Execution on Standalone and Spark Cluster

Contents

1. **Abstract**:

Same program is being executed On Standalone spark context(local machine) and on Spark cluster, two programs are considered one with dataset of small size and one with dataset of large size. The program processing dataset of small size and big size executed faster on the Spark Cluster system and executed slower on Standalone system. HortonWorks platform has been used to execute the Python files.

2. **Introduction**:

Here data_mllib_sample_libsvm_data and data_mllib_sample_libsvm_data_large datasets are considered for time consumption calculation. binomial_logistic_regression.py is executed with data_mllib_sample_libsvm_data dataset in Standalone mode and Cluster mode.binomial_logistic_regression.py is executed with data_mllib_sample_libsvm_data_large dataset in Standalone mode and Cluster mode. Time is calculated for execution of the binomial_logistic_regression.py file in Standalone and Spark Cluster mode in both large and small dataset cases.

Note: Zoom in the document to have perfect view of the snapshots being attached in the document.

3. **Logistic Regression**:

Logistic Regression is one of the most commonly used Machine Learning algorithms. Data is being classified to class 0 or class1,binary classification using logistic regression we obtain coefficients and Intercepts which are plotted to divide the data into two separate classes. Its mainly used for classification.

Logistic Regression Example: Spam Detection

Spam detection is a binary classification problem where we have to classify whether the mail is spam or not. If the email is spam, we label it as 1 and if it is not spam, we label it as 0. In order to apply Logistic Regression to identify spam detection, the following features of the email are considered:

- Sender of the email
- Number of typos in the email
- Occurrence of words/phrases like "offer", "prize", "free gift", etc.

The resulting feature vector is then used to train a Logistic classifier which emits a score in the range 0 to 1. If the score is more than 0.5, we label the email as spam. Otherwise, we don't label it as spam.

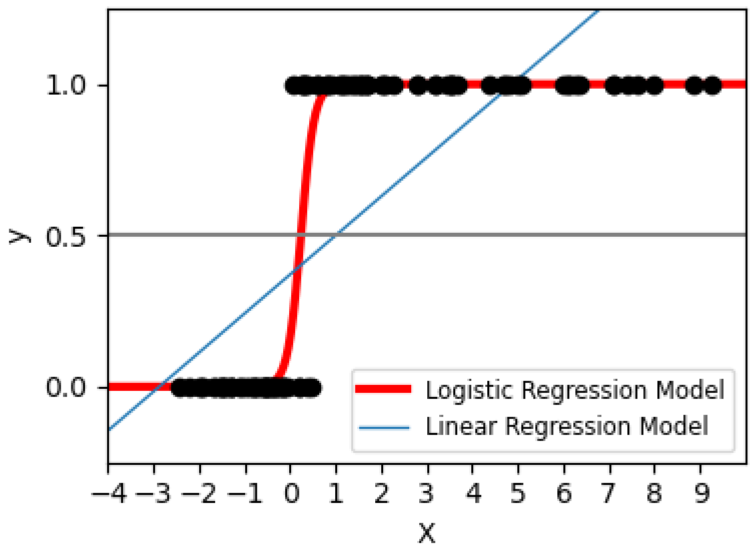
Logistic Regression Example: Credit Card Fraud

The Credit Card Fraud Detection is a major problem to be addressed in the banking industry because banks each year spend hundreds of millions of dollars due to fraud. When a credit card transaction happens, the bank takes a note of several factors. For instance, transaction date, transaction amount, place of transaction, type of purchase, etc. Based on these factors, a Logistic Regression model is developed to determine whether the transaction is a fraud or not. For instance, if the amount is too high and the bank knows that the concerned person never makes such high purchases, they may label it as a fraud.

Here the data is being classified, data is grouped to class 0 or Class1, Red Curve indicates the Logistic regression. Points greater than 0.5 belongs to class 1, less than 0.5 belongs to Class0, coefficients indicate the co-ordinates for plotting and Intercept indicates the intercept values at which the plot coincides with the axis.

Logistic regression is used for binary as well as categorical classification. Binary classification consists of two classes categorical classification consists of many classes.

Execution on Standalone and Spark Cluster



Logistic regression is a popular method to predict a categorical response. It is a special case of Generalized Linear models that predicts the probability of the outcomes. In spark.ml logistic regression can be used to predict a binary outcome by using binomial logistic regression, or it can be used to predict a multiclass outcome by using multinomial logistic regression. family parameter is used to select between these two algorithms, or leave it unset and Spark will infer the correct variant.

Multinomial logistic regression is used for binary classification by setting the family param to "multinomial". It will produce two sets of **coefficients and two intercepts**.

4. **Dataset details**:

   0 128:51 129:159 130:253 131:159

   1 159:124 160:253 161:255 162:63

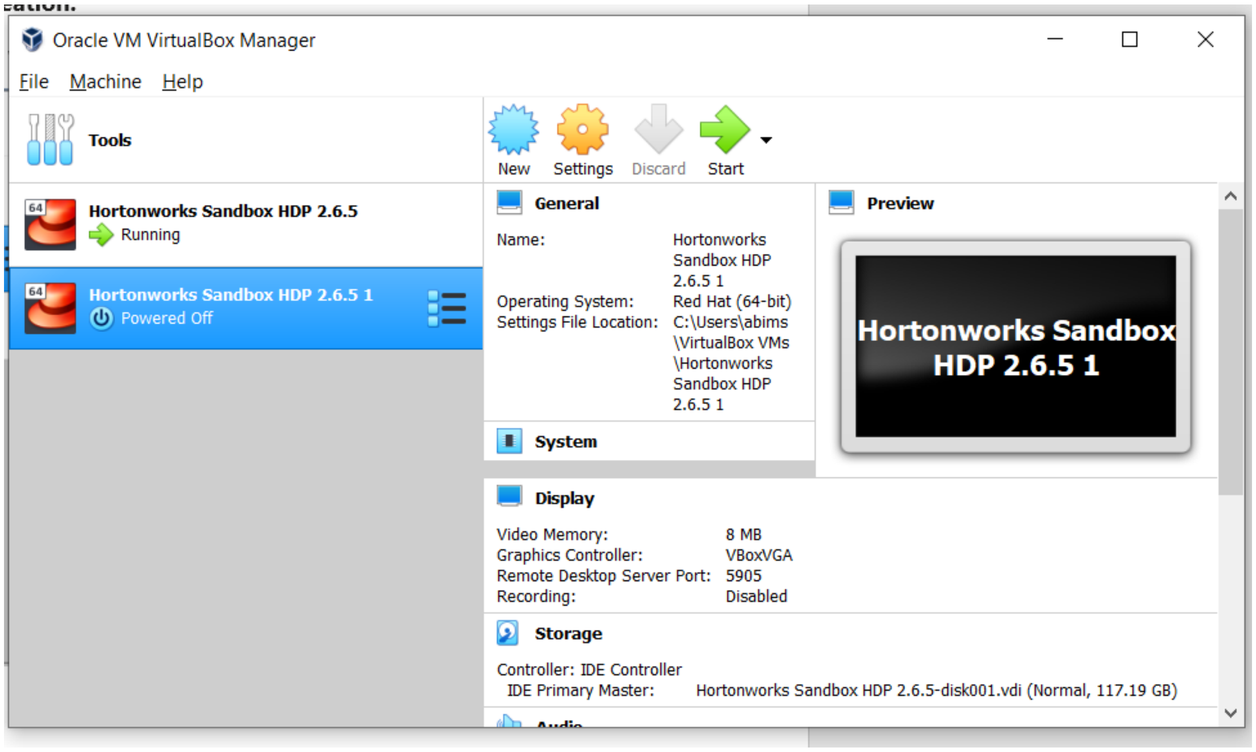
   This data is in libsvm format it is being used in the Spark framework. This a generic dataset being used in the Spark framework for classification, regression, svm problems.

   Data used in the dataset is common data mainly written for the purpose of executing Machine learning algorithms. It can be bank information, Insurance information, its generic valid dataset being used to execute machine learning algorithm in Spark framework.

   Each line contains an instance and is ended by a '\n' character. For classification, <label> is an integer indicating the class label (multi-class is supported). For regression, <label> is the target value which can be any real number. The pair <index>:<value> gives a feature (attribute) value: <index> is an integer starting from 1 and <value> is a real number. The only exception is the precomputed kernel, where <index> starts from 0. Indices must be in ASCENDING order. Labels in the testing file are only used to calculate accuracy or errors. If they are unknown, just fill the first column with any numbers.

5. HortonsWorks Sandbox Creation:

- Creating a Hortonworks Sandbox, which provides us the platform to execute the scripts. We have to select the required Sandox and we should click on Start.



Hortonworks Sandbox HDP 2.6.5
Running
Hortonworks Sandbox HDP 2.6.5 1
Powered Off

General
Name: Hortonworks Sandbox HDP 2.6.5 1
Operating System: Red Hat (64-bit)
Settings File Location: C:\Users\abims\VirtualBox VMs\Hortonworks Sandbox HDP 2.6.5 1

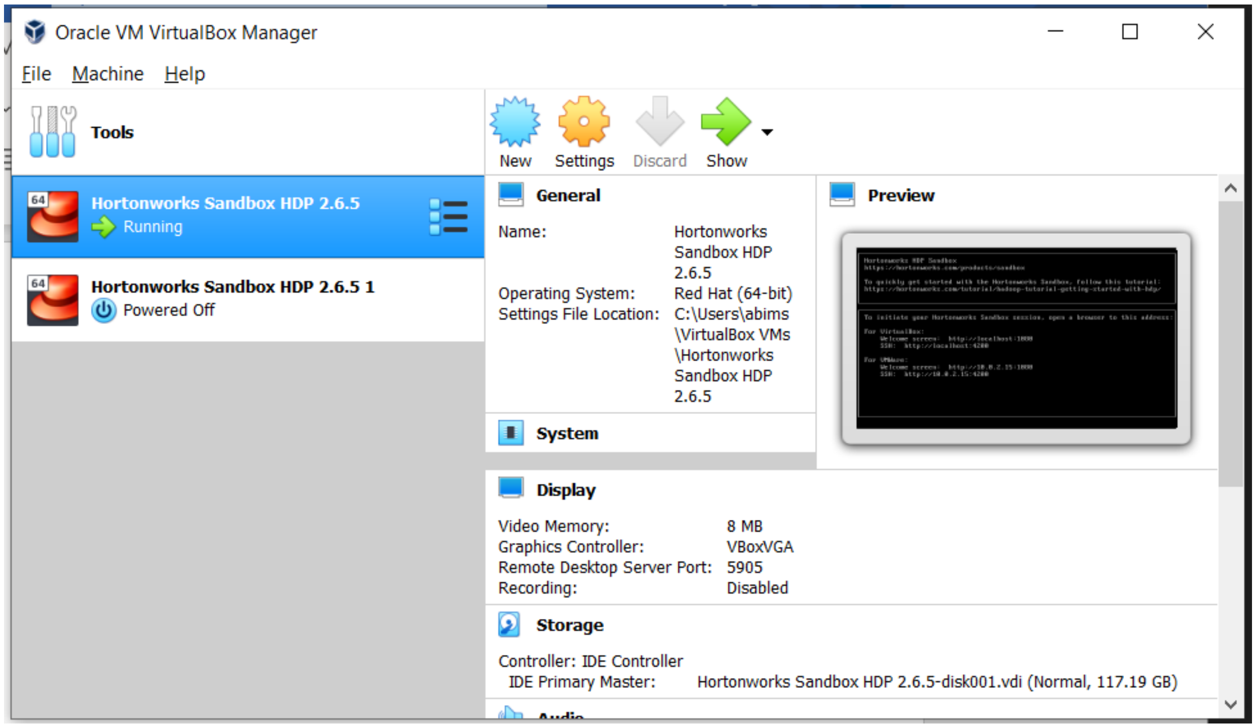
System

Display
Video Memory: 8 MB
Graphics Controller: VBoxVGA
Remote Desktop Server Port: 5905
Recording: Disabled

Storage
Controller: IDE Controller
IDE Primary Master: Hortonworks Sandbox HDP 2.6.5-disk001.vdi (Normal, 117.19 GB)

- On Starting the Sandbox VM starts running and pops up one more window with details to access the Sandbox through SSH.

Execution on Standalone and Spark Cluster


Oracle VM VirtualBox Manager
File
Machine
Help
Tools
New
Settings
Discard
Show
Hortonworks Sandbox HDP 2.6.5
Running
Hortonworks Sandbox HDP 2.6.5 1
Powered Off
Preview
Audio

General

Name:	Hortonworks Sandbox HDP 2.6.5
Operating System:	Red Hat (64-bit)
Settings File Location:	C:\Users\abims \VirtualBox VMs \Hortonworks Sandbox HDP 2.6.5

System

Display

Video Memory:	8 MB
Graphics Controller:	VBoxVGA
Remote Desktop Server Port:	5905
Recording:	Disabled

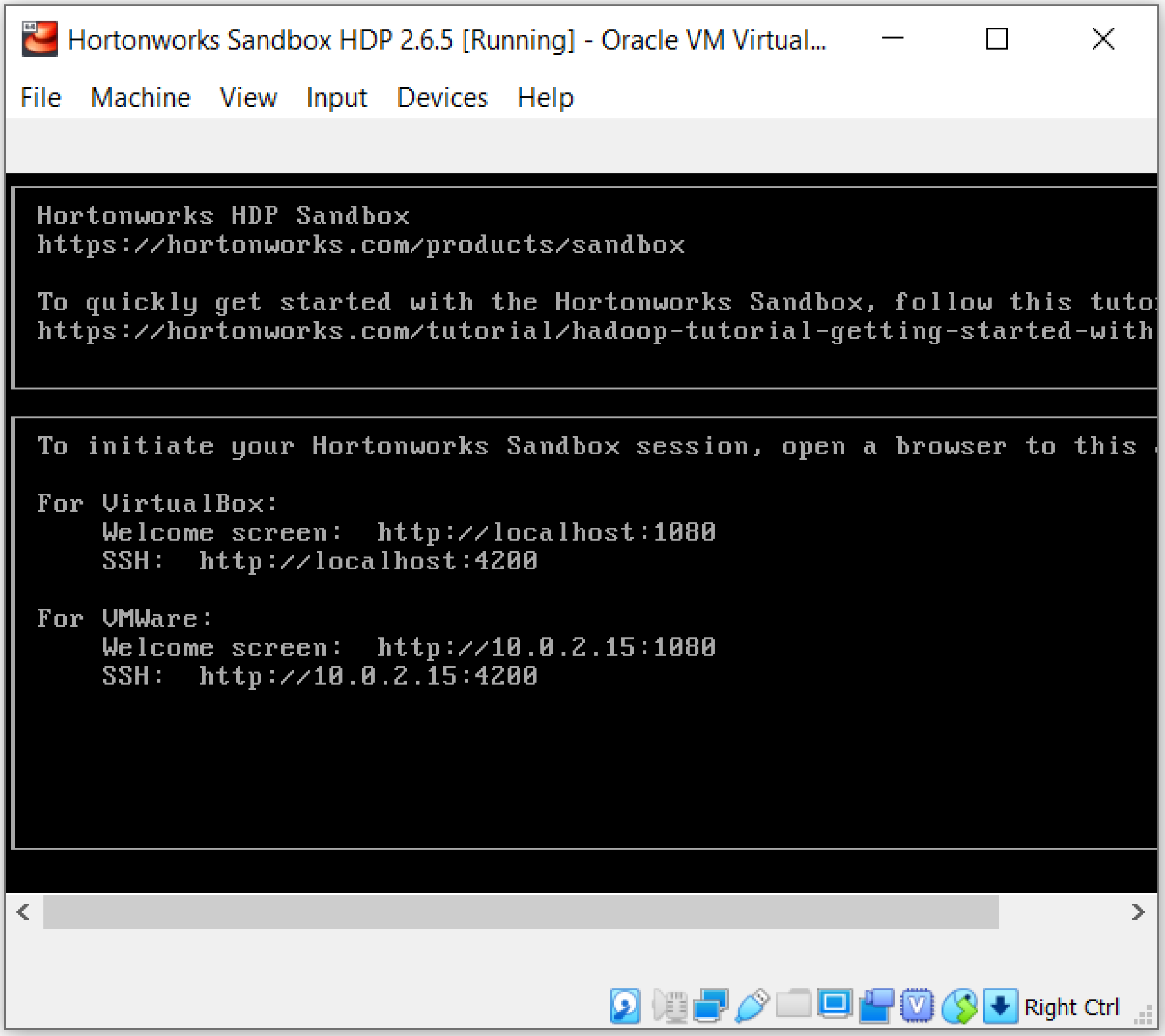
Storage

Controller: IDE Controller
IDE Primary Master: Hortonworks Sandbox HDP 2.6.5-disk001.vdi (Normal, 117.19 GB)

```
Hortonworks HDP Sandbox
https://hortonworks.com/products/sandbox

To quickly get started with the Hortonworks Sandbox, follow this tutorial:
https://hortonworks.com/tutorial/hadoop-tutorial-getting-started-with-hdp/

To initiate your Hortonworks Sandbox session, open a browser to this address:

For VirtualBox:
    Welcome screen:  http://localhost:1080
    SSH:  http://localhost:4200

For VMWare:
    Welcome screen:  http://10.0.2.15:1080
    SSH:  http://10.0.2.15:4200
```

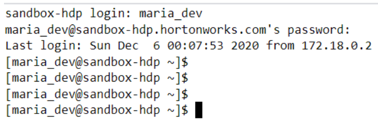
- The below window displays Sandbox welcome screen and SSH details for VMWare and Virtualbox

Execution on Standalone and Spark Cluster


Hortonworks Sandbox HDP 2.6.5 [Running] - Oracle VM Virtual...
File
Machine
View
Input
Devices
Help
Right Ctrl

```
Hortonworks HDP Sandbox
https://hortonworks.com/products/sandbox

To quickly get started with the Hortonworks Sandbox, follow this tuto
https://hortonworks.com/tutorial/hadoop-tutorial-getting-started-with

To initiate your Hortonworks Sandbox session, open a browser to this

For VirtualBox:
    Welcome screen:  http://localhost:1080
    SSH:  http://localhost:4200

For VMWare:
    Welcome screen:  http://10.0.2.15:1080
    SSH:  http://10.0.2.15:4200
```

- On Creating a Hortonworks Sandbox, We can access the terminal through SSH. We can run our scripts here.



```
sandbox-hdp login: maria_dev
maria_dev@sandbox-hdp.hortonworks.com's password:
Last login: Sun Dec  6 00:07:53 2020 from 172.18.0.2
[maria_dev@sandbox-hdp ~]$
[maria_dev@sandbox-hdp ~]$
[maria_dev@sandbox-hdp ~]$
[maria_dev@sandbox-hdp ~]$
```

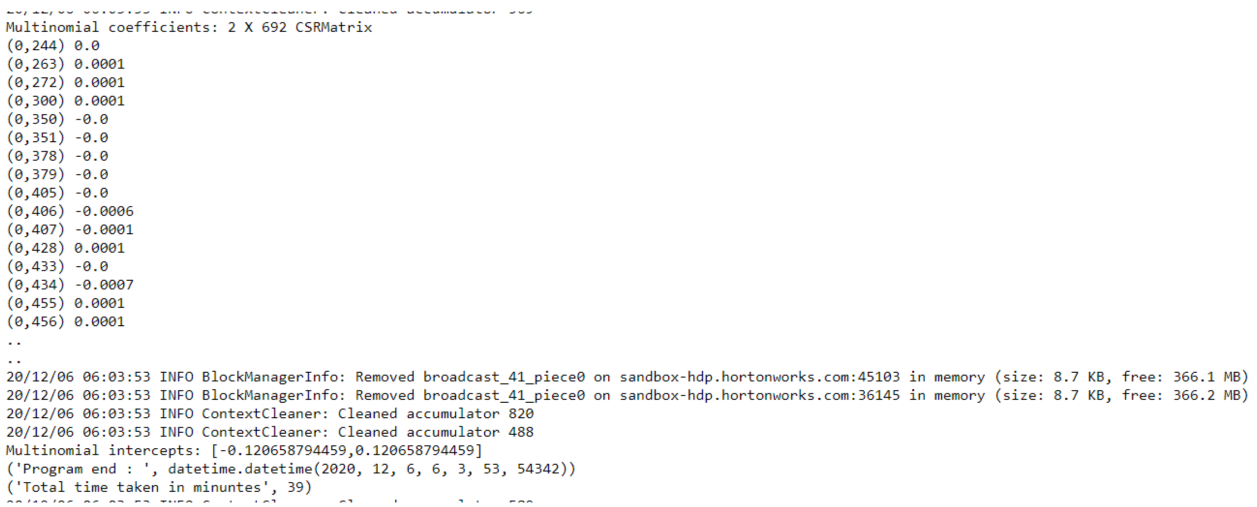
- Coefficients and Intercept is required to plot Logistic regression which mainly divides the data into different classes hence Coefficients and Intercept output is printed.

- Command: **spark-submit binomial_logistic_regression.py**

Execution on Standalone and Spark Cluster

Above command is executed to execute the python file with small dataset in a Standalone system.. Total time taken is in seconds not minutes.

Small Dataset:

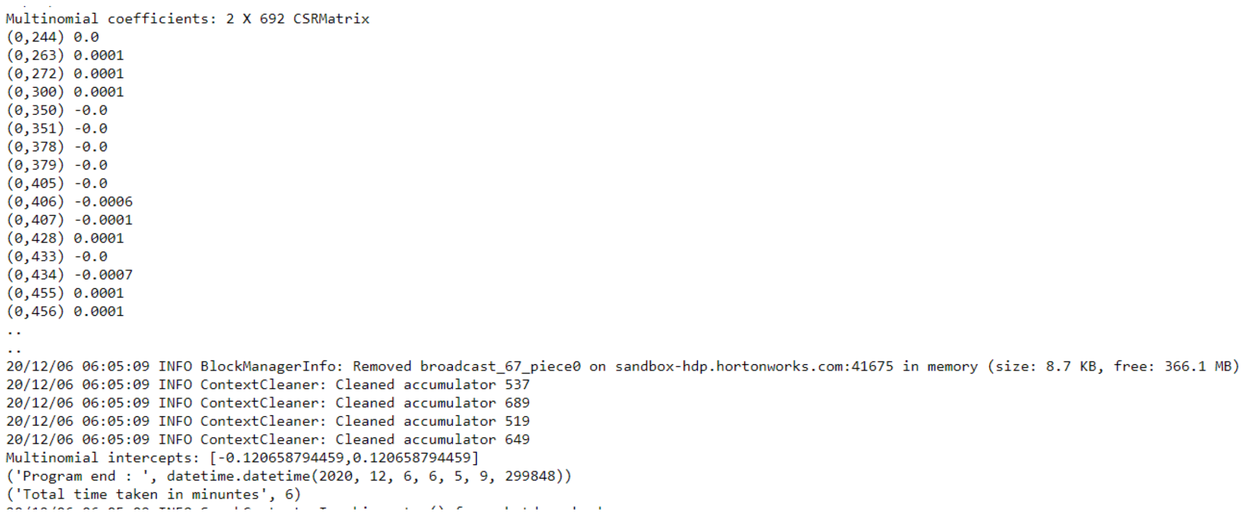


```
Multinomial coefficients: 2 X 692 CSRMatrix
(0,244) 0.0
(0,263) 0.0001
(0,272) 0.0001
(0,300) 0.0001
(0,350) -0.0
(0,351) -0.0
(0,378) -0.0
(0,379) -0.0
(0,405) -0.0
(0,406) -0.0006
(0,407) -0.0001
(0,428) 0.0001
(0,433) -0.0
(0,434) -0.0007
(0,455) 0.0001
(0,456) 0.0001
..
..
20/12/06 06:03:53 INFO BlockManagerInfo: Removed broadcast_41_piece0 on sandbox-hdp.hortonworks.com:45103 in memory (size: 8.7 KB, free: 366.1 MB)
20/12/06 06:03:53 INFO BlockManagerInfo: Removed broadcast_41_piece0 on sandbox-hdp.hortonworks.com:36145 in memory (size: 8.7 KB, free: 366.2 MB)
20/12/06 06:03:53 INFO ContextCleaner: Cleaned accumulator 820
20/12/06 06:03:53 INFO ContextCleaner: Cleaned accumulator 488
Multinomial intercepts: [-0.120658794459,0.120658794459]
('Program end : ', datetime.datetime(2020, 12, 6, 6, 3, 53, 54342))
('Total time taken in minuntes', 39)
```

- Command: **spark-submit –master yarn binomial_logistic_regression.py**

  Above command is executed to execute the python file with small dataset in a Spark Cluster system. Total time taken is in seconds not minutes.

  Small Dataset:



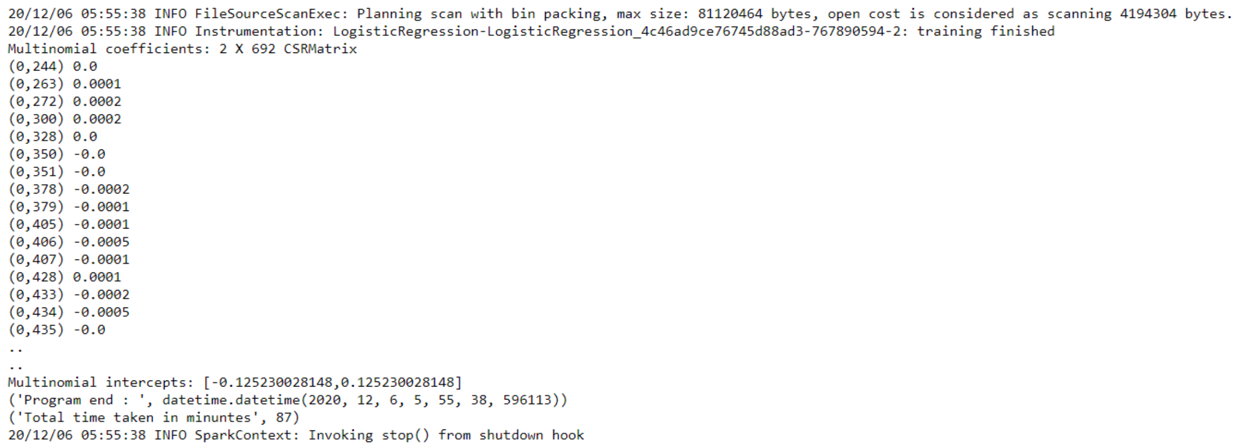
```
Multinomial coefficients: 2 X 692 CSRMatrix
(0,244) 0.0
(0,263) 0.0001
(0,272) 0.0001
(0,300) 0.0001
(0,350) -0.0
(0,351) -0.0
(0,378) -0.0
(0,379) -0.0
(0,405) -0.0
(0,406) -0.0006
(0,407) -0.0001
(0,428) 0.0001
(0,433) -0.0
(0,434) -0.0007
(0,455) 0.0001
(0,456) 0.0001
..
..
20/12/06 06:05:09 INFO BlockManagerInfo: Removed broadcast_67_piece0 on sandbox-hdp.hortonworks.com:41675 in memory (size: 8.7 KB, free: 366.1 MB)
20/12/06 06:05:09 INFO ContextCleaner: Cleaned accumulator 537
20/12/06 06:05:09 INFO ContextCleaner: Cleaned accumulator 689
20/12/06 06:05:09 INFO ContextCleaner: Cleaned accumulator 519
20/12/06 06:05:09 INFO ContextCleaner: Cleaned accumulator 649
Multinomial intercepts: [-0.120658794459,0.120658794459]
('Program end : ', datetime.datetime(2020, 12, 6, 6, 5, 9, 299848))
('Total time taken in minuntes', 6)
```

- Command: **spark-submit binomial_logistic_regression.py**

  Above command is executed to execute the python file with small dataset in a Standalone system. Total time taken is in seconds not minutes.

  Large Dataset:

Execution on Standalone and Spark Cluster

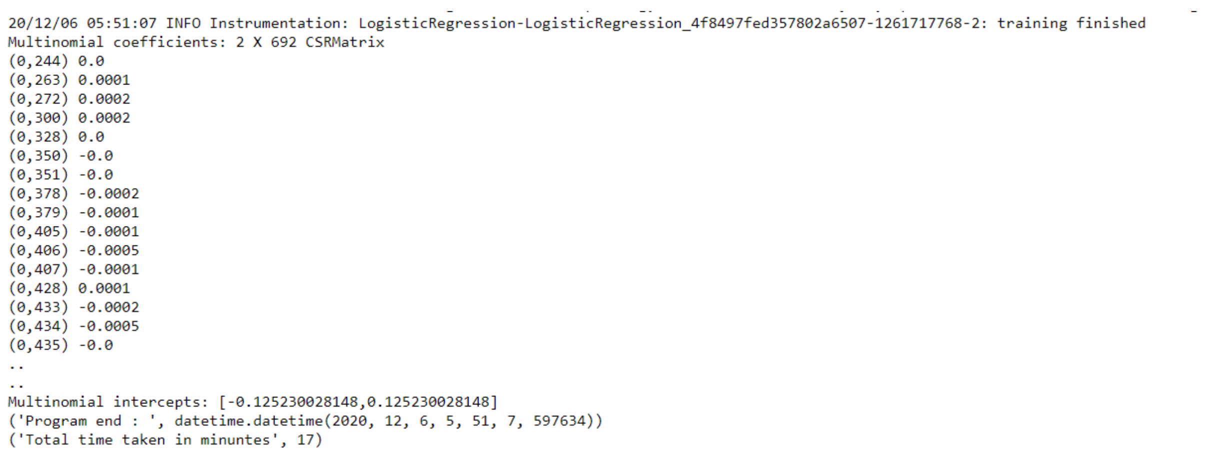


```
20/12/06 05:55:38 INFO FileSourceScanExec: Planning scan with bin packing, max size: 81120464 bytes, open cost is considered as scanning 4194304 bytes.
20/12/06 05:55:38 INFO Instrumentation: LogisticRegression-LogisticRegression_4c46ad9ce76745d88ad3-767890594-2: training finished
Multinomial coefficients: 2 X 692 CSRMatrix
(0,244) 0.0
(0,263) 0.0001
(0,272) 0.0002
(0,300) 0.0002
(0,328) 0.0
(0,350) -0.0
(0,351) -0.0
(0,378) -0.0002
(0,379) -0.0001
(0,405) -0.0001
(0,406) -0.0005
(0,407) -0.0001
(0,428) 0.0001
(0,433) -0.0002
(0,434) -0.0005
(0,435) -0.0
..
..
Multinomial intercepts: [-0.125230028148,0.125230028148]
('Program end : ', datetime.datetime(2020, 12, 6, 5, 55, 38, 596113))
('Total time taken in minuntes', 87)
20/12/06 05:55:38 INFO SparkContext: Invoking stop() from shutdown hook
```

- Command: **spark-submit –master yarn binomial_logistic_regression.py**

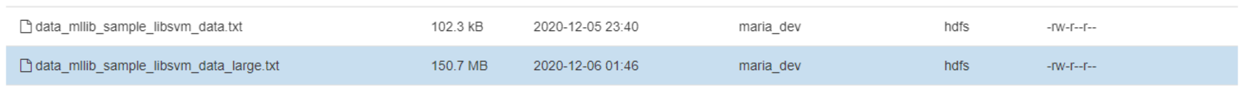
  Above command is executed to execute the python file with small dataset in a Standalone system. Total time taken is in seconds not minutes.

  Large Dataset:



```
20/12/06 05:51:07 INFO Instrumentation: LogisticRegression-LogisticRegression_4f8497fed357802a6507-1261717768-2: training finished
Multinomial coefficients: 2 X 692 CSRMatrix
(0,244) 0.0
(0,263) 0.0001
(0,272) 0.0002
(0,300) 0.0002
(0,328) 0.0
(0,350) -0.0
(0,351) -0.0
(0,378) -0.0002
(0,379) -0.0001
(0,405) -0.0001
(0,406) -0.0005
(0,407) -0.0001
(0,428) 0.0001
(0,433) -0.0002
(0,434) -0.0005
(0,435) -0.0
..
..
Multinomial intercepts: [-0.125230028148,0.125230028148]
('Program end : ', datetime.datetime(2020, 12, 6, 5, 51, 7, 597634))
('Total time taken in minuntes', 17)
```

Dataset files uploaded to the HDFS which are being used in the Python files to perform Classification and Regression.

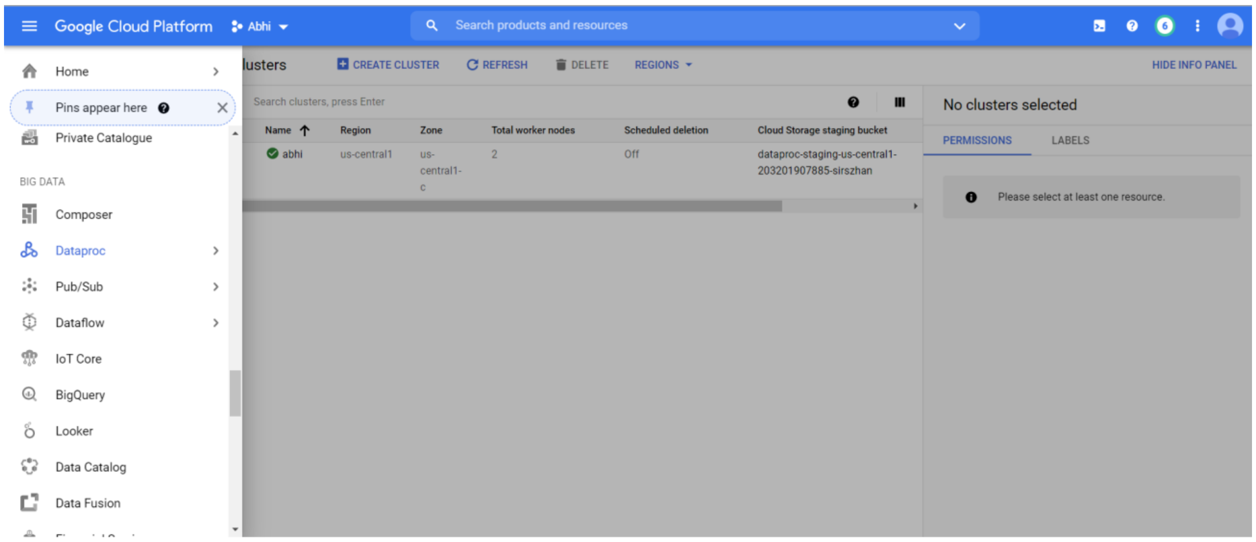


data_mllib_sample_libsvm_data.txt	102.3 kB	2020-12-05 23:40	maria_dev	hdfs	-rw-r--r--
data_mllib_sample_libsvm_data_large.txt	150.7 MB	2020-12-06 01:46	maria_dev	hdfs	-rw-r--r--

6. Google Cloud Platform :

- Take the Google Cloud Platform subscription which is free for 90 days**.**
- Executed on Google Cloud platform as well. Dataproc is selected on the list of options present in the dropdown menu. Dataproc contains the main Cluster details. We can create our own Cluster.

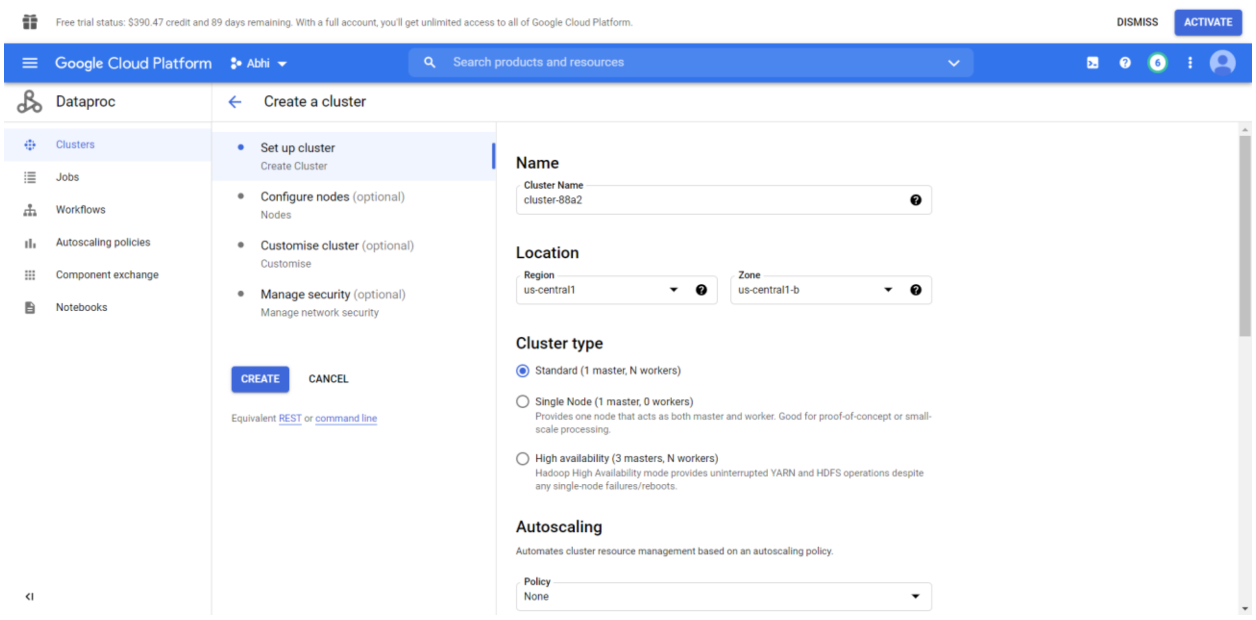
Execution on Standalone and Spark Cluster



Name	Region	Zone	Total worker nodes	Scheduled deletion	Cloud Storage staging bucket
abhi	us-central1	us-central1-c	2	Off	dataproc-staging-us-central1-203201907885-sirszhan

Please select at least one resource.

- On selecting Dataproc click on **CREATE CLUSTER** button , specify the Cluster name, Cluster type and other required details and click on create.



Free trial status: $390.47 credit and 89 days remaining. With a full account, you'll get unlimited access to all of Google Cloud Platform.

Name

Cluster Name: cluster-88a2

Location

Region: us-central1

Zone: us-central1-b

Cluster type

(●) Standard (1 master, N workers)

( ) Single Node (1 master, 0 workers)
Provides one node that acts as both master and worker. Good for proof-of-concept or small-scale processing.

( ) High availability (3 masters, N workers)
Hadoop High Availability mode provides uninterrupted YARN and HDFS operations despite any single-node failures/reboots.

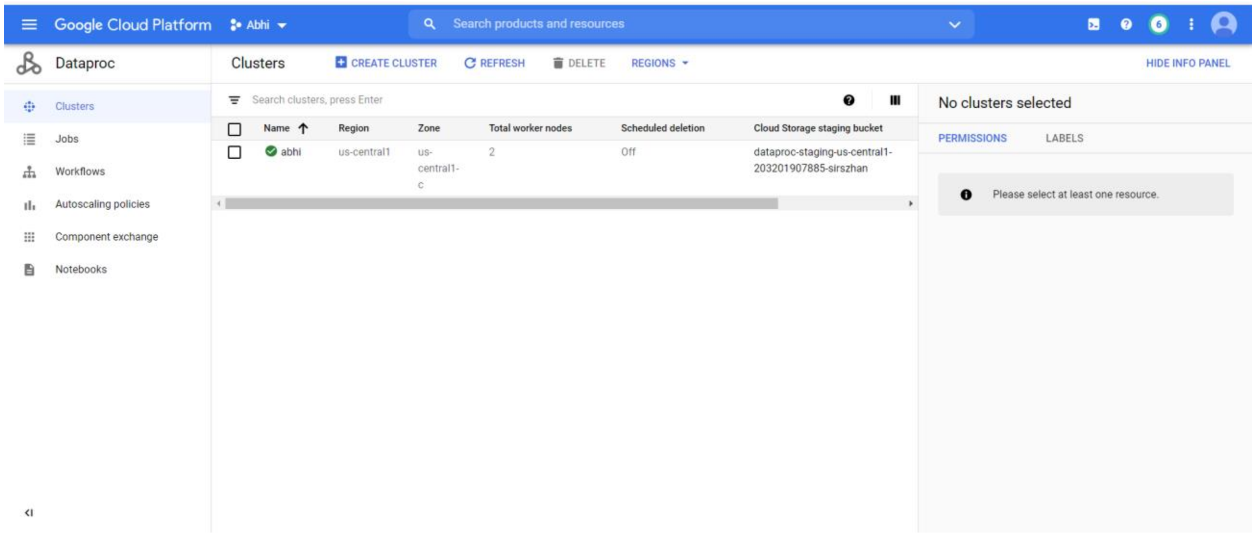
Autoscaling

Automates cluster resource management based on an autoscaling policy.

Policy: None

- On selecting Dataproc option we can see our new created Cluster. "Abhi" named cluster has been created here.
  Note: Zoom in the document to have perfect view of the attached snapshots in the document. Applicable to all attached snapshots.

Execution on Standalone and Spark Cluster

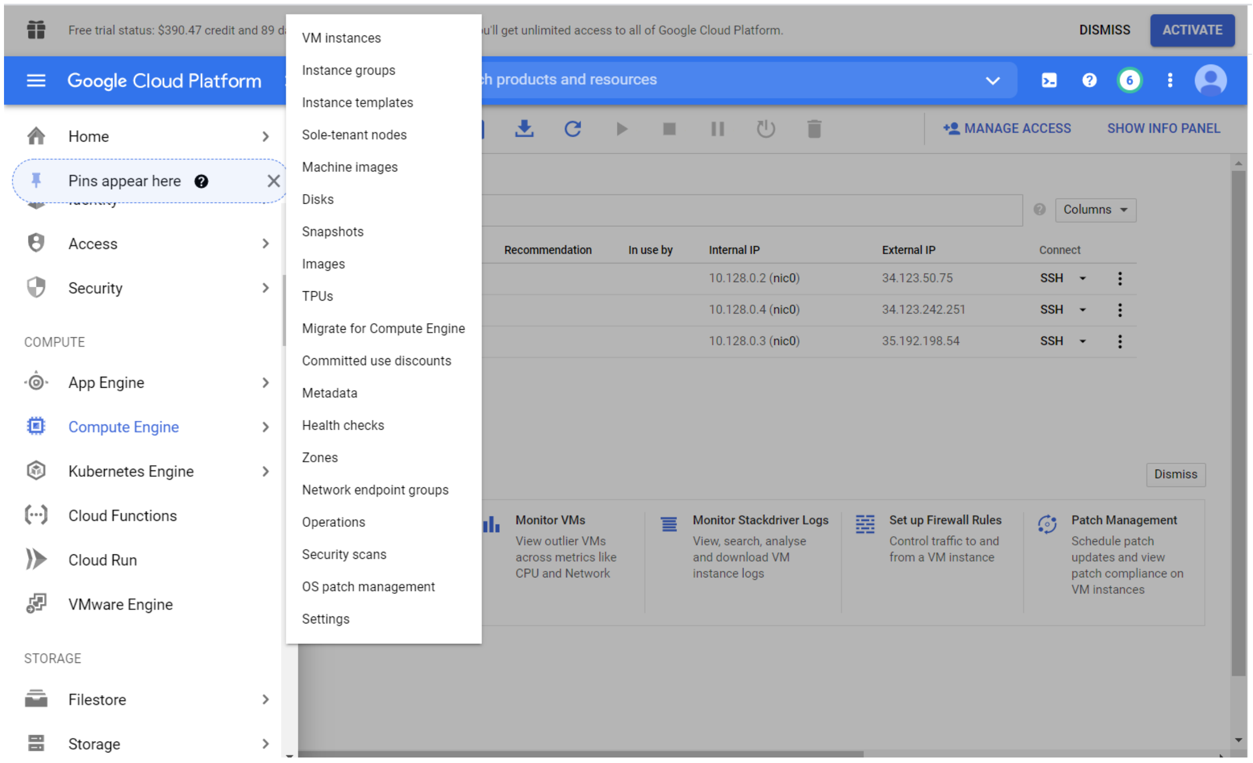


Name	Region	Zone	Total worker nodes	Scheduled deletion	Cloud Storage staging bucket
abhi	us-central1	us-central1-c	2	Off	dataproc-staging-us-central1-203201907885-sirszhan

No clusters selected

Please select at least one resource.

- Select "COMPUTE ENGINE" option here. It contains the main node and the worker nodes with all the details specific to master or worker nodes.



Free trial status: $390.47 credit and 89 d… u'll get unlimited access to all of Google Cloud Platform.

Recommendation	In use by	Internal IP	External IP	Connect
		10.128.0.2 (nic0)	34.123.50.75	SSH
		10.128.0.4 (nic0)	34.123.242.251	SSH
		10.128.0.3 (nic0)	35.192.198.54	SSH

Monitor VMs
View outlier VMs across metrics like CPU and Network

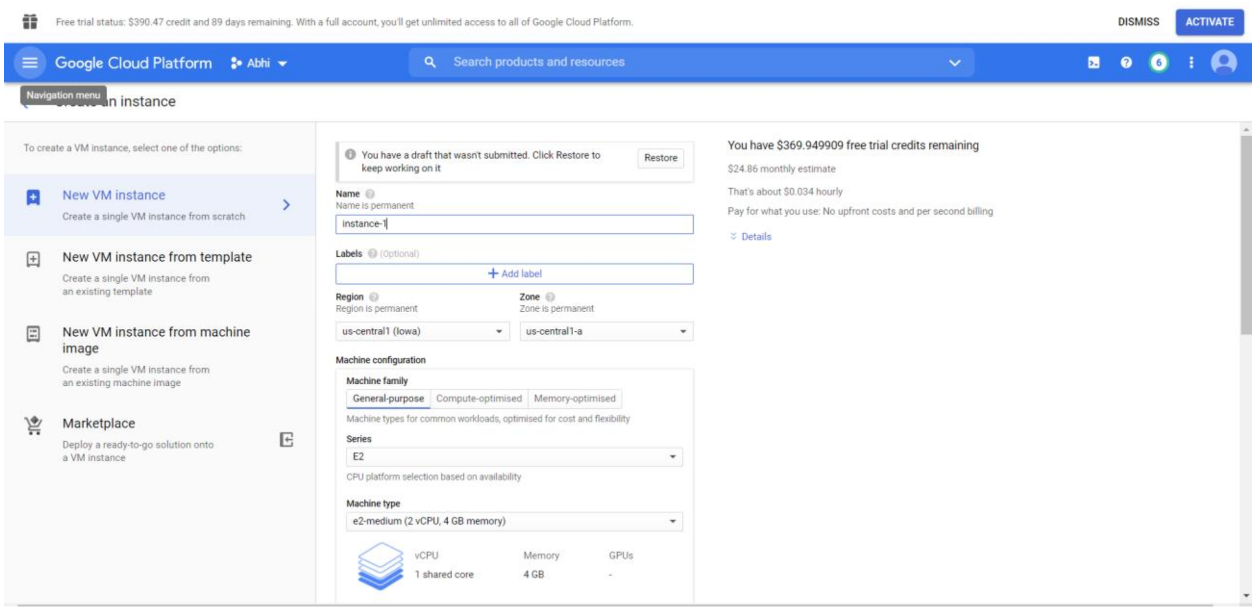
Monitor Stackdriver Logs
View, search, analyse and download VM instance logs

Set up Firewall Rules
Control traffic to and from a VM instance

Patch Management
Schedule patch updates and view patch compliance on VM instances

- Create VM instances with name and other required details.

Execution on Standalone and Spark Cluster



Free trial status: $390.47 credit and 89 days remaining. With a full account, you'll get unlimited access to all of Google Cloud Platform.

Create an instance

To create a VM instance, select one of the options:

New VM instance — Create a single VM instance from scratch

New VM instance from template — Create a single VM instance from an existing template

New VM instance from machine image — Create a single VM instance from an existing machine image

Marketplace — Deploy a ready-to-go solution onto a VM instance

You have a draft that wasn't submitted. Click Restore to keep working on it

Name: instance-1 (Name is permanent)

Labels (Optional)

Region: us-central1 (Iowa) (Region is permanent)

Zone: us-central1-a (Zone is permanent)

Machine configuration

Machine family: General-purpose | Compute-optimised | Memory-optimised

Machine types for common workloads, optimised for cost and flexibility

Series: E2

CPU platform selection based on availability

Machine type: e2-medium (2 vCPU, 4 GB memory)

vCPU	Memory	GPUs
1 shared core	4 GB	-

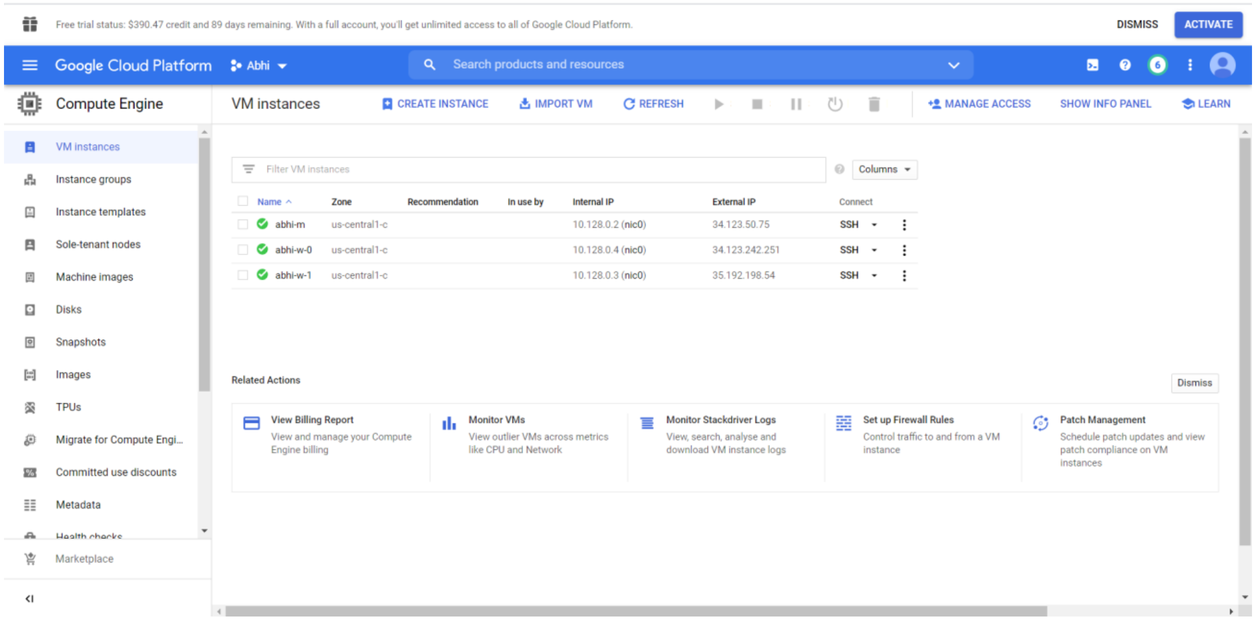
You have $369.949909 free trial credits remaining

$24.86 monthly estimate

That's about $0.034 hourly

Pay for what you use: No upfront costs and per second billing

- Select the master node "Abhi" or the worker nodes on which we are planning to execute or jobs or scripts. SSH option has been given in the picture click on SSH to access the terminal to execute scripts for master or worker nodes.



Free trial status: $390.47 credit and 89 days remaining. With a full account, you'll get unlimited access to all of Google Cloud Platform.

Name	Zone	Recommendation	In use by	Internal IP	External IP	Connect
abhi-m	us-central1-c			10.128.0.2 (nic0)	34.123.50.75	SSH
abhi-w-0	us-central1-c			10.128.0.4 (nic0)	34.123.242.251	SSH
abhi-w-1	us-central1-c			10.128.0.3 (nic0)	35.192.198.54	SSH

Related Actions

View Billing Report — View and manage your Compute Engine billing

Monitor VMs — View outlier VMs across metrics like CPU and Network

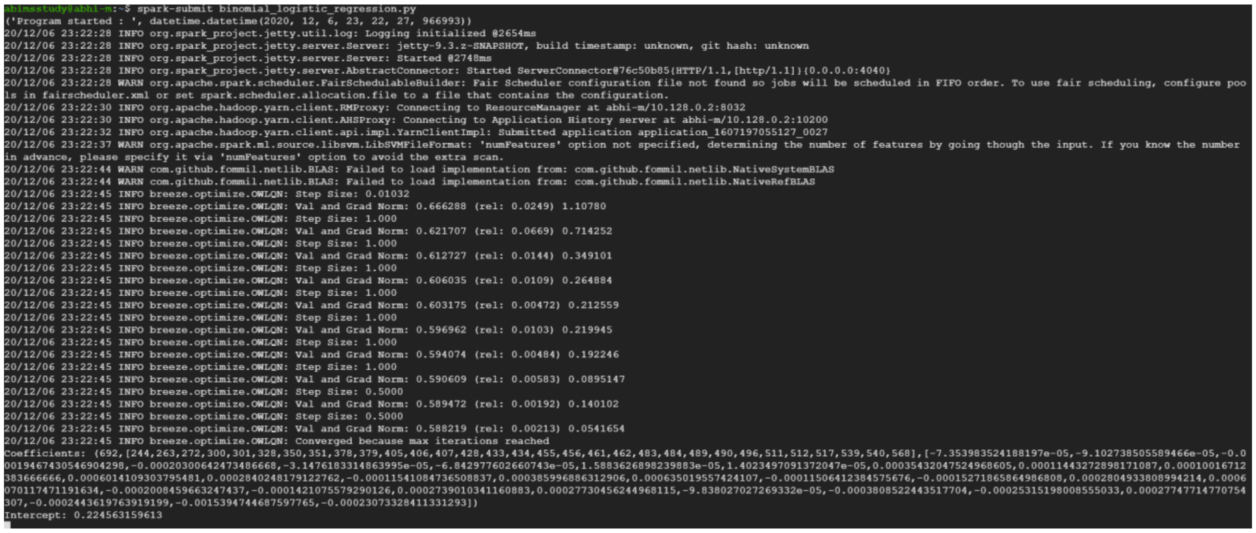
Monitor Stackdriver Logs — View, search, analyse and download VM instance logs

Set up Firewall Rules — Control traffic to and from a VM instance

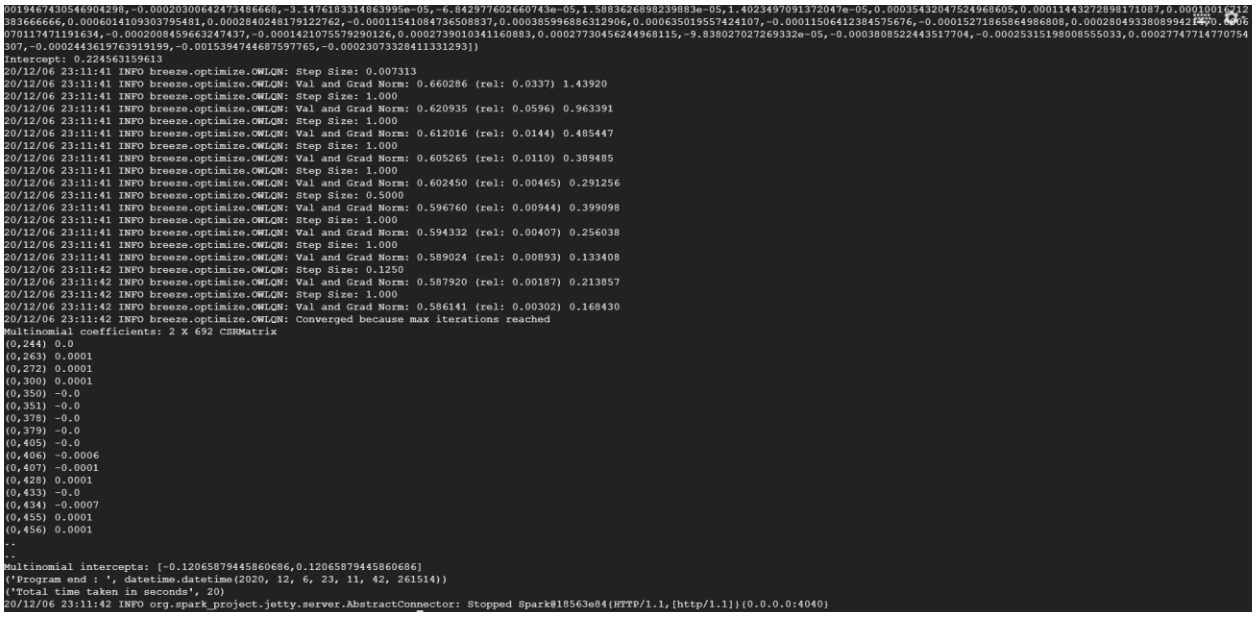
Patch Management — Schedule patch updates and view patch compliance on VM instances

- Command: **spark-submit binomial_logistic_regression.py**
  Above command is executed to execute the python file with small dataset in a Standalone system.
  Small Dataset:

Execution on Standalone and Spark Cluster



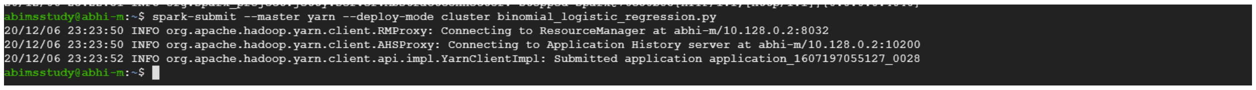
```
abimsstudy@abhi-m:~$ spark-submit binomial_logistic_regression.py
('Program started : ', datetime.datetime(2020, 12, 6, 23, 22, 27, 966993))
20/12/06 23:22:28 INFO org.spark_project.jetty.util.log: Logging initialized @2654ms
20/12/06 23:22:28 INFO org.spark_project.jetty.server.Server: jetty-9.3.z-SNAPSHOT, build timestamp: unknown, git hash: unknown
20/12/06 23:22:28 INFO org.spark_project.jetty.server.Server: Started @2748ms
20/12/06 23:22:28 INFO org.spark_project.jetty.server.AbstractConnector: Started ServerConnector@76c50b85{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
20/12/06 23:22:28 WARN org.apache.spark.scheduler.FairSchedulableBuilder: Fair Scheduler configuration file not found so jobs will be scheduled in FIFO order. To use fair scheduling, configure poo
ls in fairscheduler.xml or set spark.scheduler.allocation.file to a file that contains the configuration.
20/12/06 23:22:30 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at abhi-m/10.128.0.2:8032
20/12/06 23:22:30 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at abhi-m/10.128.0.2:10200
20/12/06 23:22:32 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1607197055127_0027
20/12/06 23:22:37 WARN org.apache.spark.ml.source.libsvm.LibSVMFileFormat: 'numFeatures' option not specified, determining the number of features by going though the input. If you know the number
in advance, please specify it via 'numFeatures' option to avoid the extra scan.
20/12/06 23:22:44 WARN com.github.fommil.netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
20/12/06 23:22:44 WARN com.github.fommil.netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 0.01032
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.666288 (rel: 0.0249) 1.10780
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.621707 (rel: 0.0669) 0.714252
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.612727 (rel: 0.0144) 0.349101
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.606035 (rel: 0.0109) 0.264884
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.603175 (rel: 0.00472) 0.212559
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.596962 (rel: 0.0103) 0.219945
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.594074 (rel: 0.00484) 0.192246
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.590609 (rel: 0.00583) 0.0895147
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 0.5000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.589472 (rel: 0.00192) 0.140102
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Step Size: 0.5000
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.588219 (rel: 0.00213) 0.0541654
20/12/06 23:22:45 INFO breeze.optimize.OWLQN: Converged because max iterations reached
Coefficients: (692,[244,263,272,300,301,328,350,351,378,379,405,406,407,428,433,434,455,456,461,462,483,484,489,490,496,511,512,517,539,540,568],[-7.353983524188197e-05,-9.102738505589466e-05,-0.0
0019467430546904298,-0.0002030064247348668,-3.1476183314863995e-05,-6.842977602660743e-05,1.5883626898239883e-05,1.4023497091372047e-05,0.0003543204752496860 5,0.00011443272898171087,0.00010016712
383666666,0.0006014109303795481,0.0002840248179122762,-0.00011541084736508837,0.000385996886312906,0.000635019557424107,-0.00011506412384575676,-0.00015271865864986808,0.0002804933808994214,0.0006
070117471191634,-0.0002008459663247437,-0.0001421075579290126,0.0002739010341160883,0.00027730456244968115,-9.838027027269332e-05,-0.0003808522443517704,-0.00025315198008555033,0.00027747714770754
307,-0.0002443619763919199,-0.0015394744687597765,-0.00023073328411331293])
Intercept: 0.224563159613
```



```
0019467430546904298,-0.0002030064247348668,-3.1476183314863995e-05,-6.842977602660743e-05,1.5883626898239883e-05,1.4023497091372047e-05,0.00035432047524968605,0.00011443272898171087,0.00010016712
383666666,0.0006014109303795481,0.0002840248179122762,-0.00011541084736508837,0.000385996886312906,0.000635019557424107,-0.00011506412384575676,-0.00015271865864986808,0.0002804933808994214,0.0006
070117471191634,-0.0002008459663247437,-0.0001421075579290126,0.0002739010341160883,0.00027730456244968115,-9.838027027269332e-05,-0.0003808522443517704,-0.00025315198008555033,0.00027747714770754
307,-0.0002443619763919199,-0.0015394744687597765,-0.00023073328411331293])
Intercept: 0.224563159613
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Step Size: 0.007313
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.660286 (rel: 0.0337) 1.43920
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.620935 (rel: 0.0596) 0.963391
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.612016 (rel: 0.0144) 0.485447
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.605265 (rel: 0.0110) 0.389485
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.602450 (rel: 0.00465) 0.291256
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Step Size: 0.5000
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.596760 (rel: 0.00944) 0.399098
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.594332 (rel: 0.00407) 0.256038
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:11:41 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.589024 (rel: 0.00893) 0.133408
20/12/06 23:11:42 INFO breeze.optimize.OWLQN: Step Size: 0.1250
20/12/06 23:11:42 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.587920 (rel: 0.00187) 0.213857
20/12/06 23:11:42 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:11:42 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.586141 (rel: 0.00302) 0.168430
20/12/06 23:11:42 INFO breeze.optimize.OWLQN: Converged because max iterations reached
Multinomial coefficients: 2 X 692 CSRMatrix
(0,244) 0.0
(0,263) 0.0001
(0,272) 0.0001
(0,300) 0.0001
(0,350) -0.0
(0,351) -0.0
(0,378) -0.0
(0,379) -0.0
(0,405) -0.0
(0,406) -0.0006
(0,407) -0.0001
(0,428) 0.0001
(0,433) -0.0
(0,434) -0.0007
(0,455) 0.0001
(0,456) 0.0001
..
..
Multinomial intercepts: [-0.12065879445860686,0.12065879445860686]
('Program end : ', datetime.datetime(2020, 12, 6, 23, 11, 42, 261514))
('Total time taken in seconds', 20)
20/12/06 23:11:42 INFO org.spark_project.jetty.server.AbstractConnector: Stopped Spark@18563e84{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
```

- Command: **spark-submit –deploy-mode cluster binomial_logistic_regression.py**
  Above command is executed to execute the python file with small dataset in a Spark Cluster system.

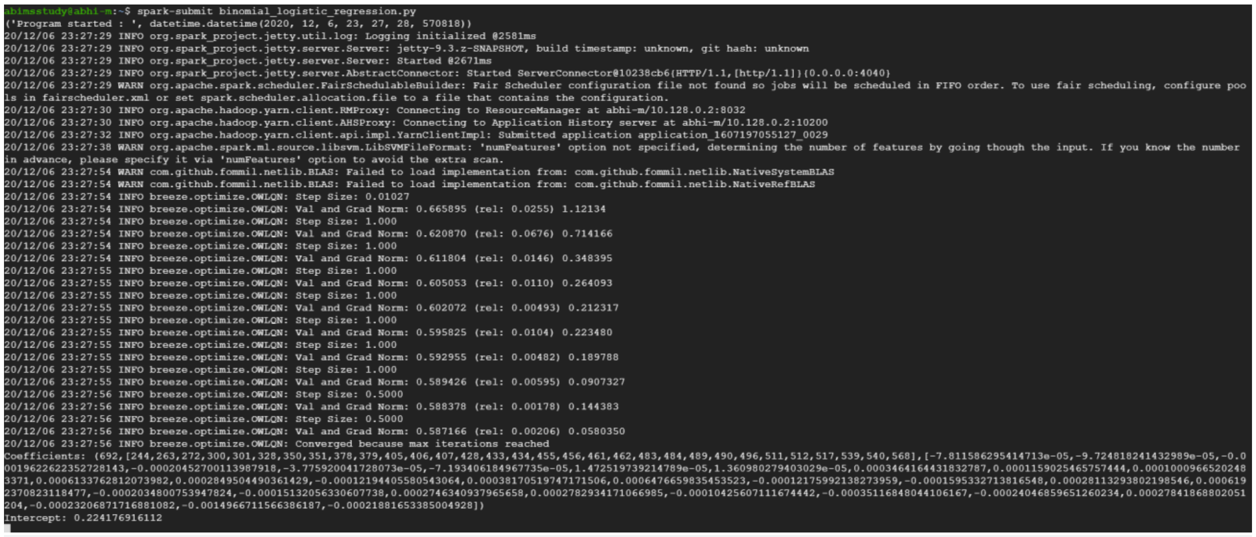
  Small Dataset:



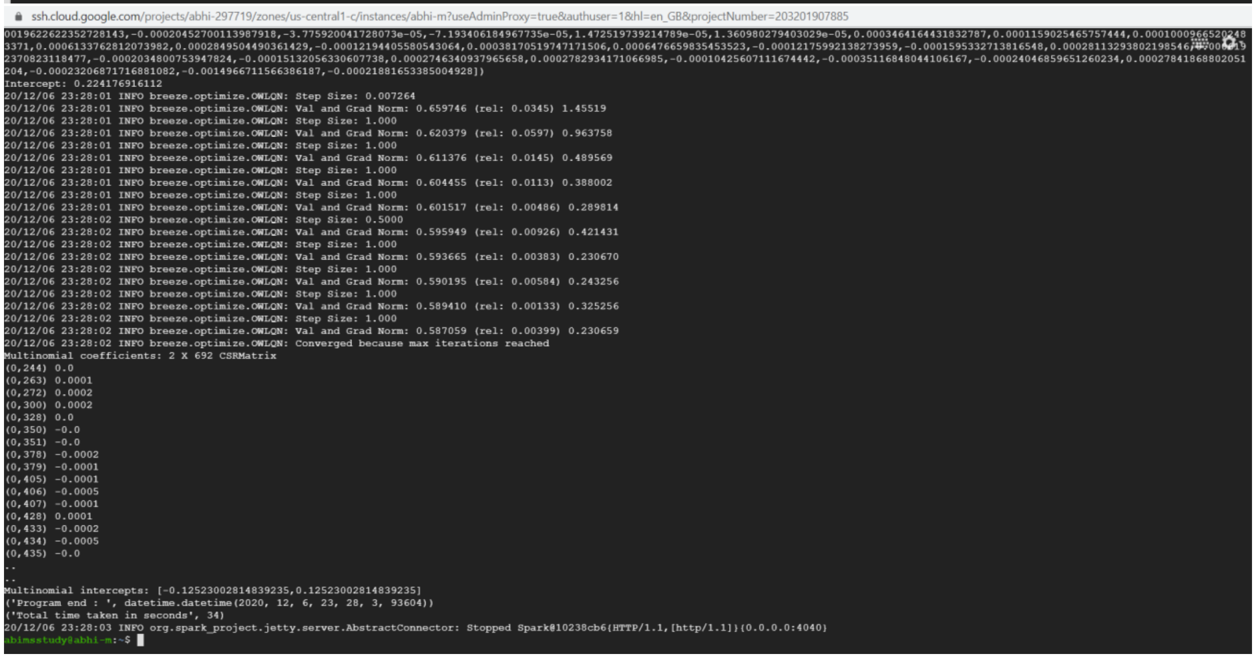
```
abimsstudy@abhi-m:~$ spark-submit --master yarn --deploy-mode cluster binomial_logistic_regression.py
20/12/06 23:23:50 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at abhi-m/10.128.0.2:8032
20/12/06 23:23:50 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at abhi-m/10.128.0.2:10200
20/12/06 23:23:52 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1607197055127_0028
abimsstudy@abhi-m:~$
```

- Command: **spark-submit binomial_logistic_regression.py**
  Above command is executed to execute the python file with small dataset in a Standalone system.
  Large Dataset:

Execution on Standalone and Spark Cluster



```
abimsstudy@abhi-m:~$ spark-submit binomial_logistic_regression.py
('Program started : ', datetime.datetime(2020, 12, 6, 23, 27, 28, 570818))
20/12/06 23:27:29 INFO org.spark_project.jetty.util.log: Logging initialized @2581ms
20/12/06 23:27:29 INFO org.spark_project.jetty.server.Server: jetty-9.3.z-SNAPSHOT, build timestamp: unknown, git hash: unknown
20/12/06 23:27:29 INFO org.spark_project.jetty.server.Server: Started @2671ms
20/12/06 23:27:29 INFO org.spark_project.jetty.server.AbstractConnector: Started ServerConnector@10238cb6{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
20/12/06 23:27:29 WARN org.apache.spark.scheduler.FairSchedulableBuilder: Fair Scheduler configuration file not found so jobs will be scheduled in FIFO order. To use fair scheduling, configure poo
ls in fairscheduler.xml or set spark.scheduler.allocation.file to a file that contains the configuration.
20/12/06 23:27:30 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at abhi-m/10.128.0.2:8032
20/12/06 23:27:30 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at abhi-m/10.128.0.2:10200
20/12/06 23:27:32 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1607197055127_0029
20/12/06 23:27:38 WARN org.apache.spark.ml.source.libsvm.LibSVMFileFormat: 'numFeatures' option not specified, determining the number of features by going though the input. If you know the number
in advance, please specify it via 'numFeatures' option to avoid the extra scan.
20/12/06 23:27:54 WARN com.github.fommil.netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
20/12/06 23:27:54 WARN com.github.fommil.netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
20/12/06 23:27:54 INFO breeze.optimize.OWLQN: Step Size: 0.01027
20/12/06 23:27:54 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.665895 (rel: 0.0255) 1.12134
20/12/06 23:27:54 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:27:54 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.620870 (rel: 0.0676) 0.714166
20/12/06 23:27:54 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:27:54 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.611804 (rel: 0.0146) 0.348395
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.605053 (rel: 0.0110) 0.264093
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.602072 (rel: 0.00493) 0.212317
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.595825 (rel: 0.0104) 0.223480
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.592955 (rel: 0.00482) 0.189788
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:27:55 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.589426 (rel: 0.00595) 0.0907327
20/12/06 23:27:56 INFO breeze.optimize.OWLQN: Step Size: 0.5000
20/12/06 23:27:56 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.588378 (rel: 0.00178) 0.144383
20/12/06 23:27:56 INFO breeze.optimize.OWLQN: Step Size: 0.5000
20/12/06 23:27:56 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.587166 (rel: 0.00206) 0.0580350
20/12/06 23:27:56 INFO breeze.optimize.OWLQN: Converged because max iterations reached
Coefficients: (692,[244,263,272,300,301,328,350,351,378,379,405,406,407,428,433,434,455,456,461,462,483,484,489,490,496,511,512,517,539,540,568],[-7.811586295414713e-05,-9.724818241432989e-05,-0.0
0019622622352728143,-0.00020452700113987918,-3.775920041728073e-05,-7.193406184967735e-05,1.472519739214789e-05,1.360980279403029e-05,0.0003464164431832787,0.0001159025465757444,0.0001000966520248
3371,0.0006133762812073982,0.0002849504490361429,-0.00012194405580543064,0.00038170519747171506,0.0006476659835453523,-0.00012175992138273959,-0.0001595332713816548,0.00028113293802198546,0.000619
2370823118477,-0.0002034800753947824,-0.0001513205633060 7738,0.0002746340937965658,0.0002782934171066985,-0.00010425607111674442,-0.0003511684804410 6167,-0.00024046859651260234,0.00027841868802051
204,-0.00023206871716881082,-0.0014966711566386187,-0.00021881653385004928])
Intercept: 0.224176916112
```



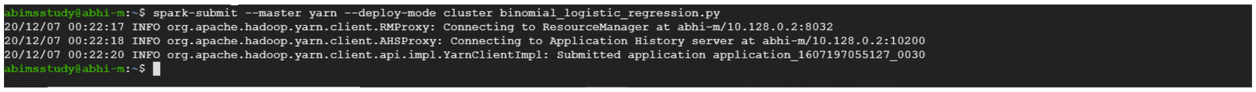
ssh.cloud.google.com/projects/abhi-297719/zones/us-central1-c/instances/abhi-m?useAdminProxy=true&authuser=1&hl=en_GB&projectNumber=203201907885

```
0019622622352728143,-0.00020452700113987918,-3.775920041728073e-05,-7.193406184967735e-05,1.472519739214789e-05,1.360980279403029e-05,0.0003464164431832787,0.0001159025465757444,0.0001000966520248
3371,0.0006133762812073982,0.0002849504490361429,-0.00012194405580543064,0.00038170519747171506,0.0006476659835453523,-0.00012175992138273959,-0.0001595332713816548,0.00028113293802198546,0.000619
2370823118477,-0.0002034800753947824,-0.00015132056330607738,0.0002746340937965658,0.0002782934171066985,-0.00010425607111674442,-0.00035116848044106167,-0.00024046859651260234,0.00027841868802051
204,-0.00023206871716881082,-0.0014966711566386187,-0.00021881653385004928])
Intercept: 0.224176916112
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Step Size: 0.007264
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.659746 (rel: 0.0345) 1.45519
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.620379 (rel: 0.0597) 0.963758
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.611376 (rel: 0.0145) 0.489569
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.604455 (rel: 0.0113) 0.388002
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:28:01 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.601517 (rel: 0.00486) 0.289814
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Step Size: 0.5000
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.595949 (rel: 0.00926) 0.421431
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.593665 (rel: 0.00383) 0.230670
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.590195 (rel: 0.00584) 0.243256
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.589410 (rel: 0.00133) 0.325256
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.587059 (rel: 0.00399) 0.230659
20/12/06 23:28:02 INFO breeze.optimize.OWLQN: Converged because max iterations reached
Multinomial coefficients: 2 X 692 CSRMatrix
(0,244) 0.0
(0,263) 0.0001
(0,272) 0.0002
(0,300) 0.0002
(0,328) 0.0
(0,350) -0.0
(0,351) -0.0
(0,378) -0.0002
(0,379) -0.0001
(0,405) -0.0001
(0,406) -0.0005
(0,407) -0.0001
(0,428) 0.0001
(0,433) -0.0002
(0,434) -0.0005
(0,435) -0.0
..
..
Multinomial intercepts: [-0.12523002814839235,0.12523002814839235]
('Program end : ', datetime.datetime(2020, 12, 6, 23, 28, 3, 93604))
('Total time taken in seconds', 34)
20/12/06 23:28:03 INFO org.spark_project.jetty.server.AbstractConnector: Stopped Spark@10238cb6{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
abimsstudy@abhi-m:~$
```

- Command: **spark-submit –deploy-mode cluster binomial_logistic_regression.py**

  Above command is executed to execute the python file with small dataset in a Spark Cluster system.

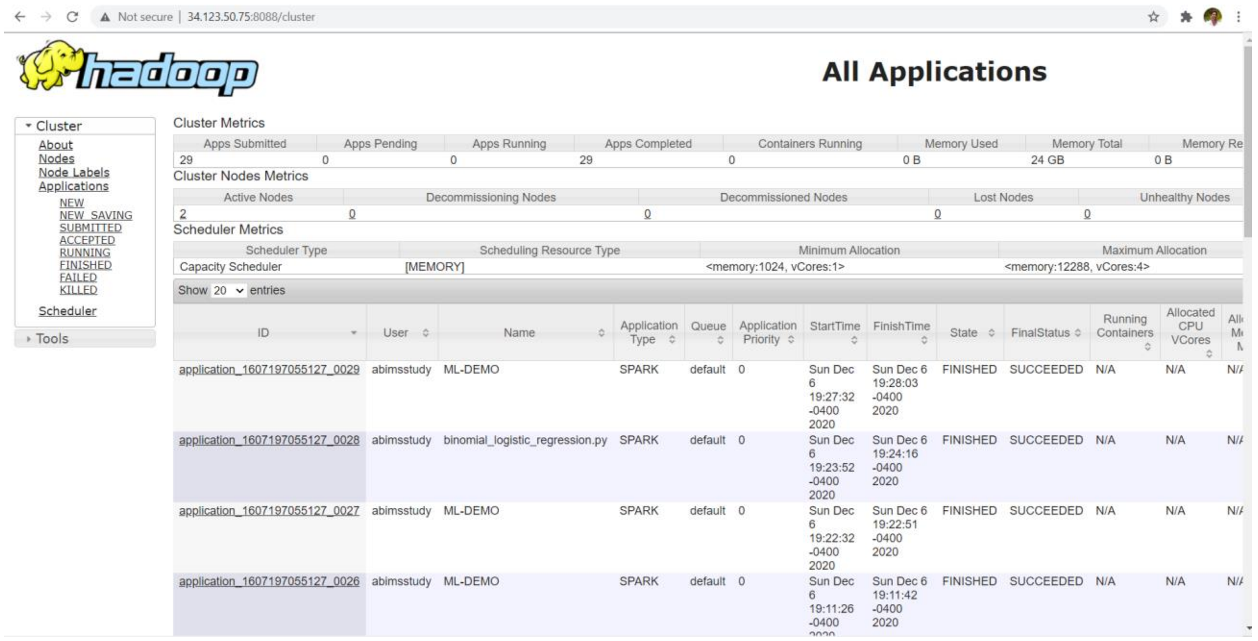
  Large Dataset:



```
abimsstudy@abhi-m:~$ spark-submit --master yarn --deploy-mode cluster binomial_logistic_regression.py
20/12/07 00:22:17 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at abhi-m/10.128.0.2:8032
20/12/07 00:22:18 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at abhi-m/10.128.0.2:10200
20/12/07 00:22:20 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1607197055127_0030
abimsstudy@abhi-m:~$
```

http://34.123.50.75:8088/cluster is clicked to access the below attached screen which shows the execution details of the python script being executed and all other required cluster details.

Execution on Standalone and Spark Cluster



Not secure | 34.123.50.75:8088/cluster

All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Re
29	0	0	29	0	0 B	24 GB	0 B

Cluster Nodes Metrics

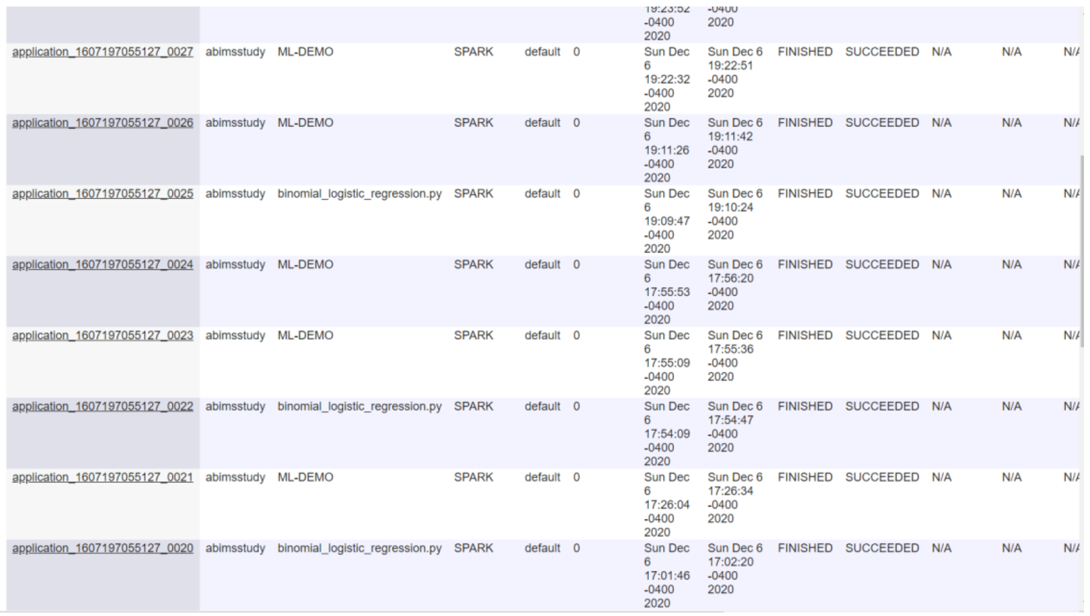
Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes
2	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:12288, vCores:4>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	All M N
application_1607197055127_0029	abimsstudy	ML-DEMO	SPARK	default	0	Sun Dec 6 19:27:32 -0400 2020	Sun Dec 6 19:28:03 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0028	abimsstudy	binomial_logistic_regression.py	SPARK	default	0	Sun Dec 6 19:23:52 -0400 2020	Sun Dec 6 19:24:16 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0027	abimsstudy	ML-DEMO	SPARK	default	0	Sun Dec 6 19:22:32 -0400 2020	Sun Dec 6 19:22:51 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0026	abimsstudy	ML-DEMO	SPARK	default	0	Sun Dec 6 19:11:26 -0400	Sun Dec 6 19:11:42 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/



						19:23:52 -0400 2020	-0400 2020					
application_1607197055127_0027	abimsstudy	ML-DEMO	SPARK	default	0	Sun Dec 6 19:22:32 -0400 2020	Sun Dec 6 19:22:51 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0026	abimsstudy	ML-DEMO	SPARK	default	0	Sun Dec 6 19:11:26 -0400 2020	Sun Dec 6 19:11:42 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0025	abimsstudy	binomial_logistic_regression.py	SPARK	default	0	Sun Dec 6 19:09:47 -0400 2020	Sun Dec 6 19:10:24 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0024	abimsstudy	ML-DEMO	SPARK	default	0	Sun Dec 6 17:55:53 -0400 2020	Sun Dec 6 17:56:20 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0023	abimsstudy	ML-DEMO	SPARK	default	0	Sun Dec 6 17:55:09 -0400 2020	Sun Dec 6 17:55:36 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0022	abimsstudy	binomial_logistic_regression.py	SPARK	default	0	Sun Dec 6 17:54:09 -0400 2020	Sun Dec 6 17:54:47 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0021	abimsstudy	ML-DEMO	SPARK	default	0	Sun Dec 6 17:26:04 -0400 2020	Sun Dec 6 17:26:34 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/
application_1607197055127_0020	abimsstudy	binomial_logistic_regression.py	SPARK	default	0	Sun Dec 6 17:01:46 -0400 2020	Sun Dec 6 17:02:20 -0400 2020	FINISHED	SUCCEEDED	N/A	N/A	N/

7. **Code**:

```
from pyspark.ml.classification import LogisticRegression

from datetime import datetime

start_time = datetime.now()

print("Program started : ",start_time)
```

Execution on Standalone and Spark Cluster

```
spark = SparkSession.builder.appName("ML-DEMO").getOrCreate()

#training_lr = spark.read.format("libsvm").load("data_mllib_sample_libsvm_data.txt")

# Load training data

training_lr = spark.read.format("libsvm").load("data_mllib_sample_libsvm_data_large.txt")

logreg = LogisticRegression(maxIter=10, regParam=0.3, elasticNetParam=0.8)

# Fit the model

lrModel = logreg.fit(training_lr)

# Print the coefficients and intercept for logistic regression

print("Coefficients: " + str(lrModel.coefficients))

print("Intercept: " + str(lrModel.intercept))

# We can also use the multinomial family for binary classification

mlogreg = LogisticRegression(maxIter=10, regParam=0.3, elasticNetParam=0.8,
family="multinomial")

# Fit the model

mlrModel = mlogreg.fit(training_lr)

# Print the coefficients and intercepts for logistic regression with multinomial family

print("Multinomial coefficients: " + str(mlrModel.coefficientMatrix))

print("Multinomial intercepts: " + str(mlrModel.interceptVector))

end_time = datetime.now()

print("Program end : ",end_time)

time_taken = end_time -start_time

print("Total time taken in minuntes",int(time_taken.seconds))
```

8. **References**:

https://scikit-learn.org/stable/auto_examples/linear_model/plot_logistic.html

https://spark.apache.org/docs/latest/mllib-linear-methods.html#logistic-regression

9. **Conclusion**:

Here binomial_logistic_regression.py is executed with small and large datasets when its executes on Standalone system more time is consumed and when we run on Spark cluster where execution is divided among clusters and less time is consumed.