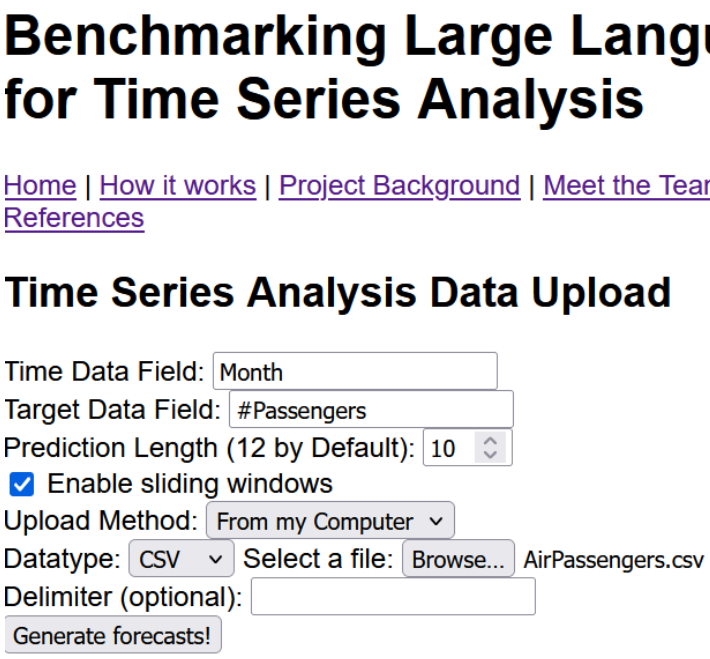
CSE 4940 Week 14 (Spring 2025) - Sage Pia

3/3/2025

My goal for this week was to find a way to animate a sliding window over a Grafana time series panel. After some searching, I was unable to find any resources and concluded that I would need to build a plugin on my own to modify time series. Since this did not seem like a viable solution, I then tried adding a new panel with buttons above the time series. The idea was to simulate a sliding window through a left and right button that changed the data and time range of the time series display when clicked. This functionality would require direct communication between Grafana and my Flask application. I tried using HTML in both a standard Text Panel and then in a Business Text Panel, but neither of these worked, as Grafana sanitizes untrusted HTML and prevented the buttons from working.

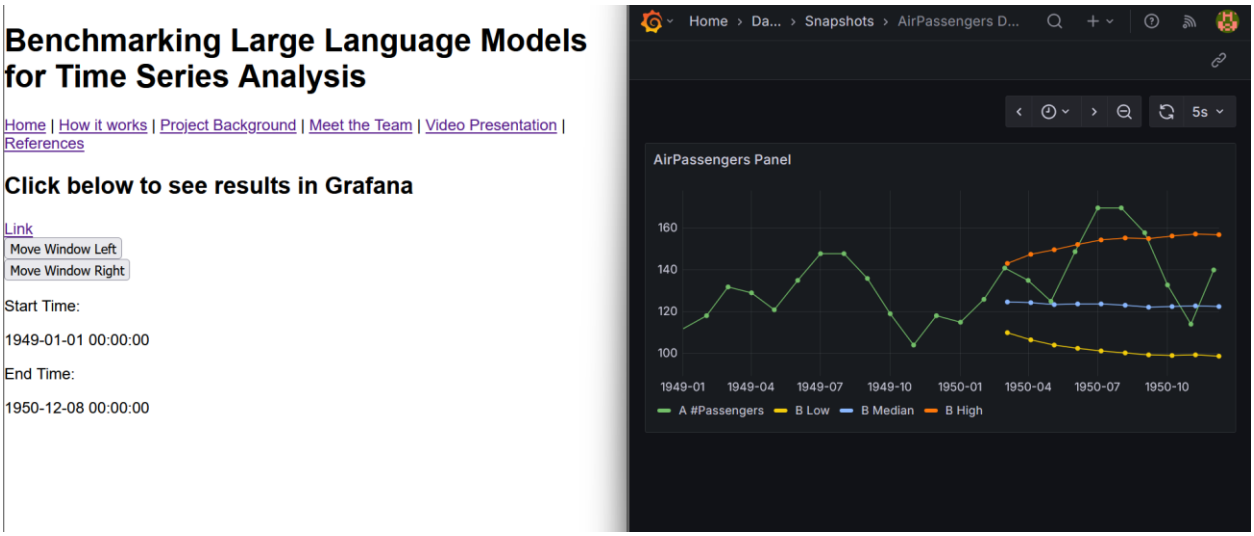
With no other choice, I pivoted to developing these buttons on the Flask app. To trigger sliding windows, the user must enable them. I also added an optional prediction length field.



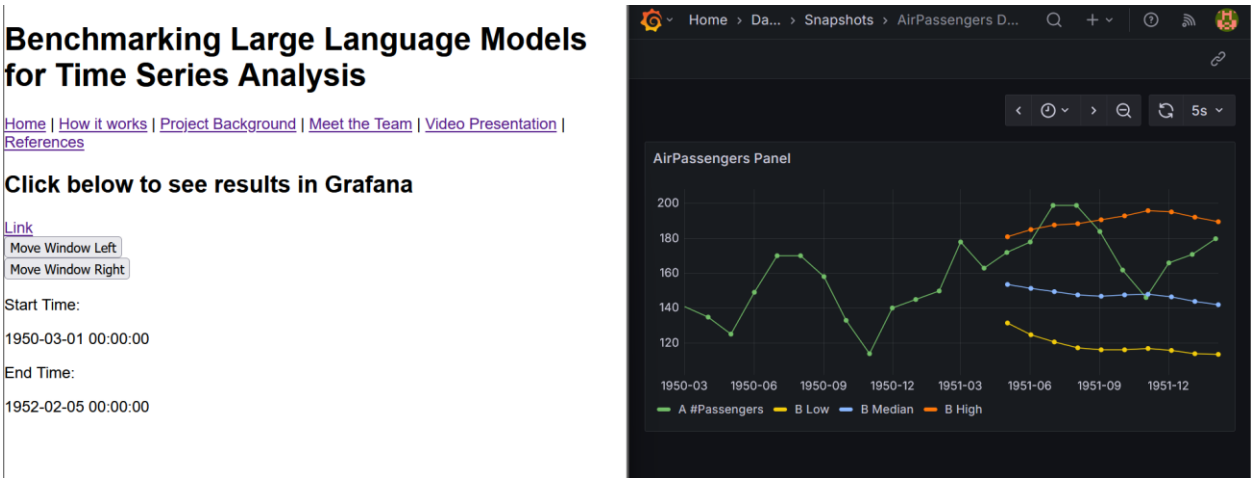
When creating windows, I didn't want to set a step value between windows of 1, as this would take too long to compute. The ETTh datasets, for example, are over 17000 entries long, and this would take too much time and storage. For now, I partition the data between 10 windows. The context length matches the distance of steps in small datasets and goes up to 512, since this is the maximum amount of historical data Chronos can handle. For

larger datasets, some data points will be skipped if they are not put into one of the 10 partitions.

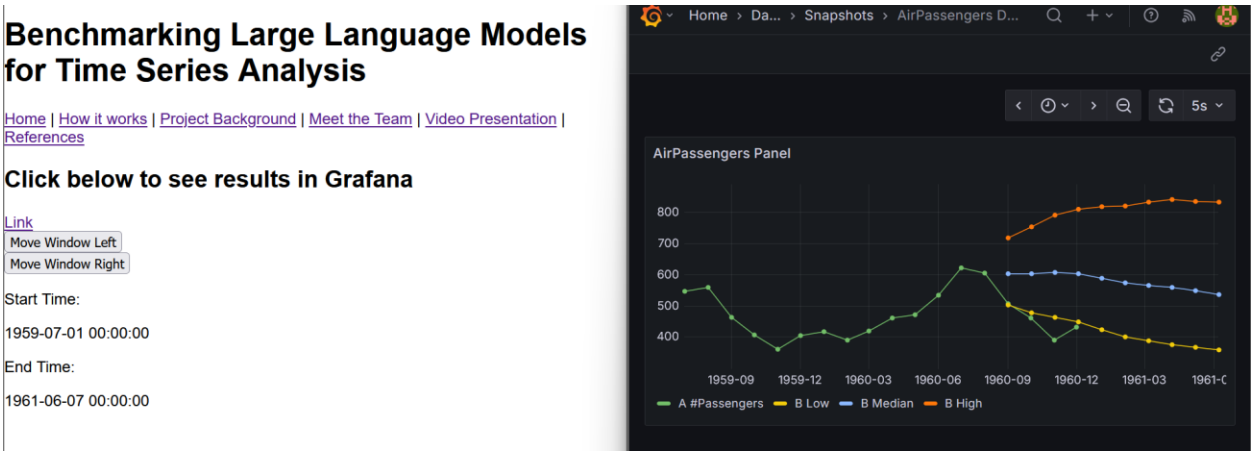
After clicking “Generate Forecasts!”, the clicking on the link automatically opens a new tab. This is important because the new buttons “Move Window Left” and “Move Window Right” shift the location of the sliding window. Also, originally, I only displayed forecasts instead of comparing them against the actual results. Now, the actual data is shown alongside the forecasts.



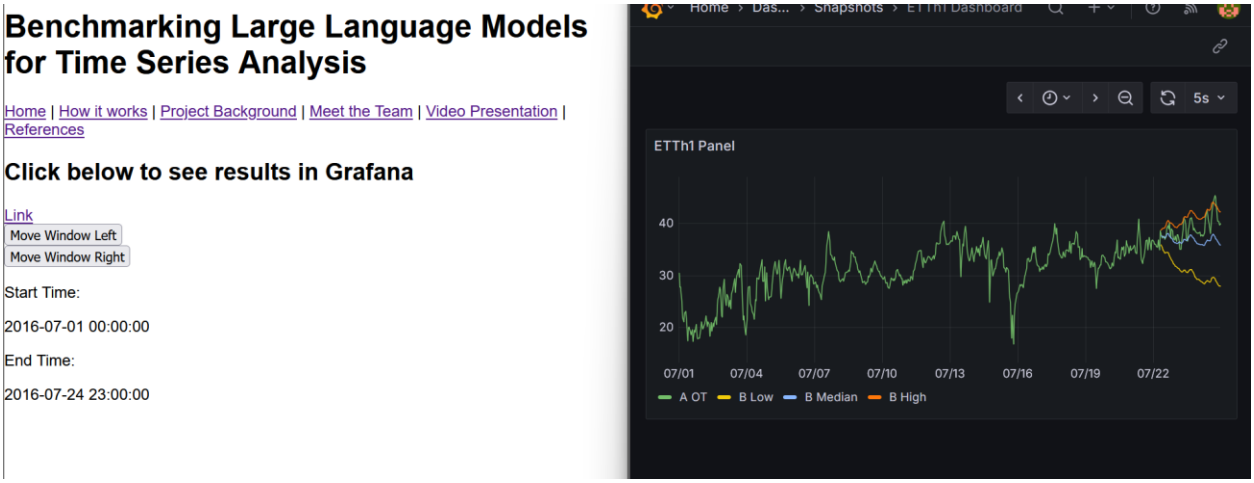
In this picture below, I clicked ”Move Window Right”, which accordingly shifted the time range of the data. As indicated by the refresh arrows and label that says ”5s”, clicking the button calls the Flask app and updates the data being displayed. The ”Start Time” and ”End Time” fields contain the latest bounds of where the window should be, as Grafana does not automatically update the range when the data updates.



Just to note, in the final window, the predictions can certainly go beyond the range of real data.



Another example of a sliding window of ETTh1.



Also, with Professor Song's help, I spoke to Technical Support Consultant Scott Cathcart and set up a Virtual Machine for us to store project files. This will help us more efficiently manage our project, as I have been doing much of the work locally and have not been able to share my files. However, after trying to ssh into the VM, I could not connect. We will need to reach out to Scott soon and figure out how to connect.