

Big Data Lab - Lab 7

RA Keerthan

CH17B078

1) Testing if producer.py works fine

producer.py reads each *iris.csv* as a dataframe and converts it to JSON which follows the same schema as that of *iris.csv*. Then it encodes each row and publishes it to the topic *irispred*. This is decoded back in subscriber.py and converted from json to dataframe following the same schema as that used in producer.py

Screenshot of working producer and subscriber:

[←](#) Job details [CLONE](#) [DELETE](#) [STOP](#) [REFRESH](#)

 job-2f6b305c

[EDIT](#)

Start time: Apr 11, 2021, 8:50:35 AM

Job output [LINE WRAP: OFF](#) (

Batch: 1

+-----+-----+-----+-----+-----+
|sepal_length|sepal_width|petal_length|petal_width|species|
+-----+-----+-----+-----+-----+
|5.1|3.5|1.4|0.2|Iris-setosa|
+-----+-----+-----+-----+-----+

Batch: 2

+-----+-----+-----+-----+-----+
|sepal_length|sepal_width|petal_length|petal_width|species|
+-----+-----+-----+-----+-----+
|4.9|3.0|1.4|0.2|Iris-setosa|
|4.7|3.2|1.3|0.2|Iris-setosa|
+-----+-----+-----+-----+-----+

Batch: 3

+-----+-----+-----+-----+-----+
|sepal_length|sepal_width|petal_length|petal_width|species|
+-----+-----+-----+-----+-----+
|4.6|3.1|1.5|0.2|Iris-setosa|
+-----+-----+-----+-----+-----+

At each instant, the row / batch of rows (depending on processing time) is printed on the console. Now that we have the rows received to the subscriber, we will move on to make real-time predictions!

2) Making real-time predictions

In order to make real-time predictions, we have to save the *pipeline*. The pipeline is saved using `.save()` method and loaded for real-time predictions using `.load()` method. We used the following pipeline:

string to index → *vector assembler* → *minmax scaling* → *random forest* → *index to string*

This pipeline is trained on a shuffled iris data with 80% training set and 20% test set. We also remove the *sepal_width* feature before giving the dataframe as input to the pipeline (reasoning given in lab5 report).

For finding accuracy, we use pyspark mllib's *MulticlassClassificationEvaluator* and apply this for every batch using *foreachBatch* method in *writeStream*.

Screenshot of real-time prediction with true label and accuracies:

The screenshot shows a Databricks job details page for job-c7796c0f. The job is titled "Job details" and has buttons for CLONE, DELETE, STOP, and REFRESH. The job is in a "Running" state, indicated by a green circle. The start time is "Apr 11, 2021, 8:58:18 AM". The job output is displayed with "LINE WRAP: OFF". The output shows two batches of predictions and accuracies.

Batch	predicted species	true species
Batch 40	Iris-setosa	Iris-setosa
Batch 40	Accuracy	100.0
Batch 41	Iris-versicolor	Iris-versicolor
Batch 41	Accuracy	100.0

Instance of misclassification:

```
+-----+-----+
|Batch 100 predicted species|Batch 100 true species|
+-----+-----+
|          Iris-virginica|          Iris-virginica|
+-----+-----+

+-----+
|Batch 100 Accuracy|
+-----+
|          100.0|
+-----+

+-----+-----+-----+
|Batch 101 predicted species|Batch 101 true species|
+-----+-----+-----+
|          Iris-versicolor|          Iris-virginica|
+-----+-----+-----+

+-----+
|Batch 101 Accuracy|
+-----+
|           0.0|
+-----+
```