Algorítmica y Programación

Enero - Mayo 2020

Dr. Iván S. Razo Zapata
(ivan.razo@itam.mx)

ITAM

Pandas intro

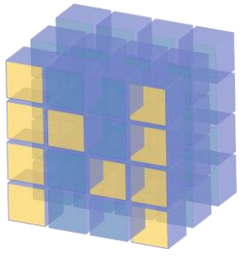
Pandas

- Panel data
- Datos multi-dimensionales que reflejan mediciones a través de ciertos periodos de tiempo

person	year	income	age	sex
1	2016	1300	27	1
1	2017	1600	28	1
1	2018	2000	29	1
2	2016	2000	38	2
2	2017	2300	39	2
2	2018	2400	40	2

pandas

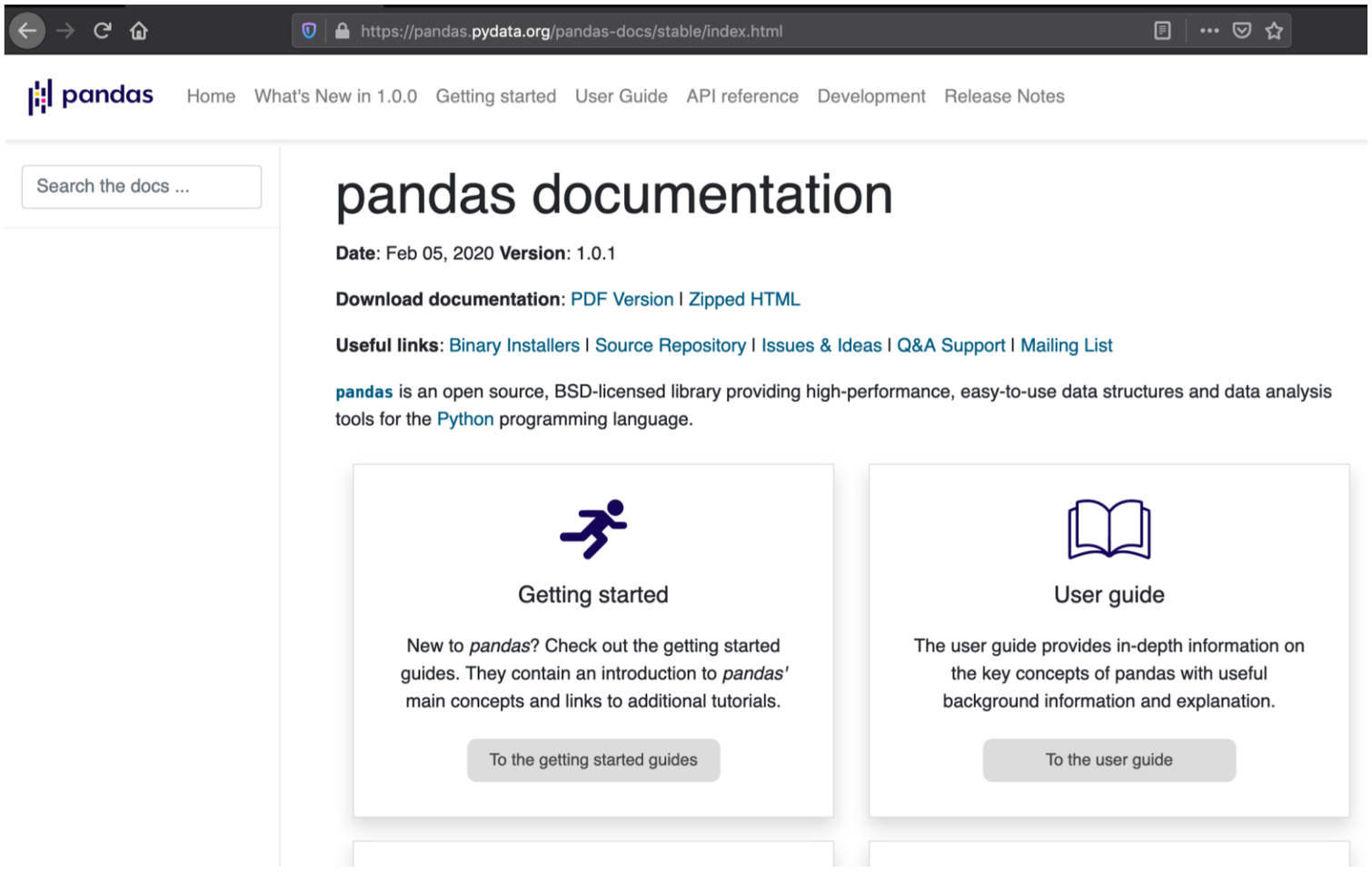
$$y_i t = \beta' x_{it} + \mu_i + \epsilon_{it}$$



NumPy

ITAM

Pandas



pandas: powerful Python data analysis toolkit

Release 1.0.1

Wes McKinney and the Pandas Development Team

Feb 05, 2020

Convenciones

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Series

- Arreglo unidimensional
- Contenido homogéneo
- Usualmente para representar series de tiempo

Series - creación

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Creacion de una Serie
serie1 = pd.Series([1,2,3])

# Creacion desde una lista
lista1 = [4,5,6]
serie2 = pd.Series(lista1)

# Creacion desde un arreglo numpy
arreglo1 = np.array([7,8,9])
serie3   = pd.Series(arreglo1)
```

Series - creación

Indices

valores

```
In [23]: serie1
Out[23]:
0    1
1    2
2    3
dtype: int64

In [24]: serie2
Out[24]:
0    4
1    5
2    6
dtype: int64

In [25]: serie3
Out[25]:
0    7
1    8
2    9
dtype: int64
```

Series - indices y valores

```
In [27]: serie1.index
Out[27]: RangeIndex(start=0, stop=3, step=1)

In [28]: serie1.values
Out[28]: array([1, 2, 3])

In [29]: serie2.index
Out[29]: RangeIndex(start=0, stop=3, step=1)

In [30]: serie2.values
Out[30]: array([4, 5, 6])

In [31]: serie3.index
Out[31]: RangeIndex(start=0, stop=3, step=1)

In [32]: serie3.values
Out[32]: array([7, 8, 9])
```

ITAM

Series - creación

```
import numpy as np
import pandas as pd

# Creacion definiendo indices y valores en base a listas
lista2 = [10, 11, 12]
lista3 = ['A1', 'A2', 'A3']
serie4 = pd.Series(lista2, index=lista3)

# Creacion con indices en base a una lista y valores en base a un arreglo
arreglo2 = np.array([13, 14, 15])
lista5 = ['B1', 'B2', 'B3']
serie5 = pd.Series(arreglo2, index=lista5)
```

```
In [39]: serie4
Out[39]:
A1    10
A2    11
A3    12
dtype: int64

In [40]: serie5
Out[40]:
B1    13
B2    14
B3    15
dtype: int64
```

ITAM

Series - accediendo a información

```
In [43]: serie5[0]
Out[43]: 13

In [44]: serie5[1]
Out[44]: 14

In [45]: serie5[2]
Out[45]: 15
```

```
In [46]: serie5['B1']
Out[46]: 13

In [47]: serie5['B2']
    ...:
Out[47]: 14

In [48]: serie5['B3']
Out[48]: 15
```

ITAM

Series - accediendo a información

```
In [49]: serie5['B4']
Traceback (most recent call last):

  File "<ipython-input-49-9eb5bae69fae>", line 1, in <module>
    serie5['B4']

  File "/anaconda3/lib/python3.7/site-packages/pandas/core/series.py", line 767, in __getitem__
    result = self.index.get_value(self, key)

  File "/anaconda3/lib/python3.7/site-packages/pandas/core/indexes/base.py", line 3132, in get_value
    raise e1

  File "/anaconda3/lib/python3.7/site-packages/pandas/core/indexes/base.py", line 3118, in get_value
    tz=getattr(series.dtype, 'tz', None))

  File "pandas/_libs/index.pyx", line 106, in pandas._libs.index.IndexEngine.get_value

  File "pandas/_libs/index.pyx", line 114, in pandas._libs.index.IndexEngine.get_value

  File "pandas/_libs/index.pyx", line 162, in pandas._libs.index.IndexEngine.get_loc

  File "pandas/_libs/hashtable_class_helper.pxi", line 1492, in
pandas._libs.hashtable.PyObjectHashTable.get_item

  File "pandas/_libs/hashtable_class_helper.pxi", line 1500, in
pandas._libs.hashtable.PyObjectHashTable.get_item

KeyError: 'B4'
```

ITAM

Series - accediendo a información

```
In [50]: serie5[4]
Traceback (most recent call last):

  File "<ipython-input-50-fc0f57d0c83d>", line 1, in <module>
    serie5[4]

  File "/anaconda3/lib/python3.7/site-packages/pandas/core/series.py", line 767, in __getitem__
    result = self.index.get_value(self, key)

  File "/anaconda3/lib/python3.7/site-packages/pandas/core/indexes/base.py", line 3124, in get_value
    return libindex.get_value_box(s, key)

  File "pandas/_libs/index.pyx", line 55, in pandas._libs.index.get_value_box

  File "pandas/_libs/index.pyx", line 70, in pandas._libs.index.get_value_box

IndexError: index out of bounds
```

Series - creación usando diccionarios

```
import pandas as pd

# Creacion en base a un diccionario
diccionario1 = {'C1':[9.87, 3, 'A'],  'C2':[11.23, 4, 'B'],
                'C3':[21.13, 2, 'C'], 'C4':[6.48, 1, 'D']}

serie6 = pd.Series(diccionario1)
```

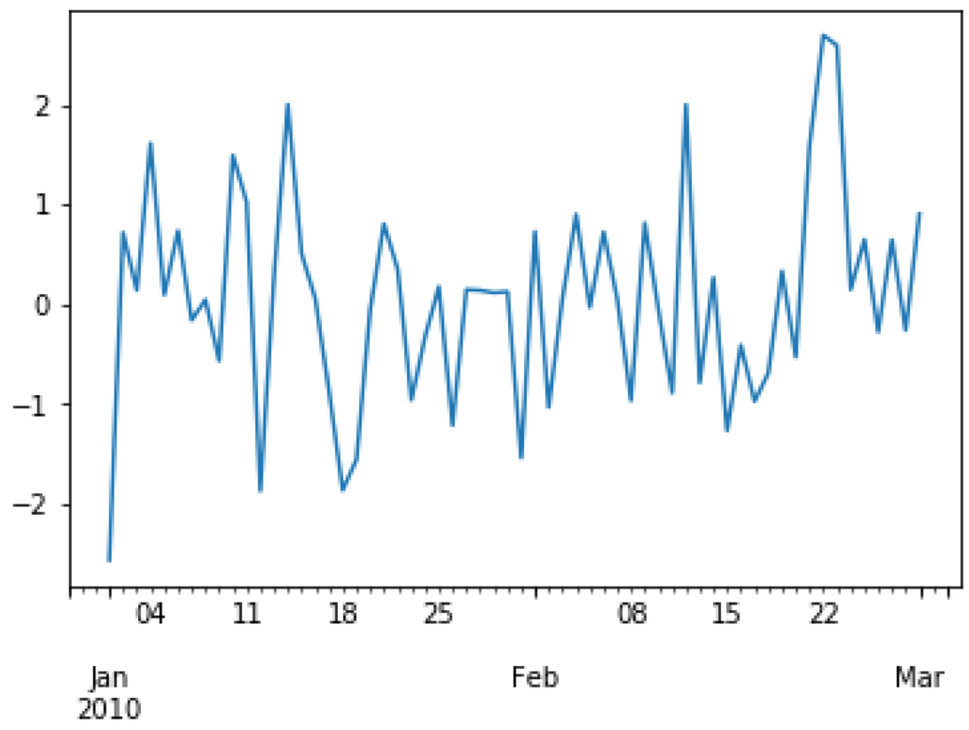
```
In [56]: serie6
Out[56]:
C1      [9.87, 3, A]
C2     [11.23, 4, B]
C3     [21.13, 2, C]
C4      [6.48, 1, D]
dtype: object
```

Series - accediendo a información

```
In [56]: serie6
Out[56]:
C1      [9.87, 3, A]
C2     [11.23, 4, B]
C3     [21.13, 2, C]
C4      [6.48, 1, D]
dtype: object

In [57]: serie6[0]
Out[57]: [9.87, 3, 'A']

In [58]: serie6[1]
Out[58]: [11.23, 4, 'B']

In [59]: serie6[2]
Out[59]: [21.13, 2, 'C']

In [60]: serie6['C1']
Out[60]: [9.87, 3, 'A']

In [61]: serie6['C3']
Out[61]: [21.13, 2, 'C']
```

Series - creación con contenido aleatorio

```
import pandas as pd
import numpy as np

serie7 = pd.Series(np.random.randn(60),
                   index=pd.date_range('1/1/2010', periods=60))

serie7.plot()
```



Series - accediendo a información

```
In [15]: serie7[0]
Out[15]: -1.2196133642573532

In [16]: serie7['2010-01-01']
Out[16]: -1.2196133642573532

In [17]: serie7[0:10]
Out[17]:
2010-01-01   -1.219613
2010-01-02   -0.105280
2010-01-03   -1.541724
2010-01-04    0.940399
2010-01-05   -0.471789
2010-01-06   -1.692668
2010-01-07    0.241670
2010-01-08    0.196089
2010-01-09   -0.855853
2010-01-10   -0.343446
Freq: D, dtype: float64

In [18]: serie7['2010-01-01':'2010-01-10']
Out[18]:
2010-01-01   -1.219613
2010-01-02   -0.105280
2010-01-03   -1.541724
2010-01-04    0.940399
2010-01-05   -0.471789
2010-01-06   -1.692668
2010-01-07    0.241670
2010-01-08    0.196089
2010-01-09   -0.855853
2010-01-10   -0.343446
Freq: D, dtype: float64
```

Series - algunas funciones

```
In [37]: serie7.sum()
Out[37]: -6.753455660506372

In [38]: serie7.mean()
Out[38]: -0.1125575943417729

In [39]: serie7.max()
Out[39]: 2.277350554836149 2

In [40]: serie7.min()
Out[40]: -2.0211781525731394
```

```
In [41]: serie7.head()
Out[41]:
2010-01-01   -0.078318
2010-01-02   -0.210397
2010-01-03    0.355277
2010-01-04    0.166112
2010-01-05   -0.186238
Freq: D, dtype: float64

In [42]: serie7.tail()
Out[42]:
2010-02-25   -0.661954
2010-02-26   -1.110969
2010-02-27   -0.124217
2010-02-28    0.607738
2010-03-01   -0.587280
Freq: D, dtype: float64
```

```
In [43]: serie7.head(3)
Out[43]:
2010-01-01   -0.078318
2010-01-02   -0.210397
2010-01-03    0.355277
Freq: D, dtype: float64

In [44]: serie7.tail(2)
Out[44]:
2010-02-28    0.607738
2010-03-01   -0.587280
Freq: D, dtype: float64

In [45]: serie8 = serie7 * 10

In [46]: serie8.head(3)
Out[46]:
2010-01-01   -0.783181
2010-01-02   -2.103971
2010-01-03    3.552766
Freq: D, dtype: float64

In [47]: serie8.tail(2)
Out[47]:
2010-02-28    6.077377
2010-03-01   -5.872801
Freq: D, dtype: float64
```

ITAM

Series - Ejercicio

- Generar una serie que contenga todos los días del 2019 y utilizando randn asigne un valor aleatorio a cada día

- Para cada mes calcular la media y crear una nueva serie (serieM) con dichos valores

- Remplazar los valores negativos de cada mes por la media de serieM

Series - creación desde un DataFrame

1 - Generar archivo csv

2 - Código en Python

```
import pandas as pd

dataFrame1 = pd.read_csv("DataSets/CDMX.csv", encoding='latin1')
dataFrame1.set_index('Alcaldia', inplace=True)
serie1 = dataFrame1['Poblacion']
```

```
In [28]: dataFrame1
Out[28]:
                          Poblacion  Superficie
Alcaldia
Álvaro Obregón               727034       96.17
Azcapotzalco                 414711       33.66
Benito Juárez                385439       26.63
Coyoacán                     620416       54.40
Cuajimalpa                   186391       74.58
Cuauhtémoc                   531831       32.40
Gustavo A. Madero           1185772       94.07
Iztacalco                    384326       23.30
Iztapalapa                  1815786      117.00
La Magdalena Contreras       239086       74.58
Miguel Hidalgo               372889       46.99
Milpa Alta                   130582      228.41
Tláhuac                      360265       85.34
Tlalpan                      650567      340.07
Venustiano Carranza          430978       33.40
Xochimilco                   415007      118.00

In [29]: serie1
Out[29]:
Alcaldia
Álvaro Obregón               727034
Azcapotzalco                 414711
Benito Juárez                385439
Coyoacán                     620416
Cuajimalpa                   186391
Cuauhtémoc                   531831
Gustavo A. Madero           1185772
Iztacalco                    384326
Iztapalapa                  1815786
La Magdalena Contreras       239086
Miguel Hidalgo               372889
Milpa Alta                   130582
Tláhuac                      360265
```

ITAM

Series - Ejercicio

- Crear dataFrame y generar series para emisiones en 1990, 2005, y 2017

- Calcular medias para cada año

- Países con más emisiones por año

CO_2 emissions per capita from 1900 to 2017.[17]

Fossil CO_2 Emissions by country/region [edit]

Country[18]	Fossil CO_2 Emissions (Mt CO_2/yr)			Fossil CO_2 Emissions		2017 – Fossil CO_2 Emissions	
	1990	2005	2017	2017 (% of world)	2017 vs 1990: change (%)	Per Land Area (t CO_2/km²/yr)	Per Capita (t CO_2/cap/yr)
World	22,674.116	30,049.809	37,077.404	100.00%	63.5%	73	4.9
World – International Aviation	258.941	422.777	543.381	1.47%	109.8%	n/a	n/a
World – International Shipping	371.804	572.169	677.248	1.83%	82.2%	n/a	n/a
Afghanistan	2.546	1.063	11.422	0.03%	348.6%	18	0.3
Albania	6.583	4.196	5.026	0.01%	-23.7%	175	1.7
Algeria	65.677	98.197	159.929	0.43%	143.5%	67	3.9
Angola	5.851	15.975	30.876	0.08%	427.7%	25	1.0
Anguilla	0.006	0.014	0.028	0.00%	366.7%	308	1.9
Antigua and Barbuda	0.223	0.283	0.624	0.00%	179.8%	1,412	6.1
Argentina	112.434	165.429	209.968	0.57%	86.7%	76	4.7
Armenia	20.699	4.542	4.832	0.01%	-76.7%	162	1.6
Aruba	0.297	0.470	0.959	0.00%	222.9%	5,328	9.1
Australia	275.408	391.590	402.253	1.08%	46.1%	52	16.5
Austria	62.918	80.994	72.249	0.19%	14.8%	861	8.3
Azerbaijan	58.077	30.485	32.544	0.09%	-44.0%	376	3.3
Bahamas	1.524	2.068	2.997	0.01%	96.7%	215	7.6
Bahrain	11.988	23.388	35.775	0.10%	198.4%	46,643	24.0
Bangladesh	13.868	38.834	84.546	0.23%	509.6%	573	0.5

Series - Ejercicio

```
import pandas as pd

dataFrame1 = pd.read_csv("DataSets/CO2.csv", encoding='latin1')
dataFrame1.set_index('Country', inplace=True)
serie1 = dataFrame1['1990']
serie2 = dataFrame1['2005']
serie3 = dataFrame1['2017']

print("Media para 1990 es: ",serie1.mean())
print("Media para 2005 es: ",serie2.mean())
print("Media para 2017 es: ",serie3.mean())
```

```
max1990 = serie1[serie1 == serie1.max()]
max2005 = serie2[serie2 == serie2.max()]
max2017 = serie3[serie3 == serie3.max()]

print("El pais que mas emitio CO2 en 1990 fue {a:s} con : {b:7.2f} toneladas".
     format(a = max1990.index[0], b = max1990[0]))

print("El pais que mas emitio CO2 en 2005 fue {a:s} con : {b:7.2f} toneladas".
     format(a = max2005.index[0], b = max2005[0]))

print("El pais que mas emitio CO2 en 2017 fue {a:s} con : {b:7.2f} toneladas".
     format(a = max2017.index[0], b = max2017[0]))
```

Series - Ejercicio

```
In [1]: runfile('C:/Users/irazoz/Documents/ITAM/Cursos/2020/AyP-2020-Enero-Mayo/Ejemplos/Ejemplo16.py',
wdir='C:/Users/irazoz/Documents/ITAM/Cursos/2020/AyP-2020-Enero-Mayo/Ejemplos')
Media para 1990 es:  126.63960287081352
Media para 2005 es:  159.24890430622006
Media para 2017 es:  188.56757416267936
El pais que mas emitio CO2 en 1990 fue  United States con : 5085.90 toneladas
El pais que mas emitio CO2 en 2005 fue  China con : 6263.06 toneladas
El pais que mas emitio CO2 en 2017 fue  China con : 10877.22 toneladas
```