Programmation Python sous Spark avec PySpark

Installation du framework Spark sous Windows

La librairie MLlib de Spark pour le Machine Learning

Programmation Python avec PySpark

Le framework Spark – Objectif du tutoriel

Apache Spark est un framework open source de calcul distribué dédié au Big Data. Sa particularité est qu'il est capable de travailler en mémoire vive. Il est très performant pour les opérations nécessitant plusieurs itérations sur les mêmes données, exactement ce dont ont besoin les algorithmes de machine learning.

Spark peut fonctionner sans Hadoop, mais il a besoin d'un gestionnaire de clusters (qu'il a en interne) et d'un système de fichiers distribués (qu'il n'a pas), ce que peut lui fournir Hadoop avec respectivement Hadoop Yarn et HDFS (Hadoop Distributed File System). De fait, les faire fonctionner ensemble est très avantageux (Hadoop, stockage ; Spark, calculs).

Au-delà des API (modules de classes et fonctions) standards, Spark intègre des librairies additionnelles : Streaming, traitement des données en flux ; SQL, accès aux données Spark avec des requêtes SQL ; GraphX, traitement des graphes ; MLlib, types de données et algorithmes pour le machine learning.

PySpark (Spark Python API) est une librairie qui permet de manipuler les objets et méthodes de **Spark** en programmation Python, et de bénéficier directement de ses avantages (gestion de la volumétrie, calcul distribué). Elle inclut la MLlib (le portage est plus avancé par rapport à R/SParkR). Ce tutoriel a pour objectif de s'initier à l'utilisation de Python/PySpark en traitant un exemple typique d'analyse prédictive.

Plan

1. Installation de Spark sous Windows
2. Installation de l'environnement de développement – Anaconda
3. Programmation Python avec PySpark
4. Références

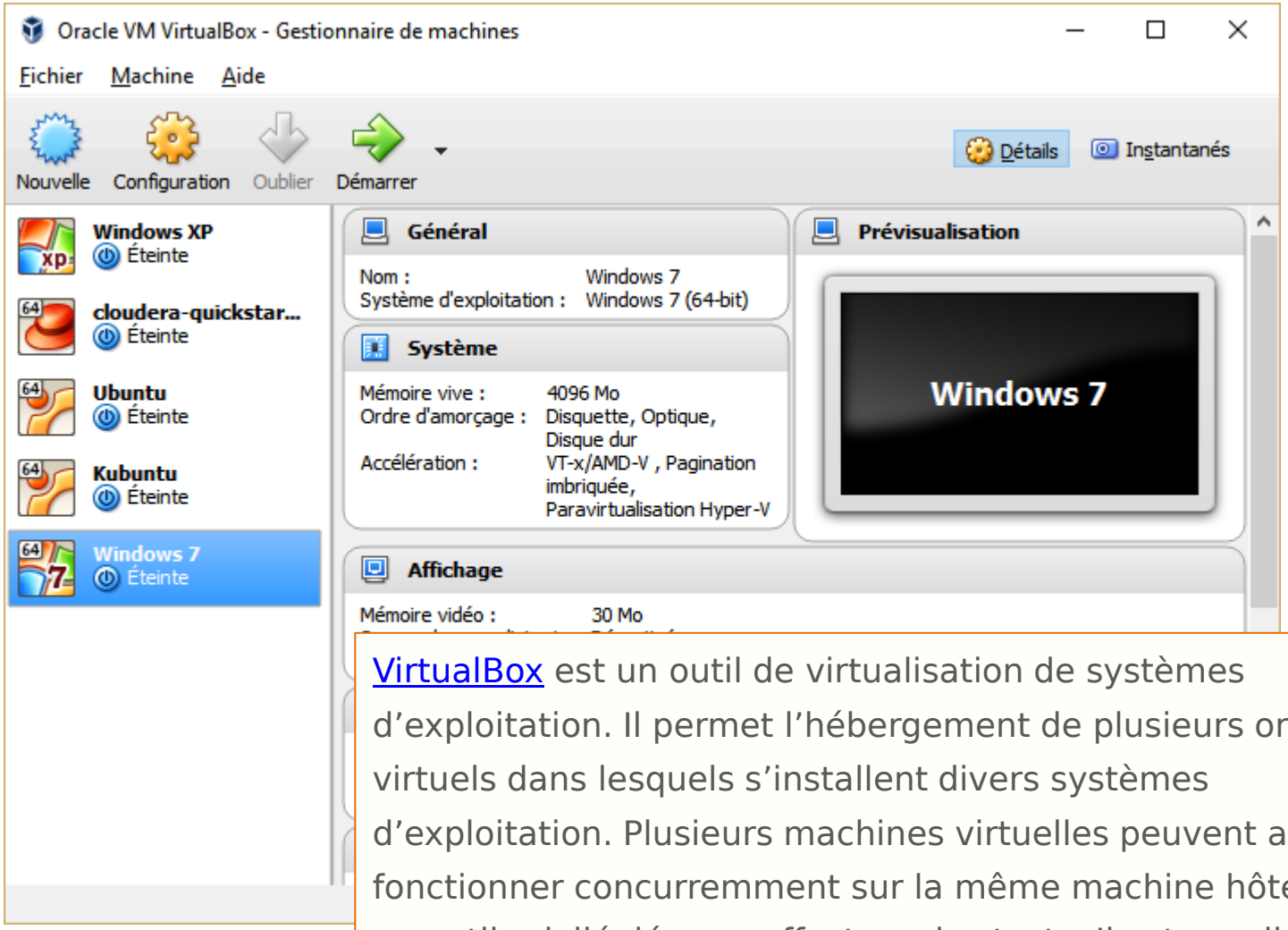
Installation et configuration du framework Spark

Plusieurs pistes sont possibles. Certains éditeurs proposent des systèmes complets (système d'exploitation + Spark) déjà configurés que l'on peut installer directement sur une machine ou une machine virtuelle (ex. Cloudera).

Mais nous pouvons également installer le framework sur un système d'exploitation préexistant. C'est le choix que nous avons fait dans ce tutoriel. Nous nous appuyons sur Windows 7 64 bits (Edition familiale)

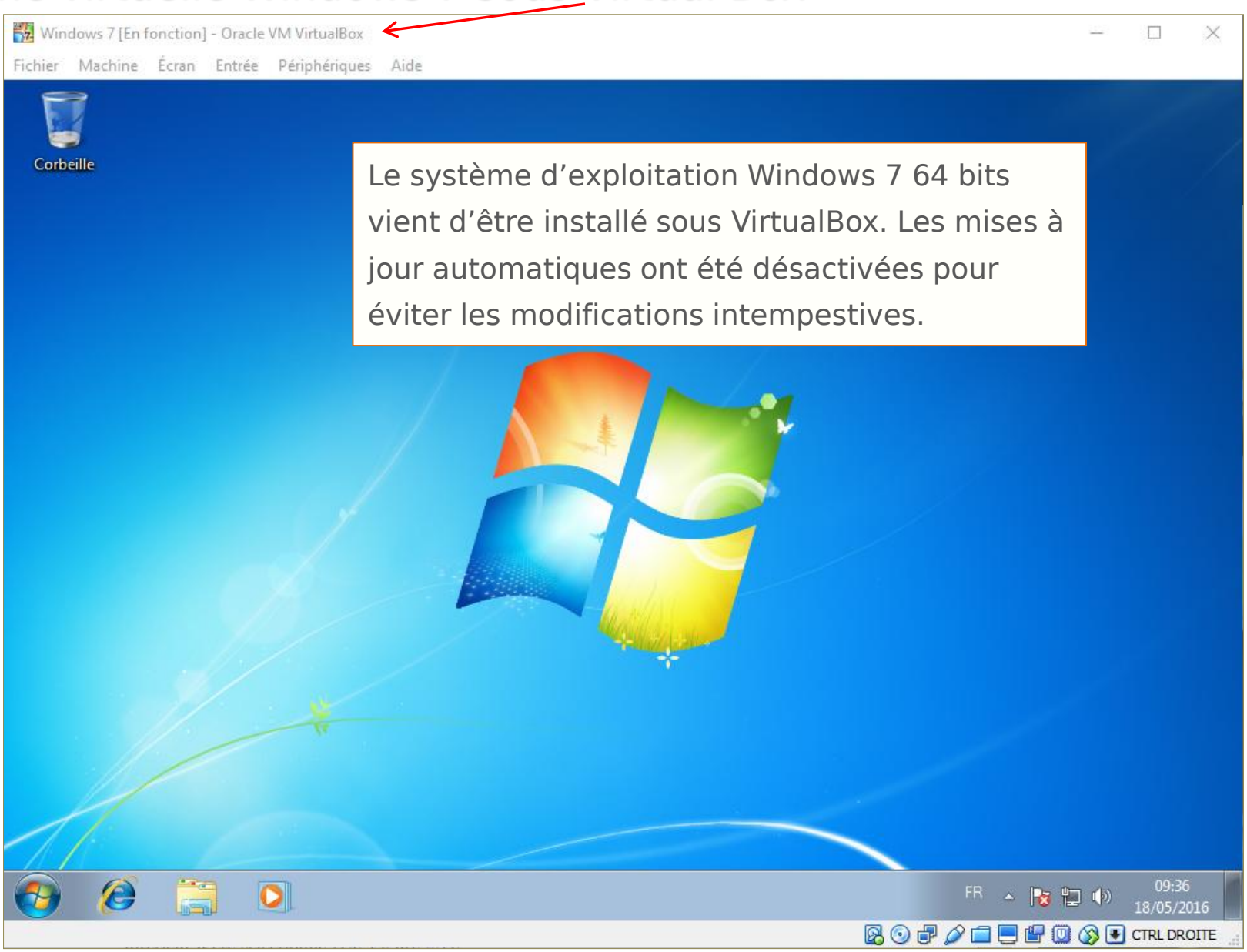
Pour éviter les interférences, nous partons d'une machine virtuelle vierge hébergée par Virtual Box, un outil de virtualisation libre.

VirtualBox



VirtualBox est un outil de virtualisation de systèmes d'exploitation. Il permet l'hébergement de plusieurs ordinateurs virtuels dans lesquels s'installent divers systèmes d'exploitation. Plusieurs machines virtuelles peuvent ainsi fonctionner concurremment sur la même machine hôte. C'est un outil privilégié pour effectuer des tests. Il est possible d'archiver différents états de la même machine.

Machine virtuelle Windows 7 sous Virtual Box



Installation de Java JDK version 8 (64 bits)

Java SE Development Kit 8 - Downloads - Windows Internet Explorer

http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

Fichier Edition Affichage Favoris Outils ?

Favoris | Sites suggérés | Galerie de composants ...

Java SE Development Kit 8 - Downloads

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Accept License Agreement | Decline License Agreement

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.72 MB	jdk-8u91-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.69 MB	jdk-8u91-linux-arm64-vfp-hflt.tar.gz
Linux x86	154.74 MB	jdk-8u91-linux-i586.rpm
Linux x86	174.92 MB	jdk-8u91-linux-i586.tar.gz
Linux x64	152.74 MB	jdk-8u91-linux-x64.rpm
Linux x64	172.97 MB	jdk-8u91-linux-x64.tar.gz
Mac OS X	227.29 MB	jdk-8u91-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.59 MB	jdk-8u91-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	98.95 MB	jdk-8u91-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.29 MB	jdk-8u91-solaris-x64.tar.Z
Solaris x64	96.78 MB	jdk-8u91-solaris-x64.tar.gz
Windows x86	182.11 MB	jdk-8u91-windows-i586.exe
Windows x64	187.41 MB	jdk-8u91-windows-x64.exe

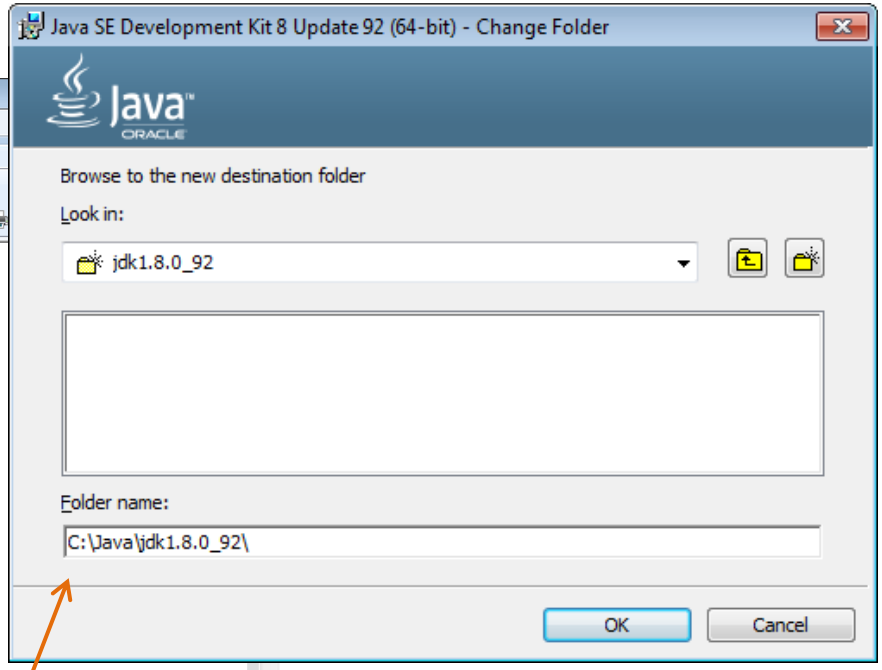
Java SE Development Kit 8u92

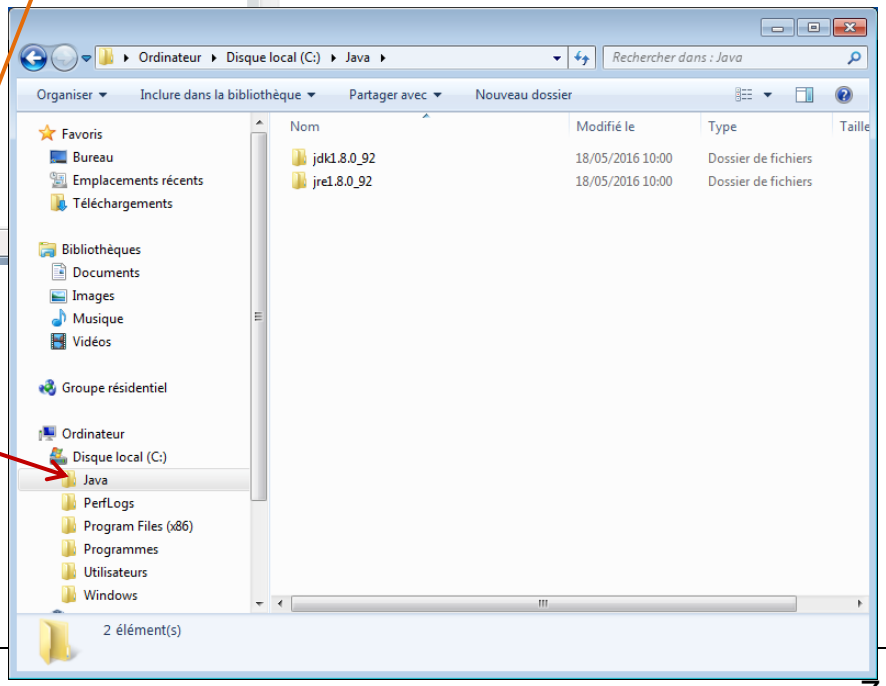
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.

Product / File Description	File Size	Download
Linux x86	160.26 MB	jdk-8u92-linux-i586.rpm
Linux x86	174.94 MB	jdk-8u92-linux-i586.tar.gz
Linux x64	158.27 MB	jdk-8u92-linux-x64.rpm
Linux x64	172.99 MB	jdk-8u92-linux-x64.tar.gz
Mac OS X	227.32 MB	jdk-8u92-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.47 MB	jdk-8u92-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	98.93 MB	jdk-8u92-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.35 MB	jdk-8u92-solaris-x64.tar.Z
Solaris x64	96.76 MB	jdk-8u92-solaris-x64.tar.gz
Windows x86	188.43 MB	jdk-8u92-windows-i586.exe
Windows x64	193.66 MB	jdk-8u92-windows-x64.exe

Terminé, mais il existe des erreurs sur la page. | Internet | Mode protégé : activé





Java a été installé à la racine, dans le dossier « Java ». 2 sous-répertoires ont été créés pour le JDK et le JRE.

L'utilitaire WinUtils

Hadoop 2.6.0 Windows 64

https://www.barik.net/archive/2015/01/19/172716/

Titus Barik in General

Hadoop 2.6.0 Windows 64-bit Binaries

The official release of Apache Hadoop 2.6.0 does not include the required binaries (e.g., `winutils.exe`) necessary to run hadoop. In order to use Hadoop on Windows, it must be compiled from source. This takes a bit of effort, so I've provided a pre-compiled, unofficial distribution below:

- hadoop-2.6.0.tar.gz
  SHA1: `205b235d77213b958f126647241a5092845a0ff8`
- hadoop-dist-2.6.0-javadoc.jar
  SHA1: `cbccbc5d7eeb03261e533cbe0b1b367fce83b181`
- hadoop-2.6.0-src.tar.gz
  SHA1: `4a1dfa9bd34d5efb7f2a0cd4dcf03db3eab46a5d`

Plusieurs utilitaires, dont ''winutils'', sont nécessaires pour pouvoir faire fonctionner Hadoop sous Windows.

La distribution Spark que nous allons utiliser est adossée à Hadoop.

Contenu du fichier archive.

Nous désarchivons le fichier dans le dossier « c:\winutils »

Ci-contre le contenu du sous-répertoire « c:\wintils\bin »

Package redistribuable MS Visual C++ 2010

Ce package doit être installé. Il est nécessaire au bon fonctionnement de « winutils ».

Chrome | Download Package redisti

Microsoft Corporation [US] https://www.microsoft.com/fr-FR/download/details.aspx?id=14632

Package redistribuable Microsoft Visual C++ 2010 (x64)

Sélectionnez une langue : Français

Télécharger

Le package redistribuable Microsoft Visual C++ 2010 installe les composants runtime des bibliothèques Visual C++ nécessaires pour exécuter les applications développées en Visual C++ sur un ordinateur sur lequel Visual C++ 2010 n'est pas installé.

- Détails
- Configuration système
- Instructions d'installation
- Informations complémentaires
- Ressources associées

FR 11:26 18/05/2016

Choix de la version de Spark et installation

Downloads | Apache Spar

spark.apache.org/downloads.html

Spark — Lightning-fast cluster computing

Download | Libraries | Documentation | Examples | Community | FAQ | Apache Software Foundation

Latest News

- Spark Summit (June 6, 2016, San Francisco) agenda posted (Apr 17, 2016)
- Spark 1.6.1 released (Mar 09, 2016)
- Submission is open for Spark Summit San Francisco (Feb 11, 2016)
- Spark Summit East (Feb 16, 2016, New York) agenda posted (Jan 14, 2016)

Download Apache Spark™

Our latest version is Spark 1.6.1, released on March 9, 2016 (release notes) (git ta

1. Choose a Spark release: 1.6.1 (Mar 09 2016)
2. Choose a package type: Pre-built for Hadoop 2.6 and later
3. Choose a download type: Direct Download
4. Download Spark: spark-1.6.1-bin-hadoop2.6.tgz
5. Verify this release using the 1.6.1 signatures and checksums.

Note: Scala 2.11 users should download the Spark source package and build with S

Chargement de la version pour Hadoop 2.6.

Nous désarchivons simplement le fichier dans un répertoire dédié. Ici : « c:\spark-1.6.1-bin-hadoop2.6 »

Ordinateur ▸ Disque local (C:) ▸ — Rechercher dans : Disque local (C:)

Organiser | Partager avec | Nouveau dossier

Favoris: Bureau, Emplacements récents, Téléchargements

Bibliothèques: Documents, Images, Musique, Vidéos

Groupe résidentiel

Ordinateur: Disque local (C:), Lecteur de CD (D:) VirtualBox Guest Additio, Echange (\\vboxsrv) (E:)

Réseau

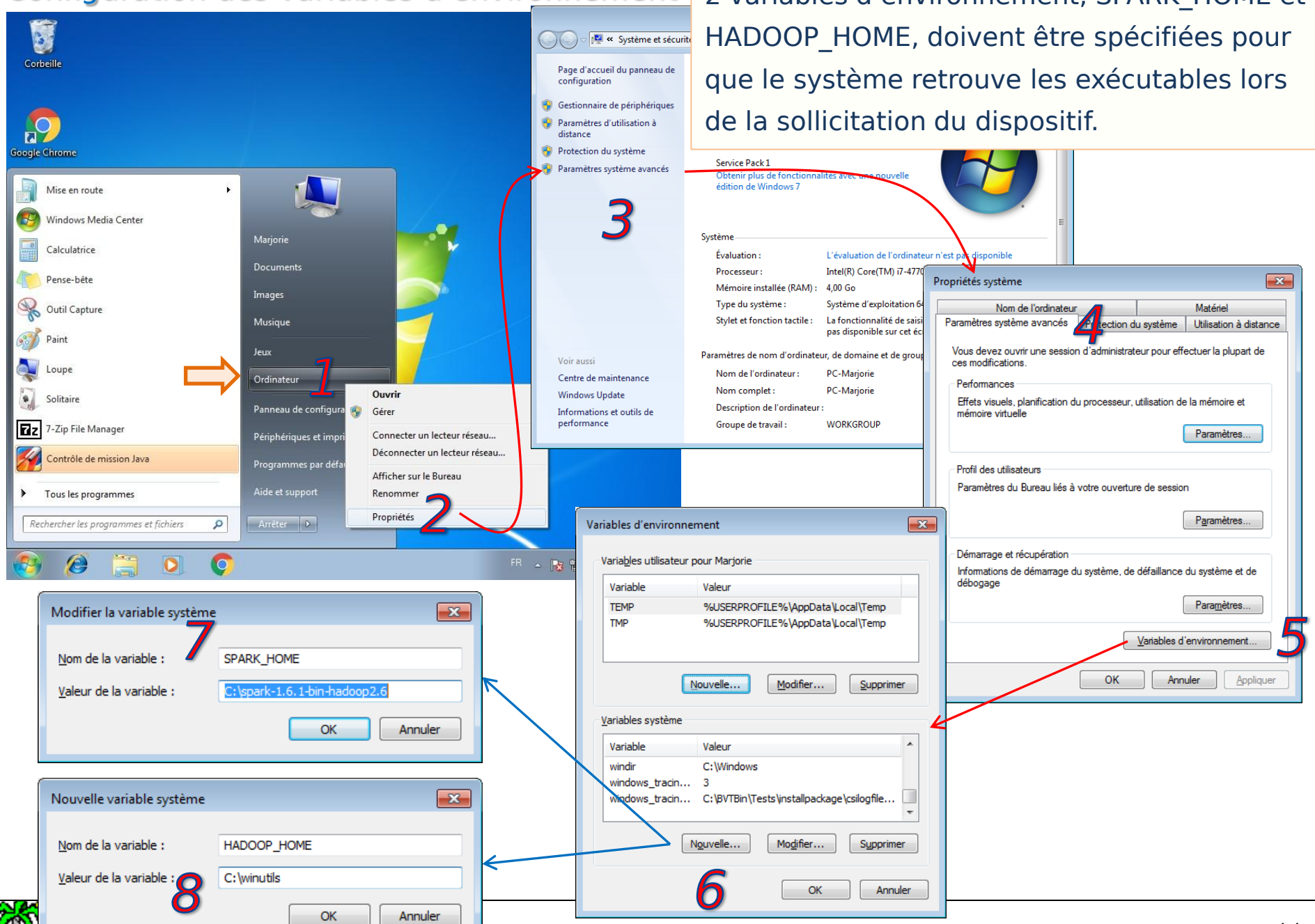
Nom	Modifié le	Type
Java	18/05/2016 09:59	Dossier de fichiers
PerfLogs	14/07/2009 05:20	Dossier de fichiers
Program Files (x86)	18/05/2016 10:15	Dossier de fichiers
Programmes	18/05/2016 10:34	Dossier de fichiers
spark-1.6.1-bin-hadoop2.6	18/05/2016 11:03	Dossier de fichiers
Utilisateurs	18/05/2016 06:42	Dossier de fichiers
Windows	18/05/2016 09:54	Dossier de fichiers
winutils	18/05/2016 10:39	Dossier de fichiers

8 élément(s)

Voici l'organisation des dossiers de notre machine à ce stade.

Configuration des variables d'environnement

2 variables d'environnement, SPARK_HOME et HADOOP_HOME, doivent être spécifiées pour que le système retrouve les exécutables lors de la sollicitation du dispositif.

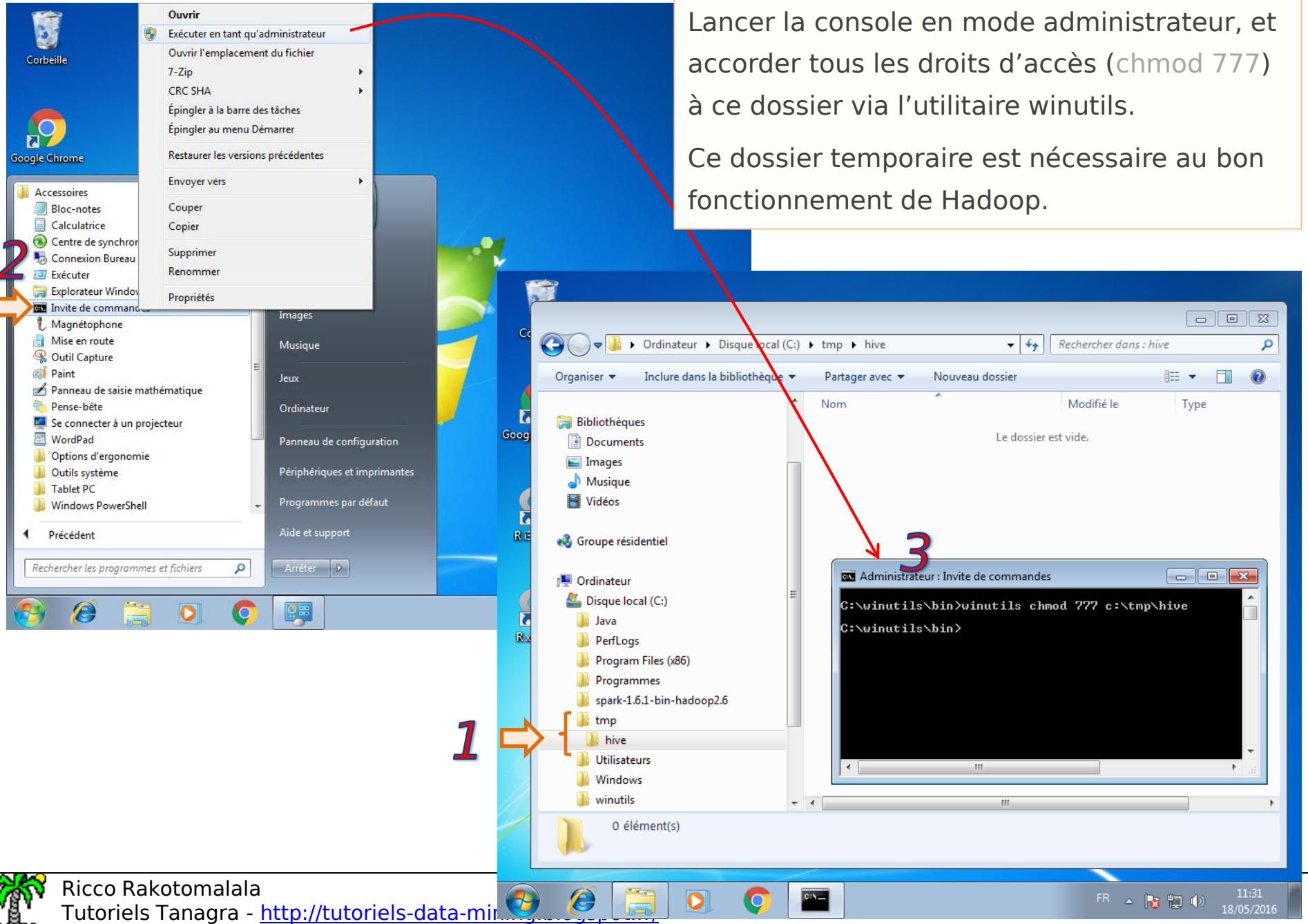


Configuration de Hadoop via winutils

Créer le dossier « c:\tmp\hive » au préalable.

Lancer la console en mode administrateur, et accorder tous les droits d'accès (chmod 777) à ce dossier via l'utilitaire winutils.

Ce dossier temporaire est nécessaire au bon fonctionnement de Hadoop.



Lancer Spark

Administrateur : Invite de commandes

```
C:\spark-1.6.1-bin-hadoop2.6\bin>dir
 Le volume dans le lecteur C n'a pas de nom.
 Le numéro de série du volume est 7882-C8DD

 Répertoire de C:\spark-1.6.1-bin-hadoop2.6\bin

18/05/2016  11:36    <REP>          .
18/05/2016  11:36    <REP>          ..
27/02/2016  07:02             1 099 beeline
27/02/2016  07:02               932 beeline.cmd
19/05/2016  12:03            21 118 derby.log
27/02/2016  07:02             1 910 load-spark-env.cmd
27/02/2016  07:02             2 143 load-spark-env.sh
19/05/2016  12:02    <REP>          metastore_db
27/02/2016  07:02             3 459 pyspark
27/02/2016  07:02             1 000 pyspark.cmd
27/02/2016  07:02             1 486 pyspark2.cmd
27/02/2016  07:02             2 384 run-example
27/02/2016  07:02             1 012 run-example.cmd
27/02/2016  07:02             2 682 run-example2.cmd
27/02/2016  07:02             2 858 spark-class
27/02/2016  07:02             1 010 spark-class.cmd
27/02/2016  07:02             2 365 spark-class2.cmd
27/02/2016  07:02             3 026 spark-shell
27/02/2016  07:02             1 008 spark-shell.cmd
27/02/2016  07:02             1 528 spark-shell2.cmd
27/02/2016  07:02             1 075 spark-sql
27/02/2016  07:02             1 050 spark-submit
27/02/2016  07:02             1 010 spark-submit.cmd
27/02/2016  07:02             1 126 spark-submit2.cmd
27/02/2016  07:02             1 049 sparkR
27/02/2016  07:02               998 sparkR.cmd
27/02/2016  07:02             1 010 sparkR2.cmd
               24 fichier(s)           58 338 octets
                3 Rép(s)  257 175 998 464 octets libres

C:\spark-1.6.1-bin-hadoop2.6\bin>spark-shell
```

1

Administrateur : Invite de commandes - spark-shell

```
-rdbms-3.2.9.jar" is already registered, and you are trying to register an ident
ical plugin located at URL "file:/C:/spark-1.6.1-bin-hadoop2.6/bin/../lib/datanu
cleus-rdbms-3.2.9.jar."
16/05/19 12:01:31 WARN General: Plugin (Bundle) "org.datanucleus" is already reg
istered. Ensure you dont have multiple JAR versions of the same plugin in the cl
asspath. The URL "file:/C:/spark-1.6.1-bin-hadoop2.6/bin/../lib/datanucleus-core
-3.2.10.jar" is already registered, and you are trying to register an identical
plugin located at URL "file:/C:/spark-1.6.1-bin-hadoop2.6/lib/datanucleus-core-3
.2.10.jar."
16/05/19 12:01:31 WARN General: Plugin (Bundle) "org.datanucleus.api.jdo" is alr
eady registered. Ensure you dont have multiple JAR versions of the same plugin i
n the classpath. The URL "file:/C:/spark-1.6.1-bin-hadoop2.6/bin/../lib/datanucl
eus-api-jdo-3.2.6.jar" is already registered, and you are trying to register an
identical plugin located at URL "file:/C:/spark-1.6.1-bin-hadoop2.6/lib/datanucl
eus-api-jdo-3.2.6.jar."
16/05/19 12:01:32 WARN Connection: BoneCP specified but not present in CLASSPATH
 (or one of dependencies)
16/05/19 12:01:33 WARN Connection: BoneCP specified but not present in CLASSPATH
 (or one of dependencies)
16/05/19 12:02:03 WARN ObjectStore: Version information not found in metastore.
hive.metastore.schema.verification is not enabled so recording the schema versio
n 1.2.0
16/05/19 12:02:04 WARN ObjectStore: Failed to get database default, returning No
SuchObjectException
16/05/19
istered.
asspath.
-3.2.10.
plugin l
.2.10.ja
16/05/19
 already
in in th
-rdbms-3
ical plu
cleus-rdbms-3.2.9.jar."
16/05/19 12:02:23 WARN General: Plugin (Bundle) "org.datanucleus.api.jdo" is alr
eady registered. Ensure you dont have multiple JAR versions of the same plugin i
he classpath. The URL "file:/C:/spark-1.6.1-bin-hadoop2.6/bin/../lib/datanucl
-api-jdo-3.2.6.jar" is already registered, and you are trying to register an
ntical plugin located at URL "file:/C:/spark-1.6.1-bin-hadoop2.6/lib/datanucl
-api-jdo-3.2.6.jar."
05/19 12:02:23 WARN Connection: BoneCP specified but not present in CLASSPATH
r one of dependencies)
05/19 12:02:24 WARN Connection: BoneCP specified but not present in CLASSPATH
r one of dependencies)
 context available as sqlContext.

scala>
```

2

Mais pour l'exploiter, il faudrait savoir coder en « scala », ce qui n'est pas mon cas (Remarque : exit permet de sortir de l'interpréteur de commandes).

Via le terminal de commande lancé en mode administrateur. Liste des exécutables.

On peut lancer Spark avec « spark-shell »

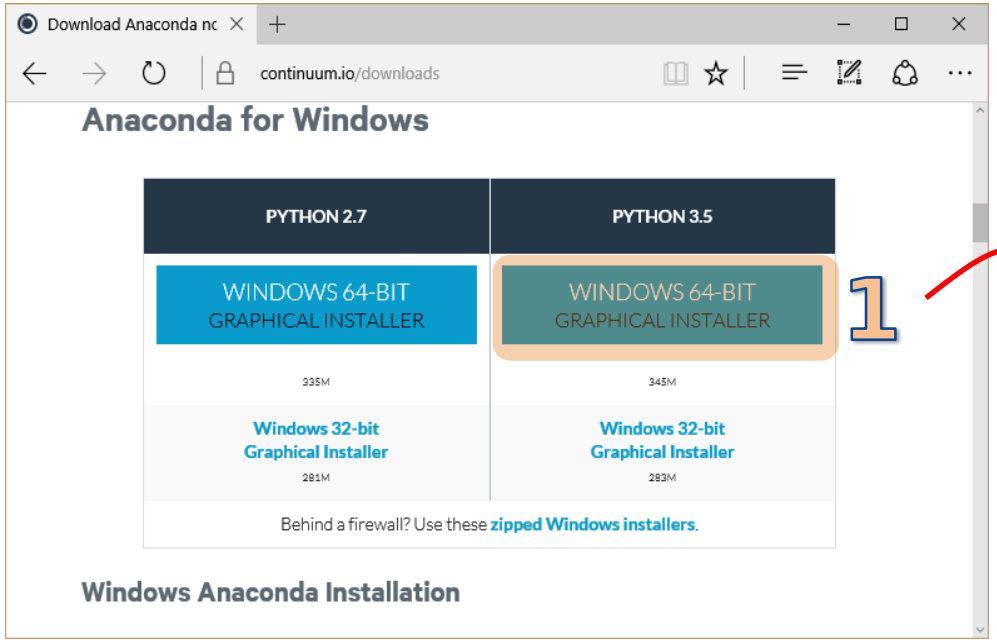
Installation et configuration de l'environnement de développement

Installation de Anaconda qui inclut
Python et l'éditeur de code Spyder

Installation de Python via la distribution Anaconda

Voir http://tutoriels-data-mining.blogspot.fr/2015/08/python-la-distribution-anaconda.html

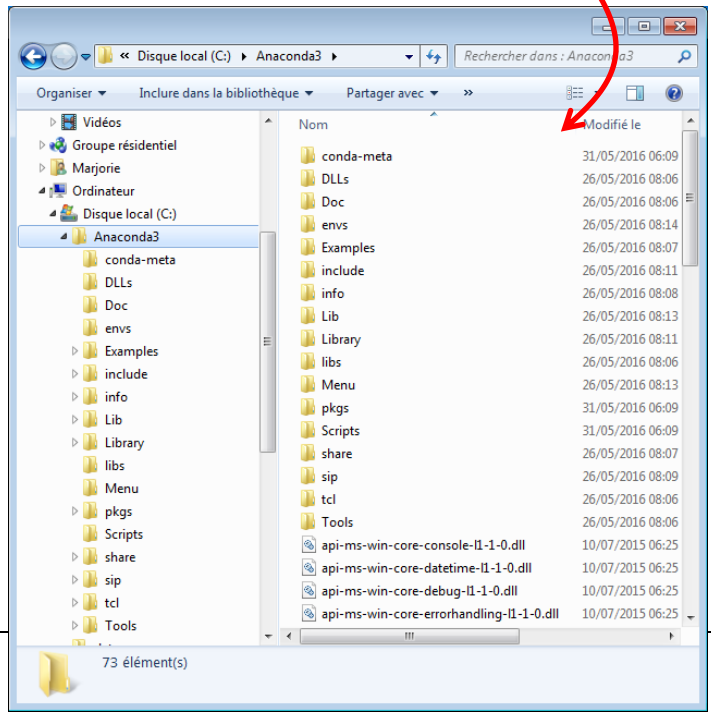
Installation en mode administrateur sous Windows.



1

2

3



La version 3.5 de Python est directement installée.

Première utilisation de PySpark

Administrateur : Invite de commandes - pyspark

```
C:\spark-1.6.1-bin-hadoop2.6\bin>pyspark
Python 3.5.1 |Anaconda 4.0.0 (64-bit)| (default, Feb 16 2016, 09:49:46) [MSC v.1
900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
16/05/26 08:44:06 INFO SparkContext: Running Spark version 1.6.1
16/05/26 08:44:08 WARN NativeCodeLoader: Unable to load native-hadoop library fo
r your platform... using builtin-java classes where applicable
16/05/26 08:44:09 INFO SecurityManager: Changing view acls to: Marjorie
16/05/26 08:44:09 INFO SecurityManager: Changing modify acls to: Marjorie
16/05/26 08:44:09 INFO SecurityManager: SecurityManager: authentication disabled
; ui acls disabled; users with view permissions: Set(Marjorie); users with modif
y permissions: Set(Marjorie)
16/05/26 08:44:12 INFO Utils: Successfully started service 'sparkDriver' on port
 49195.
16/05/26 08:44:13 INFO Slf4jLogger: Slf4jLogger started
16/05/26 08:44:14 INFO Remoting: Starting remoting
16/05/26 08:44:15 INFO Remoting: Remoting started; listening on addresses :[akka
.tcp://sparkDriverActorSystem@10.0.2.15:49208]
16/05/26 08:44:15 INFO Utils: Successfully started service 'sparkDriverActorSyst
em' on port 49208.
16/05/26 08:44:15 INFO SparkEnv: Registering MapOutputTracker
16/05/26 08:44:15 INFO SparkEnv: Registering BlockManagerMaster
16/05/26 08:44:16 INFO DiskBlockManager: Created local directory at C:\Users\Mar
jorie\AppData\Local\Temp\blockmgr-87a89b4b-b971-47cc-a99b-32ead77b84c9
16/05/26 08:44:16 INFO MemoryStore: MemoryStore started with capacity 517.4 MB
16/05/26 08:44:16 INFO SparkEnv: Registering OutputCommitCoordinator
16/05/26 08:44:17 INFO Utils: Successfully started service 'SparkUI' on port 404
0.
16/05/26 08:44:17 INFO SparkUI: Started SparkUI at http://10.0.2.15:4040
16/05/26 08:44:17 INFO Executor: Starting executor ID driver on host localhost
16/05/26 08:44:17 INFO Utils: Successfully started service 'org.apache.spark.net
work.netty.NettyBlockTransferService' on port 49215.
16/05/26 08:44:17 INFO NettyBlockTransferService: Server created on 49215
16/05/26 08:44:17 INFO BlockManagerMaster: Trying to register BlockManager
16/05/26 08:44:17 INFO BlockManagerMasterEndpoint: Registering block manager loc
alhost:49215 with 517.4 MB RAM, BlockManagerId(driver, localhost, 49215)
16/05/26 08:44:17 INFO BlockManagerMaster: Registered BlockManager
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Python version 3.5.1 (default, Feb 16 2016 09:49:46)
SparkContext available as sc, HiveContext available as sqlContext.
>>> _
```

On peut déjà fonctionner directement en lançant ‘‘pyspark’’ dans le terminal de commande.

Administrateur : Invite de commandes - pyspark

```
16/05/31 17:47:33 INFO Remoting: Remoting started; listening on addresses :[akka
.tcp://sparkDriverActorSystem@10.0.2.15:49245]
16/05/31 17:47:33 INFO SparkEnv: Registering MapOutputTracker
16/05/31 17:47:33 INFO SparkEnv: Registering BlockManagerMaster
16/05/31 17:47:33 INFO DiskBlockManager: Created local directory at C:\Users\Mar
jorie\AppData\Local\Temp\blockmgr-71bbffcd-db50-41f8-8a13-680b5ead1946
16/05/31 17:47:33 INFO MemoryStore: MemoryStore started with capacity 517.4 MB
16/05/
16/05/                                                                    404
0.
16/05/
16/05/
16/05/                                                                    net
work.ne
16/05/
16/05/31 17:47:34 INFO BlockManagerMasterEndpoint: Registering block manager loc
alhost:49252 with 517.4 MB RAM, BlockManagerId(driver, localhost, 49252)
16/05/31 17:47:34 INFO BlockManagerMaster: Registered BlockManager
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 1.6.1
      /_/

Using Python version 3.5.1 (default, Feb 16 2016 09:49:46)
SparkContext available as sc, HiveContext available as sqlContext.
>>> a = 10
>>> b = a + 15
>>> print(b)
25
>>> dir()
['HiveContext', 'SQLContext', 'SparkContext', 'StorageLevel', '__builtins__', '_
_cached__', '__doc__', '__loader__', '__name__', '__package__', '__spec__', '_py
thonstartup', 'a', 'add_files', 'atexit', 'b', 'os', 'platform', 'py4j', 'pyspar
k', 'sc', 'sqlContext', 'sqlCtx']
>>>
```

On peut insérer directement ici les commandes Python. Mais ce n’est pas très convivial, j’en conviens.

Utilisation de l'éditeur de code Spyder

Spyder est un environnement de développement disponible avec la distribution Anaconda. Son utilisation n'est pas indispensable, mais il nous facilite grandement la vie.

Spyder (Python 3.5)

Fichier Édition Recherche Source Exécution Déboguer Consoles Outils Affic

Éditeur - C:\data\breast_analysis_spark.py

breast_analysis_spark.py

```
1 # -*- coding: utf-8 -*-
2
3 # modification de la var. système path
4 # indiquant les fichiers des librairies
5 import sys
6 print(sys.path)
7 sys.path.append('C:\\spark-1.6.1-bin-hadoop2.6\\python\\lib\\pyspark.zip')
8 sys.path.append('C:\\spark-1.6.1-bin-hadoop2.6\\python\\lib\\py4j-0.9-src.zip')
9
10 # importation des classes SparkContext et SQLContext
11 from pyspark import SparkContext
12 from pyspark.sql import SQLContext
13
14 # initialisation  d'un SparkContext
15 sc = SparkContext('local','MyExample')
16
17 #initialisation d'un SQL Context
18 sqlC = SQLContext(sc)
19
20 # importation
21 import pandas
22 pdBreast = pa                                  p="\t",header=0)
23 print(pdBreast.shape)
24
25 # transformation en un DataFrame Spark
26 dfBreast = sqlC.createDataFrame(pdBreast)
27
28 # affichage du schéma
29 dfBreast.printSchema()
30
31 # affichage des 5 premières observations
32 dfBreast.show(5)
33
34 # nombre d'observations
35 dfBreast.count()
36
37 # type labeledPoint nécessaire à l'apprentissage supervisé
38 from pyspark.mllib.regression import LabeledPoint
39
40 # Transformation des données d'apprentissage
41 # Rôle de map : Return a new RDD by applying a function to each Row
42 lblBreast = dfBreast.map(lambda line:LabeledPoint(line[9],line[0:9]))
43
44 #affichage des 5 premières lignes
45 lblBreast.take(5)
46
47 # partition aléatoire
48 lblTrain, lblTest = lblBreast.randomSplit([0.67,0.33],1)
```

Editeur de code

Python 1

```
['', 'C:\\Anaconda3\\python35.zip', 'C:\\Anaconda3\\DLLs', '
C:\\Anaconda3\\lib', 'C:\\Anaconda3', 'c:\\anaconda3\\lib\\s
ite-packages\\setuptools-20.3-py3.5.egg', 'C:\\Anaconda3\\li
b\\site-packages', 'C:\\Anaconda3\\lib\\site-packages\\Sphin
x-1.3.5-py3.5.egg', 'C:\\Anaconda3\\lib\\site-packages\\win3
2', 'C:\\Anaconda3\\lib\\site-packages\\win32\\lib', 'C:\\An
aconda3\\lib\\site-packages\\Pythonwin']
>>> sys.path.append('C:\\spark-1.6.1-bin-hadoop2.6\\python\\
lib\\pyspark.zip')
>>> sys.path.append('C:\\spark-1.6.1-bin-hadoop2.6\\python\\
lib\\py4j-0.9-src.zip')
>>> from pyspark import SparkContext
>>> from pyspark.sql import SQLContext
>>> sc = SparkContext('local','MyExample')
Using Spark's default log4j profile: org/apache/spark/log4j-
defaults.properties
16/05/31 17:59:56 INFO SparkContext: Running Spark version 1
.6.1
16/05/31 17:59:56 WARN NativeCodeLoader: Unable to load nati
ve-hadoop library for your platform... using builtin-java cl
asses where applicable
16/05/31 17:59:56 INFO Securi
o: Marjorie
16/05/31 17:59:56 INFO Securi
to: Marjorie
16/05/31 17:59:56 INFO Securi
hentication disabled; ui acls
issions: Set(Marjorie); users
arjorie)
16/05/31 17:59:57 INFO Utils:
sparkDriver' on port 50711.
16/05/31 17:59:57 INFO Slf4jL
16/05/31 17:59:57 INFO Remoti
16/05/31 17:59:58 INFO Utils:
sparkDriverActorSystem' on po
16/05/31 17:59:58 INFO SparkE
r
16/05/31 17:59:58 INFO SparkE
ter
16/05/31 17:59:58 INFO DiskBl
tory at C:\Users\Marjorie\App
67-2ec9-4f79-ac68-1c4a041eddb
16/05/31 17:59:58 INFO Remoti
on addresses :[akka.tcp://spa
0728]
16/05/31 17:59:58 INFO Memory
capacity 517.4 MB
16/05/31 17:59:58 INFO SparkE
```

Console Historique Console IPython

Droits d'accès : RW | Fins de ligne : CRLF | Encodage : UTF-8 | Ligne : 18 | Colonne : 22 | Mémoire : 27 %

Console Python - Visualisation des sorties (Remarque : on peut aussi utiliser la console IPython, mais elles est moins verbeuse, on a moins de détail sur les étapes des opérations. *Ex. ici la création d'un contexte*).

Programmation Python avec PySpark

Exploiter via Python la librairie MLlib de machine learning pour Spark dans un schéma d'analyse prédictive très classique

1 MLlib est une librairie de machine learning pour Spark. Il intègre les algorithmes usuels de fouille de données (classement, régression, clustering – Voir MLlib Guide).

Il permet l'exploitation des capacités de Spark en matière de manipulation et de traitement des gros volumes sans que l'on ait quelque chose de particulier à faire, sauf à connaître les structures de données et les commandes dédiées.

2 PySpark est un package qui fournit des outils permettant d'exploiter les fonctionnalités de Spark et MLlib à partir de Python.

Nous programmons toujours en langage Python, mais de nouvelles structures de données et jeux d'instructions sont à notre disposition pour exploiter pleinement la puissance de Spark pour le traitement des données massives.

Données : Breast Cancer Wisconsin (Serveur UCI)

« Disque local (C:) ▸ data

Rechercher dans : data

Organiser ▾ Inclure dans la bibliothèque ▾ Partager avec ▾ »

- Documents
- Images
- Musique
- Vidéos
- Groupe résidentiel
- Marjorie
- Ordinateur
- Disque local (C:)
  - Anaconda3
  - data
  - Java
  - PerfLogs
  - Program Files (x86)

5 élément(s)

Nom	Modifié le
breast.csv	18/05/2016 11:58
breast_analysis_spark.py	31/05/2016 10:14
breast_analysis_spark.r	18/05/2016 17:46
breast_bin.csv	31/05/2016 08:59
breast_spark_train_test.r	19/05/2016 16:58

Il s'agit d'un fichier texte avec séparateur tabulation. La dernière colonne ''target'' représente la variable cible, elle a été recodée en 0 (begnin) et 1 (malignant). Cf. tutoriel SparkR.

C:\data\breast_bin.csv - Notepad++

Fichier Édition Recherche Affichage Encodage Langage Paramétrage Macro Exécution Compléments Documents ? X

breast_bin.csv

1	clump	ucellsize	ucellshape	mgadhesion	sepics	bnuclei	bchromatin	normnucl	mitoses	target
2	4	2	2	1	2	1	2	1	1	0
3	1	1	1	1	2	1	2	1	1	0
4	2	1	1	1	2	1	2	1	1	0
5	10	6	6	2	4	10	9	7	1	1
6	4	1	1	1	2	1	2	1	1	0
7	1	1	1	1	2	1	1	1	1	0
8	1	1	1	1	2	1	2	1	1	0
9	5	1	1	1	2	1	2	1	1	0

Ln : 2 Col : 38 Sel : 0 | 0 Dos\Windows UTF-8 INS

Le fichier « breast_bin.csv » a été placé dans le dossier « c:\data »

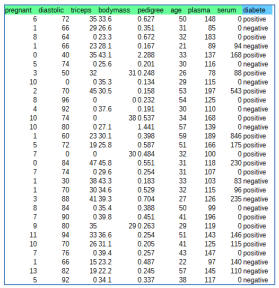
Etapes de la modélisation prédictive et de son évaluation

Y : variable cible (target)

X1, X2, … : variables explicatives (clump, …, mitoses)

f(.) une fonction qui essaie d'établir la relation Y = f(X1, X2, …)

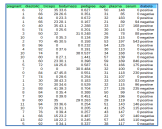
f(.) doit être « aussi précise que possible »…



pregnant	diastolic	triceps	bodymass	pedigree	age	plasma	serum	diabete
6	72	35	33.6	0.627	50	148	0	positive
1	66	29	26.6	0.351	31	85	0	negative
8	64	0	23.3	0.672	32	183	0	positive
1	66	23	28.1	0.167	21	89	94	negative
0	40	35	43.1	2.288	33	137	168	positive
5	74	0	25.6	0.201	30	116	0	negative
3	50	32	31.0	0.248	26	78	88	positive
10	0	0	35.3	0.134	29	115	0	negative
2	70	45	30.5	0.158	53	197	543	positive
8	96	0	0	0.232	54	125	0	positive
4	92	0	37.6	0.191	30	110	0	negative
10	74	0	38	0.537	34	168	0	positive
10	80	0	27.1	1.441	57	139	0	negative
1	60	23	30.1	0.398	59	189	846	positive
5	72	19	25.8	0.587	51	166	175	positive
7	0	0	0	30	0.484	32	100	0	positive
0	84	47	45.8	0.551	31	118	230	positive
7	74	0	29.6	0.254	31	107	0	positive
1	30	38	43.3	0.183	33	103	83	negative
1	70	30	34.6	0.529	32	115	96	positive
3	88	41	39.3	0.704	27	126	235	negative
8	84	0	35.4	0.388	50	99	0	negative
7	90	0	39.8	0.451	41	196	0	positive
9	80	35	29	0.263	29	119	0	positive
11	94	33	36.6	0.254	51	143	146	positive
10	70	26	31.1	0.205	41	125	115	positive
7	76	0	39.4	0.257	43	147	0	positive
1	66	15	23.2	0.487	22	97	140	negative
13	82	19	22.2	0.245	57	145	110	negative
5	92	0	34.1	0.337	38	117	0	negative

Ensemble de données (dataset)

Ensemble d'apprentissage

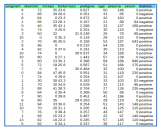


Construction de la fonction f(.) à partir des données d'apprentissage

$$Y = f(X1,X2,\ldots) + \varepsilon$$

Application du modèle (prédiction) sur l'ensemble de test

Ensemble de test



$$(Y, \hat{Y})$$

Y : valeurs observées

Y^ : valeurs prédites par f(.)

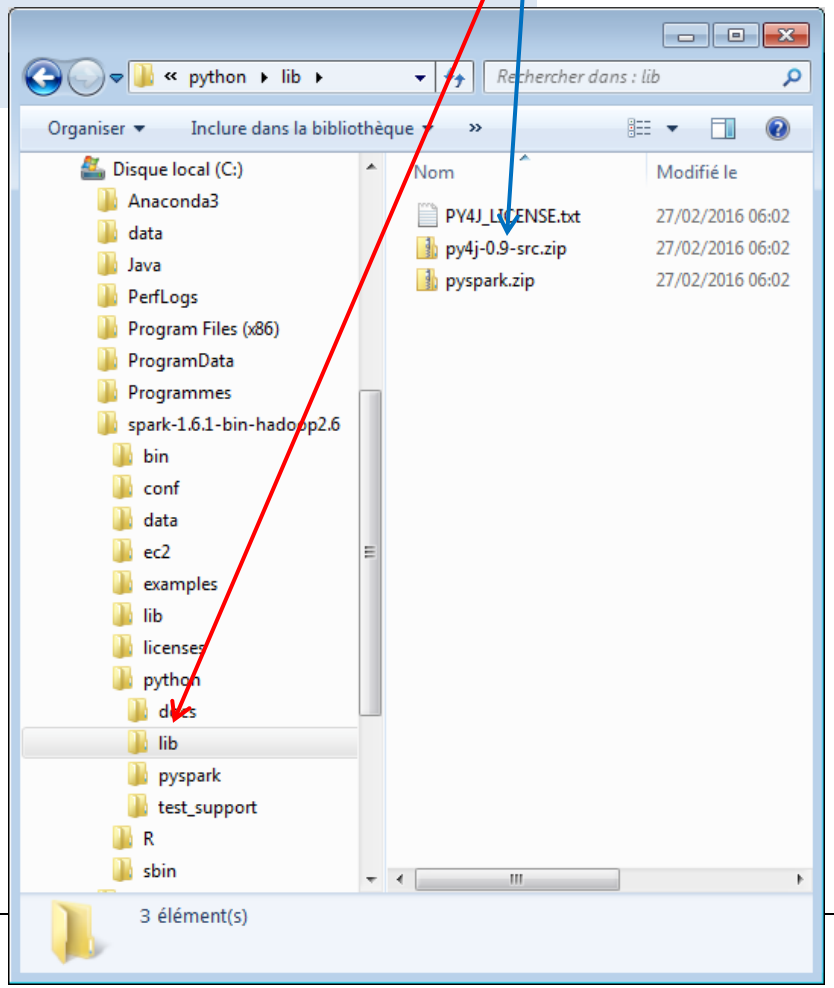
Mesures de **performances** par confrontation entre Y et Y^ : matrice de confusion + mesures

Etape 1 : Modification des chemins d'accès aux packages

```
# modification de la var. système path indiquant les fichiers des librairies
import sys
#vérifier les chemins d'accès
print(sys.path)
#modifier
sys.path.append('C:\\spark-1.6.1-bin-hadoop2.6\\python\\lib\\pyspark.zip')
sys.path.append('C:\\spark-1.6.1-bin-hadoop2.6\\python\\lib\\py4j-0.9-src.zip')
#re-vérifier
print(sys.path)
```

Il faut indiquer à Python l'emplacement du package PySpark

```
#1er affichage (print)
['', 'C:\\Anaconda3\\python35.zip', 'C:\\Anaconda3\\DLLs',
'C:\\Anaconda3\\lib', 'C:\\Anaconda3', 'c:\\anaconda3\\lib\\site-
packages\\setuptools-20.3-py3.5.egg', 'C:\\Anaconda3\\lib\\site-
packages', 'C:\\Anaconda3\\lib\\site-packages\\Sphinx-1.3.5-
py3.5.egg', 'C:\\Anaconda3\\lib\\site-packages\\win32',
'C:\\Anaconda3\\lib\\site-packages\\win32\\lib',
'C:\\Anaconda3\\lib\\site-packages\\Pythonwin']
#2nd affichage (print)
['', 'C:\\Anaconda3\\python35.zip', 'C:\\Anaconda3\\DLLs',
'C:\\Anaconda3\\lib', 'C:\\Anaconda3', 'c:\\anaconda3\\lib\\site-
packages\\setuptools-20.3-py3.5.egg', 'C:\\Anaconda3\\lib\\site-
packages', 'C:\\Anaconda3\\lib\\site-packages\\Sphinx-1.3.5-
py3.5.egg', 'C:\\Anaconda3\\lib\\site-packages\\win32',
'C:\\Anaconda3\\lib\\site-packages\\win32\\lib',
'C:\\Anaconda3\\lib\\site-packages\\Pythonwin', 'C:\\spark-1.6.1-
bin-hadoop2.6\\python\\lib\\pyspark.zip', 'C:\\spark-1.6.1-bin-
hadoop2.6\\python\\lib\\py4j-0.9-src.zip']
```



Etape 2 : Initialisation des « contextes »

```
# importation des classes SparkContext et SQLContext
# de la librairie pyspark
from pyspark import SparkContext
from pyspark.sql import SQLContext

# initialisation  d'un SparkContext
# moteur spark démarré sur une machine locale
# point d'entrée sur les fonctionnalités Spark
sc = SparkContext('local','MyExample')

# initialisation d'un SQL Context
# point d'entrée pour la manipulation des DataFrame
# et des fonctionnalités SQL
sqlC = SQLContext(sc)
```

Très verbeux, ces informations ne sont réellement intéressantes que pour tracer les erreurs.

```
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
16/05/31 18:36:45 INFO SparkContext: Running Spark version 1.6.1
16/05/31 18:36:45 WARN NativeCodeLoader: Unable to load native-hadoop
library for your platform... using builtin-java classes where applicable
16/05/31 18:36:45 INFO SecurityManager: Changing view acls to: Marjorie
16/05/31 18:36:45 INFO SecurityManager: Changing modify acls to: Marjorie
16/05/31 18:36:45 INFO SecurityManager: SecurityManager: authentication
disabled; ui acls disabled; users with view permissions: Set(Marjorie); users
with modify permissions: Set(Marjorie)
16/05/31 18:36:46 INFO Utils: Successfully started service 'sparkDriver' on port
58273.
16/05/31 18:36:46 INFO Slf4jLogger: Slf4jLogger started
16/05/31 18:36:46 INFO Remoting: Starting remoting
16/05/31 18:36:47 INFO Utils: Successfully started service
'sparkDriverActorSystem' on port 58290.
16/05/31 18:36:47 INFO SparkEnv: Registering MapOutputTracker
16/05/31 18:36:47 INFO SparkEnv: Registering BlockManagerMaster
16/05/31 18:36:47 INFO DiskBlockManager: Created local directory at
C:\Users\Marjorie\AppData\Local\Temp\blockmgr-ddf5818d-3743-42dd-b41e-
82b753af5131
16/05/31 18:36:47 INFO Remoting: Remoting started; listening on addresses
:[akka.tcp://sparkDriverActorSystem@10.0.2.15:58290]
16/05/31 18:36:47 INFO MemoryStore: MemoryStore started with capacity
517.4 MB
16/05/31 18:36:47 INFO SparkEnv: Registering OutputCommitCoordinator
16/05/31 18:36:47 INFO Utils: Successfully started service 'SparkUI' on port
4040.
16/05/31 18:36:47 INFO SparkUI: Started SparkUI at http://10.0.2.15:4040
16/05/31 18:36:47 INFO Executor: Starting executor ID driver on host localhost
16/05/31 18:36:47 INFO Utils: Successfully started service
'org.apache.spark.network.netty.NettyBlockTransferService' on port 58298.
16/05/31 18:36:47 INFO NettyBlockTransferService: Server created on 58298
16/05/31 18:36:47 INFO BlockManagerMaster: Trying to register BlockManager
16/05/31 18:36:47 INFO BlockManagerMasterEndpoint: Registering block
manager localhost:58298 with 517.4 MB RAM, BlockManagerId(driver,
localhost, 58298)
16/05/31 18:36:47 INFO BlockManagerMaster: Registered BlockManager
```

Etape 3 : Chargement des données

```
# importation des données avec la librairie pandas
import pandas
pdBreast = pandas.read_table("c:\\data\\breast_bin.csv",sep="\t",header=0)
print(pdBreast.shape) # (699, 10)
# transformation en un DataFrame Spark
dfBreast = sqlC.createDataFrame(pdBreast)
# affichage du schéma
dfBreast.printSchema()
# affichage des 5 premières observations
dfBreast.show(5)
# nombre d'observations
dfBreast.count() # 699
```

Il est possible d'importer directement au format CSV, mais passer par la librairie « pandas » avant de convertir au format DataFrame de Spark facilite l'opération et réduit considérablement les risques d'erreurs.

```
# affichage du schéma
root
 |-- clump: long (nullable = true)
 |-- ucellsize: long (nullable = true)
 |-- ucellshape: long (nullable = true)
 |-- mgadhesion: long (nullable = true)
 |-- sepics: long (nullable = true)
 |-- bnuclei: long (nullable = true)
 |-- bchromatin: long (nullable = true)
 |-- normnucl: long (nullable = true)
 |-- mitoses: long (nullable = true)
 |-- target: long (nullable = true)
```

```
# affichage des 5 premières lignes (observations)
+-----+---------+----------+----------+------+-------+----------+--------+-------+------+
|clump|ucellsize|ucellshape|mgadhesion|sepics|bnuclei|bchromatin|normnucl|mitoses|target|
+-----+---------+----------+----------+------+-------+----------+--------+-------+------+
|    4|        2|         2|         1|     2|      1|         2|       1|      1|     0|
|    1|        1|         1|         1|     2|      1|         2|       1|      1|     0|
|    2|        1|         1|         1|     2|      1|         2|       1|      1|     0|
|   10|        6|         6|         2|     4|     10|         9|       7|      1|     1|
|    4|        1|         1|         1|     2|      1|         2|       1|      1|     0|
+-----+---------+----------+----------+------+-------+----------+--------+-------+------+
```

''target'' est la variable cible qualitative, c'est un entier codé 0/1.

Etape 4 : Subdivision des données en échantillons d'apprentissage et de test

Pour disposer d'une mesure honnête des performances du modèle dans la population, il faut l'évaluer sur un échantillon qui n'a pas pris part à sa construction. Habituellement, on scinde en 2 échantillons les données disponibles : le premier sert à l'apprentissage, le second (test) sert à l'évaluation.

```
# le type LabeledPoint est nécessaire à l'apprentissage supervisé
# il distingue 'label', variable cible ; 'features' vecteur des descripteurs
from pyspark.mllib.regression import LabeledPoint

# Transformation des données d'apprentissage
# Rôle de la méthode map : Return a new RDD by applying a function to each Row
# colonne n°9 : target ; colonnes 0 à 8 (9 non-inclus) : descripteurs
lblBreast = dfBreast.map(lambda line:LabeledPoint(line[9],line[0:9]))

#affichage des 5 premières lignes
lblBreast.take(5)

# partition aléatoire en apprentissage (67% ≈ 2/3) et test (33% ≈ 1/3)
lblTrain, lblTest = lblBreast.randomSplit([0.67,0.33],1)

# nombre d'observations en apprentissage
lblTrain.count() # 462
# et en test
lblTest.count() # 237
```

```
# noter la structure LabeledPoint
# (label, [features])
[LabeledPoint(0.0,
[4.0,2.0,2.0,1.0,2.0,1.0,2.0,1.0,1.0]),
LabeledPoint(0.0,
[1.0,1.0,1.0,1.0,2.0,1.0,2.0,1.0,1.0]),
LabeledPoint(0.0,
[2.0,1.0,1.0,1.0,2.0,1.0,2.0,1.0,1.0]),
LabeledPoint(1.0,
[10.0,6.0,6.0,2.0,4.0,10.0,9.0,7.0,1.0])
, LabeledPoint(0.0,
[4.0,1.0,1.0,1.0,2.0,1.0,2.0,1.0,1.0])]
```

Etape 5 : Modélisation et affichage des résultats – Régression logistique

```
# importation de la régression logistique provenant de MLlib
from pyspark.mllib.classification import LogisticRegressionWithLBFGS
# construction du modèle prédictif sur l'échantillon d'apprentissage
modele = LogisticRegressionWithLBFGS.train(lblTrain, iterations=10, intercept=True)
# affichage du modèle
print(modele)
# ou encore, des coefficients du modèle
modele._coeff # DenseVector([0.0786, 0.2752, 0.1784, 0.1479, -0.1906, 0.313, -0.056, 0.2057, 0.0306])
# et de la constante
modele.intercept # -3.4088611629288357
```

```
# print(modele)
(weights=[0.0786109235948,0.275231120174,0.178366308716,0.147893015801,-
0.19055502104,0.313034229773,-0.0560224481539,0.205707149356,0.0306275090908],
intercept=-3.4088611629288357)
```

Avec dir(modele), nous avons accès à la liste des propriétés et méthodes de l'objet.

Etape 6 : Prédiction sur l'échantillon test

La prédiction ''pred'' est directement codée 0 (begnin) et 1 (malignant).

```
# prédiction - en entrée les descripteurs (''features'') de l'échantillon test
pred = modele.predict(lblTest.map(lambda x:x.features))

# affichage des prédictions pour les 10 premiers individus
pred.take(10) # [0, 0, 0, 0, 0, 0, 0, 0, 1, 0]

# concaténer (zip) les prédictions et les étiquettes dans la même structure
# les prédictions 0/1, valeurs entières, ont été converties en flottant (float)
LabelsAndPredictions = lblTest.map(lambda x:x.label).zip(pred.map(lambda x:float(x)))

# nombre d'observations
LabelsAndPredictions.count() # 237

# affichage des 10 premières valeurs
LabelsAndPredictions.take(10)
# [(0.0, 0.0), (0.0, 0.0), (0.0, 0.0), (0.0, 0.0), (0.0, 0.0), (0.0, 0.0), (0.0, 0.0),
(0.0, 0.0), (1.0, 1.0), (0.0, 0.0)]
```

La colonne ''label'' (Y) de l'échantillon test a été accolée à la prédiction ''pred'' (Y^) pour former un nouvel ensemble de données. On observe que les 10 premières prédictions sur l'échantillon test sont correctes, mais le sont-elles toutes ?

Etape 7 : Matrice de confusion et mesures des performances

Contrairement à R (SparkR), nous disposons d'outils dédiés pour mesurer de différentes manières les performances sur l'échantillon test.

```
# importation de l'outil pour l'évaluation des classifieurs multiclasses
from pyspark.mllib.evaluation import MulticlassMetrics

# instanciation
metrics = MulticlassMetrics(LabelsAndPredictions)

# affichage de la matrice de confusion
metrics.confusionMatrix().toArray()

# rappel – modalité 0.0 = begnin
metrics.recall(0.0) # 0.987 = 153 / (153 + 2)

# precision - modalité 0.0 = begnin
metrics.precision(0.0) # 0.950 = 153 / (153 + 8)

# arrêt du contexte Spark – ne pas oublier !
sc.stop()
```

```
Matrice de confusion
array([[ 153.,    2.],
       [   8.,   74.]])
```

Références

Références

Machine Learning Library (MLlib) Guide

Liste des méthodes de Machine Learning disponibles dans MLlib.

Welcome to Spark Python API Docs!

Listes des classes / méthodes disponibles dans PySpark.

Lucas Allen, « Spark Dataframes and MLlib », Tech Powered Math, August 2015.

Tutoriel : importation directe d'un fichier CSV, parsage des données, transformation des variables, et mise en œuvre de la régression linéaire de MLlib.

Jose A. Dianes, « MLlib: Classification with Logistic Regression » (1, 2), July 2015.

Traitement des données « Challenge KDD Cup 1999 » avec la régression logistique. Grosse volumétrie (pour nos PC tout du moins) avec 4898431 individus en apprentissage.