IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Author - Michael Rafferty

Feb 7 2023

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context
- Problems you want to find answers

Section 1

Methodology

Methodology

Executive Summary

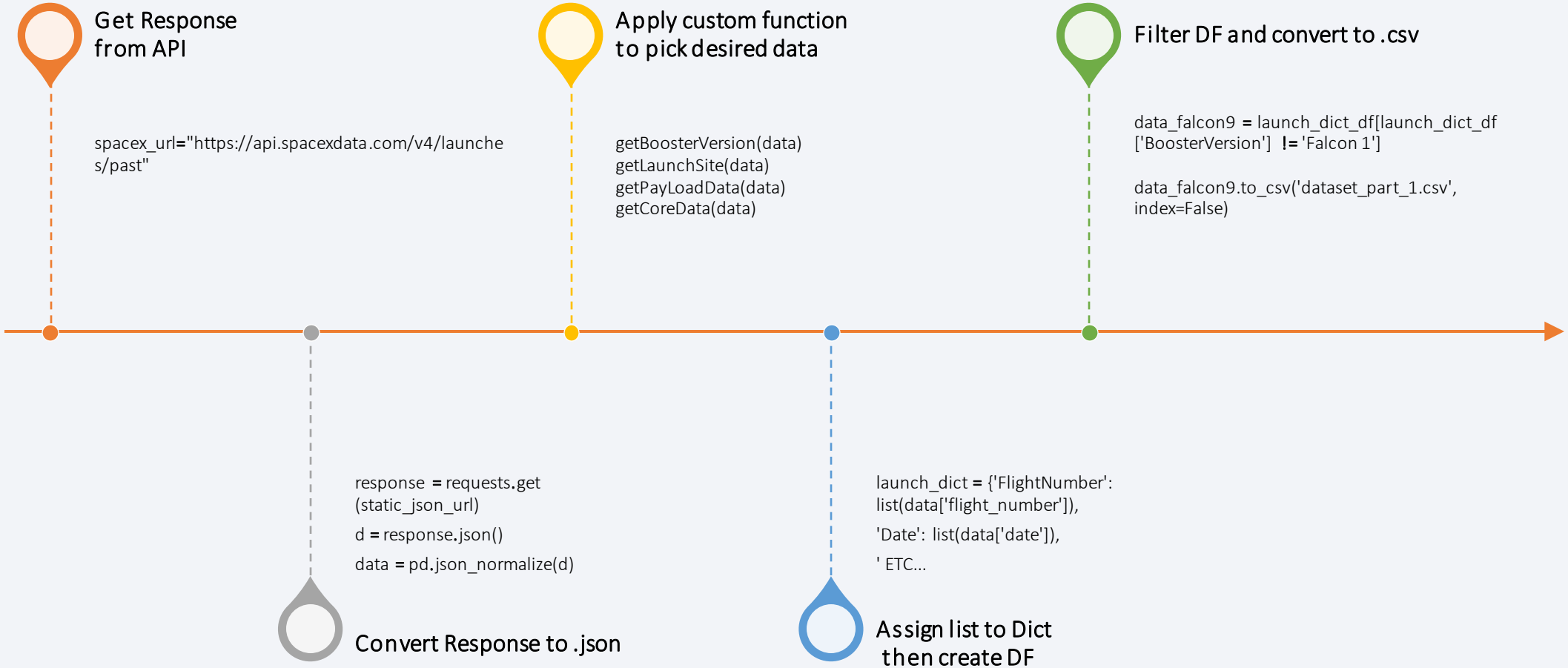
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
- You need to present your data collection process use key phrases and flowcharts

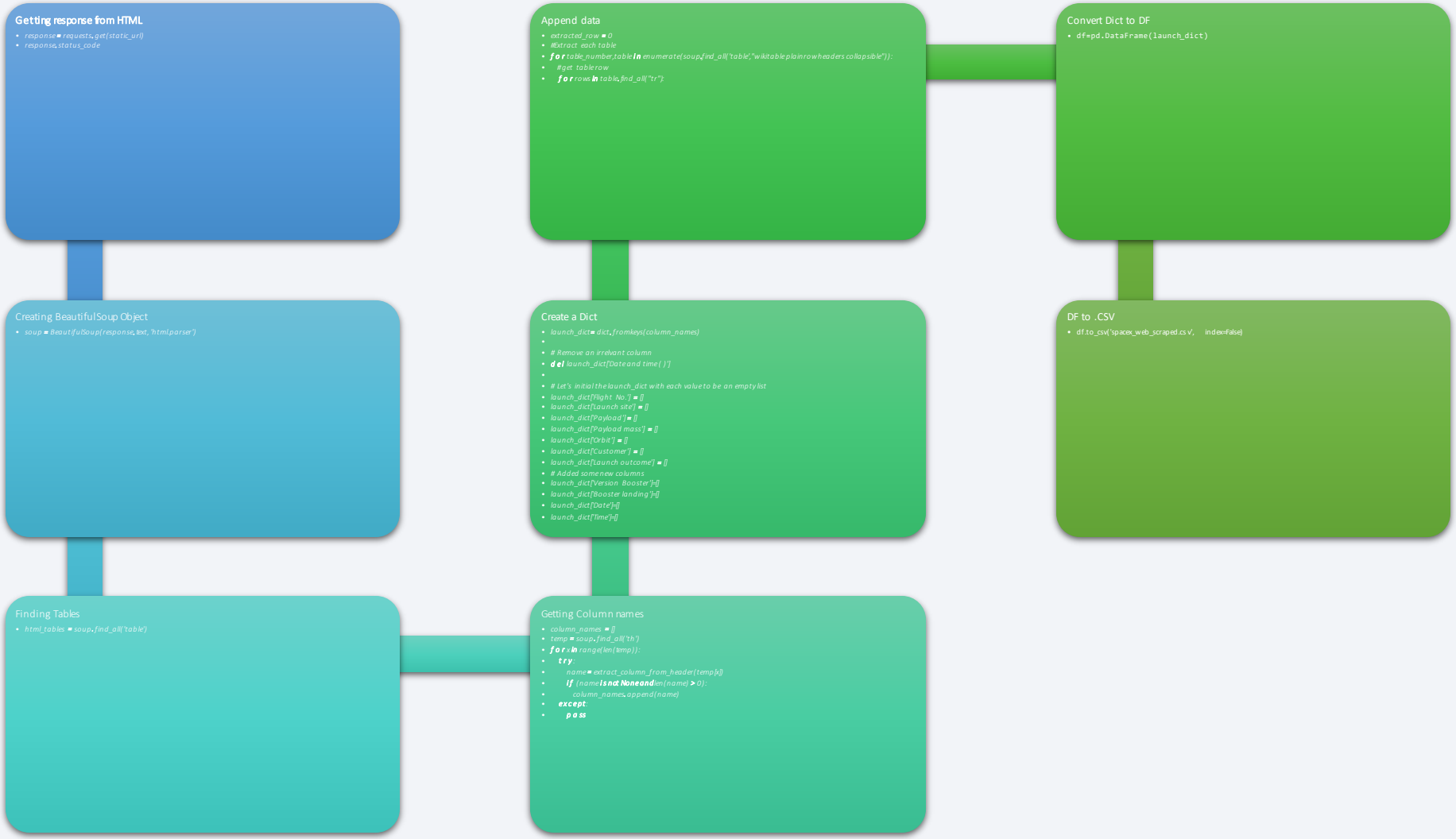
Data Collection – SpaceX API

Git Hub Link



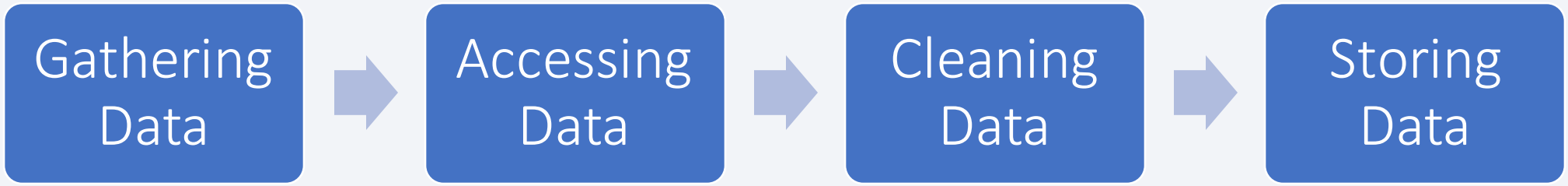
Data Collection - WebScraping

Git Hub Link



Data Wrangling

- The data wrangling process involves several steps, the goal is to take incoherent, inconsistent or incomplete data and turn it into data that can be used reliably in a future model.



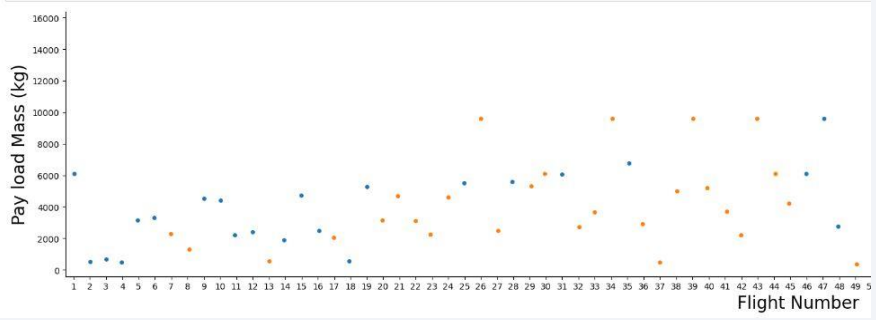
EDA with Data Visualization

Git Hub Link

Scatter Plot Graph:
[sns.catplot()]

- Flight Number and Launch Site
- Payload and Launch Site
- FlightNumber and Orbit type
- Payload and Orbit type

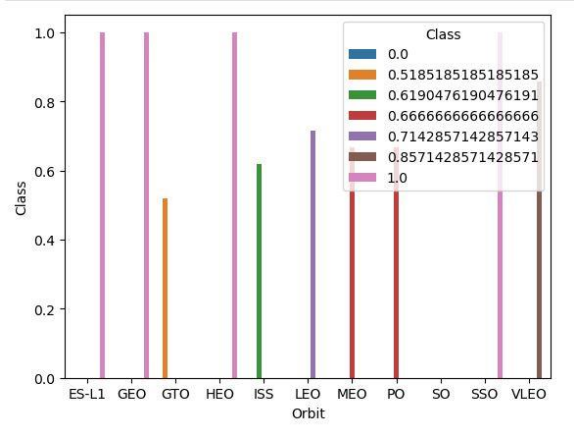
Scatter plots are a useful tool for data visualization, particularly for understanding relationships between two continuous variables. They are good for identifying outliers, spotting trends, and are easy to interpret. They can also handle large datasets.



Bar Graph:
[Sns.barplot()]

- success rate of each orbit type

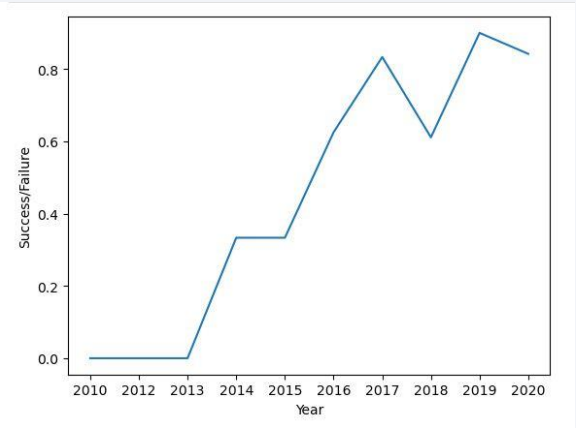
Bar graphs are a simple and effective way to represent and compare data. They are good for comparing values, visualizing frequencies, showing trends over time, and are suitable for small datasets. They are also easy to interpret.



Line Graph:
[Plt.plot()]

- launch success yearly trend

Line graphs are effective for visualizing continuous data over time, showing trends and patterns, comparing values, and are easy to interpret. They are best suited for continuous data with many data points.



EDA with SQL

Git Hub Link

SQL queries that were made:

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
- Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

Build an Interactive Map with Folium

Git Hub Link

We utilized Folium to better visualize the physical locations of the launch sites, relative to other points of interest (ie Highways, Railways, Cities etc...) and used PolyLines to measure distance between points. We also used color coded labels to notate the success or failure of launches at each site.

Objects used in this model were

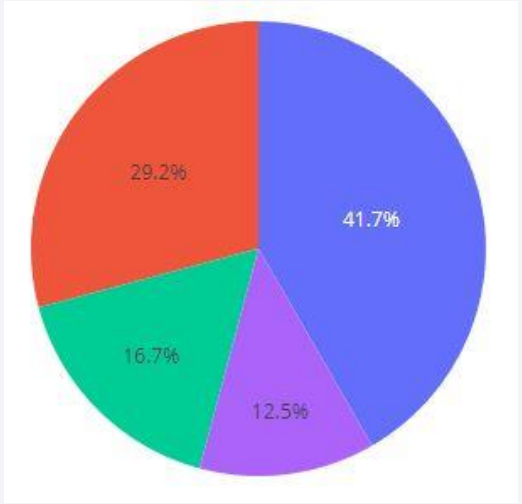
- *Map Markers - folium.Marker()*
- *Icon Marker - folium.Icon()*
- *Circle Marker - folium.Circle()*
- *Poly Line - folium.PolyLine()*
- *Marker Cluster - folium.MarkerCluster()*

Build a Dashboard with Plotly Dash

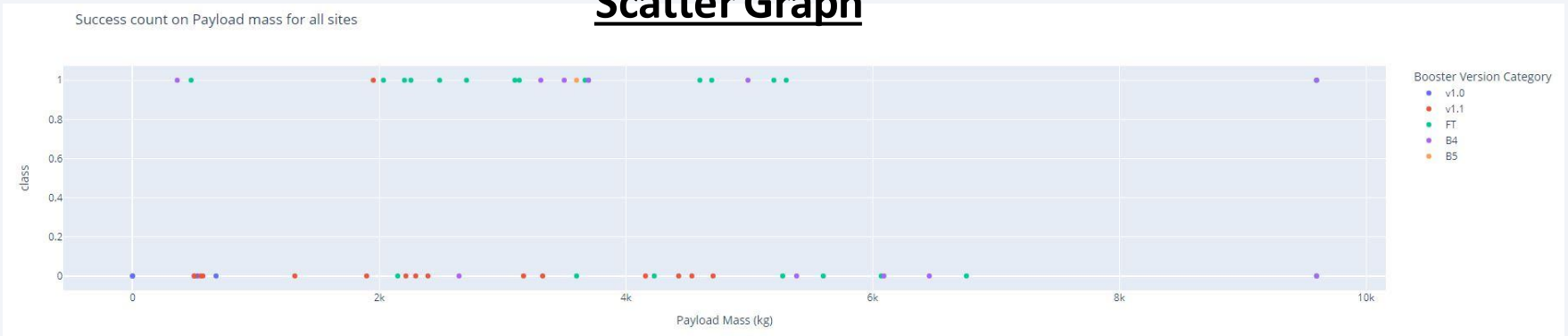
Git Hub Link

Plotly Dash is a popular open-source framework for data visualization that offers interactive, customizable, and wide range of charts with a built-in reactive framework. It has a simple API, making it easy to use even for those without extensive programming experience. Plotly Dash provides a flexible and powerful platform for data visualization that is suitable for various projects.

Pie Chart



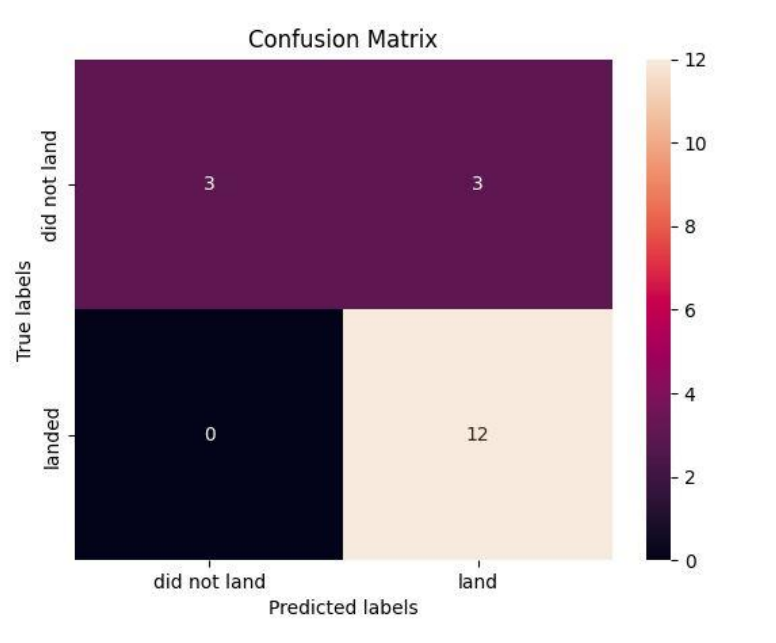
Scatter Graph



Predictive Analysis (Classification)

Git Hub Link

We used a variety of models in order to determine the best method for accurately predicting future Launch results. The models we used were Logistic Regression, SVM, Decision Tree and KNN. We used a confusion matrix to visualize the data.




```
Test set Accuracy of LogReg 0.8333333333333334
Test set Accuracy of SVM    0.8333333333333334
Test set Accuracy of Tree   0.8888888888888888
Test set Accuracy of KNN    0.8333333333333334
```

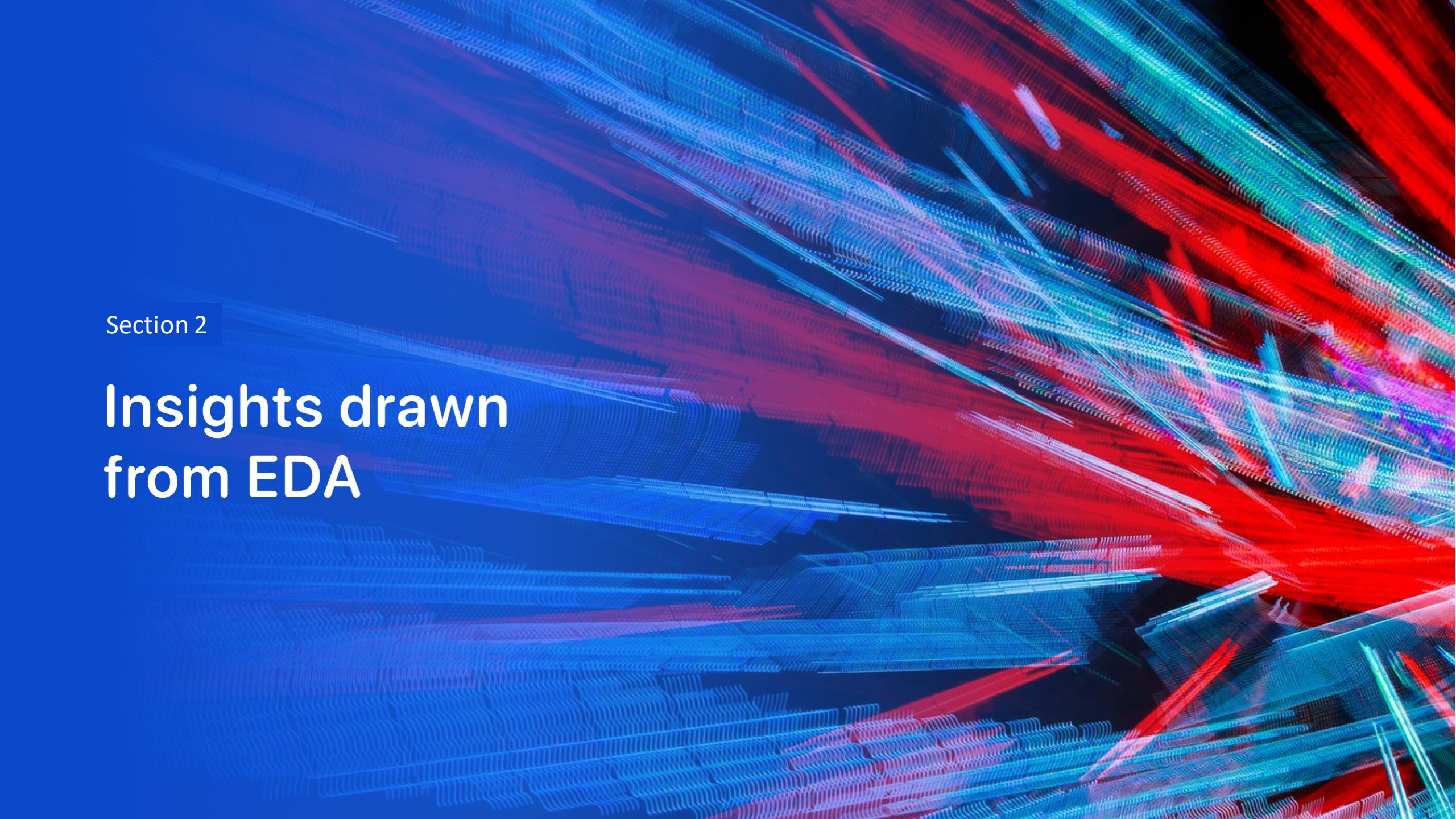
As seen here, the Decision Tree wins in accuracy

Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

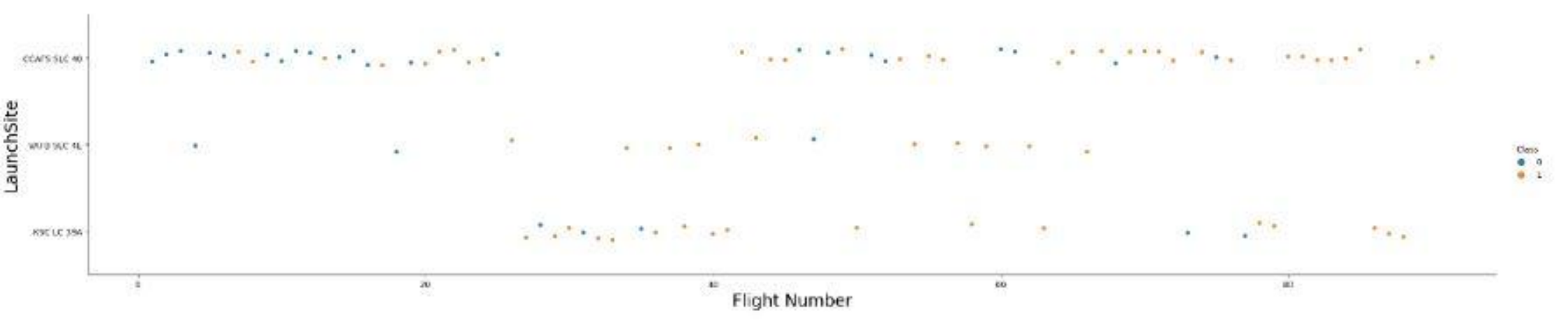
Section 2

Insights drawn from EDA



Flight Number vs. Launch Site

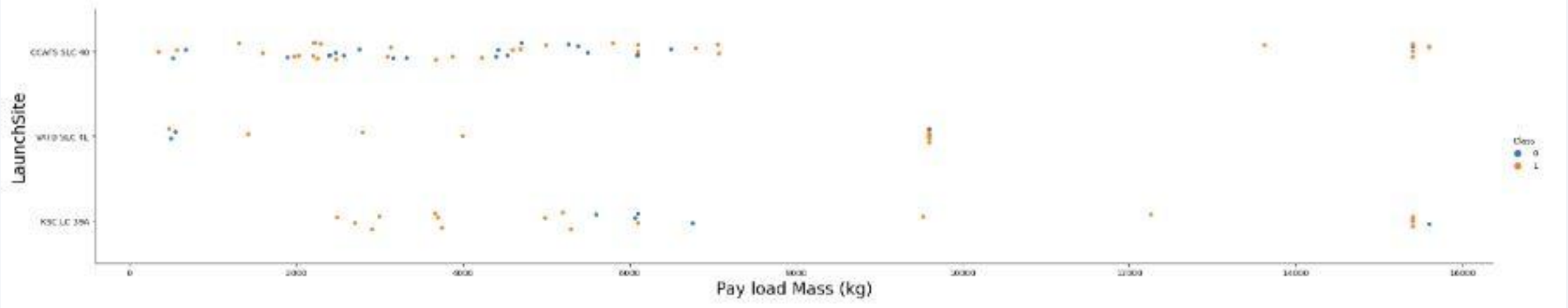
Git Hub Link



CCAFS SLC 40 has the most launches of all the sites. Additionally, the more launches a site performs, the better success rate they have.

Payload vs. Launch Site

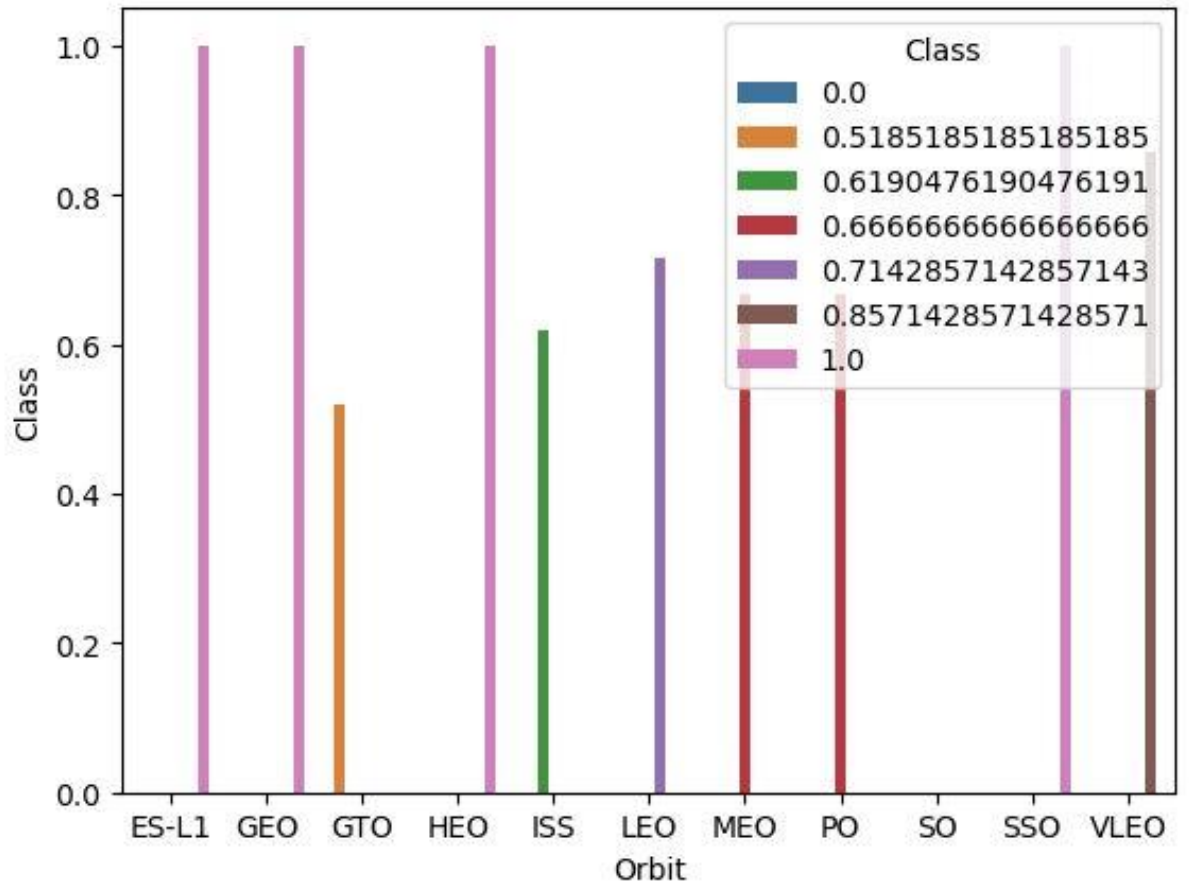
Git Hub Link



The higher payload launches appear to have a much higher rate of success, compared to the lighter payload launches.

Success Rate vs. Orbit Type

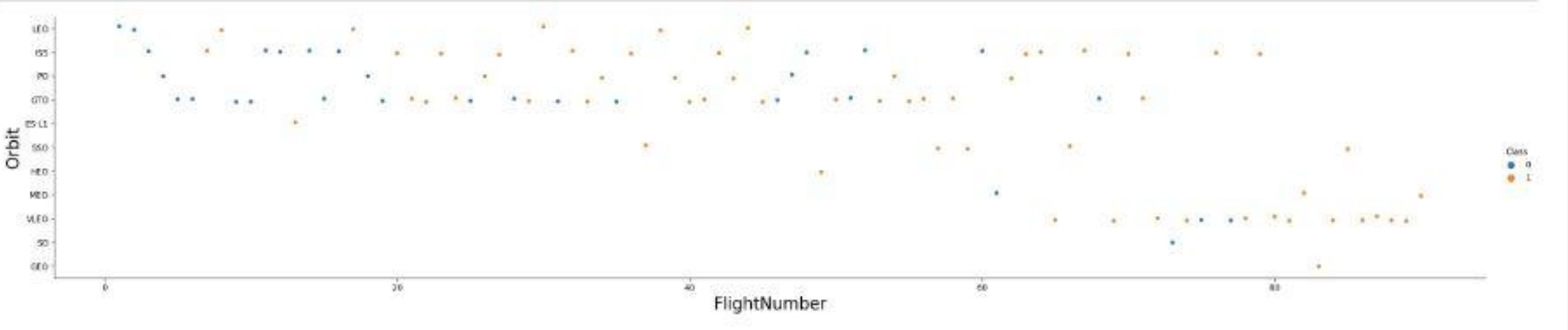
Git Hub Link



ES-L1, GEO, HEO and SSO have the highest success rate of all the Orbit Types, while SO has the worst.

Flight Number vs. Orbit Type

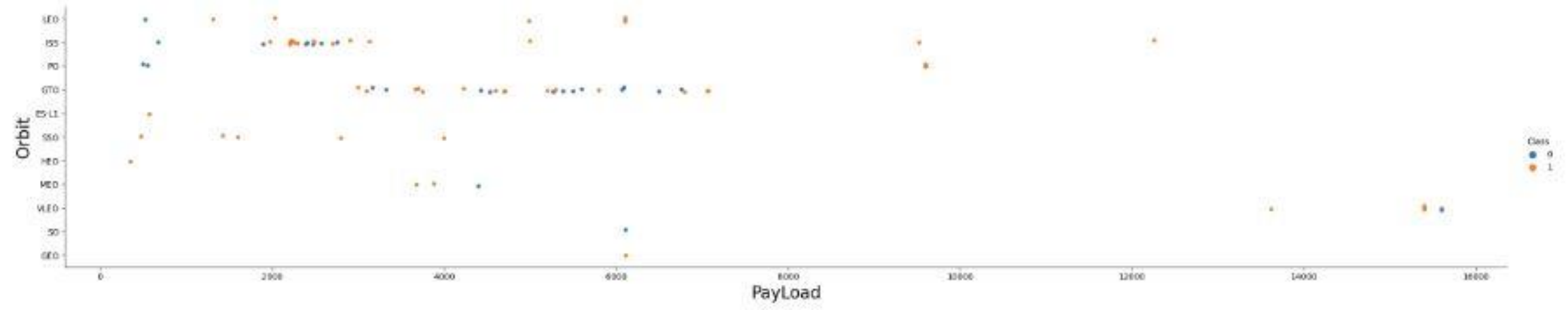
Git Hub Link



This chart shows that SpaceX stuck with mainly LEO, ISS, PO and GTO orbit types at the beginning of their launch history, then started branching to other orbit types, appearing to focus on VLEO recently.

Payload vs. Orbit Type

Git Hub Link

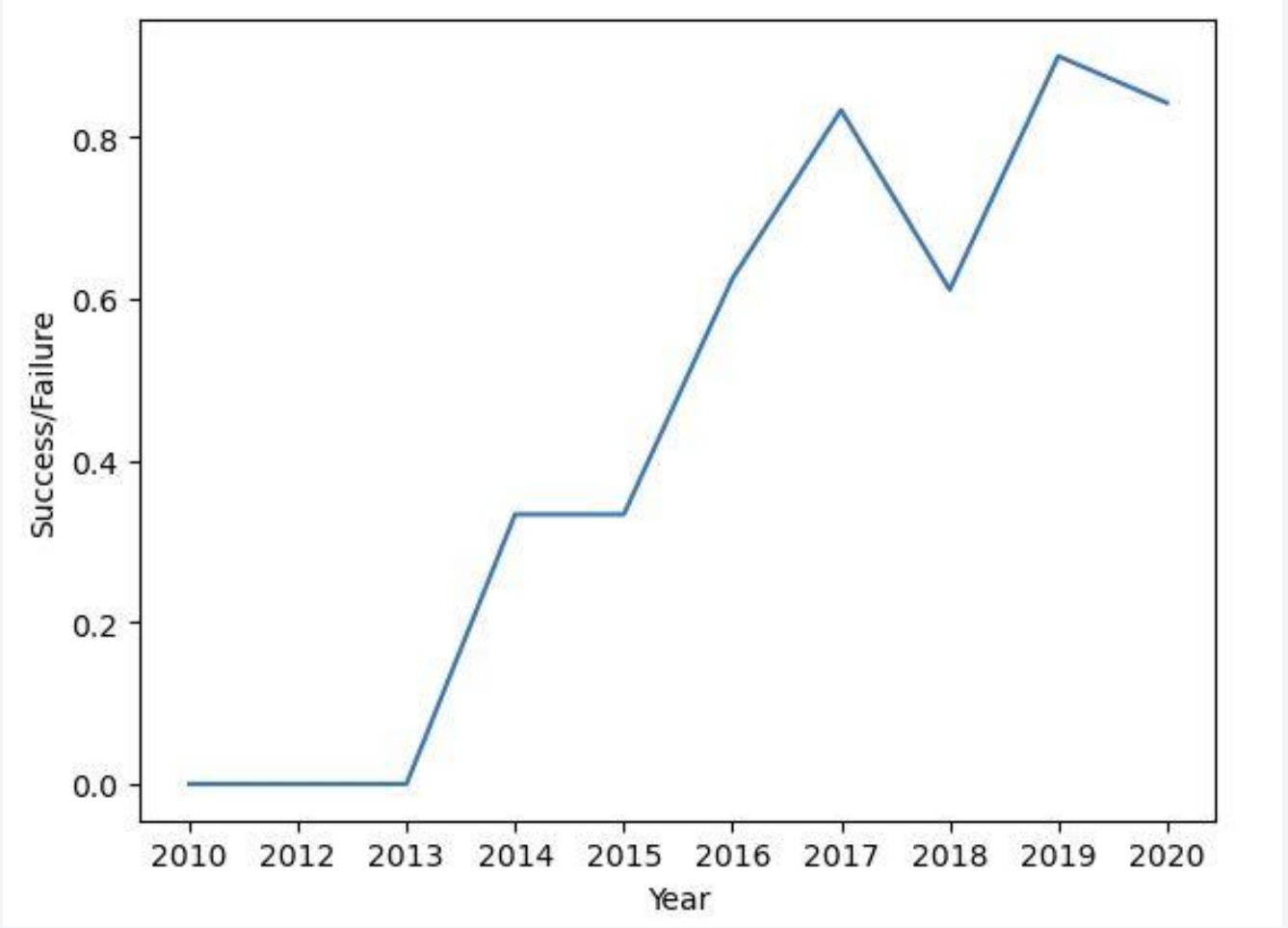


We observe that heavier Payloads correlate to better success with LEO and ISS Orbit Types but heavier Payloads correlate to a worse success rate with MEO and VLEO Orbit Types

Launch Success Yearly Trend

Git Hub Link

We can see clear improvement in success rate as the years go on.



All Launch Site Names

Git Hub Link

We used SQL to grab the unique names of each Launch Site from the SPACE X Table. We used the function **distinct** to only show each value once.

Display the names of the unique launch sites in the space mission

```
%sql select distinct(LAUNCH_SITE) from SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Launch Site Names Begin with 'CCA'

Git Hub Link

We used SQL to list the first 5 Launch Sites in the table that start with the characters "CCA"

The **limit** 5 function limits the results to 5.

The **Like** keyword filters the result to matches and the % at the end of CCA infers that there can be any characters after CCA.

Like CCA% is equivalent to "Starts with CCA"

Display 5 records where launch sites begin with the string 'CCA'

```
%sql SELECT LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) like 'CCA%' limit 5;
```

```
 * sqlite:///my_data1.db
Done.
```

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

Git Hub Link

Display the total payload mass carried by boosters launched by NASA (CRS)

```
%sql SELECT sum(PAYLOAD_MASS__KG_) as 'Total PayLoadmass by NASA' from SPACEXTBL where CUSTOMER ='NASA (CRS)';
```

```
 * sqlite:///my_data1.db
Done.
```

Total PayLoadmass by NASA
45596

We uses SQL to calculate the total payload mass carried by boosters launched by NASA.

The **Sum** function is used to add the total of all the columns (payload_mass__kg_) while using the **WHERE** keyword to only include columns where the customers is also "Nasa (CRS)"

Average Payload Mass by F9 v1.1

Git Hub Link

Display average payload mass carried by booster version F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1';
```

```
 * sqlite:///my_data1.db
Done.
```

payloadmass
2928.4

We used SQL to calculate the average payload mass carried by the F9 v1.1 boosters.

The **avg** function takes the average of all the values from the column (PAYLOAD_MASS__KG_), and used the **where** keyword to only use entries that also have the column (BOOSTER_VERSION) with the exact entry of 'F9 v1.1'

First Successful Ground Landing Date

Git Hub Link

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
%sql SELECT min(Date) from SPACEXTBL where [Landing _Outcome] ='Success (ground pad)';
```

```
 * sqlite:///my_data1.db
Done.
```

min(Date)
01-05-2017

Here we used SQL to show when the first successful Ground Pad landing occurred.

We used the **MIN** function to select the earlist successful landing and the **WHERE** keyword to only use entries that have the Landing_Outcome column with a value of 'Success (ground pad)'

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql select BOOSTER_VERSION from SPACEXTBL where [LANDING _OUTCOME]='Success (drone ship)' \
and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000;
```

```
 * sqlite:///my_data1.db
Done.
```

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

We used SQL to create a list of Booster Versions that have succeeded in landing on a Drone Ship with payloads between 4000kg and 6000kg.

First we **Selected** BOOSTER_VERSION from the SPACEX Table and then we used the **WHERE** keyword to limit the results to only successful landings to a Drone Ship and with Payloads greater than 4000kg but less than 6000kg

Git Hub Link

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%sql Select(select count(MISSION_OUTCOME) from SPACEXTBL where Mission_Outcome like '%Success%')as Successful, \
(select count(MISSION_OUTCOME) from SPACEXTBL where Mission_Outcome like '%Failure%')as Failure ;
```

```
 * sqlite:///my_data1.db
Done.
```

Successful	Failure
100	1

We used SQL to tabulate and list the successful and unsuccessful launch attempts by Space X.

First we used the **Count** function to enumerate the Mission Outcome column, while using the **Like** keyword to organize them into Success or Failure output column.

Git Hub Link

Boosters Carried Maximum Payload

Git Hub Link

```
%sql select distinct BOOSTER_VERSION, max(PAYLOAD_MASS__KG_) as [Maximum Payload Mass]
from SPACEXTBL group by Booster_Version order by [Maximum Payload Mass] desc;
```

```
 * sqlite:///my_data1.db
Done.
```

Booster_Version	Maximum Payload Mass
F9 B5 B1060.3	15600
F9 B5 B1060.2	15600
.....	
F9 FT B1038.1	475
F9 B4 B1045.1	362
F9 v1.0 B0004	0
F9 v1.0 B0003	0

Here we used SQL to create an ordered list showing the maximum payload of each of their boosters.

We used the **Distinct** function to only list one occurrence of each booster with the **Max** keyword to choose the occurrence with the highest payload, then we used the **Order** by and **Desc** keywords to list them from largest to smallest.

2015 Launch Records

Git Hub Link

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

```
%sql SELECT substr(Date, 4, 2) as month, Booster_Version, Launch_Site from \
SPACEXTBL where substr(Date,7,4)='2015' and [Landing _Outcome] = 'Failure (drone ship)';
```

```
 * sqlite:///my_data1.db
Done.
```

month	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

Here we used SQL to list the Boosters that failed to land on Drone Ships in the Year 2015.

We use **SUBSTR** to extract a portion of a string by specifying the starting position and the number of characters to be extracted (*Column, start position, length of characters*) and then use the **WHERE** keyword with **SUBSTR** to determine which entries to use, ie the ones that have 2015 and if their Landing_Outcome is a failure on a Drone Ship.

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

We used SQL to count and list the total number of each Success and Failure for each landing class.

We first **Select** Landing Outcome type as a column then in a separate column we get the total **Count** of each occurrence. We use the SPACEXTBL for reference and then use the **WHERE** Keyword to limit the results between 04-06-2010 and 20-03-2017. We then **ORDER** them by Landing Outcome total from highest to lowest with **DESC**.

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT [Landing _Outcome] as 'Landing Outcome', count([Landing _Outcome]) as 'Total Count' \
From SPACEXTBL Where Date between '04-06-2010' and '20-03-2017' \
group by [Landing _Outcome] order by count([Landing _Outcome]) desc;
```

```
 * sqlite:///my_data1.db
Done.
```

Landing Outcome	Total Count
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

Git Hub Link


Section 3
Launch Sites
Proximities Analysis

Launch Sites Displayed on Folium Map

Git Hub Link

He we have all the Launch Sites listed, they are located on the East Coast of the Florida Coastline and the West Coast of the Southern California Coast Line.

Color Coded Launch Outcomes

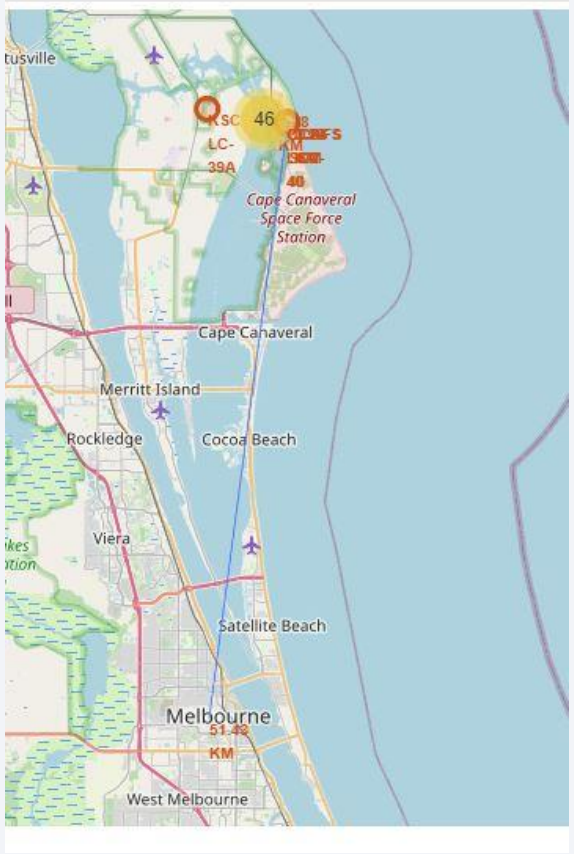
Git Hub Link

We can use Folium to further label and visualize data. Here we were able to mark each individual launch with a color coded label. Green is for a successful mission and Red is for a failure.

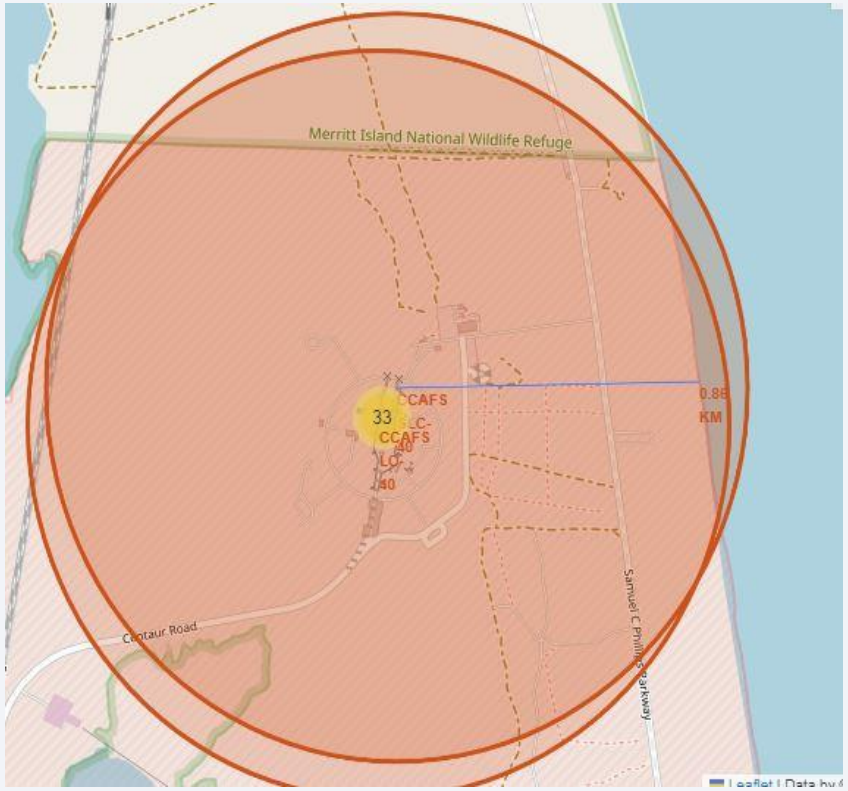
By interacting with the map you can expand each launch site to view the results.

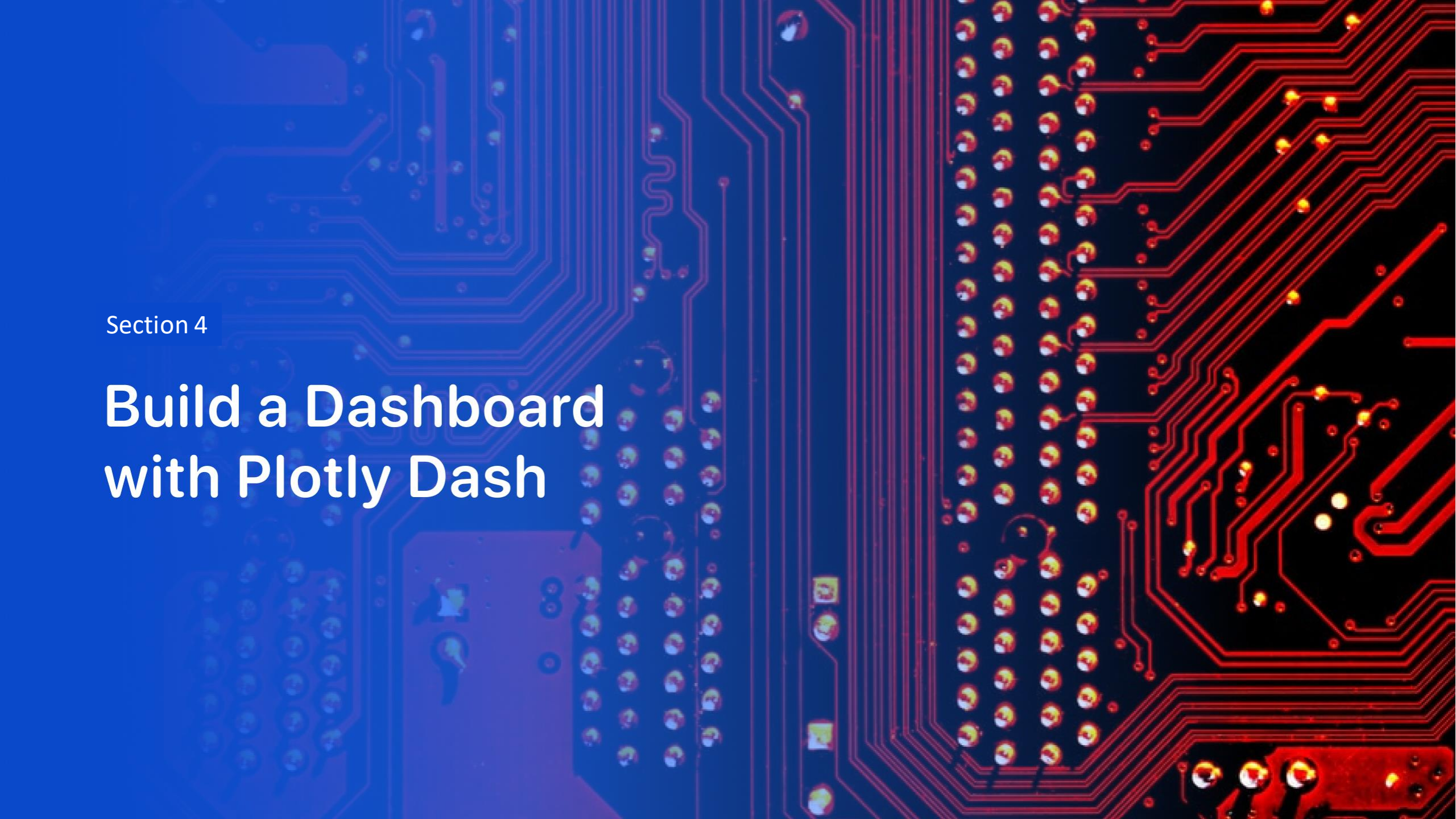
Launch Sites proximity to Land Marks

Git Hub Link



Here we used Line markers to illustrate and display the distance between the launch sites and the Coast Line as well as the nearest large city. We can also use this feature to show relative distance to land marks such as Roads, Railways, Highways or even other launch sites.

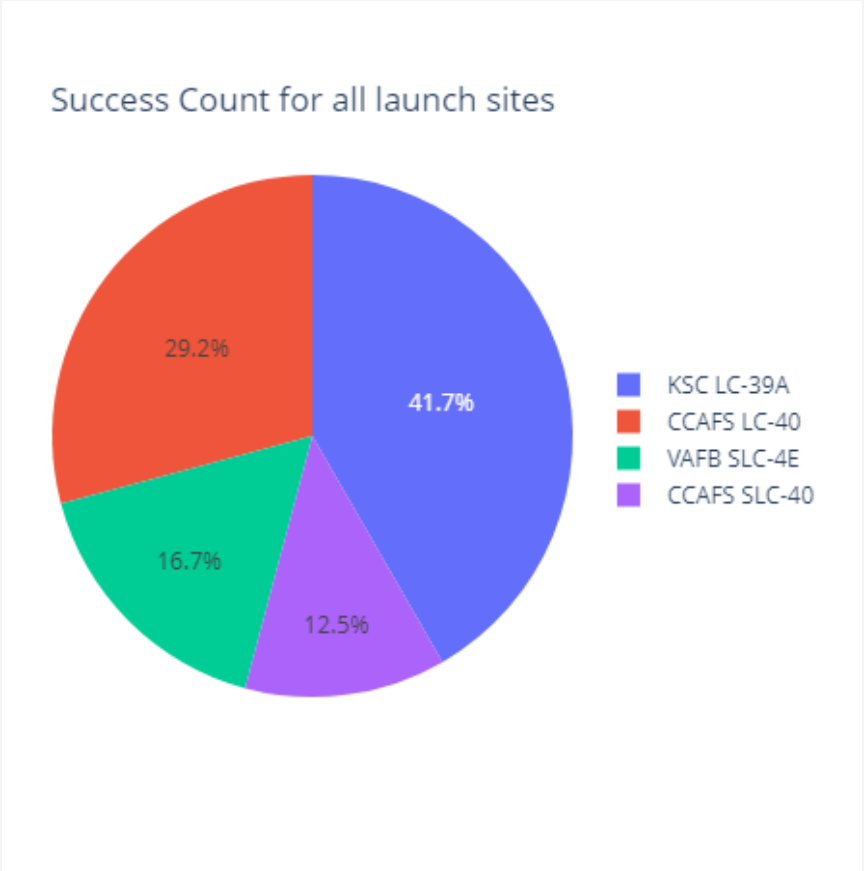



Section 4
Build a Dashboard with Plotly Dash

Launch Success for All Sites

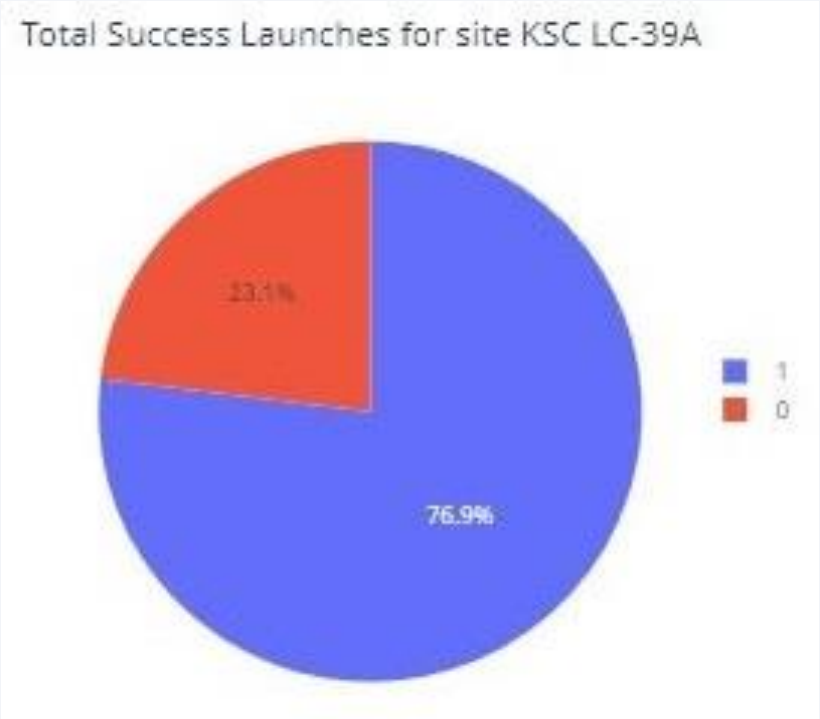
Git Hub Link

Plotly gives us easy to digest visualizations. Here we can see that the site "KSC LC-39A" has had the most successful number of launches compared to all the other sites.



Launch Success for KSC LC 39A Launch Site

Here we have focused on just the most successful site, KSC LC 39A with a 76.9% success rate.



Git Hub Link

PayLoad VS. Launch Outcome, All Sites

Git Hub Link

Here we can see that the success rate is better for the lightweight launches than the heavier ones. Also, a point to note, there were far more launches made with loads lighter than 6k, perhaps the success rate for the lighter loads is in part due to practice

Section 5

Predictive Analysis (Classification)

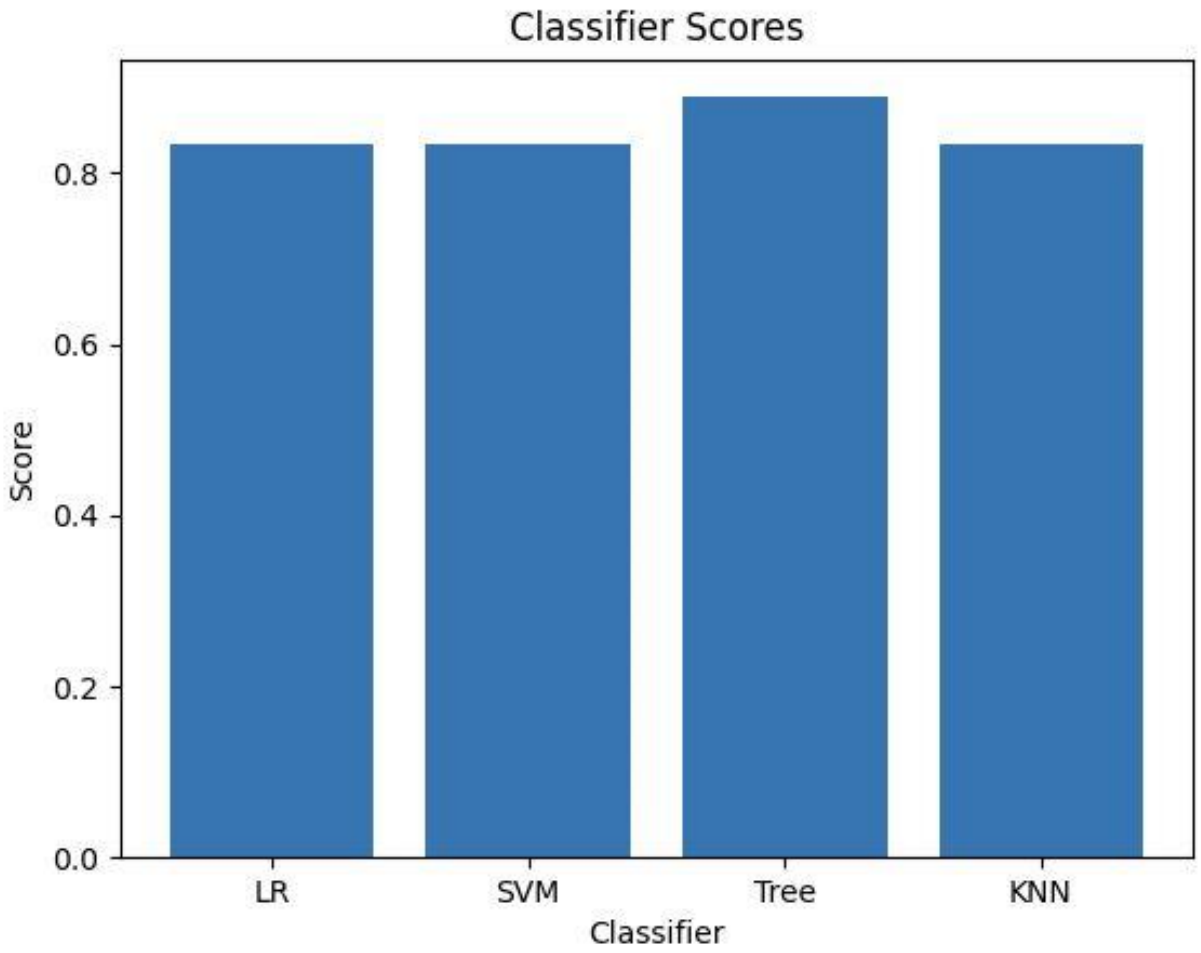
Classification Accuracy

Git Hub Link

We trained four separate models, using the same data and test conditions. All but 1 had the same accuracy but our Decision Tree model was slightly more accurate.

```
Test set Accuracy of LogReg 0.8333333333333334
Test set Accuracy of SVM    0.8333333333333334
Test set Accuracy of Tree   0.8888888888888888
Test set Accuracy of KNN    0.8333333333333334
```

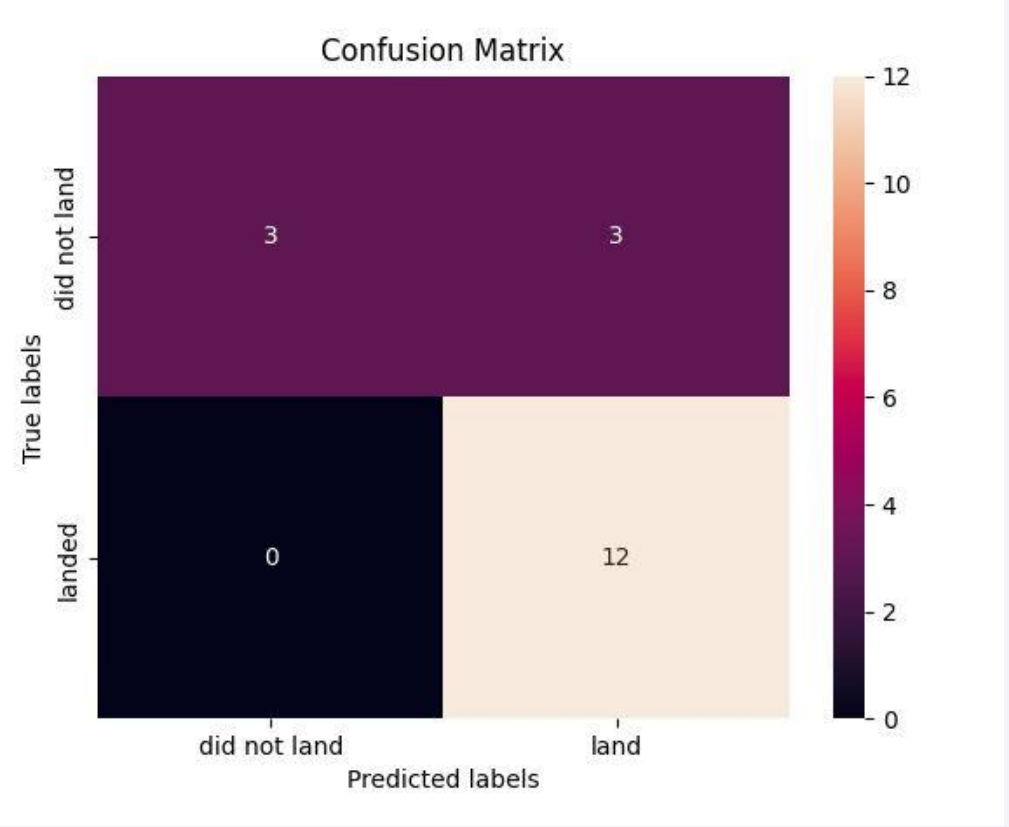
As seen here, the Decision Tree wins in accuracy



Confusion Matrix

Git Hub Link

Confusion matrices are a valuable tool in evaluating the performance of machine learning models. They provide a clear and visual representation of the model's accuracy and can be used to compute various performance metrics. Confusion matrices also allow for the analysis of the model's errors, which can provide insights into the strengths and weaknesses of the model and help in its improvement. In short, confusion matrices are a useful tool for understanding and improving the performance of machine learning models.



Conclusions

- Launches with Orbit Types (ES-L1, HEO, SSO and GEO) Have the highest success rate as of the time of this data window
- As time and experience continues, Space X has dramatically increased their success rates
- The most successful launch site has been KSC LC 39A
- Overall, Payload has a pretty big effect on the success of the mission, with the heavier the payload equating to a less likely hood of success
- For the data we have and the needs of our project, the Decision Tree Classifier is the most accurate model for our needs.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!