Novel Approach To Setup Apache Spark And Python For Pyspark Programming To Count The Number of Telugu Words In Big Data

Alugolu Avinash
Asst.Professor
Centurion University of Technology and Management-Andhra Pradesh

ABSTRACT

Spark is a tool which is distributed in-memory computing framework which was developed at AmpLab, UCB. It's API is primarily programmed and implemented in scala programming language and then provides support for other programming languages like Java, Python, R are developed in later versions. Pyspark[1] is an API developed in python programming language for doing spark programming and writing spark applications in Python simple and easy style, but execution process is similar for all the above mentioned API programming languages. In this paper authors give complete idea about step by step setup process of pyspark in local windows machine.

Keywords Apache, Pyspark, Architecture, RDD, Dataset , Programming language, API.

Introduction Spark is a parallel distributing computing framework which is built from scala programming language to work on Big Data or Real time streaming data. It is from Apache Foundation. While Pyspark is an API of spark to work basically on Data Frames[2] on Spark framework. Python is the programming language which is mainly used to work on pyspark.In order to install pyspark in our local machine we need to gather some basic tools from online resources[3].

Procedure to Setup PySpark:
Here we have mentioned the process step by step.

Step1: Install Anaconda latest software in local Machine by downloading the setup file from the online website source www.anaconda.com/distribution. Download latest version which gives more opportunity to get extensive collection libraries[4]. Once installed, you have Anaconda including python 3 and juypter installed on your machine.

Step 2: Install Spark, to install spark framework on our machine the following three necessary steps need to be done. We can download our required version of spark framework from the online web resource, its corresponding link which is provided here http://spark.apache.org/downloads.html from this webpage chose required latest version of API and download. The *tgz file can be unpacked e.g. with the help of 7-zip software, which we can get from online source in the internet[5]. Before we unpack the *tgz create a new folder called spark in C: drive like C:/spark and Unpack the *tgz into C:/spark location, the target folder for the unpacking of the above file should be something like: spark-2.2.0-bin-hadoop2.7

Fig1: Online web source page to download spark

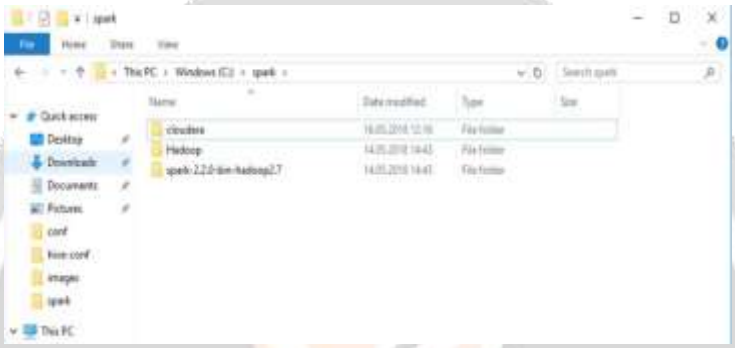

Fig:2 Installation location of Spark

Step3: After successful installation of anaconda and spark in local machine next we need to download winutilities.exe file from online resource since Sparkjobs which you can see at localhost:4040/jobs in yarn mode need access to HDFS (Hadoop Distributed File System), since temporary data keep on uploaded to HDFS into hdfs:///user/< userid >. Consequently our machine needs a hadoop client which can be downloaded directly from the public share platform github[6] ,we can get it here. https://github.com/steveloughran/winutils/tree/master/hadoop-2.7.1/bin.

Fig 3: Win utilities online source

Before proceeding further we must Create a new folder in C:/spark called Hadoop and a subfolder called bin C:/spark/Hadoop/bin and copy the downloaded winutils.exe tool into the bin folder.

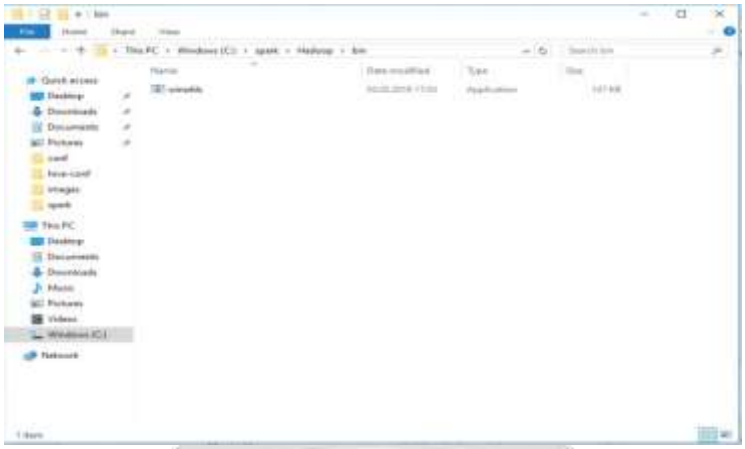

Fig 4 : Win Utilities storage location in local machine

Step 4: Download cloudera client config as per the hierarchy , In order to access the cluster the cloudera config needs to be downloaded[7]. To do this, you genrrally need to follow a link of this format: http:// *YOUR_CLOUDERA_MANAGER_IP* /cmf/services/10/client-config So, Consider the following example, if your IP is: myClouderaIP.com:1234, then your link will be:http://myClouderaIP.com:1234/cmf/services/10/client-config Download the zip and extract using some unzip software tool in a new subfolder from C:/spark called cloudera C:/spark/cloudera/

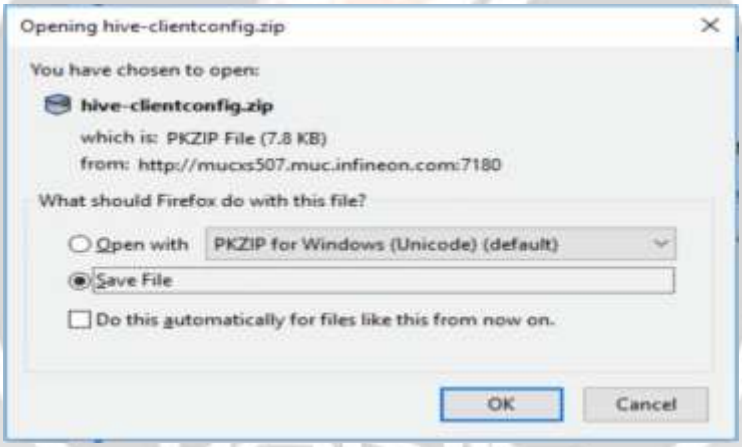

Fig5:Cloudera client config

Here, before we proceed for further procedure we need to notice Important thing that the files (*.xml and other) should be copied direct under the cloudera folder, no subfolder like hive-clientconfig should be there. Perform Copy operation to do copy the hive-site.xml the last step is to copy the hive-site.xml from C:/spark/cloudera/hive- site.xml to the location C:/spark/spark-2.2.0-bin-hadoop2.7/conf/hive-site.xml There is already existing one but we should replace with the new one.

Step5: Setting up environment variables in Windows machine, Since all needed sources/binaries and configs has been downloaded and saved in C:/spark a couple of environment variables need to be set in order to let jupyter know to use pyspark and how to access the cluster For spark the following environment variables need to be stein[8]. Follow this step wise process

- !setx SP_ROOT_HOME "C:\spark"
- !setx SPARK_HOME "C:\spark\spark-2.2.0-bin-hadoop2.7"
- !setx PYSPARK_PYTHON "/opt/anaconda3/bin/python3"
- !setx PYTHONPATH "C:\spark\spark-2.2.0-bin-hadoop2.7\python;C:\spark\spark-2.2.0-bin-hadoop2.7\python\lib;C:\spark\spark-2.2.0-bin-hadoop2.7\python\lib\py4j-0.10.4-src.zip;"
- !setx PYSPARK_LIBS "C:\spark\spark-2.2.0-bin-hadoop2.7\python\lib\pyspark.zip;"

- !setx HADOOP_CONF_DIR "C:\spark\cloudera;"
- !setx YARN_CONF_DIR "C:\spark\cloudera;"

Especially For the winutils.exe the following environment variables need to be setin

- !setx HADOOP_HOME "C:\spark\Hadoop"
- !setx PATH "%PATH%;%SPARK_HOME%\bin;%HADOOP_HOME%\bin"

Once the setin process is done you may see the following successful greeting message like"SUCCESS: Specified value was saved ".Once it is done next important point to be noticed is to Close the complete browser with juypter and the cmd running in the back and restart juypter via Anaconda again[9]. The environment variables are become active and triggered in juypter after the restart of tools.

Step6: Testing pyspark , In order to test pyspark functionality we have considered a small problem statement that counts number of telugu words in a document and display the total word count result on the interface[10].

Part-1 Code

Line1: **simplerdd=sc.textFile("telugu.txt")**

Explanation:

The above line reads the text document and creates RDD(Resilient Distributed Dataset) .

Line 2:

data=simplerdd.filter(lambda x:x!=' ').map(lambda word:(word,1))

Explanation: The second line filters all space characters and with the help of map function it creates group of tuples

In the format (word ,1) .

Line 3:

data.collect()

Explanation:

With the help of third line

We can collect the key-value pairs and display on screen.

Part-1-Output

[('
హలో
', 1),
(' హాయ్', 1),
(' ఎలా', 1),
(' ఉన్నారు', 1),
(' స్పార్క్', 1),
(' ఒక', 1),
(' అద్భుతం', 1),
(' నేర్చుకోవడం', 1),
(' చాలా', 1),
(' సులభం', 1),
(' హాయ్', 1),
(' ఎలా', 1),
(' ఉన్నారు', 1)]

Part-2 Code

Line 1:

data=simplerdd.filter(lambda x:x!=' ').map(lambda word:(word,1)).reduceByKey(lambda x,y:x+y)

Explanation:

Line 1 code reduces all key-value pair tuples using reducebykey function and produces total count of each word in the document.

Line 2:

data.collect()

With the help of second line we can collect the words and its count and display on screen.

Part-2 Output

[(' హలో ', 72),
('', 41),
(' హాయ్ ', 36),
(' ఎలా ', 72),
(' ఉన్నారు ', 72),
(' స్పార్క్ ', 36)
(' ఒక', 12),
(' అద్భుతం', 10),
(' నేర్చుకోవడం', 13),
(' చాలా', 31),
(' సులభం', 41)]

Conclusion

Apache Spark framework is a cluster computing platform designed for fast, speed side computing and extends the popular functionality MapReduce model to efficiently support and do more type of computations on real time data or big data, It also can do interactive query processing and stream processing[11]. As Spark integrates very closely with **big data** tool, Tight integration is the ability to build an application that seamlessly combines different computation model using ML lib[12][13].

References

1. Community effort driving standardization of Apache Spark through expanded role in Hadoop Project, Cloudera, Databricks, IBM, Intel,andMapR,OpenSourceStandards,http://finance.yahoo.com/news/communityeffortdrivingstandardizationapache162000526.html, Retrieved July 1 2014.
2. Big Data: what I is and why it mater, 2014,http://www.sas.com/en_us/insights/big-data/whatis-big-data.html
3. Nick Lewis, 2014, information security threat questions.
4. Michael Goldberg, 2012, Cloud Security Alliance Lists 10 Big data securityChallenges,http://datainformed.com/cloud-security-alliance-lists-10-bigdata-security-challenges/
5. Securosis, 2012, Securing Big Data: Security Recommendations for Hadoop and No SQL Environment,https://securosis.com/assets/library/reports/SecuringBigData_FINAL.pdf
6. Steve Hurst, 2013, To 10 Security Challenges for 2013, http://www.scmagazine.com/top-10-securitychallenges-for 2013/article/281519/
7. Mark Hoover, 2013, Do you know big data's top 9 challenges?,http://washingtontechnology.com/articles/2013/02/28/big-data-challenges.aspx
8. MarketWired,2014,http://www.marketwired.com/press-release/apache-spark-beats-the-world-recordforfastest processing-of-big-data-1956518.htm
9. R.B.Donkin,HadoopAndFriends,http://people.apache.org/~rdonkin/hadooptalk/hadoop.html , Retrieved May 2014.
10. Hadoop, Welcome to Apache Hadoop,http://hadoop.apache.org/, Retrieved May 2014.
11. Casey Stella, 2014, Spark for Data Science: A CaseStudy, http://hortonworks.com/blog/spark-datascience-case-study/
12. Abhi Basu, Real-Time Healthcare Analytics on ApacheHadoopusingSparkandShark,http://www.intel.com/content/dam/www/public/uen/documents/white-papers/big-data-real time health care-analyticswhite paper .pdf, Retrieved December 2014.
13. Spark MLib, Apache Spark performance,https://spark.apache.org/mllib/ , Retrieved October 2014.

BIOGRAPHY

	Mr. Alugolu Avinash is working as Asst. Professor at Centurion University of Technology and Management, Andhra Pradesh. I have 10 years of teaching experience and my research area is machine learning.