

You have 1 free story left this month. [Sign up and get an extra one for free.](#)

# How to make XGBoost available in the Spark Notebook



Bogdan Cojocar [Follow](#)

Feb 7, 2018 · 2 min read ★



This is a step by step tutorial on how to install XGBoost (an efficient implementation of gradient boosting) on the Spark Notebook (tool for doing Apache Spark and Scala analysis and plotting graphs, similar to the Jupyter notebook).

If you don't have the Spark Notebook installed you can follow this quick guide.

## Step 1: Build XGBoost

For this step we need to clone the repository from github and build the project:

```
git clone --recursive https://github.com/dmlc/xgboost
```

Next we need to go into the newly cloned repository and build the project:

```
cd xgboost  
make -j4
```

For Mac users you have to do an additional step, before building with `make` :

```
cp make/config.mk ./config.mk
```

## Step 2: Create the XGBoost jar

Before we are able to build the jar we need to make sure we have `JAVA_HOME` set up and pointing to the JDK directory. We also need maven installed.

Next we need to run maven in the `xgboost` directory and publish the artifact on your local maven repository:

```
mvn install
```

This command will also run some tests, but we can skip that to make the installation faster. If you wish to do that simply run this command instead:

```
mvn -DskipTests install
```

### Step 3: Include the XGBoost jar into the Spark Notebook

As we now have the jar available in our local repository we can include it as a dependency in the notebook. In order to do that we need to go to the notebook and open the metadata window in the menu:

```
Edit -> Edit Notebook Metadata
```

A window will open containing a configuration JSON file. We will add the xgboost dependency in the `customDeps` property as shown in the screenshot below:

```
"customDeps": [  
  "ml.dmlc:xgboost4j-spark:0.8-SNAPSHOT"  
]
```

#### Edit Notebook Metadata



Manually edit the JSON below to manipulate the metadata for this Notebook. We recommend putting custom metadata attributes in an appropriately named sub-structure, so they don't conflict with those of others.

```
1 {  
2   "id": "1281640a-8a9e-4604-998f-96bc8c0cbc44",  
3   "name": "xgboost_tutorial",
```

```
4  "user_save_timestamp": "1970-01-01T01:00:00.000Z",
5  "auto_save_timestamp": "1970-01-01T01:00:00.000Z",
6  "language_info": {
7    "name": "scala",
8    "file_extension": "scala",
9    "codemirror_mode": "text/x-scala"
10 },
11 "trusted": true,
12 "sparkNotebook": null,
13 "customLocalRepo": null,
14 "customRepos": null,
15 "customDeps": [
16   "ml.dmlc:xgboost4j-spark:0.8-SNAPSHOT"
17 ],
18 "customImports": null,
19 "customArgs": null,
20 "customSparkConf": null,
21 "customVars": null,
22 "kernelSpec": {
23   "name": "spark",
```

OK

Cancel

Please note that at the moment this tutorial was created I was using

xgboost version 0.8.

The last step is to restart the kernel of the notebook and we should have the dependency available. To restart the kernel go in the menu at the kernel tab and run restart:

Kernel -> Restart

Now we should be able to import XGBoost:

```
import ml.dmlc.xgboost4j.scala.spark
import ml.dmlc.xgboost4j.scala.spark
```

Took: 0.814s, at 2018-02-07 17:16

When we add `import ml.dmlc.xgboost4j.scala.spark` into a cell of the notebook and we run the code (in the menu `Cell -> Run` ) it should run successfully.

Xgboost Machine Learning Apache Spark Scala Big Data



6 claps



WRITTEN BY

**Bogdan Cojocar**

Follow

Big data consultant. I write about the wonderful world of data.

If you enjoy my articles:

<https://www.buymeacoffee.com/paxdata>. Cheers!

## More From Medium

**How to run a real-time pipeline in AWS Kinesis using PySpark Structured Streaming**

Bogdan Cojocar in  
Towards Data Science



**Delta lake with Spark: What and Why?**

Jyoti Dhiman in Towards  
Data Science



**Predict Customer Churn using PySpark Machine Learning**

Marvin Lütke in Towards  
Data Science



**Logistic Regression with PySpark**

Gülcan Ögündür in The  
Startup



## PySpark Under the Hood: RandomSplit() and Sample() Inconsistencies Examined

Meltem Tutar in Udemey Tech Blog



## Binary Classifier Evaluation made easy with HandySpark

Daniel Godoy in Towards Data Science



## Deploy a Python model (more efficiently) over Spark

Schaun Wheeler in Towards Data Science



## Want to Master Your Data? Here's Why You Should Care About Metadata

Nam Nguyen in Towards Data Science



## Discover Medium

Welcome to a place where words matter. On Medium, smart voices and original ideas take center stage - with no ads in sight. [Watch](#)

## Make Medium yours

Follow all the topics you care about, and we'll deliver the best stories for you to your homepage and inbox. [Explore](#)

## Become a member

Get unlimited access to the best stories on Medium — and support writers while you're at it. Just \$5/month. [Upgrade](#)

---

# Medium

[About](#)

[Help](#)

[Legal](#)