DATABRICKS Assignment

Databricks Day02 - Sivaprakash V

Initiate session

```
# initialize the session
from pyspark import SparkContext
from pyspark.sql import SparkSession

sc = SparkContext.getOrCreate()
spark = SparkSession.builder.appName('Databricks first program').getOrCreate()
```

Write Data to CSV, JSON, Parquet

sample data

```
# Create a sample DataFrame
data = [
    (1, "Alice", 34, "NY", 5000.50),
    (2, "Bob", 45, "CA", 7000.75),
    (3, "Charlie", 29, "TX", 4000.10),
    (4, "Diana", 39, "WA", 6500.80),
]

columns = ["ID", "Name", "Age", "State", "Salary"]

df = spark.createDataFrame(data, columns)
df.show()
```

```
+---+-------+---+-----+-------+
| ID|   Name|Age|State| Salary|
+---+-------+---+-----+-------+
|  1|  Alice| 34|   NY| 5000.5|
|  2|    Bob| 45|   CA|7000.75|
|  3|Charlie| 29|   TX| 4000.1|
|  4|  Diana| 39|   WA| 6500.8|
+---+-------+---+-----+-------+
```

writing to CSV

```
csv_path = "dbfs:/user/hive/warehouse/employee_csv"

df.write.format("csv").option("header", "true").mode("overwrite").save(csv_path)
```

writing to JSON

```
json_path = "dbfs:/user/hive/warehouse/employee_json"

df.write.format("json").mode("overwrite").save(json_path)
```

writing to parquet file

```
parquet_path = "dbfs:/user/hive/warehouse/employee_parquet"

df.write.format("parquet").mode("overwrite").save(parquet_path)
```

writing to delta

```
delta_path = "dbfs:/user/hive/warehouse/employee_delta"

df.write.format("delta").mode("overwrite").save(delta_path)
```

writing to table

```
df.write.format('delta').saveAsTable("mydata_delta", mode='overwrite')
```

Reading the written data

```
%sql
select * from mydata_delta;
```

ID	Name	Age	State	Salary
3	Charlie	29	TX	4000.1
1	Alice	34	NY	5000.5
4	Diana	39	WA	6500.8
2	Bob	45	CA	7000.75

Exploratory Data Analysis on Credit card data

```
df = spark.read.format("delta").load("dbfs:/user/hive/warehouse/credit_card")
```

Data exploration

```
df.printSchema()
```

```
root
 |-- RowNumber: long (nullable = true)
 |-- CustomerId: long (nullable = true)
 |-- Surname: string (nullable = true)
 |-- CreditScore: long (nullable = true)
 |-- Geography: string (nullable = true)
 |-- Gender: string (nullable = true)
 |-- Age: long (nullable = true)
 |-- Tenure: long (nullable = true)
 |-- Balance: double (nullable = true)
 |-- NumOfProducts: long (nullable = true)
 |-- IsActiveMember: long (nullable = true)
 |-- EstimatedSalary: double (nullable = true)
 |-- Exited: long (nullable = true)
```

```
# Number of rows
num_rows3 = df.count()
# Number of columns
num_columns3 = len(df.columns)
print(f"Number of rows: {num_rows3}")
print(f"Number of columns: {num_columns3}")
```

```
Number of rows: 10000
Number of columns: 13
```

```
from pyspark.sql.functions import col, sum, when, lit
# Calculate null counts
null_counts2 = df.select(
    [sum(when(col(c).isNull(), 1).otherwise(0)).alias(c) for c in df.columns]
)
null_counts_dict2 = null_counts2.collect()[0].asDict()
transposed_null_counts2 = spark.createDataFrame(
    [(key, value) for key, value in null_counts_dict2.items()],
    schema=["Column", "Null Count"]
)
# Show the transposed DataFrame
transposed_null_counts2.show()
```

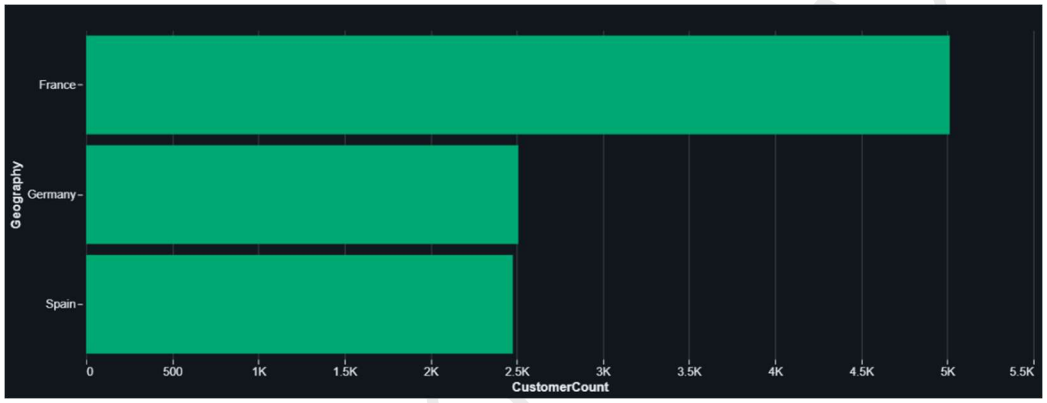
```
+---------------+----------+
|         Column|Null Count|
+---------------+----------+
|      RowNumber|         0|
|     CustomerId|         0|
|        Surname|         0|
|    CreditScore|         0|
|      Geography|         0|
|         Gender|         0|
|            Age|         0|
|         Tenure|         0|
|        Balance|         0|
|  NumOfProducts|         0|
| IsActiveMember|         0|
|EstimatedSalary|         0|
|         Exited|         0|
+---------------+----------+
```

In this query we have number of customers from each geography region. Used bar graph and pie chart

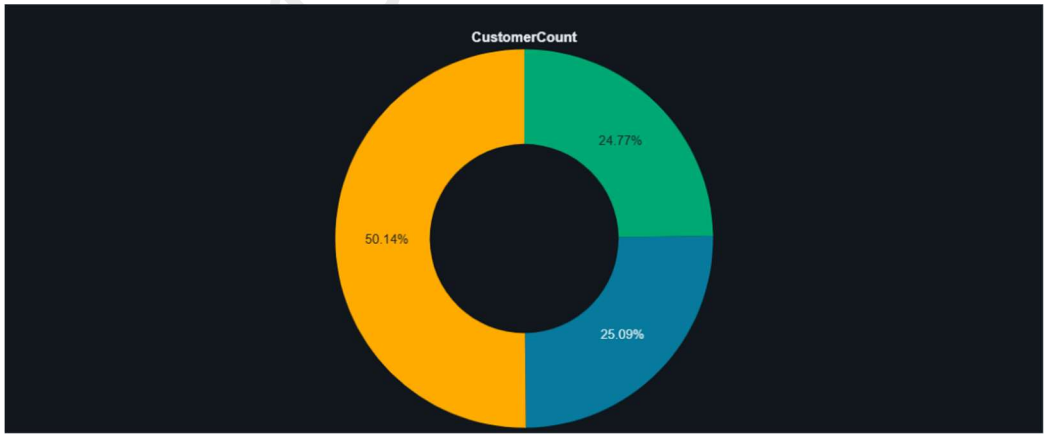
```
%sql
SELECT Geography, COUNT(*) AS CustomerCount
FROM credit_card
GROUP BY Geography;
```

Geography	CustomerCount
Germany	2509
France	5014
Spain	2477

Bar graph



Pie chart



In this query we have grouped data by geography, gender to find the average of creditscore, and customer count. And plotted using line graph and pie chart

```
spark.sql("SELECT Geography, Gender, AVG(CreditScore) AS AvgCreditScore, COUNT(*) AS CustomerCount FROM credit_card GROUP BY Geography, Gender").di
```

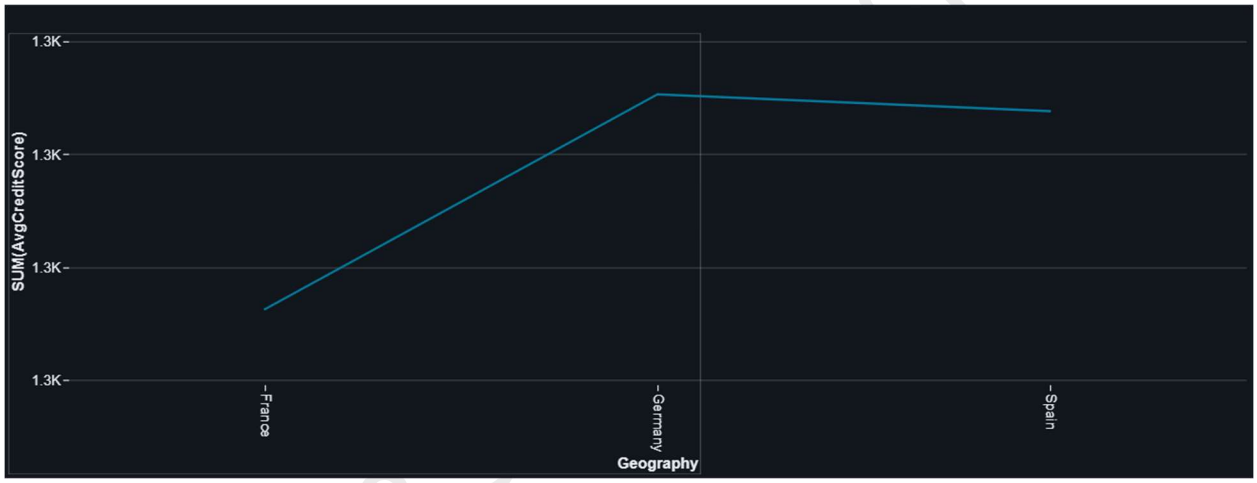
Python

Geography	Gender	AvgCreditScore	CustomerCount
Germany	Female	653.0938809723386	1193
France	Male	650.0646567381038	2753
France	Female	649.1857585139319	2261
Spain	Male	650.9920749279539	1388
Germany	Male	649.9665653495441	1316
Spain	Female	651.7695133149679	1089

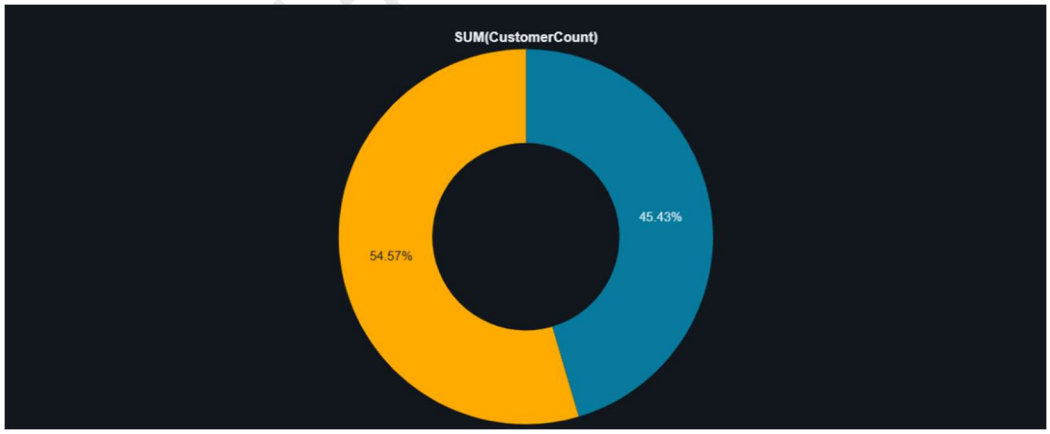
Databricks visualization. Run in Databricks to view.

Databricks visualization. Run in Databricks to view.

Line graph



Pie chart

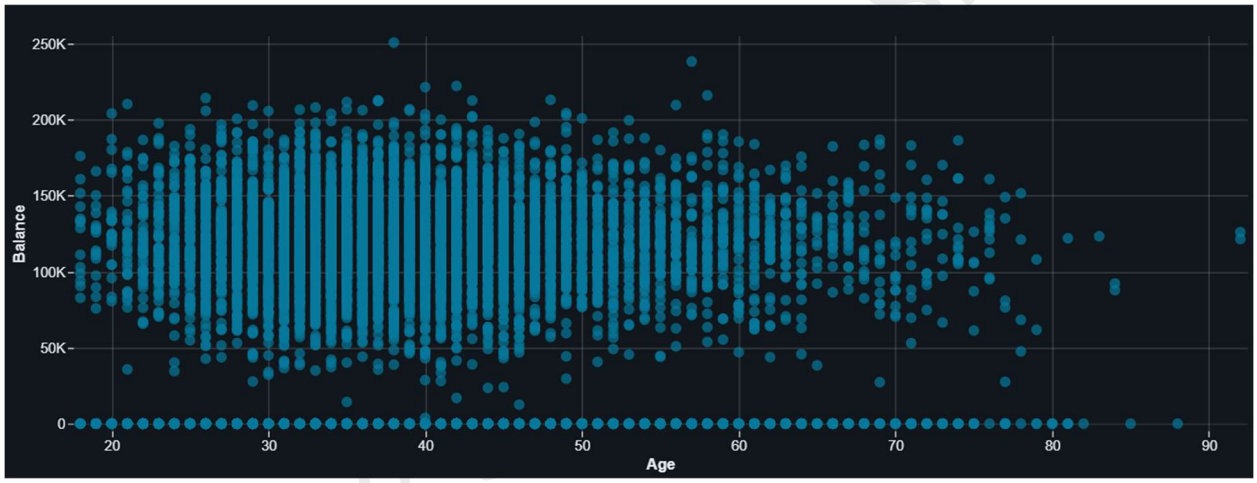


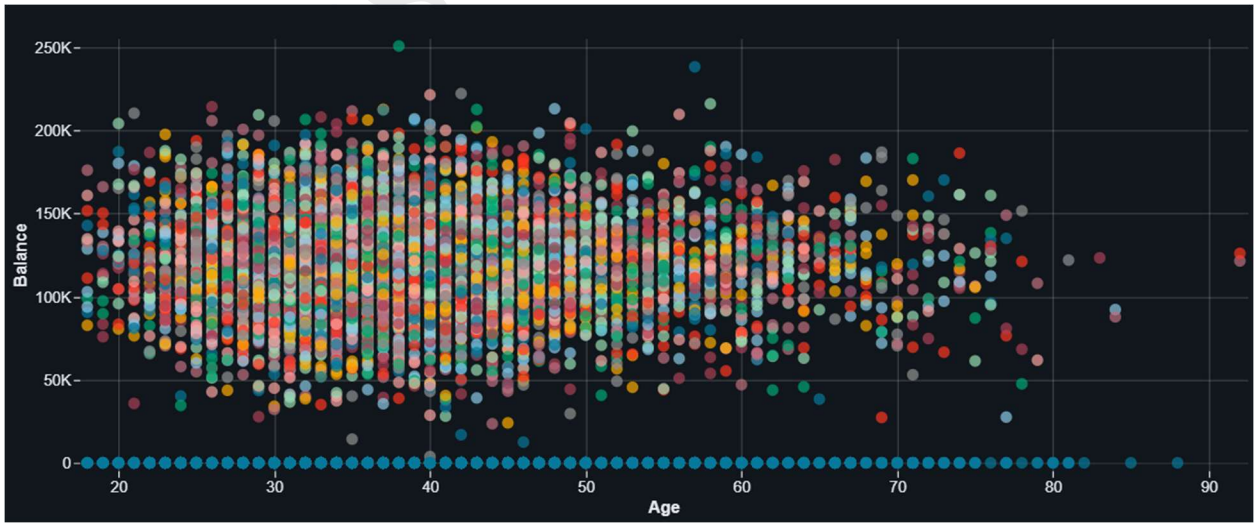
In this query we have taken age and balance. Then plotted it using scatter plot and bubble plot

```
%sql
SELECT Age, Balance
FROM credit_card;
```

66	0.0
51	40685.92
35	136857.0
27	152328.88
33	56084.69
56	78707.16
26	109166.37
36	169831.46

Scatter plot



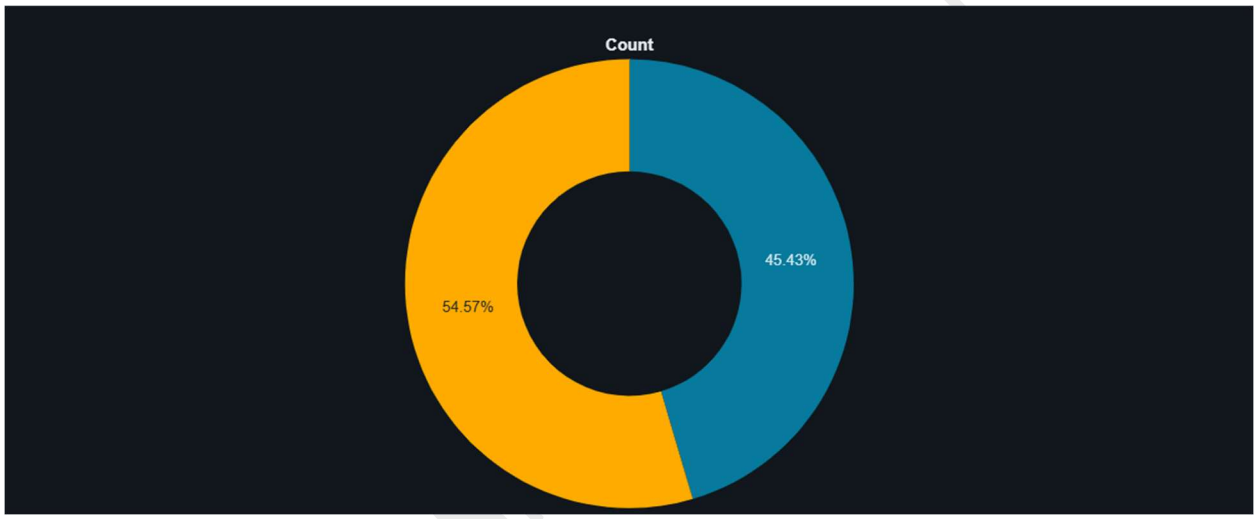


In this query we have grouped data based on gender and find their count. Used pie chart and bar graph.

```
spark.sql("SELECT Gender, COUNT(*) AS Count FROM credit_card GROUP BY Gender").display()
```

Gender	Count
Female	4543
Male	5457

Pie chart



Bar graph