Updated Date: 08/29/2024

Context:

Sync's customers run periodic production workloads such as a Databricks Job. Sync's entity for managing a given Job is called a **project**. Every time a job runs (say, twice daily), it submits metadata about that run to the associated project. Leading up to the next run, Sync will generate and apply a recommendation for how that customer should adjust their hardware configuration to achieve some desired outcome – typically minimizing cost.

The file `test_project_data.json` contains submission data for several different projects. Please code up solutions to the questions below in **Python** and make your solution available to us either by sending your solution files or making them available via a version control software such as Github. In addition to code, include a document with any relevant figures or written thoughts that are relevant to your analysis (no need to go nuts, just enough to see your process and to speak to when we meet in-person).

Questions:

1. [Forecasting] A critical feature for our modeling is **Data Size**. Develop a suitable forecasting model for **input_mb_read** for projects `480cc3f789be` and `235c88aba875`. If the two datasets necessitate different strategies then create two separate forecasting models. Note that forecasting is near term, within the next few submission periods at most.

2. [Modeling] The Job run duration has some relationship to the input data size. Develop a single closed-form model that suitably relates **spark_duration_hr** as a function of **total_bytes_read** for all projects in the dataset (not just the two from Q1). For example, `duration=m*total_bytes_read+b` is a possible solution where `(m,b)` are unique for each project (unfortunately you will not find that this family of curves is not suitable for all the data, it's not that easy!).

3. [Application] Use your solutions to (1) and (2) to predict the application duration 6 hours and 12 hours after the latest submission for both projects in (1).

Tips:

- Use good engineering practices wherever appropriate. "Good" here means practices that make code more readable and maintainable by others or your future self, so practices like *descriptive function names* and *type hints* are important. Less concerning is what *style* of programming – go with what you're most comfortable with.
- Comments which clarify any assumptions or more complex reasoning for pieces of code are always welcome.

- There is no absolute correct answer to these questions. At Sync we are solving problems that are yet unsolved. Sometimes this means doing our best to develop solutions given the data and knowledge at hand, knowing that there may be gaps in the solution. The best we can do is try to be aware and honest of where our solution may fall short.