EXPERIMENT NO. 9

Aim: Installation and Configuration of Apache Spark. Execution of ML algorithms using Apache Spark Mlib.

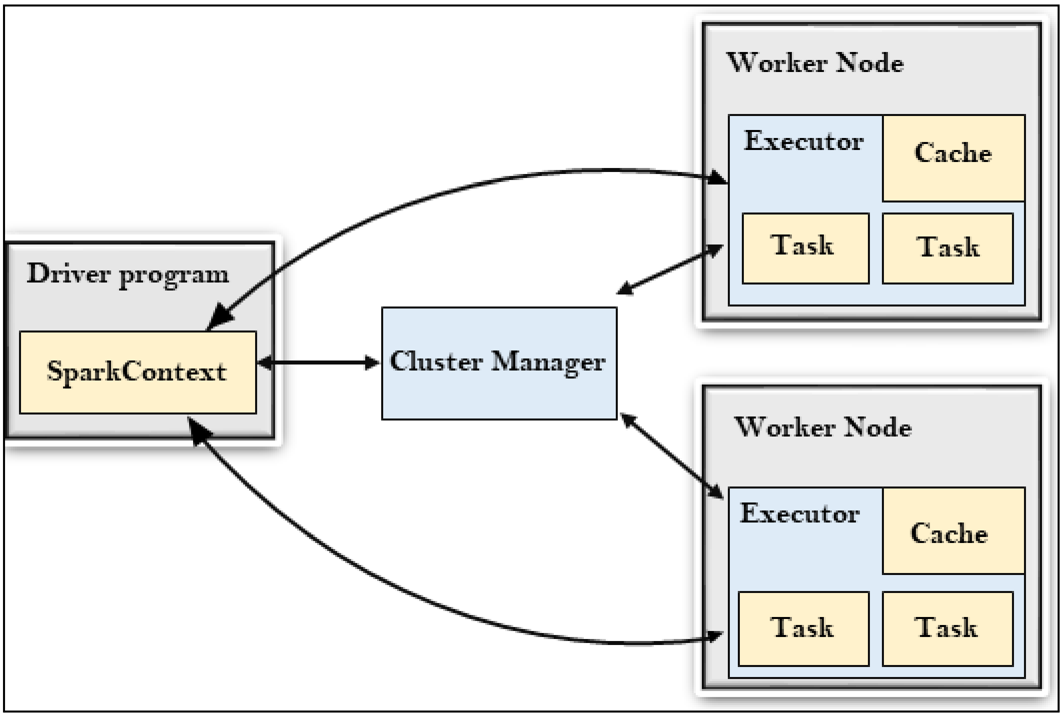

Figure (1). Architecture of Apache Spark

Apache Spark is an open-source, distributed computing system used for big data processing. It provides high-speed computation through in-memory processing and supports various workloads, including batch processing, real-time streaming, machine learning (MLlib), and graph processing (GraphX).

Theory: Explain in detail Apache Spark Architecture with application

Practical Steps

1. Register Google Colab using ur email ID.

#PySpark is the Python library for Apache Spark, an open-source, distributed, and highly scalable big data processing framework

1. pip install pyspark

2. from pyspark.sql import SparkSession

#importing the SparkSession class from the pyspark.sql module.

3.spark = SparkSession.builder.appName('Missing').getOrCreate()

# create new Spark session with specified configuration file

Data set:https://drive.google.com/file/d/1t5WQrtqMuW-C6oeJ1IfGsPFjZa1hl5xQ/view?usp=sharing

4.training = spark.read.csv('file.csv', header=True , inferSchema=True)

# Read data from csv file and store in training

5. training.show()

# print records

6.training.columns

#Print columns only..

7.from pyspark.ml.feature import VectorAssembler

# The VectorAssembler is a feature transformation tool provided by the Apache Spark library for machine learning

```
8.feature=VectorAssembler(inputCols=["Age","Experience"],outputCol="Indepedant feature")
```

# Its primary purpose is to assemble or combine multiple feature columns in a DataFrame into a single feature vector column.

```
9.output=feature.transform(training)
```

# Show the transformed DataFrame, which includes the 'features' column

```
10.output.show()
```

```
11.finaldata=output.select("Indepedant feature","Salary")
```

# DataFrame will have the "features" column with the assembled feature vectors, which is often used as the input for machine learning models.

```
12.finaldata.show()
```

```
13.from pyspark.ml.regression import LinearRegression
```

# It is part of Apache Spark's Machine Learning (MLlib) library and is used for performing linear regression in a distributed and scalable manner.

```
14.train_data,test_data=finaldata.randomSplit([0.75,0.25])
```

# 'training_data' will contain approximately 75% of the data.

# 'testing_data' will contain approximately 25% of the data.

```
15.reg=LinearRegression(featuresCol='Indepedant feature',labelCol='Salary')
```

#Specify the independent features and the target variable from your dataset

16.reg=reg.fit(train_data)

# Train the model on your data

17.reg.coefficients

#Each element in the coefficients array corresponds to the coefficient associated with the respective independent feature.

18.reg.intercept

#The intercept is the constant term in the linear equation that represents the point at which the regression line crosses the y-axis.

19.Pred_result=reg.evaluate(test_data)

#This is a method or function that is used to assess the model's performance on a given dataset.

20.Pred_result.predictions.show()

# Once trained, you can use the model to make predictions.

21.Pred_result.meanAbsoluteError,Pred_result.meanSquaredError

#typically used to calculate and report the model's prediction errors.

Conclusion:

Hence we study how to execute machine learning algorithms using apache spark…

References:

1.https://www.javatpoint.com/apache-spark-architecture
2.https://www.interviewbit.com/blog/apache-spark-architecture/
3.https://www.youtube.com/watch?v=g_5kooM7wTY