Spark based LINEAR REGRESSION Algorithm & Sample Linear Data set in HDFS Cluster

Hadoop multi node , Spark cluster

Presented to
Animesh Chaturvedi

Agenda

- Teammates
- About Hadoop
- Hadoop Setup
- About Spark
- Spark Setup
- Algorithm
- Executing using Spark on the HDFS cluster

Teammates

- J . Anil Kumar - 21BDS024
- R . Vinay Kumar - 21BDS056
- Sidharth Kaushik - 21BDS064

Hadoop:

Hadoop is a distributed computing platform that uses clusters of commodity hardware to store and process large amounts of data. It is based on the Hadoop Distributed File System (HDFS) and the MapReduce programming model. Hadoop enables scalable, reliable, and efficient processing of big data.

Single - Node :

Hadoop single node refers to running Hadoop on a single machine, rather than a cluster. It is useful for development, testing and learning purposes. Setting up Hadoop single node involves installing Hadoop and configuring it to run on a local machine, with all Hadoop components running on a single node.

Multi - Node:

Hadoop multi-node refers to a setup where multiple machines work together in a Hadoop cluster to store and process large datasets. This setup enables parallel processing, fault tolerance, and high availability of data. The multi-node configuration includes master and worker nodes, with each node performing specific roles in the Hadoop ecosystem.

HDFS Setup

Configure Hadoop :

Modify the configuration in following "XML" files :

- core-site.xml
- hdfs-site.xml
- yarn-site.xml
- mapred-site.xml

Setup Java path :

Setup the java path in the following files:

- hadoop-env.sh
- yarn-env.sh
- mapred-env.sh

Files

hdfs-site.xml
~/hadoop-3.3.4/etc/hadoop

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
 <property>
 <name>dfs.replication</name>
 <value>1</value>
 </property>

<property>
<name>dfs.namenode.name.dir</name>
<value>/home/hadoop/hadoop-3.3.4/etc/hadoop/name/namenode</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>/home/hadoop/hadoop-3.3.4/etc/hadoop/data/datanode</value>
</property>
</configuration>
```

core-site.xml
~/hadoop-3.3.4/etc/hadoop

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
 <name>fs.defaultFS</name>
 <value>hdfs://master:9000</value>  </property>
 <property>
<name>hadoop.proxyuser.dataflair.groups</name> <value>*</value>
 </property>
 <property>
<name>hadoop.proxyuser.dataflair.hosts</name> <value>*</value>
 </property>
 <property>
<name>hadoop.proxyuser.server.hosts</name> <value>*</value>
 </property>
 <property>
<name>hadoop.proxyuser.server.groups</name> <value>*</value>
 </property>
```

mapred-site.xml
~/hadoop-3.3.4/etc/hadoop

```xml
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
 <name>mapreduce.framework.name</name>  <value>yarn</value>
 </property>
 <property>
 <name>mapreduce.application.classpath</name>

<value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/lib/*</value>
 </property>

<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

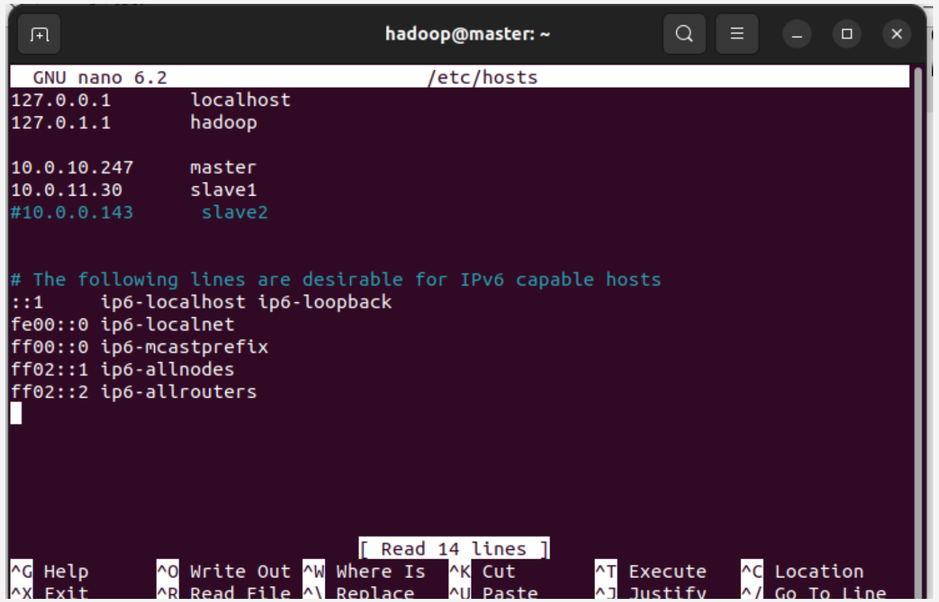
yarn-site.xml
~/hadoop-3.3.4/etc/hadoop

```xml
<?xml version="1.0"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->
<configuration>

<!-- Site specific YARN configuration properties -->
 <property>
 <name>yarn.nodemanager.aux-services</name>
 <value>mapreduce_shuffle</value>
 </property>
 <property>
 <name>yarn.nodemanager.env-whitelist</name>

<value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
 </property>

<property>
<name>yarn.nodemanager.aux-services</name> <value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
<description>The hostname of the RM.</description>
<name>yarn.resourcemanager.hostname</name>
<value>master</value>
</property>
<property>
<description>The address of the applications manager interface in the RM.</description>
<name>yarn.resourcemanager.address</name>
<value>master:8032</value>
</property>

</configuration>
```

HDFS Setup

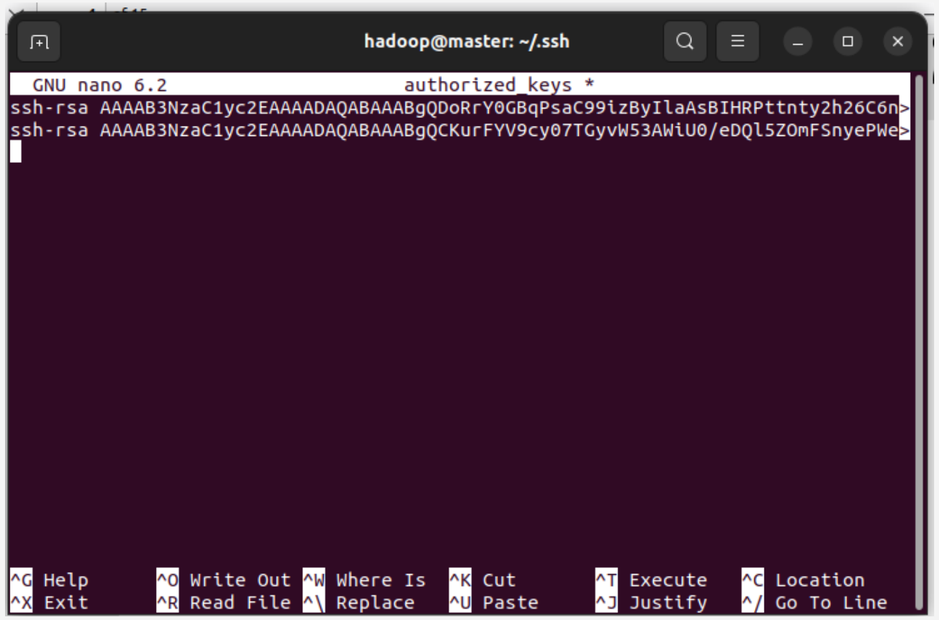
Add IP address of nodes

Add all IP addresses of nodes under
/home/username/etc/hosts



Setup SSH Keys of nodes

SSH keys generated by each system should
add under
/home/username/.ssh/authorized_keys



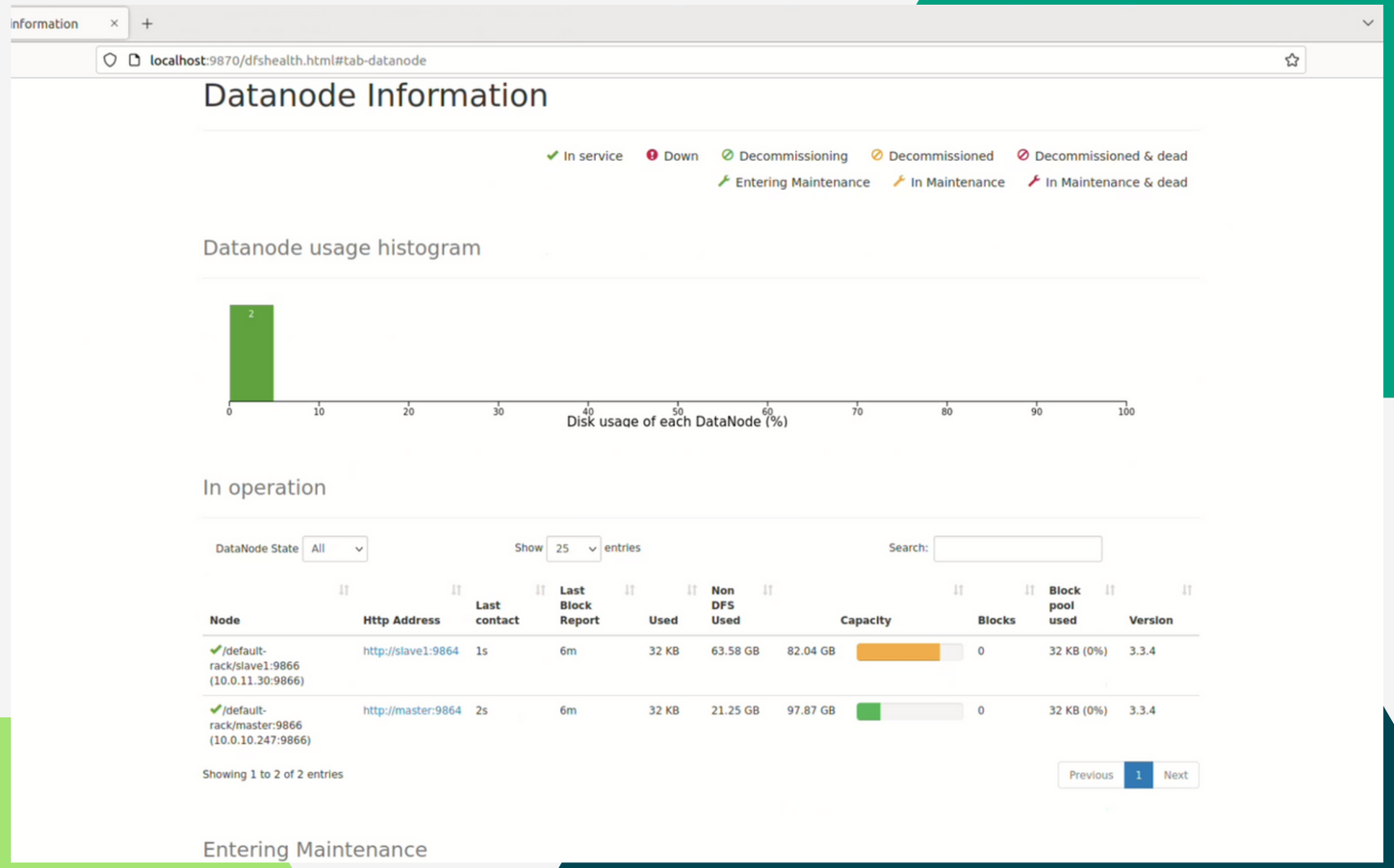
- **Start the Hadoop daemons**

Start the Hadoop deamos on the master node using this command:
"start-all.sh"

- **Verify the cluster**

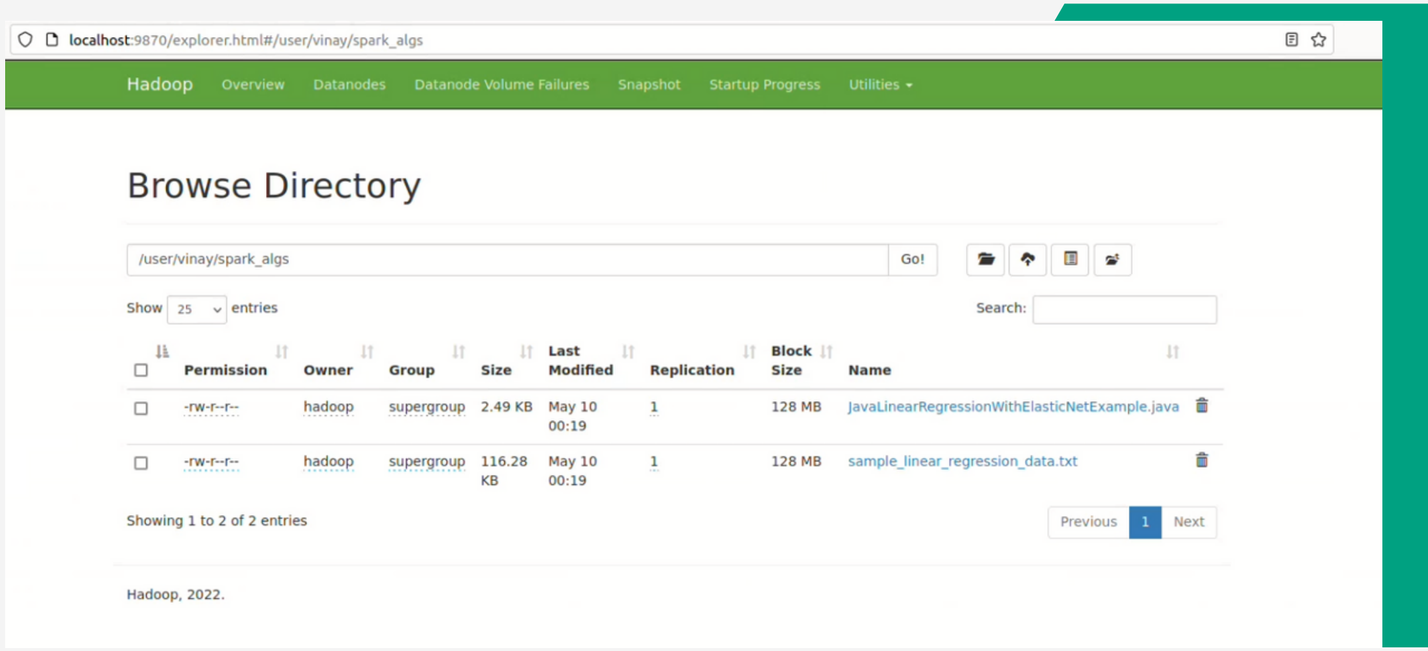
Verify the hadoop cluster is running by checking the Hadoop web page, which can be accesed at localhost

http://<master-node-IP_address>:9870

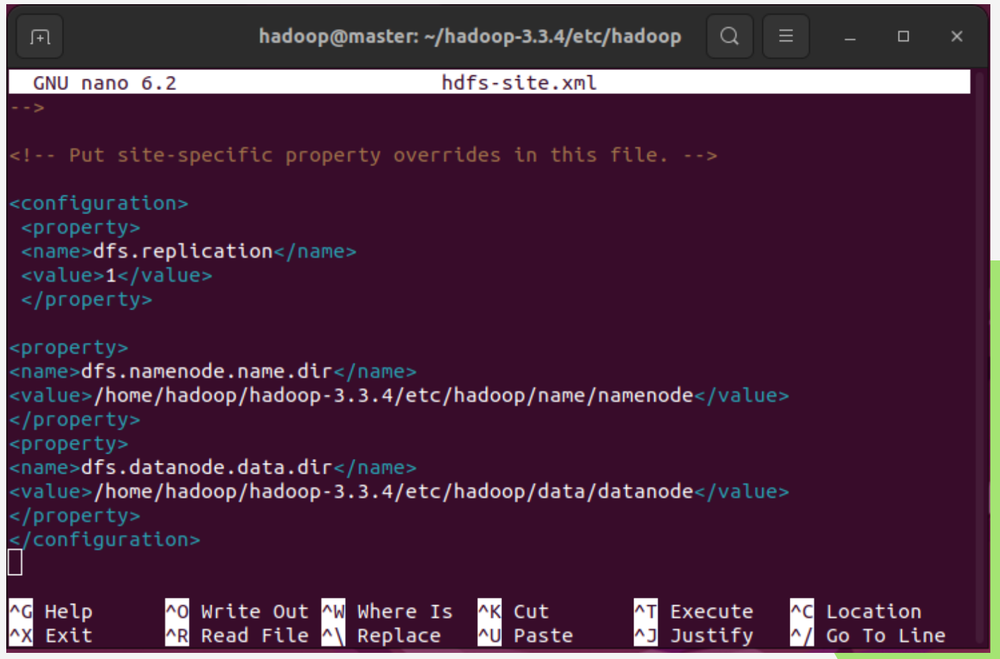


- **Upload Data set to hadoop cluster**

Dataset storage can be seen here
http://<master-node-IP_address>:9870



- **Location of Dataset in Datanode**



Spark:

Apache Spark is an open-source distributed computing system designed for big data processing and analytics.It provides a unified platform for batch processing, real-time streaming, graph processing, and machine learning workloads.Spark uses in-memory processing for faster data processing and provides APIs in Java, Python, Scala, and R.Spark runs on top of Hadoop Distributed File System (HDFS) and other storage systems like Amazon S3, and provides integration with various data sources such as Hive, Cassandra, and Kafka.Spark has gained popularity due to its ease of use, scalability, and performance improvements over traditional big data processing systems.

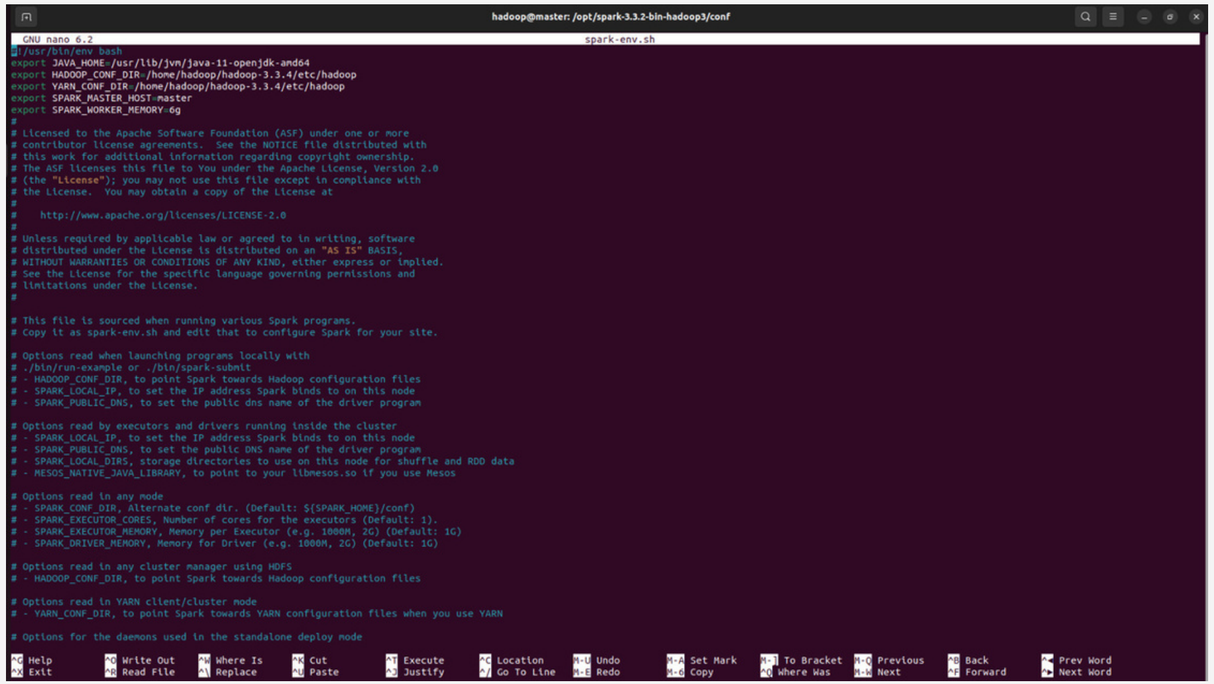
Spark cluster :

A Spark cluster is a group of computers working together to process large-scale data workloads.In a Spark cluster, the workload is distributed among the cluster nodes, enabling parallel processing for faster data processing and analytics.

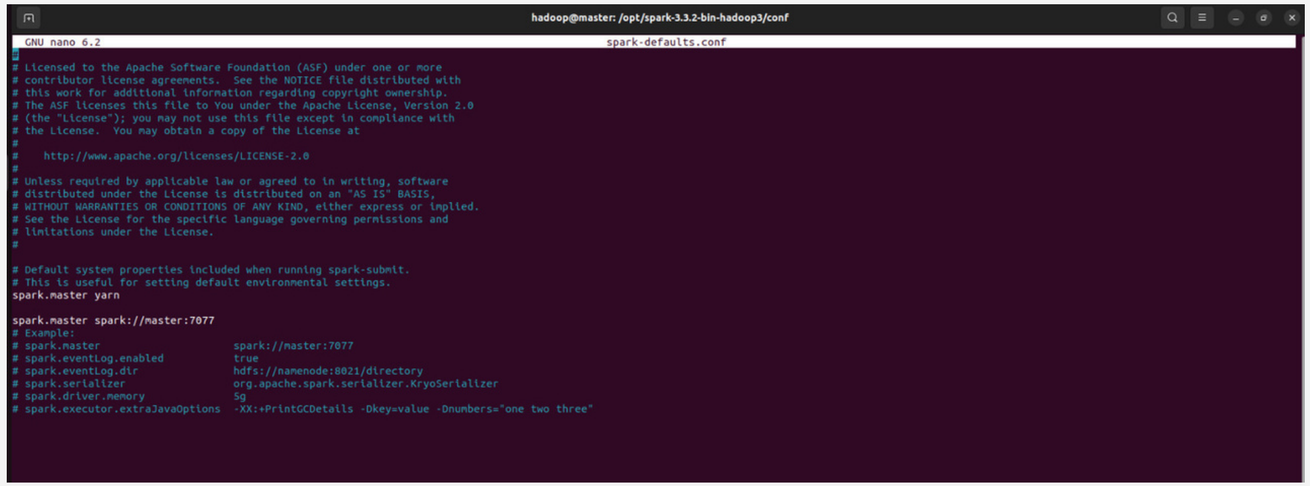
Spark Setup

Configure spark

- Spark-env.sh



- Spark-deafults.conf



SPARK

Spark Master at spark://master:7077

URL: spark://master:7077
Alive Workers: 2
Cores in use: 16 Total, 0 Used
Memory in use: 20.9 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (2)

Worker Id	Address	State	Cores	Memory	Resources
worker-20230509184703-10.0.11.30-46321	10.0.11.30:46321	ALIVE	8 (0 Used)	14.3 GiB (0.0 B Used)	
worker-20230510001703-10.0.10.247-35929	10.0.10.247:35929	ALIVE	8 (0 Used)	6.6 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Data Set:

Sample_Linear_Regression

A sample linear regression dataset typically contains a set of data points, where each data point has one or more independent variables and a dependent variable. The goal of linear regression is to create a linear model that can predict the value of the dependent variable based on the values of the independent variables.


```
-9.490009878824548 1:0.4551273600657362 2:0.36644694351969087 3:-0.38256108933468047 4:-0.4458430198517267 5:0.33109790358914726 6:0.8067445293443565 7:-0.2624341731773887
8:-0.44850386111659524 9:-0.07269284838169332 10:0.5658035575800715
0.2577820163584905 1:0.8386555657374337 2:-0.1270180511534269 3:0.499812362510895 4:-0.22686625128130267 5:-0.6452430441812433 6:0.18869982177936828 7:-0.5804648622673358
8:0.651931743775642 9:-0.6555641246242951 10:0.1748547635725912 2
-4.438869807456516 1:0.5025608135349202 2:0.14208069682973434 3:0.16004976900412138 4:0.505019897181302 5:-0.9371635223468384 6:-0.2841601610457427 7:0.6355938616712786
8:-0.1646249064941625 9:0.9480713629917628 10:0.4268125156464581 7
-19.782762789614537 1:-0.0388509668871313 2:-0.4166870051763918 3:0.8997202693189332 4:0.6409836467726933 5:0.273289095712564 6:-0.26175701211620517 7:-0.2794902492677298
8:-0.1306778297187794 9:-0.08536581111046115 10:-0.05462315824828923
-7.966593841555266 1:-0.06195495876886281 2:0.6546448480299902 3:-0.6979368909424835 4:0.6677324708883314 5:-0.07938725467767771 6:-0.43885601665437957 7:-0.608071585153688
8:-0.6414531182501653 9:0.7313735926547045 10:-0.02681867634761192 5
-7.896274316726144 1:-0.15805658673794265 2:0.26573958270655806 3:0.3997172901343442 4:-0.3693430998846541 5:0.14324061105995334 6:-0.25797542063247825 7:0.7436291919296774
8:0.6114618853239959 9:0.2324273700703574 10:-0.25128128782199144
-8.464803554195287 1:0.39449745853945895 2:0.817229160415142 3:-0.6077058562362969 4:0.6182496334554788 5:0.2558665508269453 6:-0.07320145794330979 7:-0.38884168866510227
8:0.07981886851873865 9:0.2702220289127761 4 10:-0.7474843534024693
2.1214592666251364 1:-0.005346215048158909 2:-0.9453716674280683 3:-0.9270309666195007 4:-0.03231229009138969 5:0.31010676221964206 6:-0.20846743965751569 7:0.8803449313707621
8:-0.2307783121654172 2 9:0.29246395759528565 10:0.5409312755478819
1.0720117616524107 1:0.7880855916368177 2:0.19767407429003536 3:0.9520689432368168 4:-0.845829774129496 5:0.5502413918543512 6:-0.44235539500246457 7:0.7984106594591154
8:-0.2523277127589152 9:-0.1373808897290778 10:-0.3353514432305029
-13.772441561702871 1:-0.3697050572653644 2:-0.1145281158275928 3:-0.807098168238352 4:0.4903066124307711 5:-0.6582805242342049 6:0.6107814398427647 7:-0.7204208094262783
8:-0.8141063661170889 9:-0.9459402662357332 10:0.09666938346350307
-5.082010756207233 1:-0.43560342773870375 2:0.9349906440170221 3:0.8090021580031235 4:-0.3121157071110545 5:-0.9718883630945336 6:0.6191882496201251 7:0.0429886073795116
8:0.67031111001540 2 9:0.16692329718223786 10:0.3764921386950297 3
7.88778653653123 7 1:0.1127644026381038 3 2:-0.7684997525607482 3:0.1770172737885798 4:0.7902845707138706 5:0.2529503304079441 6:-0.23483801763662826 7:0.8072501895004851
8:0.6673992021927047 9:-0.4796127376677324 10:0.9244724404994455
14.323146365332388 1:-0.2049276879687938 2:0.1470694373531216 3:-0.48366999792166787 4:0.643491115907358 5:0.3183669486383729 6:0.22821350958477082 7:-0.02360525108614930 4
8:-0.2770587742156372 9:0.4759632645837743 6 10:0.7107229819632654
-20.057482615789212 1:-0.3205057828114841 2:0.51605972926996 3:0.4521564098818151 6 4:0.01712446974606241 5:0.5508198371849293 6:-0.2478254241316491 7:0.7256483175955235 8:0.39418662792516
9:-0.6797384914236382 10:0.6001217520150142
-0.8995693247765151 1:0.4508991072414843 2:0.589749448443134 3:0.6464818311502738 4:0.7005669004769028 5:0.9699584106930381 6:-0.7417466269908464 7:0.22818964839784495
8:0.08574936236270037 9:-0.6945765138377225 10:0.06915201979238828
-19.16829262296376 1:0.09798746565879424 2:-0.34288007110901964 3:0.440249350802451 4:-0.22440768392359534 5:-0.9695067570891225 6:-0.7942032659310758 7:-0.79228620551739 8
8:-0.6535487038528798 9:0.7952676470618951 10:-0.1622831617066689
5.601801561245534 1:0.6949189734965766 2:-0.32697929564739403 3:-0.15359663581829275 4:-0.8951865090520432 5:0.2057889391931318 6:-0.6676656789571533 7:-0.03553655732400762
8:0.14550349954571096 9:0.03460054207819185 4 10:0.4223352065067103
-3.2256352187273354 1:0.35278245969741096 2:0.7022211035026023 3:0.5686638754605697 4:-0.4202155290448111 5:-0.26102723928249216 6:0.01068821594141677 9 7:-0.4311544807877927
8:0.9500151672991208 9:0.14380635780710693 10:-0.7549354840975826
1.5299675726687754 1:-0.13079299081883855 2:0.0983382230287082 3:0.15347083875928424 4:0.45507300685816965 5:0.1921083467305864 6:0.6361110540492223 7:0.7675261182370992
8:-0.2543488202081907 9:0.2927051050236915 10:0.680182444769418
-0.250102447941961 1:-0.8062832278617296 2:0.8266289890474885 3:0.22684501241708888 4:0.1726291966578266 5:-0.6778773666126594 6:0.9993906921393696 7:0.1789490173139363
8:0.5584053824232391 9:0.0349589470436817 4 10:-0.8505720014852347
12.792267926563595 1:-0.008461200645088818 2:-0.648273596036564 3:-0.005334477339629995 4:0.3781469006858833 5:0.30565234666790686 6:-0.2822867492866177 7:0.10175120738413801
```

Algorithm:

Linear Regression

```java
/*
 * Licensed to the Apache Software Foundation (ASF) under one or more
 * contributor license agreements.  See the NOTICE file distributed with
 * this work for additional information regarding copyright ownership.
 * The ASF licenses this file to You under the Apache License, Version 2.0
 * (the "License"); you may not use this file except in compliance with
 * the License.  You may obtain a copy of the License at
 *
 *    http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

//package org.apache.spark.examples.ml;

// $example on$
import org.apache.spark.ml.regression.LinearRegression;
import org.apache.spark.ml.regression.LinearRegressionModel;
import org.apache.spark.ml.regression.LinearRegressionTrainingSummary;
import org.apache.spark.ml.linalg.Vectors;
import org.apache.spark.sql.Dataset;
import org.apache.spark.sql.Row;
import org.apache.spark.sql.SparkSession;
// $example off$

public class JavaLinearRegressionWithElasticNetExample {
  public static void main(String[] args) {
    SparkSession spark = SparkSession
      .builder()
      .appName("JavaLinearRegressionWithElasticNetExample")
      .getOrCreate();

    // $example on$
    // Load training data.
    Dataset<Row> training = spark.read().format("libsvm")
```

```java
// $example off$

public class JavaLinearRegressionWithElasticNetExample {
  public static void main(String[] args) {
    SparkSession spark = SparkSession
      .builder()
      .appName("JavaLinearRegressionWithElasticNetExample")
      .getOrCreate();

    // $example on$
    // Load training data.
    Dataset<Row> training = spark.read().format("libsvm")
      .load("hdfs://master:50000/user/sample_linear_regression_data.txt");

    LinearRegression lr = new LinearRegression()
      .setMaxIter(10)
      .setRegParam(0.3)
      .setElasticNetParam(0.8);

    // Fit the model.
    LinearRegressionModel lrModel = lr.fit(training);

    // Print the coefficients and intercept for linear regression.
    System.out.println("Coefficients: "
      + lrModel.coefficients() + " Intercept: " + lrModel.intercept());

    // Summarize the model over the training set and print out some metrics.
    LinearRegressionTrainingSummary trainingSummary = lrModel.summary();
    System.out.println("numIterations: " + trainingSummary.totalIterations());
    System.out.println("objectiveHistory: " + Vectors.dense(trainingSummary.objectiveHistory()));
    trainingSummary.residuals().show();
    System.out.println("RMSE: " + trainingSummary.rootMeanSquaredError());
    System.out.println("r2: " + trainingSummary.r2());
    // $example off$

    spark.stop();
  }
}
```

iiit@master: ~/ml_algorithm/JLR | iiit@slave-1: ~ | iiit@slave-2: ~ | iiit@master: ~/ml_algorithm

```
iiit@master:~/ml_algorithm/JLR$
```

Output:

```
+--------------------+
|           residuals|
+--------------------+
|  -9.889232683103197|
|  0.5533794340053553|
|  -5.204019455758822|
| -20.566686715507508|
|    -9.4497405180564|
|  -6.909112502719487|
|  -10.00431602969873|
|  2.0623978070504845|
|  3.1117508432954772|
|  -15.89360822941938|
|  -5.036284254673026|
|  6.4832158769943335|
|  12.429497299109002|
|  -20.32003219007654|
|    -2.0049838218725|
| -17.867901734183793|
|   7.646455887420495|
| -2.2653482182417406|
|-0.10308920436195645|
|  -1.380034070385301|
+--------------------+
only showing top 20 rows
```

Thank You