Extending Python features in PySpark

Tirumalesh Nagothi
Date: April 1, 2023

Project Idea: I aim to contribute to Apache Spark by adding new features and techniques to PySpark, such as encoding and imputing the missing values of different methods using Pandas-API within Spark and also testing the integration of the libraries such as boto3(integrates with AWS cloud infrastructure, i.e., S3) and TensorFlow (machine learning and deep learning models).

Introduction: Spark, an open-source project, is a diverse framework that lets extensive data systems work efficiently. This includes the streaming of data, training Machine Learning models, visualization, etc. This has an engine to execute multiple languages to implement data engineering, data science, and machine learning concepts on single-node machines or clusters.

Objectives:
The main aim is to prepare the comprehensive data by converting it into numerical before training the machine learning model. So, to do that, the basic things such as:

1. The null values in the entire dataset should be imputed, such as replaced with a value using various methods.
2. The object type of values is encoded with numeric values such as labeling with serial numbers, count of each unique value, and one-hot encoding(currently for a Boolean feature).
3. Ensure that every value in the dataset is set for training the machine learning model.

The next one is to test the running of the third-party libraries, such as boto3 and TensorFlow's implementation.

Methodology: To achieve the above objectives, I will clone the entire source code of spark and other packages from GitHub and ensure to which path I should add the new feature and new package integration. In the process, I will run my new features in Python code against the simple datasets to ensure how the execution engine reacts to the integration. Then, I will test against the large datasets to check for any alternative mode of implementation.
Ultimately, I will run multiple test cases and, including exception cases to improve the usage, write inline documentation for each feature(method/function) to make it easy to understand.

Tools Used:

- Python3.9
- Git terminal

Expected Outcomes:

1. Each integration should be able to be implemented in the spark environment using PySpark.
2. Immediate solutions are to be provided in case of any issue by runtime error.
3. The given dataset should be able to train a Machine Learning model.
4. Files should be communicated with AWS-S3.
5. Able to run the TensorFlow models.

Reference(s):

1. List of trending open-source big data projects: https://www.projectpro.io/article/best-open-source-big-data-projects-github/516
2. Getting started with spark: https://spark.apache.org
3. Pandas API on spark: https://spark.apache.org/docs/latest/api/python/getting_started/quickstart_ps.html
4. My GitHub repository: https://github.com/tirumaleshn2458/spark.git
5. Spark GitHub repository: https://github.com/apache/spark.git
6. Boto3 GitHub repository: https://github.com/boto/boto3
7. TensorFlow GitHub repository: https://github.com/tensorflow/tensorflow
8. Building spark and launch pyspark from the source code on the local machine: https://spark.apache.org/docs/latest/building-spark.html#building-submodules-individually

9. Guide to contributing to Open-Source projects: https://dev.to/codesphere/how-to-start-contributing-to-open-source-projects-on-github-534n
10. Integration with Cloud Infrastructures: https://spark.apache.org/docs/latest/cloud-integration.htmlxs

Conclusion: The features mentioned above will be going through development, with an introduction to multiple ways(options) to implement them. In the end, the user should be able to send/receive the files from AWS-S3, run TensorFlow models, and quickly execute the imputation of missing values and encoding techniques, i.e., using a single line. Documentation will also be provided at the project's end, guiding the user to test and use it.