico y ejes
plt.title("Gráfico de Dispersión Años de servicio vs Salario")
plt.xlabel('Años de servicio')
plt.ylabel('Salario Anual')

# Desplegar imagen en pantalla
plt.show()
```

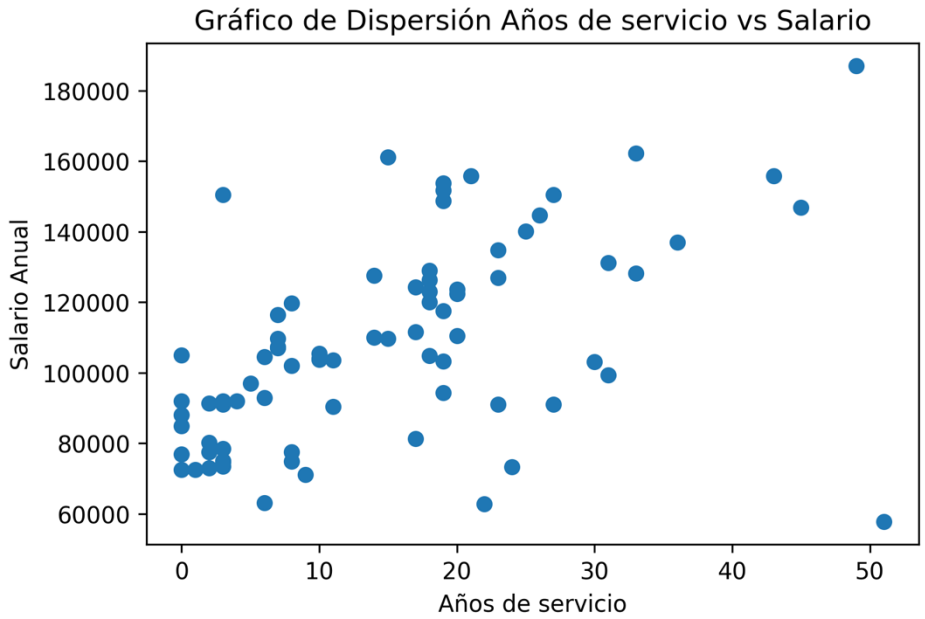


Diagrama Dispersión

- Agregar colores y tamaño.

```
# Importar librería de numpy
import numpy as np

# Almacenar variables x = años servicio y y = salario
x = df.service
y = df.salary

# Arreglo de colores
colores = np.arange(len(x))
print(colores)

# Agregar un scatter con colores y transparencia
plt.scatter(x, y, c=colores, alpha = 0.5, s = (y*0.002))
```

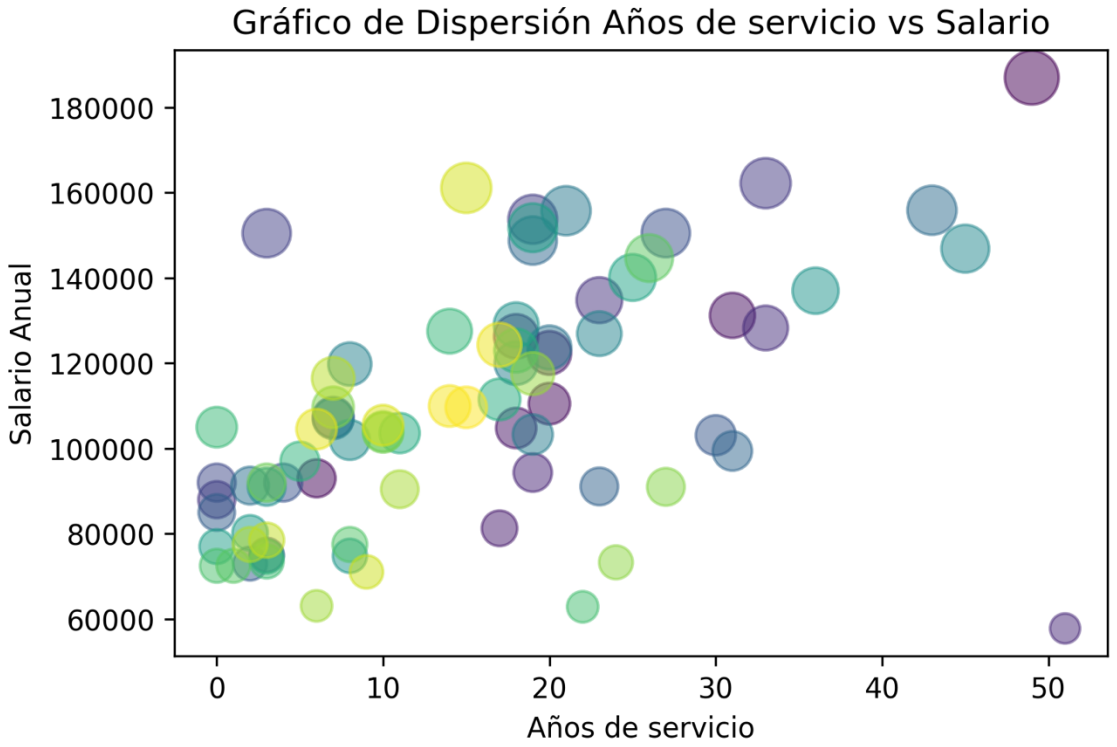


Diagrama Dispersión

- Agregar colores, tamaño y leyenda.

```
# Almacenar variables x = años servicio y y = salario
x = df.service
y = df.salary
lab = ['Male', 'Female']

# Arreglo de colores
colores = np.arange(len(x))

# Títulos: gráfico y ejes
plt.title("Gráfico de Dispersión Años de servicio vs Salario")
plt.xlabel('Años de servicio')
plt.ylabel('Salario Anual')

# Agregar un scatter con colores, transparencia y leyenda
plt.scatter(x, y, c=colores, alpha = 0.5, s = (y*0.002), label = "datos")
plt.legend(loc='best')

plt.show()
```

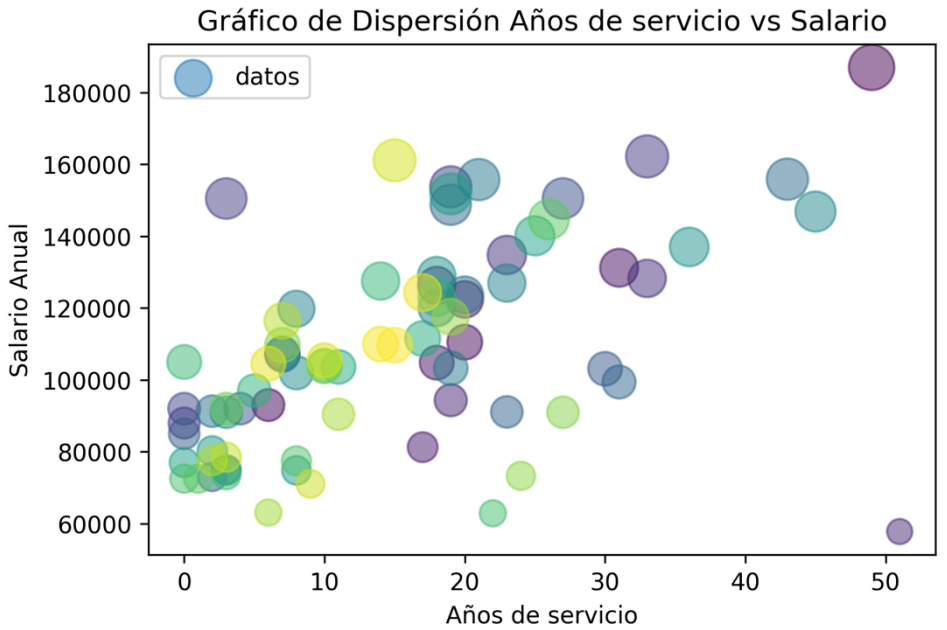
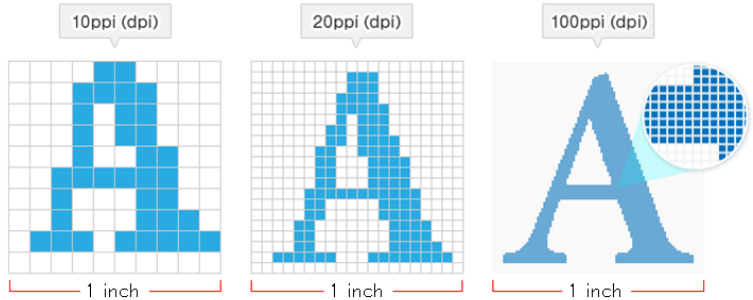
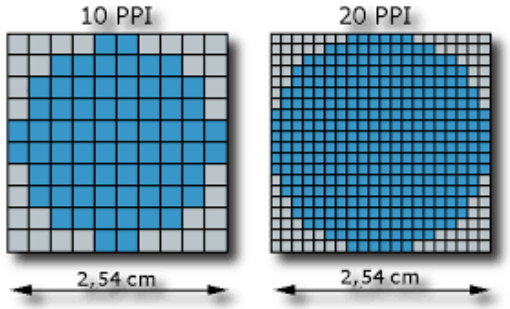


Diagrama de Dispersión – Exportando un Gráfico

Con matplotlib es posible exportar imágenes de alta calidad. Para esto es necesario definir elementos tales como:

- Título de gráfico
- Títulos de los ejes
- Título de la leyenda






```
# guarda la imagen en un archivo
plt.savefig("dispersión.tiff", dpi=300, quality = 95, bbox_inches='tight')
```

MPG dataframe

El data frame cuenta con 234 filas, 38 modelos de automóviles (1999 - 2008)
11 columnas (variables).

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
1	audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact
2	audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
3	audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact
4	audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact
5	audi	a4	2.8	1999	6	auto(l5)	f	16	26	p	compact
6	audi	a4	2.8	1999	6	manual(m5)	f	18	26	p	compact
7	audi	a4	3.1	2008	6	auto(av)	f	18	27	p	compact
8	audi	a4 quattro	1.8	1999	4	manual(m5)	4	18	26	p	compact
9	audi	a4 quattro	1.8	1999	4	auto(l5)	4	16	25	p	compact
10	audi	a4 quattro	2.0	2008	4	manual(m6)	4	20	28	p	compact
11	audi	a4 quattro	2.0	2008	4	auto(s6)	4	19	27	p	compact
12	audi	a4 quattro	2.8	1999	6	auto(l5)	4	15	25	p	compact
13	audi	a4 quattro	2.8	1999	6	manual(m5)	4	17	25	p	compact
14	audi	a4 quattro	3.1	2008	6	auto(s6)	4	17	25	p	compact
15	audi	a4 quattro	3.1	2008	6	manual(m6)	4	15	25	p	compact

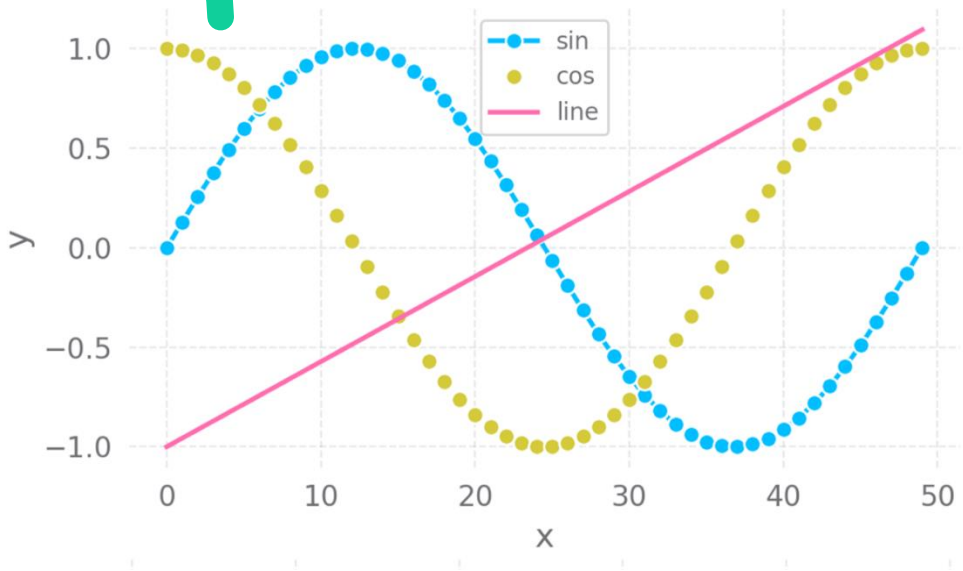
Ejercicio

- Emplear el dataframe mpg.csv
- Se deberá generar un gráfico de dispersión del desplazamiento del motor vs rendimiento en autopista.

1. Se deberá generar un segundo gráfico de desplazamiento del motor vs rendimiento en ciudad.
2. Exportar un gráfico con una resolución de 300 dpi

Gráficos de Línea

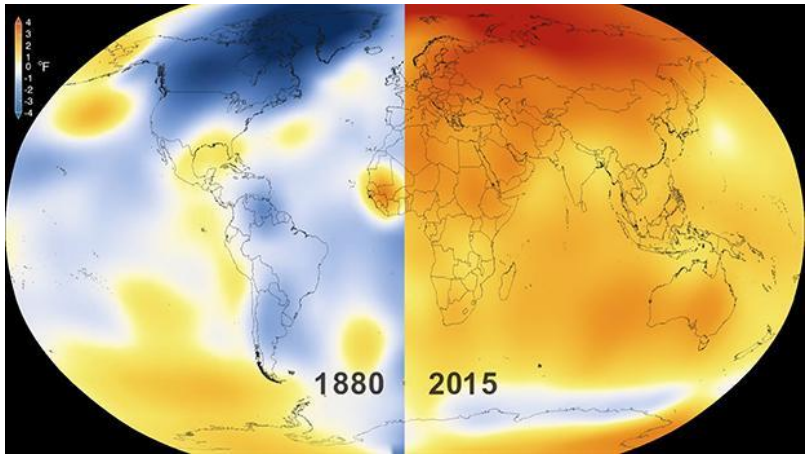
- Representar valores cuantitativos en función de la variable independiente.
- Es posible combinar la gráfica con puntos, para enfatizar valores específicos.
- Si existe más de una categoría en la gráfica, deberán utilizarse leyendas y tipos de línea para diferenciar los datos.

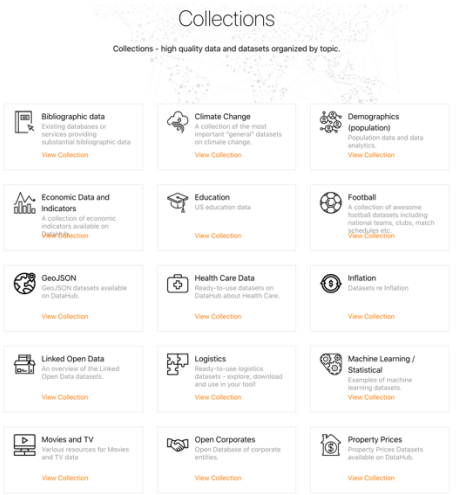


Gráficos de línea

- Se deberá importar el dataframe temp.csv (Repositorio GH)

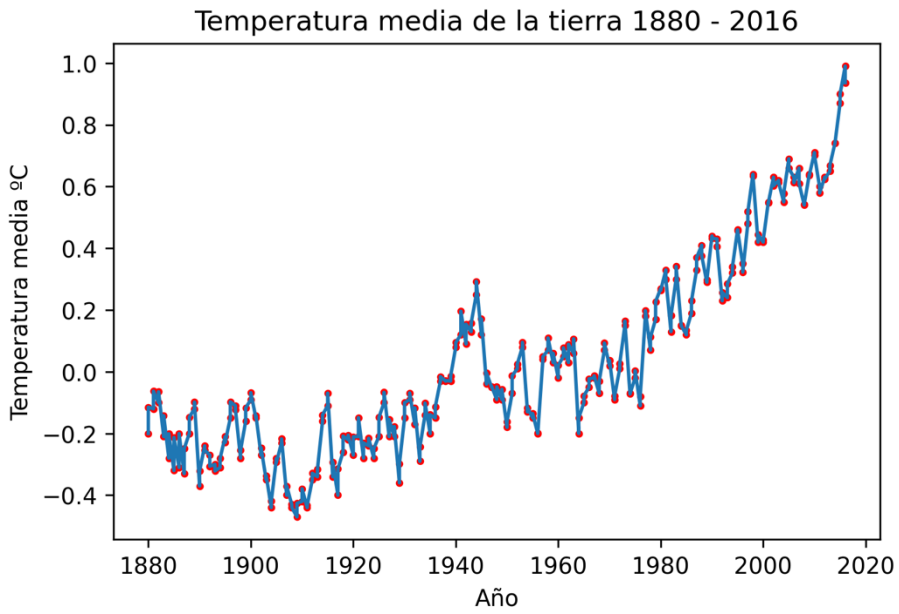
```
# Importar pandas y matplotlib
import matplotlib.pyplot as plt
import pandas as pd

# Importar data frame de temperatura
df_temp = pd.read_csv('https://raw.githubusercontent.com/ulises1229/INTRO-PYTHON-ENESJ/master/data/temp.csv', sep =',')

df_temp
```





Gráficos de línea

```python
# Importar pandas y matplotlib
import matplotlib.pyplot as plt
import pandas as pd

# Importar data frame de temperatura
df_temp = pd.read_csv('https://raw.githubusercontent.com/ulises1229/INTRO-PYTHON-ENESJ/master/data/temp.csv', sep =',')

df_temp

# Separa dos variables del dataframe
año = df_temp.Year
temp = df_temp.Mean

# Generar el diagrama de línea
plt.plot(año,temp)

#Agregar un scatter
plt.scatter(año, temp, s = 5, c = "red")

# Agregar títulos
plt.title("Temperatura media de la tierra 1880 - 2016")
plt.xlabel('Año')
plt.ylabel('Temperatura media ºC')

#Guardar el el gráfico en un archivo
plt.savefig("line_temp.tiff", dpi=300, quality = 95, bbox_inches='tight')
```



Ejercicio 3

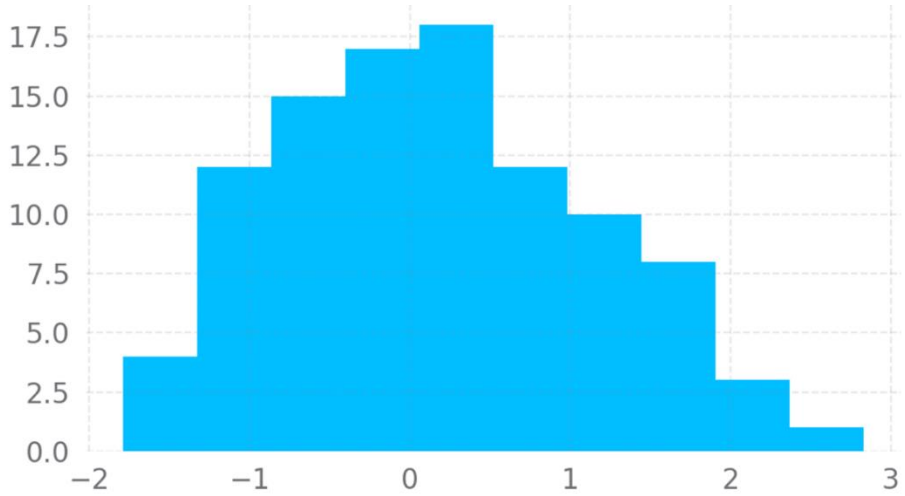
Emplear el dataframe **co2.csv**

Está en el repositorio de GitHub

1. Se deberá generar un gráfico de línea con fecha vs Interpolated.
2. Se deberá generar un gráfico de línea con fecha vs Trend.
3. Se deberá exportar el gráfico con una calidad de 300 dpi.

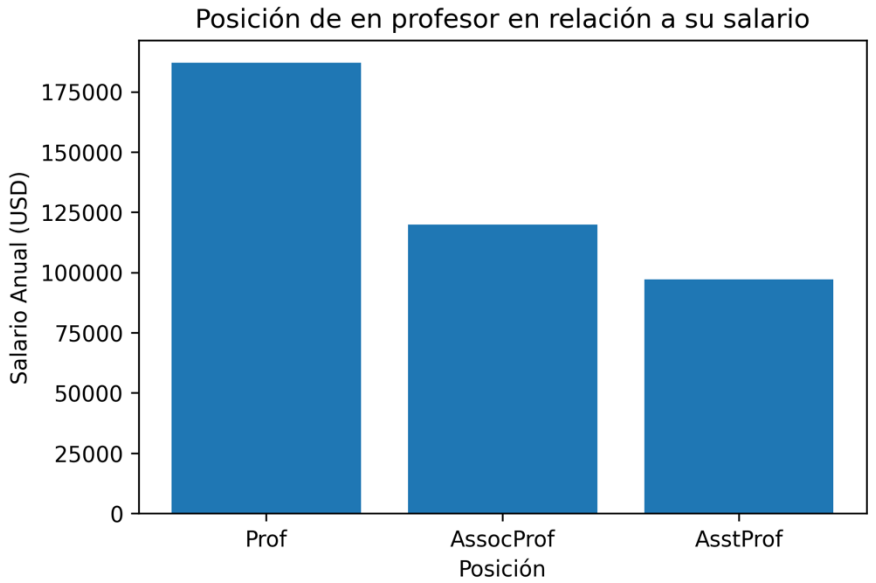
Gráficos de barras

- Mostrar comparaciones, desviaciones o rankings. Puede estar en formato vertical u horizontal.
- Utilizar solo un color, a menos que haya más de una categoría. En dado caso se usan leyendas.
- Evitar utilizar bordes para las barras.
- Distancia entre barras, deberá ser suficiente para que no se traslapen.



Gráficos de barras

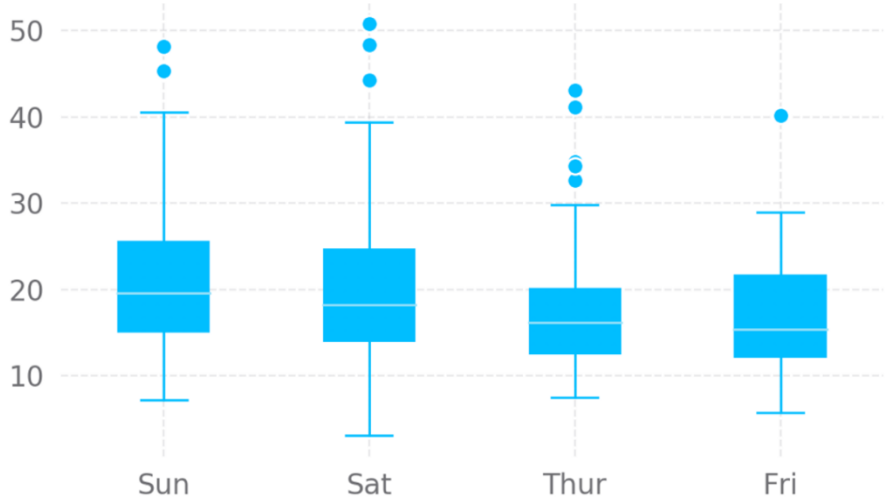
```python
# Importar pandas y matplotlib
import matplotlib.pyplot as plt
import pandas as pd

#Leer el dataframe salaries.csv
df = pd.read_csv("https://raw.githubusercontent.com/ulises1229/INTRO-PYTHON-ENESJ/master/data/salaries.csv")

# Recuperar rank y salario
rank = df['rank']
sueldo = df.salary

# Generar una gráfica de barras
plt.bar(rank, sueldo)

# Agregar títulos
plt.title("Posición de en profesor en relación a su salario")
plt.xlabel("Posición")
plt.ylabel("Salario Anual (USD)")

plt.savefig("salario.tiff", dpi=300, quality = 95, bbox_inches='tight')
```



Boxplots

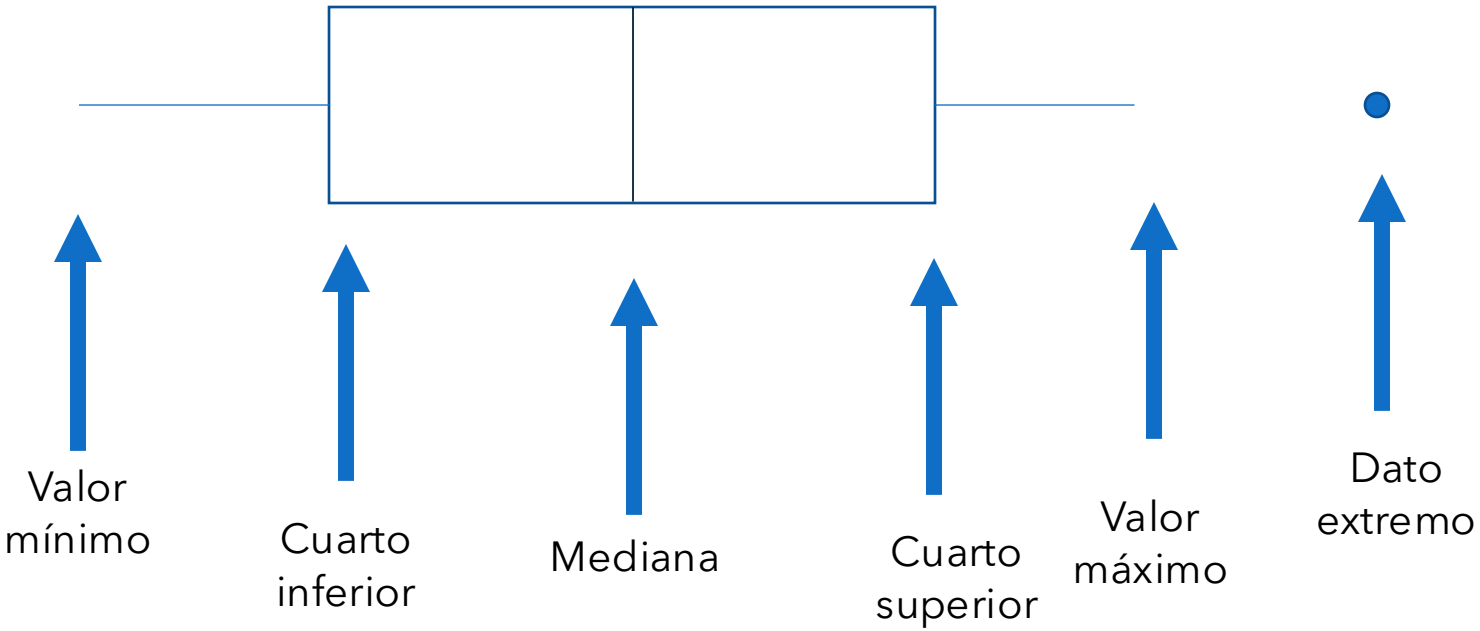
- y variable numérica
- x variable nominal

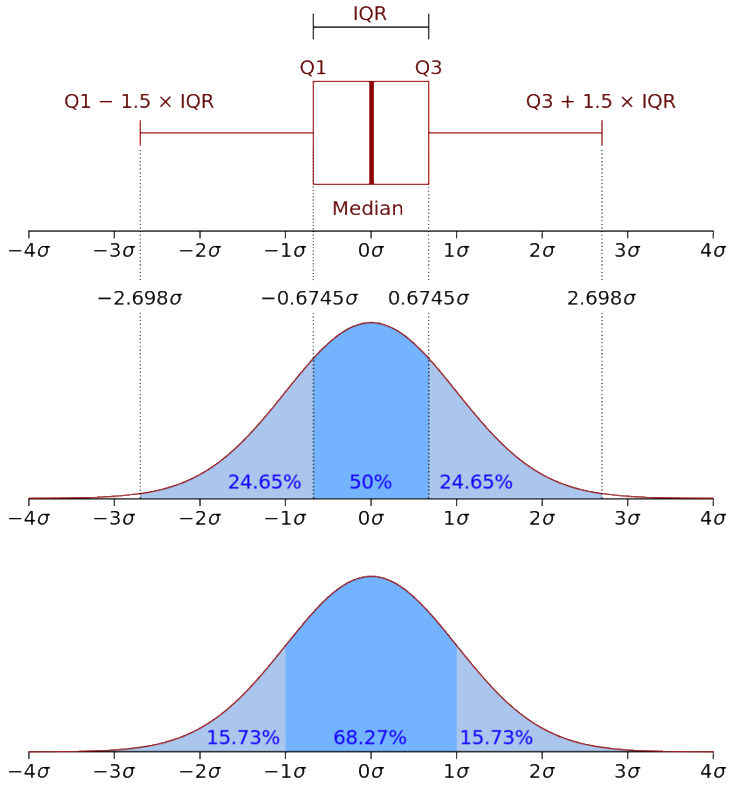
  (categórica, objetos)



Boxplots

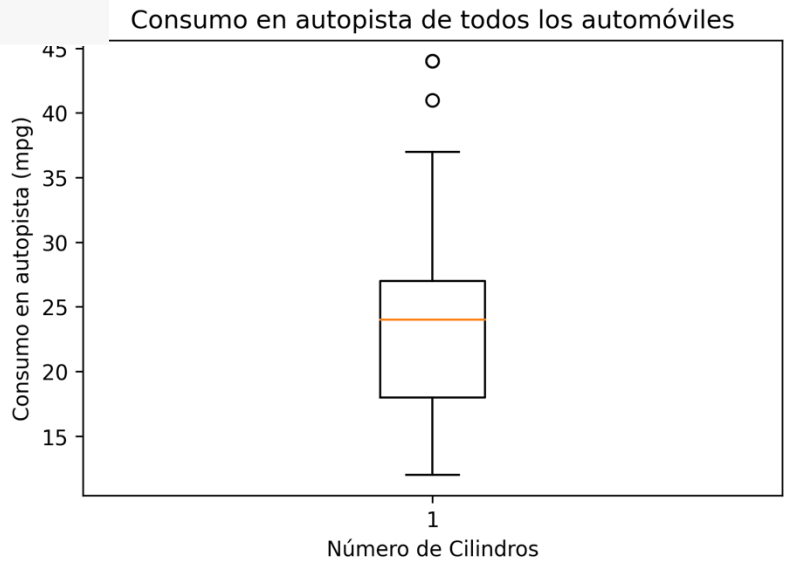
Cuartiles: Valores que dividen la muestra en 4 partes iguales





Boxplots

```python
# Importar pandas y matplotlib
import matplotlib.pyplot as plt
import pandas as pd

# importar ma

#Leer el dataframe salaries.csv
mpg = pd.read_csv("https://raw.githubusercontent.com/ulises1229/INTRO-PYTHON-ENESJ/master/data/mpg.csv")

plt.boxplot(mpg.hwy)
# Agregar títulos
plt.title("Consumo en autopista de todos los automóviles")
plt.xlabel("Número de Cilindros")
plt.ylabel("Consumo en autopista (mpg)")

# exportar imágen
plt.savefig("mpg.tiff", dpi=300, quality = 95, bbox_inches='tight')
```



3. Análisis Exploratorio de Datos