**SHINDE SHUBHAM SUNIL**
**smail:** me18b183@smail.iitm.ac.in
**Course:** CS4830 - Big Data Laboratory
**Instructor:** Prof. Balaraman Ravindran

**Lab-5 Assignment**
**Roll No.:** ME18B183
**Semester:** JAN-MAY 2022
**Due Date:** $12^{th}$ March, 2022

---

**Problem 1**
Download the Iris dataset and upload it into your bucket.

***Solution:***

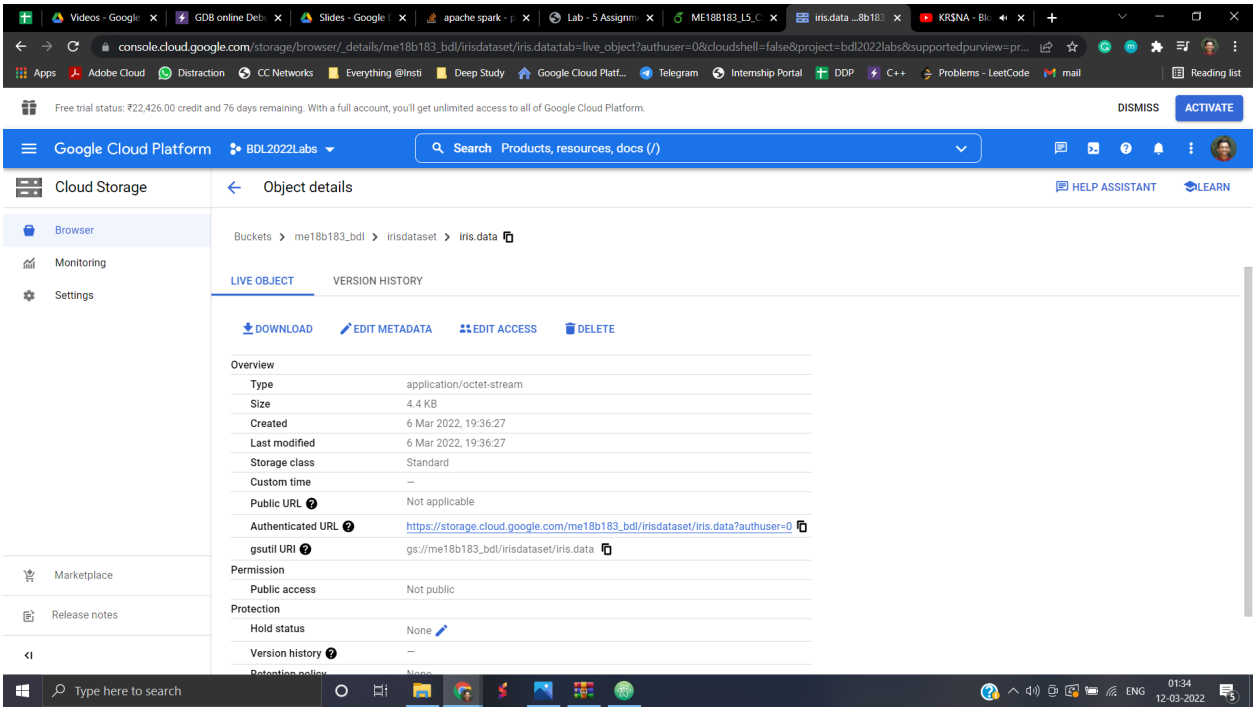


Figure 1: Screenshot of the file *iris* stored in the bucket.

**Problem 2**
Count the number of Iris Virginica flowers which have sepal width greater than 3 cm and petal length smaller than 2 cm.

***Solution:***
SQL code (*q2.sql*) to count the number of Iris Virginica flowers which have sepal width greater than 3 cm and petal length smaller than 2 cm using BigQuery:

```
SELECT
  COUNT(*)
FROM
  bdl2022labs.irisdata.irisdatatable
WHERE
  Species='Iris-virginica'
  AND SepalWidth > 3.0
  AND PetalLength < 2.0
```

Screenshot of the SQL query:

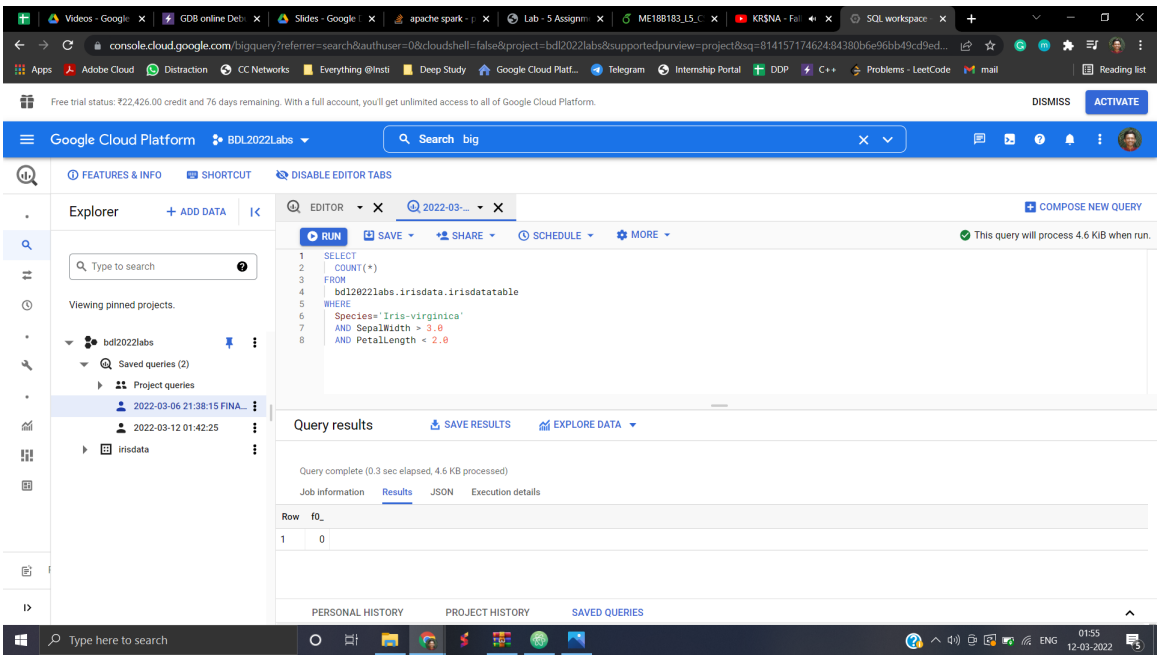


Figure 2: Screenshot of the query result in BigQuery.

As displayed in Fig.2, we conclude that there are **no** Iris Virginica flowers which have sepal width greater than 3 cm and petal length smaller than 2 cm.

**Problem 3**

Train a ML model on the dataset to predict the class of iris plant and report the accuracy for different preprocessing techniques and models. Provide the details of data exploration and feature engineering steps.

***Solution:***

Data Exploration of the dataset:

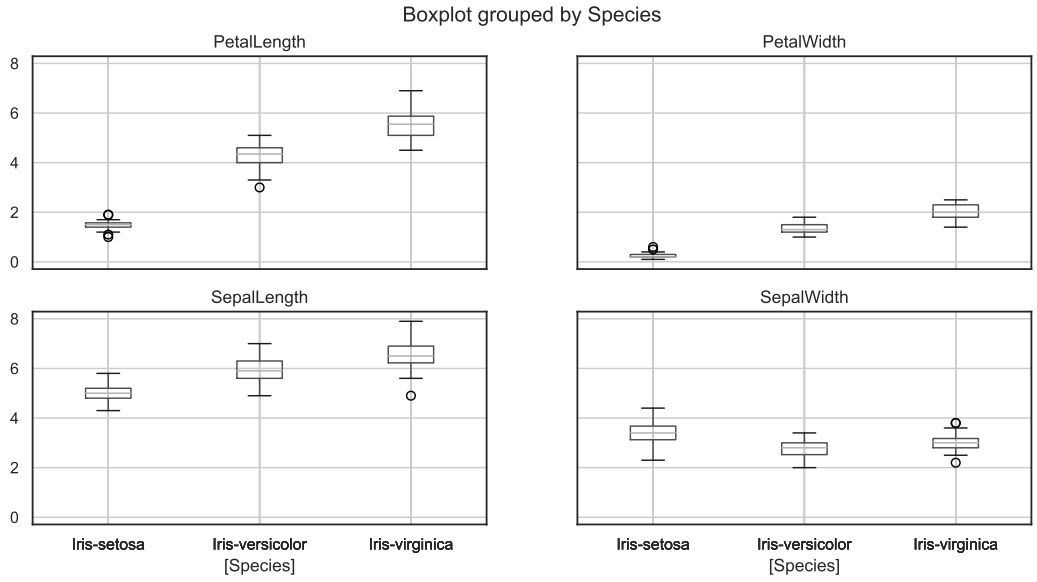


Figure 3: Boxplot grouped by Species.

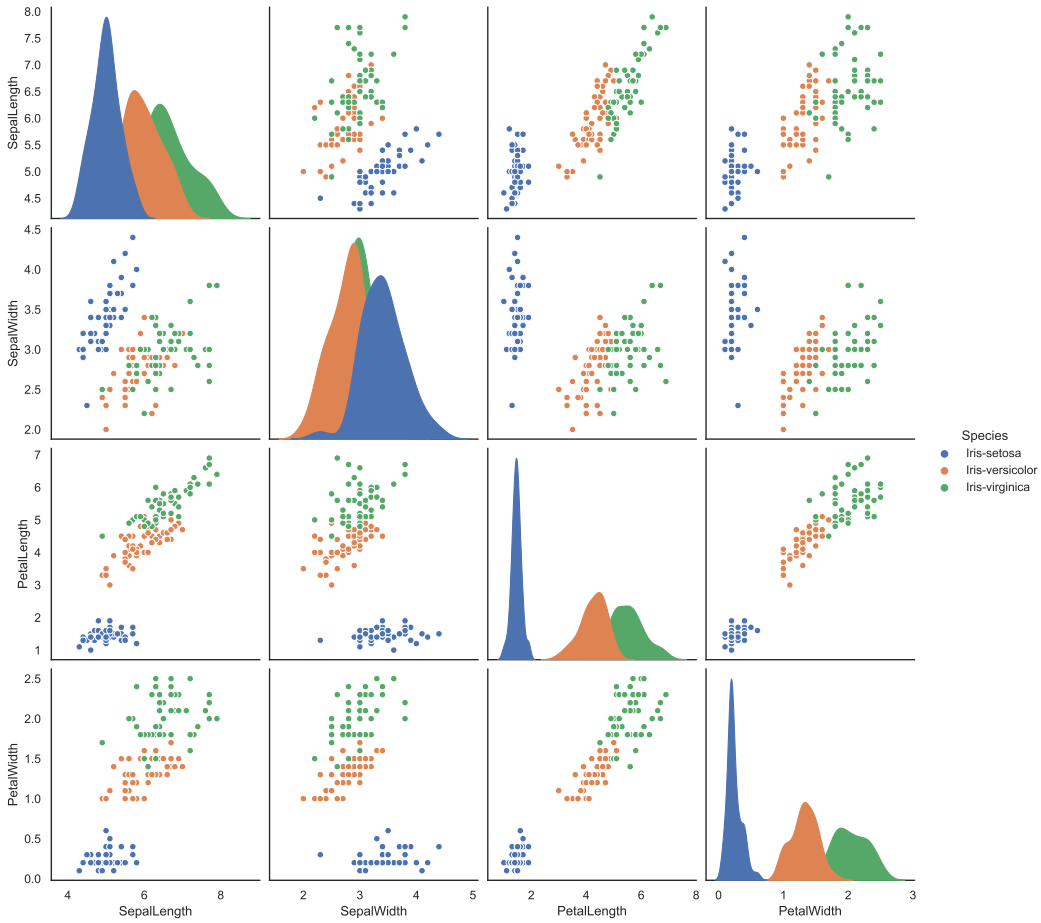


Figure 4: Pairplot of the dataset.

From Fig.3 and Fig.4, we can confirm that all three classes are almost equal in number, therefore class imbalance is absent in this dataset. In the boxplots, especially the *PetalLength* boxplot, we can see that petal length can be a great feature for differentiating the flowers.

Data Preprocessing of the data:

- **Class encoding:** We encode the classes by numerical values in order for the ML model to learn from the dataset.
- **MinMaxScaler:** Scaling the features is a good idea in machine learning. Scaling of the data makes it easy for a model to learn and understand the problem. I have used MinMaxScaler to scale the dataset, which subtracts the minimum from each feature value followed by dividing its range.
- **train-test split:** I have used a 70/30% train-test split for training and testing the dataset. This is a good practice which allows us to check if the ML model is overfitting or not.

| ML Algorithm | Train Accuracy | Test Accuracy |
|---|---|---|
| Decision Trees Classifier | 1.0 | 0.9736 |
| Logistic Regression | 0.9286 | 0.8947 |
| Naive Bayes Classifier | 0.9107 | 0.7632 |
| Random Forest Classifier | 0.9911 | 0.9737 |

Table 1: Train and test accuracy for various ML models.

Screenshots from Google Dataproc:

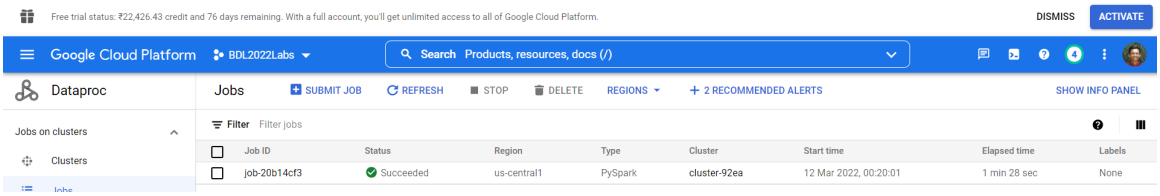


Figure 5: Dataproc job submission.

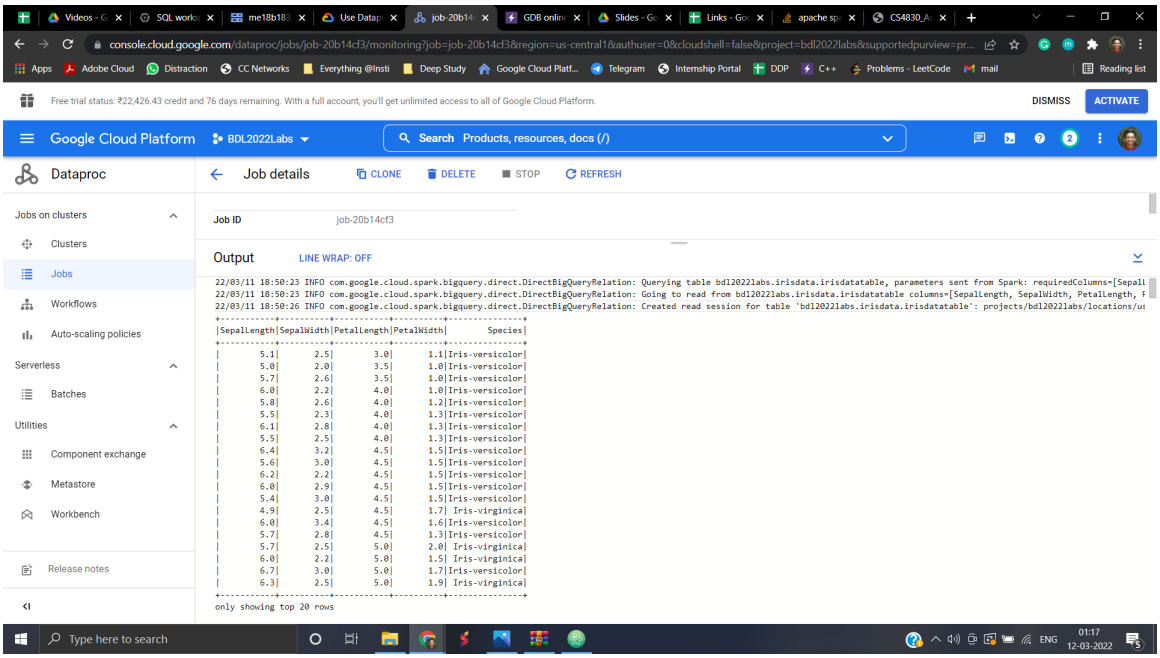


Figure 6: Log of Dataproc job displaying the data that is read.

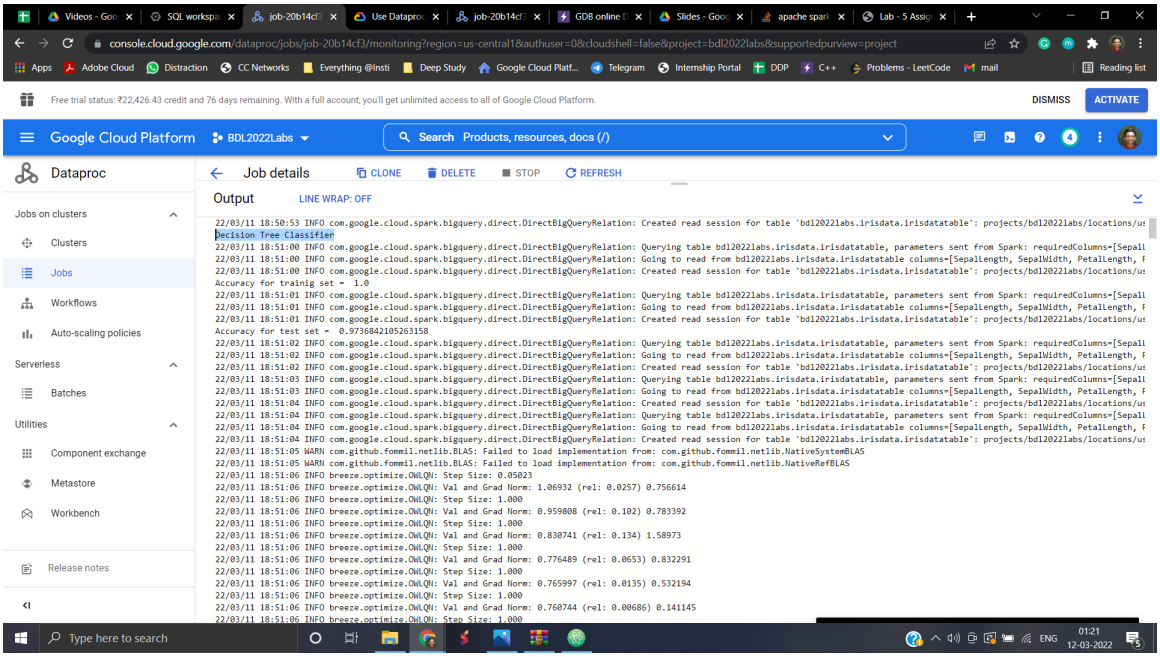


Figure 7: Log of Dataproc job displaying the Decision Tree results.

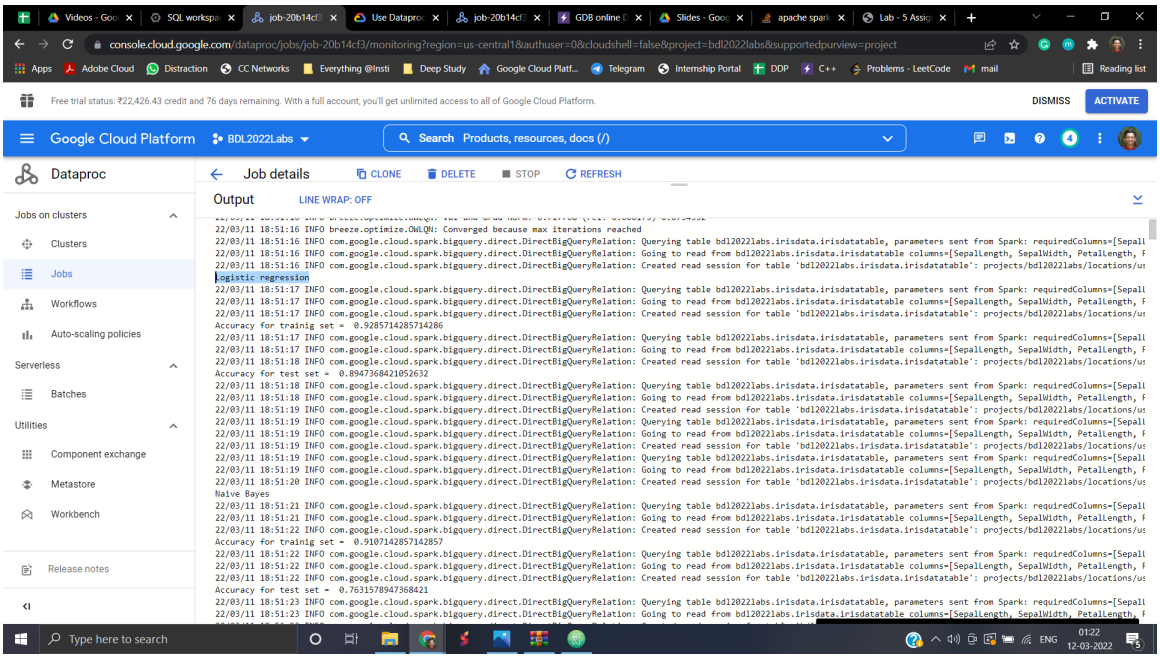


Figure 8: Log of Dataproc job displaying the Logistic Regression and Naive Bayes results.

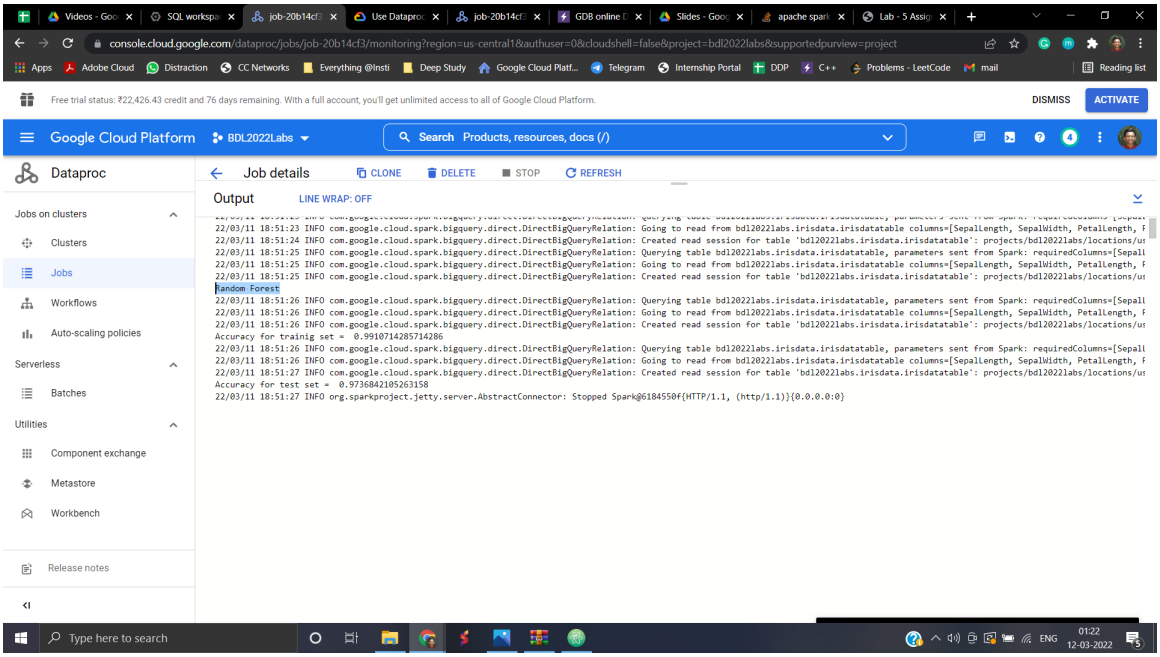


Figure 9: Log of Dataproc job displaying the Random Forest results.

# ∗∗ End of Assignment ∗∗