PySpark on Dataproc

Apache Spark is an open source all-in-one data processing engine for big data workflows. It allows high performance batch processing, real-time streaming, SQL-querying, interactive analytics, data science and machine learning. It is written in Scala and supports variety of modern languages involving Python, Java, R and SQL. PySpark is an API to deal with Spark works in Python.

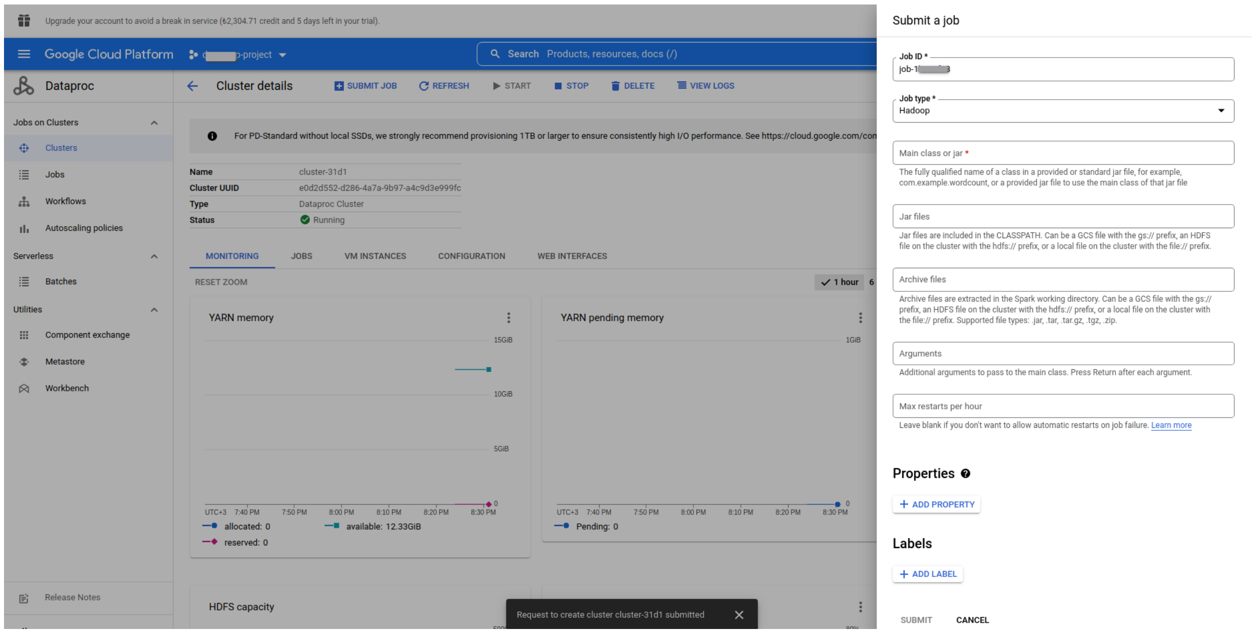
PySpark can be run locally or on cloud with variety of architectures that would be time consuming or risky while operating. Dataproc service of the GCP makes easier to handle the infrastructure for ETL tasks and data science as well as for Apache Spark in a fully managed and highly scalable manner.

Step0: Prerequisites

To follow this tutorial, on need to have an active GCP account with activated Dataproc, BigQuery and Compute Engine APIs. Moreover, a dataset on BigQuery was prepared with dbt in previous tutorial for the project assignment of Data Engineering Zoomcamp by Datatalks.Club. it is also necessary install and initialize Google Cloud CLI.

Step1: Creating a Dataproc cluster

It is a common practice to hande Apache Spark workloads with a flexible infrastructure that can scale immediately when needed. Hence, Dataproc build them on clusters. So, the first step is to build one by clicking on *Create Cluster* button on DataProc page and select the option for running on a compute engine (the other option is for power users who want to use Kubernetes).



Upgrade your account to avoid a break in service (₺2,304.71 credit and 5 days left in your trial).

d[redacted]p-project

For PD-Standard without local SSDs, we strongly recommend provisioning 1TB or larger to ensure consistently high I/O performance. See https://cloud.google.com/con

Name	cluster-31d1
Cluster UUID	e0d2d552-d286-4a7a-9b97-a4c9d3e999fc
Type	Dataproc Cluster
Status	Running

allocated: 0
available: 12.33GiB
reserved: 0
Pending: 0

Request to create cluster cluster-31d1 submitted

Job ID *
job-1[redacted]3

Job type *
Hadoop

Main class or jar *
The fully qualified name of a class in a provided or standard jar file, for example, com.example.wordcount, or a provided jar file to use the main class of that jar file

Jar files
Jar files are included in the CLASSPATH. Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix.

Archive files
Archive files are extracted in the Spark working directory. Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix. Supported file types: .jar, .tar, .tar.gz, .tgz, .zip.

Arguments
Additional arguments to pass to the main class. Press Return after each argument.

Max restarts per hour
Leave blank if you don't want to allow automatic restarts on job failure. Learn more

- It is better to build your cluster on location where your BigQuery dataset is hosted.
- Even a single (1 master, 0 workers) node cluster is satisfactory for this simple tutorial.
- It would be helpfull to involve Jupyter Notebook (and also Docker maybe) as optional components
- Default values for the rest of the settings are acceptable.

Step2: Transferring ML code

Having an up and running Dataproc cluster, you will have 2 more buckets called:

- dataproc-staging-...
- dataproc-temp-...

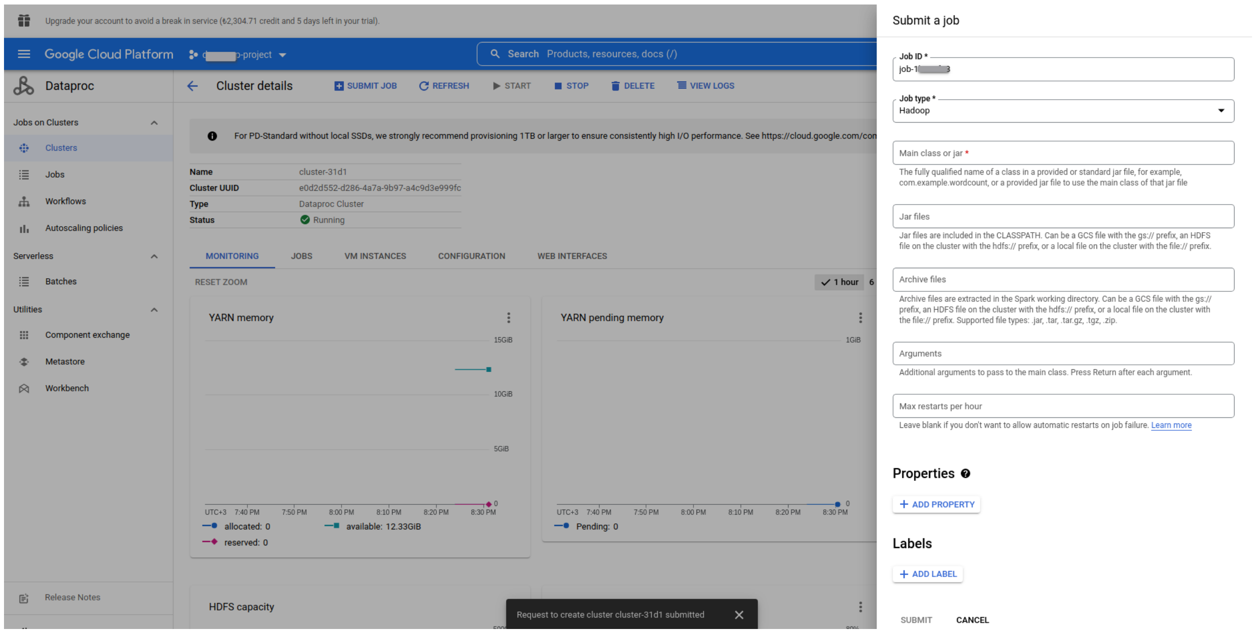
The next step is to transfer the machine learning code to the staging bucket using:

```
gsutil cp your_ml_code.py gs://your-bucket-name
```

Then, navigate trough your code file on GCP bucket to copy the URL.

Step3: Submitting the ML task

Jump in to the Dataproc page on GCP and click on *Submit Job* button that will open a window:



Upgrade your account to avoid a break in service (₺2,304.71 credit and 5 days left in your trial).

For PD-Standard without local SSDs, we strongly recommend provisioning 1TB or larger to ensure consistently high I/O performance. See https://cloud.google.com/con

Name	cluster-31d1
Cluster UUID	e0d2d552-d286-4a7a-9b97-a4c9d3e999fc
Type	Dataproc Cluster
Status	Running

Request to create cluster cluster-31d1 submitted

Job ID *: job-1[redacted]3

Job type *: Hadoop

Main class or jar *

The fully qualified name of a class in a provided or standard jar file, for example, com.example.wordcount, or a provided jar file to use the main class of that jar file

Jar files

Jar files are included in the CLASSPATH. Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix.

Archive files

Archive files are extracted in the Spark working directory. Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix. Supported file types: .jar, .tar, .tar.gz, .tgz, .zip.

Arguments

Additional arguments to pass to the main class. Press Return after each argument.

Max restarts per hour

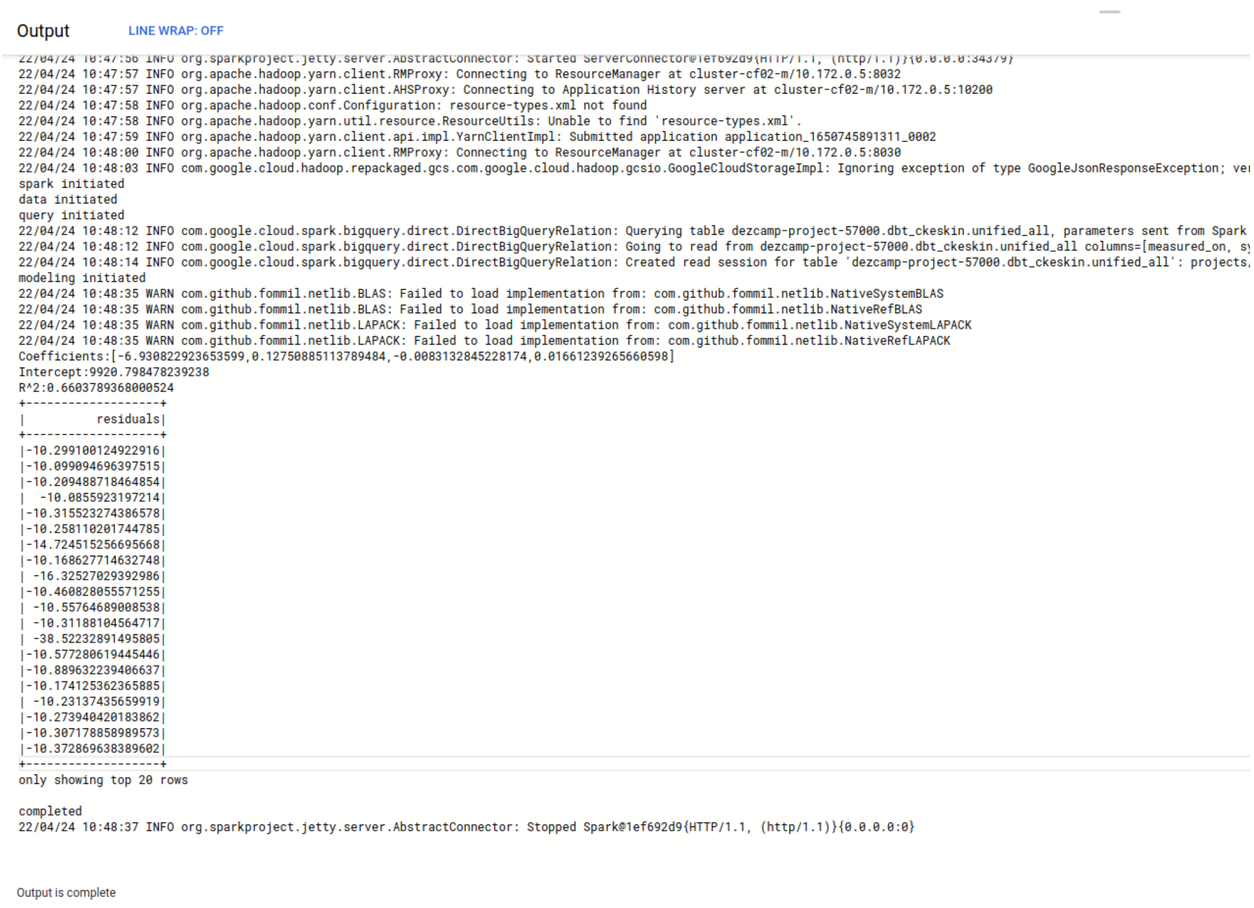
Leave blank if you don't want to allow automatic restarts on job failure. Learn more

Here, set:

- Job type as PySpark
- Ling for your ML code copied above (take care of the notation: gs://prefix...)
- Attach additional python file links if you have
- Insert `gs://spark-lib/bigquery/spark-bigquery-latest_2.12.jar` to involve required jar files.
- Submit the job

Step: Results

While the job is executed, the logs can be seen on Output tab where the results are also displayed.



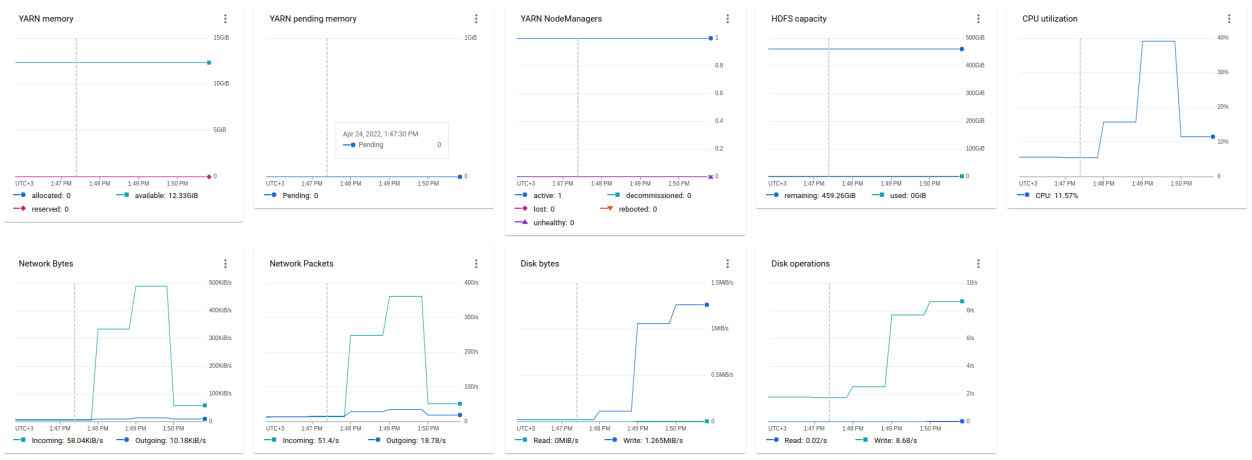
Output   LINE WRAP: OFF

```
22/04/24 10:47:56 INFO org.sparkproject.jetty.server.AbstractConnector: Started ServerConnector@1ef692d9{HTTP/1.1, (http/1.1)}{0.0.0.0:34379}
22/04/24 10:47:57 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at cluster-cf02-m/10.172.0.5:8032
22/04/24 10:47:57 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at cluster-cf02-m/10.172.0.5:10200
22/04/24 10:47:58 INFO org.apache.hadoop.conf.Configuration: resource-types.xml not found
22/04/24 10:47:58 INFO org.apache.hadoop.yarn.util.resource.ResourceUtils: Unable to find 'resource-types.xml'.
22/04/24 10:47:59 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1650745891311_0002
22/04/24 10:48:00 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at cluster-cf02-m/10.172.0.5:8030
22/04/24 10:48:03 INFO com.google.cloud.hadoop.repackaged.gcs.com.google.cloud.hadoop.gcsio.GoogleCloudStorageImpl: Ignoring exception of type GoogleJsonResponseException; ve
spark initiated
data initiated
query initiated
22/04/24 10:48:12 INFO com.google.cloud.spark.bigquery.direct.DirectBigQueryRelation: Querying table dezcamp-project-57000.dbt_ckeskin.unified_all, parameters sent from Spark
22/04/24 10:48:12 INFO com.google.cloud.spark.bigquery.direct.DirectBigQueryRelation: Going to read from dezcamp-project-57000.dbt_ckeskin.unified_all columns=[measured_on, s
22/04/24 10:48:14 INFO com.google.cloud.spark.bigquery.direct.DirectBigQueryRelation: Created read session for table 'dezcamp-project-57000.dbt_ckeskin.unified_all': projects,
modeling initiated
22/04/24 10:48:35 WARN com.github.fommil.netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
22/04/24 10:48:35 WARN com.github.fommil.netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
22/04/24 10:48:35 WARN com.github.fommil.netlib.LAPACK: Failed to load implementation from: com.github.fommil.netlib.NativeSystemLAPACK
22/04/24 10:48:35 WARN com.github.fommil.netlib.LAPACK: Failed to load implementation from: com.github.fommil.netlib.NativeRefLAPACK
Coefficients:[-6.930822923653599,0.12750885113789484,-0.0083132845228174,0.01661239265660598]
Intercept:9920.798478239238
R^2:0.6603789368000524
+-------------------+
|          residuals|
+-------------------+
|-10.299100124922916|
|-10.099094696397515|
|-10.209488718464854|
|  -10.0855923197214|
|-10.315523274386578|
|-10.258110201744785|
|-14.724515256695668|
|-10.168627714632748|
| -16.32527029392986|
|-10.460828055571255|
| -10.55764689008538|
| -10.31188104564717|
| -38.52232891495805|
|-10.577280619445446|
|-10.889632239406637|
|-10.174125362365885|
| -10.23137435659919|
|-10.273940420183862|
|-10.307178858989573|
|-10.372869638389602|
+-------------------+
only showing top 20 rows

completed
22/04/24 10:48:37 INFO org.sparkproject.jetty.server.AbstractConnector: Stopped Spark@1ef692d9{HTTP/1.1, (http/1.1)}{0.0.0.0:0}
```

Output is complete

MOreover, while the cluster is running a task, it is possible to observ cluster workload and performance on *Job Details*


YARN memory
allocated: 0
available: 12.33GiB
reserved: 0
YARN pending memory
Pending: 0
YARN NodeManagers
active: 1
decommissioned: 0
lost: 0
rebooted: 0
unhealthy: 0
HDFS capacity
remaining: 459.26GiB
used: 0GiB
CPU utilization
CPU: 11.57%
Network Bytes
Incoming: 58.04KiB/s
Outgoing: 10.18KiB/s
Network Packets
Incoming: 51.4/s
Outgoing: 18.78/s
Disk bytes
Read: 0MiB/s
Write: 1.265MiB/s
Disk operations
Read: 0.02/s
Write: 8.68/s