CC - AI

Programación para la Computación Científica - IA

Anatomia de un DataFrame

Pandas

Universidad Sergio Arboleda
Prof. John Corredor

Agenda

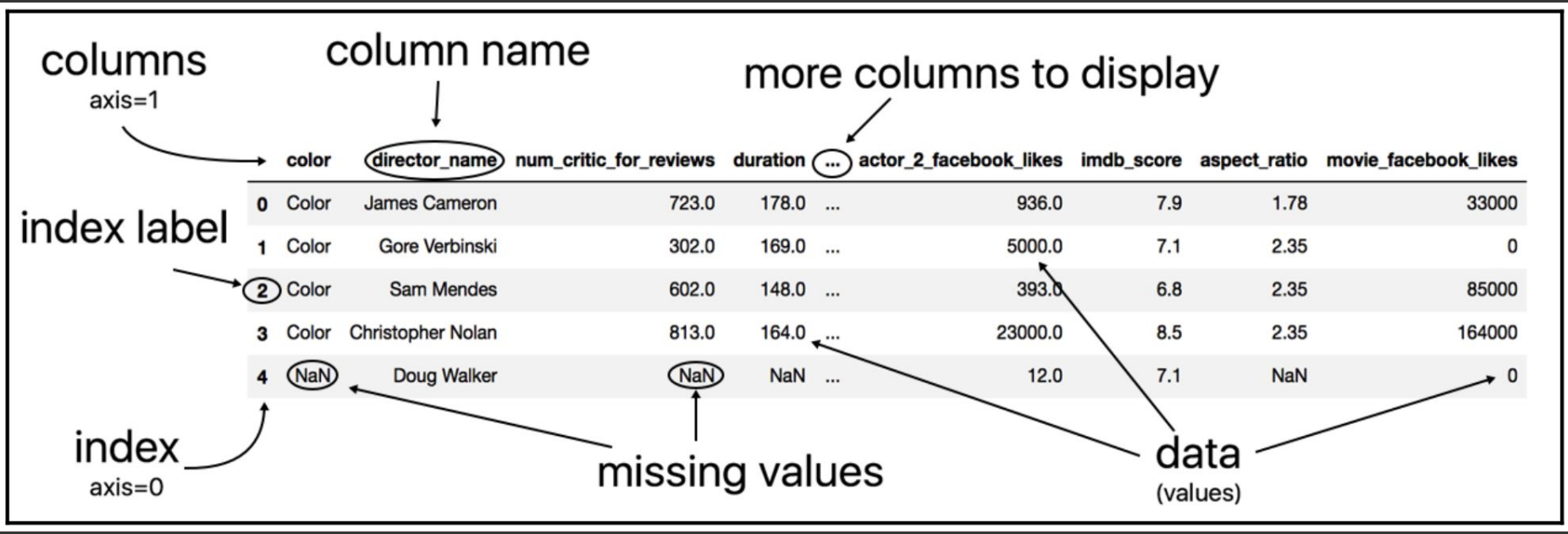
- Anatomía de un DataFrame
- Acceder a los principales componentes del DataFrame
- Comprender los tipos de datos
- Seleccionando una sola columna de datos como una serie
- Llamando a los métodos de la serie
- Trabajar con los operadores en una serie
- Encadenar los métodos de la serie juntos
- Hacer que el índice sea significativo
- Renombrar los nombres de las filas y las columnas
- Crear y eliminar columnas

—

```
import pandas as pd
import numpy as np

# pd.set_option('max_columns', n, 'max_rows', m)

movies = pd.read_csv('movies.csv')
movies.head()
```



	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

Accediendo a los componentes principales del DataFrame (Atributos)

```
columns = movie.columns
index = movies.index
data = movies.values

columns
index
data
```

los objetos del índice se refieren a todos los posibles objetos que pueden ser utilizados para el índice o las columnas. Todos ellos son subclases del pd.Index. Aquí está la lista completa de los objetos del Índice: CategoricalIndex, MultiIndex, IntervalIndex, Int64Index, UInt64Index, Float64Index, RangeIndex, TimedeltaIndex, DatetimeIndex, PeriodIndex.

Accediendo a los componentes principales del DataFrame (Atributos)

Tipos de Datos

```
type(index)
type(columns)
type(data)
```

Valores

```
index.values
columns.values
```

Los valores del atributo DataFrame retorna una matriz n-dimensional NumPy, o ndarray.

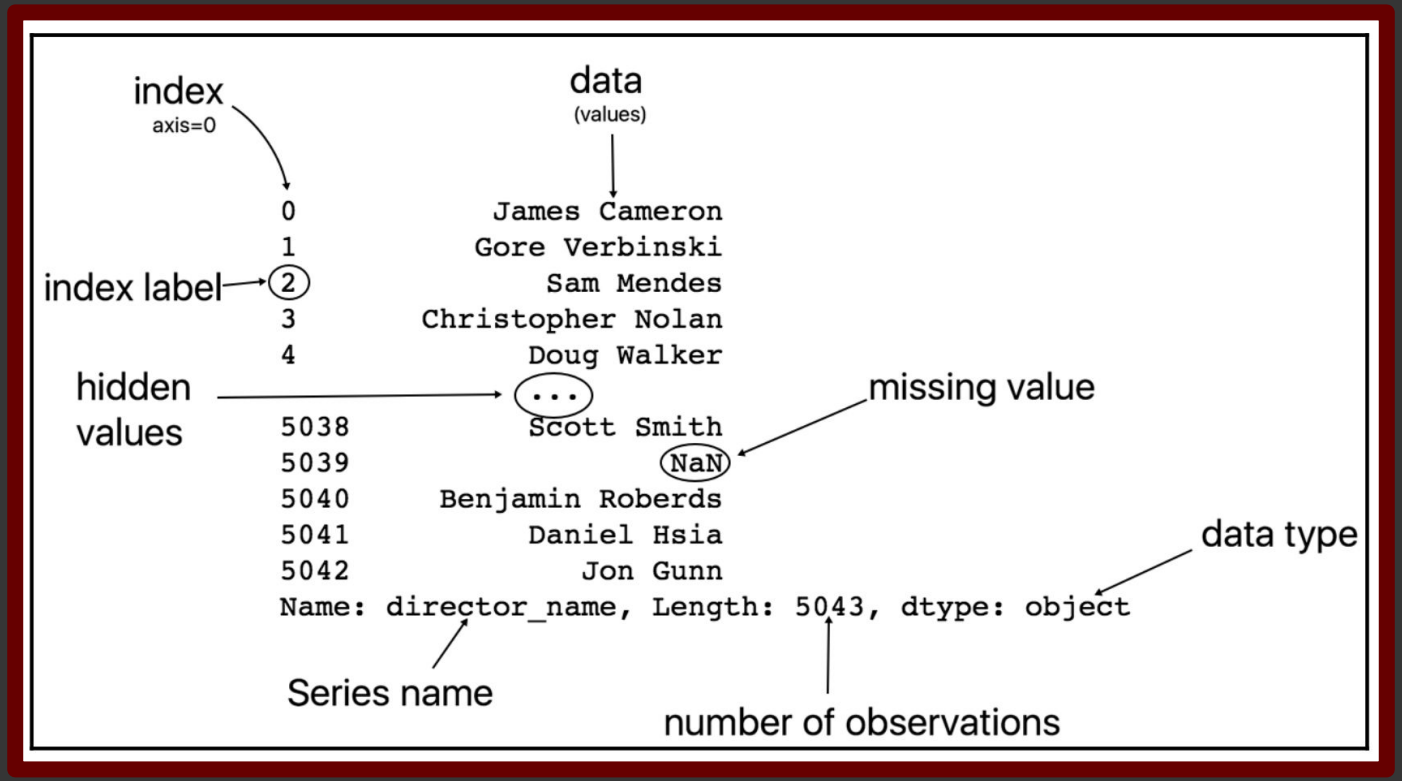
Tipos de Datos

```
movies = pd.read_csv('movie.csv')
movies.dtypes
movie.get_dtype_counts()
```

Columna de datos como una serie

```
movies['director_name']
```

```
movies.director_name
```




```
type(movie['director_name'])
```

Series como variables

```
director = movie['director_name'] # save Series to variable
director.name
```

```
director.to_frame().head()
```

Series y Metodos: llamadas

```
s_attr_methods = set(dir(pd.Series))
len(s_attr_methods)
```

```
df_attr_methods = set(dir(pd.DataFrame))
len(df_attr_methods)
```

```
len(s_attr_methods & df_attr_methods)
```

Llamar a los métodos de la Serie es la forma principal de usar las habilidades que la Serie ofrece.

Tanto los Series como los DataFrames tienen un enorme poder. La función dir se usa para descubrir todos los atributos y métodos de una Serie.

Series y Metodos: llamadas

```
movie = pd.read_csv('movie.csv')
director = movie['director_name']                    #Contiene Strings
actor_1_fb_likes = movie['actor_1_facebook_likes']   #Contiene Numeros
```

```
director.head()
actor_1_fb_likes.head()
```

```
pd.set_option('max_rows', 8)
director.value_counts()

actor_1_fb_likes.value_counts()
```

- Series y Metodos: llamadas

director.value_counts()

actor_1_fb_likes.value_counts()

Uno de los métodos más útiles para el tipo de datos de objetos Serie es value_counts, que cuenta todas las ocurrencias de cada valor único.

El método value_counts suele ser más útil para las series con tipos de datos de objetos, pero en ocasiones también puede proporcionar una visión de las series numéricas. Usado con actor_1_fb_likes, parece que los números más altos han sido redondeados al millar más cercano ya que es poco probable que tantas películas hayan recibido exactamente 1.000 likes:

- director.size # len

  director.shape # len

  len(director) # len

  director.count() # retorna numero de valores “non-missing”

  actor_1_fb_likes.count()

  actor_1_fb_likes.quantile()

```
actor_1_fb_likes.min(), actor_1_fb_likes.max(), \
actor_1_fb_likes.mean(), actor_1_fb_likes.median(), \
actor_1_fb_likes.std(), actor_1_fb_likes.sum()
```

```
actor_1_fb_likes.describe()   # resumen estadistico
director.describe()
actor_1_fb_likes.quantile(.2)
actor_1_fb_likes.quantile([.1, .2, .3, .4, .5, .6, .7, .8, .9])
director.isnull() # Serie de Booleans (tamaño de la serie)
fillna metodo remplaza los valores perdidos
actor_1_fb_likes_filled = actor_1_fb_likes.fillna(0)
actor_1_fb_likes_filled.count()
dropna metodo: elimina los elementos con valores perdidos
actor_1_fb_likes_dropped = actor_1_fb_likes.dropna()
actor_1_fb_likes_dropped.size
```

- director.value_counts()
  director.value_counts(normalize=True)

director.notnull()

director.hasnans

El método value_counts es uno de los métodos de Series más informativos y muy utilizado durante el análisis exploratorio, especialmente con columnas categóricas.
Por defecto devuelve los recuentos, pero al establecer el parámetro normalizar en True, se devuelven las frecuencias relativas en su lugar, lo que proporciona otra vista de la distribución

Operadores sobre series

```
pd.options.display.max_rows = 6
5 + 9    # plus operator example. Adds 5 and 9
5 <= 9   # less than or equal to operator
'abcde' + 'fg'    # plus operator for strings. C
not (5 <= 9)      # not is an operator
7 in [1, 2, 6]    # in operator checks for membership of a list
set([1,2,3]) & set([2,3,4])
[1, 2, 3] - 3
a = set([1,2,3])
a[0]
```

Operadores sobre series

```
movie = pd.read_csv('movie.csv')
imdb_score = movie['imdb_score']
imdb_score
```

```
imdb_score + 1
imdb_score * 2.5
imdb_score // 7
imdb_score > 7
director = movie['director_name']
director == 'James Cameron'
```

Operadores sobre series

```
imdb_score.add(1)                  # imdb_score + 1
imdb_score.mul(2.5)                # imdb_score * 2.5
imdb_score.floordiv(7)             # imdb_score // 7
imdb_score.gt(7)                   # imdb_score > 7
director.eq('James Cameron')       # director == 'James Cameron'
imdb_score.astype(int).mod(5)
a = type(1)
type(a)
a = type(imdb_score)
a([1,2,3])
```

Encadenando metodos de la serie

```
movie = pd.read_csv('movie.csv')
actor_1_fb_likes = movie['actor_1_facebook_likes']
director = movie['director_name']

director.value_counts().head(3)
actor_1_fb_likes.isnull().sum()
actor_1_fb_likes.dtype
actor_1_fb_likes.fillna(0).astype(int).head()
actor_1_fb_likes.isnull().mean()
```

Indice Significativo

```
movie = pd.read_csv('movie.csv')
movie.shape
movie2 = movie.set_index('movie_title')
movie2
```

Indice Significativo

movie_title	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	actor_1_facebook_likes	
Avatar	Color	James Cameron	723.0	178.0	0.0	855.0	Joel David Moore	1000.0	76050
Pirates of the Caribbean: At World's End	Color	Gore Verbinski	302.0	169.0	563.0	1000.0	Orlando Bloom	40000.0	30940
Spectre	Color	Sam Mendes	602.0	148.0	0.0	161.0	Rory Kinnear	11000.0	20007
...	...	...	...	...	...	...	...	...	
A Plague So Pleasant	Color	Benjamin Roberds	13.0	76.0	0.0	0.0	Maxwell Moody	0.0	
Shanghai Calling	Color	Daniel Hsia	14.0	100.0	0.0	489.0	Daniel Henney	946.0	1
My Date with Drew	Color	Jon Gunn	43.0	90.0	16.0	16.0	Brian Herzlinger	86.0	8

4916 rows × 27 columns

pd.read_csv('movie.csv', index_col='movie_title')

Indice Significativo

movie_title	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	actor_1_facebook_likes	
Avatar	Color	James Cameron	723.0	178.0	0.0	855.0	Joel David Moore	1000.0	7
Pirates of the Caribbean: At World's End	Color	Gore Verbinski	302.0	169.0	563.0	1000.0	Orlando Bloom	40000.0	3
Spectre	Color	Sam Mendes	602.0	148.0	0.0	161.0	Rory Kinnear	11000.0	2
...	...	...	...	...	...	...	...	...	
A Plague So Pleasant	Color	Benjamin Roberds	13.0	76.0	0.0	0.0	Maxwell Moody	0.0	

```
movie2.reset_index()
```

Renombrar los nombres de las filas y columnas

```
movie = pd.read_csv('movie.csv', index_col='movie_title')
idx_rename = {'Avatar':'AVATARES', 'Spectre': 'Espectro'}
col_rename = {'director_name':'Nombre del Director',
              'num_critic_for_reviews': 'Revisiones Críticas'}
movie.rename(index=idx_rename, columns=col_rename).head()
```

Renombrar los nombres de las filas y columnas

```
movie = pd.read_csv('data/movie.csv', index_col='movie_title')
index = movie.index
columns = movie.columns
index_list = index.tolist()
column_list = columns.tolist()
index_list[0] = 'AVATAR'
index_list[2] = 'Espectro'
column_list[1] = 'Nombre del Director'
column_list[2] = 'Revisiones Criticas'
```

Renombrar los nombres de las filas y columnas

```
print(index_list[:5])
print(column_list)
movie.index = index_list
movie.columns = column_list
movie.head()
```

Creando y Borrando Columnas

```
movie = pd.read_csv('movie.csv')
movie['Pelis_Vistas'] = 0
movie.columns
```

Creando y Borrando Columnas

```
movie['FacebookLikesActorDirector'] =  (movie['actor_1_facebook_likes'] +
                                        movie['actor_2_facebook_likes'] +
                                        movie['actor_3_facebook_likes'] +
                                        movie['director_facebook_likes'])

movie['FacebookLikesActorDirector'].isnull().sum()
```

Creando y Borrando Columnas

```
movie['is_cast_likes_more'] = (movie['cast_total_facebook_likes'] >=
                               movie['FacebookLikesActorDirector'])
```

```
movie['is_cast_likes_more'].all()
```

```
movie = movie.drop('FacebookLikesActorDirector', axis='columns')
```

```
movie['TotalFaceLikesActor'] = (movie['actor_1_facebook_likes'] +
                                movie['actor_2_facebook_likes'] +
                                movie['actor_3_facebook_likes'])
movie['TotalFaceLikesActor'] = movie['TotalFaceLikesActor'].fillna(0)
```

```
movie['is_cast_likes_more'] = movie['cast_total_facebook_likes'] >= \
                              movie['TotalFaceLikesActor']
movie['is_cast_likes_more'].all()
```

```
movie['ActorRepartoLikePCT'] = (movie['TotalFaceLikesActor'] /
                                movie['cast_total_facebook_likes'])

movie['ActorRepartoLikePCT'].min(), movie['ActorRepartoLikePCT'].max()
```

```
movie.set_index('movie_title')['ActorRepartoLikePCT'].head()
```

```
profit_index = movie.columns.get_loc('gross') + 1
profit_index
```

```
movie.insert(loc=profit_index,
             column='Ganancias',
             value=movie['gross'] - movie['budget'])
movie.head()
```

References

- Python Programming: An Introduction to Computer Science. John Zelle
- Big Data con Python. Rafael Caballero Enrique Martín Adrián Riesco
- Aprende Python en un Fin de Semana Alfredo Moreno Muñoz Sheila Córcoles Córcoles
- Learn Python Programming Fabrizio Romano
- Python Data Analytics Fabio Nelli
- Expert Python Programming Michael Jasworski Tarek Ziadé
- Statistical analysis of questionnaires: a unified approach based on R and Stata by Francesco Bartolucci. Boca Raton: CRC Press, 2016.
- Data visualisation: a handbook for data driven design by Andy Kirk. Los Angeles: Sage, 2016.
- Learning tableau: leverage the power of tableau 9.0 to design rich data visualizations and build fully interactive dashboards by Joshua N. Milligan. Mumbai: Packt Publishing, 2015.