Sab_24_Jun

January 31, 2018

1 Python ?

2 1. Introducción a Python para la ciencia de datos:

Python es un lenguaje de programación de propósito general que se está volviendo más y más popular para hacer ciencia de datos.

2.1 1.1 Python Scientific Stack

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

3 2. Preparación de datos (Extraer, importar y limpiar):

```
- £Dónde están los datos que necesitas?
- £Cómo prepararlos para usarlos en tu caso?.
- Fuentes de Datos, Scrapping, Limpieza de Datos y Almacenamiento
```

4 3. Análisis Exploratorio de Datos:

```
Para conocer mejor la información con la que estás tratando, es mejor explorarla.
Análisis básico de datos, ayudado con herramientas gráficas.
Conceptos básicos de Pandas
```

---

5 Preparacion de datos / Fuentes de datos

5.1 £Dónde consigo los datos?

5.1.1 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

5.1.2 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/

5.2 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.

5.2.1 Arreglos

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

```
In [3]: import numpy as np

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

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

In [4]: 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 [5]: type(y)

Out[5]: numpy.ndarray

In [6]: y * y

Out[6]: array([ 0,  1,  4,  9, 16])

In [7]: np.ndim(y)

Out[7]: 1

In [13]: y.size

Out[13]: 5

In [14]: y.dtype

Out[14]: dtype('int64')

In [15]: np.identity(n = 5)
```

```
Out[15]: array([[ 1.,  0.,  0.,  0.,  0.],
                [ 0.,  1.,  0.,  0.,  0.],
                [ 0.,  0.,  1.,  0.,  0.],
                [ 0.,  0.,  0.,  1.,  0.],
                [ 0.,  0.,  0.,  0.,  1.]])

In [16]: np.ones(shape= [2,4])

Out[16]: array([[ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.]])

In [17]: np.zeros(shape= [2,4])

Out[17]: array([[ 0.,  0.,  0.,  0.],
                [ 0.,  0.,  0.,  0.]])

In [18]: y = [0, 1, 2, 3, 4]
         y * y


         ---------------------------------------------------------------------------

         TypeError                                 Traceback (most recent call last)

         <ipython-input-18-ebcb8d6f5d54> in <module>()
           2
           3 y = [0, 1, 2, 3, 4]
     ----> 4 y * y
           5


         TypeError: can't multiply sequence by non-int of type 'list'


In [19]: np.ndim(y)

Out[19]: 1
```

5.2.2 Matrices (Arreglos de dos dimensiones)

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

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

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

```
In [10]: x

Out[10]: array([[1, 2, 3],
                [4, 5, 6],
                [7, 8, 9]])

In [11]: np.ndim(x)

Out[11]: 2
```

5.2.3 Selección escalar

```
In [12]: x = np.array([[1.0,2,3],[4,5,6]])
         x[1,2]

Out[12]: 6.0
```

5.2.4 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

5.2.5 Operaciones con arreglos y matrices

Reshaping

```
In [21]: x

Out[21]: array([[ 1.,  2.,  3.],
                [ 4.,  5.,  6.]])

In [22]: x.flatten()

Out[22]: array([ 1.,  2.,  3.,  4.,  5.,  6.])

In [23]: x.dtype

Out[23]: dtype('float64')

In [24]: x.T

Out[24]: array([[ 1.,  4.],
                [ 2.,  5.],
                [ 3.,  6.]])
```

Operaciones La creación y manipulación de matrices es una buena herramienta, pero el verdadero poder de NumPy es la capacidad de realizar operaciones matemáticas con muchos valores rápida y fácilmente.

```
In [25]: x + 100

Out[25]: array([[ 101.,  102.,  103.],
                [ 104.,  105.,  106.]])

In [26]: x * 2

Out[26]: array([[  2.,   4.,   6.],
                [  8.,  10.,  12.]])

In [27]: x ** 2

Out[27]: array([[  1.,   4.,   9.],
                [ 16.,  25.,  36.]])

In [28]: x + x

Out[28]: array([[  2.,   4.,   6.],
                [  8.,  10.,  12.]])

In [29]: x * x

Out[29]: array([[  1.,   4.,   9.],
                [ 16.,  25.,  36.]])

In [30]: np.mean(x)

Out[30]: 3.5

In [31]: x.mean()

Out[31]: 3.5

In [32]: np.mean(x, axis = 0) #Promedio por columna

Out[32]: array([ 2.5,  3.5,  4.5])

In [33]: x.std()

Out[33]: 1.707825127659933

In [34]: np.sum(x, axis=1)

Out[34]: array([  6.,  15.])

In [35]: x.sum(axis=0)

Out[35]: array([ 5.,  7.,  9.])
```

Linear álgebra (numpy.linalg)

- Productos entre vectores y matrices
- Descomposiciones
- Cálculo de eigenvalues

Los arreglos de numpy funcionan muy bien para arreglos de n-dimensiones con valores numéricos, pero seamos realistas quedan muy atrás al momento de lidiar con datos heterogéneos.

Para almacenar datos de una fuente externa, como un libro de Excel o base de datos, necesitamos una estructura de datos que pueda contener diferentes tipos de datos.

También es deseable poder hacer referencia a filas y columnas de los datos usando etiquetas en lugar de índices numerados.

5.3 Meet Pandas

5.4 Pandas y DataFrames http://pandas.pydata.org/

pandas:

- Librería Open Source para análisis de datos
- Python
- Equivalente a data.frame y Dplyr de R-lang

pandas is an open source Python library for data analysis. Python has always been great for prepping and munging data, but it's never been great for analysis - you'd usually end up using R or loading it into a database and using SQL (or worse, Excel). pandas makes Python great for analysis.

- 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.

5.5 `import pandas as pd`

```
In [42]: import pandas as pd
```

```
In [41]: # Series
         s = pd.Series([7, 'Heisenberg', 3.14, -1789710578, 'Happy Eating!'])
         s
```

```
Out[41]: 0               7
         1      Heisenberg
         2            3.14
         3     -1789710578
         4   Happy Eating!
         dtype: object
```

```
In [44]: s = pd.Series([0.1, 1.2, 2.3, 3.4, 4.5])
         s

Out[44]: 0    0.1
         1    1.2
         2    2.3
         3    3.4
         4    4.5
         dtype: float64

In [45]: s = pd.Series([0.1, 1.2, 2.3, 3.4, 4.5],
                       index = ['a','b','c','d','e'])
         s

Out[45]: a    0.1
         b    1.2
         c    2.3
         d    3.4
         e    4.5
         dtype: float64

In [79]: pd.DataFrame({
             'municipio': ['mty', 'spgg', 'sta_c', 'garcia'],
             'temp_celcius_jueves': [35, 36, 37, 38],
             'temp_celcius_viernes': [36, 37, 39, 38]
         })

Out[79]:   municipio  temp_celcius_jueves  temp_celcius_viernes
         0       mty                   35                    36
         1      spgg                   36                    37
         2     sta_c                   37                    39
         3    garcia                   38                    38

In [43]: # DataFrame
         data = {'year': [2010, 2011, 2012, 2011, 2012, 2010, 2011, 2012],
                 'team': ['Bears', 'Bears', 'Bears', 'Packers', 'Packers', 'Lions', 'Lions', 'L
                 'wins': [11, 8, 10, 15, 11, 6, 10, 4],
                 'losses': [5, 8, 6, 1, 5, 10, 6, 12]}
         football = pd.DataFrame(data, columns=['year', 'team', 'wins', 'losses'])
         football

Out[43]:    year     team  wins  losses
         0  2010    Bears    11       5
         1  2011    Bears     8       8
         2  2012    Bears    10       6
         3  2011  Packers    15       1
         4  2012  Packers    11       5
         5  2010    Lions     6      10
         6  2011    Lions    10       6
         7  2012    Lions     4      12
```

```
In [86]: mi_df = pd.DataFrame({
             'municipio': ['mty', 'spgg', 'sta_c', 'garcia'],
             'temp_celcius_jueves': [35, 36, 37, 38],
             'temp_celcius_viernes': [36, 37, 39, 38]
         })


         mi_df[(mi_df['temp_celcius_jueves'] < 37) or
               (mi_df['temp_celcius_viernes'] < 40)]


         ---------------------------------------------------------------------------

         ValueError                                Traceback (most recent call last)

         <ipython-input-86-9d0fbf724fc9> in <module>()
           6
           7
         ----> 8 mi_df[(mi_df['temp_celcius_jueves'] < 37) or
           9       (mi_df['temp_celcius_viernes'] < 40)]


         /Users/israel/anaconda3/envs/viakable/lib/python3.5/site-packages/pandas/core/generic.p
         885         raise ValueError("The truth value of a {0} is ambiguous. "
         886                          "Use a.empty, a.bool(), a.item(), a.any() or a.all()."
         --> 887                          .format(self.__class__.__name__))
         888
         889     __bool__ = __nonzero__


         ValueError: The truth value of a Series is ambiguous. Use a.empty, a.bool(), a.item(),
```

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

```
In [46]: s[['a','c']]

Out[46]: a    0.1
         c    2.3
         dtype: float64

In [47]: s[s>2]

Out[47]: c    2.3
         d    3.4
         e    4.5
         dtype: float64

In [48]: data.tail()
```

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

AttributeError                            Traceback (most recent call last)

<ipython-input-48-8453368af19f> in <module>()
----> 1 data.tail()


AttributeError: 'dict' object has no attribute 'tail'
```

In [49]:
```
data = pd.read_csv('train.csv')
data.head()
```

Out[49]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [50]:
```
data.tail()
```

Out[50]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

```
         889    male  26.0      0      0      111369  30.00  C148        C
         890    male  32.0      0      0      370376   7.75   NaN        Q
```

```
In [87]: data.pop()
```

```
Out[87]:        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 [52]: age = data['Age']
         sum(age.isnull())
```

```
Out[52]: 177
```

```
In [53]: np.mean(age)
```

```
Out[53]: 29.69911764705882
```

```
In [54]: age = data['Age'].fillna(29.69)
```

```
In [55]: sum(age.isnull())
```

```
Out[55]: 0
```

```
In [56]: pclass = data['Pclass']
         pclass.unique()
```

```
Out[56]: array([3, 1, 2])
```

```
In [57]: data.Pclass.head()
```

```
Out[57]: 0    3
         1    1
         2    3
         3    1
         4    3
         Name: Pclass, dtype: int64
```

```
In [58]: data[['Age', 'Pclass']].head()
```

```
Out[58]:     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 [59]: del data['Ticket']
         data.head()
```

```
Out[59]:    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
```

5.5.1 Datos

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

5.5.2 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

5.5.3 Datos ordenados (Tidy data)

Notebook 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

5.5.4 £Dónde consigo los datos?

5.6 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

5.7 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/

5.8 Dataset para hoy - train.csv

5.8.1 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 hoy usaremos el dataset `train.csv` y completarás el análisis de qué tipo de personas eran propensos a sobrevivir . En un futuro, te pedimos aplicar las herramientas de aprendizaje automático para predecir que los pasajeros sobrevivieron a la tragedia.

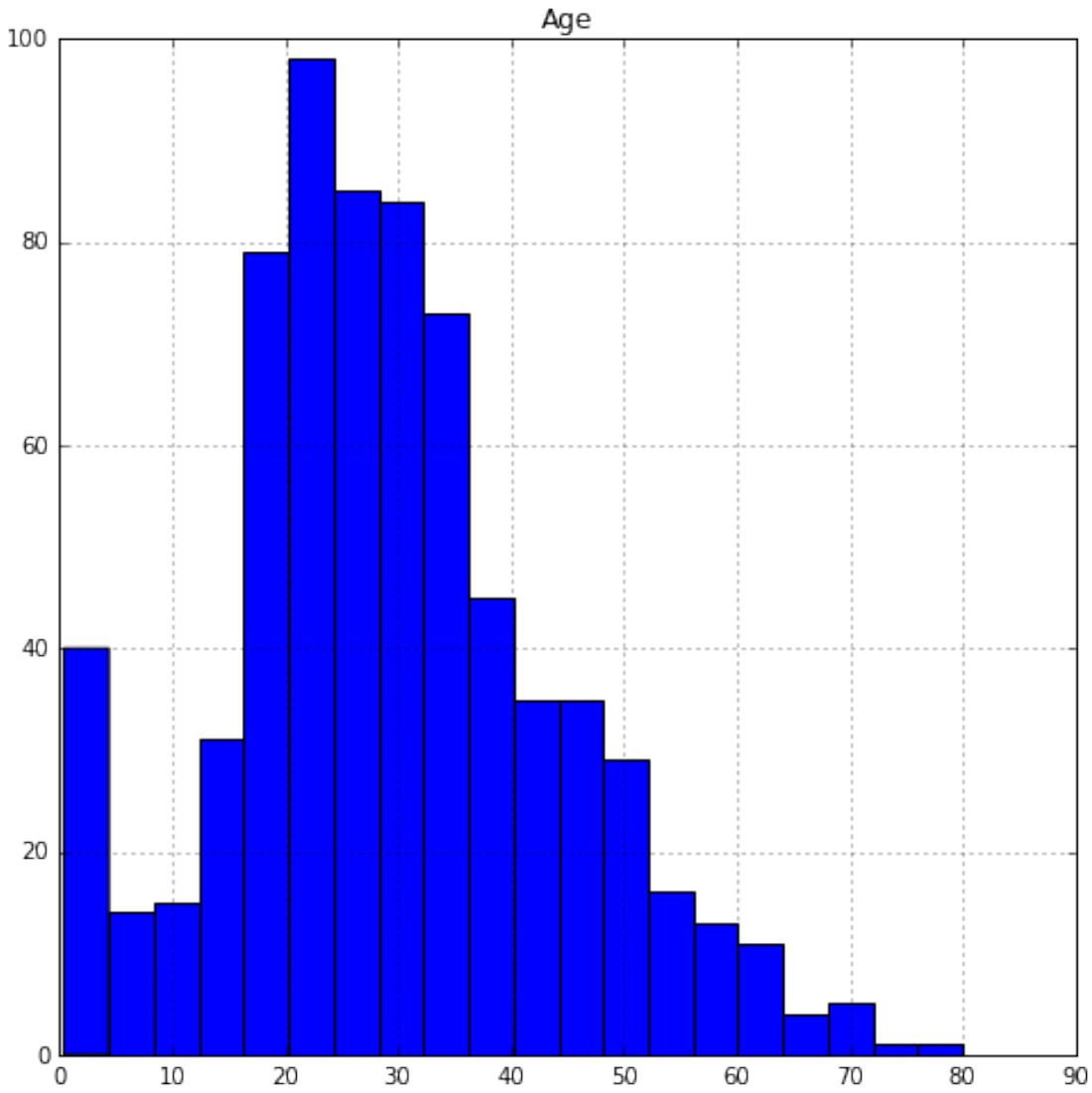
5.8.2 Matplotlib

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 [62]: import matplotlib
         %matplotlib inline
```

Histogramas

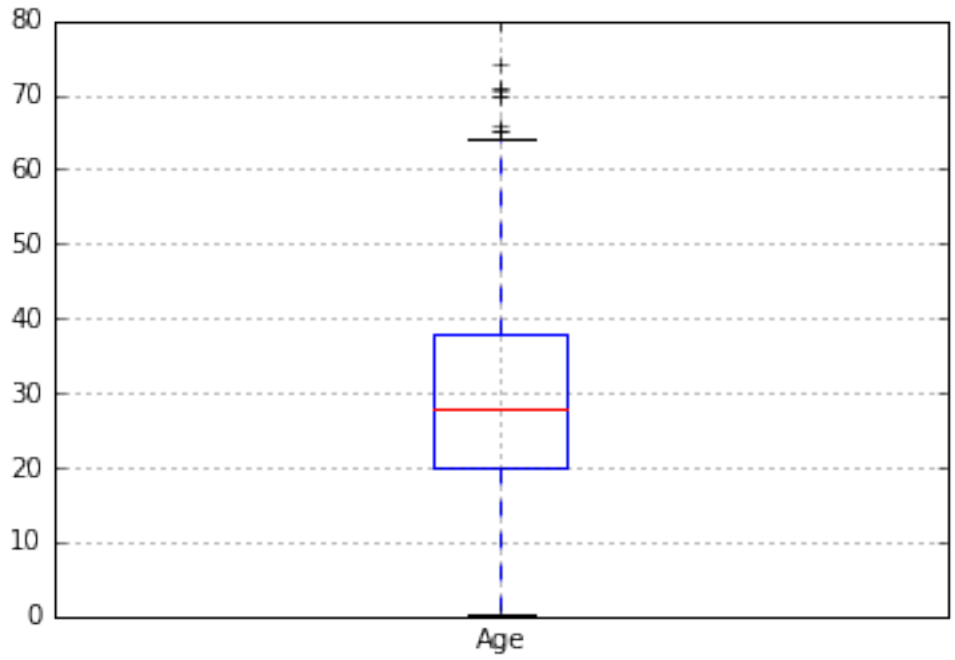
```
In [63]: data.hist(bins = 20, column="Age", figsize=(8,8), color="blue")

Out[63]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x10cfe9be0>]], dtype=object)
```



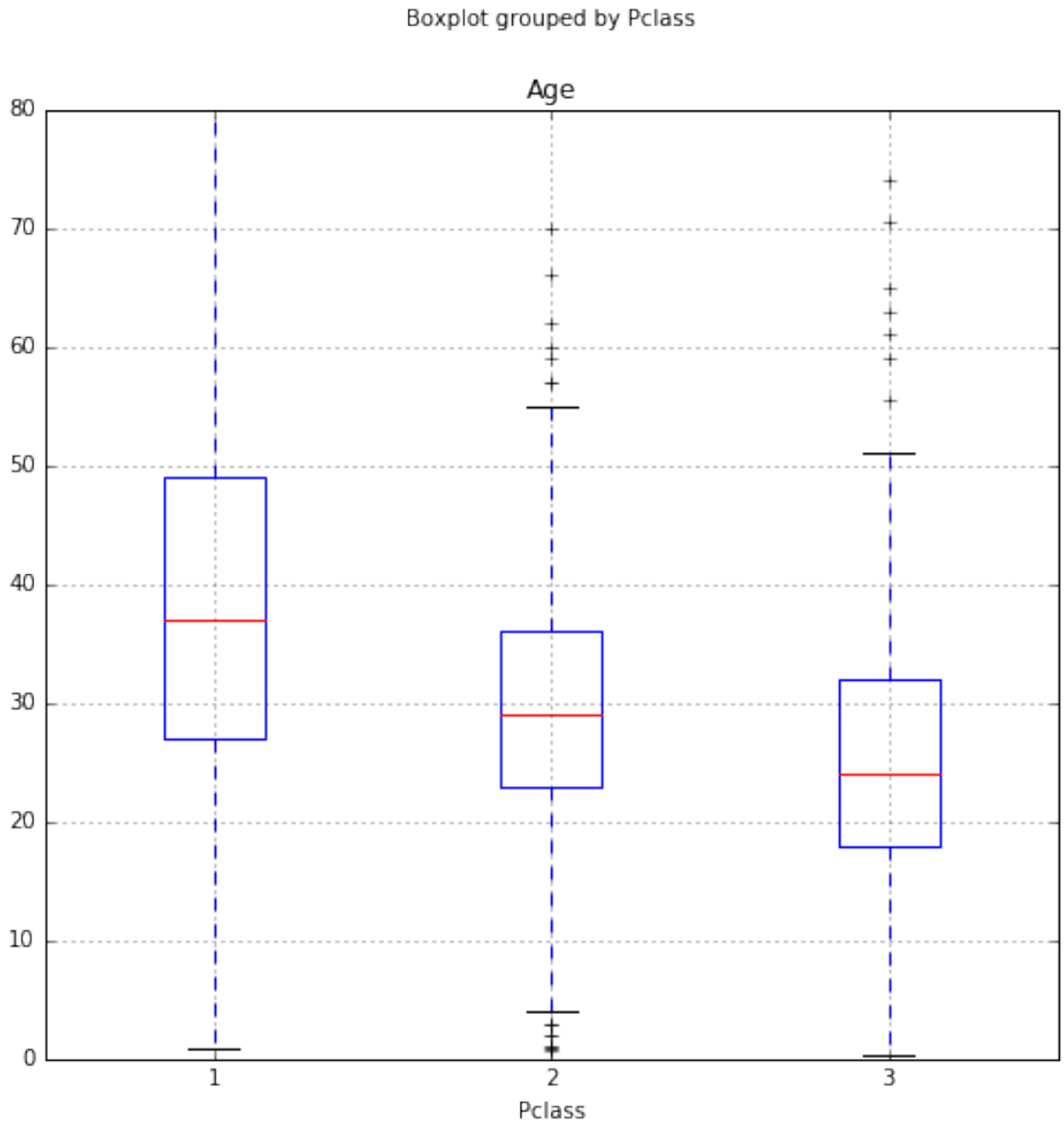
Boxplot

```
In [64]: data.boxplot(column="Age", return_type='axes')
```

```
Out[64]: <matplotlib.axes._subplots.AxesSubplot at 0x10f340198>
```

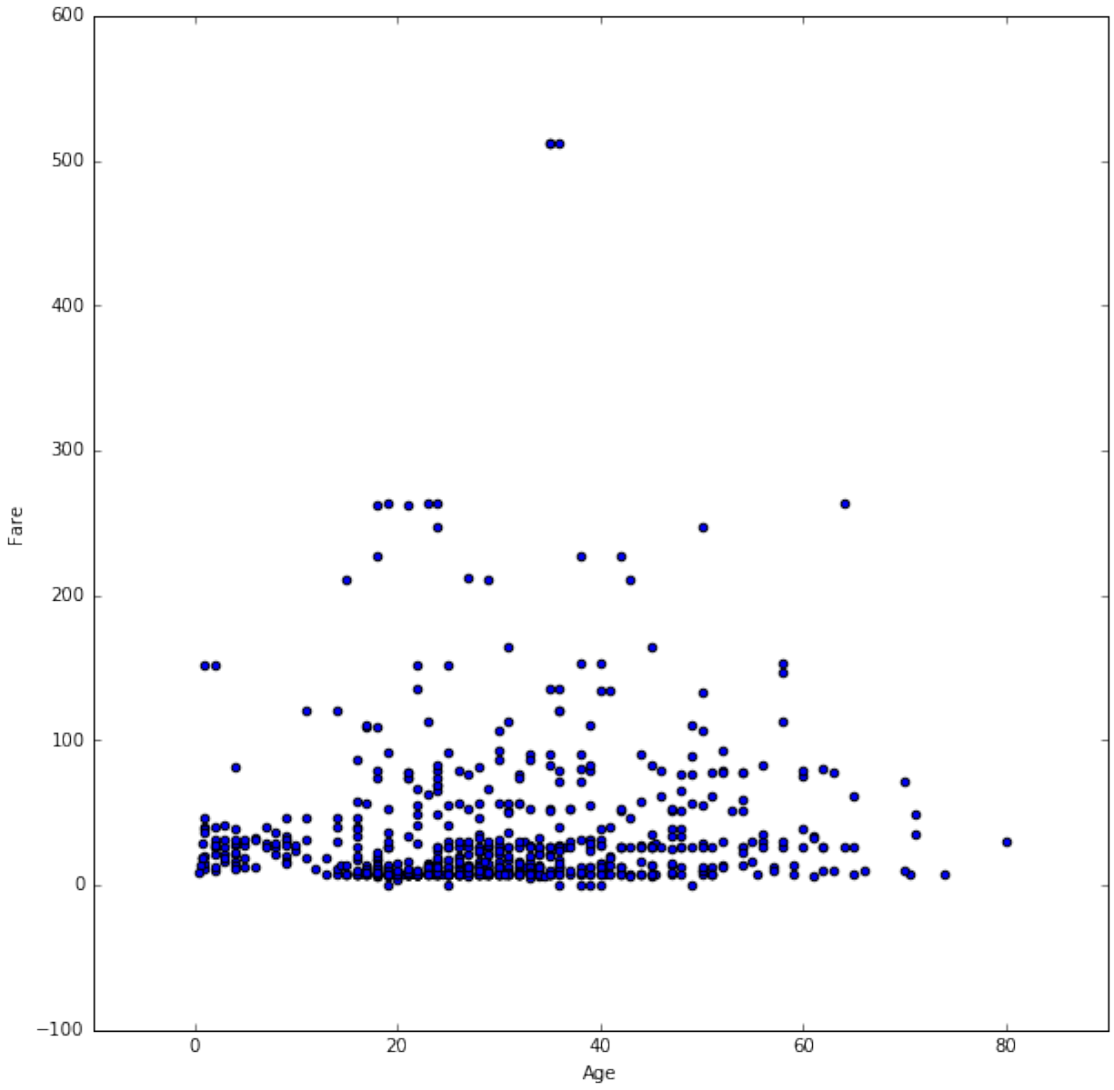



```
In [65]: data.boxplot(column="Age", by="Pclass", figsize= (8,8))

Out[65]: <matplotlib.axes._subplots.AxesSubplot at 0x10fa2aba8>
```




```
In [66]: ### Scatterplots

         data.plot(kind="scatter",
                   x="Age",
                   y="Fare",
                   figsize=(10,10))
```

```
Out[66]: <matplotlib.axes._subplots.AxesSubplot at 0x10fa3d780>
```



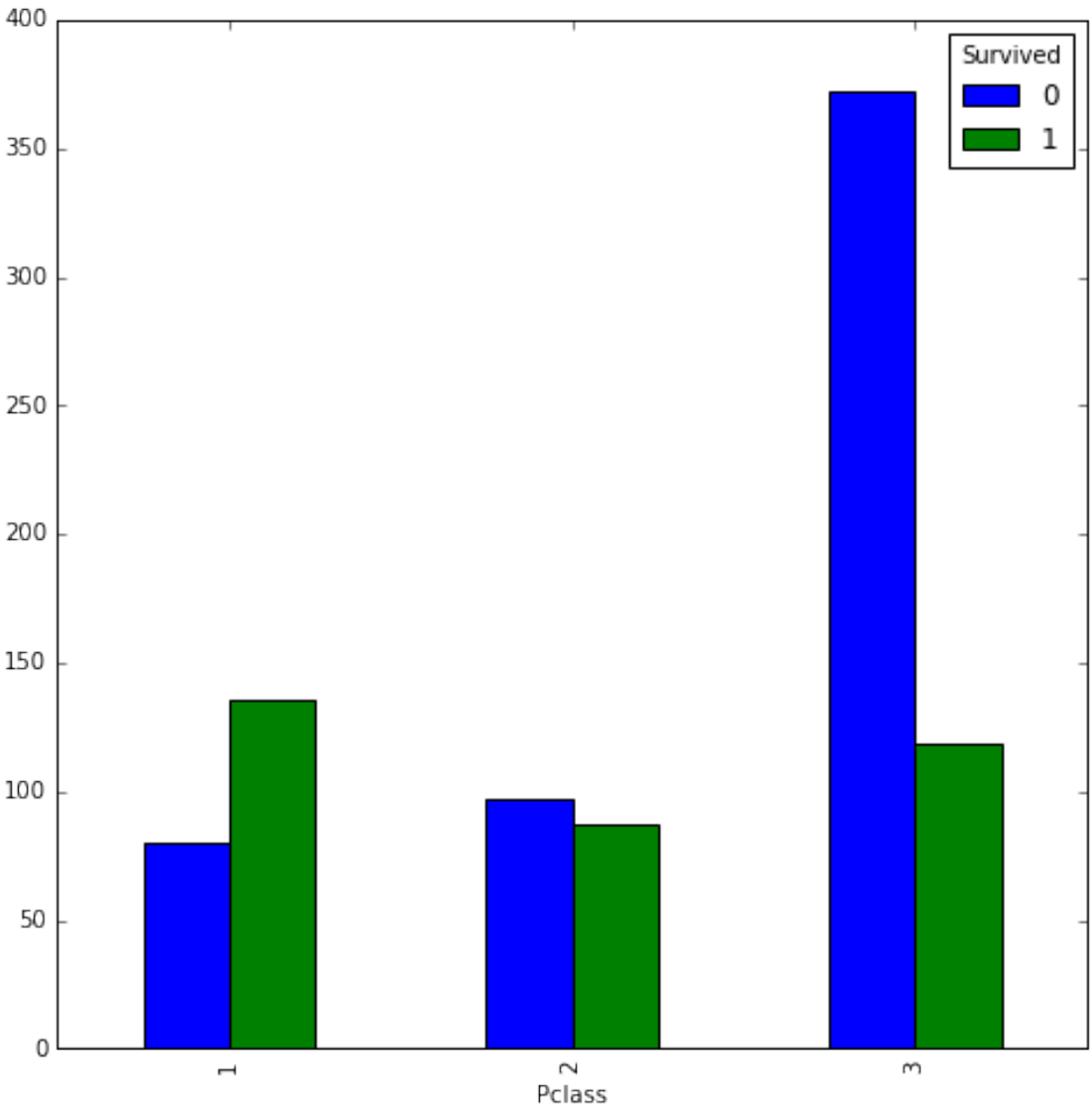
Barras

```
In [67]: tabla = pd.crosstab(index=data["Pclass"], columns=data["Survived"])
         tabla
```

```
Out[67]: Survived    0    1
         Pclass
         1          80  136
         2          97   87
         3         372  119
```

```
In [68]: tabla.plot(kind="bar", figsize=(8,8))
```

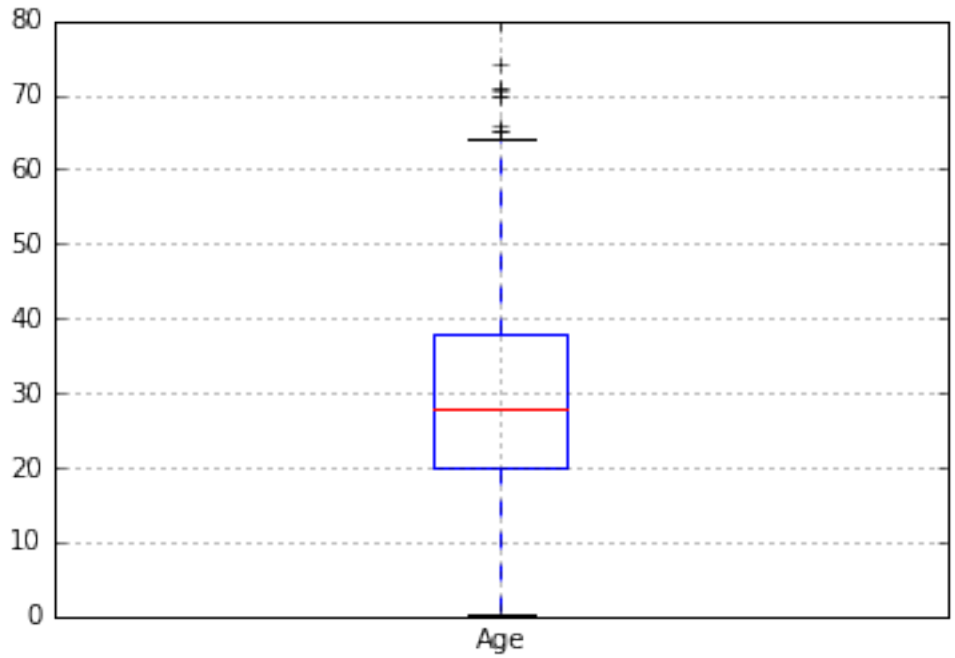
```
Out[68]: <matplotlib.axes._subplots.AxesSubplot at 0x10fba3518>
```



EJERCICIO Usando el dataframe anterior, crear un gráfico de barra donde se vea la cantidad de sobrevivientes en proporción al total de pasajeros por clase.

RESPUESTA

```
In [88]: data.boxplot(column='Age', return_type='axes')

Out[88]: <matplotlib.axes._subplots.AxesSubplot at 0x110579940>
```




```
In [71]: tabla = pd.crosstab(index=data["Pclass"],
                             columns=data["Survived"],
                             margins=True) #Agregando margins=True nos da los totales
```

```
tabla
```

Out[71]:

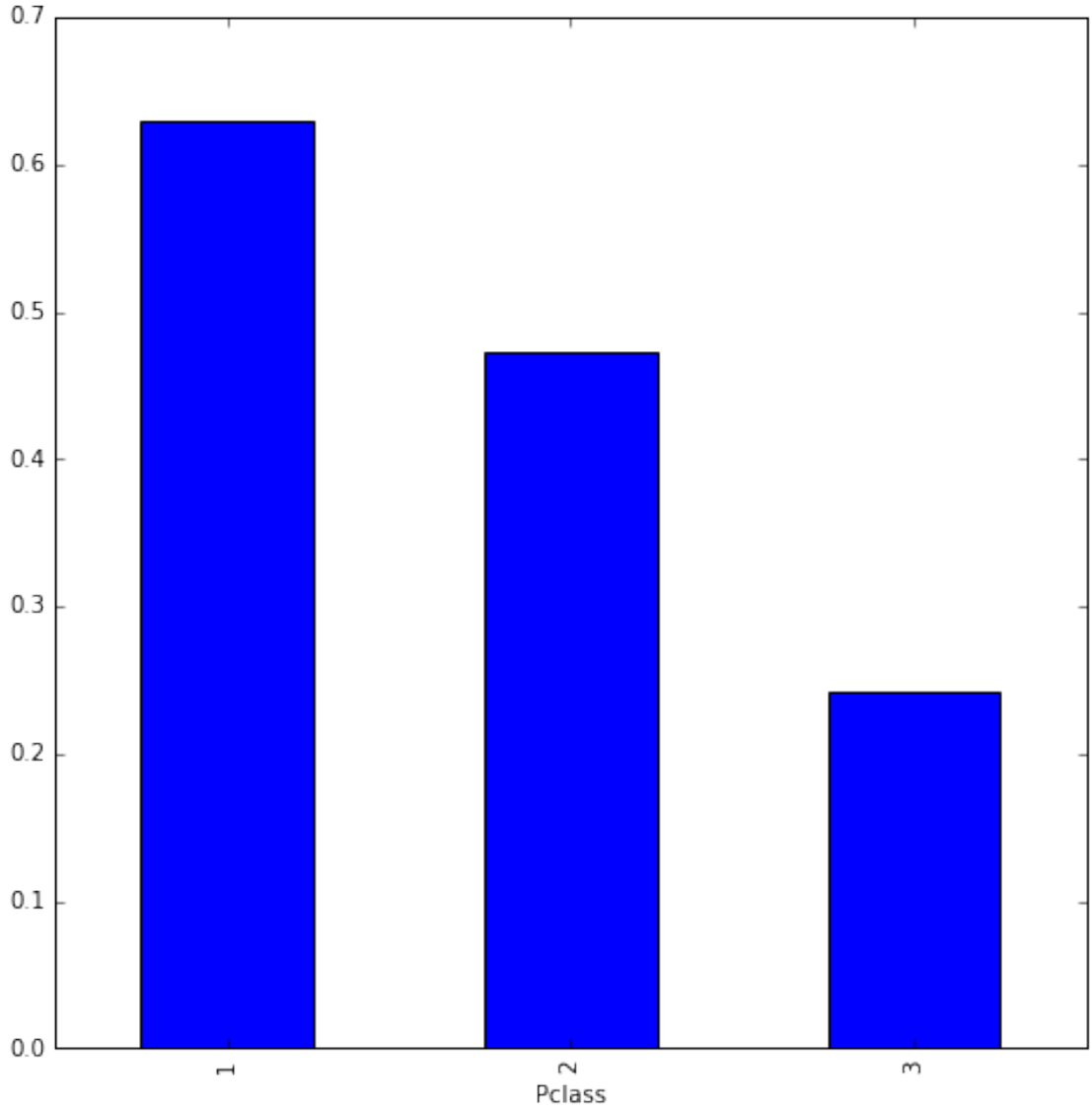
Survived	0	1	All
Pclass			
1	80	136	216
2	97	87	184
3	372	119	491
All	549	342	891

```
In [72]: #Obtenemos la proporción dividiento entre el total por filas
         tabla = tabla.div(tabla["All"], axis=0)
         tabla
```

Out[72]:

Survived	0	1	All
Pclass			
1	0.370370	0.629630	1.0
2	0.527174	0.472826	1.0
3	0.757637	0.242363	1.0
All	0.616162	0.383838	1.0

```
In [73]: tabla = tabla.drop(tabla.index[3])[1]

In [74]: tabla.plot(kind="bar", figsize=(8,8))

Out[74]: <matplotlib.axes._subplots.AxesSubplot at 0x10fd7c6a0>
```



5.9 Web Scrapping con PANDAS

```
In [75]: # Asignar el resultado del metodo read_html a densidad_paises
         # Parametro header=0 para establecer el header de la tabla
         # Consultar: http://pandas.pydata.org/pandas-docs/stable/gotchas.html#html-gotchas
         densidad_paises = pd.read_html('https://simple.wikipedia.org/wiki/List_of_countries_by
```

```
# Observar los primeros 10 registros
densidad_paises[0][:10]
```

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

ImportError                               Traceback (most recent call last)

<ipython-input-75-470c114ca553> in <module>()
  2 # Parametro header=0 para establecer el header de la tabla
  3 # Consultar: http://pandas.pydata.org/pandas-docs/stable/gotchas.html#html-gotchas
----> 4 densidad_paises = pd.read_html('https://simple.wikipedia.org/wiki/List_of_countries
  5
  6 # Observar los primeros 10 registros

/Users/israel/anaconda3/envs/viakable/lib/python3.5/site-packages/pandas/io/html.py in
868     _validate_header_arg(header)
869     return _parse(flavor, io, match, header, index_col, skiprows,
--> 870                   parse_dates, tupleize_cols, thousands, attrs, encoding)

/Users/israel/anaconda3/envs/viakable/lib/python3.5/site-packages/pandas/io/html.py in
720     retained = None
721     for flav in flavor:
--> 722         parser = _parser_dispatch(flav)
723         p = parser(io, compiled_match, attrs, encoding)
724

/Users/israel/anaconda3/envs/viakable/lib/python3.5/site-packages/pandas/io/html.py in
679     else:
680         if not _HAS_LXML:
--> 681             raise ImportError("lxml not found, please install it")
682     return _valid_parsers[flavor]
683

ImportError: lxml not found, please install it
```

In [76]:
```
# Que tipo de objeto tenemos?
type(densidad_paises)
```

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

NameError                                 Traceback (most recent call last)
```

```
<ipython-input-76-9e63175da141> in <module>()
  1 # Que tipo de objeto tenemos?
----> 2 type(densidad_paises)

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

In [ ]:
```
# Como lo convertimos en un dataframe?

densidad_paises_dataframe = pd.DataFrame(densidad_paises[0])
```

```
# Que tipo de objeto tenemos?
type(densidad_paises_dataframe)
```

```
densidad_paises_dataframe.head()
```

```
densidad_paises_dataframe.keys()
```

```
densidad_paises_clean = densidad_paises_dataframe.copy()
```

```
# http://pandas.pydata.org/pandas-docs/stable/dsintro.html#column-selection-addition-d
densidad_paises_clean.pop('Unnamed: 1')
```

```
# Verificar
densidad_paises_clean
```

```
densidad_paises_clean.pop('Area (mi2)')
```

```
densidad_paises_clean.pop('Density (/mi2)')
```

```
# for cycle
column_list = []
for element in column_list:
    dataframe.pop(element)
```

```
densidad_paises_clean.head()
```

```
del densidad_paises_clean['Notes']
```

```
densidad_paises_clean.head()
```

```
densidad_paises_clean.rename(columns={'Population': 'Pop'}, inplace=True)
```

```
densidad_paises_clean.rename(columns={'Area (km2)': 'Area'}, inplace=True)
```

```
densidad_paises_clean.rename(columns={'Density (/km2)': 'Density'}, inplace=True)
```

5.9.1 Exportar ?

5.9.2 APIs ?

Siguiente NB: Estadistica_24_Jun