Preparaci?n de datos

January 31, 2018

0.1 Python Scientific Stack

- NumPy
- Scipy
- Jupyter (Ipython)
- matplotlib
- pandas

ąPor suerte ya lo tenemos con ANACONDA!

0.2 Numpy (Numerical Python)

http://www.numpy.org/

Las listas en python son muy poderosas y versátiles pero fallan en un aspecto importante para la ciencia de datos.

NumPy agrega mayor soporte para arreglos y matrices, constituyendo una biblioteca de funciones matemáticas de alto nivel para operar con esos vectores o matrices.

0.2.1 Arreglos

$$y = [0, 1, 2, 3, 4]$$

```
In [1]: import numpy as np

        x = [0, 1, 2, 3, 4]
        y = np.array(x)
        y

Out[1]: array([0, 1, 2, 3, 4])

In [2]: help(np.array)

Help on built-in function array in module numpy.core.multiarray:

array(...)
    array(object, dtype=None, copy=True, order=None, subok=False, ndmin=0)

    Create an array.
```

```
Parameters
----------
object : array_like
    An array, any object exposing the array interface, an
    object whose __array__ method returns an array, or any
    (nested) sequence.
dtype : data-type, optional
    The desired data-type for the array.  If not given, then
    the type will be determined as the minimum type required
    to hold the objects in the sequence.  This argument can only
    be used to 'upcast' the array.  For downcasting, use the
    .astype(t) method.
copy : bool, optional
    If true (default), then the object is copied.  Otherwise, a copy
    will only be made if __array__ returns a copy, if obj is a
    nested sequence, or if a copy is needed to satisfy any of the other
    requirements (`dtype`, `order`, etc.).
order : {'C', 'F', 'A'}, optional
    Specify the order of the array.  If order is 'C', then the array
    will be in C-contiguous order (last-index varies the fastest).
    If order is 'F', then the returned array will be in
    Fortran-contiguous order (first-index varies the fastest).
    If order is 'A' (default), then the returned array may be
    in any order (either C-, Fortran-contiguous, or even discontiguous),
    unless a copy is required, in which case it will be C-contiguous.
subok : bool, optional
    If True, then sub-classes will be passed-through, otherwise
    the returned array will be forced to be a base-class array (default).
ndmin : int, optional
    Specifies the minimum number of dimensions that the resulting
    array should have.  Ones will be pre-pended to the shape as
    needed to meet this requirement.

Returns
-------
out : ndarray
    An array object satisfying the specified requirements.

See Also
--------
empty, empty_like, zeros, zeros_like, ones, ones_like, fill

Examples
--------
>>> np.array([1, 2, 3])
array([1, 2, 3])

Upcasting:
```

```
>>> np.array([1, 2, 3.0])
array([ 1.,  2.,  3.])

More than one dimension:

>>> np.array([[1, 2], [3, 4]])
array([[1, 2],
       [3, 4]])

Minimum dimensions 2:

>>> np.array([1, 2, 3], ndmin=2)
array([[1, 2, 3]])

Type provided:

>>> np.array([1, 2, 3], dtype=complex)
array([ 1.+0.j,  2.+0.j,  3.+0.j])

Data-type consisting of more than one element:

>>> x = np.array([(1,2),(3,4)],dtype=[('a','<i4'),('b','<i4')])
>>> x['a']
array([1, 3])

Creating an array from sub-classes:

>>> np.array(np.mat('1 2; 3 4'))
array([[1, 2],
       [3, 4]])

>>> np.array(np.mat('1 2; 3 4'), subok=True)
matrix([[1, 2],
        [3, 4]])
```

```
In [3]: type(y)

Out[3]: numpy.ndarray

In [4]: y * y

Out[4]: array([ 0,  1,  4,  9, 16])

In [5]: np.ndim(y)

Out[5]: 1
```

0.2.2 Matrices (Arreglos de 2 dimensiones)

$$x = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

```
In [6]: x = np.array([[1,2,3],[4,5,6],[7,8,9]])
        print(x)
        x
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

```
Out[6]: array([[1, 2, 3],
               [4, 5, 6],
               [7, 8, 9]])
```

```
In [7]: np.ndim(x)
```

```
Out[7]: 2
```

0.2.3 Selección escalar

```
In [8]: x = np.array([[1.0,2,3],[4,5,6]])
        x[1,2]
```

```
Out[8]: 6.0
```

0.2.4 Array slicing

- a[comienzo:fin] # elementos desde el índice del comienzo hasta el índice fin-1
- a[comienzo:] # del número en comienzo hasta el fin
- a[:fin] # desde el principio hasta fin-1
- a[:] # todo el arreglo
- a[comienzo:fin:paso] # elementos desde el índice del comienzo hasta el índice fin-1, por paso

0.2.5 Operaciones con arreglos y matrices

Siguiente clase!

0.2.6 Generación de arreglos y matrices

Siguiente clase!

0.3 Pandas!

- pandas es un módulo de alto rendimiento que ofrece un amplio conjunto de estructuras para trabajar con datos.
- pandas ayuda al manejo de datos estructurados que contienen muchas variables
- permite el manejo de "missing values"
- pandas también proporciona métodos robustos para la importación y exportación de una amplia gama de formatos.

```
In [9]: # pd por convención
        import pandas as pd

        data = pd.read_csv('train.csv')
        data.head()
```

```
Out[9]:    PassengerId  Survived  Pclass  \
        0            1         0       3
        1            2         1       1
        2            3         1       3
        3            4         1       1
        4            5         0       3

                                                        Name     Sex   Age  SibSp  \
        0                            Braund, Mr. Owen Harris    male  22.0      1
        1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
        2                             Heikkinen, Miss. Laina  female  26.0      0
        3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
        4                           Allen, Mr. William Henry    male  35.0      0

           Parch            Ticket     Fare Cabin Embarked
        0      0         A/5 21171   7.2500   NaN        S
        1      0          PC 17599  71.2833   C85        C
        2      0  STON/O2. 3101282   7.9250   NaN        S
        3      0            113803  53.1000  C123        S
        4      0            373450   8.0500   NaN        S
```

0.3.1 Estructuras de datos

pandas provee estructuras para el manejo de datos, Series, DataFrames y Panels.

- Series: son arreglos de una dimensión.
- DataFrames: son colecciones de series (2 dimensiones).
- Panels: son colecciones de DataFrames (3 dimensiones).

```
In [10]: type(data)
```

```
Out[10]: pandas.core.frame.DataFrame
```

```
In [11]: s = pd.Series([0.1, 1.2, 2.3, 3.4, 4.5])
         s
```

```
Out[11]: 0    0.1
         1    1.2
         2    2.3
         3    3.4
         4    4.5
         dtype: float64

In [12]: s = pd.Series([0.1, 1.2, 2.3, 3.4, 4.5], index = ['a','b','c','d','e'])
         s

Out[12]: a    0.1
         b    1.2
         c    2.3
         d    3.4
         e    4.5
         dtype: float64
```

El índice es parte de la "magia" de las estructuras de datos en pandas

```
In [13]: s[['a','c']]

Out[13]: a    0.1
         c    2.3
         dtype: float64

In [14]: s[s>2]

Out[14]: c    2.3
         d    3.4
         e    4.5
         dtype: float64

In [15]: data.tail()

Out[15]:      PassengerId  Survived  Pclass                                          Name  \
         886          887         0       2                         Montvila, Rev. Juozas
         887          888         1       1                  Graham, Miss. Margaret Edith
         888          889         0       3      Johnston, Miss. Catherine Helen "Carrie"
         889          890         1       1                          Behr, Mr. Karl Howell
         890          891         0       3                            Dooley, Mr. Patrick

                 Sex   Age  SibSp  Parch      Ticket   Fare Cabin Embarked
         886    male  27.0      0      0      211536  13.00   NaN        S
         887  female  19.0      0      0      112053  30.00   B42        S
         888  female   NaN      1      2  W./C. 6607  23.45   NaN        S
         889    male  26.0      0      0      111369  30.00  C148        C
         890    male  32.0      0      0      370376   7.75   NaN        Q

In [16]: data.describe()
```

```
Out[16]:        PassengerId    Survived      Pclass         Age       SibSp  \
         count   891.000000  891.000000  891.000000  714.000000  891.000000
         mean    446.000000    0.383838    2.308642   29.699118    0.523008
         std     257.353842    0.486592    0.836071   14.526497    1.102743
         min       1.000000    0.000000    1.000000    0.420000    0.000000
         25%     223.500000    0.000000    2.000000   20.125000    0.000000
         50%     446.000000    0.000000    3.000000   28.000000    0.000000
         75%     668.500000    1.000000    3.000000   38.000000    1.000000
         max     891.000000    1.000000    3.000000   80.000000    8.000000

                     Parch        Fare
         count  891.000000  891.000000
         mean     0.381594   32.204208
         std      0.806057   49.693429
         min      0.000000    0.000000
         25%      0.000000    7.910400
         50%      0.000000   14.454200
         75%      0.000000   31.000000
         max      6.000000  512.329200

In [17]: age = data['Age']
         sum(age.isnull())

Out[17]: 177

In [18]: np.mean(age)

Out[18]: 29.69911764705882

In [19]: age = data['Age'].fillna(29.69)

In [20]: sum(age.isnull())

Out[20]: 0

In [21]: pclass = data['Pclass']
         pclass.unique()

Out[21]: array([3, 1, 2])

In [22]: data.Pclass.head()

Out[22]: 0    3
         1    1
         2    3
         3    1
         4    3
         Name: Pclass, dtype: int64

In [23]: data[['Age', 'Pclass']].head()
```

```
Out[23]:     Age  Pclass
         0  22.0       3
         1  38.0       1
         2  26.0       3
         3  35.0       1
         4  35.0       3
```

Podemos borrar columnas usando los comando del, pop(), drop() - del modifica nuestro dataframe borrando la Serie seleccionada. - pop() borra la Serie pero la regresa como un output - drop() regresa un dataframe sin la Serie, pero no modifica el dataframe original

```
In [24]: del data['Ticket']
         data.head()
```

```
Out[24]:    PassengerId  Survived  Pclass  \
         0            1         0       3
         1            2         1       1
         2            3         1       3
         3            4         1       1
         4            5         0       3

                                                        Name     Sex   Age  SibSp  \
         0                            Braund, Mr. Owen Harris    male  22.0      1
         1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
         2                             Heikkinen, Miss. Laina  female  26.0      0
         3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
         4                           Allen, Mr. William Henry    male  35.0      0

            Parch     Fare Cabin Embarked
         0      0   7.2500   NaN        S
         1      0  71.2833   C85        C
         2      0   7.9250   NaN        S
         3      0  53.1000  C123        S
         4      0   8.0500   NaN        S
```

0.3.2 Datos

Los datos son características cualitativas o cuantitativas pertenecientes a un objeto, o un conjunto de objetos

0.3.3 Datos en bruto

"Raw data is a term for data collected on source which has not been subjected to processing or any other manipulation."

- Los datos en bruto llegan directamente de la fuente y no tienen la estructura necesaria para realizar análisis con ellos eficientemente.
- Requieren pre-procesamiento para ser utilizados.

- Por lo general suelen verse de la siguiente manera:

Video, audio, páginas web, también son fuentes de datos
Los datos no son lo más importante.
Lo más importante es la pregunta y los datos soportan la respuesta a nuestras interrogaciones.

0.3.4 Datos ordenados (Tidy data)

- Las variables deben de ser entendibles para el humano

Codebook! Documento para poder entender la información de la tabla.

- Descripción de las características con sus unidades
- Instrucciones sobre las transformaciones que aplicamos a nuestros datos en bruto para trabajarlos

ESTO ES MUY IMPORTANTE
Existen historias de terror:
http://www.cc.com/video-clips/dcyvro/the-colbert-report-austerity-s-spreadsheet-error

0.3.5 £Dónde consigo los datos?

0.4 Datasets publicos:

- http://archive.ics.uci.edu/ml/
- http://www.kaggle.com
- https://www.quora.com/Where-can-I-find-large-datasets-open-to-the-public

0.5 Datos en México

- http://datos.gob.mx
- http://data.mx.io
- http://www.inegi.org.mx/
- http://inegifacil.com/
- http://datos.imss.gob.mx/
- https://datos.jalisco.gob.mx/
- https://datosabiertos.unam.mx/

0.6 Bajar y leer archivos desde la web con python

Siguiente clase!

0.6.1 Lista de formatos para guardar información

https://en.wikipedia.org/wiki/List_of_file_formats
Algunos formatos esenciales:

- JSON (JavaScript Object Notation):

es un formato estándar abierto que utiliza el texto legible por humanos para transmitir objetos de datos que constan de pares atributo-valor . Es el formato de datos más común utilizado para la comunicación navegador / servidor asíncrono (AJAJ), en sustitución de gran parte XML que es utilizado por AJAX.

```
In [25]: {"employees":[
             {"firstName":"John", "lastName":"Doe"},
             {"firstName":"Anna", "lastName":"Smith"},
             {"firstName":"Peter", "lastName":"Jones"}
         ]}

Out[25]: {'employees': [{'firstName': 'John', 'lastName': 'Doe'},
           {'firstName': 'Anna', 'lastName': 'Smith'},
           {'firstName': 'Peter', 'lastName': 'Jones'}]}
```

- XML (Extensible Markup Language):

es un lenguaje que fue concebido para describir información. Su función principal es ayudarnos a organizar contenidos y eso hace que los documentos XML sean portables hacia diferentes tipos de aplicaciones. <employees> <employee> <firstName>John</firstName> <lastName>Doe</lastName> </employee> <employee> <firstName>Anna</firstName> <lastName>Smith</lastName> </employee> <employee> <firstName>Peter</firstName> <lastName>Jones</lastName> </employee> </employees>

- HTML (HyperText Markup Language):

por otro lado ha sido concebido para mostrar información, determinar cómo actúa y que hace. Su función radica en ayudarnos a darle formato a los diversos contenidos de una página.

- CSV (Comma-Separated Values)
- XLS (Microsoft Excel worksheet sheet)
- XLSX (Office Open XML worksheet sheet)

0.7 Dataset para el curso

0.7.1 Predecir la supervivencia en el Titanic

https://www.kaggle.com/c/titanic/data

El hundimiento del Titanic es uno de los naufragios más infames de la historia. El 15 de abril de 1912, durante su viaje inaugural, el Titanic se hundió después de chocar con un iceberg, matando de 1,502 a 2,224 pasajeros.

Una de las razones por las cuales se perdieron tantas vidas fue que no había suficientes botes salvavidas. Aunque hubo algún elemento de suerte involucrada en sobrevivir al hundimiento, algunos grupos de personas tenían más probabilidades de sobrevivir que otros, como las mujeres, los niños y personas de la clase alta.

Para este curso usaremos el dataset anterior para En este desafío , le pedimos que complete el análisis de qué tipo de personas eran propensos a sobrevivir . En particular , le pedimos que aplicar las herramientas de aprendizaje automático para predecir que los pasajeros sobrevivieron a la tragedia.

0.7.2 Matplotlib

Las visualizaciones son una de las herramientas más poderosas a su disposición para explorar los datos y comunicar tus ideas. La biblioteca pandas incluye capacidades básicas para graficar con el paquete matplotlib.

```
In [1]: import matplotlib
        %matplotlib inline
```

```
/Users/israel/anaconda3/envs/viakable/lib/python3.5/site-packages/matplotlib/font_manager.py:27
  warnings.warn('Matplotlib is building the font cache using fc-list. This may take a moment.')
/Users/israel/anaconda3/envs/viakable/lib/python3.5/site-packages/matplotlib/font_manager.py:27
  warnings.warn('Matplotlib is building the font cache using fc-list. This may take a moment.')
```

Histogramas

```
In [2]: data.hist(bins = 20, column="Age", figsize=(8,8), color="blue")
```

```
        ---------------------------------------------------------------------------

        NameError                                 Traceback (most recent call last)

        <ipython-input-2-8214a71a9286> in <module>()
    ----> 1 data.hist(bins = 20, column="Age", figsize=(8,8), color="blue")


        NameError: name 'data' is not defined
```