Introduction

The purpose of this work is to create a dashboard that takes exploratory scientific, organizational, and agency-level metrics. The dashboard takes results offered international of available tools. Python and its packages are used to create visualizations for the dashboard. Jupyter Notebook is used to develop the dashboard. Pandas is used to manipulate the data. Matplotlib and Seaborn are used to create the static three-visualizations. Then the dashboard of them yields a plotted against total variables, with three different color-coding scales. Finally, Plotly is used to create the dashboard. The dashboard allows the user to interact with the data. The user can hover over the plotted points, zooming a specific area of interest, or panning in the workspace. Jump-ties separate the data and view parts to display the characteristics that specific factor.

Importing Libraries

Data Wrangling Libraries

```
import pandas as pd
import numpy as np
```

Data Visualization Libraries

Since Matplotlib is used for static visualizations, Both plotly-express and plotly.graph_objects are used. Plotly Dash is used to link the visualizations and generate the dashboard. Jupyter Dash package is used to display the dashboard within the Jupyter Notebook.

```
%matplotlib inline
%matplotlib inline
%matplotlib inline
%matplotlib inline
```

```
import plotly.express as px
import plotly.graph_objects as go
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.ticker as mticker
import matplotlib.cm as cm
import matplotlib.colors as mcolors
import matplotlib.gridspec as gridspec
import jupyter_dash
import dash
import dash_core_components as dcc
import dash_bootstrap_components as dbc
import dash_html_components as html
```

```
import numpy
```

Due to several functionality limitations, the Matplotlib API wrapped Calling and plotting per requests contained the desired classes. The Seaborn API was used to replace the calls calculus step, then calculus calc filter as planned for the program.

```
%matplotlib inline
%matplotlib inline
```

```
from dash import dcc
from dash import html
from dash.dependencies import Input, Output, State, ClientSideFunction
```

Dataset

The assignment uses three datasets, one for each visualization. The first two datasets are exoplanet using Binning Technique TAPS. The third dataset is the Cory Kessler Dover Time Series Dataset. The exoplanet datasets are used to create the static visualizations and the interactive and changing versions for the first two datasets (second for exoplanet). However, the instances used the second version of exoplanet. The Cory Kessler Dover Time Series Dataset is used to create the static visualization and the changing version.

The first dataset (DS) consists of a selection of DS2 comes as rows, as well as a range of features as columns. For the assignment bands, including between 3 months and 3 years are selected. Only with binning times are dropped. The resulting dataset DS contains 3000 bands with 19 features.

Feature	Column Name	Number Type
ID of the exoplanet	id	Integer
Total JD of the transit	total_jd	Decimal
Total Flux of the transit	total_flux	Decimal
Total Error of the transit	total_error	Decimal
Flux error before binning	flux_error_before	Decimal
Flux error after binning	flux_error_after	Decimal
Number of data points before binning	n_before	Integer
Number of data points after binning	n_after	Integer
Transit Depth	transit_depth	Decimal
Transit Duration	transit_duration	Decimal
Period	period	Decimal
Impact Parameter	b	Decimal
Stellar Radius	stellar_radius	Decimal
Planet Radius	planet_radius	Decimal
Planet Temperature	planet_temp	Decimal
Stellar Temperature	stellar_temp	Decimal

The second dataset (dss_data) consists of the various of and variable Binning Press. The source datasets DS used (DS2 and DS1) also contain the same features as the first dataset. Each one of the features are used as the input variables. Binning Technique and binning statistics are the same of the first series.

The following is the table which shows the features and variables used to the manager. The Binning Statistics contain one numerical variable. Only 2 of the 10 features are the same of the features are related to different directions. A planet surface could be using hyperbolic and parabolic orbits (trajectories). Hyperbolic orbits are open orbits.

Feature	Column Name	Number Type
ID of the exoplanet	id	Integer
Total Flux of the transit	total_flux	Decimal
Total Error of the transit	total_error	Decimal
Flux error before binning	flux_error_before	Decimal
Flux error after binning	flux_error_after	Decimal
Number of data points before binning	n_before	Integer
Number of data points after binning	n_after	Integer
Transit Depth	transit_depth	Decimal
Transit Duration	transit_duration	Decimal
Period	period	Decimal

Calling Binning APIs

Combinating first two datasets. The part should not be executed by the user.

```
df = pd.read_csv('dss_data.csv', encoding='latin-1')
```

```
df = df.drop(['Unnamed: 0'], axis=1)
df = df.drop(['id'], axis=1)
df = df.drop(['total_jd'], axis=1)
df = df.drop(['flux_error_before'], axis=1)
df = df.drop(['flux_error_after'], axis=1)
df = df.drop(['n_before'], axis=1)
df = df.drop(['n_after'], axis=1)
df = df.drop(['transit_depth'], axis=1)
df = df.drop(['transit_duration'], axis=1)
df = df.drop(['period'], axis=1)
df = df.drop(['b'], axis=1)
df = df.drop(['stellar_radius'], axis=1)
df = df.drop(['planet_radius'], axis=1)
df = df.drop(['planet_temp'], axis=1)
df = df.drop(['stellar_temp'], axis=1)
df = df.drop(['Unnamed: 0.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1'], axis=1)
df = df.drop(['Unnamed: 0.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1
```