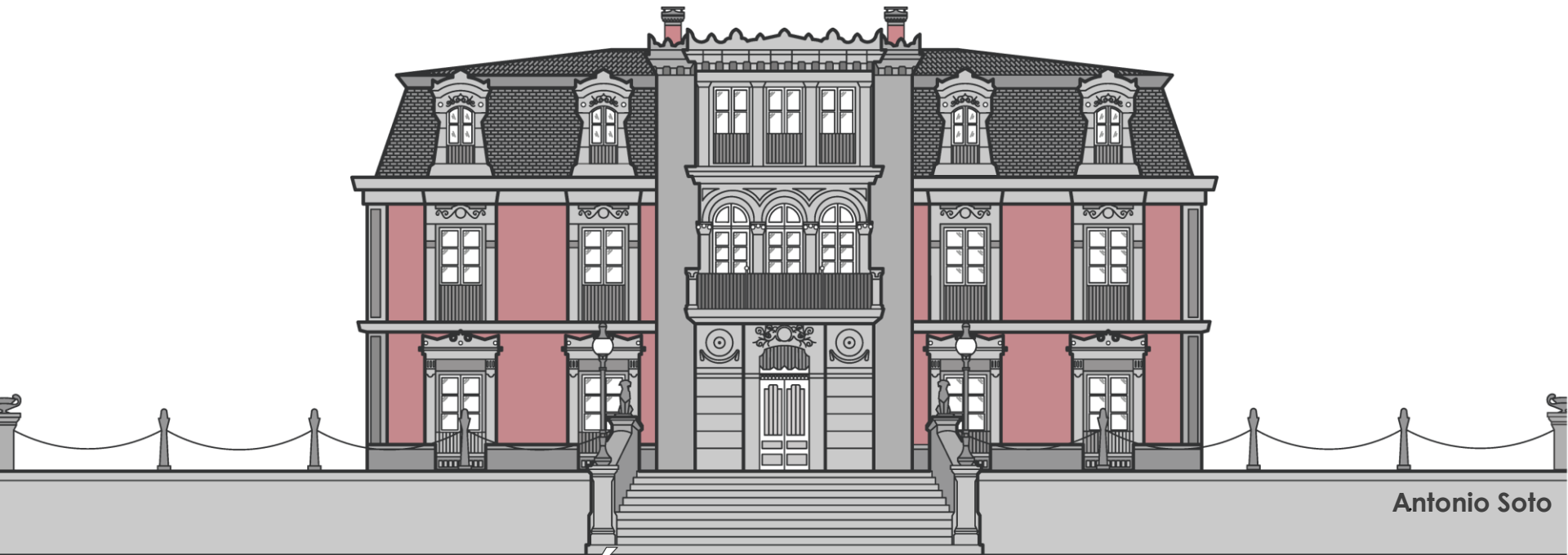


# Lenguajes de Análisis de Información: PYTHON 3.x



Antonio Soto

**MÁSTER IFFE B.S.**

Director Verne Tech



Director

Verne Tech



637.505.941



ajsoto@vernegroup.com



<https://www.linkedin.com/in/antoniosql>

# ¡HOLA!



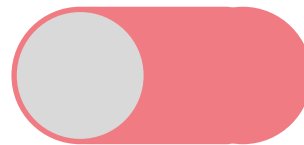
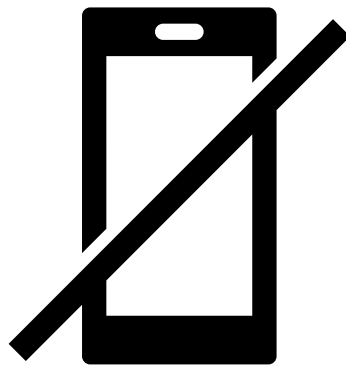
Soy Antonio Soto, llevo más de 20 años dedicado al mundo del análisis de datos, desde los primeros sistemas Data Warehouse, hasta el desarrollo de soluciones integradas con Inteligencia Artificial de hoy en día, pasando por todos los puntos intermedios. Me he tocado viajar por el mundo diseñando sistemas y soluciones enfocadas a la toma de decisiones en todos los sectores y en empresas de diferentes tamaños.



<https://www.amazon.es/dp/B08G55PK1X>

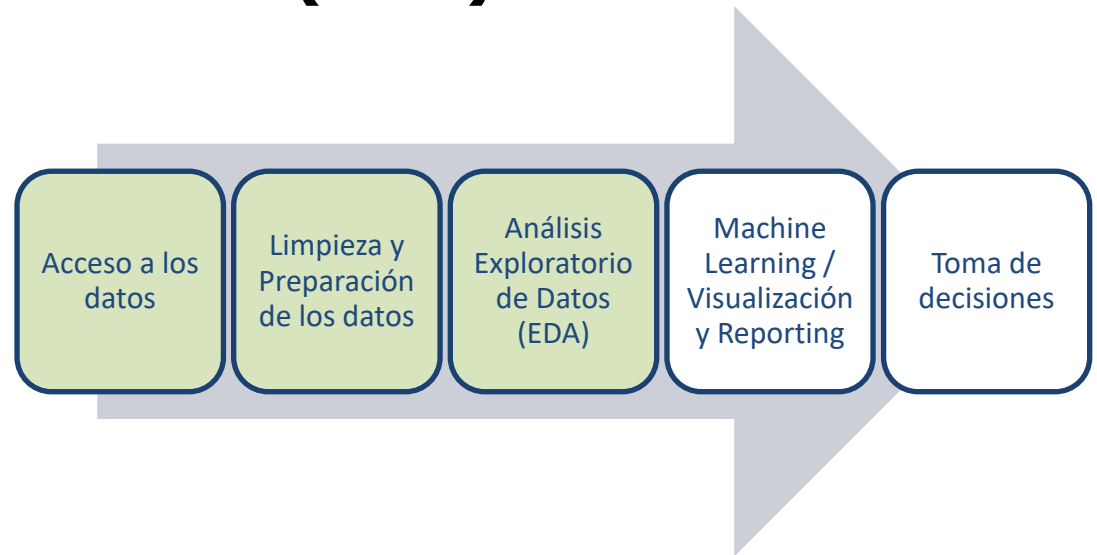
Por favor, durante la sesión,  
no te olvides...

**¡ APAGA EL MÓVIL !**



¿Qué  
aprenderemos?

1. INTRODUCCIÓN A PYTHON
2. LIMPIEZA Y PREPARACIÓN DE DATOS
3. ANÁLISIS EXPLORATORIO DE DATOS (EDA)



1

# INTRODUCCIÓN A PYTHON

¿Qué es Python?

Entornos de Desarrollo

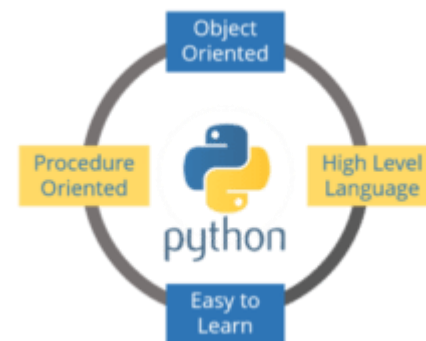
Estructura y Elementos del Lenguaje

Módulos, paquetes y espacios de nombres

Python Standard Library

# 1 Introducción a Python: ¿Qué es Python?

- Fácil de aprender
  - En 8 de cada 10 programas superiores en USA
- Completo
  - No solo estadística
- Librerías Data Science
- Orientado a Objetos
- Libre, gratuito y multiplataforma
- Lenguaje de Alto nivel
  - Fácil de leer por personas



# 1 Introducción a Python: ¿Qué es Python?

---

- Distribuciones
  - [Www.Python.org](http://www.python.org) CPython
  - Anaconda
  - ActivePython
  - WinPython
- Python Software Foundation
  - <https://www.python.org/psf-landing/>



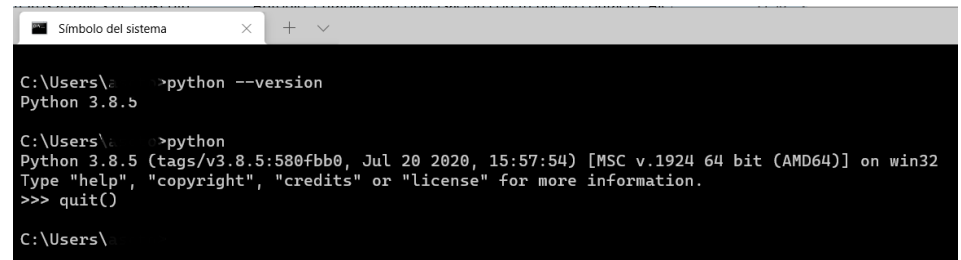
# 1 Introducción a Python: ¿Qué es Python?

- Instalación

- Descarga desde <https://www.python.org/downloads>
- Comprobamos instalación en consola
  - MacOS: `python3 --version`
  - Windows: `python --version`

- Shell

- MacOS: `python3`
- Windows: `python`



```
Símbolo del sistema
C:\Users\>python --version
Python 3.8.5

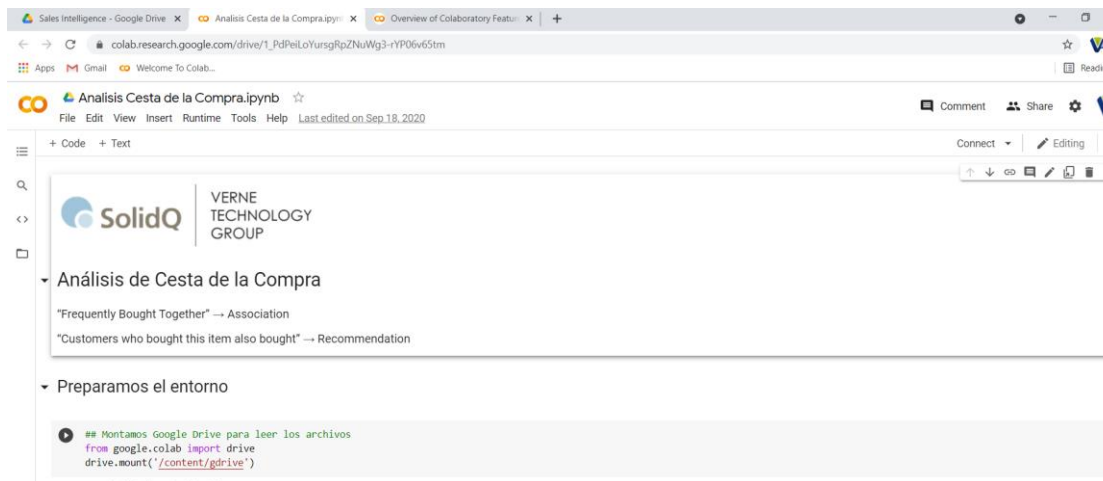
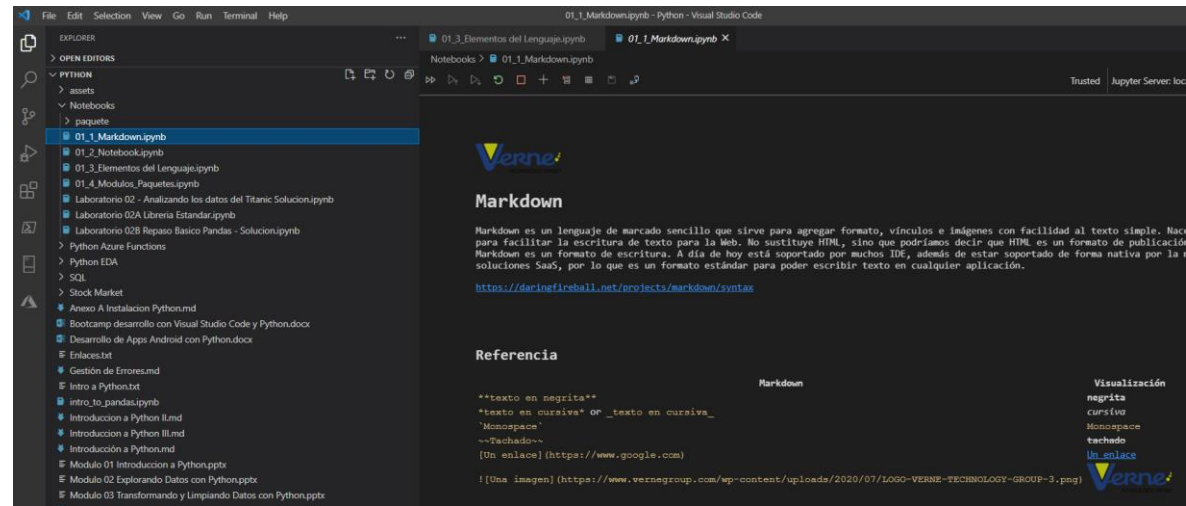
C:\Users\>python
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> quit()

C:\Users\
```

# 1 Introducción a Python: Entornos de Desarrollo

IDE

Notebook



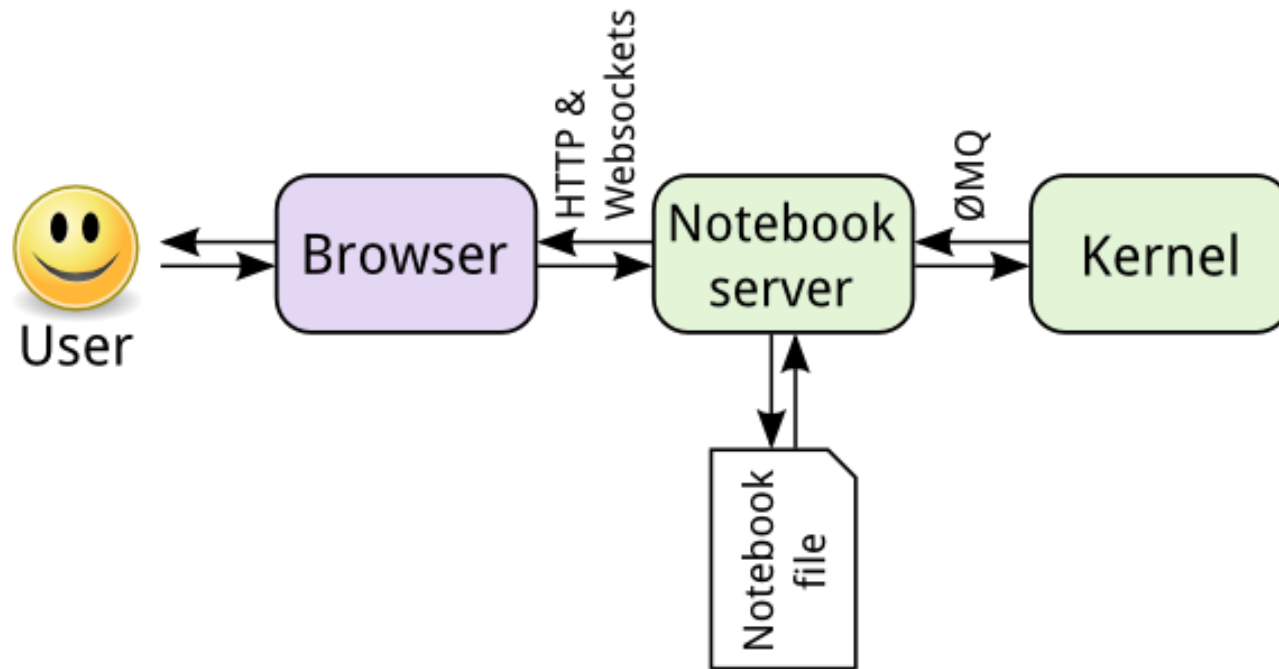
Python

# 1 Introducción a Python: Entornos de Desarrollo NOTEBOOKS



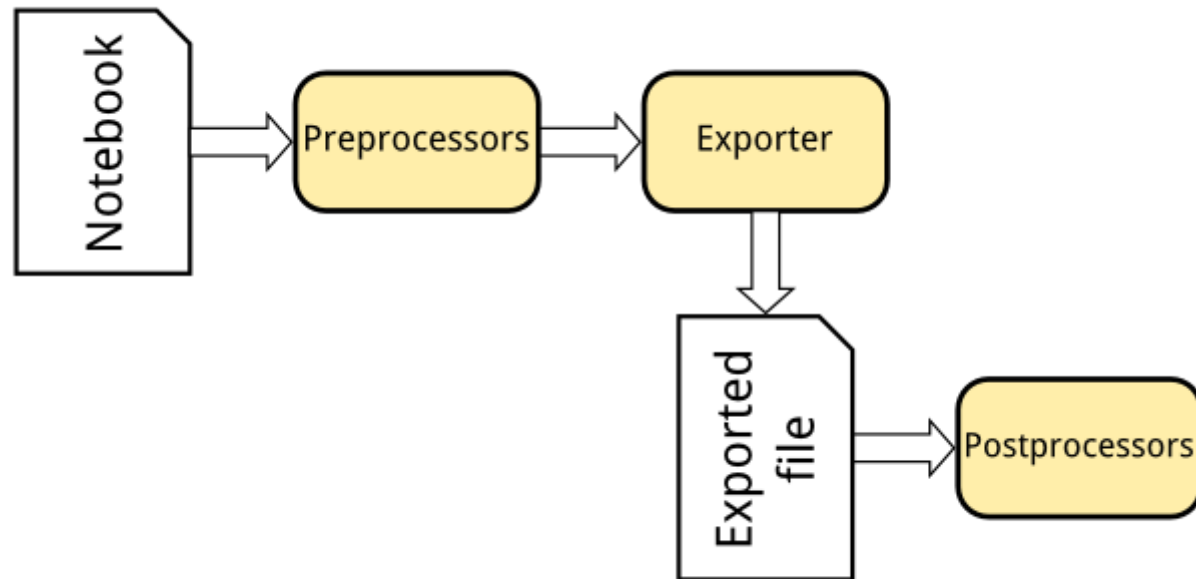
- REPL – Read-Evaluate-Print-Loop
- Prototipo, desarrollo rápido, exploración,... → Interactivo
- Colaboración en equipos
- Kernel Ipython

# 1 Introducción a Python: Entornos de Desarrollo Arquitectura NOTEBOOKS



# 1 Introducción a Python: Entornos de Desarrollo

## Exportación NOTEBOOKS



# 1 Introducción a Python: Entornos de Desarrollo

## Entornos para NOTEBOOKS

Microsoft Azure Notebooks Preview Solidq

Libraries What's New Status Help

SolidQ

Solidq > Libraries > SolidQ

Run + New Settings Share Clone 0 Clones ☆ Star (0) Terminal Shutdown Preview Edit File Download Delete

Search Show hidden items

FILE NAME	FILE TYPE	MODIFIED
Motor Recomendacion R.ipynb	Notebook	Jul 4, 2018
README.md	Markdown	Mar 29, 2018
Serie de tiempo con Pandas.ipynb	Notebook	Mar 31, 2018
u.data	DATA	Jul 4, 2018

Showing 4 search results (1 hidden)

< 1 >

https://colab.research.google.com/notebooks/welcome.ipynb

Hola, Colaboratory

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

CÓDIGO TEXTO CELDA CELDA COPIAR EN DRIVE

Conectar Edición

Te damos la bienvenida a Colaboratory

Colaboratory es un entorno gratuito de Jupyter Notebook que no requiere configuración y que se ejecuta completamente en la nube. Puedes consultar más información en la sección de [preguntas frecuentes](#).

Introducción

- Descripción general de Colaboratory
- Cargar y guardar datos: archivos locales, Drive, Hojas de cálculo y Google Cloud Storage
- Importar bibliotecas e instalar dependencias
- Usar Google Cloud BigQuery
- Formularios, Gráficos, Markdown y Widgets
- TensorFlow con GPU
- Curso intensivo de aprendizaje automático: Introducción a Pandas y Primeros pasos con TensorFlow

Funciones destacadas

Ejecución de TensorFlow

Colaboratory permite ejecutar código de TensorFlow en el navegador con un solo clic. En el siguiente ejemplo se añaden dos matrices.

$$\begin{bmatrix} 1. & 1. & 1. \\ 1. & 1. & 1. \end{bmatrix} + \begin{bmatrix} 1. & 2. & 3. \\ 4. & 5. & 6. \end{bmatrix} = \begin{bmatrix} 2. & 3. & 4. \\ 5. & 6. & 7. \end{bmatrix}$$

# 1 Introducción a Python: Entornos de desarrollos. Markdown


Markdown es un lenguaje de marcado sencillo que sirve para agregar formato, vínculos e imágenes con facilidad al texto simple

## Referencia

### Markdown

```
**texto en negrita**  
*texto en cursiva* or _texto en cursiva_  
`Monospace`  
~~Tachado~~  
[Un enlace] (https://www.google.com)  
[Una imagen] (https://www.vernegroup.com/wp-content/uploads/2020/07/LOGO-VERNE-TECHNOLOGY-GROUP-3.png)
```

### Visualización

***negrita***  
*cursiva*  
`Monospace`  
~~tachado~~  
[Un enlace](#)  


# 1 Demostración

---

01\_1\_Markdown

01\_2\_Notebook



# 1 Introducción a Python: Estructuras y Elementos del Lenguaje

---

- Ejecución de código en Notebooks
- Tipos de Datos y Operaciones
- Control de Ejecución
- Excepciones

# 1 Introducción a Python: Módulos, Paquetes y Espacios de Nombre. Módulos en Python

```
└─ paquete
   └─ __init__.py
   └─ modulo1.py
   └─ modulo2.py
   └─ modulo3.py
```

```
import modulo # importar un módulo
import paquete.modulo1 # importar un módulo que está de
ntro de un paquete
import paquete.subpaquete.modulo1 # importar un módulo
que está dentro de un subpaquete

# Si las rutas (lo que se conoce como "namespace" son l
argas, se pueden generar alias por medio del modificado
"as"

import modulo as m
import paquete.modulo1 as pm
import paquete.subpaquete.modulo1 as psm
```

# 1 Introducción a Python: Python Standard Library

```
import os
os.getcwd() #directorio de trabajo actual
os.chdir('/home/nbuser/') #cambia el directorio
os.system('mkdir today') #ejecuta el comando 'mkdir'
dir(os) #lista de todas las funciones del módulo
help(os) #devuelve un manual de ayuda
```

## Funciones

<a href="#"><u>abs()</u></a>	<a href="#"><u>delattr()</u></a>	<a href="#"><u>hash()</u></a>	<a href="#"><u>memoryview()</u></a>	<a href="#"><u>set()</u></a>
<a href="#"><u>all()</u></a>	<a href="#"><u>dict()</u></a>	<a href="#"><u>help()</u></a>	<a href="#"><u>min()</u></a>	<a href="#"><u>setattr()</u></a>
<a href="#"><u>any()</u></a>	<a href="#"><u>dir()</u></a>	<a href="#"><u>hex()</u></a>	<a href="#"><u>next()</u></a>	<a href="#"><u>slice()</u></a>
<a href="#"><u>ascii()</u></a>	<a href="#"><u>divmod()</u></a>	<a href="#"><u>id()</u></a>	<a href="#"><u>object()</u></a>	<a href="#"><u>sorted()</u></a>
<a href="#"><u>bin()</u></a>	<a href="#"><u>enumerate()</u></a>	<a href="#"><u>input()</u></a>	<a href="#"><u>oct()</u></a>	<a href="#"><u>staticmethod()</u></a>
<a href="#"><u>bool()</u></a>	<a href="#"><u>eval()</u></a>	<a href="#"><u>int()</u></a>	<a href="#"><u>open()</u></a>	<a href="#"><u>str()</u></a>
<a href="#"><u>breakpoint()</u></a>	<a href="#"><u>exec()</u></a>	<a href="#"><u>isinstance()</u></a>	<a href="#"><u>ord()</u></a>	<a href="#"><u>sum()</u></a>
<a href="#"><u>bytearray()</u></a>	<a href="#"><u>filter()</u></a>	<a href="#"><u>issubclass()</u></a>	<a href="#"><u>pow()</u></a>	<a href="#"><u>super()</u></a>
<a href="#"><u>bytes()</u></a>	<a href="#"><u>float()</u></a>	<a href="#"><u>iter()</u></a>	<a href="#"><u>print()</u></a>	<a href="#"><u>tuple()</u></a>
<a href="#"><u>callable()</u></a>	<a href="#"><u>format()</u></a>	<a href="#"><u>len()</u></a>	<a href="#"><u>property()</u></a>	<a href="#"><u>type()</u></a>
<a href="#"><u>chr()</u></a>	<a href="#"><u>frozenset()</u></a>	<a href="#"><u>list()</u></a>	<a href="#"><u>range()</u></a>	<a href="#"><u>vars()</u></a>
<a href="#"><u>classmethod()</u></a>	<a href="#"><u>getattr()</u></a>	<a href="#"><u>locals()</u></a>	<a href="#"><u>repr()</u></a>	<a href="#"><u>zip()</u></a>
<a href="#"><u>compile()</u></a>	<a href="#"><u>globals()</u></a>	<a href="#"><u>map()</u></a>	<a href="#"><u>reversed()</u></a>	<a href="#"><u>__import__()</u></a>
<a href="#"><u>complex()</u></a>	<a href="#"><u>hasattr()</u></a>	<a href="#"><u>max()</u></a>	<a href="#"><u>round()</u></a>	

- PIP: Gestor de Paquetes
- Obtiene los paquetes del repositorio <https://pypi.org/>  
*pip install <paquete>*
- Entornos Virtuales  
*python -m venv <nombreEntorno>*  
*<nombreEntorno>\Scripts\actíivate*  
*deactivate*
- Requisitos
  - *pip freeze > requirements.txt*
  - *pip install -r requirements.txt*

# 1 Demostración

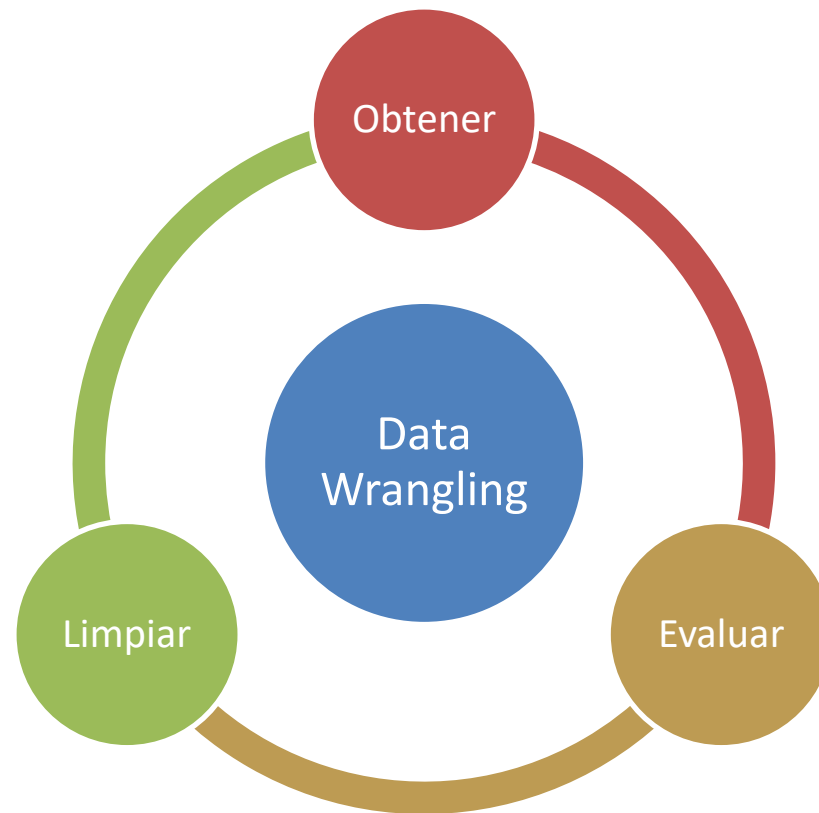
---

01\_3\_Elementos del Lenguaje

01\_4\_Módulos\_Paquetes

2

# LIMPIEZA Y PREPARACIÓN DE DATOS





## 2 Limpieza y Preparación de Datos: DATAFRAMES

Disponibles en R y Python  
Pandas (También Spark)

### Tablas

- Cada columna de un tipo

### Tareas Comunes:

- Crear subconjuntos por filas y columnas
- Filtrado lógico de filas y columnas

Column1	Column2	...	ColumnN
1	ABC	...	12.2
2	XYZ	...	13.1
3	ABC	...	12.8
4	XYZ	...	10.9
5	ABC	...	3.75

## 2

# Limpieza y Preparación de Datos: PANDAS

```
import pandas as pd  
import os  
dir = "c:\data"  
file = "values.csv"  
path = os.path.join(dir, file)  
frame1 = pd.read_csv(path)
```

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47



## 2

# Limpieza y Preparación de Datos: PANDAS

Seleccionar una columna

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47

```
frame1 = frame1["Col2"]
```

Col2
14
13
34
23

## 2

## Limpieza y Preparación de Datos: PANDAS

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47

```
frame1 = frame1[["Col1", "Col2"]]
```

Col1	Col2
2012	14
2013	13
2013	34
2014	23

## Filtrar

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47

```
frame1 = frame1[1:3:1]
```

Col1	Col2	Col3
2013	13	76
2013	34	65

Filtrar n° de filas

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47

```
frame1 = frame1[:3]
```

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65

## Filtrar por fila y columna

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47

```
frame1 = frame1["Col2"][1:2]
```

Col2
13
34



## Agregar Columna Calculada

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47

```
frame1["Col4"] = frame1["Col2"] + frame1["Col3"]
```

Col1	Col2	Col3	Col4
2012	14	45	59
2013	13	76	89
2013	34	65	99
2014	23	47	70

## Eliminar una columna

Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47

```
frame1.drop("Col3", axis=1, inplace=True)
```

Col1	Col2
2012	14
2013	13
2013	34
2014	23

## GroupBy

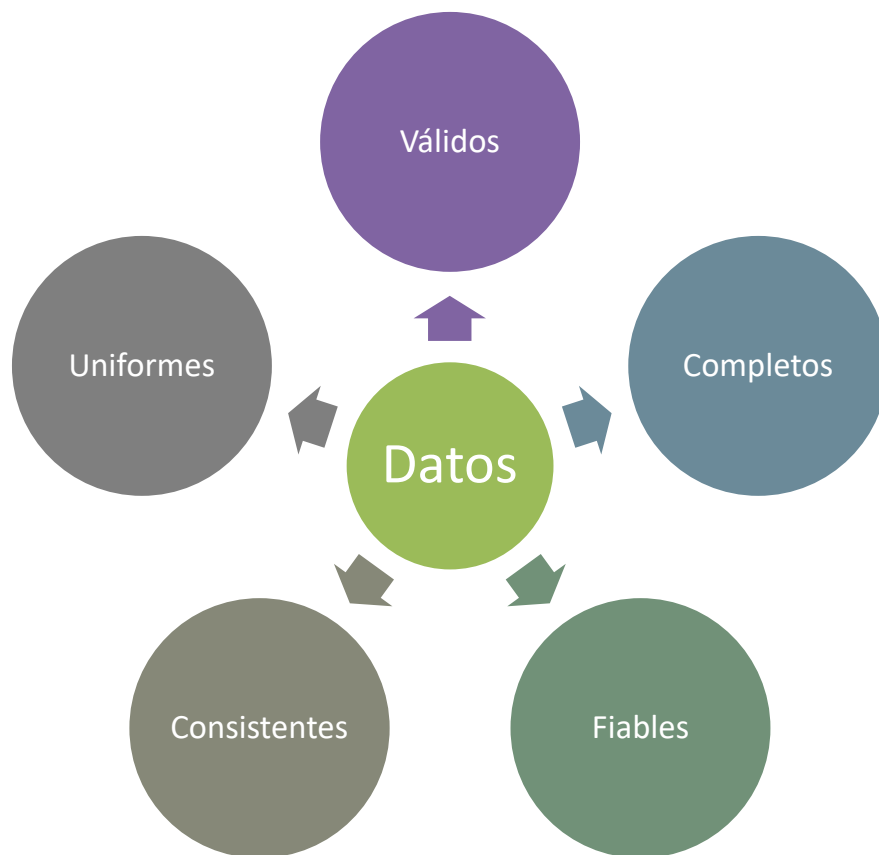
Col1	Col2	Col3
2012	14	45
2013	13	76
2013	34	65
2014	23	47

```
frame1= frame1.groupby("Col1").sum()
```

Col2	Col3
14	45
47	141
23	47

## 02\_01\_Repaso Básico Pandas

¿Y todo esto para qué?



## 2 Limpieza y Preparación de Datos: Visualización

---

- Explorar los datos con visualización
- Entender las relaciones entre los datos
- Crear múltiples vistas de los datos
- Comprender las fuentes de posibles errores

## 2 Limpieza y Preparación de Datos: Visualización

---

- Las relaciones en los datos pueden ser complejas
- La exploración de datos requiere de múltiples vistas
- Las Vistas revelan diferentes aspectos de las relaciones
- Diferentes tipos de plots muestran diferentes tipos de relaciones

## 2 Limpieza y Preparación de Datos: Visualización

- matplotlib es la librería por referencia para gráficas en Python
- e.g. matplotlib.pyplot
- pandas.DataFrame.plot construido sobre matplotlib.pyplot
- Existen otras muchas librerías creadas sobre matplotlib
- Para algunos tipos específicos o más control debemos utilizar matplotlib.pyplot directamente



## 2 Limpieza y Preparación de Datos: Visualización

```
pandas.DataFrame.plot(kind = 'someType', ax =  
ax, ...)
```

'line' : line plot (default)

'bar' : vertical bar plot

'barh' : horizontal bar plot

'kde' or 'density': Kernel Density Estimation plot

'scatter' : scatter plot

## 2 Limpieza y Preparación de Datos: Visualización

---

ax – pyplot axis

x, y – coordenadas

color – color de línea o símbolo

s – tamaño por valor

Shape –figura

alpha – transparencia

## 2 Limpieza y Preparación de Datos: Visualización

### 1. Importar librerías

```
import matplotlib.pyplot as plt
```

### 2. Definir y borrar una figura

```
fig1 = plt.figure(figsize=(9, 9))  
fig1.clf()
```

### 3. Definir uno o más ejes

```
ax = fig1.gca()
```

### 4. Aplicar método plot

```
pandas.DataFrame.plot(kind = 'someType', ax = ax, ...)
```

### 5. Guardar figura

```
fig1.savefig('scatter2.png')
```

# 2

## Limpieza y Preparación de Datos: Preparación de Datos

---

No pueden faltar datos

Valores erróneos

Valores consistentes

## Muestras de Datos:

- Probar los conceptos en conjuntos pequeños y después extenderlos
- Muestreo Aleatorio
- Muestro de Fecha
- Selección de periodos

## Nulos y Repetidos:

- Tener repetidos y valores nulos es algo común
- Algunos algoritmos no puede manejar valores faltantes
- Los valores repetidos sesgan el resultado

## Nulos o valores que faltan:

- Eliminar filas
- Sustituir por un valor específico
- Interpolar valores
- Rellenar hacia adelante
- Rellenar hacia atrás
- Imputar

Python – `pandas.DataFrame.isna()`

## Valores Repetidos:

- Evaluar si hay patrón
- Sesgo

Python – `DataFrame.drop_duplicates()`



## Outliers y Errores

- Errores y outliers pueden sesgar el entrenamiento del modelo
- Existen múltiples fuentes de "errores"
  - Medidas erróneas
  - Errores de entrada
  - Valores traspuestos en la tabla
- Descubrirlos y evaluarlos con resúmenes estadísticos y visualización

## Outliers:

- Outliers de un punto
  - Observaciones anómalas con respecto a la mayoría de observaciones de una característica
- Outliers de Contexto
  - Observaciones consideradas anómalas para un contexto específico
- Outliers colectivos
  - Colección de observaciones anómalas pero que aparecen cerca de otra porque tiene un valor anómalo similar
- Visualización
  - Un matriz de plot de puntos ayuda a visualizar los outliers
  - Python – `pandas.tools.plotting.scatter_matrix`

## Eliminar Outliers:

Python: DataFrame = DataFrame[expression\_filtro]

```
frame1 = frame1[(frame1["Col1"] > 40.0) &  
                 (frame1["Col2"] < 30.0) &  
                 (frame1["Col3"] < 3.0)]
```

*Para grandes fuentes de datos podemos utilizar PyOD que es una librería especializada con más de 20 algoritmos para la detección de outliers*

## 02\_02\_Analizando los datos del Titanic

3

# ANÁLISIS EXPLORATORIO DE DATOS

EDA con Python

## Exploratory Data Analysis

### Univariate

#### Categorical

**Numerical**

- crosstab

**Visual**

- countplot

#### Continuous

**Numerical**

- describe

**Visual**

- boxplot

### Multivariate

#### Category to Category

**Visual**

- factorplot

#### Category to Continuous

**Visual**

- jointplot

#### Continuous to Category

**Visual**

- factorplot

Las variables continuas pueden tomar cualquier valor

- Temperatura
- Distancia
- Peso

Variables discretas tienen valores fijos

- Categóricas o Nominales: SI son solo etiquetas y no tienen un orden natural
- Ordinales o rangos: Si tienen un orden natural
- Ejemplos
  - ✓ Número de personas
  - ✓ Número de ruedas de un coche

## Variables Categóricas:

- Las Categorías son metadatos
- Demasiadas categorías pueden derivar en problemas
  - No disponemos de datos suficientes por categoría
  - Demasiadas dimensiones en un modelo
- A menudo necesitamos combinar categorías
  - Reducir el número de categorías
  - Grupos como categorías



## Cuantificar variables continuas:

- Convertir variables continuas en categóricas
  - No todas las variables continuas son “Números verdaderos” (True numeric)
- Agrupar valores dentro de categorías
  - Pequeño , mediano, grande
  - Caliente, Frío
  - Grupos de ingresos

03\_01\_EDA Inicial

03\_02\_Nulos y Repetidos

# 3 EDA: Codificación de Variables Categóricas

Debemos de codificar nuestras variables categóricas  
Disponemos de diferentes técnicas de codificación

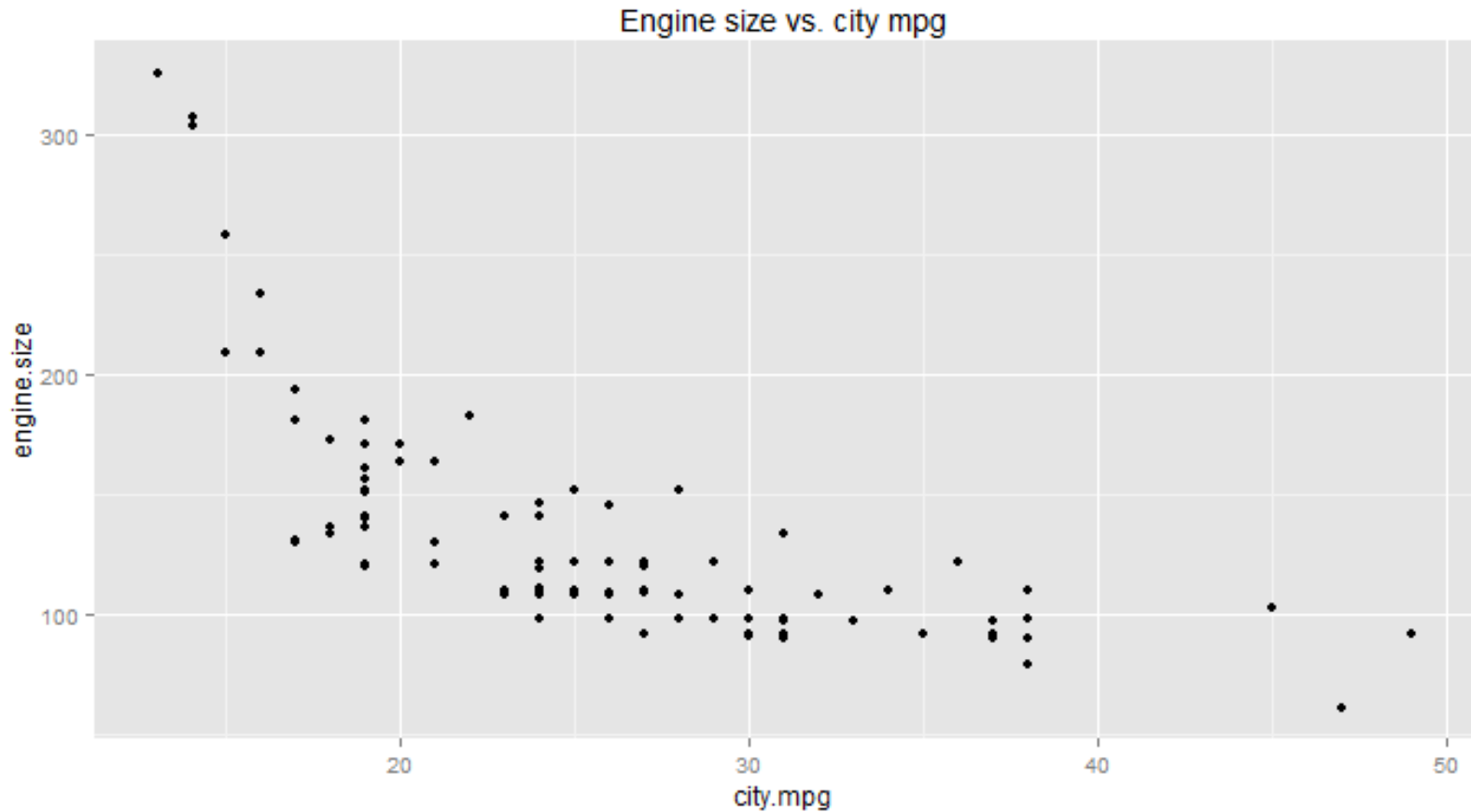
- Asignar valores manualmente
- Label Encoding
  - ✓ Etiquetas
- OrdinalEncoder
  - ✓ Ordinales (Primaria, Secundaria,...)
- OneHot Encoding
  - ✓ No Ordinales (Países, Categorías de Productos,...)
  - ✓ Mas flexible utilizar `pandas.get_dummies()`

Clasificación

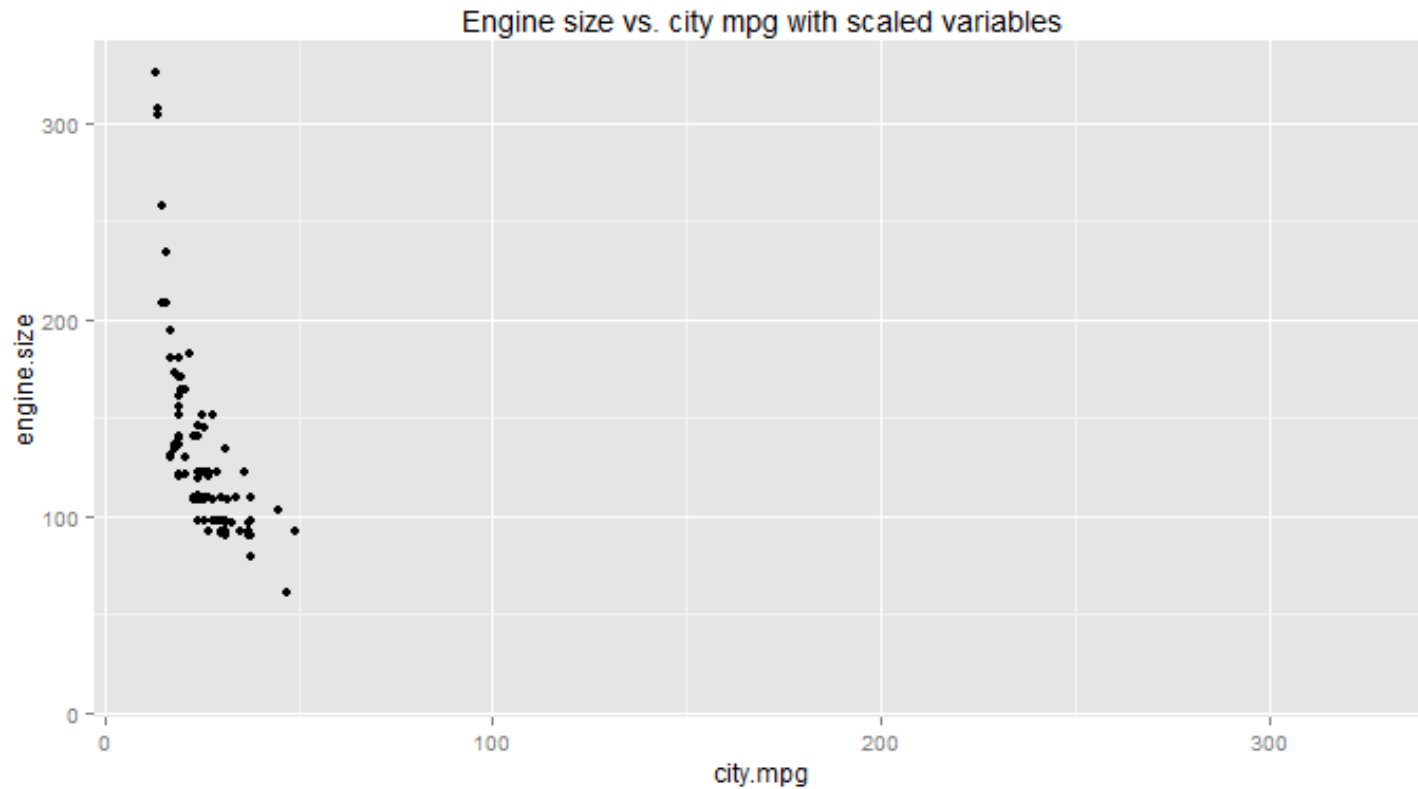
Preprocessing.LabelEncoder

## 03\_03\_Codificación de Variables

- Las variables numéricas necesitan una Escala similar
- A menudo se Escala para tener una media cero para cada columna de forma independiente
- Pueden necesitar quitar la tendencia
- Otras escalas pueden ser min-max
- Escalar después de tratar los outliers

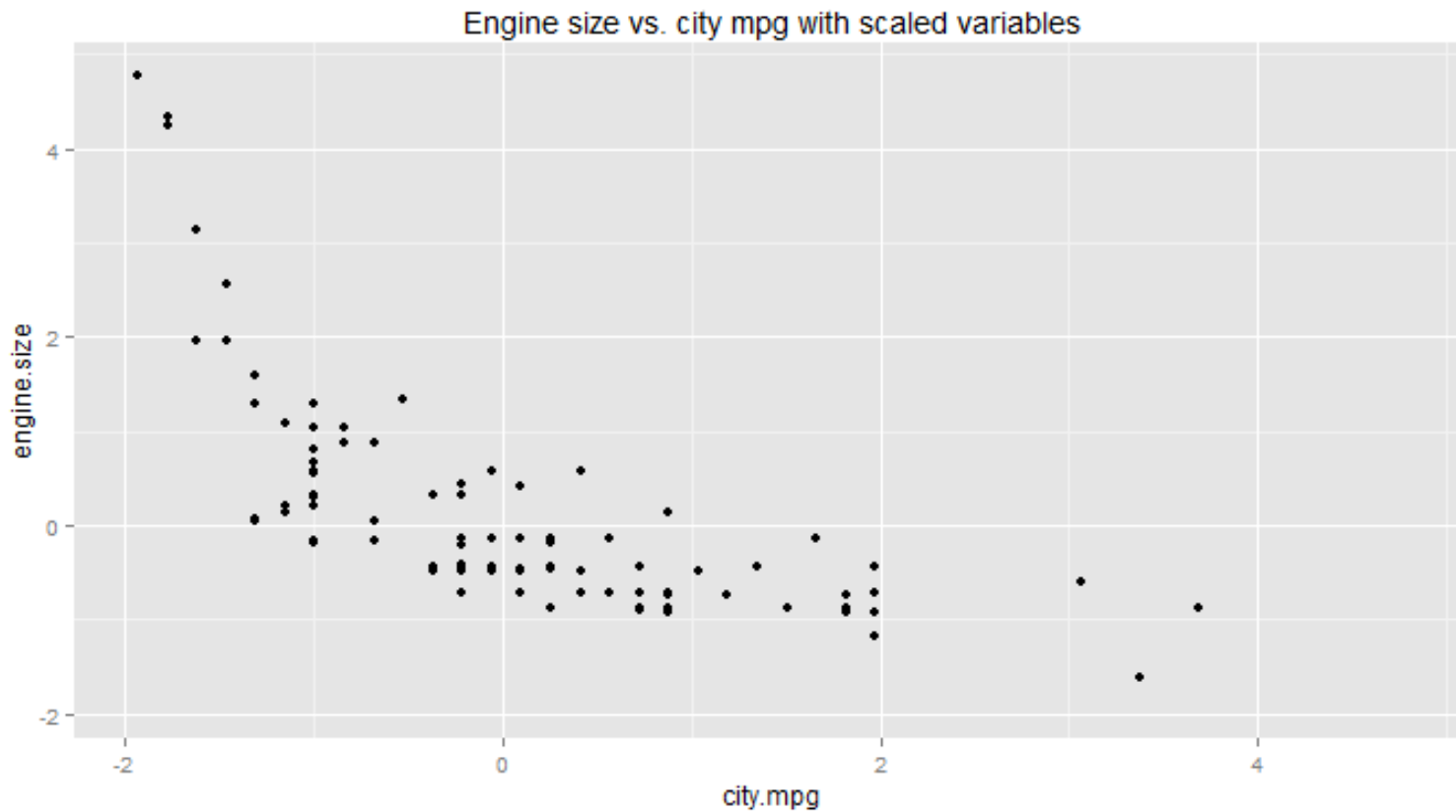


## No escalado





## Escalado



- Modulo de Normalización de Datos
- Python:  
e.g. `scikit-learn.preprocessing.Scale()`
- Opciones:

### StandardScaler

- Asume que los datos siguen una distribución normal dentro de cada característica por lo que centra la distribución en 0 con una desviación estandar de 1

### MinMaxScaler

- Reduce el rango de valores en 0 y 1 (o -1 y 1 si tiene negativos)

### RobustScaler

- Como MinMax pero utilizando un rango de cuartiles

### Normalizer

- Escala cada valor dividiéndolo por su magnitud en un espacio de n dimensiones siendo n el número de características

## 03\_04\_Escalado de Características

# ¡Gracias!



637.505.941



[ajsoto@vernegroup.com](mailto:ajsoto@vernegroup.com)



<https://www.linkedin.com/in/antoniosql>

