Deliverable 1

Team Name: Waterboys

Rahmatullah Nekyar, Usman Siddiqui, Tony Zeng, Hao (Frank) Xu, Weiqiang Zhang

Architecture breakdown of UML Diagram

The two main features of the Pandas library reside in the data structures **DataFrame** and **Series**, both inheriting from NDFrame. DataFrames are two-dimensional, mutable, and potentially heterogeneous tabular data structure with labelled axes that form the core of the pandas library. Series are the one-dimensional analogue of DataFrames. DataFrame and Series have many composition links from different classes.

NDFrame has dependencies on the _Window objects, where they provide rolling/expanding/EWM calculations such as sum and mean. A use case of rolling sum would be df.rolling(3).sum, where the sum of current row and previous two are calculated.

GroupBy is used to split the data into groups based on some criteria. It is inherited by DataFrameGroupBy and SeriesGroupBy. Where DataFrameGroupBy is associated with DataFrame and SeriesGroupby is associated with Series. The GroupBy class is associated with the Resampler, which is used to change the intervals on which datasets are recorded. An example of a resampling function would be aggregating weekly datasets into day by day intervals.

Pandas arrays use NumPy arrays as wrappers instead of Python's list data structure, as numpy arrays are overall more efficient, allowing for better handling of big data. At the core, since it is a multi-dimensional data structure, it can represent matrices and vectors and implement operations such as slicing, numerical calculations and computations with better memory and speed performance in comparison to the base python list data structure.

One simple way we can improve the architecture is by applying factory design principles. The groupby methods of the Series and DataFrame class should be using an abstract function, either through inheritance or an interface, to generate the appropriate GroupBy objects. This will essentially be our factory method. This allows the groupby methods to be generalized as their only difference lies with the distinct GroupBy objects it creates.

Our Software Development Process

Kanban is an agile software development method. It uses a Kanban board to represent the workflow. The board contains columns ("To Do", "In Progress", "Done") for each stage of a task. Each team member can add tasks to the board and update it as the task progresses. Kanban focuses on flexibility, visibility and limiting WIP.

We chose Kanban for several reasons. Most of our team is still trying to familiarize ourselves with Pandas, so we see ourselves changing tasks and shifting priorities as we develop. We ruled out the Waterfall method because it is not dynamic or adaptive. Conversely, Kanban, or more generally agile development, provides the flexibility to tackle this issue.

We also decided against Extreme programming because it is a rigorous method which requires daily updates. The main focus of this project will be contributing to an open source project. This does not require constant code commits but rather bug fixes and feature upgrades at deliverable due dates.

Kanban is also simple to use and easy to adopt. This is great for our team because we want our development method to be an aid rather than an obstacle. It provides clear visualization of our workflow, which makes progression feel meaningful. One of the main ideas of Kanban is limiting work in progress. This importantly increases our code quality, and we see it increasing our productivity as well. Kanban also encourages leadership from every member of the team. We are all capable and responsible developers that can take advantage of this principle.
south
Specifically, this is how we will use Kanban:

- We will use trello as our Kanban board
- Planning will be done when deliverables are released, tasks will be created with varying due dates
- Tasks will be added and re-prioritize as needed
- WIP limited to 2 tasks per person
- Daily board meetings will happen at 9pm in our messenger group chat

A potential con can be the bottlenecking tasks. Certain tasks that are bottlenecks may delay kanban production schedule, unless we detect the bottleneck and focus resources and our time onto it. Another con could be outdated tasks leading to confusion. Since development is dynamic, some tasks may become irrelevant, leading to the board being flooded. Good management of the board and a backlog column for these tasks will remove board clustering. So overall, despite the potential cons, Kanban is a great software development process that will fit our projects needs.