Introducci?n a python para ML

March 24, 2017

1 Introducción a Python para Machine Learning

1.1 5to año - Ingeniería en Sistemas de Información

1.1.1 Facultad Regional Villa María

1.1.2 Introducción

Python es un lenguaje de software libre, no tipado, interpretado y orientado a objetos. Se presenta como principal elección entre los lenguajes de machine learning junto con R, Matlab y Octave (dependiendo la comunidad que lo use).

Posee un enorme ecosistema de librerías de machine learning. Esto sumado a su flexibilidad y su simpleza de uso hacen que sea uno de los lenguajes más populares para la computación científica.

Cabe destacar que si bien el lenguaje es OO, las librerías desarrolladas en él en general siguen el paradigma funcional debido a que resulta más natural para este tipo de problemas y es más fácil de entender para las personas que no conocen el paradigma OO.

Algunos IDEs: Jupyter (web), Pycharm, Rodeo.

1.1.3 jupyter

Aplicación web de código abierto que permite crear y compartir documentos (notebooks) mostrando iterativamente el flujo de código en lenguajes como python y R, permitiendo incluir gráficos, explicaciones en Markdown y fórmulas en LaTeX.

Resulta un gran lenguaje para aplicaciones pequeñas de machine learning ya que permite utilizar el código de a fragmentos. Cada fragmento puede ser de texto o de código (dependiendo la selección del combo box superior: si está "Markdown" seleccionado tomará a la celda como texto - de estar "Code" tomará el fragmento como código).

Se recomienda como el IDE a utilizar en el transcurso de la materia.

¿Cómo instalarlo?

- En Linux:

```
sudo pip3 install jupyter
```

- En Windows:
  - Descargar la plataforma para DataScience "Anaconda" desde

https://www.continuum.io/downloads#windows

Se deberá seleccionar la versión del intérprete Python (3.6 recomendada), y la arquitectura de Windows correspondiente (32 o 64 bits).

- Instalar la plataforma Anaconda.

¿Cómo usarlo?

- En Linux: abrir la terminal, situarse en el directorio donde está el notebook, ejecutar jupyter notebook y abrir el notebook desde la interfaz web.
- En Windows: abrir la consola de windows (cmd), ejecutar jupyter notebook y abrir el notebook desde la interfaz web.

Prueben ejecutar y mirar el código del notebook de esta clase! Más notebooks de prueba: https://try.jupyter.org/.

Info adicional:

- Markdown es un conversor de texto plano a HTML. Funciona como suerte de versión "liviana" de LaTex.
- LaTex es un sistema de tipografía de alta calidad, que permite armar documentos con una sintaxis predefinida. Su sintaxis es mucho más verborrágica comparado con las herramientas típicas de ofimática, pero es mucho más flexible. Para armar documentos completos en LaTex de forma sencilla recomendamos Overleaf (editor LaTex Web). Por su parte, a LaTex en jupyter se lo utiliza principalmente para fórmulas.

Para sus explicaciones en texto, jupyter por defecto utiliza Markdown. Para utilizar LaTex, debe encerrarse la sentencia en $ ecuación $ para ecuaciones en la misma línea o bien entre $$ ecuación $$ para ecuaciones en una nueva línea.

Ejemplo

$$\lim_{x \to a} \frac{f(x) - f(a)}{x - a}$$

1.1.4 numpy

Librería open-source que dota a Python de manejo de vectores y matrices, junto con un conjunto de librerías para procesarlas, asemejando a python al manejo estilo Matlab.

El bloque básico de numpy es el ndarray, un array multidimensional de elementos homogéneos y de tamaño fijo. Estos objetos nunca son creados directamente, sino por un método destinado a tal efecto.

```
In [2]: # estructura fundamental de python: listas

        lista = [1,2,3,6]
        lista

Out[2]: [1, 2, 3, 6]

In [3]: lista2 = [1,2] + [3,6] # principal característica: listas completamente fle
        lista2
```

```
Out[3]: [1, 2, 3, 6]

In [4]: lista[1] # notar que Python y numpy indexan desde el 0

Out[4]: 2

In [5]: lista[1:] # ":" significa "todos los elementos siguientes"

Out[5]: [2, 3, 6]

In [6]: lista.append(5)
        lista.append('a')
        lista

Out[6]: [1, 2, 3, 6, 5, 'a']

In [7]: import numpy as np

        array = np.array([1,2,3,4,5])
        array

Out[7]: array([1, 2, 3, 4, 5])

In [8]: array[0]

Out[8]: 1

In [9]: matriz = np.array([[1,2],[3,4],[5,6],[7,8]]) # para python, una matriz es u
        matriz

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

In [10]: matriz[1,1]

Out[10]: 4

In [11]: matriz[1,:] # pedimos todos los elementos de la segunda fila

Out[11]: array([3, 4])

In [13]: np.linspace(1,2,5) # método muy útil para crear arrays de una dimensión

Out[13]: array([ 1.  ,  1.25,  1.5 ,  1.75,  2.  ])

In [17]: # comando muy útil para obtener ayuda.
         #Puede utilizarse sobre paquetes, métodos, u objetos que contengan documen

         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 [ ]:
```
# También puede utilizarse sin argumentos, lo cual invoca a la ayuda intera

    help()
```

```
Welcome to Python 3.6's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at http://docs.python.org/3.6/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".
```

1.1.5 sci-py

sci-py es una colección de librerías y algoritmos importante para machine learning. Normalmente el paquete a utilizar debe importarse por separado. Algunos ejemplos:

```
 cluster                      : Vector Quantization / Kmeans
 fftpack                      : Discrete Fourier Transform algorithms
 integrate                    : Integration routines
 interpolate                  : Interpolation Tools
 io                           : Data input and output
 lib                          : Python wrappers to external libraries
 lib.blas                     : Wrappers to BLAS library
 lib.lapack                   : Wrappers to LAPACK library
 linalg                       : Linear algebra routines
 misc                         : Various utilities that don't have another home.
 ndimage                      : n-dimensional image package
 odr                          : Orthogonal Distance Regression
 optimize                     : Optimization Tools
 signal                       : Signal Processing Tools
 sparse                       : Sparse Matrices
 sparse.linalg                : Sparse Linear Algebra
 sparse.linalg.dsolve         : Linear Solvers
 sparse.linalg.dsolve.umfpack : :Interface to the UMFPACK library:
 sparse.linalg.eigen          : Sparse Eigenvalue Solvers
 sparse.linalg.eigen.arpack   : Eigenvalue solver using iterative methods.
 sparse.linalg.eigen.lobpcg   : Locally Optimal Block Preconditioned
                                  Conjugate Gradient Method (LOBPCG)
 spatial                      : Spatial data structures and algorithms
 special                      : Airy Functions

 stats                        : Statistical Functions
```

```
In [14]: # ejemplo: distribución normal con media=0 y varianza=1
         from scipy import stats
         np.random.seed(13) # semilla aleatoria: muy útil para replicar experimento
         dist = stats.norm(0,1)
         r = dist.rvs(10) # diez muestras aleatorias
         p = dist.pdf(0) # pdf en x=0
         c = dist.cdf(0) # cdf en x=0
         print(r)
         print('pdf =',p)
         print('cdf =',c)
```

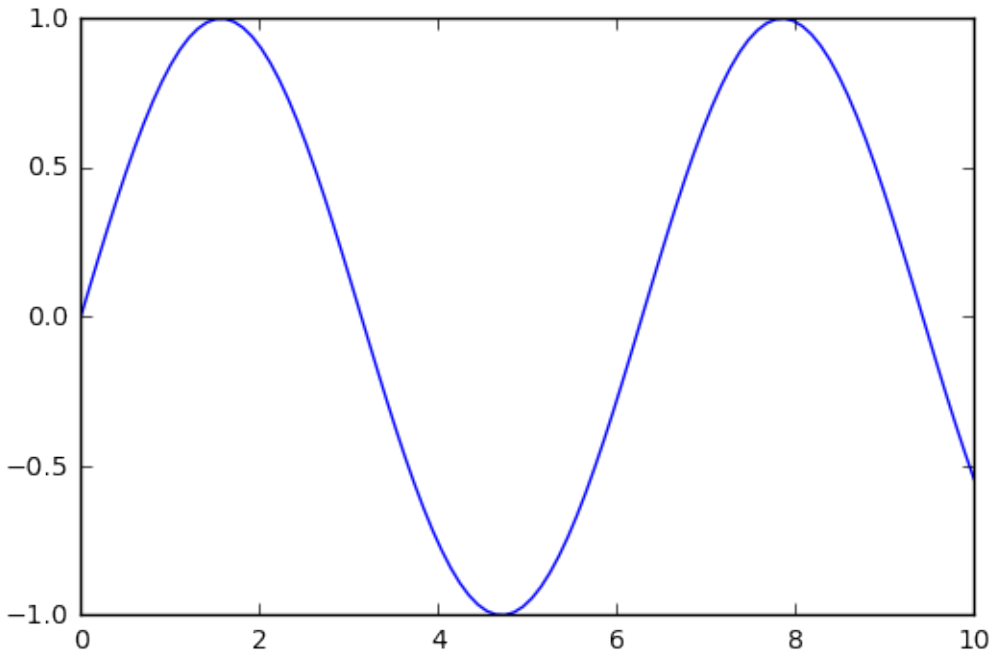
```
[-0.71239066  0.75376638 -0.04450308  0.45181234  1.34510171  0.53233789
  1.3501879   0.86121137  1.47868574 -1.04537713]
pdf = 0.398942280401
cdf = 0.5
```

1.1.6 matplotlib

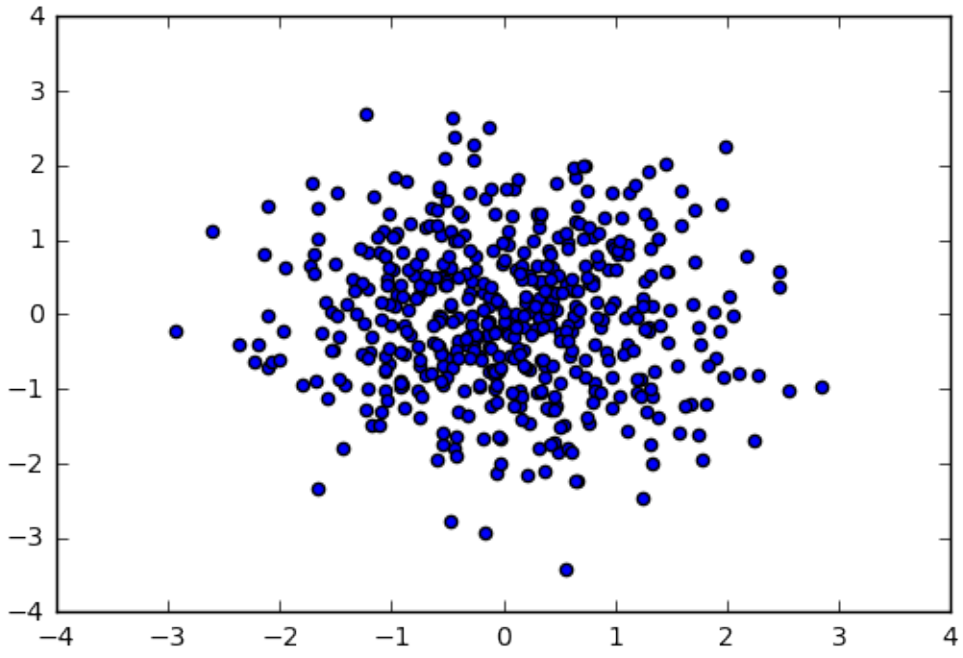
matplotlib es la librería estándar para la visualización de datos, algo que resulta fundamental para machine learning. Es muy flexible y simple de usar. Se integra con IPython, lo cual permite ver gráficos en un notebook.

La función principal de la misma es plot(), la cual toma un número variable de argumentos y grafica de forma tan simple como escribiendo plot(x,y)

```
In [15]: %matplotlib inline
         import matplotlib.pyplot as plt

         x = np.linspace(0, 10, 100) # x= 0.0, 0.101, 0.202,..., 10.0
         plt.plot(x, np.sin(x))
         plt.show()
```




```
In [18]: x = np.random.normal(size=500)
         y = np.random.normal(size=500)
         plt.scatter(x, y); # plot de puntos
         plt.show()
```



1.1.7 sci-kit learn

Es una librería que contiene un conjunto de algoritmos de ML de regresión, clasificación, clustering, etc. así como utilidades como sets de datos de ejemplo.

sci-kit learn intenta definir una interfaz común para todos los algoritmos. El método común de la gran mayoría de los estimadores es fit(), el cual adapta los datos de entrenamiento al modelo elegido. Dependiendo de la naturaleza de cada algoritmo, se definen distintos métodos comunes para cada uno de ellos.

Los datos en sci-kit learn, en la gran mayoría de los casos, se asumen como un array de dos dimensiones, de forma X=[n_ejemplos, n_características]. Cada ejemplo es un ítem a ser procesado (ejemplo, clasificado), pudiendo ser algo tan simple como una fila de excel a imágenes o videos.

Por su parte, las características, cuya cantidad se establece a-priori, se usan para describir cada ítem de forma cuantitativa. Normalmente son representadas por valores reales, aunque también puede tratarse de valores booleanos o discretos.

1.1.8 Dataset Iris

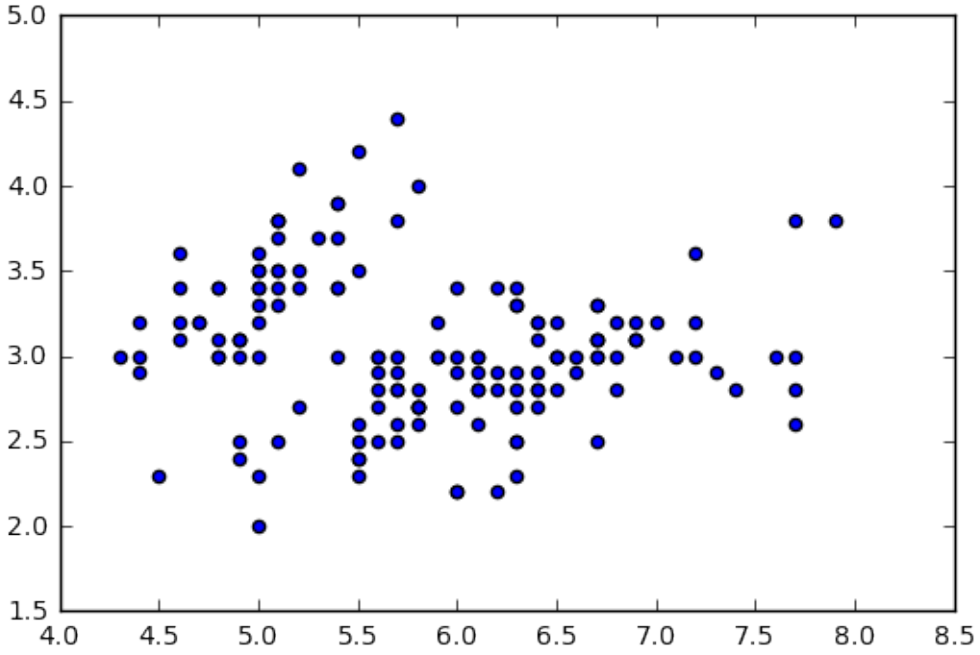
"Iris" es un dataset que sirve como "Hello World" de ML. Consiste en un conjunto de ejemplos que contiene tres flores de Iris: setosa, virginica y versicolor. Las mismas están distribuídas en 50 ejemplos de cada flor, y sus características indican el largo y ancho tanto de su sépalo como de su pétalo, y también se especifica de qué clase de Iris se trata (setosa, virgínica o versicolor).

```
In [20]: from sklearn.datasets import load_iris

         iris = load_iris()

         X, y = iris.data, iris.target

         plt.scatter(X[:,0], X[:,1]) # visualizamos cómo se relacionan el largo y e
         plt.xlabel= 'Largo del sépalo'
         plt.ylabel= 'Ancho del sépalo'
         plt.show

Out[20]: <function matplotlib.pyplot.show>
```

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

$$B = \begin{bmatrix} -1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}$$

Matrices Ejercicio Clase 1



1.1.9 Ejercicios

1. Instalar el entorno Python y un IDE de su elección para la próxima clase. Las clases van a estar enfocadas en Python y es muy recomendable que los ejercicios sean realizados en dicho lenguaje, pero pueden utilizar cualquier lenguaje a elección (en tal caso, tener en cuenta que muchos lenguajes no tienen un ecosistema amplio de librerías de ML).

2. Realizar una práctica básica con numpy: dadas las matrices de la figura "Matrices Ejercicio Clase 1"

- Construir las matrices en numpy
- Hacer con dichas matrices $A - B^T$ e imprimir su resultado.
- Hacer con dichas matrices $(A.B)^{-1}$ e imprimir su resultado.

Recomendación: Usar un notebook de jupyter para hacer los ejercicios.

Fecha de entrega: 03/04/2017

Formato para entregar los ejercicios: Enviar a inteligenciafrvm@gmail.com el link del ejercicio en su Github donde esté alojada la implementación semanal. Alternativamente, enviar el link del Gist público o privado donde está alojado el código. Lo ideal que estén los commits paso a paso para ver cómo se fue llevando a cabo el desarrollo del ejercicio.

Aclaración: Los ejercicios deben ser resueltos de forma **individual**.

```
In [ ]:
```