DATA SCIENCE

Pandas

DS-BUE 7/ Clase 2

Hoja de ruta:

19:10	Recapitulación clase anterior
19:20	Exposicion dialogada Pandas
19:50	Ejercicio grupal
20:30	jBreak
20:45	Hands on: Tutoriales de Pandas
21:45	Presentación DESAFÍO PANDAS y cierre

Resumen: Actividad grupal

Resumen: Actividad grupal

?

0	0	0	0	0	0	0	0	0

?

1	1	1	1	1	1	1	1	1

?

0	0	0
0	0	0
0	0	0

?

1	1	1
1	1	1
1	1	1

Resumen: Actividad grupal

0	1		4	5		-4	-4
2	4	-			=	-2	-1
10	10					6	5

Qué estamos haciendo?
Cuales son sus dimensiones?

Elementos en python

Listas y Tuplas
Diccionarios
Vectores y Matrices
Dataframes -Hoy

Pandas

Pandas es una librería de python destinada al análisis de datos, que proporciona unas estructuras de datos flexibles y que permiten trabajar con ellos de forma muy eficiente. Pandas ofrece las siguientes estructuras de datos:

- Series: Son arrays unidimensionales con indexación (arrays con índice o etiquetados), similar a los diccionarios. Pueden generarse a partir de diccionarios o de listas.
- DataFrame: Son estructuras de datos similares a las tablas de bases de datos relacionales como SQL.
- Panel, Panel4D y PanelND: Estas estructuras de datos permiten trabajar con más de dos dimensiones. Dado que es algo complejo y poco utilizado trabajar con arrays de más de dos dimensiones no trataremos los paneles en estos tutoriales de introducción a Pandas.

Datasets para Data Science

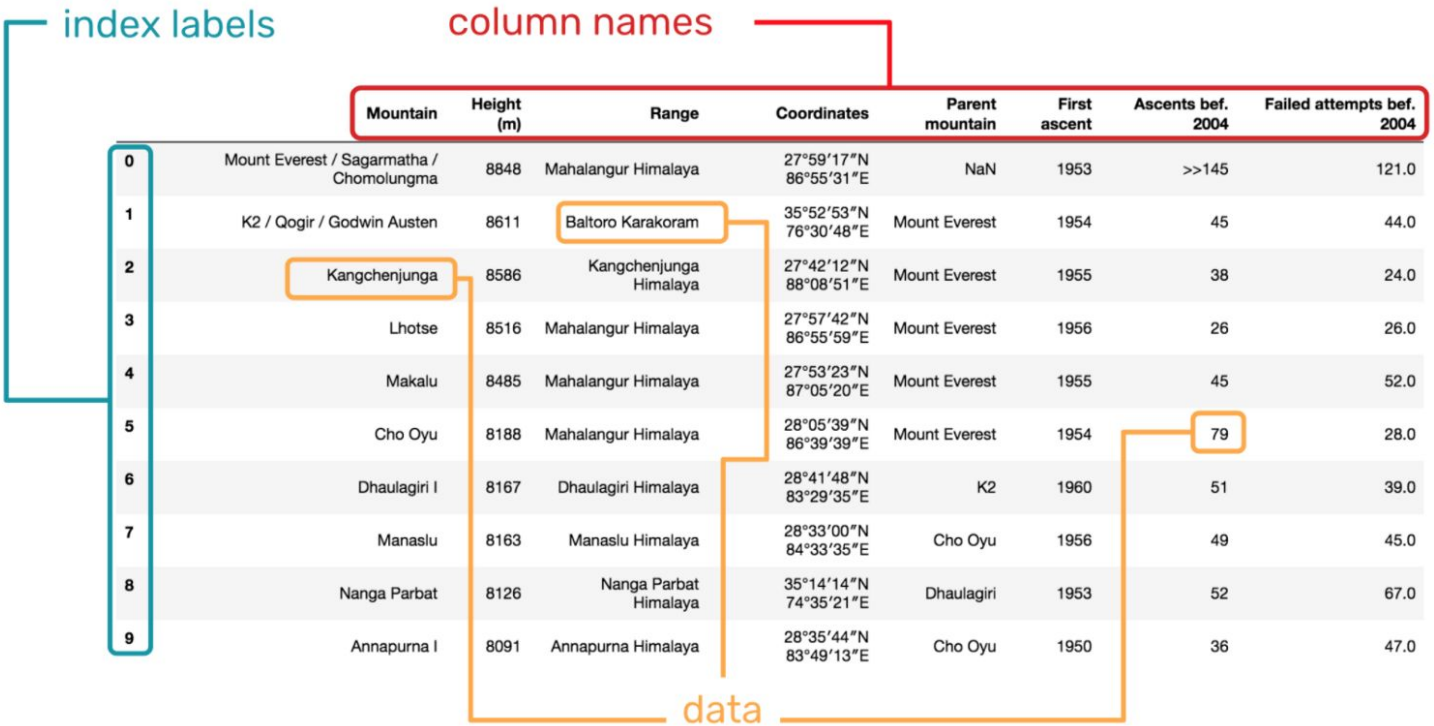
Dataset es el conjunto de datos que utilizaremos en el workflow de data science. Los podemos generar, obtener de una base de datos o simular. Suelen venir en .txt, .csv, .xlsx, .json, etc.

DATOS POBLACIONALES:
http://www.ign.gob.ar/nuestrasactividades/geografia/datosargentina/divisionpolitica

Los **arrays** funcionan muy bien cuando tenemos datos numréicos y limpios.

Son poco útiles cuando tenemos *datos faltantes o poco estructurados* (clases, continuos, discretos, etc).

Estructura del Dataframe



	Mountain	Height (m)	Range	Coordinates	Parent mountain	First ascent	Ascents bef. 2004	Failed attempts bef. 2004
0	Mount Everest / Sagarmatha / Chomolungma	8848	Mahalangur Himalaya	27°59′17″N 86°55′31″E	NaN	1953	>>145	121.0
1	K2 / Qogir / Godwin Austen	8611	Baltoro Karakoram	35°52′53″N 76°30′48″E	Mount Everest	1954	45	44.0
2	Kangchenjunga	8586	Kangchenjunga Himalaya	27°42′12″N 88°08′51″E	Mount Everest	1955	38	24.0
3	Lhotse	8516	Mahalangur Himalaya	27°57′42″N 86°55′59″E	Mount Everest	1956	26	26.0
4	Makalu	8485	Mahalangur Himalaya	27°53′23″N 87°05′20″E	Mount Everest	1955	45	52.0
5	Cho Oyu	8188	Mahalangur Himalaya	28°05′39″N 86°39′39″E	Mount Everest	1954	79	28.0
6	Dhaulagiri I	8167	Dhaulagiri Himalaya	28°41′48″N 83°29′35″E	K2	1960	51	39.0
7	Manaslu	8163	Manaslu Himalaya	28°33′00″N 84°33′35″E	Cho Oyu	1956	49	45.0
8	Nanga Parbat	8126	Nanga Parbat Himalaya	35°14′14″N 74°35′21″E	Dhaulagiri	1953	52	67.0
9	Annapurna I	8091	Annapurna Himalaya	28°35′44″N 83°49′13″E	Cho Oyu	1950	36	47.0

Dataframe

In [3]:
```
import pandas as pd
# create a simple dataset of people
data = {'Name': ["John", "Anna", "Peter", "Linda"],
'Location' : ["New York", "Paris", "Berlin", "London"],
'Age' : [24, 13, 53, 33]
}
data_pandas = pd.DataFrame(data)
# IPython.display allows "pretty printing" of dataframes
# in the Jupyter notebook
```

In [4]:
```
data_pandas
```

Out[4]:

	Age	Location	Name
0	24	New York	John
1	13	Paris	Anna
2	53	Berlin	Peter
3	33	London	Linda

Dataframe

Importador de Datos

pd.read_csv(filename) - De un archivo CSV
pd.read_table(filename) - Desde un archivo de texto delimitado (como TSV)
pd.read_excel(filename) - De un archivo Excel
pd.read_sql(query, connection_object) - Lee desde una BaseDeDatos/Tabla SQL
pd.read_json(json_string) - Lee desde una cadena, URL o archivo con formato JSON
pd.read_html(url) - Analiza una URL html, una cadena o un archivo y extrae tablas a una lista
pd.read_clipboard() - Toma el contenido del porta papeles
pd.DataFrame(dict) - Desde un diccionario

Dataframe

Exportador de Datos

df.to_csv(filename) - Escribir en un archivo CSV
df.to_excel(filename) - Escribir en un archivo Excel
df.to_sql(table_name, connection_object) - Escribir en una tabla SQL
df.to_json(filename) - Escribir en un archivo con formato JSON

Dataframe

Visualizar / Inspeccionar Datos

df.head(n) - Primeras n filas del DataFrame
df.tail(n) - Las últimas n filas del DataFrame
df.shape() - Número de filas y columnas
df.info() - Índice, tipo de datos y memoria
df.describe() - Estadísticas resumidas de columnas numéricas
df.value_counts(dropna=False) - Ver valores y recuentos únicos

Dataframe

Selección

df[col] - Devuelve la columna con la etiqueta col como Serie
df[[col1, col2]] - Devuelve columnas como un nuevo DataFrame
df.iloc[0] - Selección por posición
df.iloc[0,:] - Primera fila
df.iloc[0,0] - Primer elemento de la primera columna
df.loc['index_one'] - Selección por índice

Dataframe

Selección

```
[11]: data = pd.Series(['a', 'b', 'c'], index=[1, 3, 5])
      data
```

```
[11]: 1    a
      3    b
      5    c
      dtype: object
```

```
[14]: data.loc[1]
```

```
[14]: 'a'
```

```
[15]: data.loc[1:3]
```

```
[15]: 1    a
      3    b
      dtype: object
```

```
[16]: data.iloc[1]
```

```
[16]: 'b'
```

```
[17]: data.iloc[1:3]
```

```
[17]: 3    b
      5    c
      dtype: object
```

Dataframe

Limpieza de datos

df.columns = ['a', 'b', 'c'] - Renombrar columnas
pd.isnull() - Comprueba valores nulos, devuelve Boolean Arrays
pd.notnull() - El opuesto a pd.isnull()
df.dropna() - Elimina todas las filas que contienen valores nulos
df.dropna(axis=1) - Elimina todas las columnas que contienen valores nulos
df.dropna(axis=1, thresh=n) - Elimina todas las filas que tienen menos de n valores no nulos

Dataframe

```
In [18]: df.drop(2)
```

Out[18]:

	A	B	C	D
0	1	2	3	4
1	2	3	4	5
3	4	5	6	7
4	5	6	7	8

```
In [19]: df.drop([0,3])
```

Out[19]:

	A	B	C	D
1	2	3	4	5
2	3	4	5	6
4	5	6	7	8

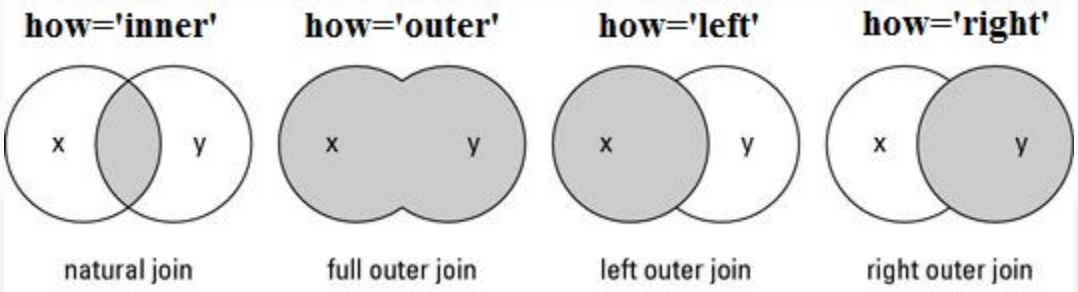
```
df
```

	A	B	C	D
0	1.0	2.0	3.0	4.0
1	5.0	6.0	NaN	8.0
2	0.0	11.0	12.0	NaN

```
df.dropna()
```

	A	B	C	D
0	1.0	2.0	3.0	4.0

Dataframe



#inner join in python pandas

pd.merge(df1, df2, on='Customer_id', how='inner')

 # outer join in python pandas

pd.merge(df1, df2, on='Customer_id', how='outer')

 # left join in python

pd.merge(df1, df2, on='Customer_id', how='left')

 # right join in python pandas

pd.merge(df1, df2, on='Customer_id', how='right')

Dataframe

Filtro, orden y agrupamiento

df[df[col]> 0.5] - Filas donde la columna col es mayor que 0,5
df[(df[col] > 0.5]) & (df[col] < 0.7)] - Filas donde 0.7 > col > 0.5
df.sort_values(col1) - Ordenar los valores por la col1 en orden ascendente
df.sort_values(col2, ascending=False) - Ordena los valores por col2 en orden descendente
df.sort_values([col1, col2], ascending=[True, False]) - Ordena los valores por la col1 de forma ascendente y luego por la col2 en orden descendente
df.groupby(col) - Devuelve un objeto groupby para los valores de una columna
df.groupby([col1, col2]) - Devuelve un objeto groupby para valores de múltiples columnas
df.groupby(col1)[col2].mean() - Devuelve la media de los valores en col2, agrupados por los valores en col1 (la media se puede remplazar con casi cualquier función de la sección Estadísticas)
df.apply(np.mean) - Aplica la función np.mean() en cada columna

Dataframe

Estadísticas

Todas estas funciones también se pueden aplicar a una serie

df.describe() - Resumen de estadísticas para columnas numéricas
df.mean() - Devuelve la media de todas las columnas
df.corr() - Devuelve la correlación entre columnas en un DataFrame
df.count() - Devuelve el número de valores no nulos en cada columna DataFrame
df.max() - Devuelve el valor más alto en cada columna
df.min() - Devuelve el valor más bajo en cada columna
df.median() - Devuelve la media de cada columna
df.std() - Devuelve la desviación estándar de cada columna

HANDS-ON TRAINING

Ahora trabajemos un ejercicio juntos:

Clase04.ipynb

HANDS-ON
TRAINING

Clase_04_01_Pandas.ipynb
Clase_04_02_Pandas.ipynb

Desafío PANDAS

Iris Versicolor

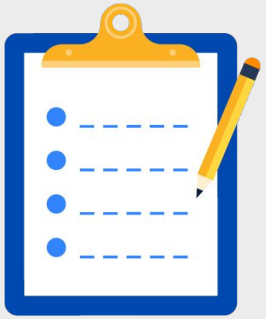
Iris Setosa

Iris Virginica

Tenemos el dataset de iris y queremos identificar y eliminar las instancias que no tengan **ningún valor en algunos** de los campos: largo y ancho del pétalo y sépalo. **Que funciones usaría? Me van a quedar valores nulos en el dataset?**

	sepal length	sepal width	petal length	petal width	target
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

En resúmen...



Pandas.

- Dataframe
- Operaciones
- Filtros booleanos
- Drop
- Mucho más en el notebook

Desafío PANDAS segunda parte: 5 mínutos

Iris Versicolor

Iris Setosa

Iris Virginica

Formar cuatro grupos:

Tenemos el dataset de iris y queremos identificar y eliminar las instancias que no tengan **ningún** valor en los campos: largo y ancho del pétalo y sépalo. Que funciones usaría?

Diseñen y exponga una estrategia para reemplazar los valores faltantes en el atributo largo de pétalo.

.

Desafío PANDAS: creando nuestro dataset

Ahora entren a Trello y busquen el formulario mi dataset:

- **Vamos a generar nuestro propio dataset de la clase. En caso de no saber una respuesta poner NaN.**
- **Qué preguntas podrían responder con este dataset?**
- **En la próxima clase vamos a trabajar con este dataset y sacar conclusiones interesantes.**

Para la próxima...

- Ver los videos de **Matplotlib** (nivel II)
- Repasar herramientas de python, numpy y pandas
- Comenzar a ver la primera entrega