Demo 1: Logistic Regression Model with PySpark

Sara El-Metwally

Demo Introduction:
Python Libraries like Pandas and scikit-learn are suitable for mid-size data processing. Machine learning projects always deal with big data sizes that can not fit into one computer memory. The solution for big data processing is distributing its computation among different computing machine. Each machine will have a running code on subsets of data items and the results will be aggregated at the end in order to deliver a complete solution . PySpark is a python API that can handle the parallelization of data processing and provide an easy way to manipulate different issues related to distributing data, code, and collecting results.

Demo Objective:
Get your hand dirty with the first implementation of machine learning model using PySpark API.

Demo Tools:

- **PySpark** installed with command:
  `conda install -c conda-forge pyspark`

Demo Data Set:

- **Link:** https://raw.githubusercontent.com/guru99-edu/R-Programming/master/adult_data.csv
- The data set has personal information of people from different counties and their annual income classified into two categories below or equal 50K$ and above 50K.
- We would like to build a machine learning model that predict the yearly personal income of each person depends on his personal information such as his age, experience, weekly working hours, capital-gain-loss, age, country, etc. The predicted income has two categorical values {0: income <=50K}, {1: income >50K}

Guiding Steps:
(Follow the steps provided in the Notebook created for this demo**):**

- In order to begin with a Spark, you need to initiate a SparkContext with *SparkContext().*
- You need to create a SQL context, *SQLContext()*, to connect to a data source (i.e. your CSV file).
- You can do some data types conversions on some data columns (i.e. features) using a library *Pyspark.sql.types*.
- You can do some data exploration and filtering steps in order to gain some insights and remove noise (*groubby, crosstab, select, sort, drop, dropna* etc.)
- Create a pipeline imported from a library *pyspark.ml* with a set of stages in order to transform each column (i.e. feature) into a suitable representation for a machine learning model.

- Each column that has a string value will pass to the *StringIndexer* and *OneHotEncoder* to have a new suitable representation for a machine learning model.
- The *VectorAssembler* will be created to aggregate the transformation results for all columns.
- The model from the created pipeline will be transformed into a *DataFrame* in order to have a faster computation.
- The *LogisticRegresstion* model is imported from *PySpark.ml.classification*.
- The model will be trained, tested, and evaluated.

Lab Notebook:
https://github.com/SaraEl-Metwally/TOT-ITI-EPITA/blob/main/Big%20Data%20and%20Cloud%20assignment-Final.ipynb