Apache Spark Installation Guide

Click on the relevant link below for your environment-

* + - * [Mac OS X](#_Spark_installation_and_1)
      * [Linux](#_Install_Spark_and)
      * [Windows](#_Install_Spark_in)

Document prepared by: Basant Khati

For Team G,

Datathon 2017

April 16, 2017

# Installing Spark in Mac OS X

**Install Java Development Kit**

Download and install it from [oracle.com](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html)

Add following code to your e.g. .bash\_profile

# For Apache Spark

if which java > /dev/null; then export JAVA\_HOME=$(/usr/libexec/java\_home); fi

Install Apache Spark

You can use Mac OS package manager Brew (<http://brew.sh/>)

brew update

brew install scala

brew install apache-spark

Set up env variables

Add following code to your e.g. .bash\_profile

# For a ipython notebook and pyspark integration

if which pyspark > /dev/null; then

export SPARK\_HOME="/usr/local/Cellar/apache-spark/2.1.0/libexec/"

export PYTHONPATH=$SPARK\_HOME/python:$SPARK\_HOME/python/build:$PYTHONPATH

export PYTHONPATH=$SPARK\_HOME/python/lib/py4j-0.10.4-src.zip:$PYTHONPATH

fi

You can check SPARK\_HOME path using following brew command

$ brew info apache-spark

apache-spark: stable 2.1.0, HEAD

Engine for large-scale data processing

https://spark.apache.org/

/usr/local/Cellar/apache-spark/2.1.0 (1,312 files, 213.9M) \*

Built from source on 2017-02-13 at 00:58:12

From: https://github.com/Homebrew/homebrew-core/blob/master/Formula/apache-spark.rb

Ipython profile

Since [profiles are not supported](http://jupyter.readthedocs.io/en/latest/migrating.html#since-jupyter-does-not-have-profiles-how-do-i-customize-it) in jupyter and now you can see following deprecation warning

$ ipython notebook --profile=pyspark

[TerminalIPythonApp] WARNING | Subcommand `ipython notebook` is deprecated and will be removed in future versions.

[TerminalIPythonApp] WARNING | You likely want to use `jupyter notebook` in the future

[W 01:45:07.821 NotebookApp] Unrecognized alias: '--profile=pyspark', it will probably have no effect.

It seems that it is not possible to run various custom startup files as it was with ipython profiles. Thus, the easiest way will be to run pyspark init script at the beginning of your notebook manually or follow [alternative way](https://gist.github.com/ololobus/4c221a0891775eaa86b0#alternatively).

Run jupyter

$ jupyter-notebook

Initialize pyspark

In [1]: import os

execfile(os.path.join(os.environ["SPARK\_HOME"], 'python/pyspark/shell.py'))

Out[1]: <pyspark.context.SparkContext at 0x10a982b10>

sc variable should be available

In [2]: sc

Out[2]: <pyspark.context.SparkContext at 0x10a982b10>

# Installing Spark in Linux

**Installing Anaconda and setting up Jupyter**

Start by downloading Anaconda 3 and running the installer. Accept the defaults, but set the destination to */opt/anaconda3*.

[root@biginsights ~]# **wget https://repo.continuum.io/archive/Anaconda3-4.1.1-Linux-x86\_64.sh**

--2016-09-28 01:05:08-- https://repo.continuum.io/archive/Anaconda3-4.1.1-Linux-x86\_64.sh

Resolving repo.continuum.io (repo.continuum.io)... 54.221.225.221, 54.225.212.75, 54.225.68.13, ...

Connecting to repo.continuum.io (repo.continuum.io)|54.221.225.221|:443... connected.

HTTP request sent, awaiting response... 200 OK

Length: 425991075 (406M) [application/octet-stream]

Saving to: ‘Anaconda3-4.1.1-Linux-x86\_64.sh’

100%[=================================================================================>] 425,991,075 4.29MB/s in 1m 48s

2016-09-28 01:06:56 (3.76 MB/s) - ‘Anaconda3-4.1.1-Linux-x86\_64.sh’ saved [425991075/425991075]

[root@biginsights ~]# **chmod +x Anaconda3-4.1.1-Linux-x86\_64.sh**

[root@biginsights ~]# **./Anaconda3-4.1.1-Linux-x86\_64.sh**

Welcome to Anaconda3 4.1.1 (by Continuum Analytics, Inc.)

In order to continue the installation process, please review the license

agreement.

Please, press ENTER to continue

>>>

...

cryptography

A Python library which exposes cryptographic recipes and primitives.

Do you approve the license terms? [yes|no]

>>> **yes**

Anaconda3 will now be installed into this location:

/root/anaconda3

- Press ENTER to confirm the location

- Press CTRL-C to abort the installation

- Or specify a different location below

[/root/anaconda3] >>> **/opt/anaconda3**

PREFIX=/opt/anaconda3

installing: python-3.5.2-0 ...

...

installing: conda-env-2.5.1-py35\_0 ...

Python 3.5.2 :: Continuum Analytics, Inc.

creating default environment...

installation finished.

Do you wish the installer to prepend the Anaconda3 install location

to PATH in your /root/.bashrc ? [yes|no]

[no] >>> **yes**

Prepending PATH=/opt/anaconda3/bin to PATH in /root/.bashrc

A backup will be made to: /root/.bashrc-anaconda3.bak

For this change to become active, you have to open a new terminal.

Thank you for installing Anaconda3!

Share your notebooks and packages on Anaconda Cloud!

Sign up for free: https://anaconda.org

[root@biginsights ~]#

Reload your bash profile and verify that you are indeed using Python 3.

[root@biginsights ~]# **source ~/.bashrc**

[root@biginsights ~]# **python**

Python 3.5.2 |Anaconda 4.1.1 (64-bit)| (default, Jul 2 2016, 17:53:06)

[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux

Type "help", "copyright", "credits" or "license" for more information.

>>>

[root@biginsights ~]#

Optionally, verify that you can start the notebook server. Switch to a non-privileged user.

[root@biginsights ~]# **su – nick**

[nick@biginsights ~]$

Add Anaconda to the PATH by exporting a new value for the PATH variable (or by editing your *~/.bash\_profile* and loading it).

[nick@biginsights ~]$ **export PATH=/opt/anaconda3/bin:$PATH**

[nick@biginsights ~]$

Start Jupyter notebook server by running *jupyter notebook* with the host IP for your environment.

[nick@biginsights ~]$ **jupyter notebook --ip="192.168.153.131" --no-browser**

[W 01:38:03.248 NotebookApp] Unrecognized JSON config file version, assuming version 1

[I 01:38:03.427 NotebookApp] [nb\_conda\_kernels] enabled, 1 kernels found

[I 01:38:03.435 NotebookApp] Writing notebook server cookie secret to /home/manchev/.local/share/jupyter/runtime/notebook\_cookie\_secret

[I 01:38:03.701 NotebookApp] ✓ nbpresent HTML export ENABLED

[W 01:38:03.701 NotebookApp] ✗ nbpresent PDF export DISABLED: No module named 'nbbrowserpdf'

[I 01:38:03.704 NotebookApp] [nb\_conda] enabled

[I 01:38:03.736 NotebookApp] [nb\_anacondacloud] enabled

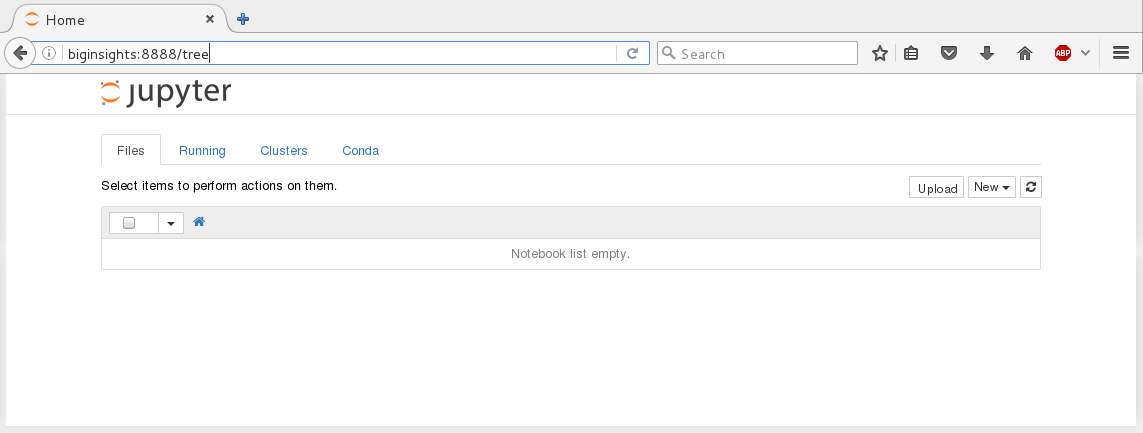
[I 01:38:03.740 NotebookApp] Serving notebooks from local directory: /home/manchev

[I 01:38:03.740 NotebookApp] 0 active kernels

[I 01:38:03.740 NotebookApp] The Jupyter Notebook is running at: http://192.168.153.131:8888/

[I 01:38:03.740 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).

Open a web browser and point to the IP address at port 8888. You should be able to see the main Jupyter page.

[](http://cleverowl.uk/wp-content/uploads/2016/10/jupyter_pyspark_r_1.png)

**Installation of the kernels**

Check what kernels are currently available to Jupyter.

[root@biginsights ~]# **jupyter kernelspec list**

Available kernels:

python3 /opt/anaconda3/lib/python3.5/site-packages/ipykernel/resources

[root@biginsights ~]#

Create a directory to host the PySpark kernel.

[root@biginsights ~]# **mkdir -p /usr/local/share/jupyter/kernels/pyspark**

[root@biginsights ~]#

Create a kernel file named *kernel.json* with the following content and put it in */usr/local/share/jupyter/kernels/pyspark/*. Don't forget to replace values for SPARK\_HOME, PYTHONPATH, and PYTHONSTARTUP with values matching your environment.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | {    "display\_name": "PySpark",    "language": "python",    "argv": [ "/opt/anaconda3/bin/python", "-m", "ipykernel", "-f", "{connection\_file}" ],    "env": {      "SPARK\_HOME": "/usr/iop/current/spark-client",      "PYSPARK\_PYTHON": "/opt/anaconda3/bin/python3",      "PYTHONPATH": "/usr/iop/current/spark-client/python/:/usr/iop/current/spark-client/python/lib/py4j-0.9-src.zip",      "PYTHONSTARTUP": "/usr/iop/current/spark-client/python/pyspark/shell.py",      "PYSPARK\_SUBMIT\_ARGS": "--master yarn-client pyspark-shell"    }  } |

Check that Jupyter can now pick up the new kernel.

[root@biginsights ~]# **jupyter kernelspec list**

Available kernels:

python3 /opt/anaconda3/lib/python3.5/site-packages/ipykernel/resources

pyspark /usr/local/share/jupyter/kernels/pyspark

[root@biginsights ~]#

Create another directory for the R kernel.

[root@biginsights ~]# **mkdir -p /usr/local/share/jupyter/kernels/r**

[root@biginsights ~]#

Place a *kernel.json* file inside with the following content:

|  |  |
| --- | --- |
| 1  2  3  4  5 | {   "argv": ["R", "--slave", "-e", "IRkernel::main()", "--args", "{connection\_file}"],   "display\_name":"R",   "language":"R"  } |

Verify that kernel is now in the available kernels list:

[root@biginsights ~]# **jupyter kernelspec list**

Available kernels:

python3 /opt/anaconda3/lib/python3.5/site-packages/ipykernel/resources

pyspark /usr/local/share/jupyter/kernels/pyspark

r /usr/local/share/jupyter/kernels/r

[root@biginsights ~]#

As the R kernel is not part of the CRAN repos it has to be compiled from sources. Start by getting the following additional packages (if they are not already part of your OS installation):

[root@biginsights ~]# **yum install -y openssl-devel openssl libcurl-devel libssh2-devel**

...

[root@biginsights ~]#

Create links to *libssl.so.1.0.0* and *libcrypto.so.1.0.0* under */usr/lib64* to avoid errors like "libssl.so.10: cannot open shared object file" during compilation.

[root@biginsights ~]# **ln -s /opt/anaconda3/lib/libssl.so.1.0.0 /usr/lib64/libssl.so.1.0.0**

[root@biginsights ~]# **ln -s /opt/anaconda3/lib/libcrypto.so.1.0.0 /usr/lib64/libcrypto.so.1.0.0**

[root@biginsights ~]#

Start R and install the following packages.

[root@biginsights ~]# **R**

R version 3.3.0 (2016-05-03) -- "Supposedly Educational"

Copyright (C) 2016 The R Foundation for Statistical Computing

Platform: x86\_64-redhat-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.

You are welcome to redistribute it under certain conditions.

Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.

Type 'contributors()' for more information and

'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or

'help.start()' for an HTML browser interface to help.

Type 'q()' to quit R.

> **install.packages('git2r')**

...

>

Add the following packages, making sure that they get compiled and installed correctly:

> **install.packages('devtools')**

...

> **install.packages('repr')**

...

> **install.packages('IRdisplay')**

...

> **install.packages('crayon')**

...

> **install.packages('pbdZMQ')**

...

>

Now you can use *devtools* to get and compile IRkernel.

> **devtools::install\_github('IRkernel/IRkernel')**

…

>

If you are planning to use SparkR now it's the time to get and install the *SparkR* package (make sure to use the correct package version for your version of Spark).

> **devtools::install\_github('apache/spark@v1.6.1', subdir='R/pkg')**

…

>

Start (restart) your notebook server and verify that the new kernels are available.

**Testing the kernels**

Create a HDFS directory for your non-privileged user and put some test data inside.

[root@biginsights ~]# **su - hdfs**

[hdfs@biginsights ~]$ **hadoop fs -mkdir /user/nick**

[hdfs@biginsights ~]$ **hadoop fs -chown nick:nick /user/nick**

[hdfs@biginsights ~]$ **hadoop fs -chmod 775 /user/nick**

[hdfs@biginsights ~]$

Get some test data and upload it to the HDFS directory.

[root@biginsights ~]# **su - nick**

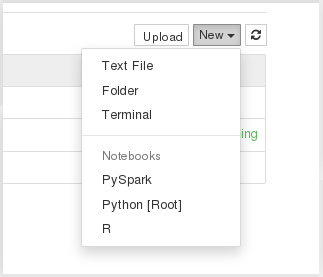
[nick@biginsights ~]$ **wget https://github.com/databricks/spark-csv/raw/master/src/test/resources/cars.csv**

...

[nick@biginsights ~]$ **hadoop fs -put cars.csv .**

[nick@biginsights ~]$

Open the Jupyter web page (IP:8888) and select New.

[](http://cleverowl.uk/wp-content/uploads/2016/10/jupyter_pyspark_r_2.png)

Select PySpark and read some data from the test file. You can put the following code in a cell and run the cell, making sure that it executes successfully.

|  |  |
| --- | --- |
| 1  2 | lines = sc.textFile("cars.csv")  lines.count() |

Next, you can test R and SparkR by parsing the CSV file and loading the data in an R Data Frame.

If the CSV data source is not already part of your Spark installation, you'll have to download it and add it to your Spark libraries.

[root@biginsights ~]# **wget http://central.maven.org/maven2/com/databricks/spark-csv\_2.11/1.5.0/spark-csv\_2.11-1.5.0.jar**

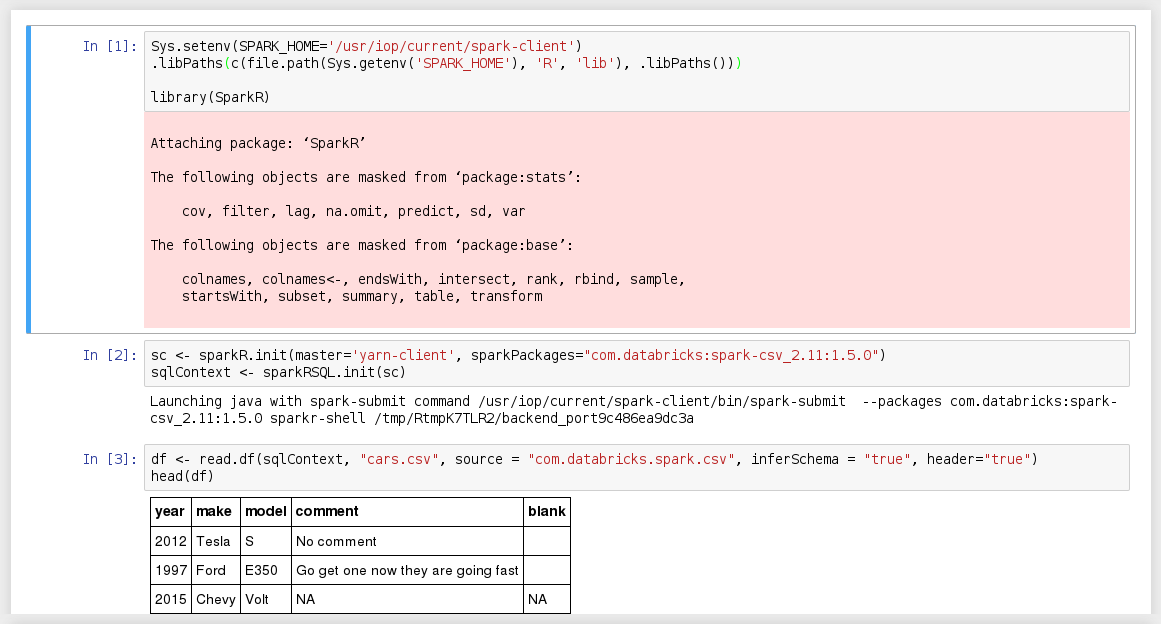
...

[root@biginsights ~]# **mv spark-csv\_2.11-1.5.0.jar /usr/iop/current/spark-client/lib**

[root@biginsights ~]#

Next, create a new notebook using the R kernel. Put the following code inside (mind the correct *spark-csv* version) and run it.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | Sys.setenv(SPARK\_HOME='/usr/iop/current/spark-client')  .libPaths(c(file.path(Sys.getenv('SPARK\_HOME'), 'R', 'lib'), .libPaths()))    library(SparkR)    sc <- sparkR.init(master='yarn-client', sparkPackages="com.databricks:spark-csv\_2.11:1.5.0")  sqlContext <- sparkRSQL.init(sc)    df <- read.df(sqlContext, "cars.csv", source = "com.databricks.spark.csv", inferSchema = "true", header="true")  head(df) |

[](http://cleverowl.uk/wp-content/uploads/2016/10/jupyter_pyspark_r_4.png)

Congratulations, you now have a PySpark/R/SparkR work environment based on Jupyter.

# 

# Installing Spark in Windows

# \*\* (I’ve not tested this myself since I don’t have a Windows machine)

**Install and Setup**

Spark provides APIs in Scala, Java, Python (PySpark) and R. We use PySpark and Jupyter, previously known as IPython Notebook, as the development environment. There are many articles online that talk about Jupyter and what a great tool it is, so we won’t introduce it in details here.

This Guide Assumes you already have Anaconda and Gnu On Windows installed. See <https://mas-dse.github.io/startup/anaconda-windows-install/>

1. Go to [http://www.java.com](http://www.java.com/) and install Java 7+.

2. Get Spark pre-built package from the [downloads page](http://spark.apache.org/downloads.html) of the Spark project website.

3. Open PowerShell by pressing ⊞ Win-R, typing “powershell” in Run dialog box and clicking “OK”. Change your working directory to where you downloaded the Spark package.

4. Type the commands in red to uncompress the Spark download. Alternatively, you can use any other software of your preference to uncompress.

> **gzip -d spark-2.1.0-bin-hadoop2.7.tgz**  
> **tar xvf spark-2.1.0-bin-hadoop2.7.tar**

5. Type the commands in red to move Spark to the c:\opt\spark\ directory.

> **mkdir C:\opt\**  
> **move spark-2.1.0-bin-hadoop2.7 C:\opt\spark\**

6. Type the commands in red to download winutils.exe for Spark.

> **cd C:\opt\spark\bin\**  
> **curl -k -L -o winutils.exe https://github.com/steveloughran/winutils/blob/master/hadoop-2.6.0/bin/winutils.exe?raw=true**

7. Create an environment variable with variable name = SPARK\_HOME and variable value = C:/opt/spark. This link provides a good description of [how to set environment variable in windows](http://www.forbeslindesay.co.uk/post/42833119552/permanently-set-environment-variables-on-windows)

8. Type the commands in red to create a temporary directory.

> **mkdir ~/Documents/jupyter-temp/**  
> **cd ~/Documents/jupyter-temp/**

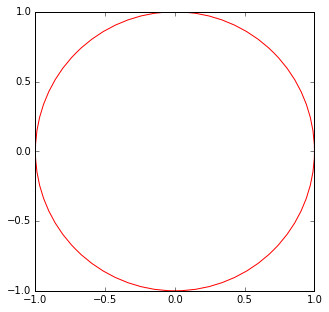
9. Type the commands in red to install, configure and run Jupyter Notebook. Jupyter Notebook will launch using your default web browser.

> **conda install jupyter -y**  
> **ipython kernelspec install-self**  
> **jupyter notebook**

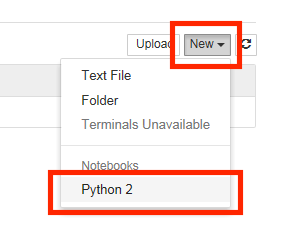
**First Spark Application**

In our first Spark application, we will run a Monte Carlo experiment to find an estimate for $\pi$.

Here is how we are going to do it. The figure bellow shows a circle with radius $r = 1$ inscribed within a 2×2 square. The ratio between the area of the circle and the area of the square is $\frac{\pi}{4}$. If we sample enough points in the square, we will have approximately $\rho = \frac{\pi}{4}$ of these points that lie inside the circle. So we can estimate $\pi$ as $4 \rho$.



1. Create a new Notebook by selecting **Python 2** from the **New** drop down list at the right of the page.



2. First we will create the Spark Context. Copy and paste the red text into the first cell then click the  (run cell) button:

**import os**  
**import sys**  
  
**import findspark**  
**findspark.init()**  
  
**from pyspark import SparkContext**  
  
**sc = SparkContext(master="local[4]")**

3. Next, we draw a sufficient amount of points inside the square. Copy and paste the red text into the next cell then click the  (run cell) button:

**import numpy as np**  
  
**TOTAL = 1000000**  
**dots = sc.parallelize([2.0 \* np.random.random(2) - 1.0 for i in range(TOTAL)]).cache()**  
**print("Number of random points:", dots.count())**  
  
**stats = dots.stats()**  
**print('Mean:', stats.mean())**  
**print('stdev:', stats.stdev())**

Output:

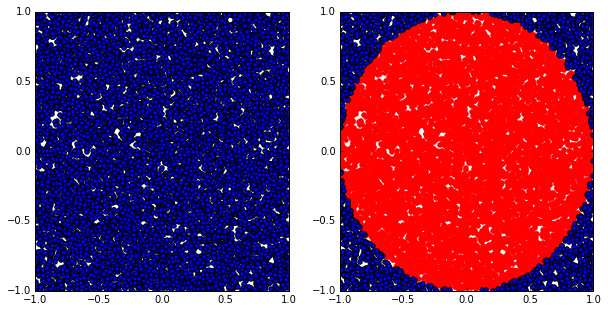
('Number of random points:', 1000000)  
('Mean:', array([-0.0004401 , 0.00052725]))  
('stdev:', array([ 0.57720696, 0.57773085]))

4. We can sample a small fraction of these points and visualize them. Copy and paste the red text into the next cell then click the  (run cell) button:

**%matplotlib inline**  
**from operator import itemgetter**  
**from matplotlib import pyplot as plt**  
  
**plt.figure(figsize = (10, 5))**  
  
**# Plot 1**  
**plt.subplot(1, 2, 1)**  
**plt.xlim((-1.0, 1.0))**  
**plt.ylim((-1.0, 1.0))**  
  
**sample = dots.sample(False, 0.01)**  
**X = sample.map(itemgetter(0)).collect()**  
**Y = sample.map(itemgetter(1)).collect()**  
**plt.scatter(X, Y)**  
  
**# Plot 2**  
**plt.subplot(1, 2, 2)**  
**plt.xlim((-1.0, 1.0))**  
**plt.ylim((-1.0, 1.0))**  
  
**inCircle = lambda v: np.linalg.norm(v) <= 1.0**  
**dotsIn = sample.filter(inCircle).cache()**  
**dotsOut = sample.filter(lambda v: not inCircle(v)).cache()**  
  
**# inside circle**  
**Xin = dotsIn.map(itemgetter(0)).collect()**  
**Yin = dotsIn.map(itemgetter(1)).collect()**  
**plt.scatter(Xin, Yin, color = 'r')**  
  
**# outside circle**  
**Xout = dotsOut.map(itemgetter(0)).collect()**  
**Yout = dotsOut.map(itemgetter(1)).collect()**  
**plt.scatter(Xout, Yout)**

Output:

﻿<matplotlib.collections.PathCollection at 0x17a78780>



5. Finally, let’s compute the estimated value of $\pi$. Copy and paste the red text into the next cell then click the  (run cell) button:

**pi = 4.0 \* (dots.filter(inCircle).count() / float(TOTAL))**  
**print("The estimation of \pi is:", pi)**

Output:

('The estimation of \\pi is:', 3.142204)