Artículo
Niyaz Khafizov · 3 ago, 2018 Lectura de 4 min

The way to launch Jupyter Notebook + Apache Spark + InterSystems IRIS

Hi all. Today we are going to install Jupyter Notebook and connect it to Apache Spark and InterSystems IRIS.

Note: I have done the following on Ubuntu 18.04, Python 3.6.5.

Introduction

If you are looking for well-known, widely-spread and mainly popular among Python users notebook instead of **Apache Zeppelin**, you should choose **Jupyter notebook. Jupyter notebook** is a very powerful and great data science tool. it has a big community and a lot of additional software and integrations. Jupyter notebook allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. And most importantly, it is a big community that will help you solve the problems you face.

Check requirements

If something doesn't work, look at the "Possible problems and solutions" paragraph in the bottom.

First of all, ensure that you have Java 8 (**java -version** returns "1.8.x"). Next, download apache spark and unzip it. After, run the following in the terminal:

```
pip3 install jupyter

pip3 install toree

jupyter toree install --sparkhome=/pathtospark/spark-2.3.1-bin-hadoop2.7 --interpreters=PySpark --user
```

Now, open the terminal and run **vim ~/.bashrc** . Paste in the bottom the following code (this is environment variables):

```
export JAVAHOME=/usr/lib/jvm/  installed java 8
export PATH="$PATH:$JAVAHOME/bin"
export SPARKHOME=/  path to spark/spark-2.3.1-bin-hadoop2.7
export PATH="$PATH:$SPARKHOME/bin"
export PYSPARKDRIVERPYTHON=jupyter
export PYSPARKDRIVERPYTHONOPTS="notebook"
```

```
File  Edit  View  Search  Terminal  Help
 1 .bashrc +
 # this, if it's already enabled in /etc/bash.bashrc and /etc/profile
 # sources /etc/bash.bashrc).
 if ! shopt -oq posix; then
   if [ -f /usr/share/bash-completion/bash_completion ]; then
     . /usr/share/bash-completion/bash_completion
   elif [ -f /etc/bash_completion ]; then
     . /etc/bash_completion
   fi
 fi

 #my variables

 export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
 export PATH="$PATH:$JAVA_HOME/bin"
 export SPARK_HOME=/home/guardian/Desktop/spark-2.3.1-bin-hadoop2.7
 export PATH="$PATH:$SPARK_HOME/bin"
 export PYSPARK_DRIVER_PYTHON=jupyter
 export PYSPARK_DRIVER_PYTHON_OPTS="notebook"
 ~
 ~
 NORMAL  .bashrc  +                                          96%  1
```

And run **source /.bashrc** .

Check that it works

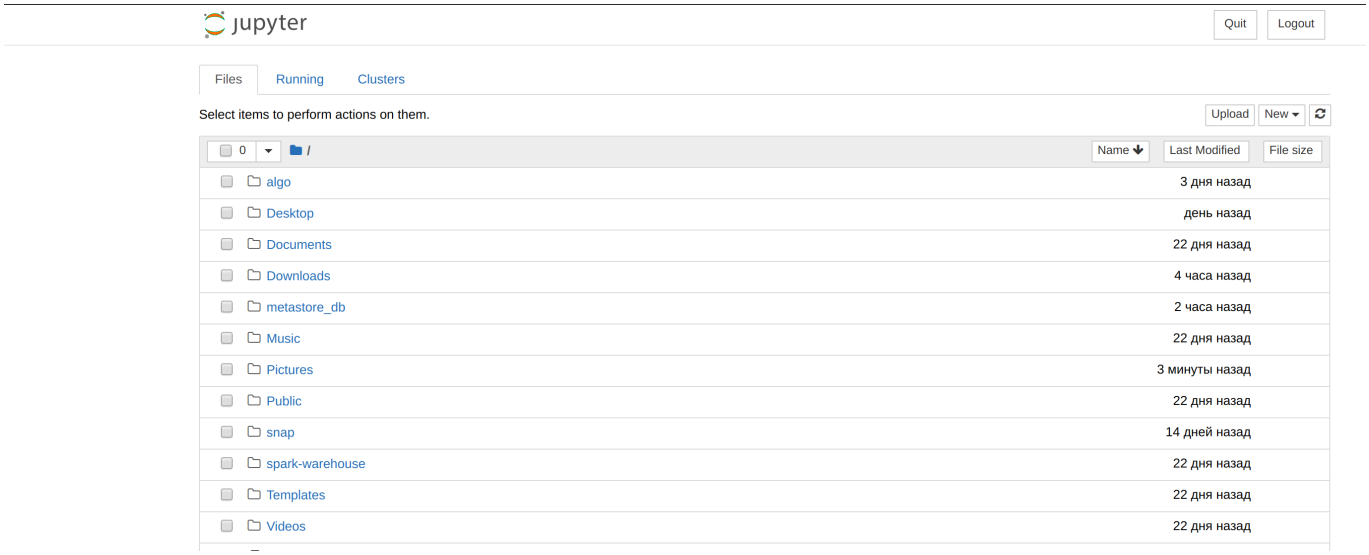
Now, let us launch Jupyter notebook. Run **pyspark** in the terminal.

```
File  Edit  View  Search  Terminal  Help

guardian@guardian:~$ pyspark
[I 16:07:42.424 NotebookApp] Serving notebooks from local directory: /home/guard
ian
[I 16:07:42.424 NotebookApp] The Jupyter Notebook is running at:
[I 16:07:42.424 NotebookApp] http://localhost:8888/?token=4d74c58c87b45d7cf2f556
73bbe523a9054c7a574acf3254
[I 16:07:42.424 NotebookApp] Use Control-C to stop this server and shut down all
 kernels (twice to skip confirmation).
[C 16:07:42.425 NotebookApp]

    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        http://localhost:8888/?token=4d74c58c87b45d7cf2f55673bbe523a9054c7a574ac
f3254
[I 16:07:43.174 NotebookApp] Accepting one-time-token-authenticated connection f
rom 127.0.0.1
```

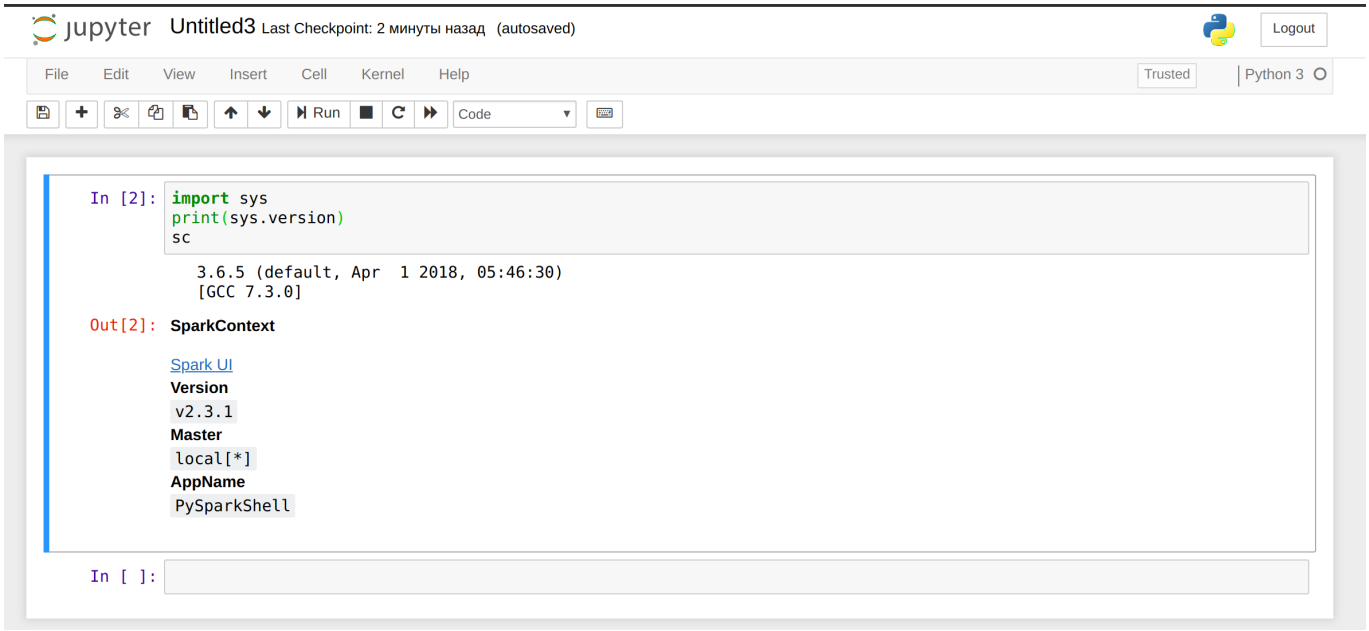
Open in your browser the returned URL. Should be something like the image below:



Click on **new**, choose **Python 3**, paste the following code into a paragraph:

```
import sys
print(sys.version)
sc
```

Your output should look like this:



Stop jupyter using ctrl-c in the terminal.

Note: To add custom jars just move desired jars into $SPARKHOME/jars.

So, we want to work with intersystems-jdbc and intersystems-spark (we will also need a jpmml library). Let us copy required jars into spark. Run the following in the terminal:

```
sudo cp /path to intersystems iris/dev/java/lib/JDK18/intersystems-jdbc-3.0.0.jar /path to spark/spark-2.3.1-bin-hadoop2.7/jars
```

sudo cp /path to intersystems iris/dev/java/lib/JDK18/intersystems-spark-1.0.0.jar /path to spark/spark-2.3.1-bin-hadoop2.7/jars

sudo cp /path to jpmml/jpmml-sparkml-executable-version.jar /path to spark/spark-2.3.1-bin-hadoop2.7/jars

Ensure that is works. Run **pyspark** in the terminal again and run the following code (from the previous article):

```
from pyspark.ml.linalg import Vectors
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.clustering import KMeans
from pyspark.ml import Pipeline
from pyspark.ml.feature import RFormula
from pyspark2pmml import PMMLBuilder

dataFrame=spark.read.format("com.intersystems.spark").\
option("url", "IRIS://localhost:51773/NAMESPACE").option("user", "dev").\
option("password", "123").\
option("dbtable", "DataMining.IrisDataset").load() # load iris dataset

(trainingData, testData) = dataFrame.randomSplit([0.7, 0.3]) # split the data into two sets
assembler = VectorAssembler(inputCols = ["PetalLength", "PetalWidth", "SepalLength", "SepalWidth"],
outputCol="features") # add a new column with features

kmeans = KMeans().setK(3).setSeed(2000) # clustering algorithm that we use

pipeline = Pipeline(stages=[assembler, kmeans]) # First, passed data will run against assembler and after
will run against kmeans.
modelKMeans = pipeline.fit(trainingData) # pass training data

pmmlBuilder = PMMLBuilder(sc, dataFrame, modelKMeans)
pmmlBuilder.buildFile("KMeans.pmml") # create pmml model
```

My output:

```
In [2]: from pyspark.ml.linalg import Vectors
        from pyspark.ml.feature import VectorAssembler
        from pyspark.ml.clustering import KMeans
        from pyspark.ml import Pipeline
        from pyspark.ml.feature import RFormula
        from pyspark2pmml import PMMLBuilder

        dataFrame=spark.read.format("com.intersystems.spark").\
        option("url", "IRIS://localhost:51773/NEWSAMPLE").option("user", "dev").\
        option("password", "123").\
        option("dbtable", "DataMining.IrisDataset").load() # load iris dataset

        (trainingData, testData) = dataFrame.randomSplit([0.7, 0.3]) # split the data into two sets
        assembler = VectorAssembler(inputCols = ["PetalLength", "PetalWidth", "SepalLength", "SepalWidth"], outputCol="feature

        kmeans = KMeans().setK(3).setSeed(2000) # clustering algorithm that we use

        pipeline = Pipeline(stages=[assembler, kmeans]) # First, passed data will run against assembler and after will run aga
        modelKMeans = pipeline.fit(trainingData) # pass training data

        pmmlBuilder = PMMLBuilder(sc, dataFrame, modelKMeans)
        pmmlBuilder.buildFile("KMeans.pmml") # create pmml model

Out[2]: '/home/guardian/KMeans.pmml'
```

The output file is a jpmml kmeans model. Everything works!

Possible problems and solutions

- **command not found: 'jupyter':**

1. vim /.bashrc;
2. add in the bottom **export PATH="$PATH:/.local/bin"** ;
3. in terminal source /.bashrc.
4. If it doesn't help, reinstall pip3 and jupyter.

- **env: 'jupyter': No such file or directory:**

1. In /.bashrc export PYSPARK_DRIVER_PYTHON=/home/.../.local/bin/jupyter.

- **TypeError: 'JavaPackage' object is not callable:**

1. Check that the required .jar file in /.../spark-2.3.1-bin-hadoop2.7/jars;
2. Restart notebook.

- **Java gateway process exited before sending the driver its port number:**

1. Your Java version should be 8 (probably works with Java 6/7 too, but I didn't check it);
2. **echo $JAVA_HOME** should return to you Java 8 version. If not, change the path in /.bashrc;
3. Paste **sudo update-alternatives --config java** in the terminal and choose a proper java version;
4. Paste **sudo update-alternatives --config javac** in the terminal and choose a proper java version.

- **PermissionError: [Errno 13] Permission denied: '/usr/local/share/jupyter'**

1. Add --user at the end of your command in the terminal

- **Error executing Jupyter command 'toree': [Errno 2] No such file or directory**

1. Run the command without sudo.

- A specific error may appear if you use system variables like **PYSPARK_SUBMIT_ARGS** and other spark/pyspark variables or because of /.../spark-2.3.1-bin-hadoop2.7/conf/spark-env.sh changes.

1. Delete these variables and check spark-env.sh.

Links

- **Jupyter**
- Apache Toree
- **Apache Spark**
- **Load a ML model into InterSystems IRIS**
- **K-Means clustering of the Iris Dataset**
- **The way to launch Apache Spark + Apache Zeppelin + InterSystems IRIS**

#Inteligencia Artificial #API #Mejores prácticas #Compatibilidad #Python #InterSystems IRIS

URL de fuente:https://es.community.intersystems.com/node/451281