

Assignment 4

BIG DATA ANALYSIS

Competition

REPORT

TABLE OF CONTENTS

Team:	2
Software Used	2
Analytics Process	4
Challenges	7
Implementation on PySpark:	9
References	9

TEAM: = (3 MEMBERS)

Vidhan Dholakia (20151326)

SOFTWARE USED:

Microsoft Azure:

Cloud computing is when I can access computing services like servers, storage, networking, etc. over the Internet from a provider like Azure or AWS. In this assignment, I have used Microsoft Azure, which is a cloud computing platform. MS Azure is cost-effective, secure, reliable, flexible and durable as compared to the local or on-premises servers. I can directly access MS Azure using my authorized account. The advantage in such cloud-based platform is that I pay only for the things I use. Once I am done with the task, I must turn off the Virtual Machines and make sure every process working in the background is executed and completed. Failing to release resources in time would cost the user more every time.

Hadoop / MapReduce:

Apache Hadoop is an open source framework for distributed storage and processing of large data on commodity hardware. The base Apache Hadoop framework is composed of several modules including Hadoop Common, Hadoop Distributed File System (HDFS), Hadoop YARN (Yet Another Resource Negotiator), and Hadoop MapReduce. Hadoop MapReduce is a programming model for large scale data processing.

Spark (PySpark):

PySpark is the Python API for Spark. PySpark is a great language to learn in order to create more scalable analyses and pipelines. It is used for performing exploratory data analysis at scale, building machine pipelines, etc. PySpark fully supports interactive use i.e. can be easily run. `./bin/pyspark` to launch an interactive shell.

Installing and Configuration:

By default, PySpark requires python to be available on the system PATH and use it to run programs; an alternate Python executable may be specified by setting the `PYSPARK_PYTHON` environment variable in `conf/spark-env.sh` (or `.cmd` on Windows).

All of PySpark's library dependencies, including Py4J, are bundled with PySpark and automatically imported. Standalone PySpark applications should be run using the `bin/pyspark` script, which automatically configures the Java and Python environment using the settings in `conf/spark-env.sh` or `.cmd`. The script automatically adds the `bin/pyspark` package to the `PYTHONPATH`.

Hortonworks Data Platform Sandbox:

The HDP Sandbox makes it easy to get started with Apache Hadoop, Apache Spark, Apache Hive, Apache HBase, and Data Analytics Studio. The Hortonworks Sandbox is a single node implementation of HDP. It is packaged as a virtual machine to make evaluation and experimentation with HDP fast and easy. HDP is free to download and use in my enterprise and I can download it from <https://www.cloudera.com/downloads.html>.

How to Download the Data Set?

I have been given the list of both positive and negative words and also one sample data file. Other than this, I have a large data file to calculate the sentiment for each of the mentioned products.

ANALYTICS PROCESS:

Logging into Microsoft Azure to create Virtual Machine for Hortonworks Data Platform.

The screenshot shows the Microsoft Azure portal dashboard. At the top, there's a search bar and navigation icons. Below the header, the 'Azure services' section displays various service tiles like 'Create a resource', 'Virtual machines', 'App Services', 'Storage accounts', 'SQL databases', 'Azure Database for PostgreSQL', 'Azure Cosmos DB', 'Kubernetes services', and 'More services'. The 'Recent resources' section contains a table with columns 'NAME', 'TYPE', and 'LAST VIEWED'. The 'Navigate' section has icons for 'Subscriptions', 'Resource groups', 'All resources', and 'Dashboard'. The 'Tools' section includes links to 'Microsoft Learn', 'Azure Monitor', 'Security Center', and 'Cost Management'.

NAME	TYPE	LAST VIEWED
Free Trial	Subscription	30 min ago
HDP	Virtual machine	2 h ago
HDP-ip	Public IP address	3 h ago

Overview of the Hortonworks Data Platform virtual machine.

The screenshot displays the 'Overview' page for a virtual machine named 'HDP'. The left sidebar shows navigation options like 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', and various settings. The main content area shows VM details: Resource group (HDP), Status (Stopped), Location (Central US), Subscription (Free Trial), Subscription ID, Computer name, Operating system (Linux), Size (Standard B1s), and Tags. A table on the right lists network-related properties like Public IP address, Private IP address, and Virtual network/subnet. Below this, there are charts for 'CPU (average)' and 'Network (total)' usage over time, with a 'Show data for last:' selector set to '1 hour'.

Property	Value
Resource group (change)	HDP
Status	Stopped
Location	Central US
Subscription (change)	Free Trial
Subscription ID	7e5a840c-8c3a-44b3-965e-cd96b6ddd462
Computer name	(not available)
Operating system	Linux
Size	Standard B1s (1 vcpu, 1 GiB memory)
Tags (change)	Click here to add tags
Azure Spot	N/A
Public IP address	52.238.249.191
Private IP address	10.0.0.4
Public IP address (IPv6)	-
Private IP address (IPv6)	-
Virtual network/subnet	HDP-vnet/default
DNS name	Configure
Scale Set	N/A

Add extra ports for HDP and Ambari server to communicate with the user. The ports for Ambari server is 8080, for SSH connection is 2222, for Zeppelin notebook is 9995.

Microsoft Azure

Search resources, services, and docs (G+)

18vbd@queensu.ca
QUEEN'S UNIVERSITY

Home > HDP - Networking

HDP - Networking
Virtual machine

Search (Ctrl+J)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Settings

Networking

Disks

Size

Security

Extensions

Continuous delivery (Preview)

Availability + scaling

Configuration

Identity

Attach network interface Detach network interface

Network Interface: hdp357

Effective security rules

Topology

Virtual network/subnet: HDP-vnet/default NIC Public IP: 52.238.249.191 NIC Private IP: 10.0.0.4 Accelerated networking: Disabled

Inbound port rules

Outbound port rules

Application security groups

Load balancing

Network security group HDP-nsg (attached to network interface: hdp357)

Impacts 0 subnets, 1 network interfaces

Add inbound port rule

Priority	Name	Port	Protocol	Source	Destination	Action
300	SSH	22	TCP	Any	Any	Allow
320	HTTP	80	TCP	Any	Any	Allow
340	HTTPS	443	TCP	Any	Any	Allow
350	Port_8080	2000-50000	Any	Any	Any	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

Ambari Server loading with credentials as mentioned on Hortonworks Data Platform page.

← → ↺ Not secure | 52.238.251.186:8080/#/main/dashboard/metrics

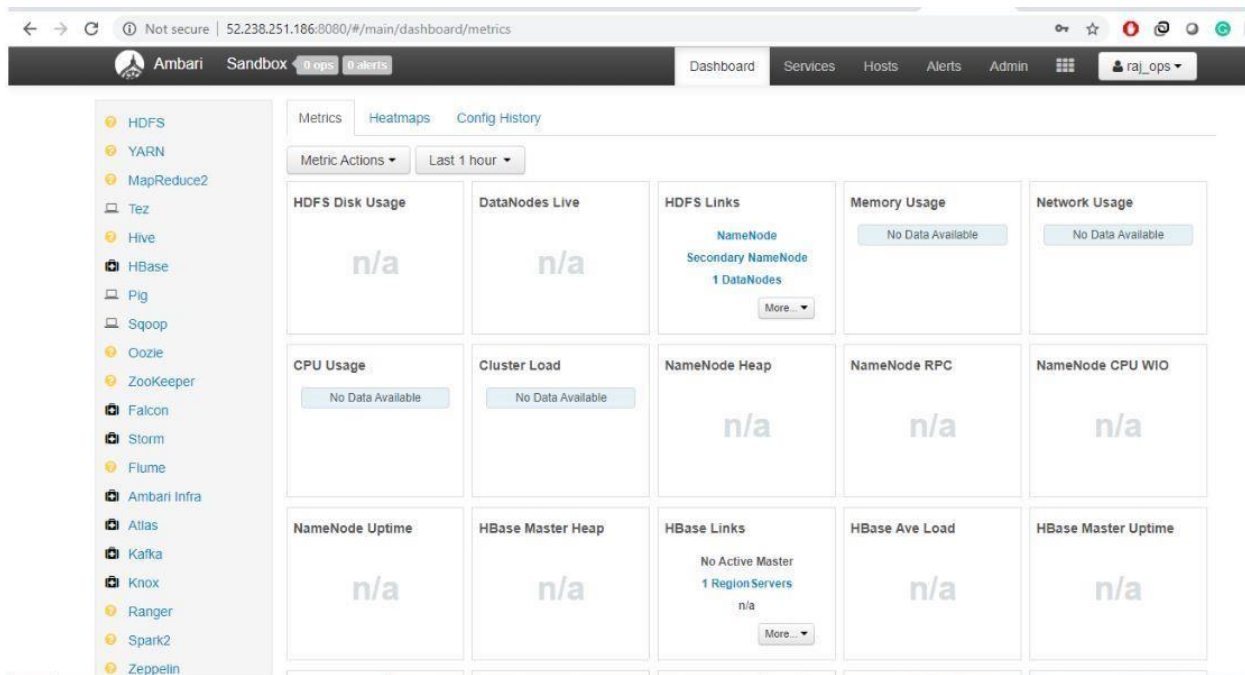
Ambari

raj_ops

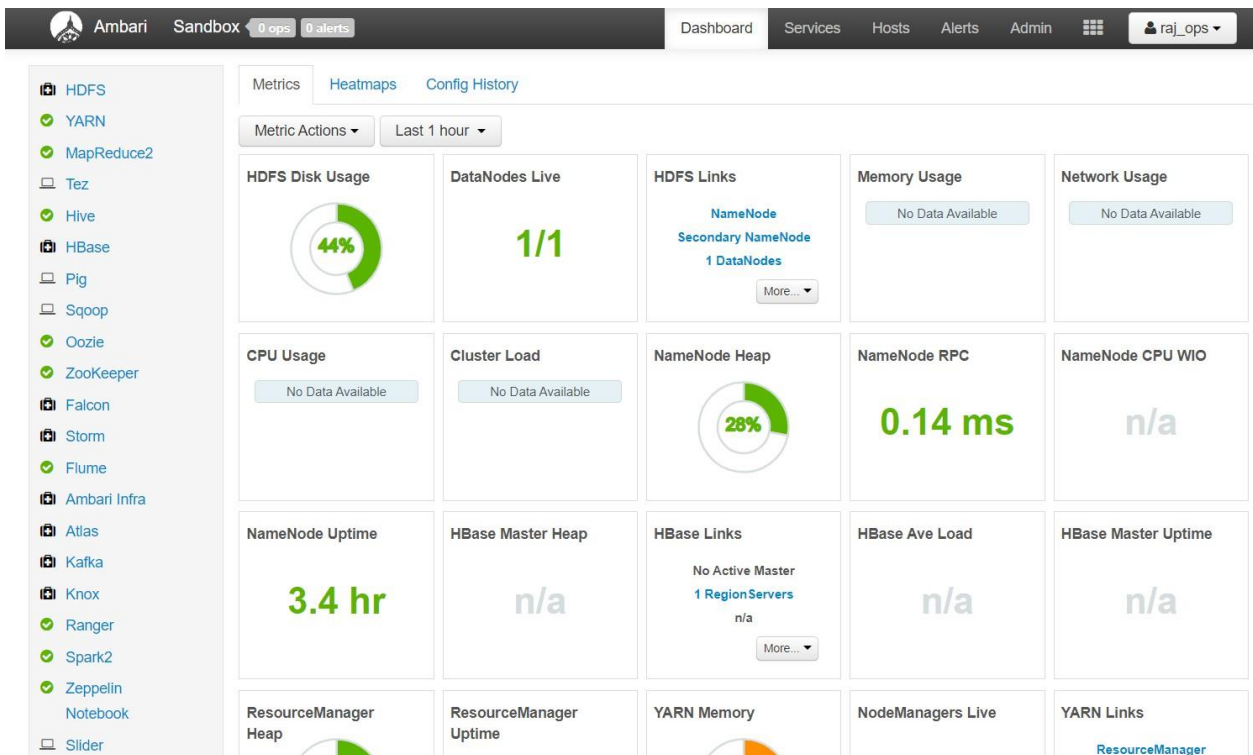
Loading...

Licensed under the Apache License, Version 2.0.
See third-party tools/resources that Ambari uses and their respective authors

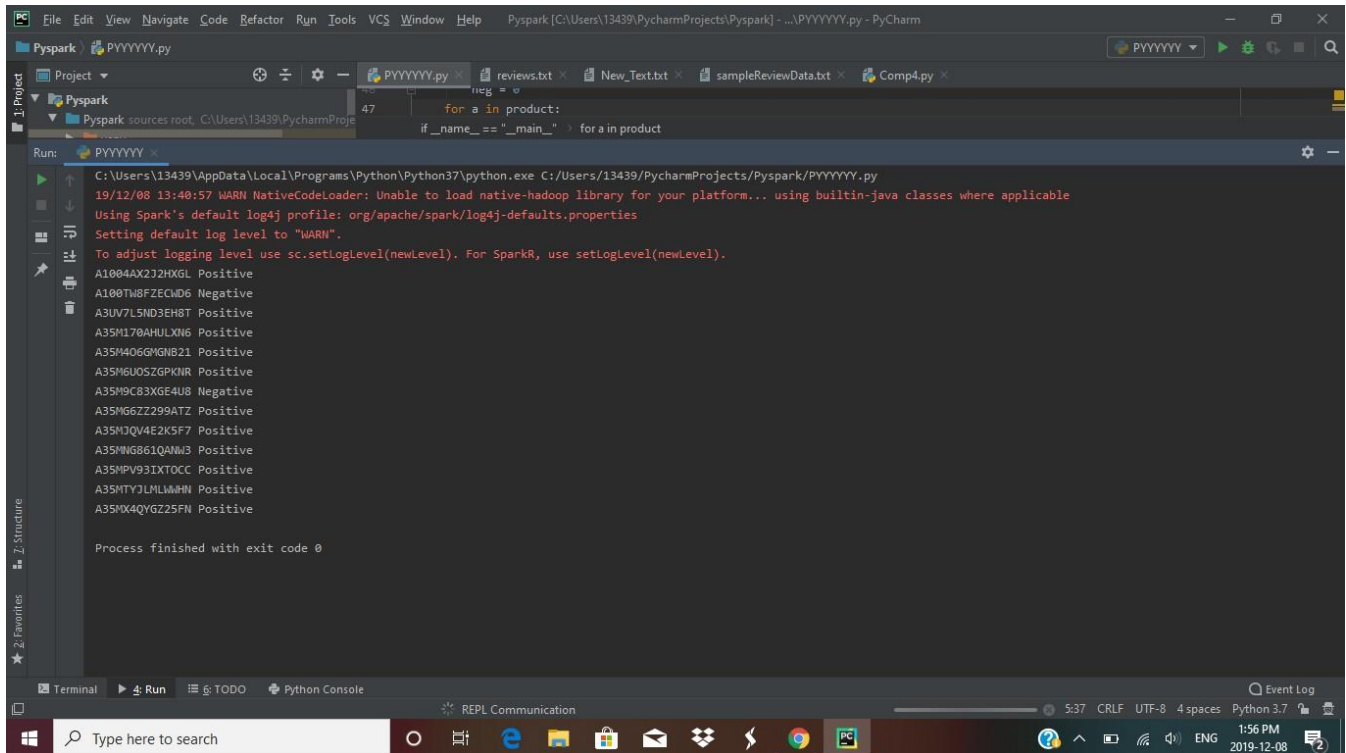
This is how Ambari server looks but the services aren't responding and hence I need to either restart all services or just the required services



With all services running, Ambari would look like this.



Pyspark output on local machine with set of data this is the output obtained for the code attached as a zip file in this competition



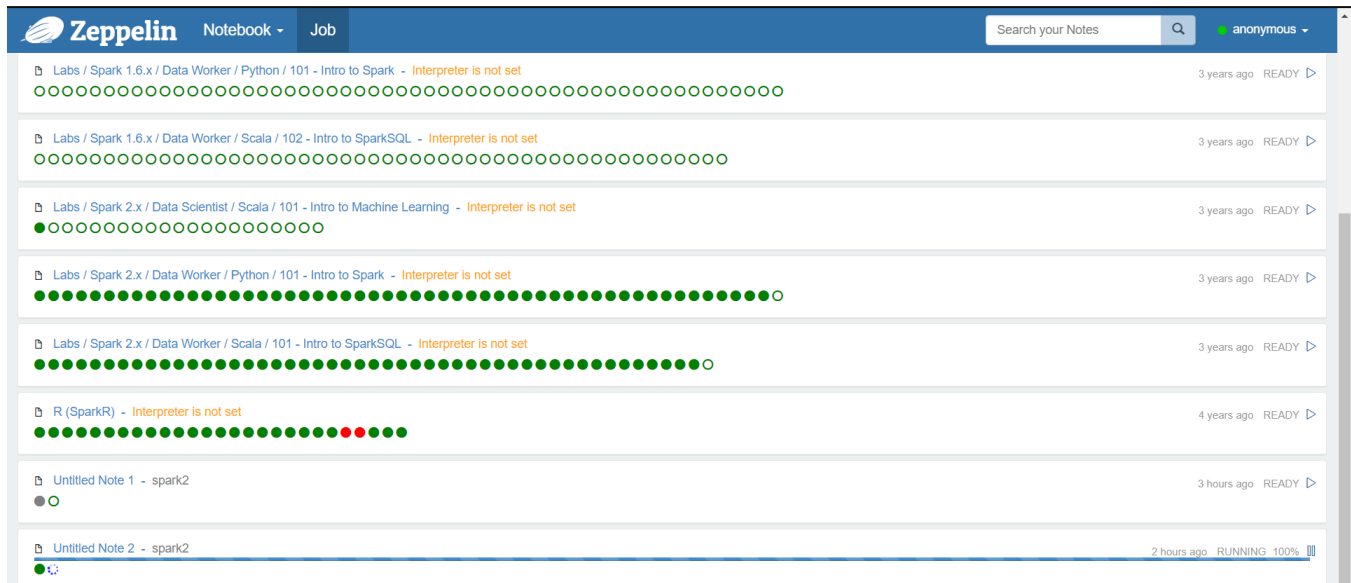
The screenshot shows the PyCharm IDE interface. The top toolbar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The project structure on the left shows a 'Pyspark' project with a 'Pyspark' sub-project containing 'sources root'. The main editor displays a Python file 'PYYYYYY.py' with a loop: `for a in product: if __name__ == '__main__': for a in product`. The Run console at the bottom shows the execution output: `C:\Users\13439\AppData\Local\Programs\Python\Python37\python.exe C:/Users/13439/PycharmProjects/Pyspark/PYYYYYY.py`, followed by a warning about the native-hadoop library, log settings, and a list of 15 data points with their sentiment labels (e.g., 'A1004AX22HXGL Positive', 'A100TH8FZECMD6 Negative'). The process finished with exit code 0. The bottom status bar shows '5:37 CRLF UTF-8 4 spaces Python 3.7' and the system clock '1:56 PM 2019-12-08'.

CHALLENGES:

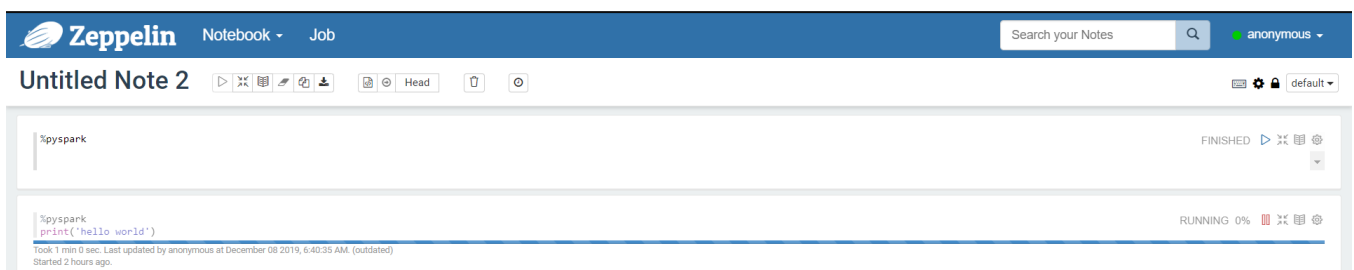
I came across many of the challenges as described:

- It was tough to set up the Microsoft Azure Virtual Machine which took most of the time in the competition.
- I started the work on this competition by using a physical computer to see what I are to deal with. I Ire able to get a desired output on the sample reviews.
- Large data was the problem to handle, uploading data onto cloud would take a lot of time.
- Azure virtual machine was not supportive enough with lack of documentation and very less time to pull the task I found it difficult to implement it on cloud.
- Availability of a lot of resmyces on the Azure cloud also creates problem as one should be aware of the choice of what resmyces should be used as it saves time and credits.

- Zeppelin Issues: The Azure lagged most of the time. As per the following screenshot, I'm having trouble to work on PySpark on Zeppelin. Despite having the PySpark interpreter, I got the following output (Untitled Note 2 running from past 2 hms and having no response):



Even a simple import function of PySpark I'm lagging a lot.



Another problem that I ran into was YARN out of memory error. I tried upgrading my storage as well as virtual memory but in vain. All services got stopped when the YARN memory gets filled and I had to restart my services which again takes around 1 hmy. Unfortunately, I had to complete my task with the memory problem.

IMPLEMENTATION ON PYSPARK:

- To run on local machine, I needed a *pyspark* setup first which requires *python* along with *JDK 8* and *winutils*.
- I implemented my code on *PyCharm IDE* with all the available libraries of PySpark, I did this as above screenshots show I couldn't implement it on Zeppeline as it wasn't responsive most of the times.
- The spark code works in a pretty simple way I have 3 text files as input data viz. *positive_data.txt*, *negative_data.txt* and *reviews.txt* file.
- I convert this input txt's to RDD with Spark Context and use them to carry out map-reduce.
- With help of this map-reduce technique I predicted if the review/sentiment from a comment was positive or negative. For I mapped number of positive and negatives for each tag. Based on the number, I decide if it's a positive review or negative. Since, I not having a neutral feedback, I had to use greater than and less than function to deduce the positive and negative feedback.

REFERENCES:

- [1] <https://www.python.org/doc/>
- [2] <https://docs.microsoft.com/en-us/azure/>
- [3] https://datasciencetoday.net/index.php/en-us/tutorials-top/163-sentiment-analysis_using-python-part-i-machine-learning-model-comparison
- [4] <http://saptak.in/spark/001-configuring-hortonwork-sandbox-azure.html>
- [5] <https://spark.apache.org/docs/latest/api/python/index.html>
- [6] <https://www.cloudera.com/tutorials/getting-started-with-hdp-sandbox/1.html>
- [7] <https://towardsdatascience.com/a-brief-introduction-to-pyspark-ff4284701873>