Monash University

FIT5202 - Data processing for Big Data (SSB 2024)

Assignment 1: Analysing Historical Loan Data

Due Date: 23:55 Friday 19/Jan/2024 (End of week 3)

Weight: 10% of the final marks

Background

Monash **Lo**an **Co**rporation(**MoLoCo**, an imaginary company) is an established loan servicing company offering various home loans, consumer credit and other unsecured loans defined as below.

- Home loan: Customers borrow money from a bank or company to purchase a property(house/apartment), usually the property is used as a security (called a mortgage). If a customer can't repay the loan, the lender has the right to confiscate the property and sell it.
- Consumer credit: Customers borrow money to purchase goods or services (e.g. an iPhone), and the goods are not secured against the loan. This type of credit usually attracts a higher interest rate. Usually the customers are required to make a periodical payment set by the contract. As a simple example, if you purchase a $1000 phone with 20% interest rate over 12 months, you will repay $100 each month. The size of consumer credit is usually small.
- Other unsecured loans: Other types of loans do not require any type of collateral. For example, in an unsecured business loan, customers borrow money to start a business. If the business fails, the lender could lose money depending on the terms and contracts.

Over many years of operation, they have accumulated many customers and collected large amounts of data from customers and applications.

MoLoCo is still using loan assessors and risk managers to process applications, which is slow and inefficient. It plans to execute a digital transformation strategy to optimise operational costs and improve customer experience. In the first stage, it will utilise big data processing, machine learning and streaming processing to manage loan risk.

The Problem and Project

One of the main risks of a loan business is **default**, which means customers borrow money from the company but fail to repay in some way. We will use big data processing to help the company reduce this risk.

In **Assignment 1,** we will analyse historical loan data to help the company better understand their customers.

The Dataset

You are provided with the dataset in a zip file containing the following files (available in Moodle).

1) metadata.pdf: description of the schema of each data file

2) previous_application.csv: contains data from previous/existing applications
3) application_data.csv: contains current application and customer information
4) value_dict.csv: contains a dictionary of values. Some columns in application tables are stored as integers to save storage, you need to join this table to find the meaning of those values.

(note: the dataset is not cleansed. Please ask in the Ed forum if you notice any issues.)

Assignment Information

The assignment consists of three parts: Working with **RDD**, Working with **Dataframes**, and Comparison of three forms of Spark abstractions. In this assignment, you are required to implement various solutions based on RDDs and DataFrames in PySpark for the given queries related to loan data analysis.

Getting Started

- Download the dataset from Moodle.
- Download a template file for submission purposes:
  - ***A1_template.ipynb*** file in Jupyter notebook to write your solution. Rename it into the format (for example: ***A1_xxx0000.ipynb*****.** This file contains your code solution.)

    note: xxx0000 is your authcate ID/initial of student email.
- You will be using Python 3+ and PySpark 3.5.0 for this assignment (The environment is provided as a Docker image.) (Unit Information >> Software, Documentation, and Resources).

Part 1: Working with RDDs (30%)

In this section, you need to create RDDs from the given datasets, perform partitioning in these RDDs and use various RDD operations to answer the queries for eCommerce analysis.

1.1 Data Preparation and Loading (5%)

1. Write the code to create a SparkContext object using SparkSession. To create a SparkSession you first need to build a SparkConf object that contains information about your application, and use Melbourne time as the session timezone. Give an appropriate name for your application and run Spark locally with as many working processors as logical cores on your machine.
2. Load all CSV files into RDDs.
3. For each RDD, remove the header rows and display the total count and first 10 records. (Hint: You can use csv.reader to parse rows into RDDs.)
4. **Drop** the following columns from RDDs:
   a. previous_application: sellerplace_area, name_seller_industry
   b. application_data: All columns start with **flag_** and **amt_credit_req_**(except for amt_credit_req_last_year).

1.2 Data Partitioning in RDD (15%)

1. For each RDD, print out the total number of partitions and the number of records in each partition. Answer the following questions:
   a. How many partitions do the above RDDs have?
   b. How is the data in these RDDs partitioned by default, when we do not explicitly specify any partitioning strategy?
   c. Can you explain why it will be partitioned in this number? If I only have one single-core CPU on my PC, what is the default partition's number? (Hint: search the Spark source code to try to answer this question.)

   Write the code and your explanation in Markdown cells. (4%)
2. The metadata shows that days in the dataset are stored as a relative number. For example, if the application date is 2/Jan/2024, -1 means 1/Jan/2024, -2 means 31/Dec/2023.
   a. Create a UDF function that takes two parameters: a date and an integer value, and returns a date in ISO format (YYYY-MM-DD). (note: the integer can be either positive or negative). (3%)
   b. Assuming all applications are made on 1/Jan/2024, create a new column named **decision_date,** use the UDF function to fill its values from days_decisions (3%)
3. Join application_data and previous_application with value_dict and replace integer values with string values from the dictionary. (5%)

1.3 Query/Analysis (10%)

For this part, write relevant **RDD operations** to answer the following questions using previous_application RDD.

1. Calculate the total **approved** loan amount for each year, each month. Print the results in the format of **year, month, total_amount**. (5%)

2. For each hour when the applications start (0-23), compute and print the percentage ratio of application cancellation(number of cancelled applications/total number of application*100%). (5%)

Part 2. Working with DataFrames (45%)

In this section, you need to load the given datasets into PySpark DataFrames and use *DataFrame functions* to answer the queries.

2.1 Data Preparation and Loading (5%)

1. Load CSVs into separate dataframes. When you create your dataframes, please refer to the metadata file and use appropriate data type for each column.
2. Display the schema of all dataframes.

2.2 Query/Analysis (40%)

Implement the following queries using dataframes. You need to be able to perform operations like filtering, sorting, joining and grouping by using the functions provided by the DataFrame

API.

1. Calculate the average income for each education_type group, and print the result as a table. (4%)
2. Find the applicants who made credit requests last year with an average credit score of less than 0.5 from the three credit rating sources. (**note: impute null value in credit score with 0.5, not 0**). (4%)
3. Transform the 'days_birth' column in the application_data to **age**(integer rounded down) and **date_of_birth**; then show the schema. You are allowed to use the UDF defined in part 1. (4%)
4. Using an age bucket of 10(0-10, 11-20, 21-30, etc..), compute the percentage of applicants owning a car and a property and print the results as a table. (8%)
5. Draw a barchart to show the total number of uncancelled applications from male/female in each year. (10%)
6. Draw a scatter plot of the applicants' age and their total approved credit. You may use log scales for the XY axis if necessary. (10%)

Part3: RDDs vs DataFrame vs Spark SQL (25%)

Implement the following queries using RDDs, DataFrames in SparkSQL **separately**. Log the time taken for each query in each approach using the "%%time" built-in magic command in Jupyter Notebook and **discuss the performance difference between these three approaches by using the same hardware**. For a fair comparison, you may want to: 1) reduce other factors of interference (e.g. other applications or background services on your laptop.); 2) run each implementation multiple times and get an average value.

Complex Query (high-risk applicants): Find the top 100 applicants who are married with children and have a total approved credit that is more than five times their incomes (regardless of any payments made), sorted by the total credit/income ratio. (hint: intermediate dataframes/tables are allowed if necessary)

Observe the query execution time among RDD, DataFrame, SparkSQL; which is the fastest and why? Please include proper references. (Maximum 500 words.)

Submission

You should submit your final version of the assignment solution online via Moodle. You must submit the files created:

- Your jupyter notebook file **(e.g., A1_authcate.ipynb**).
- **A pdf file** saved from Jupyter Notebook with **all output** following the file naming format as follows: **A1_authcate.pdf**

Note that both submitted (jupyter and pdf) files will be scanned using plagiarism detection software. The highest similarity score among students may be interviewed to prove the originality of the task.

Assignment Marking Rubric

Some of the simple queries have deterministic answers (i.e. right or wrong answer), they will be marked automatically. You will receive zero or full marks according to the correctness of your answer.

For complex queries and explanation questions, you will receive marks based on the quality of your work. Even if the result is incorrect, you will still receive partial marks if the logic is reasonable.

In your submission, the jupyter notebook file should contain the **code and its output**. It should follow *programming standards, readability of the code, and organisation of code*. Please find the PEP 8 -- Style Guide for Python Code for your reference. Here is the link: https://peps.python.org/pep-0008/. Penalty applies if your code is hard to understand with insufficient comments.

A detailed marking rubric is provided in Moodle.

Late submissions

Special Considerations are now to be submitted centrally. This means that students MUST submit an online Special Consideration form via Monash Connect. For more details, please refer to the **Unit Information** section in Moodle.

There is a **10% penalty per day, including weekends** for a late submission. Also, the cut-off date is 7 days after the due date. No submission will be accepted after the cut-off date unless you have a special consideration.

Mark Release and Review

- Mark will be released within 10 business days after the submission deadline.
- Reviews and disputes regarding the mark will be accepted maximum 7 days after the release date (including weekend).

Other Information

Where to get help

You can ask questions about the assignment in the Assignments section in the Ed Forum accessible on the unit's Moodle Forum page. This is the preferred venue for assignment clarification-type questions. You should check this forum regularly, as the responses of the teaching staff are "official" and can constitute amendments or additions to the assignment specification. Also, you can attend scheduled consultation sessions if the problem and the confusion are still unresolved.

Plagiarism and collusion

Plagiarism and collusion are serious academic offences at Monash University. Students must not share their work with any other students. Students should consult the policy linked below for more information.

https://www.monash.edu/students/academic/policies/academic-integrity See also the video linked on the Moodle page under the Assignment block.
Students involved in collusion or plagiarism will be subject to disciplinary penalties, which can include:

- The work not being assessed
- A zero grade for the unit
- Suspension from the University
- Exclusion from the University

Generative AI Statement

As per the University's policy on the guidelines and practice pertaining to the usage of Generative AI, all use of generative AI is **restricted** for this assessment. You should **not** use generative artificial intelligence (AI) to generate any materials or content in relation to the assessment task.

The teaching team restricts all use of generative AI to ensure that students apply their own critical thinking and reasoning skills when working on the assessments. In addition, generative AI tools may produce inaccurate content and this could have a negative impact on students' comprehension of big data topics.

Data source acknowledgement:

The dataset is a remix based on several real-world datasets. We thank the authors/owners for sharing the original datasets.