SolidQ

Modulo 02

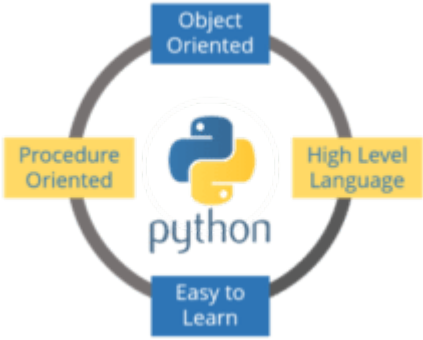
Preparación de datos

Agenda

- Introducción a Python
- Exploración de datos
- Preparación de datos

¿Por qué Python?

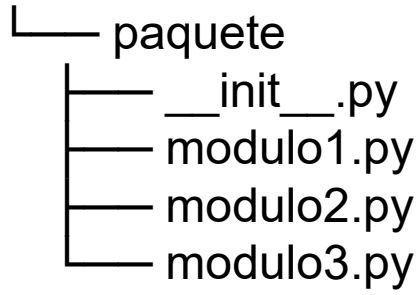
- Fácil de aprender
  - En 8 de cada 10 programas superiores en USA
- Completo
  - No solo estadística
- Librerías Data Science
- Orientado a Objetos
- Libre, gratuito y multiplataforma
- Lenguaje de Alto nivel
  - Fácil de leer por personas



Distribuciones

- Www.Python.org CPython
- Anaconda
- ActivePython
- WinPython

Módulos en Python




```
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
```

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
```

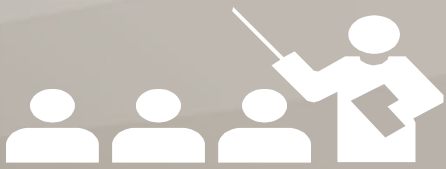
Demo 02 - A

Entendiendo la Librería Estándar de Python

Librerías no estándar más comunes

- **NumPy** : Acrónimo de *Numerical Python*. Su características más potente es que puede trabajar con matrices (array) de n dimensiones. También ofrece funciones básicas de algebra lineal, transformada de Fourier, capacidades avanzadas con números aleatorios, y herramientas de integración con otros lenguajes de bajo nivel como Fortran, C y C++
- **SciPy:** Acrónimo de *Scientific Python*. SciPy está construida sobre la librería NumPy. Es una de las más útiles por la gran variedad que tiene de módulos de alto nivel sobre ciencia e ingeniería, como transformada discreta de Fourier, álgebre lineal, y matrices de optimización. s.
- **Matplotlib:** es una librería de gráficos, desde histogramas, hasta gráficos de líneas o mapas de calor. También se pueden usar comandos de Latex para agregar expresiones matemáticas a tu gráfica.
- **Pandas:** se utiliza para operaciones y manipulaciones de datos estructurados. Es muy habitual usarlo en la fase de depuración y preparación de los datos. Es una librería que se ha añadido recientemente, pero su gran utilidad ha impulsado el uso de Python en la comunidad científica.
- **Scikit Learn** para machine learning: Construida sobre NumPy, SciPy y matplotlib, esta librería contiene un gran número de eficientes herramientas para machine learning y modelado estadístico, como por ejemplo, algoritmos de clasificación, regresión, clustering y reducción de dimensionalidad.

Demo 02-B Librerías No Estándar



Numpy

Pandas

Matplotlib

Agenda

- Introducción a Python
- **Exploración de datos**
- Preparación de datos

Data Frames

- 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

Column 1	Column 2	…	Column *N*
1	ABC	…	12.2
2	XYZ	…	13.1
3	ABC	…	12.8
4	XYZ	…	10.9
5	ABC	…	3.75

Pandas

Leyendo datos

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

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

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

Selección de columnas

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 nº 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
14
13

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

Otros métodos útiles

```
isnull()
groupby(key|expression, axis)
copy()
where(Boolean)
```

GroupBy

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

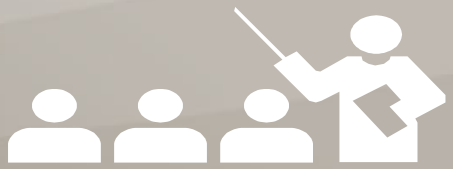
```
frame1= frame1.groupby("Col1").sum()
```

Col2	Col3
14	45
47	141
23	47

Otras operaciones

```
Pandas.DataFrame.apply(function, axis)
Pandas.Series.Map(function, dictionary | series)
Pandas.DataFrame.applymap(function)
```

Demo 02-C Pandas



Selección de columnas

Selección de celdas

Leer datos

Comprobaciones

Laboratorio 02 - B

Repaso básico de Pandas

Visualizar Datos

Análisis de datos exploratorio

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

Vistas de los datos

- 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

Python plotting

- 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

Tipos para pandas.DataFrame.plot()

```
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

Opciones para pandas.DataFrame.plot()

- ax – pyplot axis
- x, y – coordenadas
- color – color de línea o símbolo
- s – tamaño por valor
- Shape –figura
- alpha – transparencia

Pandas Plotting

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()
```

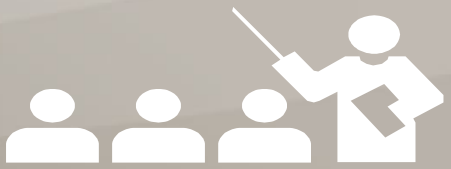
4. Aplicar método plot

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

5. Guardar figura

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

Demo 02-D Visualización de datos



Técnicas de visualización

Agenda

- Introducción a Python
- Exploración de datos
- **Preparación de datos**

Criterios 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

Variables continuas vs. discretas

- 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

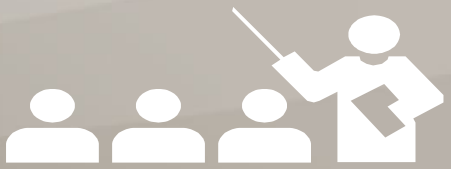
Más detalle en el Módulo 5: Técnicas Avanzadas

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

Demo 02 – E Trabajo con variables



Estadística descriptiva

Mostrar asociaciones gráficamente

Mapeando características ordinales

Codificando etiquetas de clase

Limpieza y transformación de datos

Limpieza y Transformación de datos

- Proceso de preparación de datos
- Valores que faltan y repetidos
- Outliers y errores
- Escalado

Limpieza y transformación (manipulación de datos)

- Los datos no llegan, habitualmente, en la forma en la que los necesitamos para el análisis
- La manipulación de datos es la parte que más tiempo consume
- Es un proceso iterativo
  - A menudo se descubren con la visualización
  - Ayuda a resolver problemas de modelado

Valores nulos y repetidos

Valores nulos y repetidos

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

Valores que faltan

Col1	Col2	Col3	Col4	Col5
12456	0.99	Male	43	Small
98567	1.23		55	Medium
34567	9999	Female	NA	Large
67231	0.72	Male	35	?

Cómo tratar los valores que faltan

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

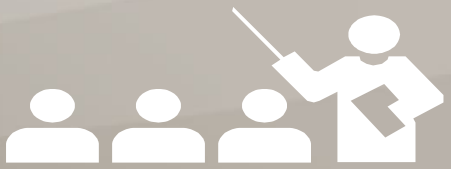
Python – pandas.DataFrame.isna()

Valores repetidos

Key Col	Col2	Col3	Col4	Col5
12456	0.99	Male	43	Small
98567	1.23	Male	55	Medium
34567	1.55	Female	43	Large
34567	1.55	Female	43	Large
34567	1.55	Female	43	Large
34567	.78	Male	43	Large
67231	0.72	Male	35	Small

Python – DataFrame.drop_duplicates()

Demo 02 – F Nulos y repetidos



Gestionar nulos

Limpiar outliers y errores

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

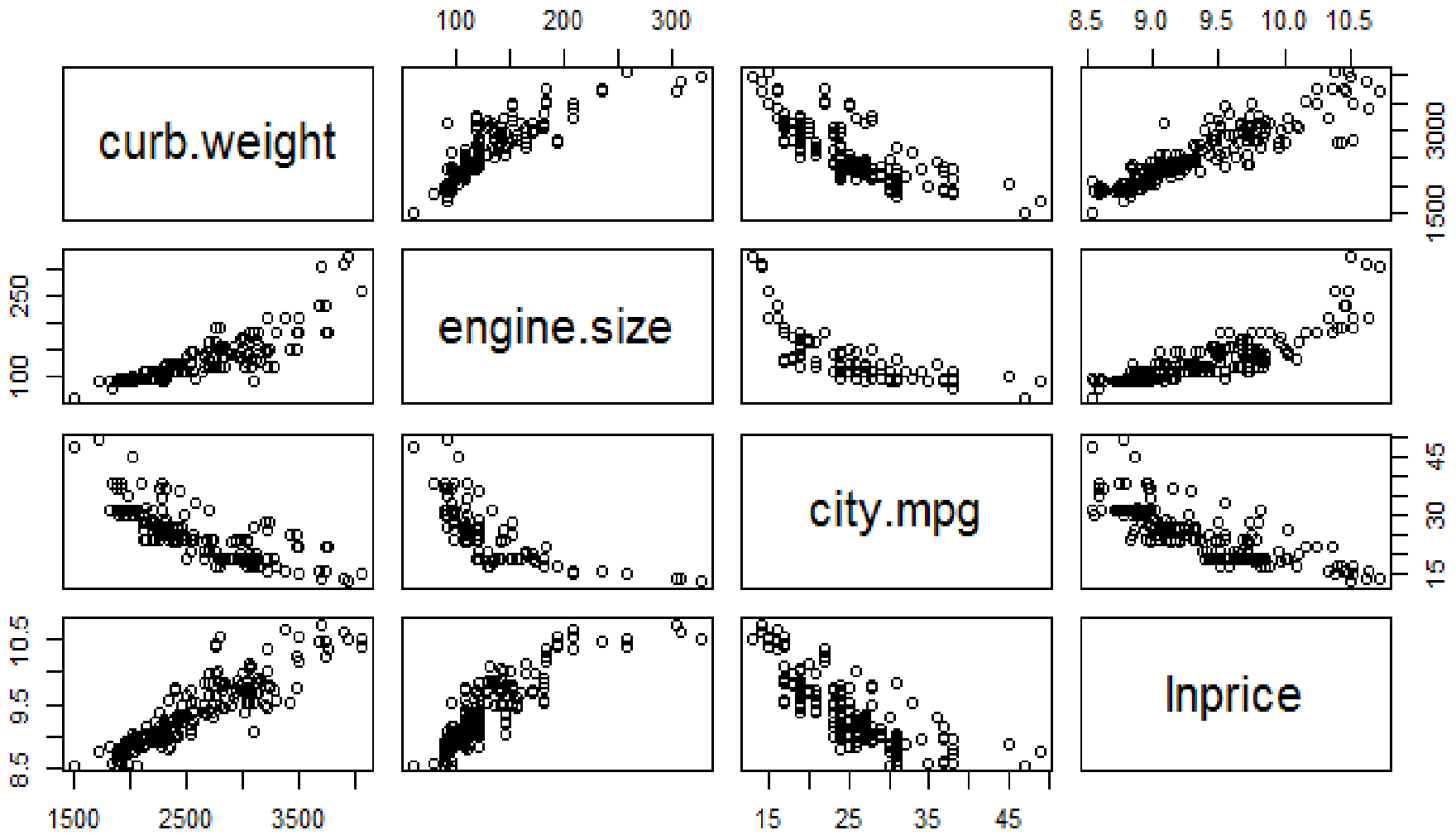
Outliers

- Outilers 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

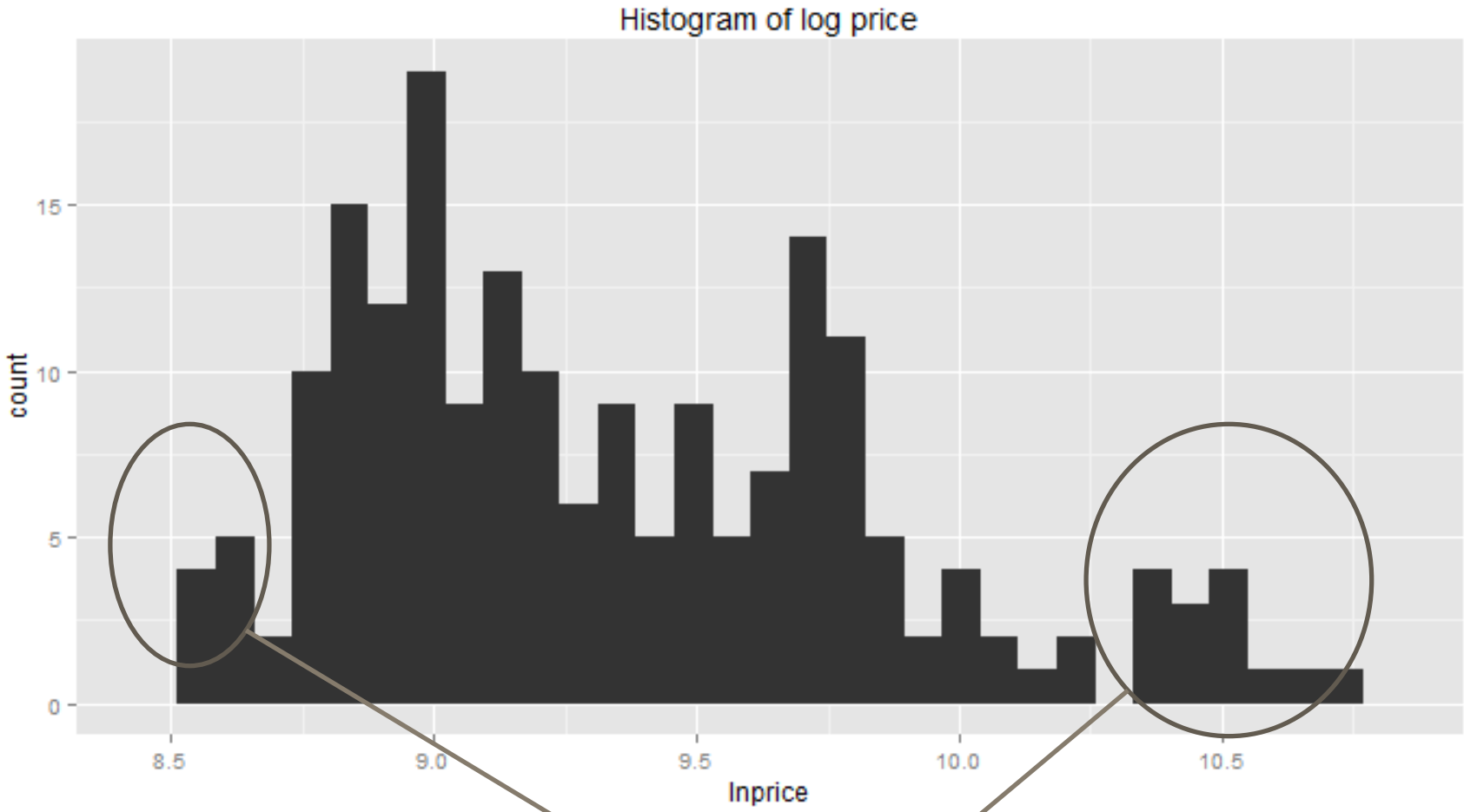
Visualizar Outliers

- Un matriz de plot de puntos ayuda a visualizar los outliers
- Python – pandas.tools.plotting.scatter_matrix

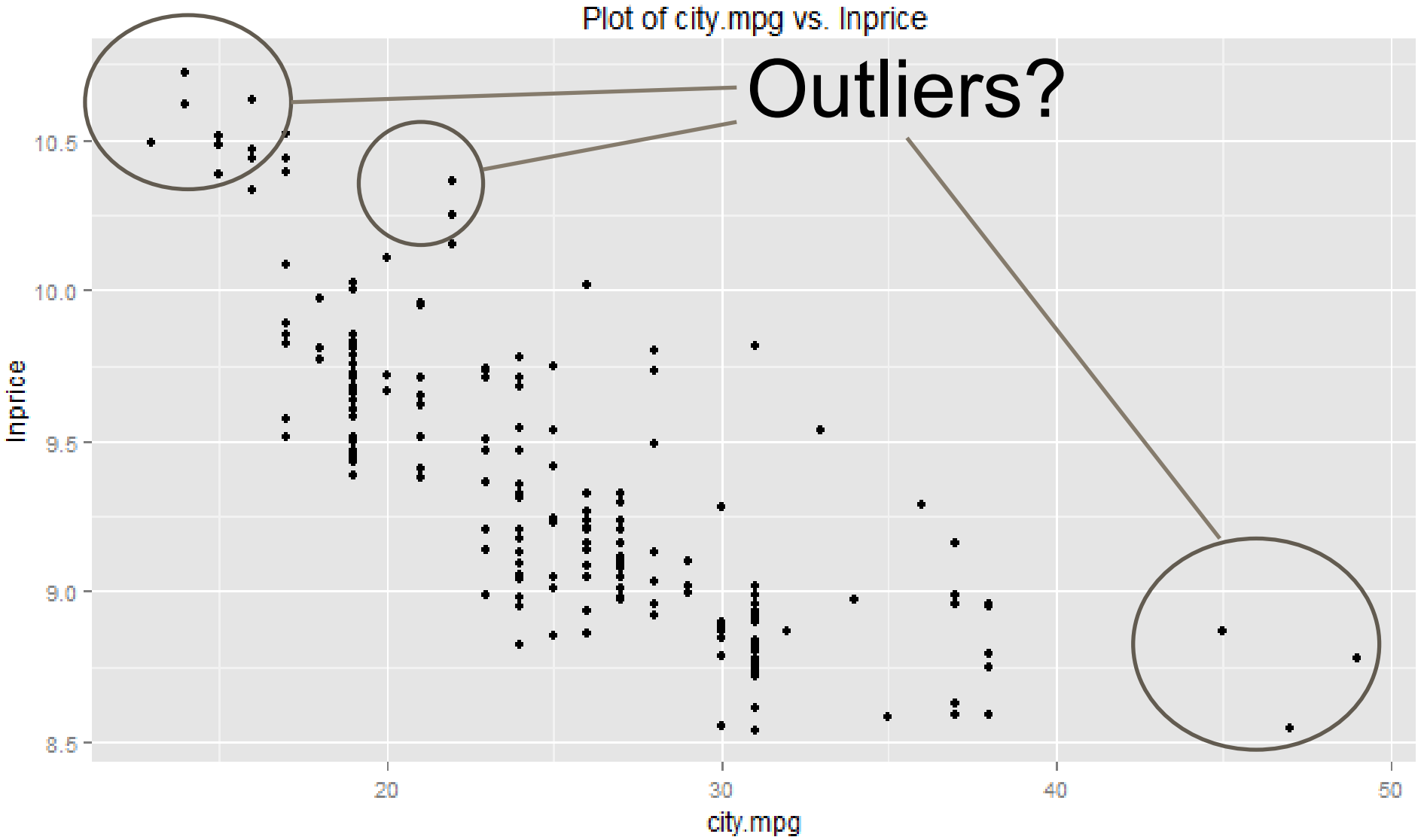
Visualizar Outliers



Identificar Outliers y Errores



Identificar Outliers y Errores



Limpiar Outliers y Errors

- Tratamientos de error
  - Censurar
  - Recortar
  - Interpolar
  - Sustituir

Eliminar Outliers

Python: DataFrame = DataFrame[expression_filtro]

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

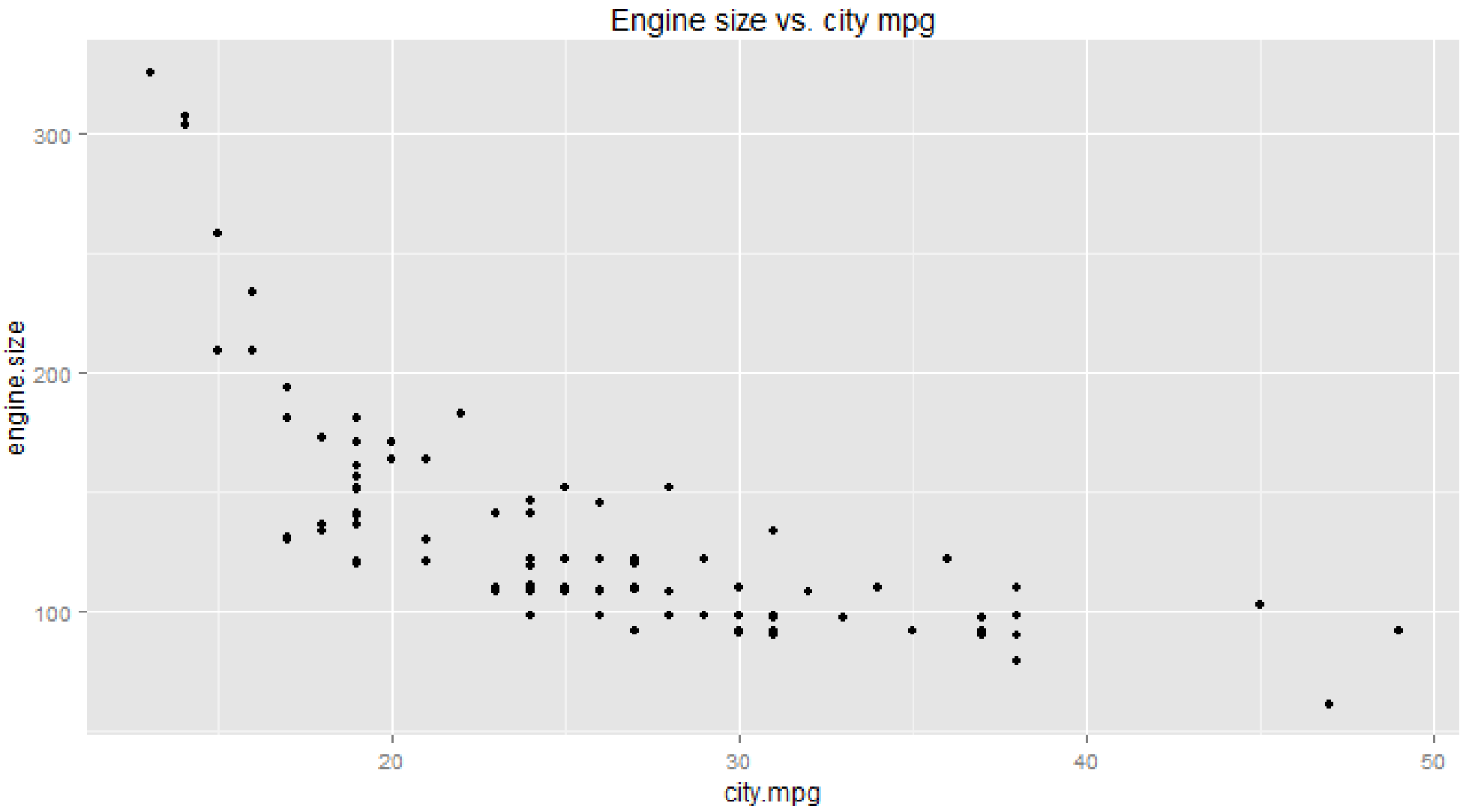
Para grandes fuentes de datos podemos utilizar PyOD que es una librería especializada con más de 20 algoritmos para la detección de outliers

Escalado de datos

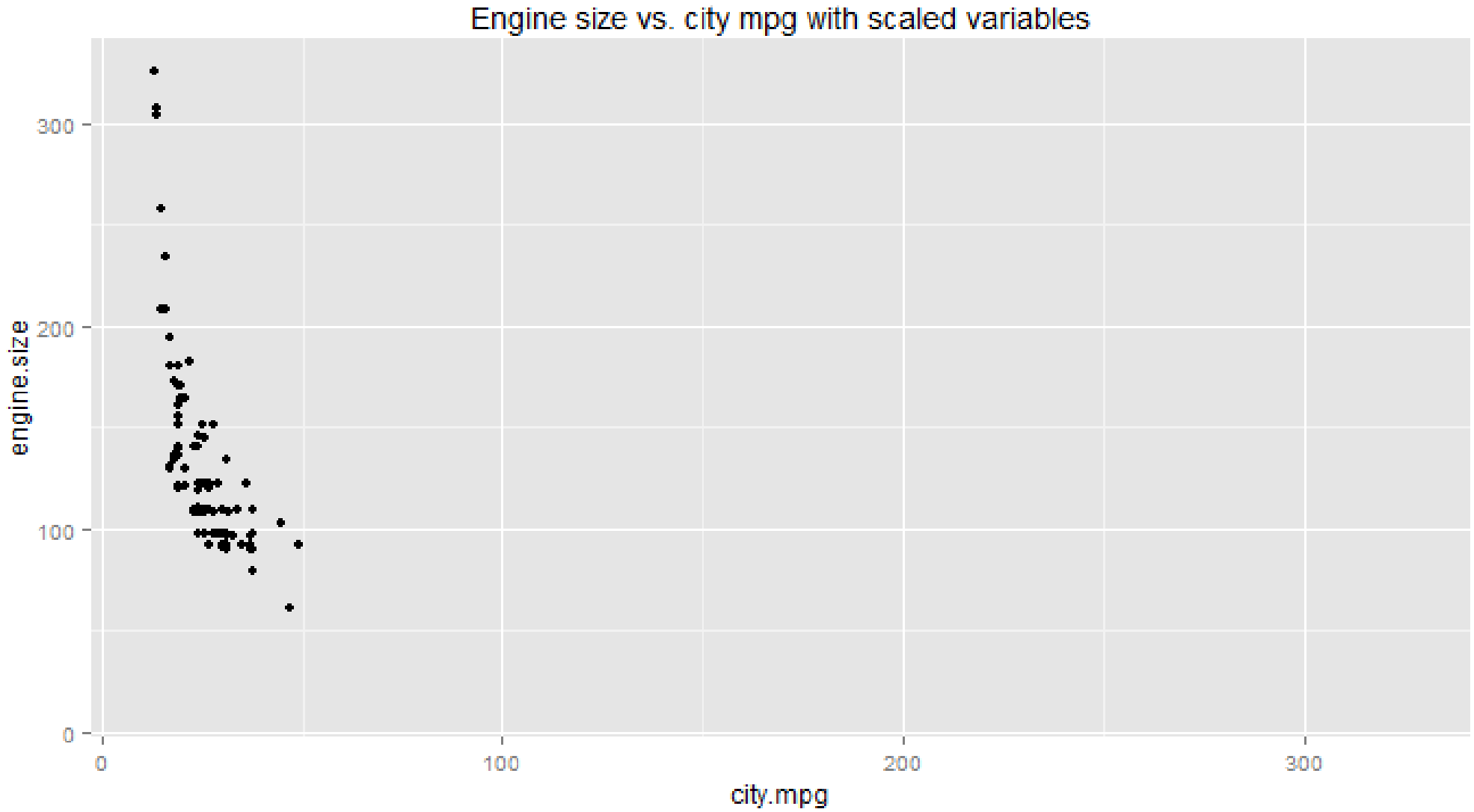
Escalado→ Normalización

- 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

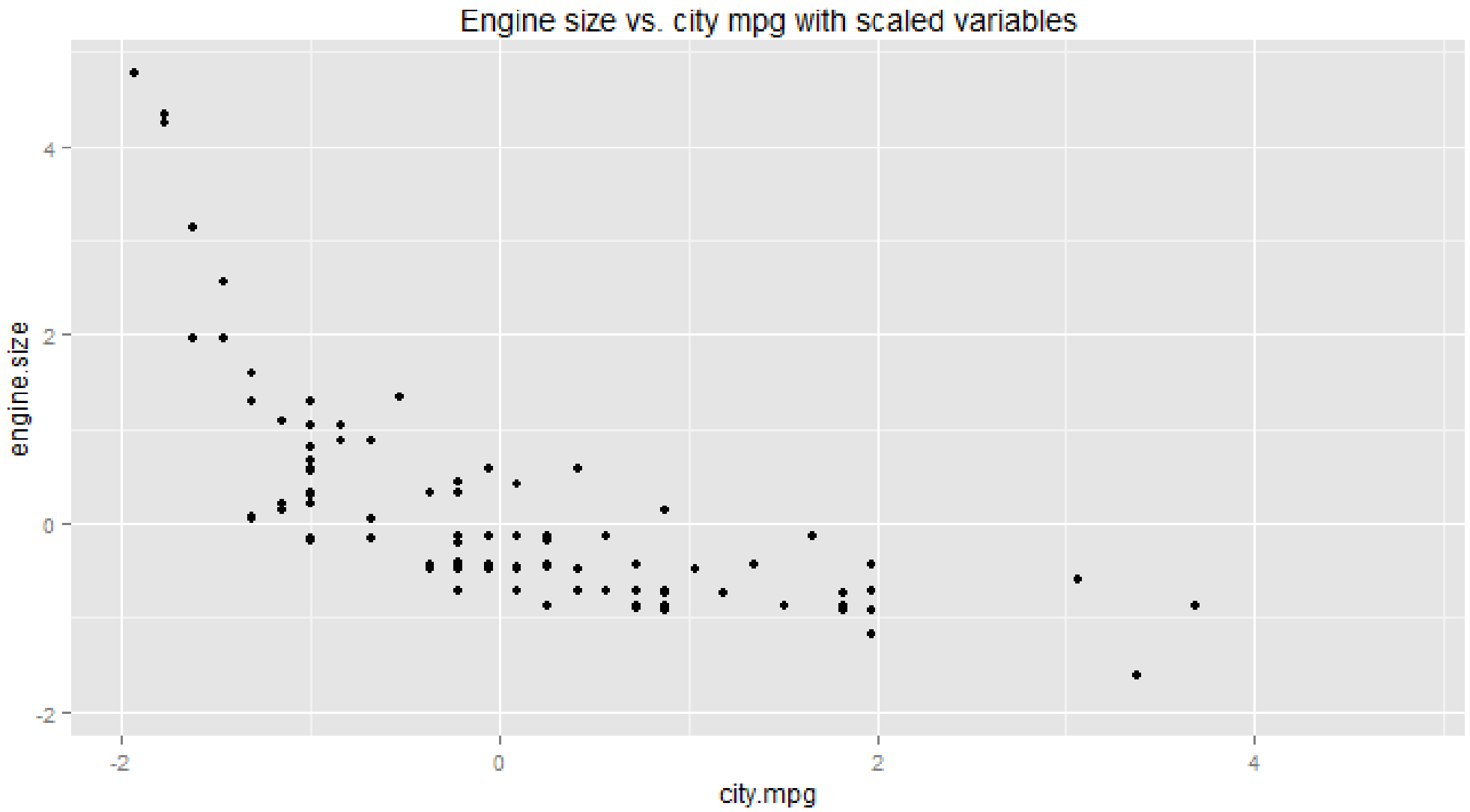
Scatter plot de dos características numéricas



No escalado


Engine size vs. city mpg with scaled variables
engine.size
300
200
100
0
0
100
200
300
city.mpg

Escalado


Engine size vs. city mpg with scaled variables
engine.size
city.mpg
4
2
0
-2
-2
0
2
4

Escalado

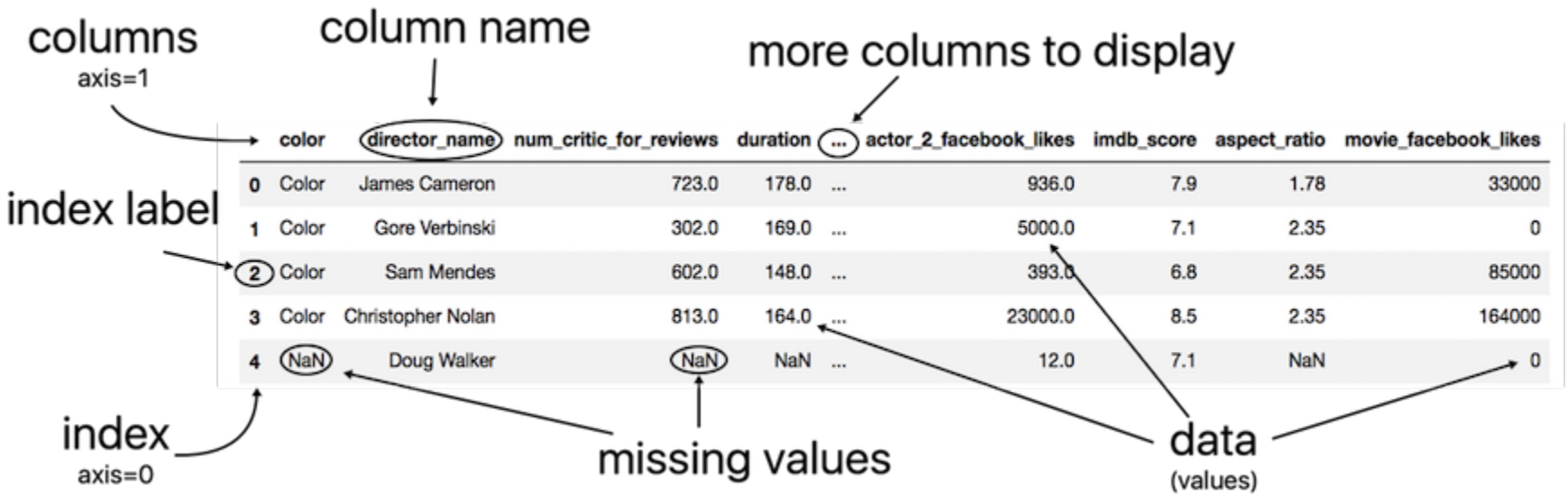
- Modulo de Normalización de Datos
- Python:

  e.g. `scikit-learn.preprocessing.Scale()`

Laboratorio 02 Final

Analizando los datos del Titanic

Resumen



	color	director_name	num_critic_for_reviews	duration	...	actor_2_facebook_likes	imdb_score	aspect_ratio	movie_facebook_likes
0	Color	James Cameron	723.0	178.0	...	936.0	7.9	1.78	33000
1	Color	Gore Verbinski	302.0	169.0	...	5000.0	7.1	2.35	0
2	Color	Sam Mendes	602.0	148.0	...	393.0	6.8	2.35	85000
3	Color	Christopher Nolan	813.0	164.0	...	23000.0	8.5	2.35	164000
4	NaN	Doug Walker	NaN	NaN	...	12.0	7.1	NaN	0

www.solidq.com

info@solidq.com