COMP7305SA assignment4

QIU Jiaqi (3036195734), LIANG Guoxun (3036199417)

July 2024

1 Basic information of Project

- The name of AKS cluster: comp7305
- The resource group name used for your AKS cluster: ass4
- The storage account name for this Spark program: comp7305
- The blob container name for this Spark program: sparkstore
- The service account name for this Spark program: ass4
- The Spark image for running Spark on AKS: ipton17/ass4aks
- The URL for downloading the dataset: https://www.kaggle.com/datasets/janiobachmann/bank-marketing-dataset
- The description of the dataset: This dataset contains customer information from bank marketing campaigns, covering various aspects such as personal attributes, account status, and interactions with the bank. It is suitable for analyzing customer engagement in bank marketing activities. The dataset includes 17 features and has over 11,000 records.

2 Question One

Question 1 is designed to calculate the average balance and loan rate for the three marital statuses based on the marital data. Currently, the marital statuses include three categories: single, divorced, and married. Researching the impact of different marital statuses (single, divorced, married) on personal financial situations can not only help banks and financial institutions better design products and services to meet the needs of different customer groups but also provide a basis for the government when formulating relevant policies.

```
loan_rate_by_marital = bank_df.groupBy("marital").agg(
    format_number(avg("balance"), 2).alias("avg_balance"),
    (count(when(col("loan") == "yes", True)) / count("*")).alias("loan_yes_rate")
).orderBy(col("loan_yes_rate").desc())

loan_rate_by_marital.show()

overall_loan_rate = round(
    bank_df.where(col("loan") == "yes").count() / bank_df.count(), 2
)
print(f"Overall Loan Rate: {overall_loan_rate:.2%}")
```

Figure 1: The code of Q1.

2.1 Data Analysis

In this analysis, we group the dataset by marital status and compute two key metrics:

- Average Balance: The average account balance for each marital status group, rounded to two decimal places.
- Loan Approval Rate: The rate at which individuals within each marital status group have been approved for loans, expressed as a proportion of the total number of individuals in that group.

The results show that:

```
+--------+-----------+-------------------+
| marital|avg_balance|      loan_yes_rate|
+--------+-----------+-------------------+
|divorced|   1,371.84| 0.1554524361948956|
| married|   1,599.93| 0.1437568886789482|
|  single|   1,457.26|0.09835133598635588|
+--------+-----------+-------------------+

Overall Loan Rate: 13.00%
```

Figure 2: The result of Q1.

- Divorced individuals have an average balance of 1,371.84 and a loan approval rate of approximately 15.55%.
- Married individuals have an average balance of 1,599.93 and a loan approval rate of approximately 14.38%.
- Single individuals have an average balance of 1,457.26 and a loan approval rate of approximately 9.84%.

The overall loan approval rate across the entire dataset is 13.00%. From this analysis, it is evident that married individuals have the highest average balance, followed by single and divorced individuals. In terms of loan rates, divorced individuals have the highest loan rate, followed by married and single individuals. An unexpected observation is that despite having the lowest average balance, single individuals also have the lowest loan rate.

Hence, we don't think that the married will lead to debt. To be more precise, there is little relationship between debt and marital status.

2.2 Spark Jobs Analyses

The Physical Plan is as shown as 3:

Analysis of the job

Events (1), (2), and (4) are related to the first .groupBy().count() operations.

Events (5), (6), and (7) are related to the second .groupBy().agg() operations; it has two phases of exchanges.

Event (8) is the sort operation. The rest are related to the .cache() and .show().

There are 2 exchanges, so this job results in at least 3 stages.

In a short, job includes a series of transformations (groupBy,alias, orderBy) and actions (show and save).

```
== Physical Plan ==
AdaptiveSparkPlan (7)
+- Sort (6)
   +- Exchange (5)
      +- HashAggregate (4)
         +- Exchange (3)
            +- HashAggregate (2)
               +- Scan csv  (1)
```

Figure 3: The Physical Plan of Q1.

3 Question Two

This question focus on job categories significantly influence average balances, loan rates, and housing loan rates. Understanding these differences can help target financial products and services more effectively based on job categories.

```python
loan_rate_by_job = bank_df.groupBy("job").agg(
    format_number(avg("balance"), 2).alias("avg_balance"),
    (count(when(col("loan") == "yes", True)) / count("*")).alias("loan_yes_rate"),
    (count(when(col("housing") == "yes", True)) / count("*")).alias("housing_yes_rate")
).orderBy(col("housing_yes_rate").desc())

loan_rate_by_job.show()

overall_housing_rate = round(
    bank_df.where(col("housing") == "yes").count() / bank_df.count(), 2
)
print(f"Overall Housing Rate: {overall_housing_rate:.2%}")
```

Figure 4: The code of Q2.

3.1 Data Analysis

In terms of loan rates, entrepreneurs have the highest rates, while students have the lowest. Blue-collar workers, service industry employees, and administrative staff have moderate loan rates. Regarding housing loan rates, blue-collar workers, service industry employees, administrative staff, entrepreneurs, and technicians all have housing loan rates above the overall average.

The significant differences in loan rates and housing loan rates among different occupational categories may indicate disparities in financial security and loan demand among these groups. Financial institutions can leverage these insights to more effectively target financial products to different occupational categories. For instance, loan and housing loan products can be prioritized for blue-collar workers, service industry employees, and administrative staff, as these groups exhibit a higher demand for such loans.

3.2 Spark Jobs Analyses

The Physical Plan is as follows:

```
== Physical Plan ==
AdaptiveSparkPlan (7)
+- Sort (6)
   +- Exchange (5)
      +- HashAggregate (4)
         +- Exchange (3)
            +- HashAggregate (2)
               +- Scan csv  (1)
```

Figure 5: The Physical Plan of Q2.

Analysis of the job

Events (1), (2), and (4) are related to the first .groupBy().count() operations.

Events (5), (6), and (7) are related to the second .groupBy().agg() operations; it has two phases of exchanges.

Event (8) is the sort operation. The rest are related to the .cache() and .show().

There are 2 exchanges, so this job results in at least 3 stages.

In a short, job includes a series of transformations (groupBy,alias, orderBy) and actions (show and save).

4 Question Three

By analyzing the deposits and loans of customers in different age groups, banks can better predict the changes in customers' financial status during their life cycle and improve the relevance of customer relationship management.

Hence, we will analyze importance of age groups, average deposits and loan rates in this section.

```
age_balance_housing_stats = bank_df \
    .withColumn("age_group", floor(col("age") / 10) * 10) \
    .groupBy("age_group") \
    .agg(
        format_number(avg("balance"), 2).alias("avg_balance"),
        avg(when(col("housing") == "yes", 1).otherwise(0)).alias("housing_rate")
    ) \
    .orderBy(col("avg_balance").desc())


age_balance_housing_stats \
    .where(col("age_group").isin([20, 30, 40, 50, 60])) \
    .show()
```

Figure 6: The code of Q3.

4.1 Data Analysis

The 60 and over age group had the highest average balance of $2,481.71.As age decreases, the average balance decreases, with the 20-year-old group having an average balance of $1,183.98.This suggests that as people grow older, they generally have more time and opportunity to accumulate wealth and thus increase their savings.

The 40-year-old group has the highest homeownership rate, at 52.36 percent.The homeownership rates of the 20-year-old and 60-and-over groups are relatively low, at 44.31% and 17.98%, respectively.This may reflect the characteristics of the life cycle of homeownership at different ages: young people are accumulating wealth, middle-aged people are at the peak of homeownership, and older people have more mobility needs.

```
+---------+-----------+-------------------+
|age_group|avg_balance|       housing_rate|
+---------+-----------+-------------------+
|       60|   2,481.71| 0.1797520661157025|
|       50|   1,798.21|  0.386737400530504|
|       40|   1,483.56| 0.5235920852359208|
|       30|   1,350.98| 0.5560444650301065|
|       20|   1,183.98|0.44313725490196076|
+---------+-----------+-------------------+
```

Figure 7: The result of Q3.

4.2 Spark Jobs Analyses

The Physical Plan is as follows:

```
== Physical Plan ==
AdaptiveSparkPlan (8)
+- Sort (7)
   +- Exchange (6)
      +- HashAggregate (5)
         +- Exchange (4)
            +- HashAggregate (3)
               +- Project (2)
                  +- Scan csv  (1)
```

Figure 8: The Physical Plan of Q3.

Analysis of the job

Events (1) and (2) are related to the initial Scan and Project operations.

Events (3) and (4) are related to the first .groupBy().agg() operations; it has two phases of exchanges.

Event (5) and (6) are related to the second .groupBy().agg() operations.

Event (7) is the sort operation.

Event (8) is the final AdaptiveSparkPlan.

This plan has 2 exchanges, so it results in at least 3 stages.

Also, job includes a series of transformations (groupBy,withColumn(),orderBy,alias) and actions (show, count).

5 Question Four

In this section, we mainly focus on the education of the dataset. We have set a dataframe to count the numbers of different education levels and calculate the average balance, also the proportion of housing one.

```
edu_stats = bank_df.where(col("education") != "unknown") \
                .groupBy("education") \
                .agg(count("*").alias("count"),
                     format_number(avg("balance"), 2).alias("avg_balance"),
                     (count(when(col("housing") == "yes", True)) / count("*")).alias("housing_rate")) \
                .orderBy(desc("avg_balance"), desc("housing_rate"))

edu_stats.show()
```

Figure 9: The code of Q4.

5.1 Data Analysis

Those with tertiary education had the highest average account balance of $1,845.87.Those with secondary education had the next highest average account balance, at $1,296.48.Those with primary education had the lowest average account balance, at $1,523.03.

People with secondary education have the highest occupancy rate of 0.53. People with primary education have the second highest occupancy rate of 0.494. People with tertiary education have the lowest occupancy rate of 0.39.

The higher education group (tertiary) has the highest average account balances but the lowest occupancy rates.Conversely, the less educated group (secondary) had lower average account balances but higher occupancy rates.This may reflect differences in the financial and housing choices of people with different educational backgrounds.

```
+---------+-----+-----------+-------------------+
|education|count|avg_balance|       housing_rate|
+---------+-----+-----------+-------------------+
| tertiary| 3689|   1,845.87| 0.3914339929520195|
|  primary| 1500|   1,523.03|0.49466666666666664|
|secondary| 5476|   1,296.48|  0.533418553688824|
+---------+-----+-----------+-------------------+
```

Figure 10: The result of Q4.

5.2 Spark Jobs Analyses

The Physical Plan is as follows:

```
== Physical Plan ==
AdaptiveSparkPlan (8)
+- Sort (7)
   +- Exchange (6)
      +- HashAggregate (5)
         +- Exchange (4)
            +- HashAggregate (3)
               +- Filter (2)
                  +- Scan csv  (1)
```

Figure 11: The Physical Plan of Q4.

Analysis of the job

Events (1) and (2) are related to the initial Scan and Filter operations.

Events (3) and (4) are related to the first .groupBy().agg() operations; it has two phases of exchanges.

Event (5) is the second .groupBy().agg() operation.

Event (6) is the Exchange operation.

Event (7) is the Sort operation.

Event (8) is the final AdaptiveSparkPlan.

This plan has 2 exchanges, so it results in at least 3 stages.

Also, job includes a series of transformations (groupBy,orderBy,alias) and actions (show, count).

6 Question Five

Default rate is one of the key indicators for banks to assess the credit risk of their customers. Banks will differentiate their relationship management strategies according to the default history of different customer groups.For customers with higher default risk, banks may adopt a more prudent service model. Therefore, we will analyze the default situation from job aspect.

```python
job_default = bank_df.groupBy(
    "job"
).agg(
    count(when(col("default") == "yes", True)).alias("default_count"),
    count(when(col("default") == "no", True)).alias("non_default_count"),
    count("*").alias("total_count")
).withColumn(
    "default_rate", col("default_count") / col("total_count")
).orderBy(desc("default_rate"))

job_default.show()
```

Figure 12: The code of Q5.

6.1 Data Analysis

Entrepreneurs had the highest default rate, at 3.05 per cent. Housewives and the unemployed also have higher default rates, at 2.92% and 2.24% respectively. In contrast, students and retired persons

had the lowest default rates, at 0.28 per cent and 0.64 per cent, respectively.

The relatively high default rates for self-employed and blue-collar workers may be related to their less stable occupations. The lower default rates among managers and technicians may be related to the relative stability of their occupations.

job	default_count	non_default_count	total_count	default_rate
entrepreneur	10	318	328	0.03048780487804878
housemaid	8	266	274	0.029197080291970802
unemployed	8	349	357	0.022408963585434174
blue-collar	41	1903	1944	0.02109053497942387
self-employed	8	397	405	0.019753086419753086
technician	29	1794	1823	0.015907844212835986
management	39	2527	2566	0.015198752922837101
unknown	1	69	70	0.014285714285714285
admin.	11	1323	1334	0.008245877061469266
services	7	916	923	0.007583965330444204
retired	5	773	778	0.006426735218508998
student	1	359	360	0.002777777777777778

Figure 13: The result of Q5.

6.2 Spark Jobs Analyses

The Physical Plan is as follows:

```
== Physical Plan ==
AdaptiveSparkPlan (8)
+- Sort (7)
   +- Exchange (6)
      +- Project (5)
         +- HashAggregate (4)
            +- Exchange (3)
               +- HashAggregate (2)
                  +- Scan csv  (1)
```

Figure 14: The Physical Plan of Q5.

Analysis of the job

Events (1) and (2) are related to the initial Scan and HashAggregate operations.

Events (3) and (4) are related to the second .groupBy().agg() operations; it has two phases of exchanges.

Event (5) is the Project operation.

Event (6) is the Exchange operation.

Event (7) is the Sort operation.

Event (8) is the final AdaptiveSparkPlan.

This plan has 2 exchanges, so it results in at least 3 stages.

Also, job includes a series of transformations (groupBy,withColumn(),orderBy,alias) and actions (show, count).

7 Appendix

All questions use explain() to print the plan, and explain belongs to the action of the DataFrame.