Big Data project - by Rami Kader

Spark Lab Report

Spark ML: Iris classification

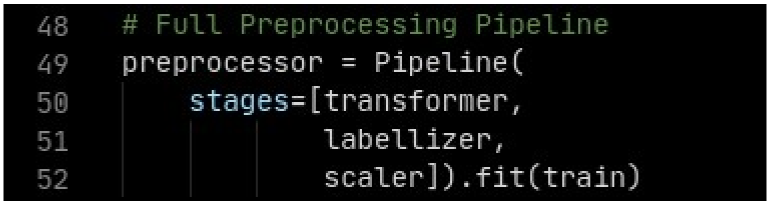
Concept: The task proposed by this project is to write a Spark Pipeline to transform and process the data provided in the Iris data-set. The goal here is to be able to implement a Machine Learning model for classification of the flowers varieties.

The models we used in this lab are:

- Decision Trees
- Logistic Regression
- Naive Bayes

We first wanted to use the Random Forest and the Gradient Boosted Trees classifiers but we found that for the Spark version we're using (v2.3.x), the Gradient Boosted Trees model only supports binary classification while the Random Forest one contains a legacy unfixed bug. Therefore we chose to go with the Naive Bayes and Logistic Regression classifiers.

Pre-processing Pipeline:



```
# Full Preprocessing Pipeline
preprocessor = Pipeline(
    stages=[transformer,
            labellizer,
            scaler]).fit(train)
```

Before applying our machine learning models to the data at our disposal, we needed to transform it a bit to fit the framework's models, and to also better exploit our models. For this we needed a transformer that assembles all training features/columns into one sub-group that would be the input for our models. We also did use a labellizer for which the sole role was to encode the string data targets into int-based categories. Finally, the scaler was applied to normalize our data using the Z-distribution which helps us prevent the divergence of our models.

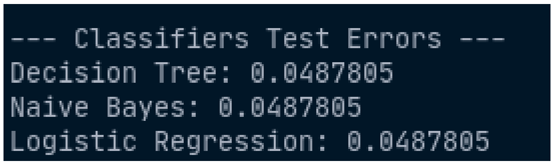
Machine Learning:

To help with the modeling part of this lab, we used the cross validation object provided by the Spark framework to assess our true models performance while trying to fix their parameters and the data we're injecting.

Shown in the table below the standard models cross-validation accuracy scores without any tuning nor data processing was introduced:

Model Name	**Cross-Validation Accuracy**
Decision Tree	~ 92%
Logistic Regression	~ 83%
Naive Bayes	~ 94%

After going through data and using the normalization scaling, and fine-tuning the most influential parameters of each of our models, we were able to obtain great performance out of each one of them:



```
--- Classifiers Test Errors ---
Decision Tree: 0.0487805
Naive Bayes: 0.0487805
Logistic Regression: 0.0487805
```

Model Name	**Cross-Validation Accuracy**
Decision Tree	~ 99.5%
Logistic Regression	~ 99.5%
Naive Bayes	~ 99.5%

PS* To run the code: python iris_ml.py (the config is already implemented in the python script)