Potencialidades de uso del lenguaje de programación de código abierto Python en el Sector Industrial Gubernamental

Sergio Rojas
Profesor Titular
Departamento de Física
Universidad Simón Bolívar, Venezuela
srojas@usb.ve

Jornastec 2017
Universidad Bolivariana de Venezuela
10 de noviembre de 2017

Resumen

El lenguaje de programación de código abierto Python [1] ofrece un sin número de potencialidades de aplicación como herramienta de cómputo en el sector industrial, que van desde la ejecución de modelos de simulación de los procesos industriales hasta la consolidación de un sofisticado sistema de gestión de operaciones, incluyendo información geográfica.

En el contexto de las necesidades latinoamericanas y caribeñas, particularmente la Venezolana, Python es una alternativa computacional de bajo costo para la gestión y seguimiento de políticas públicas tal como el seguimiento a los recursos destinados a la Salud y la Educación; el análisis y la prevención de delitos; creación y fortalecimiento de un sistema práctico y eficiente de denuncias, entre otras.

Sobre Python

Python (https://www.python.org/) es un lenguaje de programación multi-paradigma (http://en.wikipedia.org/wiki/Programming_paradigm#Multi-paradigm) que por satisfacer las exigencias de un lenguaje de programación moderno (como programación en funciones y orientada a objeto) **ha ganado mucha popularidad** en los últimos años en el medio de la computación científica, gracias a que se han incorporado al mismo módulos que facilitan la tarea de cómputo científico tales como:

- NumPy (http://www.numpy.org/) y SciPy (http://www.scipy.org/): que incorporan bibliotecas de cálculo en prácticamente todas las áreas que abarca el cómputo numérico.
- Matplotlib (http://matplotlib.org/): para satisfacer las necesidades de visualización.
- SymPy (http://www.sympy.org/en/index.html): para cubrir las necesidades de ejecutar cómputo algebraíco o matemática simbólica.
- mucho más (https://docs.python.org/3/py-modindex.html).

Python en el sector Industrial Gubernamental

EDUCATIVO: Industria gubernamental que forma el talento humano para avanzar el desarrollo de los países

ESTUDIO:Preparación de los estudiantes de Física como profesionales del siglo 21 (https://doi.org/10.1063/PT.3.3763)

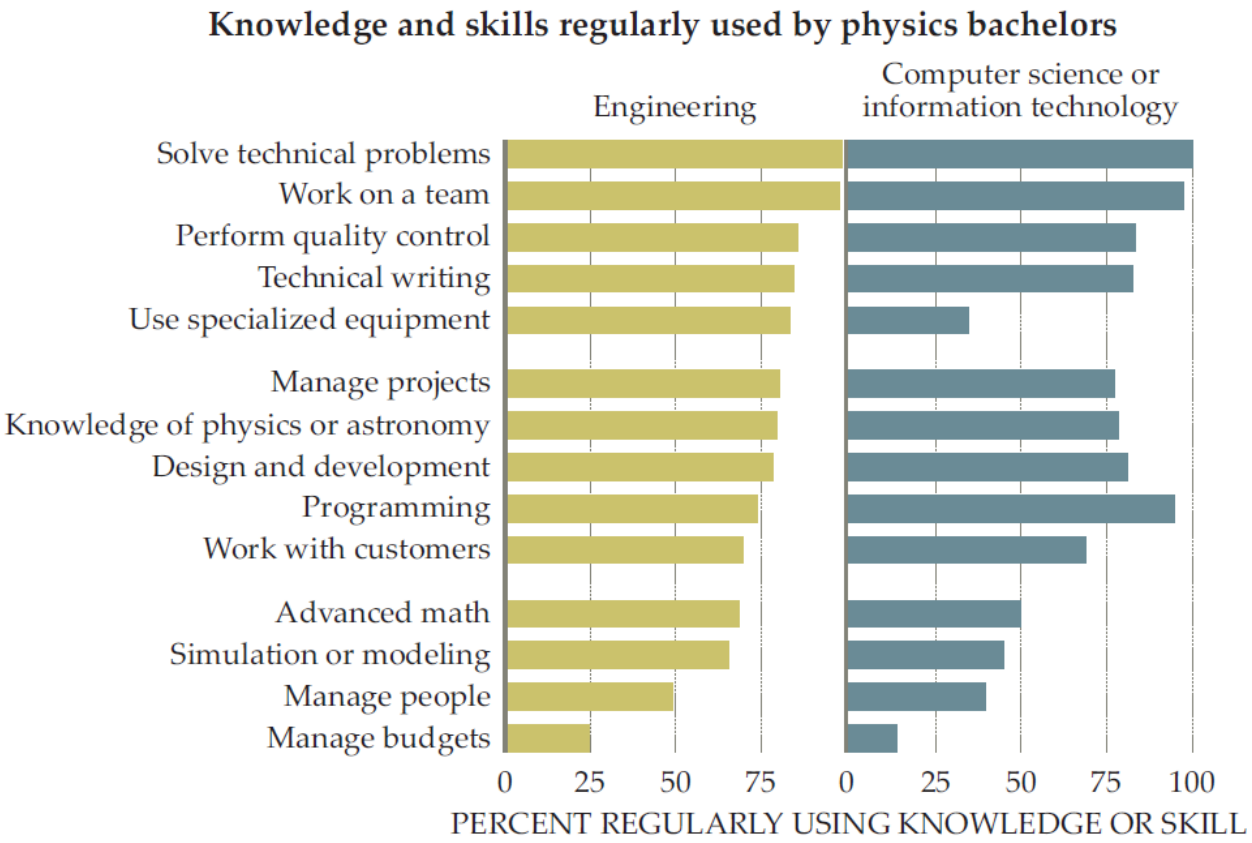
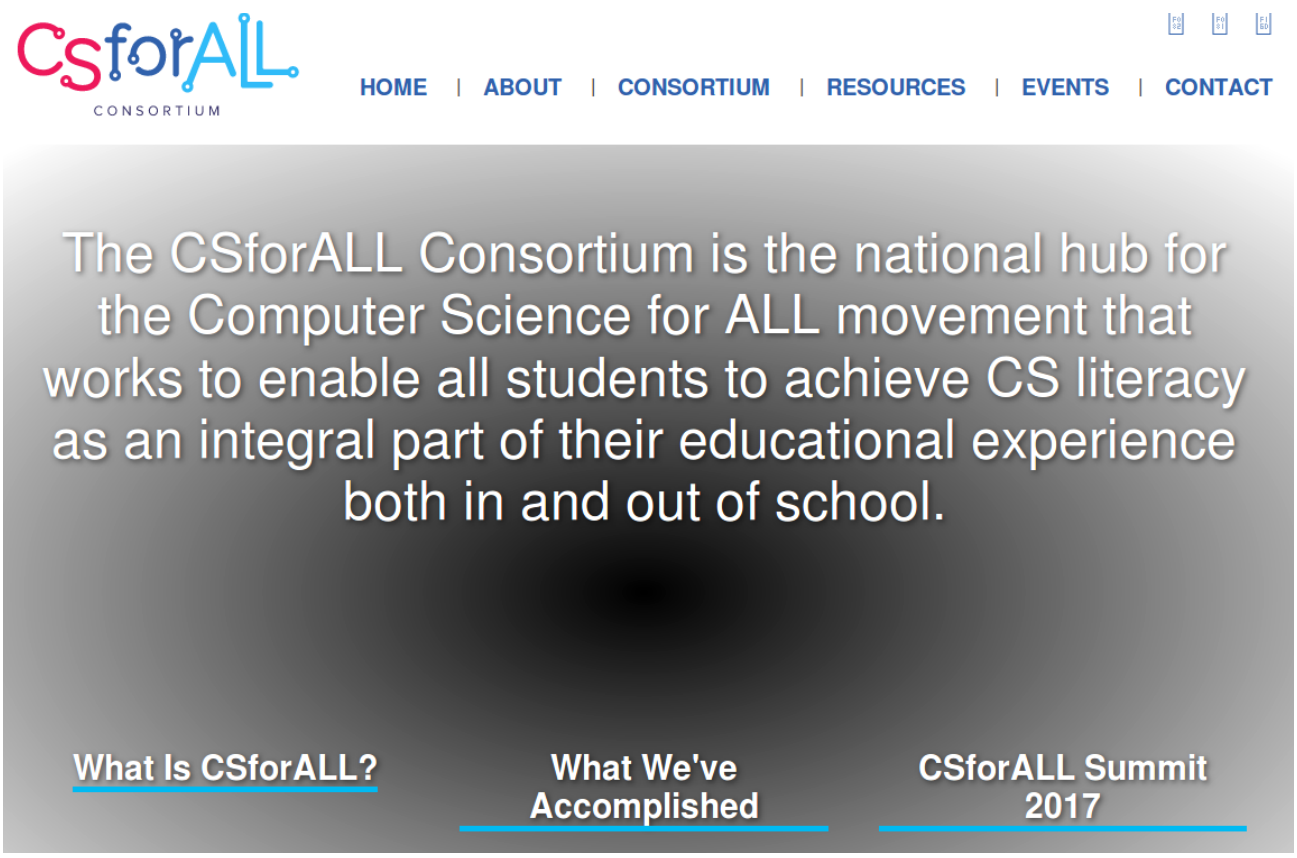
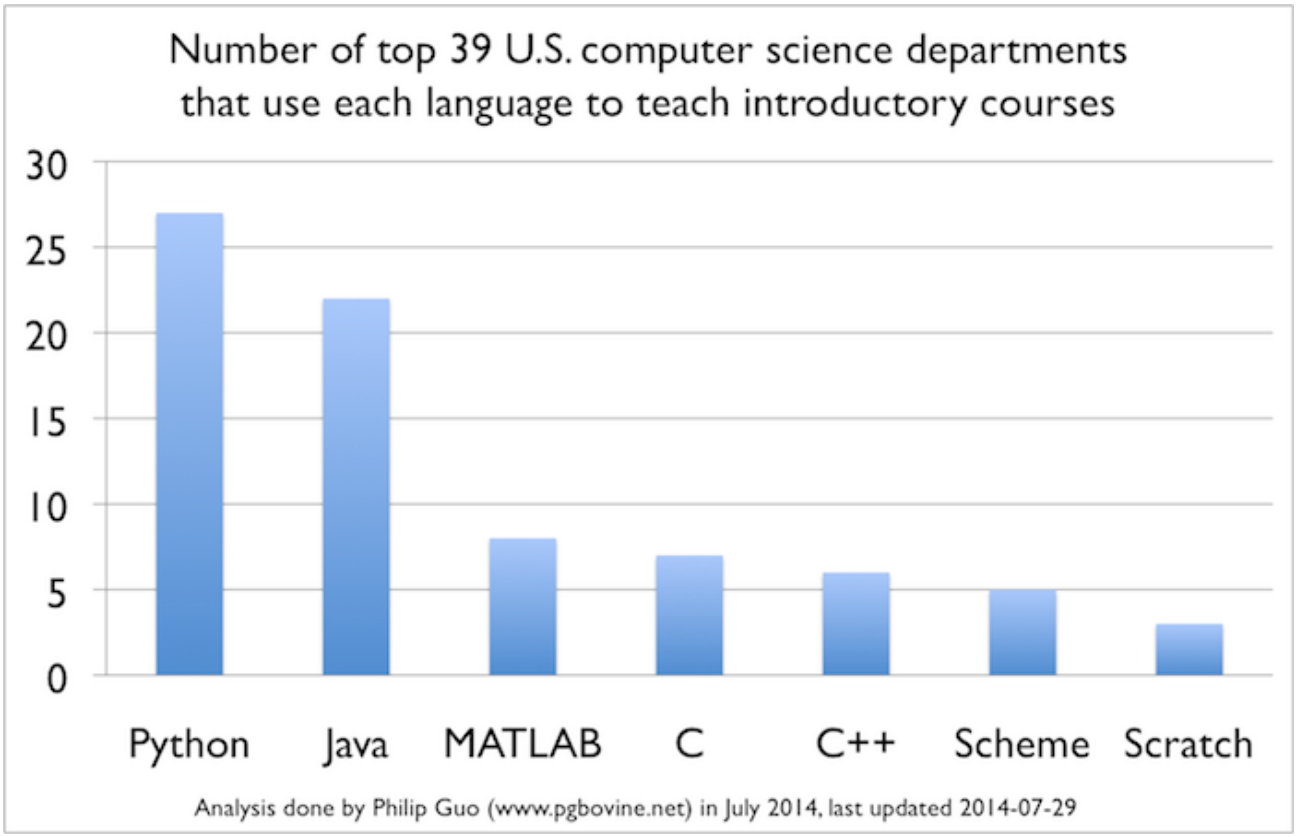


FIGURE 2. WHEN POLLED ABOUT WHAT KINDS OF KNOWLEDGE AND SKILLS they rely on daily, weekly, or monthly, physics graduates from 2013 and 2014 now working at private-sector jobs in engineering or computer science cited the broad range listed here. Graduates in both fields ranked several technical and professional skills as more useful—or more precisely, used more regularly—than a knowledge of physics. (Adapted from ref. 4.)

Movimiento:Ciencias de la Computación para todos (http://www.csforall.org/)



Estudio (2014)):Python califica de primero como lenguaje en el proceso enseñanza-aprendizaje de programación en Universidades Topes de Estados Unidos (https://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-u-s-universities)



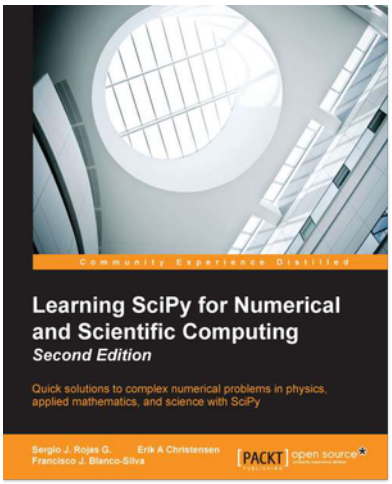
Algunas exigencias computacionales que Python satisface

- Interacción con otros Languajes de Programación: http://nbviewer.ipython.org/github/rojassergio/Learning-Scipy/blob/master/Chapter8/CHAP_08.ipynb (http://nbviewer.ipython.org/github/rojassergio/Learning-Scipy/blob/master/Chapter8/CHAP_08.ipynb)

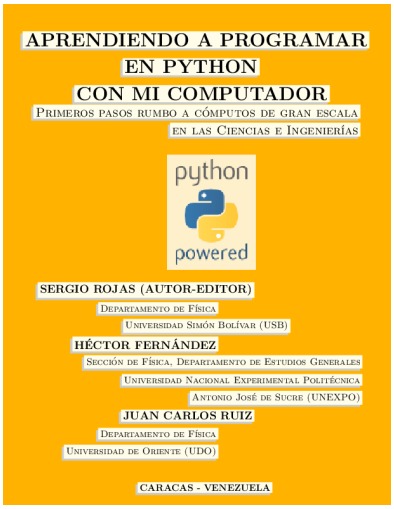
- Computación en Paralelo: http://ipyparallel.readthedocs.io/en/latest/ (http://ipyparallel.readthedocs.io/en/latest/)

- Computación con procesadores de tarjetas gráficas (GPU)): https://andreask.cs.illinois.edu/PyCuda (https://andreask.cs.illinois.edu/PyCuda)

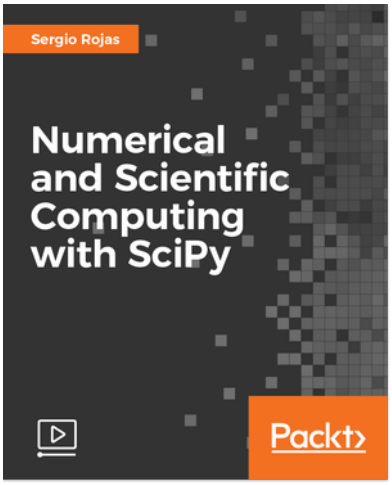
NUESTRA CONTRIBUCIÓN



(https://www.packtpub.com/big-data-and-business-intelligence/learning-scipy-numerical-and-scientific-computing-second-edition)



(https://www.researchgate.net/publication/301293668)



(https://www.packtpub.com/big-data-and-business-intelligence/numerical-and-scientific-computing-scipy-video)

POLÍTICAS PÚBLICAS: Industria gubernamental que apunta al avance eficiente del desarrollo de los países

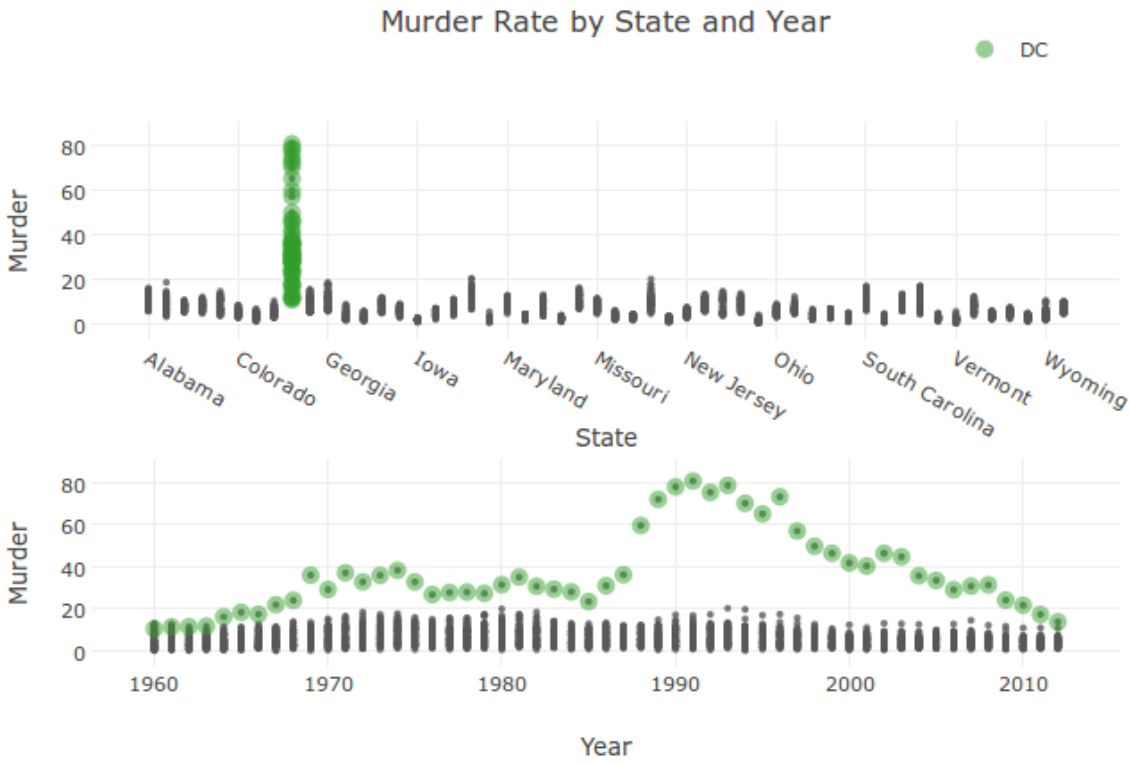
Propuesta a la Constituyente

[https://www.aporrea.org/contraloria/a252947.html (https://www.aporrea.org/contraloria/a252947.html)]:

Algún artículo de la NUEVA Constitución de la República Bolivariana de Venezuela debe establecer:

> ***Sancionar*** *con* ***destitución e inhabilitación*** *\textbf{de por vida para ejercer cargo público} a todo aquel que, ejerciendo cargo público en empresas o instituciones del Estado, por indolencia, saboteo o ejecutando prácticas burocráticas ineficientes \textbf{desatienda y deje de solucionar problemáticas sociales} de su competencia, que de una u otra forma cuentan con protección de rango constitucional.*

Estudio: Crímenes por estado según el FBI en Estados Unidos (https://github.com/ChrisBeaumont/crime)



Python en la industria petrolera

- Biblioteca para Modelaje e Inversión Geofísica:
  http://www.pygimli.org/ (http://www.pygimli.org/)
- Simulación y Estimación de Parámetros en Geofísica:
  https://pypi.python.org/pypi/SimPEG (https://pypi.python.org/pypi/SimPEG)
- Simulaciones de Fluidos en el Subsuelo y Transferencia de Calor:
  https://github.com/acroucher/PyTOUGH (https://github.com/acroucher/PyTOUGH)
- Modelaje de Flujo de Aguas Subterráneas:
  https://github.com/modflowpy/flopy (https://github.com/modflowpy/flopy)
- Simulador de Placas Tectónicas:
  https://github.com/Mindwerks/plate-tectonics (https://github.com/Mindwerks/plate-tectonics)
- Simulación de Yacimientos:
  https://pypi.python.org/pypi/PRST/0.0.3 (https://pypi.python.org/pypi/PRST/0.0.3)

Python en la industria espacial

- Imágenes satelitales:
  https://www.machinalis.com/blog/python-for-geospatial-data-processing/#disqus_thread (https://www.machinalis.com/blog/python-for-geospatial-data-processing/#disqus_thread)
  http://code.activestate.com/recipes/496966-download-satellite-images-from-nasas-site/ (http://code.activestate.com/recipes/496966-download-satellite-images-from-nasas-site/)

- Sistemas de Información Geográfico (GIS)):
  http://desktop.arcgis.com/en/analytics/python/ (http://desktop.arcgis.com/en/analytics/python/)

Regresión Logística

https://en.wikipedia.org/wiki/Logistic_regression (https://en.wikipedia.org/wiki/Logistic_regression)

$$\mathbf{h}_\theta(x) = S(\theta^{\mathbf{T}}\mathbf{X}) = \frac{1}{1 + e^{-\theta^{\mathbf{T}}\mathbf{x}}}$$

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{1}{2} + \frac{1}{2}tanh(\frac{x}{2})$$

$S(x)$ se conoce como la **función logística (sigmoid)** y es la **función de distribución acumulada** (cumulative distribution function) de la distribución logística.

El resultado (outcome) toma los valores:

$$y = \begin{cases} 1 & \text{if } \mathbf{h}_\theta(x) \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Como $S(x) \geq 0.5$ if $x \geq 0$, entonces se puede predecir $y = 1$ si $\theta^{\mathbf{T}}\mathbf{X} \geq 0$ y se puede predecir $y = 0$ si $\theta^{\mathbf{T}}\mathbf{X} < 0$ (la condición $\theta^{\mathbf{T}}\mathbf{X} = 0$ genera lo que se conoce como **el contorno o frontera de decisión**.

Determinando θ

Minimizar la función de medida (convexa):

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} j(h_\theta(x^{(i)}), y^{(i)})$$

$$j(h_\theta(x), y) = \begin{cases} -\ln\left[\mathbf{h}_\theta(x)\right] & \text{if } y = 1 \\ -\ln\left[1 - \mathbf{h}_\theta(x)\right] & \text{if } y = 0 \end{cases}$$

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left(y(i)\ln\left[h_\theta(x^{(i)})\right] + \left(1 - y^{(i)}\right)\ln\left[1 - h_\theta(x^{(i)})\right]\right)$$

$$\frac{\partial J(\theta)}{\partial \theta_k} = \frac{1}{m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)}) - y^{(i)})\right]x_k^{(i)}$$

El Programa

El sigmoid $S(x)$

In [2]:
```
def sigmoid(z):
    import scipy.special
    # This function compute sigmoid functoon
    # using SciPy expit(z) function
    # Sergio Rojas

    s = scipy.special.expit(z)

    return s
```

$J(\theta)$ y su gradiente $\nabla_\theta J(\theta)$

In [3]:

```
def JandGrad(theta, X, y):
    import numpy as np
    #   This function computes the cost J(theta)
    #   for logistic regression and the its gradient
    #   respectto the parameters theta.
    #
    # y is a one column vector
    # X is a matrix first column filled with ones (see first exercise set)
    # theta is a column vector
    # Sergio Rojas

    m = len(y) # number of training examples
    grad = np.zeros(np.size(theta))
    #  Computing J
    # ---------
    u = np.dot(X,theta)  # One column vector
    h = sigmoid(u)       # One column vector

    # The sum on J(theta) is implemented using scalar product
    # of the vectors invloved. Each one is one column vector
    #----
    yT = np.transpose(y)
    thesum = - np.dot(yT, np.log(h)) - np.dot((1.0 - yT),np.log(1.0-h));

    J = (1.0/m) * thesum;

    #  Computing grad of J(theta)
    # ---------
    grad = np.dot(np.transpose(h-y), X);

    grad = np.transpose(grad) ;    # transpose the grad
    grad = (1.0/m) * grad;

    return J, grad
```

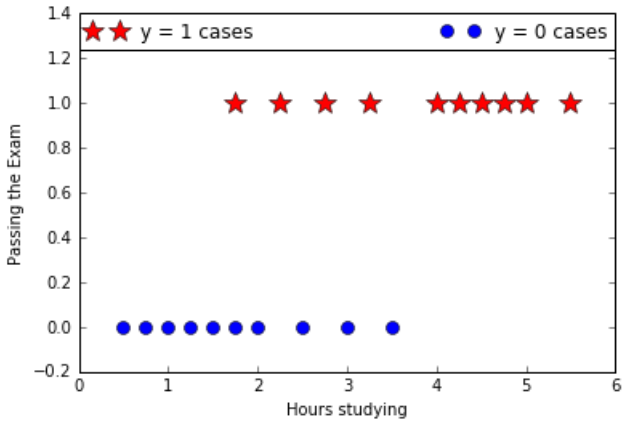
Ejemplo: probabilidad de pasar un examen Vs horas de estudio

https://en.wikipedia.org/wiki/Logistic_regression (https://en.wikipedia.org/wiki/Logistic_regression)

In [4]:
```
def plot_1D(X, y):
    # X is one dimensional
    # y is one dimensional
    #----
    import numpy as np
    import matplotlib.pyplot as plt

    fig = plt.figure()
    ax  = fig.add_subplot(1, 1, 1)

    indxpos = np.where(y==1); indxneg = np.where(y == 0);

    ax.plot(X[indxpos].flatten(), y[indxpos].flatten(),
                                'r*', label="y = 1 cases ", markersize=14)
    ax.plot(X[indxneg].flatten(), y[indxneg].flatten(),
                                'bo', label="y = 0 cases",markersize=8)
    ax.set_ylim(-.2,1.4)
    return fig, ax
```

In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.optimize as sciopt
%matplotlib inline

# Load Data
data = np.loadtxt('./Data/ex1_1D_data.dat', delimiter=',', skiprows=1)
X = data[:,0]
y = data[:,1]

fig, ax1 = plot_1D(X, y)
#
ax1.set_xlabel('Hours studying')
ax1.set_ylabel('Passing the Exam')
ax1.legend(bbox_to_anchor=(0., .9, 1., .9), loc=3,
           ncol=2, mode="expand", borderaxespad=0.)
plt.show()
```

```
/home/srojas/myProg/Anaconda3400/lib/python3.5/site-packages/matplotlib/font_
manager.py:273: UserWarning: Matplotlib is building the font cache using fc-l
ist. This may take a moment.
  warnings.warn('Matplotlib is building the font cache using fc-list. This ma
y take a moment.')
/home/srojas/myProg/Anaconda3400/lib/python3.5/site-packages/matplotlib/font_
manager.py:273: UserWarning: Matplotlib is building the font cache using fc-l
ist. This may take a moment.
  warnings.warn('Matplotlib is building the font cache using fc-list. This ma
y take a moment.')
```



In [6]:
```
if X.ndim==1: X=X[:,None] # Make a 1-column array
m, n = np.shape(X) #

# Add intercept term to X
X = np.column_stack((np.ones(m), X))

# Initialize theta parameters
initial_theta = np.zeros(n + 1)

J, gradJ = JandGrad(initial_theta, X, y)
print ('    J at initial theta: ', J)
print ('GradJ at initial theta:', gradJ)
```

```
    J at initial theta:  0.69314718056
GradJ at initial theta: [ 0.      -0.50625]
```

In [7]:
```
res = sciopt.minimize(JandGrad, initial_theta, args=(X,y), \
                      method='BFGS', jac=True, options={'maxiter':400})
theta = res.x
J = res.fun

print ('Optimal theta:', theta)
print ('Cost J at optimal theta: ', J)
print('\n')
print('Result for     Hours = {0:10.4f}'.format(theta[1]) )
print('Result for Intercept = {0:10.4f}'.format(theta[0]) )
```

```
Optimal theta: [-4.07771408  1.50464686]
Cost J at optimal theta:  0.40149392321797317


Result for     Hours =     1.5046
Result for Intercept =    -4.0777
```

La data y su probabilidad

In [8]:
```
u = np.dot(X,theta)
h = sigmoid(u)

fig, ax1 = plot_1D(data[:,0], y)

ax1.plot(data[:,0],h)

ax1.set_xlabel('Hours studying')
ax1.set_ylabel('Passing the Exam')
ax1.set_title('Probability of passing exam Vs Hours of studying')
ax1.legend(bbox_to_anchor=(0., .9, 1., .9), loc=3,
           ncol=2, mode="expand", borderaxespad=0.)
```

Predicción sobre los datos de entrenamiento

In [9]:
```
p = np.zeros(m)
temp = np.dot(X, theta);
indx=np.where(temp>=0)
p[indx] = 1;
print ('Train Accuracy: {0:4.2f}% '.format((np.mean(p == y) * 100)) )
```

```
Train Accuracy: 80.00%
```

Haciendo predicciones:

Referencias

- Hosmer, D. W. Jr., Lemeshow, S., and Sturdivant, R. X. (2013) Applied Logistic Regression, 3rd. edition, Wiley.

- Pagliuca, G (2016) Python in oil and gas industry https://www.youtube.com/watch?v=COpKs4w8y-Y (https://www.youtube.com/watch?v=COpKs4w8y-Y)

- Rojs, S., Christensen, A. E., and Blanco-Silva, F. J. (2015). Learning SciPy for Numerical and Scientific Computing - Second Edition https://www.packtpub.com/big-data-and-business-intelligence/learning-scipy-numerical-and-scientific-computing-second-edition (https://www.packtpub.com/big-data-and-business-intelligence/learning-scipy-numerical-and-scientific-computing-second-edition)

- Rojas, S., Fernández, H., and Ruiz, J. C. (2016) "Aprendiendo a programar en Python con mi computador: Primeros pasos rumbo a cómputos de gran escala en las Ciencias e Ingenierías https://github.com/rojassergio/Aprendiendo-a-programar-en-Python-con-mi-computador (https://github.com/rojassergio/Aprendiendo-a-programar-en-Python-con-mi-computador)

- Scipy Lecture Notes: Mathematical optimization: finding minima of functions http://www.scipy-lectures.org/advanced/mathematical_optimization/ (http://www.scipy-lectures.org/advanced/mathematical_optimization/)