Sloan Digital Sky Survey Software Development: Week 5 Progress Report

Riley Thai, Andy Casey (Supervisor)

December 18, 2023

1 Aims and Objectives

1. Develop a web application for multi-dimensional exploration of SDSS-V data.
2. Work on and contribute to the development of open-source Python libraries.

2 Background

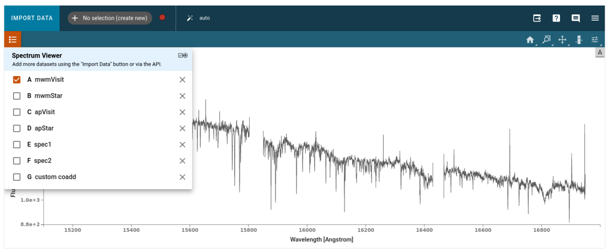
The Sloan Digital Sky Survey (SDSS) is one of the largest, longest running, and most used astronomical surveys. Started in 1998 (Almeida et al. 2023), the SDSS provides all-sky, multi-epoch spectroscopy using telescopes in both hemispheres, providing data used to probe the emergence of chemical elements, reveal the inner mechanisms of stars, and investigate the origin of planets. The latest generation of the survey, SDSS-V, aims to conduct the first homogeneous survey using an optical, ultra-wide integral-field spectroscopic map of the interstellar gas, pioneering spectroscopic monitoring and revealing changes on both short and vast timescales (Kollmeier et al. 2017). SDSS data over all survey phases has been cited more than 650,000 over 111,000 refereed papers (Almeida et al. 2023).

The SDSS has previously offered a simple web application for end users to access, explore, and investigate spectra and other data. However, the fifth generation of the survey now offers a large catalog of stars with complete stellar labels. With a larger set of spectral types explored (Majewski et al. 2017), and an even larger set of stellar labels (Casey et al., in prep), there is a need for a new tool which provides powerful exploration and visualization of large and vast datasets.

3 `specutils` Development

To be familiar with the variety of SDSS-V datatypes, I first began this project by developing loaders for the new SDSS-V datatypes. Within the Python library `specutils`, which provides tools to the `astropy` library for handling spectral data, there are a class of objects used for representing spectra, named `Spectrum1D` and `SpectrumList`. I created default loading functions for both the new and updated datatypes within the SDSS-V dataset. Built in as the sub-method to each `Spectrum` class, by specifying a filepath, a user can read a single or all spectrum from a given SDSS data file.

Most importantly, the default loader allows for any third party which use the `specutils Spectrum` objects to load the new SDSS-V datatypes, such as the Jdaviz spectrum viewer, shown in Figure 1.



Add more datasets using the "Import Data" button or via the API.

Figure 1: Spectrum data from a BOSS spectrograph, as shown within jdaviz SpectrumViewer.

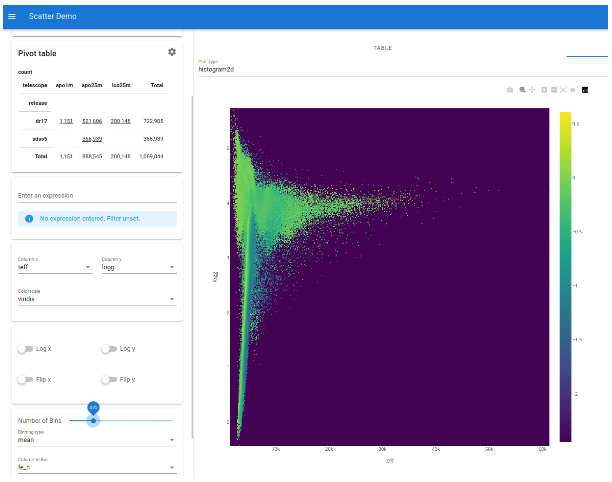
4 Plotting Dashboard Development

After completing the new loaders, I began work on the parameter dashboard. The dashboard leverages the out-of-memory dataframes and high-speed aggregations of `vaex`, and combines them with the React-like framework in Python, `solara`, to deliver an intuitive, but powerful visualization tool for the large SDSS datasets. The data is read out of memory from an Apache Arrow format, dramatically reducing system overhead, both client and server-side.

Currently, the dashboard can be used to investigate any uploaded file, or a small subset of SDSS-V data. The main method of interaction is through a sidebar on the left, which allows the user to change plotting parameters, such as the x-y scale, columns, number of bins, and type of statistic to overplot with.

The dashboard has 3 modes: histogram, in one and two dimensions, and a scatter plot. These were likely to be the most useful for data exploration, so they were implemented first.

The dashboard's `histogram2d` mode heavily utilizes `vaex`'s aggregated statistics over grids, which allows for complex multi-dimensional plotting, without suffering from the pitfalls of scatter plots, such as overplotting the dataset or underrepresenting


Scatter Demo

Pivot table

count

telescope	apo1m	apo25m	lco25m	Total
release				
dr17	1,151	521,606	200,148	722,905
sdss5		366,939		366,939
Total	1,151	888,545	200,148	1,089,844

Enter an expression

No expression entered. Filter unset.

Column x: teff
Column y: logg
Colorscale: viridis
Number of Bins: 470
Binning type: mean
Column to Bin: fe_h
Plot Type: histogram2d

Figure 2: A display showing the histogram2d plotting mode with sample data. The data is read from a `parquet` file, which reduces the memory overhead of the application. It is then delivered to the application by a `vaex`dataframe, and plotted with `Plotly` and shown in a `solara`application (the dashboard).

through downsampling. This view is shown in Figure 2.

The most advanced feature implemented so far is the expression editor, which allows the user to leverage vaex's high-speed computations to filter the dataset. This can be combined with other filtering methods, such as a pivot table and/or selection, to directly select a complex subset of data. This functionality is shown in Figure 3.

5 Difficulties encountered

1. **`vaex` and `solara` documentation is often incomplete.**
    - Throughout development, I found several functions within the source code that were not apparent on the API documentation.
    - This is common throughout open-source development, due to its decentralized nature, but I will also endeavour to ensure I properly document my work.
2. **Error messages from `solara`are obtuse (no traceback to soure code).**
    - When testing the web app, any bugs which arise often don't provide clear tracebacks, often stopping at one function, or only returning that there was an error during rendering.
    - This is due to the way `solara`runs the web application on its own server, which prevents these from appearing.
    - This has made it more difficult, but not impossible, to track down bugs, as I can't use debuggers or other tools directly.

6 Future plans

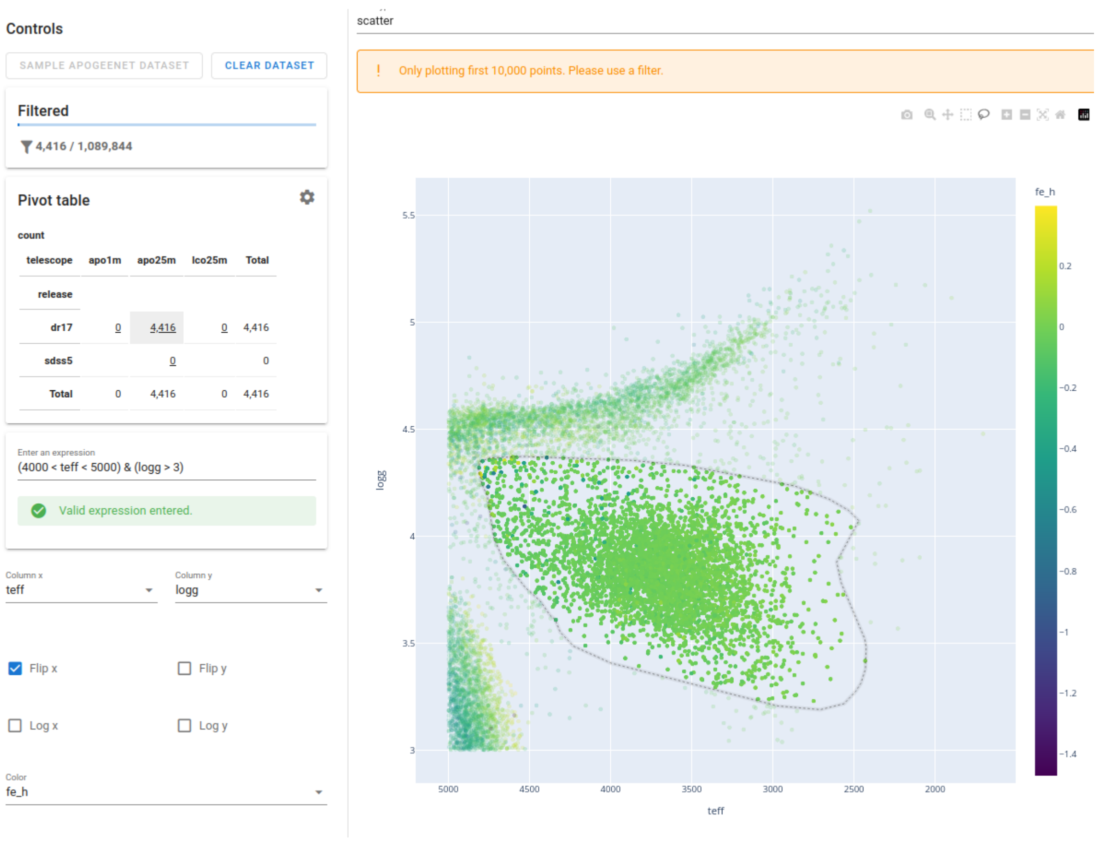
1. **Expand cross-filtering functionality.**
    (a) We can expand the cross-filtering functionality to further leverage `vaex`functionality.
    (b) Possible expansions include:
        - using the groupby object, similar to how it is implemented in `pandas`.
        - allowing for the selection of a specific bin within the histogram2d view.
        - expanding to have different subsets saved for the user to use and explore individually, rather than across the entire dashboard.
2. **Implement the webapp within the proprietary backend.**
    (a) SDSS uses a backend derivative based on FastAPI. Currently, the webapp we've developed works on Starlette, but hasn't been implemented in the proprietary backend yet.
    (b) This is due to a few bugs during startup of the development Docker setup which we still need to resolve, which will help other members of the collaboration to test it as well.

References

Almeida, Andrés et al. (Aug. 1, 2023). "The Eighteenth Data Release of the Sloan Digital Sky Surveys: Targeting and First Spectra from SDSS-V". In: *The Astrophysical Journal Supplement Series* 267. ADS Bibcode: 2023ApJS..267...44A, p. 44. ISSN: 0067-0049. DOI: `10.3847/1538-4365/acda98`. URL: `https://ui.adsabs.harvard.edu/abs/2023ApJS..267...44A` (visited on 08/22/2023).

Kollmeier, Juna A. et al. (Nov. 2017). "SDSS-V: Pioneering Panoptic Spectroscopy". In: *arXiv e-prints*. _eprint: 1711.03234, arXiv:1711.03234. DOI: `10.48550/arXiv.1711.03234`.

Majewski, Steven R. et al. (Sept. 1, 2017). "The Apache Point Observatory Galactic Evolution Experiment (APOGEE)". In: *The Astronomical Journal* 154. ADS Bibcode: 2017AJ....154...94M, p. 94. ISSN: 0004-6256. DOI: `10.3847/1538-3881/aa784d`. URL: `https://ui.adsabs.harvard.edu/abs/2017AJ....154...94M` (visited on 09/20/2023).



Filtered

4,416 / 1,089,844

Pivot table

count

telescope	apo1m	apo25m	lco25m	Total
release				
dr17	0	4,416	0	4,416
sdss5		0		0
Total	0	4,416	0	4,416

Enter an expression: (4000 < teff < 5000) & (logg > 3)

Valid expression entered.

Column x: teff

Column y: logg

[✓ Flip x] Flip y

Log x Log y

Color: fe_h

! Only plotting first 10,000 points. Please use a filter.

Figure 3: A showcase of the full cross-filtering functionality within the scatter plot. A filter is applied by an expression, then a subset is chosen using the pivot table (left). Finally a subset of the data can be selected within the scatter plot via the lasoo tool.