Monash University

FIT5202 - Data processing for Big Data

Analysing Flight-delays Data

isualization, Data Modelling)

26th September 2021 11:00 pm

Weight : 10% percent of the total mark

Background

The flight-delays prediction dataset has become one popular dataset used by the aviation industry to predict the delay given the historical flight data. Learning from data can be beneficial for the companies, e.g. aviation industry, so that they can minimize the delay to improve customer satisfaction. The insight of the data can be obtained by conducting some steps, including pre-processing, visualization, and data modelling.

There is no change in the datasets in comparison to the assignment1. The details of the dataset can be seen below.

Required datasets (available in Moodle):

- A compressed file **flight-delays.zip.**
- This zip file consists of 21 csv files and a metadata file:
  - 20 flight*.csv files
  - airports.csv (we do not use it)
  - metadata.pdf
- Note that in this assignment, the original flights.csv has been sliced and reduced into several csv files in order to accommodate the hardware limitations.
- The complete dataset also can be downloaded publicly at https://www.kaggle.com/usdot/flight-delays

Additional packages allowed in this assignment:

- numpy, pandas, scipy, and matplotlib
- If you are unsure or need to use other packages please consult to the tutors or ask Ed Forum

Information [illegible]

The flight-delays ar[illegible] was collected and published by the U.S. Department of Transportation's [illegible] ransportation Statistics. This data records the flights operated by large a[illegible] s the on-time performance of domestic flights. This data summarises various flight information such as the number of on-time, delayed, cancelled, and diverted flights published in DOT's monthly in 2015.

Assignment Information

This assignment consists of two parts:

- **Part 1:** Data Loading, Cleaning, Labelling, and Exploration
- **Part 2:** Feature extraction and ML Training

In the assignment1, our focus was on manipulating data using Spark RDD, Spark DataFrames and Spark SQL. In this assignment, we only use Spark SQL where you need to further do the pre-processing, visualization, and data modelling steps. The data modelling task aims to create the machine learning models utilizing MLlib/ML APIs to predict both departure and arrival delays from the testing data using classification tasks. You will need to build models to predict the classes.

The class labels are generated from the departure and arrival delay values range. To define the categorical label in the dataset, you need to conduct a labelling task. To do that, you need to define thresholds that can split data into two or more category labels. For binary classification, a threshold can be a *mean* or *median* in the range of departure/arrival delay values. For example, the instances in which arrival delay values between *min range* and *median* are labelled as 1 which represents a *late* condition, otherwise 0 which represents *not late* condition. In multiclass classification, the label can be defined by splitting the range into several parts (e.g using bin size) where each part has its own label.

Some of the parts in the assignment are open questions. You might have a different approach for every step. For example, in the case of multiclass labelling tasks, you can define how you would categorize the data based on the range of arrival and departure value. In another case, it is also possible for you to define the steps in spark ML transformers.

Getting S[illegible]

- Download the [illegible] le namely **flight-delays.zip**.
- There is no te[illegible] ment, please organize your answer in order so that it is easy to mark.
- You will be using Python 3+ and PySpark 3+ for this assignment.

1. Data Loading, Cleaning, Labelling, and Exploration (45%)

In this section, you will prepare the data (loading and cleaning), performing data exploration and

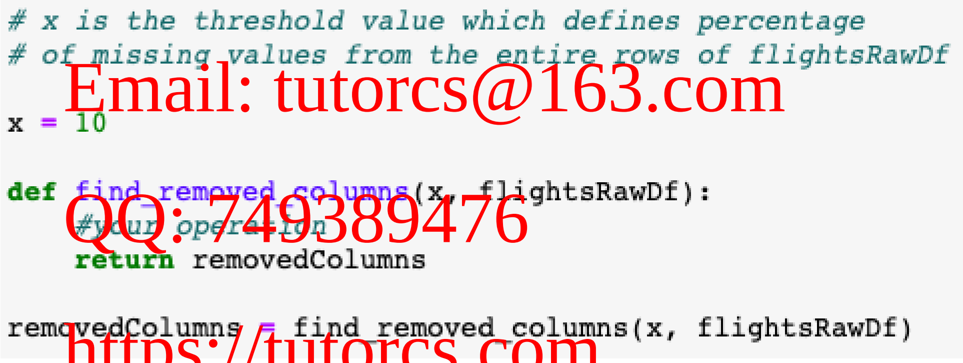
1.1 Data loading (5%)

1. Write the code to get an object using *SparkSession*, which tells Spark how to access a cluster. To create a *SparkSession* you first need to build a *SparkConf* object that contains information about your application. Give an appropriate name for your application and run Spark locally with as many working processors as logical cores on your machine.
2. Read the 20 files of “flight*.csv” file into a single Spark DataFrame namely *flightsRawDf* using spark.read.csv with both header and inferSchema attributes are set to True. Display the total number of *flightsRawDf* rows.
3. Obtain the list of columns from *flightsRawDf* and name this variable as *allColumnFlights*.

1.2 Data cleaning (15%)

1. Check for missing values (NaN and Null) in all columns, display the number of missing values for each column. Do you see any pattern in how values are missing in the data? E[illegible]e top two columns with the highest percentage of missing valu[illegible] values are missing completely at random?
2. Remove son[illegible]s in the *flightsRawDf* using threshold values. Please do the follow[illegible]

   a. Write a python function to automatically obtain the *column names* for all columns in *flightsRawDf* whose number of missing values is greater *x percent* (*x* is a threshold in percent, e.g. *x*=10) from the number of rows. These columns are deemed unworthy due to the abundance of missing values. Example of the function is as follows:



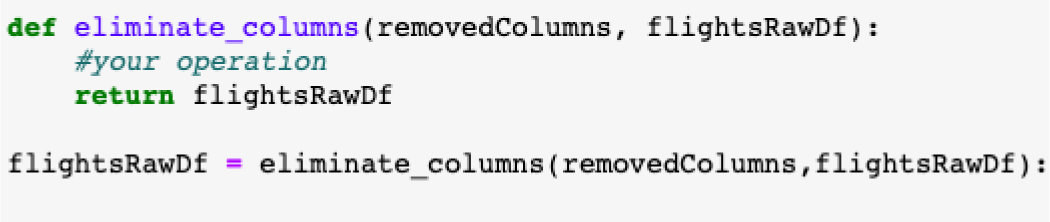
```
# x is the threshold value which defines percentage
# of missing values from the entire rows of flightsRawDf

x = 10

def find_removed_columns(x, flightsRawDf):
    #your operation
    return removedColumns

removedColumns = find_removed_columns(x, flightsRawDf)
```

   b. Once a list variable *removedColumns* is obtained, write a python function namely *eliminate_columns* so that *flightsRawDf* is updated by removing columns listed in *removedColumns* variable. Example of the function is as follows:



```
def eliminate_columns(removedColumns, flightsRawDf):
    #your operation
    return flightsRawDf

flightsRawDf = eliminate_columns(removedColumns,flightsRawDf):
```

   Then, display the number of rows and columns in *flightsRawDf*.

   c. Drop rows with Null and Nan values from *flightsRawDf*. Please name it as *flightsDf*. Display the number of rows and columns in *flightsDf*.

1.3 Data Labelling (15%)

1. Generate labels in *flightsDf* using predetermined values as follow:
   a. The new binary labels generated from arrival delay and departure delay colu[illegible]cation task purpose. The new column names are *binar*[illegible]*ryDeptDelay*, which are generated from arrival delay and [illegible]ctively. Label the data as 1 (late) if the delay value is positi[illegible] as 0 (*not late*).
   b. The new multiclass labels generated from arrival delay and departure delay columns for the classification task purpose. The new column names are *multiClassArrDelay* and *multiClassDeptDelay*, which are generated from arrival delay and departure delay respectively. For multiclass labels. Please make it three classes as *early*, *on time*, and *late* which are represented by the 0, 1, and 2 values. For example: The value below 5 is regarded as *early*, 5 to 20 is regarded as *on time*, and above 20 is regarded as *late*.

2. Auto labelling *flightsDf* using function
   a. The same task as multi class labelling task 1b above. However, in this task please write a python function to execute data labelling automatically (Task 1b uses hardcoding method). You may determine the range yourself for each category *early*, *on time*, and *late* (e.g. using bin size). Make sure to study the distribution of the data and comment on why you think your choice of bin size is appropriate, in not more than 200 words.

1.4 Data Exploration / Exploratory Analysis (10%)

1. Show the basic statistics (count mean stdev, min, max, 25 and 75 percentile) for all columns of the *flightsDf*. Observe the mean value of each column in *flightsDf*. You may see that a column like 'AIRLINE' does not have any mean value. You may see that a column has a mean value equal to zero (0). Keep in mind that this analysis is needed further to determine the selected features to build the machine learning models. You can restrict your analysis to numerical columns only.
2. For categorical columns identify total unique categories, and category name, and

frequency of the highest occurring category in the data.

3. Write code to display a histogram (only for binary labels) to show the task as follows:
   a. Percentage of flights that arrive late each month.
   b. Percentage of flights that arrive late each day of week.
   c. Perce[illegible]hts by airline.

2. Feature e[illegible]nd ML Training (55%)

In this section, you [illegible]Spark DataFrame functions and ML packages for data preparation, model building and evaluation. Other ML packages such as scikit learn would receive zero marks. Excessive usage of Spark SQL is discouraged.

2.1 Discuss the feature selection and prepare the feature columns (10%)

1. Related to the question on 1.4.1, discuss which features are likely to be feature columns and which ones are not. Further, obtain the correlation table (use default ‘pearson’ correlation) which shows the correlation between features and labels. How would you analyze the correlation for categorical columns? For the discussion, please provide 300-500 words of description.
2. Write the code to create the analytical dataset consisting of relevant columns based on your discussion above.

2.2 Preparing any Spark ML Transformers/ Estimators for features and models(10-15%)

1. Write code to create Transformers/Estimators for transforming/assembling the columns you selected above in 2.1.1.
2. Bonus task: (5%)

   Create a Custom Transformer that allows you to map Months to Season. Note the Custom Transformer can be used with the pipeline. E.g. Months 3 to 5 are mapped as “Spring”, Months 6 to 8 are mapped as “Summer”, Months 9 to 11 are mapped as “Autumn”, and Months 12, 1, and 2 are mapped as “Winter”.
3. Create ML model Estimators for Decision Tree and Gradient Boosted Tree model for binary classification for both arrival and departure delays (PLEASE DO NOT fit/transform the data yet).

4. Create ML model Estimators for Naive Bayes model for multiclass classification for both arrival and departure delays from the labels that you have created at task 1.3.2 (PLEASE DO NOT fit/transform the data yet).
5. Write code to include the above Transformers/Estimators into pipelines for all tasks (PLEASE D[illegible] the data yet).

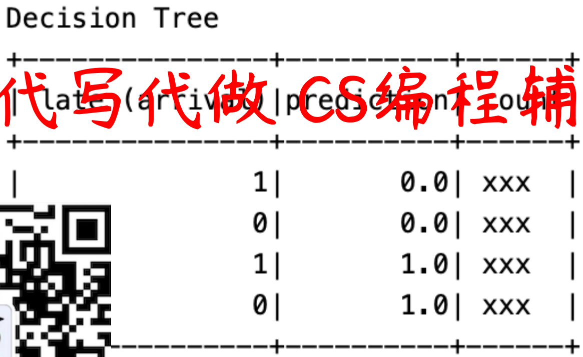
2.3 Preparing t[illegible] testing data (5%)

1. Write code [illegible]e data into 80 percent and 20 percent proportion as training and testing data. You may use *seed* for the analysis purposes. This training and testing data will be used for model evaluation of all tasks in 2.4.

2.4 Training and evaluating models (30%)

For three use cases below, please follow the instructions.

1. Binary classification task (Using Decision Tree and Gradient Boosted Tree) for both arrival and departure delay classification (4 models).
   a. Write code to use the corresponding ML Pipelines to train the models on the training data.
   b. For both models and for both delays, write code to display the count of each combination of *late*/*not late* label and prediction label in formats as shown in the example Fig.1.
   c. Compute the AUC, accuracy, recall, and precision for the *late*/*not late* label from each model testing result using pyspark MLlib/ML APIs.
   d. Discuss which metric is more proper for measuring the model performance on predicting *late/not late* events, in order to give the performers good recommendations.
   e. Discuss which is the better model, and persist the better model.
   f. Write code to print out the leaf node splitting criteria and the top-3 features with each corresponding feature importance. Describe the result in a way that it could be understood by your potential users.
   g. Discuss the ways the performance can be improved for both classifiers (500-700 words in total)



```
Decision Tree
+----------------+----------+-------+
| late (arrival)|prediction| count|
+----------------+----------+-------+
|               1|       0.0| xxx   |
|               0|       0.0| xxx   |
|               1|       1.0| xxx   |
|               0|       1.0| xxx   |
+----------------+----------+-------+
```

tput for binary classification task (example)

2. Multiclass classification task (using Naive Bayes) for *only arrival delay* classification (1 model).

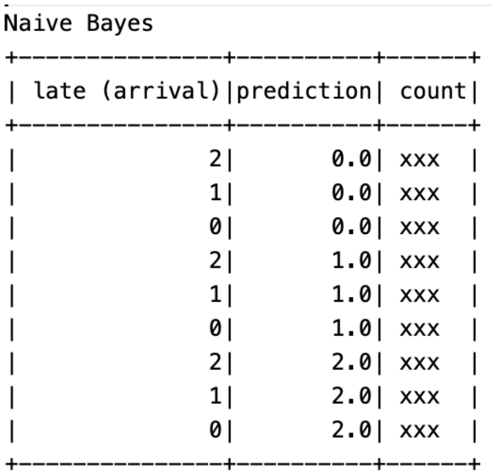
   a. Write code to use the corresponding ML Pipelines to train the model on the training data. This label is automatically generated from task 1.3.2.

   b. Write code to display the count of each combination of *early/on-time/late* label and prediction label in formats as shown in the example Fig.2.

   c. Compute the AUC, accuracy, recall, and precision for the *early/on-time/late* label from Naive Bayes model from pyspark MLlib/ML APIs.

   d. Discuss which metric is more proper for measuring the model performance on predicting *early/on-time/late* events, in order to give the performers good recommendations.

   e. Discuss the ways the performance can be improved for Naive Bayes classifiers (500-700 words in total)



```
Naive Bayes
+---------------+----------+------+
| late (arrival)|prediction| count|
+---------------+----------+------+
|              2|       0.0| xxx  |
|              1|       0.0| xxx  |
|              0|       0.0| xxx  |
|              2|       1.0| xxx  |
|              1|       1.0| xxx  |
|              0|       1.0| xxx  |
|              2|       2.0| xxx  |
|              1|       2.0| xxx  |
|              0|       2.0| xxx  |
+---------------+----------+------+
```

Fig.2. Format output for multiclass classification task (example)

Assignment Marking

The marking of this assignment is based on the quality of work that you have submitted rather than just qua[illegible]starts from zero and goes up based on the tasks you have successfully c[illegible]ity, for example how well the code submitted follows programming stand[illegible]ation, presentation of the assignment, readability of the code, reusability[illegible]ganisation of code.

Submission Requirements

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

- **A PDF** file (created from the notebook) to be submitted through Turnitin submission link. Use the browser's print function to save the notebook as PDF. Please name this pdf file based on your authcate name (e.g. **psan002.pdf**)
- **A zip file** of your Assignment-2A folder, named based on your authcate name (e.g. **psan002.zip**). 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. Your ZIP file should only contain Assignment-2A.ipynb

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 [illegible] ith any other students. Students should consult the policy linked below [illegible] n.

https://www.[illegible]s/academic/policies/academic-integrity

See also the links u[illegible] rity Resources* in Assessments block on Moodle.

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.