Monash University

FIT5202 - Data Processing for Big Data

Assignment 2A: Building Models to Predict the Prospective Customers

Due: Friday, Sep 23, 2022, 11:55 PM (Local Campus Time)
Worth: 10% of the final marks

Background

MonPG provides its loan services to its customers and is interested in selling more of its Top-up loan services to its existing customers. They hired us as the Analytics Engineer to develop a model to identify the potential customers that may have any Top Up services in the future. In addition, they want us to help them integrate the machine learning models into the streaming platform using Apache Kafka and Apache Spark Streaming to handle real-time data from the company, in order to recommend our services. In this part A of the assignment, we would only need to process the static data and train machine learning models based on them.

What you are provided with

- Two data files:
  - bureau.csv
  - customer.csv
- These files are available in Moodle in the Assessment section in the Assignment2A Data folder.
- A Metadata file is included which contains the information about the dataset.

Information on Dataset

There are two data files provided. The *customer* data contains variables related to *basic service information. For example, frequency of the loan*, *tenure of the loan*, *disbursal amount for a loan & LTV*. The *bureau* data includes the *behavioral* and *transactional attributes* of the customers, such as *current balance*, *loans amount, overdue*, etc., for various tradelines of a given customer. You can refer to this link for more details of the dataset:
https://www.kaggle.com/datasets/rizdelhi/analytics-vidya-ltfs-finhack-3?select=ltfs3_train_bureau.csv

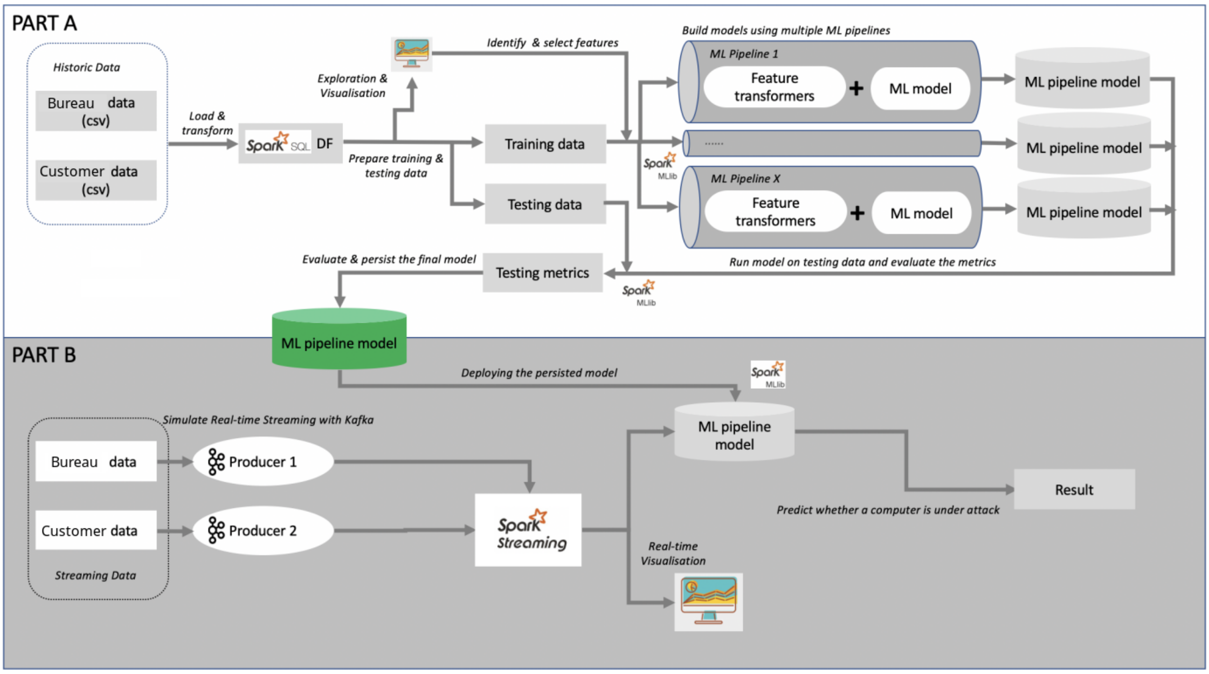
What you need to achieve

MonPG requires us to build a model using both data files. The model should be a binary classification model which could predict whether customers join Top-Up services or not. To achieve this, please derive a new column called “Top-up” from the column called "Top-Up

Month" as your label in the model (label 0 corresponding to No Top-up Service event, and label 1 for all other types of Top-Up service). You can select any columns as features from each activity data, except "Top-Up Month".

Architecture

The following figure represents the architecture of the assignment. **Part A** of the assignment consists of preparing the data, performing data exploration and extracting features, and building and persisting the machine learning models.



In both parts, for the data pre-processing and the machine learning processes, you must implement the solutions using PySpark SQL / MLlib / ML packages. For the data visualizations, please use Matplotlib packages to prepare the plots, and excessive usage of Pandas for data processing is discouraged. Please follow the steps to document the processes and write the codes in Jupyter Notebook.

Getting Started

- Download the datasets from the moodle.
- Create an ***Assignment-2A.ipynb*** file in Jupyter Notebook to write your solution for processing data.

You will be using Python 3+ and PySpark 3.0 for this assignment.

1. Data Preparation and Exploration (35%)

1.1 Creating Spark Session (4%)

1. Create a SparkConf object for using as many local cores as possible for a proper application name.
2. Then create a SparkSession using the SparkConf object.

1.2 Loading the Data (16%)

1. Load each data file into two separate Spark dataframes. Then print out the row count and schema of each dataframe.
    - For the Customer data, please specify the schema before reading the data into dataframes, and make sure 'BranchID', 'AssetID', 'ManufacturerID', and 'SupplierID' be read as String types. You may find relevant schema information from the metadata file for the rest of the columns.
    - For the Bureau data, use PySpark to auto-identify the column types.
2. For the Bureau dataframe, convert all string columns containing ',' to numeric columns (For example: '50,000' → 50000). For the 'INSTALLMENT-AMT' you also need to remove the postfix(For example: '1,405/Monthly' -> 1405).
3. Show how many rows contain a null value in the Customer dataframe, and remove all rows which contain any null value.
4. Show the null percentage of all columns in the Bureau dataframe, and drop the columns whose percentage of null is larger than 20% since those columns might have less influence on our accuracy.
5. Remove all columns containing '**HIST**' in the columns' names for the bureau dataframe and all columns related to 'time' (the columns' names contained "DATE" or "DT") in the bureau dataframe, as they are hard to be merged in the following process.

1.3 Exploring the data (15%)

1. For each numeric feature in each activity, show the basic statistics (including count, mean, std dev, min, max); for each non-numeric feature in each activity, display the top-10 values and the corresponding counts.
    - No need to show the "Top-up Month" and "ID" columns.
2. Merged rows in the Bureau dataframe by 'ID'. To achieve this question, sum the rows for numeric type columns, count distinct values for other columns with other data types, and rename them with the postfix like '_sum' or '_dist'. (For example, we did the sum function based on the 'HIGH CREDIT', and the new column's name will be 'HIGH CREDIT_sum').
    - Sum/distinct is just a simple way to merge the rows, after merging, they may lose some information, but it is an easy way to combine information together. And also helps us to reduce the size of the module.

3. Join the two dataframe with 'ID' by using inner join, then replace the 'Top-up Month' column with a new 'Top-up' column which uses 0 to indicate this customer has 'No Top-up Service' in the 'Top-up Month' column and uses 1 to indicate customer has any type of ‘Top-up Services’ except ‘No Top-up Service’.
4. For each bureau and customer data, present two plots worthy of presenting to the MonPG company, describe your plots and discuss the findings from the plots
    - Hint - 1: you can use the basic plots (e.g. histograms, line charts, scatter plots) for the relationship between a column and the “Top-up” label (such as “city” and “Top-up”, “age” and “Top-up”); or more advanced plots like correlation plots for the relationship between each column; 2: if your data is too large for the plotting, consider using sampling before plotting
    - 100 words max for each plot’s description and discussion

2. Feature extraction and ML training (55%)

2.1 Preparing the training data and testing data (4%)

1. Randomly split the dataset into 80% training data and 20% testing data for each use case.
2. Show the number of rows with different "Top-up" values of the training dataframe. Only use 20% Top-up rows to prepare rebalanced training data, with Top-Up rows and non-Top up rows being a 1:2 ratio.
    - Hint - you can use undersampling to get the rebalanced training data

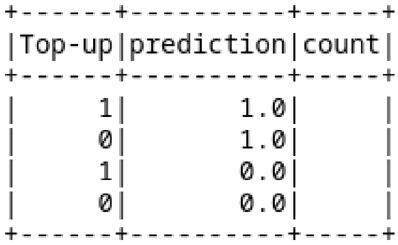
2.2 Preparing Features, Labels, and Models (16%)

1. Which features would you select based on the above data exploration? Discuss the reason for selecting them and how you plan to further transform them.
    - 400 words max for the discussion
    - Hint - things to consider include whether to scale the numeric data, whether to choose one-hot encoding or string-indexing for a specific model
    - As MonPG only has a server with limited resources. Please try to make the training process less tedious by removing the redundant/unnecessary columns that will not significantly impact the results.
    - Another hint: try to understand the performance impact of the model for each column. For example, in the “SupplierID” column, they have more than 4500 different types. That means those columns will burden the training process.
2. Create Transformers / Estimators for transforming/assembling the features you selected above in 2.2.1
    - (Bonus Task 5%) Create a custom Transformer for the column “Frequency” so that the types of policy, ["Monthly", "BI-Monthly", "Quatrly", "Half Yearly"], can be mapped to the following numbers, [0, 1, 2, 3].
        i. Hint: you can create a custom Transform class inheriting from the PySpark ML Transformer, HasInputCol, HasOutputCol, DefaultParamsReadable, and DefaultParamsWritable class, so that it can be included in the ML Pipeline in the next step. The class should allow users to specify the inputCol, outputCol, originalValues, and newValues parameters when initiating the object.

3. Prepare Estimators for Decision Tree and Gradient Boosted Tree model and include them into ML Pipelines.
    - A maximum of two pipelines can be created

2.3 Training and evaluating models (35%)

1. Use the corresponding ML Pipeline from the previous step to train the models on the rebalanced training data from 2.1.2
    - Hint - each model training might take from 1min to 20mins, depending on the complexity of the pipeline model, the amount of training data, and the VM computing power. If your module spends too much time (more than 20mins), **try to rechoose the columns**.
2. Test the models on the testing data from 2.1.1 and display the count of each combination of the Top-Up label and prediction label in formats as below.



```
+------+----------+-----+
|Top-up|prediction|count|
+------+----------+-----+
|     1|       1.0|     |
|     0|       1.0|     |
|     1|       0.0|     |
|     0|       0.0|     |
+------+----------+-----+
```

3. Compute the AUC, accuracy, recall, and precision for the Top-up label from each model testing result using pyspark MLlib / ML APIs. Discuss which metric is more proper for measuring the model performance on identifying Top-Up service.
4. Display the top 5 most important features in each model. Discuss which pipeline model is better and visualize the ROC curve for the better model you selected for each pipeline model.
    - 500 words max for the discussion
5. Using the pipeline model you selected in the previous step, re-train the pipeline model using a bigger set of rebalanced training data, with top-up events and non-top up events being a 1:2 ratio, while using all top-up events data from the full data. Then persist the better models for each pipeline model.
    - The models would be deployed in **Part B** of assignment 2.
    - If your module spends too much time or cracks due to limit memory, try to rechoose the columns, As the server provided by MonPG only has limited performance.

3. Knowledge sharing (10%)

In addition to building the machine learning models, the IT manager from MonPG would like to learn more about the internals of Spark ML, and plan to replace existing scikit learn clustering logic by Spark KMeans clustering to cater large amounts of data. You are expected to combine the theory from the lecture and the observation from Spark UI to explain what happens when training the KMeans clustering model.

3.1 How many jobs are observed when training the KMeans clustering model following the code below? Provide a screenshot from Spark UI for running a simple KMeans model training from the provided data (0.5%)

- For example, run the following code,

```
test_df = spark.createDataFrame([[0, 35.3, 37.5],
[1, 41.4, -23.5],
[2, 28.3, -13.3],
[3, 09.5, -9.0],
[4, 62.8, -18.23],
[5, 63.8, -18.33],
[6, 82.8, -17.23],
[7, 52.8, -13.43],
[8, 72.8, 48.23],
[9, 65.8, 15.43],
[10, 42.8, -13.23] ],
["ID","Att_1", "Att_2"])

assembler = VectorAssembler(
        inputCols=["Att_1", "Att_2"],
        outputCol='features')
kmeans = KMeans(k=4).fit(assembler.transform(test_df))
```

3.2 What method or what kmeans method is used to choose the initial center by default in spark? What will it do if it generates a number of centers more than 'k'?(9.5%)

- 300 words max for the discussion
- Hint - you can also refer to the Spark source code on GitHub https://github.com/apache/spark/blob/master/mllib/src/main/scala/org/apache/spark/mllib/clustering/KMeans.scala

Assignment Marking

The marking of this assignment is based on quality of work that you have submitted rather than just quantity. The marking starts from zero and goes up based on the tasks you have successfully completed and it’s quality for example how well the code submitted follows *programming standards, code documentation, presentation of the assignment, readability of the code, organisation of code and so on*. Please find the PEP 8 -- Style Guide for Python Code here for your reference.

Submission

You should submit your final version of the assignment solution online via Moodle; You must submit the following:

- A zip file of your Assignment 2A folder, named based on your authcate name (e.g. glii0039). This should contain
  - **Assignment-2A.ipynb**

  This should be a ZIP file and *not any other kind of compressed folder (e.g. .rar, .7zip, .tar).* Please do not include the data files in the ZIP file. You should keep all of your outputs in the .ipynb file
  A separate pdf file of your Assignment-2A is generated by Jupyter Notebook, name based on your authcate name (e.g. glii0039).
- The assignment submission should be uploaded and finalized by Friday, Sep 23, 2022, 11:55 PM (Local Campus Time).
- Your assignment will be assessed based on the contents of the Assignment 2 folder you have submitted via Moodle. When marking your assignments, we will use the same ubuntu setup as provided to you in Week 01.

Other Information

Where to get help

You can ask questions about the assignment on the Assignments section in the Ed Forum accessible from the 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 visit the consultation sessions if the problem and the confusions are still not solved.

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

Late submissions

Late Assignments or extensions will not be accepted unless you submit a special consideration form. ALL Special Consideration, including within the semester, is 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 the late submission.