CS4830 - BIG DATA LABORATORY
LAB 7
VISHAL RISHI MK - CH18B013

1. Write a producer.py file that reads the iris.csv line by line and writes each row into a particular topic in Kafka

The code *'producer.py'* reads the iris.csv line by line and writes the search row into a particular topic in Kafka. The internal IP of the Kafka instance is found using '*gcloud'* command from GCP terminal.

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to innate-shape-340909.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
ch18b013@cloudshell:~ (innate-shape-340909)$ gcloud compute instances list
NAME: instance-1
ZONE: us-central1-a
MACHINE_TYPE: e2-medium
PREEMPTIBLE:
INTERNAL_IP: 10.128.0.2
EXTERNAL_IP:
STATUS: TERMINATED

NAME: kafka-centos-1-vm
ZONE: us-central1-c
MACHINE_TYPE: n1-standard-1
PREEMPTIBLE:
INTERNAL_IP: 10.128.0.48
EXTERNAL_IP: 34.133.5.44
STATUS: RUNNING
```

• We add the *'iris.csv'* data file in the Kafka SSH.
• Then, we install pip, kafka and pyspark in the kafka instance. Then, we create the Kafka topic.

```
[ch18b013@kafka-centos-1-vm kafka]$ sudo bin/kafka-topics.sh --create --topic lab7_topic --bootstrap-server localhost:9092
WARNING: Due to limitations in metric names, topics with a period ('.') or underscore ('_') could collide. To avoid issues it is best to use either, but not both.
[2022-04-03 16:22:49,900] INFO Creating topic lab7_topic with configuration {} and initial partition assignment HashMap(0 -> ArrayBuffer(0)) (kafka.zk.AdminZkClient)
[2022-04-03 16:22:50,058] INFO [Controller id=0, targetBrokerId=0] Node 0 disconnected. (org.apache.kafka.clients.NetworkClient)
[2022-04-03 16:22:50,092] INFO [ReplicaFetcherManager on broker 0] Removed fetcher for partitions Set(lab7_topic-0) (kafka.server.ReplicaFetcherManager)
[2022-04-03 16:22:50,185] INFO [LogLoader partition=lab7_topic-0, dir=/tmp/kafka-logs] Loading producer state till offset 0 with message format version 2 (kafka.log.UnifiedLog$)
[2022-04-03 16:22:50,238] INFO Created log for partition lab7_topic-0 in /tmp/kafka-logs/lab7_topic-0 with properties {} (kafka.log.LogManager)
[2022-04-03 16:22:50,240] INFO [Partition lab7_topic-0 broker=0] No checkpointed highwatermark is found for partition lab7_topic-0 (kafka.cluster.Partition)
[2022-04-03 16:22:50,242] INFO [Partition lab7_topic-0 broker=0] Log loaded for partition lab7_topic-0 with initial high watermark 0 (kafka.cluster.Partition)
Created topic lab7_topic.
```

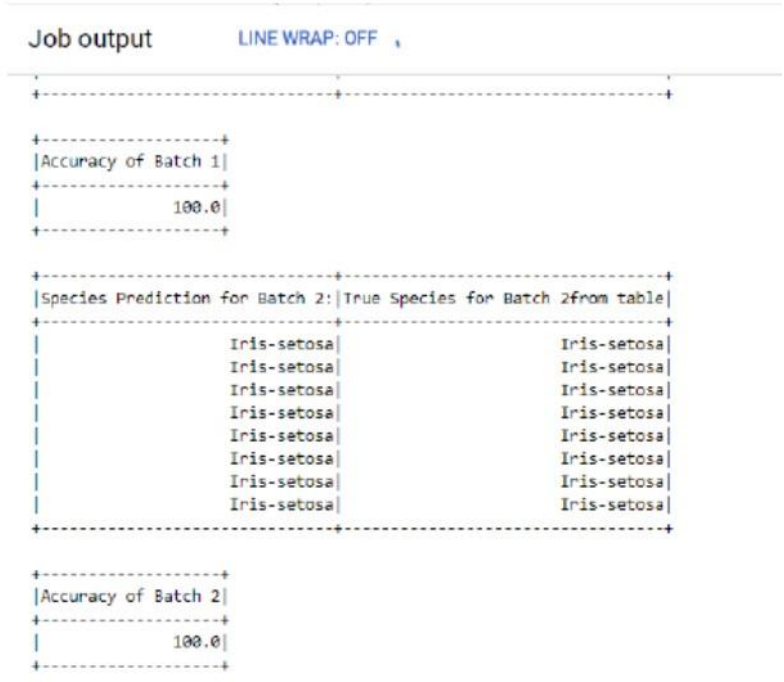
• We then open another Kafka SSH and run the *consumer.sh*. Keep this running parallelly while we run *producer.py*.
• Meanwhile, run *producer.py* in the first SSH. We see that the processing of the data is shown in the other Kafka SSH when we run *producer.py.*

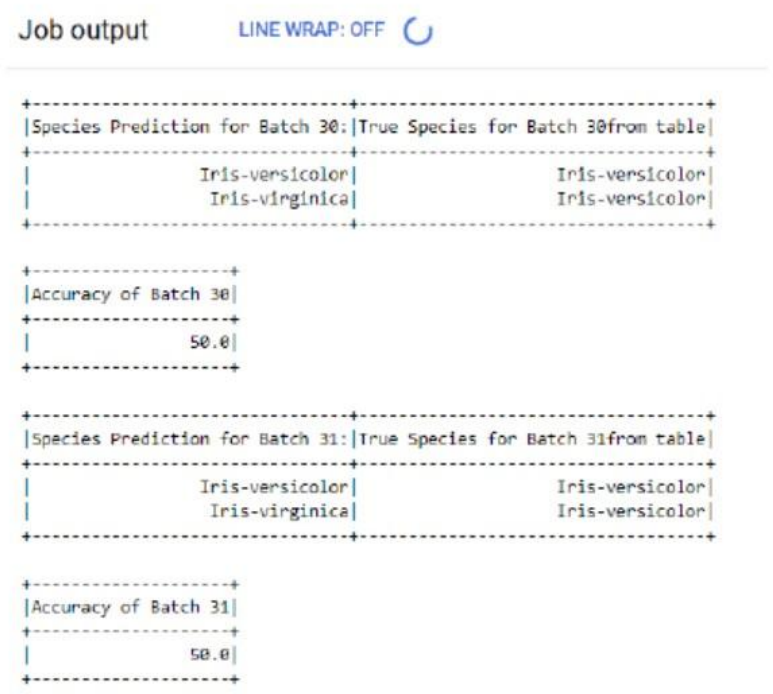
```
Last login: Sun Apr  3 17:54:23 2022 from 35.235.245.130
[ch18b013@kafka-centos-1-vm ~]$ sudo bin/kafka-console-consumer.sh --topic lab7_topic --from-beginning --bootstr
ap-server localhost:9092
sudo: bin/kafka-console-consumer.sh: command not found
[ch18b013@kafka-centos-1-vm ~]$ cd /opt/kafka/
[ch18b013@kafka-centos-1-vm kafka]$ sudo bin/kafka-console-consumer.sh --topic lab7_topic --from-beginning --boo
tstrap-server localhost:9092
{"sepal_length":1.0,"sepal_width":5.1,"petal_length":3.5,"petal_width":1.4,"species":"0.2"}
{"sepal_length":2.0,"sepal_width":4.9,"petal_length":3.0,"petal_width":1.4,"species":"0.2"}
{"sepal_length":3.0,"sepal_width":4.7,"petal_length":3.2,"petal_width":1.3,"species":"0.2"}
{"sepal_length":4.0,"sepal_width":4.6,"petal_length":3.1,"petal_width":1.5,"species":"0.2"}
{"sepal_length":5.0,"sepal_width":5.0,"petal_length":3.6,"petal_width":1.4,"species":"0.2"}
{"sepal_length":6.0,"sepal_width":5.4,"petal_length":3.9,"petal_width":1.7,"species":"0.4"}
{"sepal_length":7.0,"sepal_width":4.6,"petal_length":3.4,"petal_width":1.4,"species":"0.3"}
{"sepal_length":8.0,"sepal_width":5.0,"petal_length":3.4,"petal_width":1.5,"species":"0.2"}
{"sepal_length":9.0,"sepal_width":4.4,"petal_length":2.9,"petal_width":1.4,"species":"0.2"}
{"sepal_length":10.0,"sepal_width":4.9,"petal_length":3.1,"petal_width":1.5,"species":"0.1"}
{"sepal_length":11.0,"sepal_width":5.4,"petal_length":3.7,"petal_width":1.5,"species":"0.2"}
{"sepal_length":12.0,"sepal_width":4.8,"petal_length":3.4,"petal_width":1.6,"species":"0.2"}
{"sepal_length":13.0,"sepal_width":4.8,"petal_length":3.0,"petal_width":1.4,"species":"0.1"}
{"sepal_length":14.0,"sepal_width":4.3,"petal_length":3.0,"petal_width":1.1,"species":"0.1"}
{"sepal_length":15.0,"sepal_width":5.8,"petal_length":4.0,"petal_width":1.2,"species":"0.2"}
{"sepal_length":16.0,"sepal_width":5.7,"petal_length":4.4,"petal_width":1.5,"species":"0.4"}
{"sepal_length":17.0,"sepal_width":5.4,"petal_length":3.9,"petal_width":1.3,"species":"0.4"}
{"sepal_length":18.0,"sepal_width":5.1,"petal_length":3.5,"petal_width":1.4,"species":"0.3"}
{"sepal_length":19.0,"sepal_width":5.7,"petal_length":3.8,"petal_width":1.7,"species":"0.3"}
{"sepal_length":20.0,"sepal_width":5.1,"petal_length":3.8,"petal_width":1.5,"species":"0.3"}
{"sepal_length":21.0,"sepal_width":5.4,"petal_length":3.4,"petal_width":1.7,"species":"0.2"}
{"sepal_length":22.0,"sepal_width":5.1,"petal_length":3.7,"petal_width":1.5,"species":"0.4"}
{"sepal_length":23.0,"sepal_width":4.6,"petal_length":3.6,"petal_width":1.0,"species":"0.2"}
{"sepal_length":24.0,"sepal_width":5.1,"petal_length":3.3,"petal_width":1.7,"species":"0.5"}
{"sepal_length":25.0,"sepal_width":4.8,"petal_length":3.4,"petal_width":1.9,"species":"0.2"}
```

2. Write a subscriber.py file that uses spark streaming (can be receiver-based, dstream, or structured) for producing real-time predictions on these rows by utilizing the model trained in lab5 and calculating the accuracy (the real-time predictions, true labels, and accuracy of all should get printed on console).

Once *producer.py* has finished running in the Kafka SSH, we close the consumer SSH and then submit a job in data proc with python main file: *subscriber.py*., which uses the ML model we had used in the lab5 assignment. The job uses the ML model on the received messages from Kafka topic by the producer.

Once the job is submitted, we get the following output in the job log.





These images show the results for the classification of the data to predict the Species using the other parameters as features.