InFlight: Practical 1 for Data-Intensive Systems

Deadline: March 22, 2024 at 9PM

1 Overview

This piece of coursework involves gaining knowledge and experience by working with Apache Spark in order to analyse a large dataset. **You are expected to have read and understood all the information in this specification at least a week before the deadline. You must contact the lecturers regarding any queries well in advance of the deadline.** Note that this coursework is worth **65%** of your coursework grade.

2 Pre-requisites

1. Have a basic understanding of Python programming
2. Understand Apache Spark and its components

3 Practical Requirements

You will develop an interactive application, called *InFlight*, with **Python and Apache Spark** in order to analyse a large dataset. The dataset, published by the United States Department of Transportation (DOT), is the *Airline Reporting Carrier On-Time Performance* of all US domestic flights between 1987 and 2020 [4]. The dataset, that is over 80GB and contains around 200 million records, shows the punctuality (on-time performance) of every US domestic flight that took place between 1987 and 2020. This dataset has already been downloaded for you and is available on the teaching servers in the School. Several elements of data are recorded, such as the originating airport of the flight, destination, when the flight took off versus its originally scheduled departure time, if there was a diversion, a security delay, etc. An initial metadata file, as well as additional helper files have also been included that will help you to fully understand the dataset. These files have all been downloaded from the DOT page [4] and are in a *SupplementaryCSV* directory. You should first perform an exploration of the dataset beforehand, making sure you fully understand the structure. Also, you will have to think about ways in which you will deal with the various columns, missing/unnecessary data and anything else you may identify in the task. In developing your solution, you may want to begin by processing a specific time period, or limited set of time periods, making sure your solution works correctly, before extending it to work across the full dataset. There are three parts to this task. Note that in order to get **all** the marks in each part, **all your queries must work correctly**.

All underlying queries for this task must be completed using Apache Spark.

4 Part 1

To gain a grade up to 13.5, you should implement a set of **core features** of *InFlight*:

1. Read in and store the dataset using Apache Spark. The total file size for the *airline.csv* file is >80GB in size which will take upwards of 10 minutes depending on your machine to load. Therefore, there is a file format that is optimised for Spark that reduces this load time to seconds with the total converted file size of the dataset being a few GBs. You must therefore convert/export the *airlines.csv* file into this file format and read/work with all the airlines data in this file format. The folder containing all the data you'll require is located at */cs/datasets/CS5052/P1/* on the teaching servers. This folder contains the main *airlines.csv* file, the *readme.html* file, in addition to a *SupplementaryCSVs* folder containing additional CSVs you'll need in conjunction with the main dataset. You can access this folder from both the labs as well as via SSH (into the teaching servers). Create a helper script that reads in **all** the CSV files from the folder and converts them into this special file format, storing the final files in your local directory to use. Also remember to read the *readme.html* webpage which contains information about all the various headings. In your report state what this file format is, why it is better than using the CSV, and attach your Python convert/export script that allows the user to convert their CSV files into this particular format.
2. Allow the user to search the dataset by year, displaying the total number of flights that took place that year. Given **both** a range of years and a comma separated list of years, display to the user the total number of flights that took place that year.
3. For any user specified year, display to the user in a neatly formatted fashion, the percentage of flights that (1) took off on time, (2) took off early or (3) took off late.
4. For any user specified year, display the top reason for cancelled flights that year.
5. In 1987, 1997, 2007 and 2017, what were the top 3 airports (and where were they) that recorded the most punctual take-offs?
6. Examining all the years in the 20th century, display the top three worst performing airlines. Define how you interpret the word "worst".

5 Part 2

To gain a grade up to 16.5, you must implement visualisations, and satisfy all the following **intermediate** requirements:

1. Allow the user to compare the performance of airports in **any** number of US **states** across the entire dataset. Justify how you will compare and present the data and what metric(s) you select to measure performance.
2. Using the above metric(s) you chose to measure performance, chart/explore the performance of the various states' airports that are located within the continental United States. This analysis should be detailed, use appropriate charts and visualisations, and should answer the following questions:

- Are there any specific regions, as defined by the United States Census Bureau, of the continental United States that had the best/worst performance across the years?
- Are there any specific states in the various regions that are having an effect on the overall performance of the region?
- Are there any trends in terms of performance that you can identify as you move between all the years of the dataset?

6 Part 3

In order to obtain a grade greater than 16.5, you must implement more **advanced** features. Part 3 should build on the functionality provided by Parts 1 and 2 and provide interesting reports and/or insights into the dataset. Interesting can be interpreted any way you see fit but should include advanced analysis (for example, using predictions). Some suggestions are below:

1. Explore the other fields of the dataset. Are there any connections you can find between other elements of data (such as days/months, flight distance, diversions, etc.) and delays?
2. Using machine learning, can you build a model that can predict future performance of airports across the dataset in the next ten years? Is your model accurate? Supply appropriate evidence such as RMSE, etc.
3. Provide more advanced analysis and create a set of advanced visualisations, for example a timelapse visualisation in a web application of flight density across the dataset from 1987 to 2020, amongst others.
4. Using some kind of advanced analysis, and making use of additional external resources, explore whether there were some times in the span of 33 years where performance was worse across certain airlines than others. Is there any reason you can find/suggestions for this?
5. Any other advanced feature of your choosing.

7 Constraints, interactivity of application and further points

Your queries must all be written in Python and Apache Spark. You must **not use Pandas for any part of this assignment**. This means, for example, that you cannot convert your Spark DataFrames into Pandas DataFrames in order to graph your data. There are ways of accomplishing this without Pandas. Your application also must be interactive. To this end, you are free to build either (1) a console-based application, (2) a web application, (3) a desktop application, or a combination of them all. **You cannot use Jupyter (or equivalent) Notebooks in this assignment**. For whichever application(s) you choose, you must include appropriate installation instructions. Aside from the above constraints, you are free to use any additional libraries, tools, packages, programming languages (for example in the case of the web application) that you see fit. Your code must be written in an excellent fashion, with clarity, comments, best practice code style and error handling, appropriate formatting

for the language(s) used, etc. Your reports must be clear, cogent, with excellent grammar and punctuation and have appropriate visualisations and answer all the questions above.

8 Deliverables

Please do not include the dataset with your submission. **You must submit a single .zip file by the deadline specified on MMS for this assignment.**

1. The entire source code
2. A README file describing how to run the application.
3. A .pdf report (not more than 3500 words) describing the design and implementation of your application in addition to any difficulties you encountered. The report must demonstrate any insight obtained by implementing the features in Parts 1-3. **Please include the word count at the end of your report.** In particular, it should include:
    - Summary of the core (Part 1), intermediate (Part 2) and advanced (Part 3) features you have implemented. Where appropriate, add discussion/justification as to why they have been included/your approach for implementation and answers to the questions posed in the task. For any feature implemented, please also include the reference to the specific lines of code that pertain directly to that query implementing that feature. This can be something as simple as Feature X (Code lines a-b in file N). Please however **do not include screenshots of your code in the report**. You are free to discuss/elaborate on certain queries in your report by including some code **snippets**, if you wish. These however must be written in a different font to distinguish them from the rest of your report.
    - Include a table which presents an overview of each of the features implemented in Parts 1, 2 and 3.
    - Any problems that you encountered and your solutions.
    - Any diagrams/charts you feel are necessary.
    - A reflective summary discussing the lessons learnt and experience that you have gained after finishing this practical.
4. A short video (with a voiceover) clip in a popular format, such as .mp4, demonstrating the execution of your application. This can be a simple 5-minute screen capture with you talking over the video to describe the functionality of your system. Treat it as a video which gives a quick overview of your solution and the functionality it supports. This video **should not be longer than 5 minutes**.

9 Marking

Note that to get all the full marks for each band (up to 13.5, 16.5, etc.) and move into the next band, **all** your functionality must be implemented correctly. This means **all** queries must fully work and answer the questions posed in the task for each band. To give an example, you cannot implement an extension in Part 3, with problems in Part 1 of your submission and expect to come into the 16.5+ mark band just because you implemented an extension. In

Part 3, you are free to implement whichever extensions you wish. Your work will be marked according to the standard mark descriptors in the School. You can find these in the School Student Handbook [1].

10 Lateness

The standard penalty for late submission applies (Scheme A: 1 mark per 24 hour period, or part thereof) [2].

11 Good Academic Practice

As always, the University policy on Good Academic Practice applies [3].

References

[1] University of St Andrews. *Assessment — CS Students Handbook*. 2023. URL: https://info.cs.st-andrews.ac.uk/student-handbook/learning-%20teaching/feedback.html#Mark_Descriptors.

[2] University of St Andrews. *Assessment | CS Student Handbook*. 2023. URL: https://info.cs.st-andrews.ac.uk/student-handbook/learning-teaching/assessment.html.

[3] University of St Andrews. *Good academic practice — Current Students — University of St Andrews*. 2023. URL: https://www.st-andrews.ac.uk/%20students/rules/academicpractice/.

[4] United States Department of Transport. *Download Page*. 2023. URL: https://www.transtats.bts.gov/DL_SelectFields.aspx?gnoyr_VQ=FGJ&QO_fu146_anzr=b0-gvzr.