Asiana Holloway
October 25, 2024
STAT5165

Final Project Analyzing Employee Attrition Using Spark and Python

Employee Attrition Analysis Report

The goal of this project was to analyze employee attrition data to understand factors that contribute to employee retention and attrition rates. The analysis focused on loading, exploring, and visualizing the dataset to gain insights, with the primary visualization being a bar chart that displays the number of employees who experienced attrition compared to those who did not.

After Spark, Haddop1 and Hadoop2 were successfully downloaded and configured I was able to dive right into the project, the details are as follows: The first step I took was unzipping the Dataset in Pyspark. The dataset was provided in a compressed .zip file format. To make it accessible for analysis, the first step was to extract the file.
Using the unzip command in the terminal, I extracted the .csv file from the archive, making it available in the working directory.

Next, I loaded the dataset with PySpark. Due to the large size of the data and the distributed processing capabilities of Spark, PySpark was used to load and process the dataset. This enabled efficient handling of data in a distributed manner, especially given the two-VM setup. The dataset was read into a Spark Data Frame using `spark.read.csv`, specifying options to infer the schema and set the header.

After that process was complete, I went into data exploration and cleaning. Before conducting any analysis, I needed to understand the data structure, ensure it was loaded correctly, and clean it to remove any missing values that could affect the analysis.
I used schema verification to achieve this. The schema was printed using `df.printSchema()` to verify column names and data types. Next, I completed data cleaning. Any rows with missing values were removed using `df.dropna()` to ensure consistency in analysis.

Once those steps were successfully completed, I did Basic Data Analysis.
After loading and cleaning the data, some basic statistical analyses were performed to gain insights into numerical columns. I used descriptive statistics using `df.describe().show()`, I obtained summary statistics for the numerical columns, including count, mean, min, max, and standard deviation.

After completing basic data analysis, it was time to convert the Spark data frame to pandas data frame. Since Spark does not have native visualization tools, the data needed to be converted into a Pandas data Frame to leverage `matplotlib` for plotting. Pandas integrates well with visualization libraries in Python. The specific column "Attrition" was selected and converted to a Pandas data Frame using `df_pandas = df.select("Attrition").toPandas()`.

Once that was complete it was time to visualize the attrition rates. A bar chart was created to visually represent the count of employees who experienced attrition versus those who did not. This simple visualization provides an immediate understanding of the attrition distribution in the dataset. Using `matplotlib`, I generated a bar chart based on

Asiana Holloway
October 25, 2024
STAT5165

Final Project Analyzing Employee Attrition Using Spark and Python

the value counts of the attrition column in the pandas data Frame. The chart clearly displayed the number of employees with and without attrition.

The data processing and analysis tasks were conducted in a dual-VM setup, where one VM served as the NameNode and the other as a DataNode, allowing for distributed data handling. When tasks were attempted on a single VM, the processing speed for loading and cleaning data was slower, as Spark was limited to the resources of a single machine. Memory usage was higher, and tasks such as `df.describe().show()` and schema printing took longer.

Using a dual-VM setup significantly improved performance. By distributing the data across two VMs, PySpark could parallelize tasks, which reduced the time taken for loading, cleaning, and initial exploration steps. Distributed computing allowed for more efficient memory management, which became particularly beneficial when converting the data to pandas for visualization. Tasks such as data loading and cleaning ran faster, making the analysis workflow smoother and more responsive.

Through this analysis, I was able to load, clean, and explore the employee attrition data using PySpark in a distributed setup, which optimized processing efficiency. Converting the data to a Pandas data Frame allowed for easy visualization with `matplotlib`, and the bar chart provided a straightforward representation of attrition in the company. The dual-VM setup proved beneficial for handling large datasets, demonstrating the advantage of distributed computing in data processing tasks.

This project provided valuable insights into employee attrition rates and demonstrated the effectiveness of Spark and distributed computing for handling and analyzing large datasets. The final bar chart visualization offers a clear overview of the attrition distribution and sets the foundation for more detailed analyses in future work. Below you will find a link to my GitHub to take a further look into my project:

GitHub: https://github.com/AsianaHolloway/Big-Data-Analytics-Course-project-Stage-1