Assignment – Day 17

-Sarthak Niranjan Kulkarni (Maverick)

- sarthakkul2311@gmail.com - (+91) 93256 02791

28/11/2024 (Thursday)

Practice EDA Analysis: -

1. Reading and Displaying Data from the Loan Table in Databricks

→ data = spark.read.table("hive_metastore.default.loan")

display(data)

▸ (1) Spark Jobs

▸ data: pyspark.sql.dataframe.DataFrame = [Customer_ID: string, Age: long ... 13 more fields]

Table

	Customer_ID	Age	Gender	Occupation	Marital Status	Family Size	Income
1	IB14001	30	MALE	BANK MANAGER	SINGLE	4	50000
2	IB14008	44	MALE	PROFESSOR	MARRIED	6	51000
3	IB14012	30	FEMALE	DENTIST	SINGLE	3	58450
4	IB14018	29	MALE	TEACHER	MARRIED	5	45767
5	IB14022	34	MALE	POLICE	SINGLE	4	43521
6	IB14024	55	FEMALE	NURSE	MARRIED	6	34999
7	IB14025	39	FEMALE	TEACHER	MARRIED	6	46619
8	IB14027	51	MALE	SYSTEM MANAGER	MARRIED	3	49999
9	IB14029	24	FEMALE	TEACHER	SINGLE	3	45008
10	IB14031	37	FEMALE	SOFTWARE ENGINEER	MARRIED	5	55999
11	IB14032	24	MALE	DATA ANALYST	SINGLE	4	60111
12	IB14034	32	MALE	PRODUCT ENGINEER	MARRIED	6	null

---

2. Getting Row Count and Schema Information of the Data

→ # Total row count

data.count()

Schema information

data.printSchema()

▸ (2) Spark Jobs

```
root
 |-- Customer_ID: string (nullable = true)
 |-- Age: long (nullable = true)
 |-- Gender: string (nullable = true)
 |-- Occupation: string (nullable = true)
 |-- Marital Status: string (nullable = true)
 |-- Family Size: long (nullable = true)
 |-- Income: long (nullable = true)
 |-- Expenditure: long (nullable = true)
 |-- Use Frequency: long (nullable = true)
 |-- Loan Category: string (nullable = true)
 |-- Loan Amount: string (nullable = true)
 |-- Overdue: long (nullable = true)
 |--  Debt Record: string (nullable = true)
 |--  Returned Cheque: long (nullable = true)
 |--  Dishonour of Bill: long (nullable = true)
```

---

3. Displaying Summary Statistics for 'Income' Column

→ # Summary statistics for 'Income'

data.describe(['Income']).show()

▸ (2) Spark Jobs

```
+-------+-----------------+
|summary|           Income|
+-------+-----------------+
|  count|              468|
|   mean|68339.49145299145|
| stddev|86796.49367750238|
|    min|            28366|
|    max|           930000|
+-------+-----------------+
```

---

4. Counting Rows Grouped by Gender

→ data.groupBy('gender').count().show()

(2) Spark Jobs

```
+------+-----+
|gender|count|
+------+-----+
|  MALE|  280|
|FEMALE|  220|
+------+-----+
```

---

5. Displaying Top 5 Highest Incomes

→ # Top 5 Highest Incomes

data.orderBy(data.Income.desc()).limit(5).show()

(1) Spark Jobs

```
+----------+---+------+--------------+-------------+-----------+------+-----------+-------------+------------------+--------
+-------+-----------+---------------+-----------------+
|Customer_ID|Age|Gender|     Occupation|Marital Status|Family Size|Income|Expenditure|Use Frequency|     Loan Category|Loan Amount
|Overdue| Debt Record| Returned Cheque| Dishonour of Bill|
+----------+---+------+--------------+-------------+-----------+------+-----------+-------------+------------------+----------
+-------+-----------+---------------+-----------------+
|    IBI4157| 35|  MALE|   BANK MANAGER|       MARRIED|          4|930000|      35680|            6|           HOUSING|  6,79,040
|      5|      34,000|              5|                5|
|    IB14107| 44|FEMALE|ACCOUNT MANAGER|       MARRIED|          4|800000|      15632|            8|        AUTOMOBILE| 23,65,478
|      5|      20,145|              3|                4|
|    IB14163| 44|FEMALE|ACCOUNT MANAGER|       MARRIED|          4|800000|      15632|            8|COMPUTER SOFTWARES| 23,65,478
|      5|      20,145|              3|                4|
|    IB14256| 44|FEMALE|ACCOUNT MANAGER|       MARRIED|          4|800000|      15632|            8|COMPUTER SOFTWARES| 23,65,478
|      5|      20,145|              3|                4|
|    IB14128| 46|FEMALE|          CLERK|       MARRIED|          3|750000|      25641|            5|         GOLD LOAN|  2,14,569
|      4|      16,324|              3|                4|
+----------+---+------+--------------+-------------+-----------+------+-----------+-------------+------------------+----------
```

---

6. Grouping Employees by Salary Buckets and Counting

→ # Salary Distribution

from pyspark.sql.functions import ceil, col

Add salary buckets

data_with_buckets = data.withColumn('salary_bucket', ceil(col('Income') / 20000) * 20000)

Count employees in each bucket

data_with_buckets.groupBy('salary_bucket').count().orderBy('salary_bucket').show()

```
▸ (2) Spark Jobs
▸ data_with_buckets: pyspark.sql.dataframe.DataFrame = [Customer_ID: string, Age: long ... 14 more fields]
+-------------+-----+
|salary_bucket|count|
+-------------+-----+
|         NULL|   32|
|        40000|   70|
|        60000|  200|
|        80000|  136|
|       100000|   55|
|       440000|    1|
|       700000|    1|
|       760000|    1|
|       800000|    3|
|       940000|    1|
+-------------+-----+
```

Summary of EDA Analysis: -

I worked on a dataset from the hive_metastore.default.loan table using PySpark in Databricks. First, I loaded the data into a Spark DataFrame and displayed it to get a view of the records. I calculated the total number of rows in the dataset with the count() function, which shows how many entries there are. Then, I examined the schema of the data to understand the structure of the table, such as the column names and data types.

I also performed summary statistics for the Income column, which gave me basic measures like the count, mean, and standard deviation. I grouped the data by gender to count how many records fall into each gender category. To further explore the data, I identified the top 5 highest incomes by sorting the data in descending order based on the Income column. Finally, I created salary buckets by dividing the Income into ranges and counted how many employees fall into each bucket, helping me understand the distribution of income within the dataset.