IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Daniel Gething>
<18 May 2025>

Outline

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

Executive Summary

- Summary of methodologies
  - Use of API data collection and Webscraping techniques to gather data from Space X website and Wikipedia.
  - Use of Python to merge, shape and explore the data. Data visualization with python used to explore trends within shaped data.
  - Use of SQL to explore further trends in the data.
  - Geospatial trends visualized through folium
  - Plotly Dash used to create a centralized dashboard for data exploration.
  - Machine Learning techniques used to determine the main predictor values
- Summary of all results
  - Results of Exploratory Data, shown through graphs
  - Created graphs shown in screenshots
  - Predictor values determined through machine learning techniques.

Introduction

- Project background and context
  - SpaceX's Falcon 9 rocket is advertised on its website for $62 million. Compared to other competitors, it provides a saving of $103 million. These savings are due to SpaceX's ability to reuse the first stage of the rocket launch. By predicting whether the first stage will land, we can predict the cost of future launches. Working for a competitor company, the goal of this project is to determine the key factors and the overall success rate of this first stage.

- Problems you want to find answers
  - What are the main factors contributing to successful landing?
  - How do these features interact to determine the success rate of a successful landing?
  - What conditions need to be in place to ensure the successful landing of the first stage?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected through a combination of API and Web-scraping techniques.
- Perform data wrangling
  - Data was processed in python, One-Hot Encoding was applied to all variables
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Python's Machine Learning techniques were used to create four working models. A comparison was run to determine which method was the best.

Data Collection

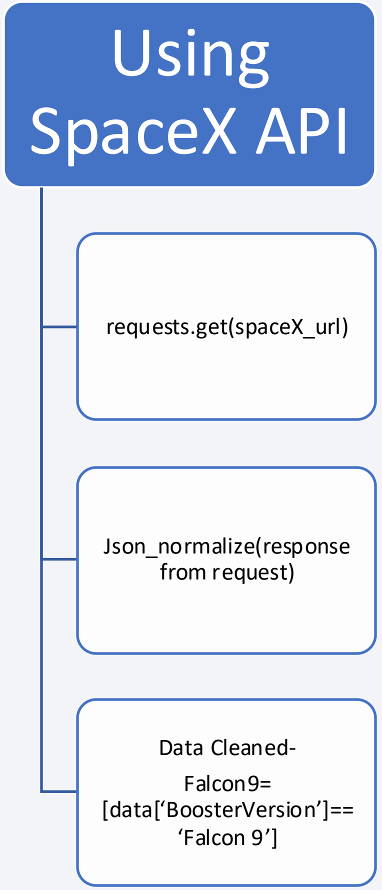
- API and Web-scraping data collection techniques.
  - The SpaceX API was utilized to collect the data
    - The was done through .json() functions which was converted into a Pandas dataframe through .json_normalize() function.
    - The data was cleaned; missing values were filled in through using the mean of continuous values.
  - Web-scraping techniques were used on the SpaceX Wikipedia page.
    - To parse the data from the webpage, Python's BeautifulSoup was used to convert tables to a Pandas dataframe for further cleaning and data analysis.

Data Collection – SpaceX API

- Used SpaceX API to collect data, then conducted basic formatting to select only the data we need for the project.

- The full script can be found here:

- https://github.com/Dako-codes/IBM-Data-Science-Capstone--SpaceXIBM-/blob/main/Space%20X%20Data%20Collection.ipynb



Using SpaceX API

requests.get(spaceX_url)

Json_normalize(response from request)

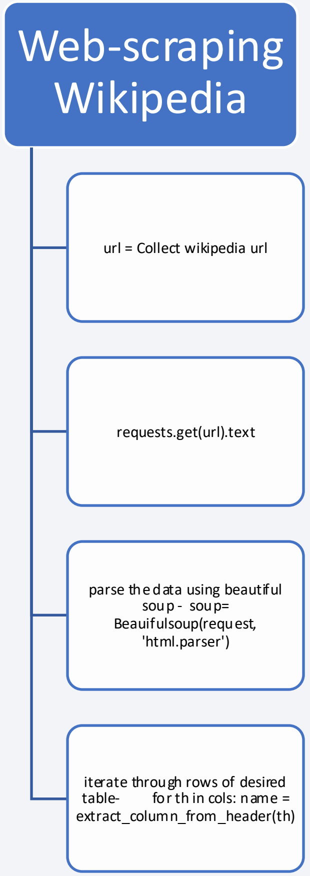
Data Cleaned-

Falcon9=
[data[‘BoosterVersion’]==
‘Falcon 9’]

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose



Web-scraping Wikipedia

url = Collect wikipedia url

requests.get(url).text

parse the data using beautiful soup - soup= Beauifulsoup(request, 'html.parser')

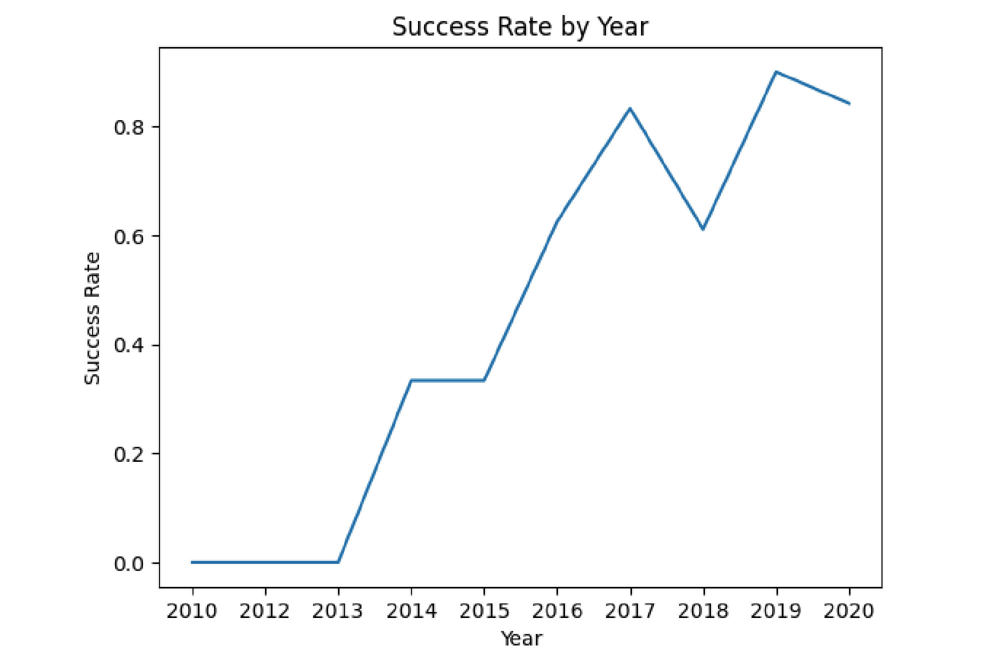
iterate through rows of desired table- for th in cols: name = extract_column_from_header(th)

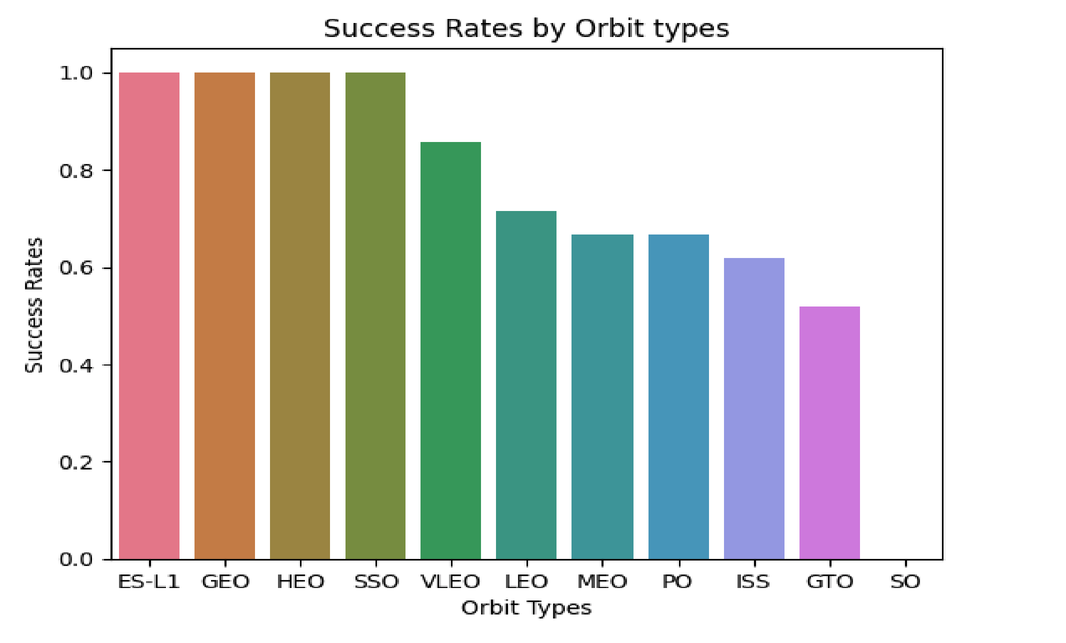
Data Wrangling

- The data was cleaned and processed using One-Hot encoding.
- Exploratory data analysis was conducted to find the training labels.
  - These labels were used to calculate the number of launches for each site, the number of launches by orbit type
- Finally, data was collected to determine whether a mission was successful or not
  - A binary variable called 'Class' was created to determine the outcome of each mission. This was converted to 1 and 0.
- The success rate of Falcon 9 booster rockets landing was determined to be 66.6% or around One Third.
- The full script can be seen at: https://github.com/Dako-codes/IBM-Data-Science-Capstone--SpaceXIBM-/blob/main/Space%20X%20Data%20Wrangling.ipynb

EDA with Data Visualization

- The graphs highlight the trend mentioned above. We can understand that certain orbits have higher success rates than others. And generally (despite a brief drop in 2018) success rates have increased over time, peaking in 2019.





The Script used and other graphs can be seen at: https://github.com/Dako-codes/IBM-Data-Science-Capstone--SpaceXIBM-/blob/main/Exploratory%20Data%20Visualisation.ipynb

EDA with SQL

- In SQL we used a range of non-visual exploratory techniques
- We determined the four launch sites used for the Falcon 9 Booster Rocket.
- We determined the total Payload Mass carried was 45596KG.
- Compared to this, the mean average Payload Mass was 2534.67KG.
- Through the Min(Date) function, we determined that the first successful ground pad landing was in 2018.
- FT-style Boosters are typically used with drone-ship landing pads
- There is an overwhelming mission success rate listed in the SQL table
- 44 Booster Versions are capable of carrying the max payload weight of 15600KG
- In 2015 there were failures to land in January and April
- Finally, we listed the number of each type of Landing_Outcome between 2010 and 2017. Determining that there were 8 successes, 7 failures and 10 launches with no attempts to land.
- The scripts and further details can be seen at: https://github.com/Dako-codes/IBM-Data-Science-Capstone--SpaceXIBM-/blob/main/Exploratory%20Data%20Analysis%20with%20SQL.ipynb

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
- Explain why you added those objects
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build a Dashboard with Plotly Dash

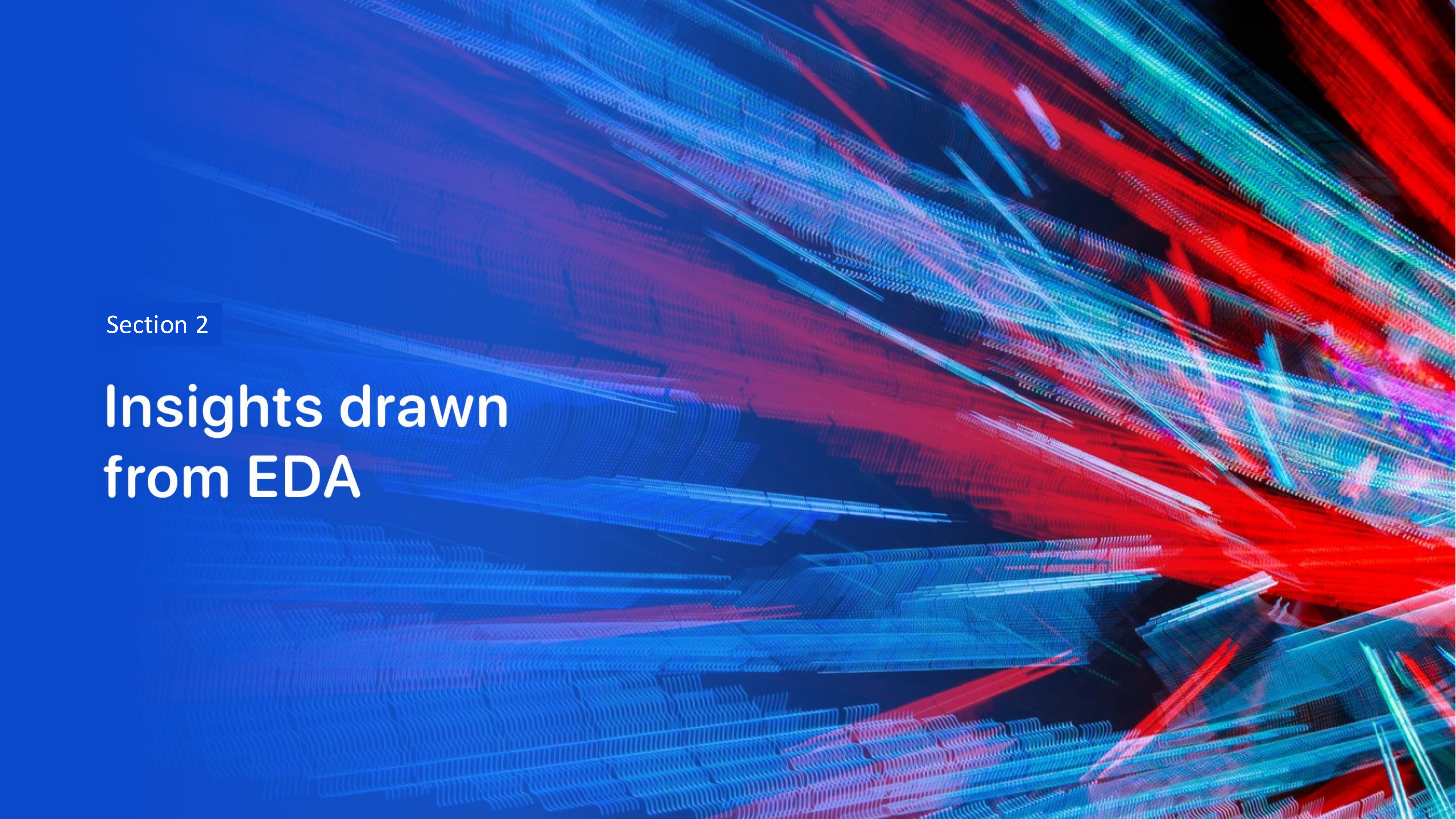
- Summarize what plots/graphs and interactions you have added to a dashboard
- Explain why you added those plots and interactions
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

Predictive Analysis (Classification)

- Using Scikit-Learn, numpy and pandas. We built four machine learning models to predict the outcome of each mission.
- This was done through a test-train split, using 20% of the data to test each models, as well as a GridSearchCV to determine and tune hyperparameters.
- Accuracy was used to determine the best model and to find the best parameters for each model.
- The best performing classification model was determined to be the Decision Tree model
- The script and further details can be found at: https://github.com/Dako-codes/IBM-Data-Science-Capstone--SpaceXIBM-/blob/main/Space%20X%20Landing%20Prediction%20with%20Machine%20Learning.ipynb

Results

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

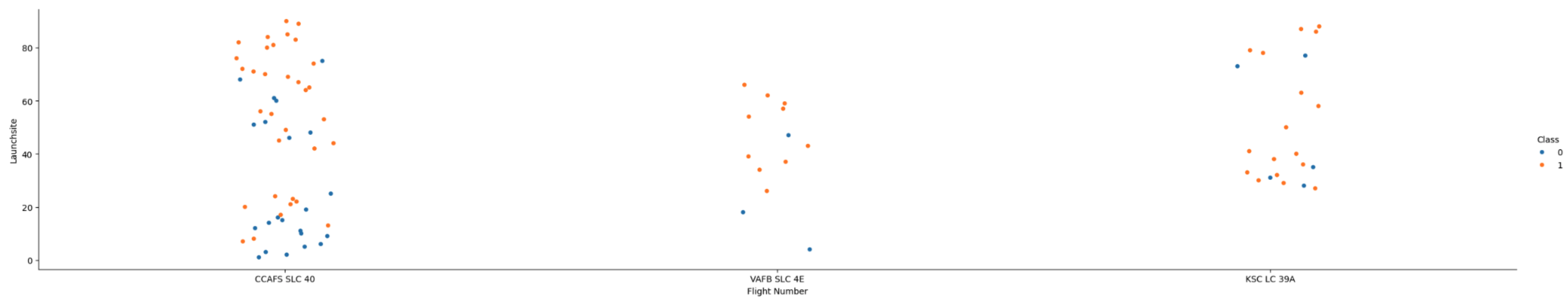


Section 2

Insights drawn from EDA

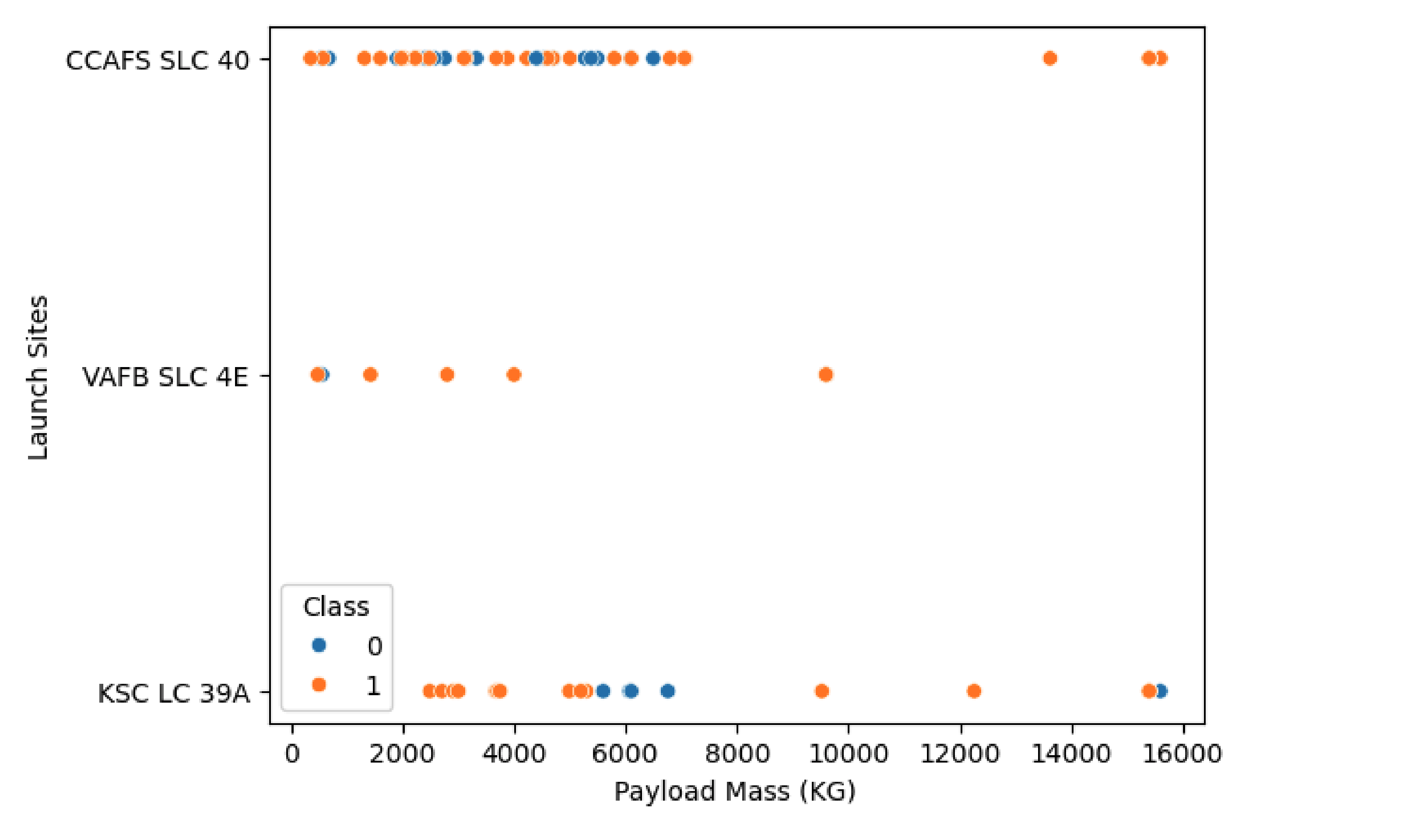
Flight Number vs. Launch Site

- We can see that the launch site CCAFS SLC 40 was used the most, while launch site VAFB SLC 4E was used the least.



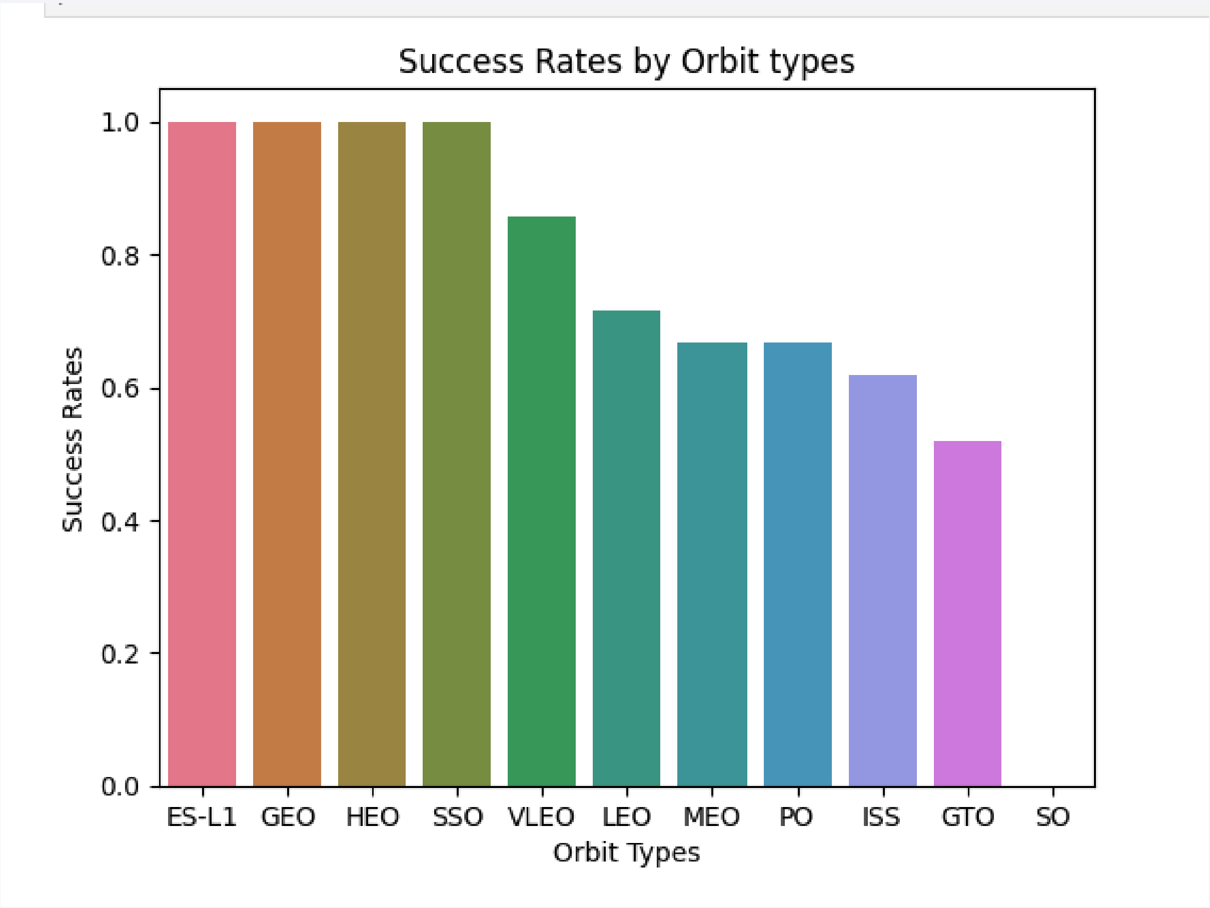
Payload vs. Launch Site

- This data shows us that there is a general trend towards lower weight Payload masses.
- Generally, each Launch Site launches similar amounts weights of Payloads.



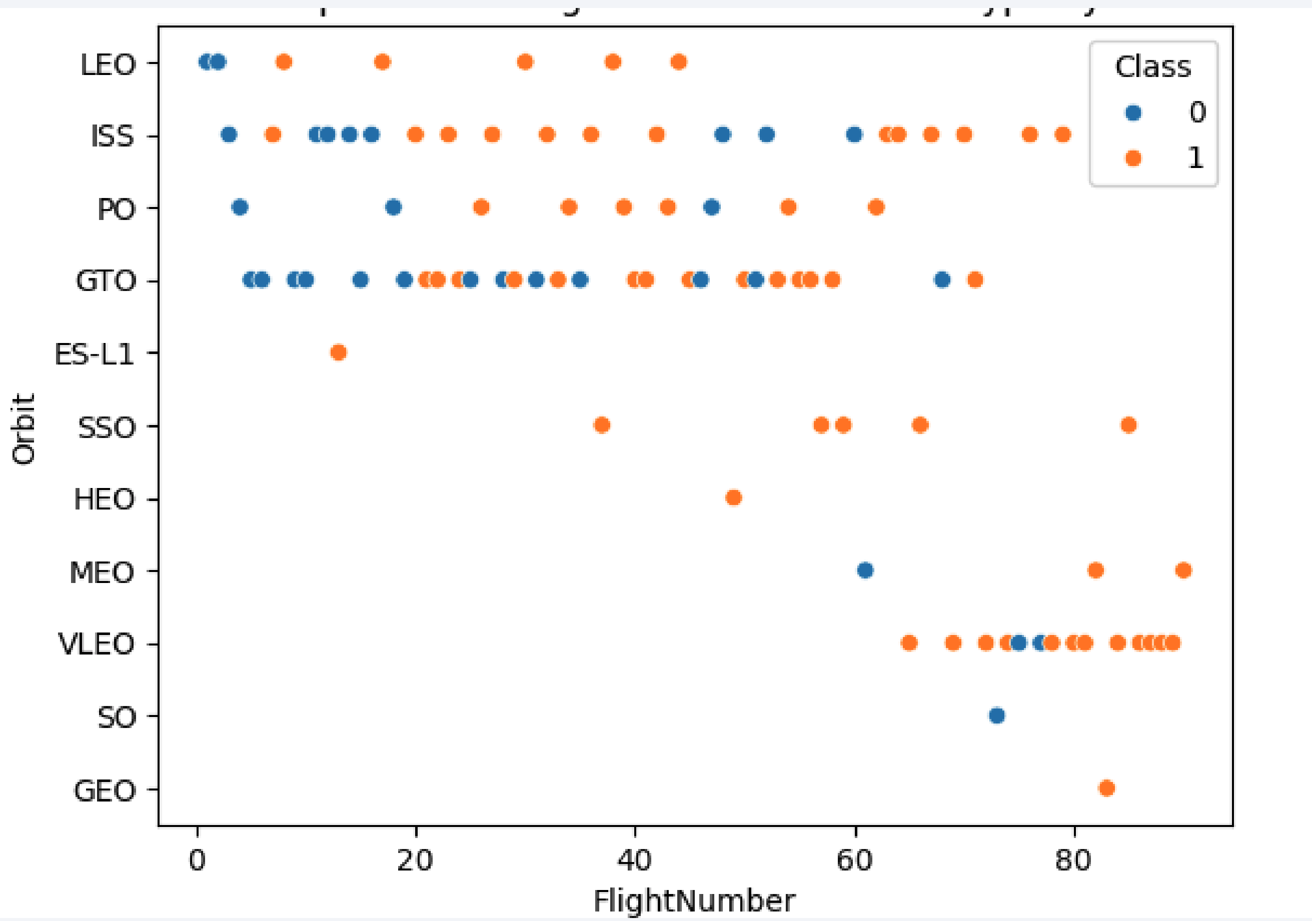
Success Rate vs. Orbit Type

- We can see that four orbit types have a perfect success rate. While one has a 0% success rate. It is unclear from the graph whether this is because of lack of data and further testing should be conducted on this.



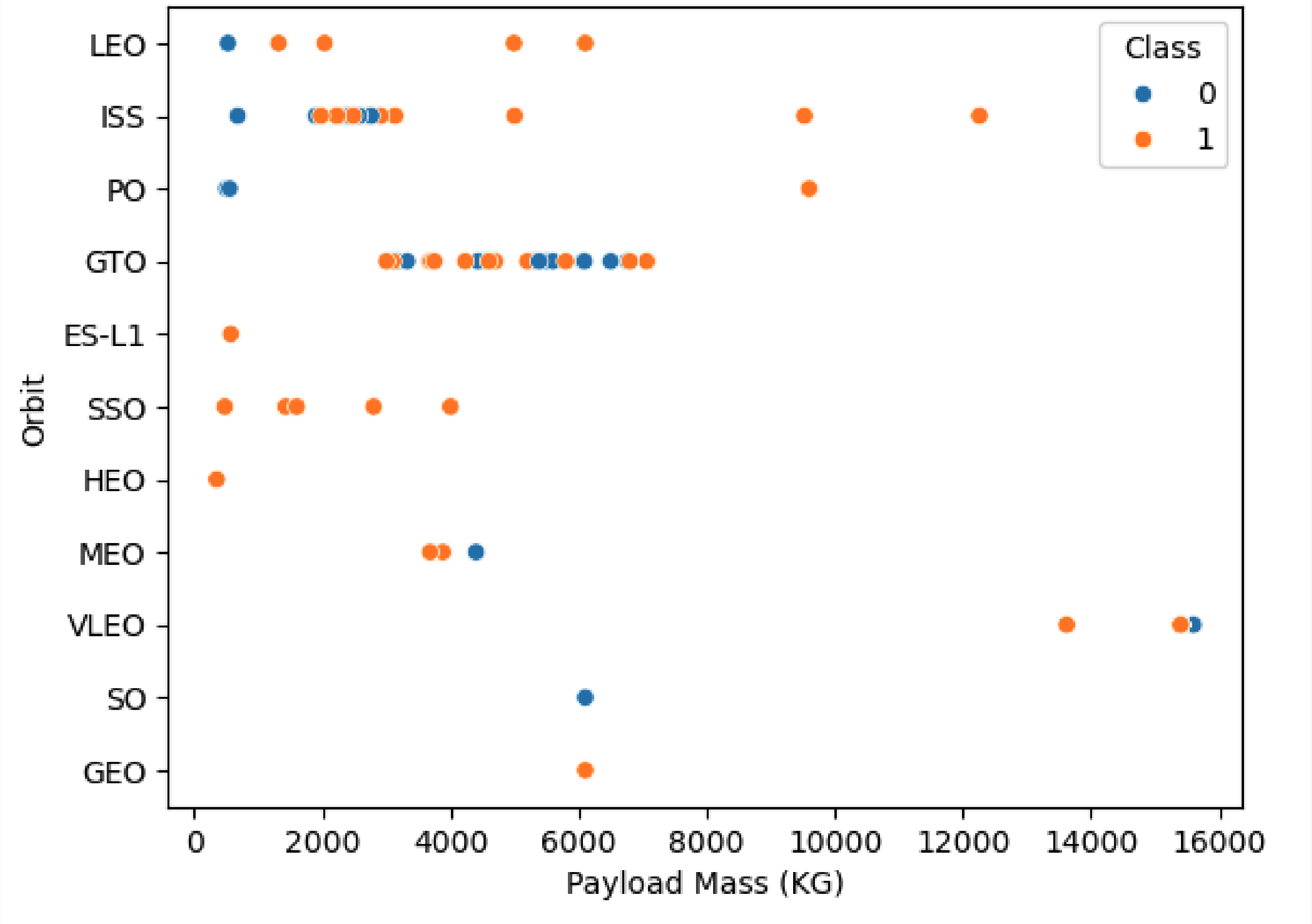
Flight Number vs. Orbit Type

- This graph shows that generally flight numbers are grouped by orbit type. There are various reasons for this, but we can presume that overtime certain orbit types became more desirable and therefore later had more attempts. While some (such as ISS) are always necessary.



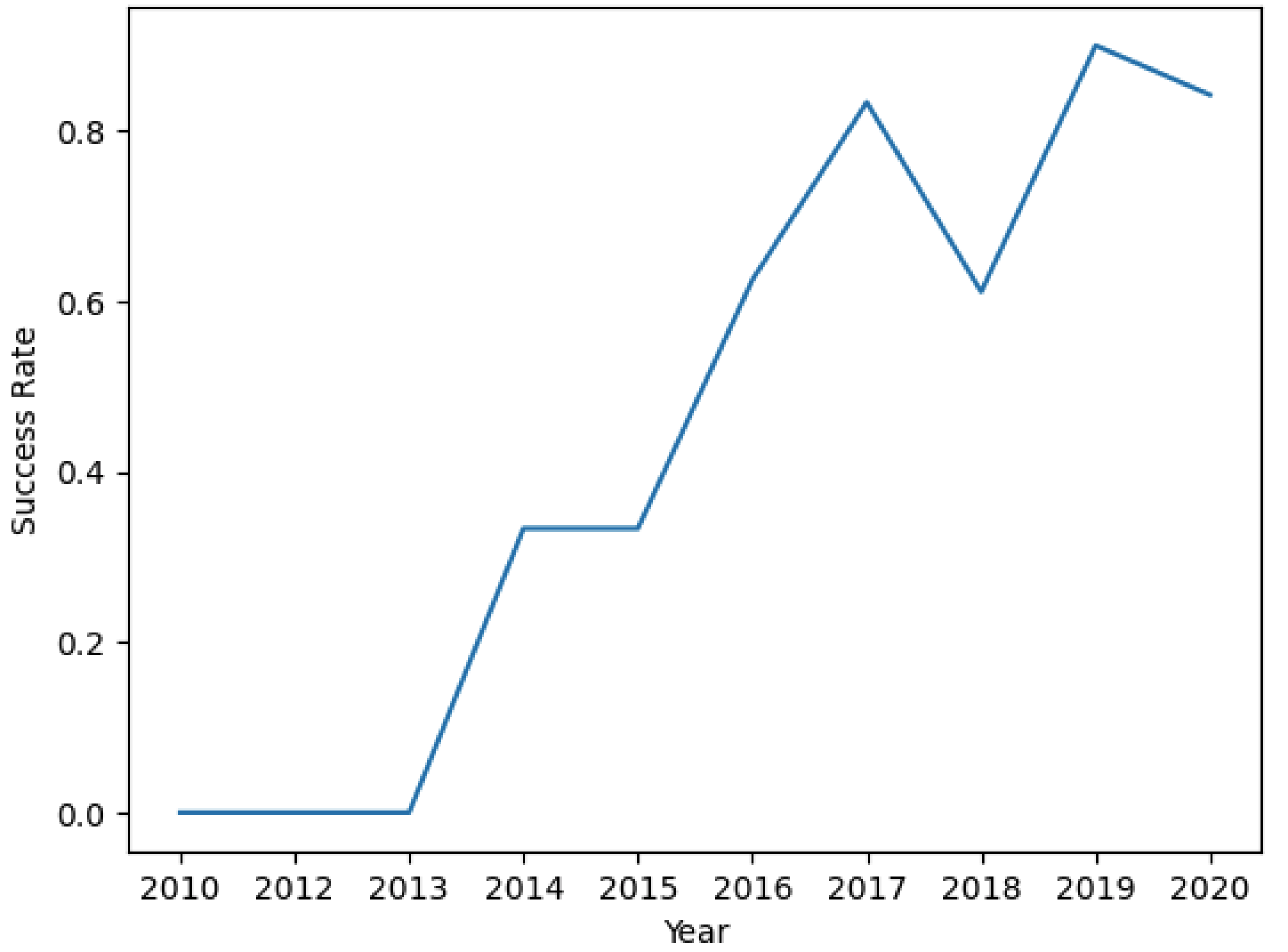
Payload vs. Orbit Type

- This graph shows that payload mass and Orbit types are largely correlated. Certain Orbit types have an optimal payload mass range.



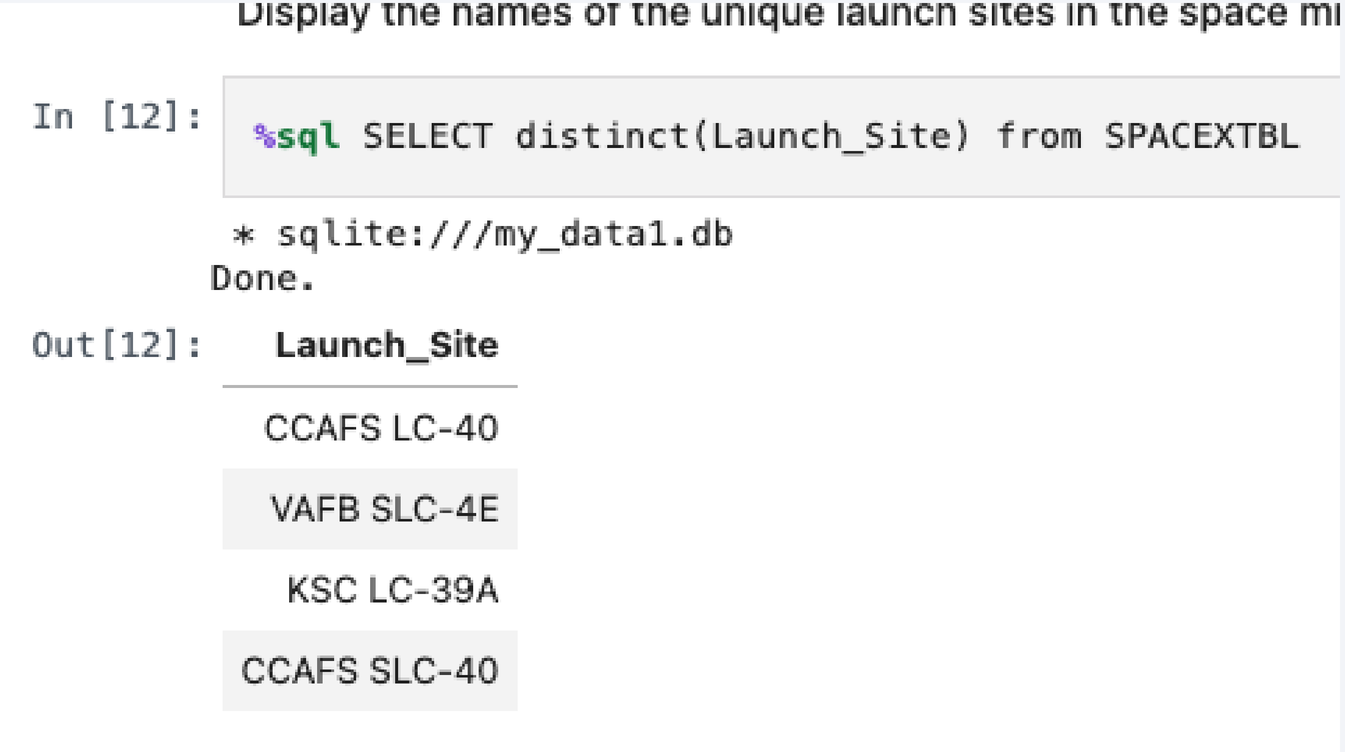
Launch Success Yearly Trend

- This graph shows a gradual increase of success rate over the decade between 2010 and 2020.
- There is a decrease in success rate compared to previous years in both 2018 and 2020 which indicates that processes at SpaceX have changed with regards to the Falcon 9 Booster.



All Launch Site Names

- Here we see a SQL query to find the distinct Launch Site names.
- This is done through selecting all the distinct types within the SQL table SPACEXTBL from the column Launch_Site

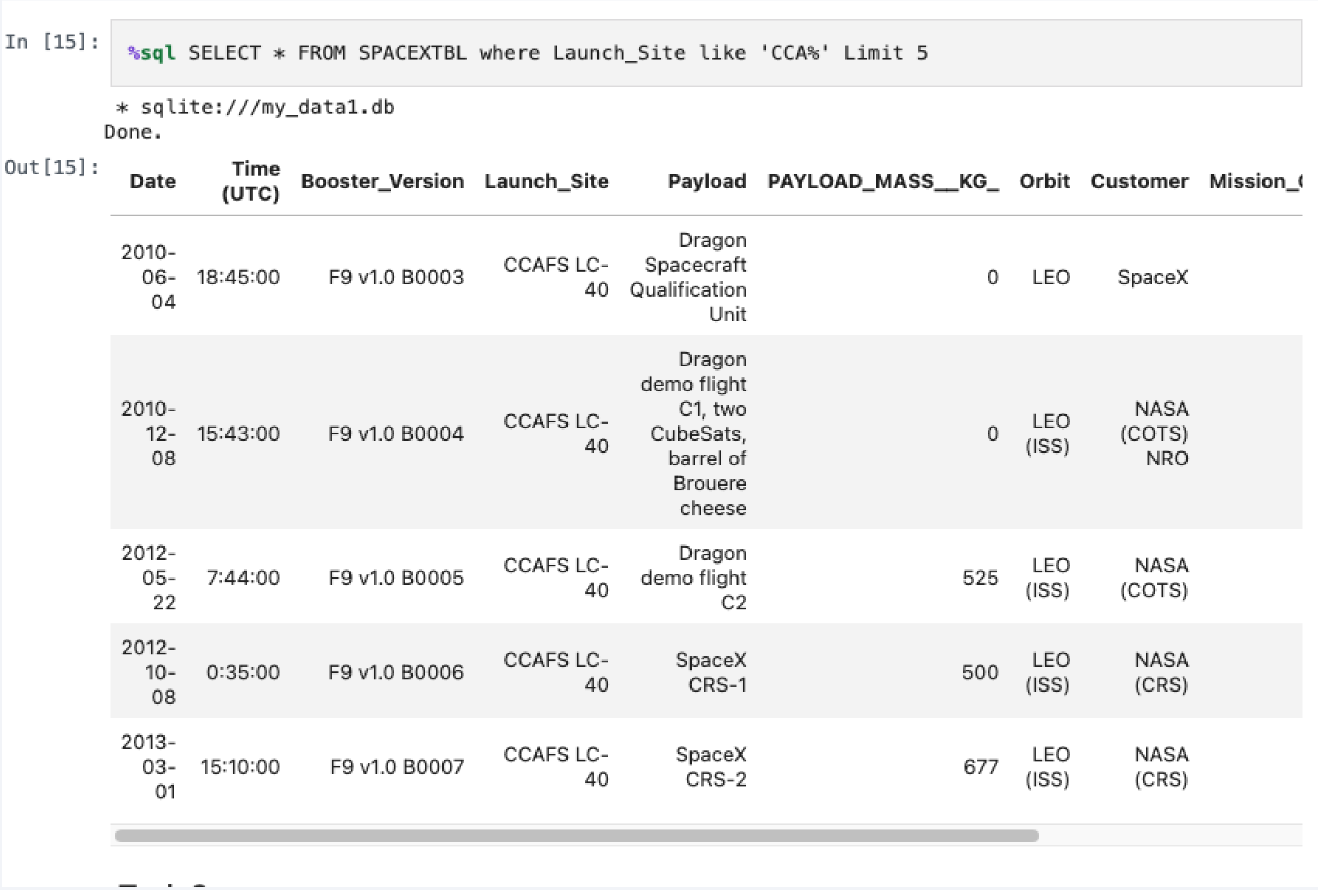


Display the names of the unique launch sites in the space mi

```
In [12]:  %sql SELECT distinct(Launch_Site) from SPACEXTBL

           * sqlite:///my_data1.db
          Done.

Out[12]:     Launch_Site
            CCAFS LC-40
            VAFB SLC-4E
            KSC LC-39A
           CCAFS SLC-40
```

Launch Site Names Begin with 'CCA'

- Here is a SQL query showing the first five results for launches beginning with CCA.
- This query by using the wildcard character % allows us to see values for both CCAFS LC-40 and CCAFS SLC-40. We've limited it to 5 to get a general idea of how the data would look.



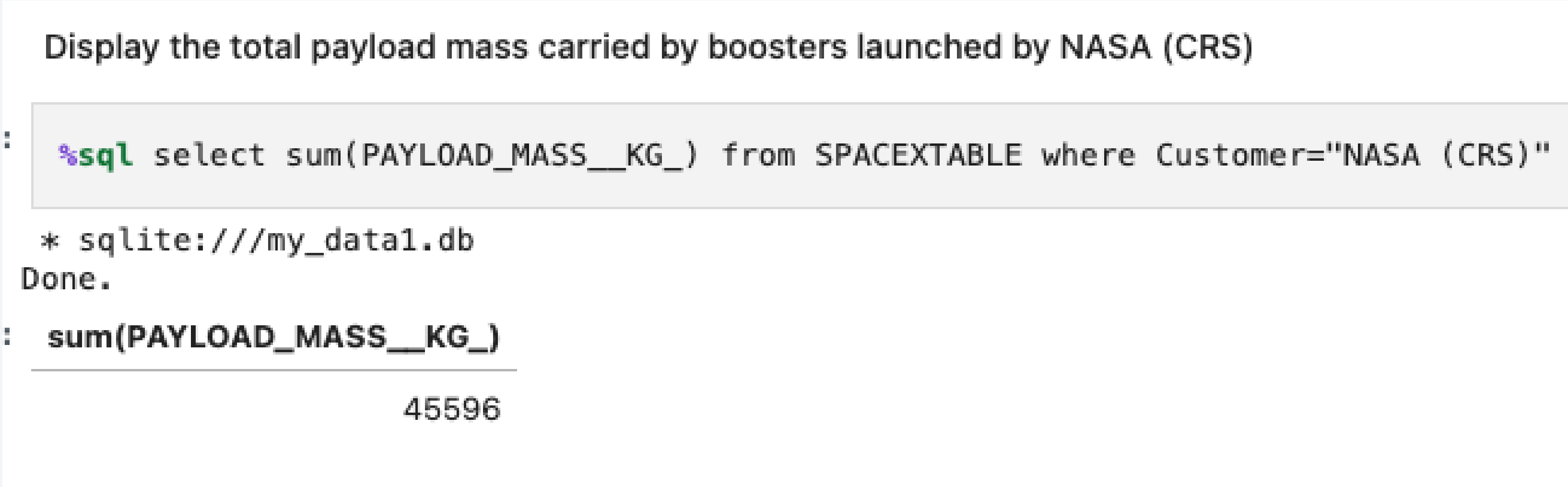
```
In [15]: %sql SELECT * FROM SPACEXTBL where Launch_Site like 'CCA%' Limit 5

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

Out[15]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_(
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	

Total Payload Mass

- This is a SQL query displaying the total of Payload Mass used by NASA (CRS)
- The Sum command allows us to total all the values in the desired column, while the where query allows us to specify which Customer we want to see- in this case NASA (CRS)



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

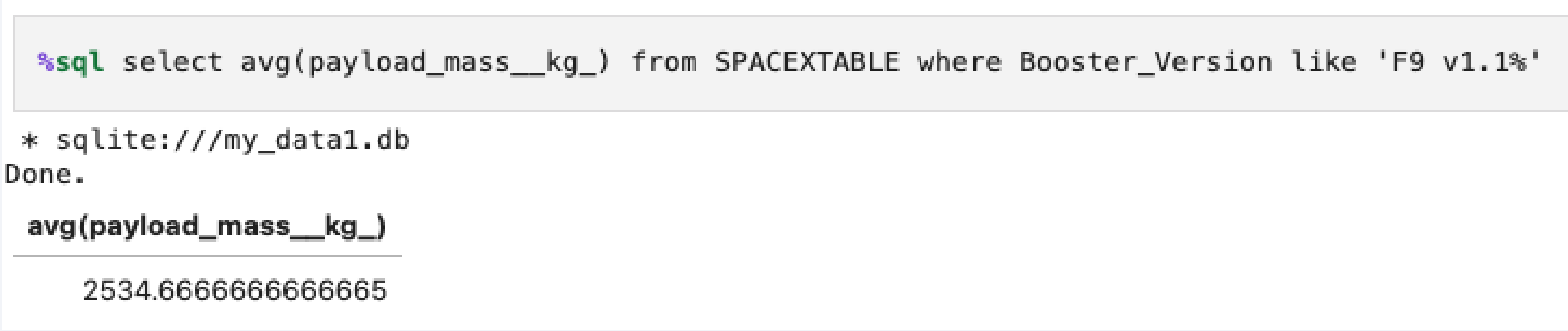
```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer="NASA (CRS)"
```

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

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- This is a SQL query displaying the average Payload mass for the F9 v1.1 booster.
- The avg query allows us to find the mean average of a specified column; in this case the payload mass.



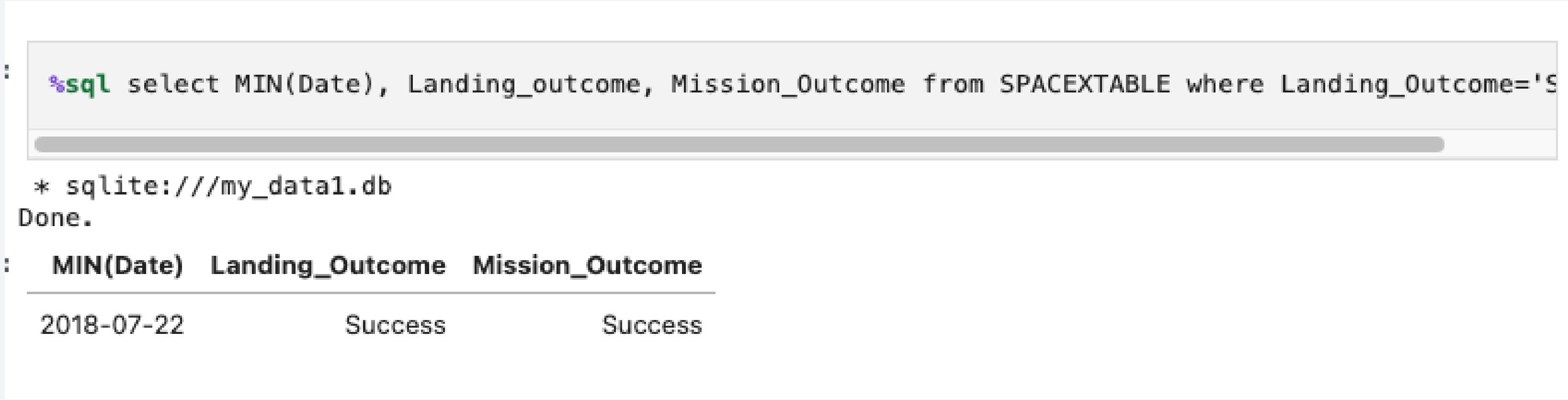
```
%sql select avg(payload_mass__kg_) from SPACEXTABLE where Booster_Version like 'F9 v1.1%'
```

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

avg(payload_mass__kg_)
2534.6666666666665

First Successful Ground Landing Date

- This shows a SQL query to find the date of the earliest successful Ground Landing
- The Min command allows us to find the smallest registered value for the specified column, here the date, while the where clause allows us to specify whether the outcome was a success or not.



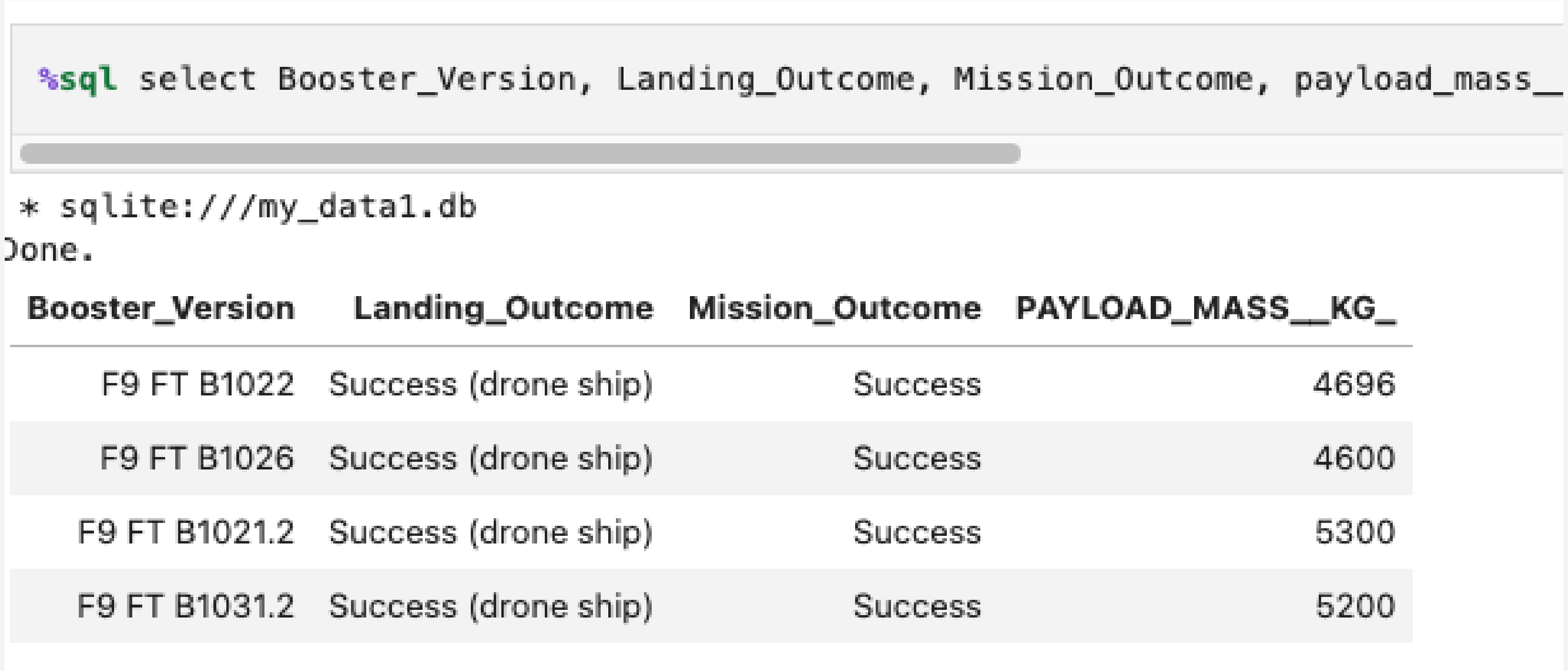
```
%sql select MIN(Date), Landing_outcome, Mission_Outcome from SPACEXTABLE where Landing_Outcome='S
```

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

MIN(Date)	Landing_Outcome	Mission_Outcome
2018-07-22	Success	Success

Successful Drone Ship Landing with Payload between 4000 and 6000

- This is a SQL request finding the values of payloads between 4000 and 6000. Four observations were found
- The between function allows us to specify the range within which the desired observations lie.



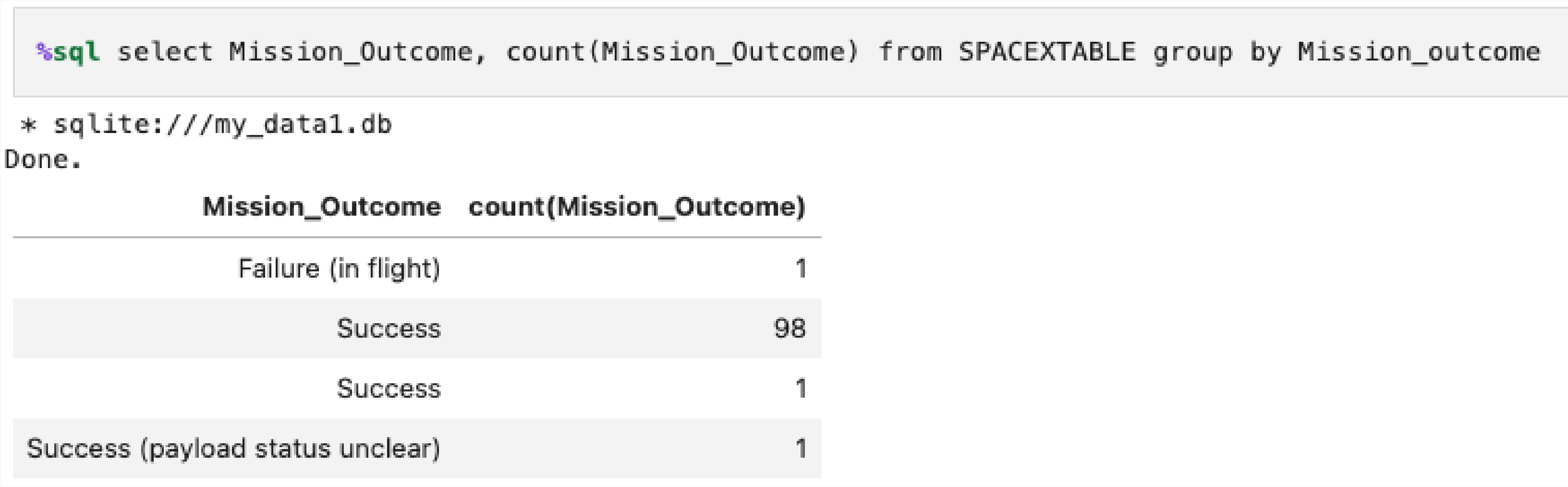
```
%sql select Booster_Version, Landing_Outcome, Mission_Outcome, payload_mass__
```

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

Booster_Version	Landing_Outcome	Mission_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	Success	4696
F9 FT B1026	Success (drone ship)	Success	4600
F9 FT B1021.2	Success (drone ship)	Success	5300
F9 FT B1031.2	Success (drone ship)	Success	5200

Total Number of Successful and Failure Mission Outcomes

- This SQL query displays the total number of each distinct Mission Outcome
- The count function displays the total number of each distinct mission outcome, while the group by clause allows us to specify how it will count the outcomes.



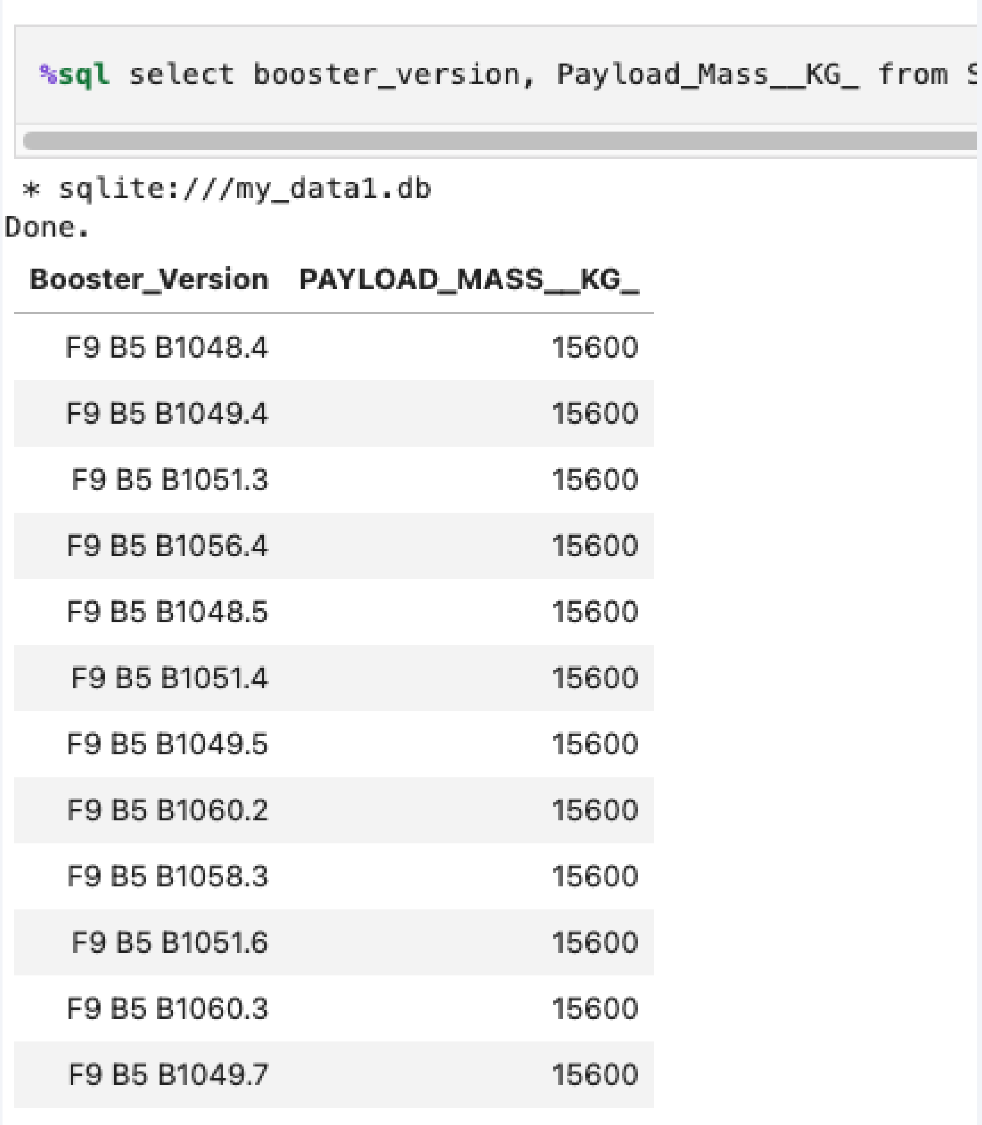
```
%sql select Mission_Outcome, count(Mission_Outcome) from SPACEXTABLE group by Mission_outcome
```

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

Mission_Outcome	count(Mission_Outcome)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- This SQL query displays which boosters carry the largest payload mass registered in the SQL table.
- In this SQL query we use a subquery with the max function to find desired columns which match that specific selection.



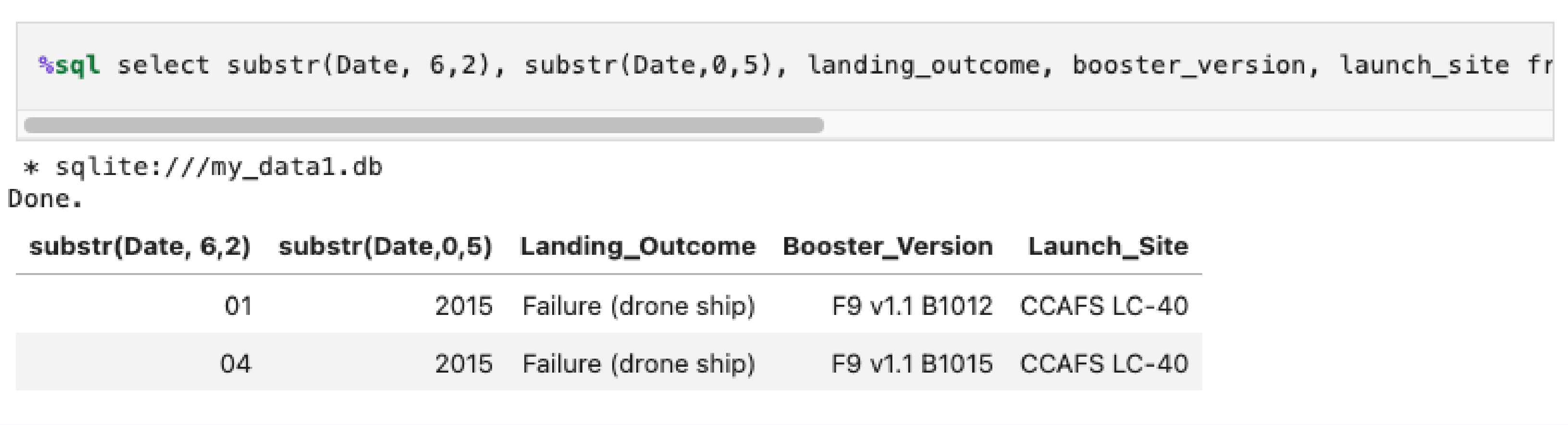
```
%sql select booster_version, Payload_Mass__KG_ from S
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

- This SQL query shows the month and year for failed Landing Outcomes in 2015.
- The substr command allows us to find specific sections of the date which was entered in a yyyy-mm-dd format. This same command is used to specify the desired year of 2015



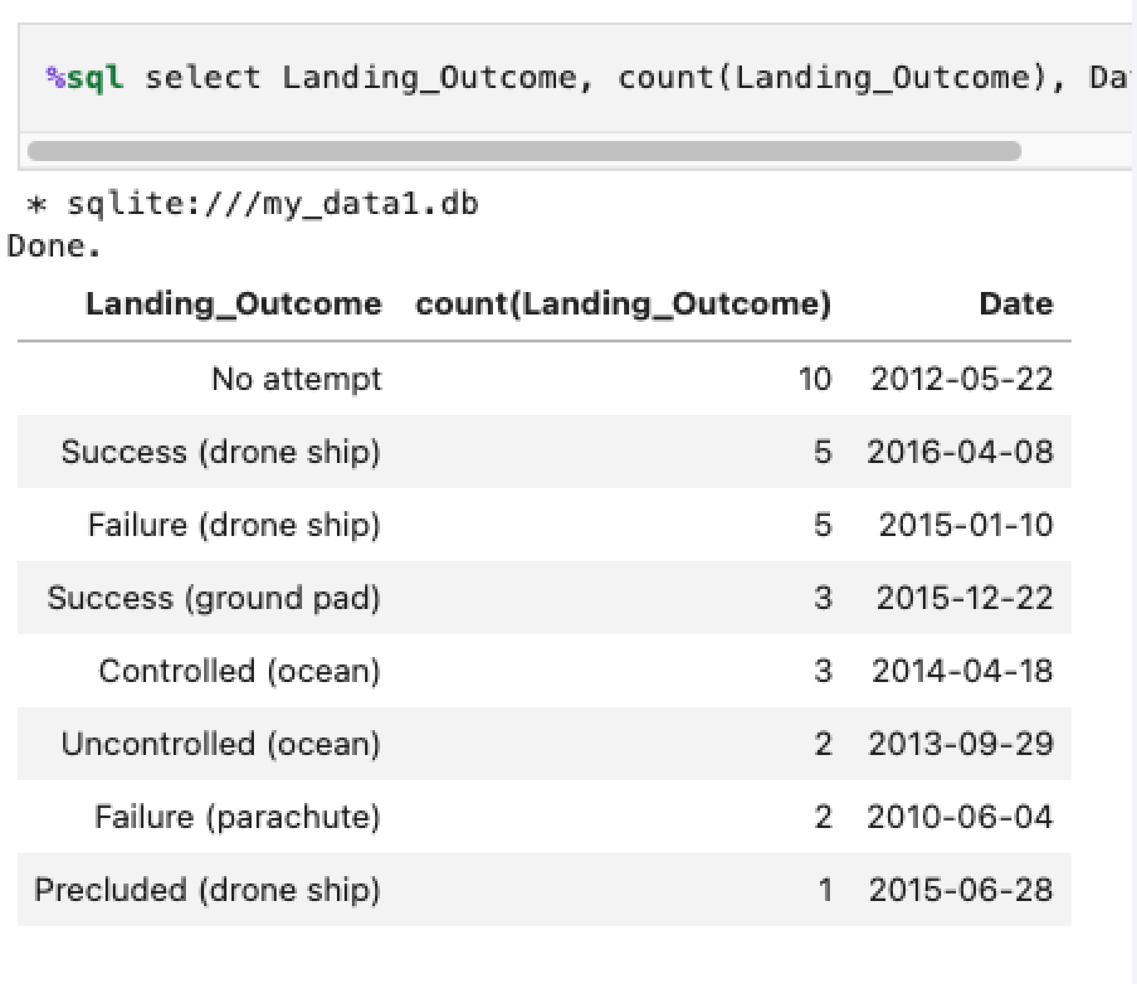
```
%sql select substr(Date, 6,2), substr(Date,0,5), landing_outcome, booster_version, launch_site fr
```

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

substr(Date, 6,2)	substr(Date,0,5)	Landing_Outcome	Booster_Version	Launch_Site
01	2015	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	2015	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- This SQL command allows us to see the ordered Landing Outcomes and their relative counts between the years 2010 and 2017.
- Here both the group by and order by clauses were used to group the data by distinct type of landing outcome and then order in descending order to make the data easier to understand.



```
%sql select Landing_Outcome, count(Landing_Outcome), Da
```

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

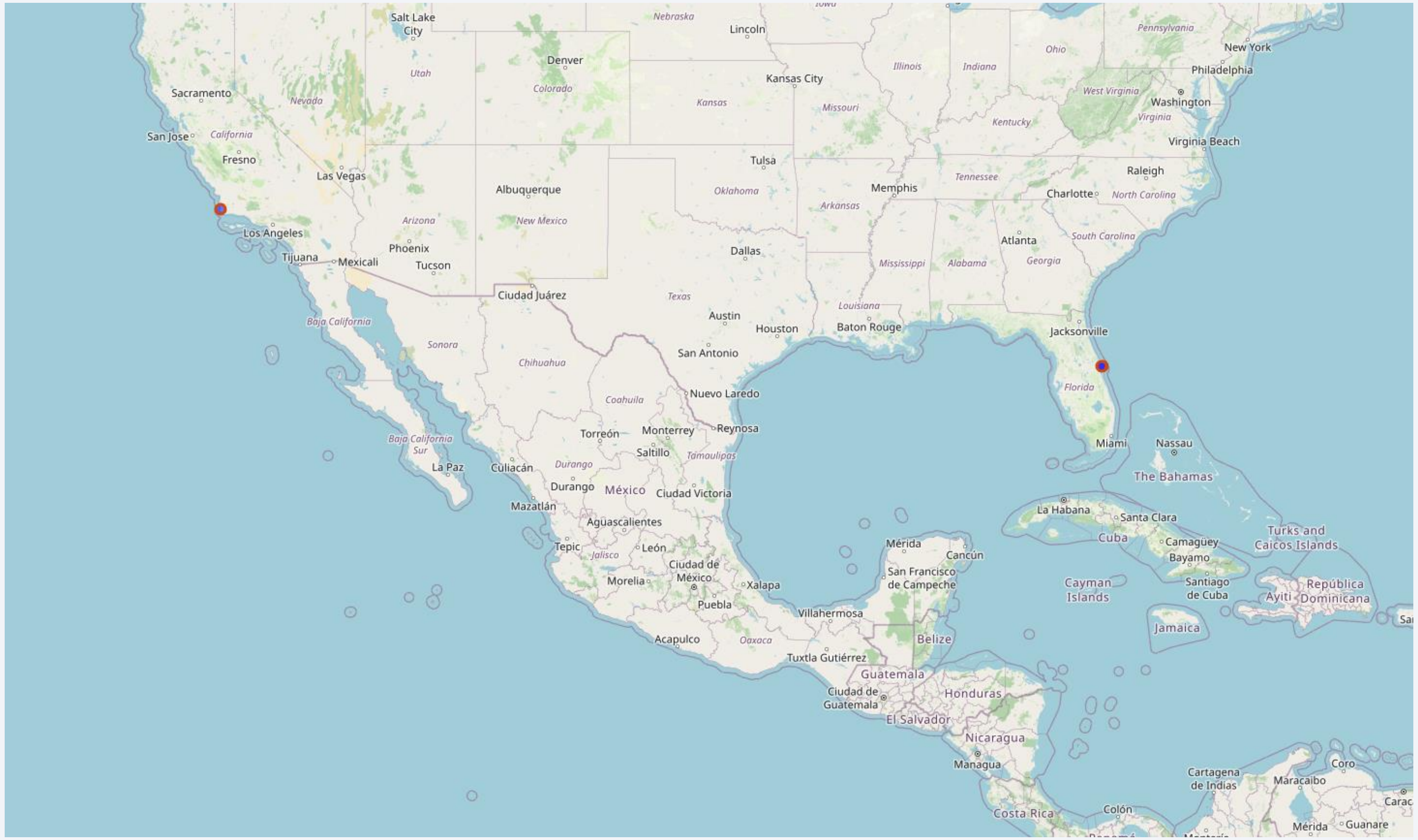
Landing_Outcome	count(Landing_Outcome)	Date
No attempt	10	2012-05-22
Success (drone ship)	5	2016-04-08
Failure (drone ship)	5	2015-01-10
Success (ground pad)	3	2015-12-22
Controlled (ocean)	3	2014-04-18
Uncontrolled (ocean)	2	2013-09-29
Failure (parachute)	2	2010-06-04
Precluded (drone ship)	1	2015-06-28

Section 3

Launch Sites Proximities Analysis

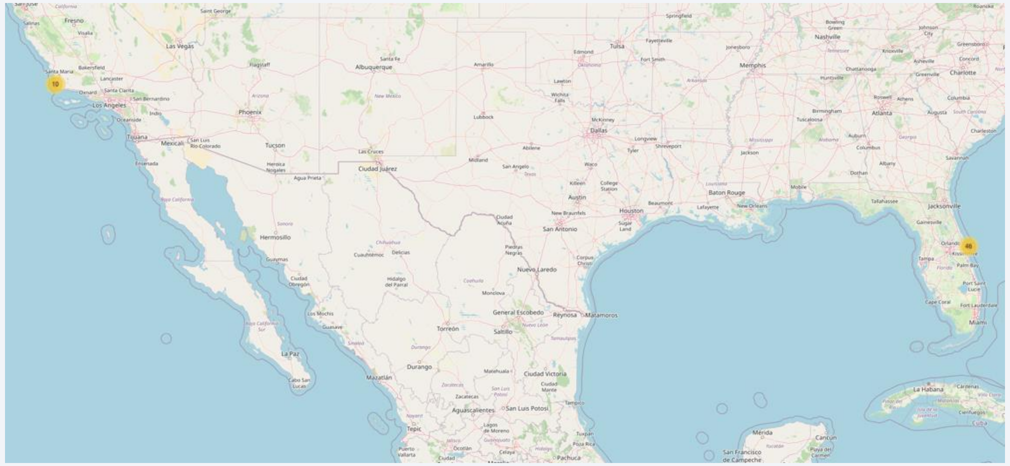
Marking Launch Points on Map of USA

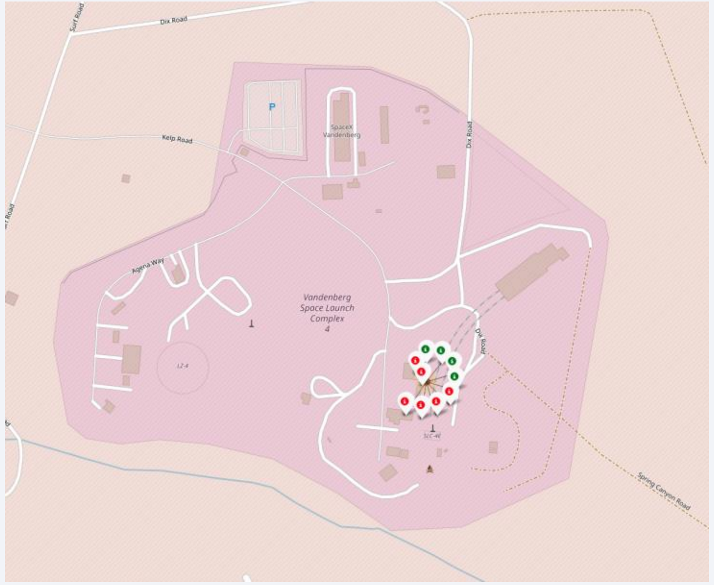
- This graph shows each launch point on a graph of the United States.
- We can see that they are clustered in two areas, southern costal Florida, and southern costal California.



Coding Successful and Unsuccessful landings.

- These graphs show both the number of launches and the success of each launch, visualized on the map by launch site.
- The launches are coded as green for successful launches and red for failed landings.

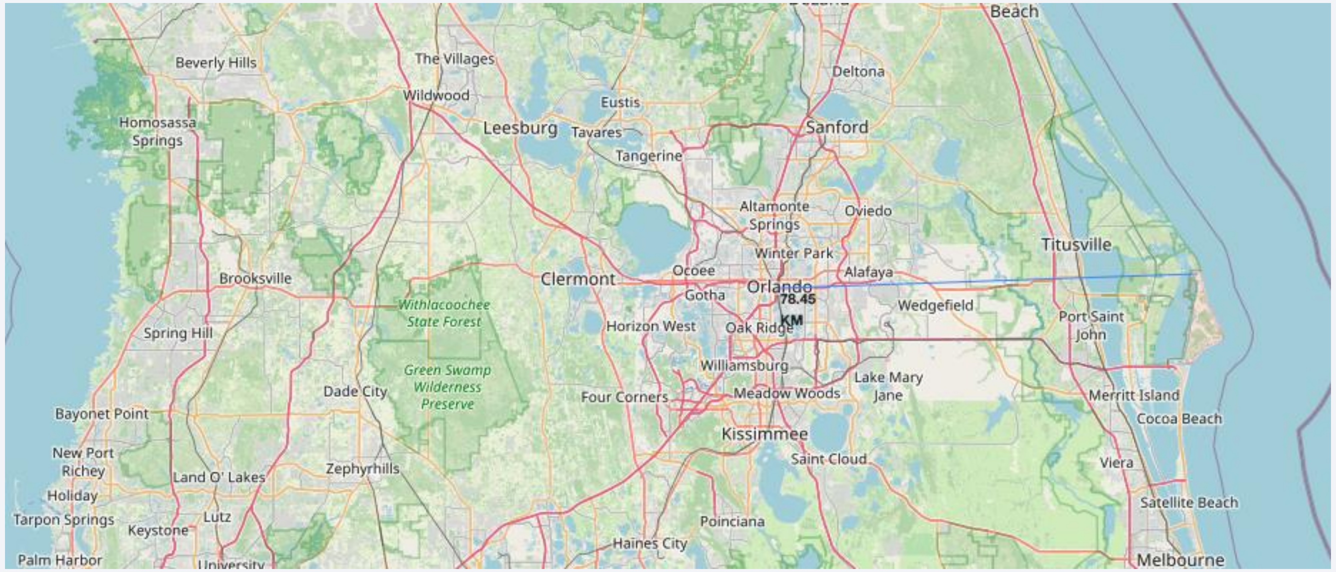


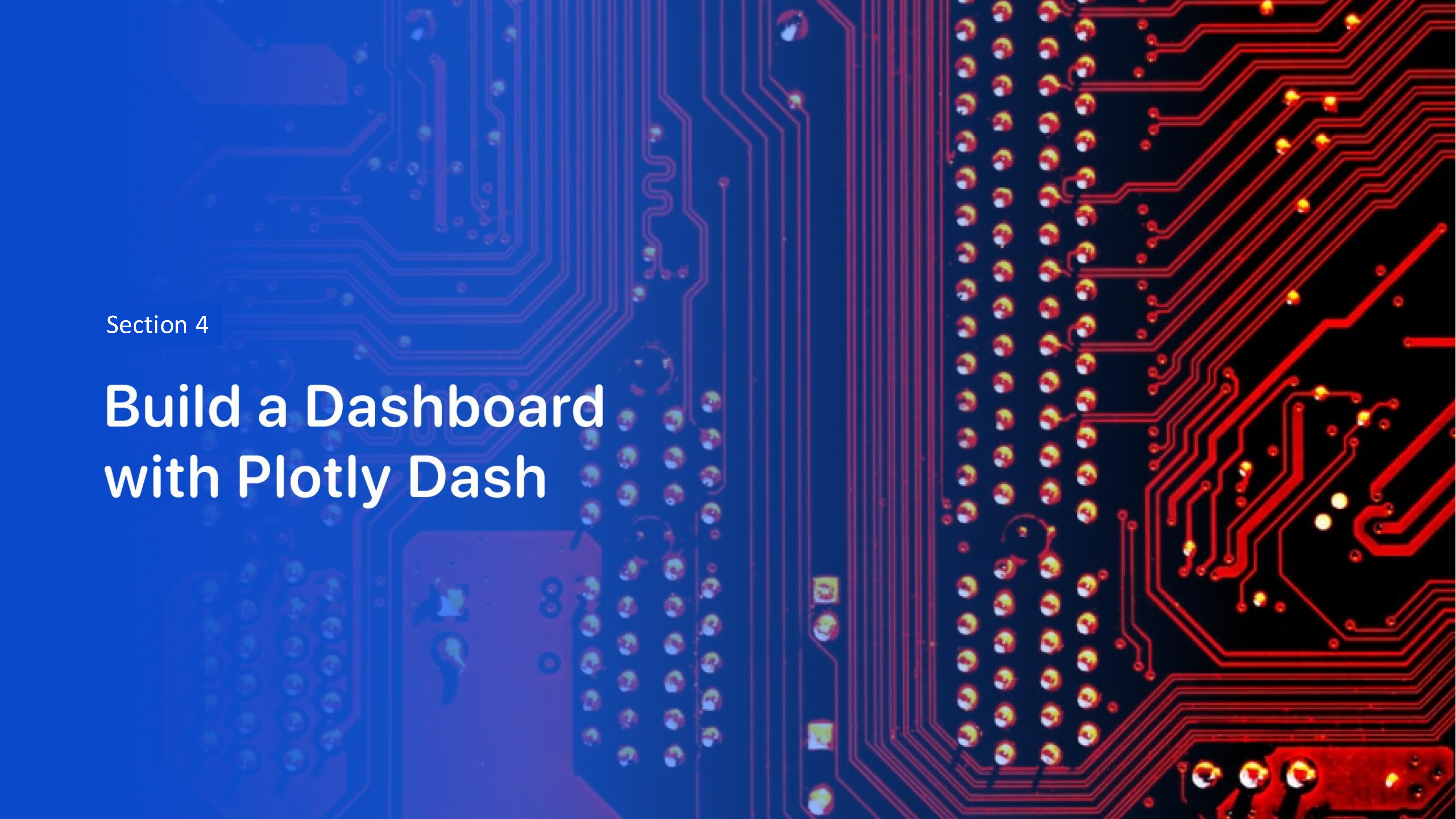


- We see that most launches are undertaken from the two launch-sites in Florida. And the launches themselves are centered around each launch pad.
- These landings are centered in costal areas, away from residentials or heavily populated areas.
- Successful and unsuccessful landings are spread relatively evenly between all launch sites.

Distance of launch site from nearest major city.

- This screenshot shows the distance of one of the launch sites to the nearest major city; Florida City.
- We can see that it is close to the city so the launch site can easily transport resources, but also that it is far enough that any effects would not be felt by the city's inhabitants.





Section 4

Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>

- Replace <Dashboard screenshot 1> title with an appropriate title
- Show the screenshot of launch success count for all sites, in a piechart
- Explain the important elements and findings on the screenshot

<Dashboard Screenshot 2>

- Replace <Dashboard screenshot 2> title with an appropriate title

- Show the screenshot of the piechart for the launch site with highest launch success ratio

- Explain the important elements and findings on the screenshot

<Dashboard Screenshot 3>

- Replace <Dashboard screenshot 3> title with an appropriate title

- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

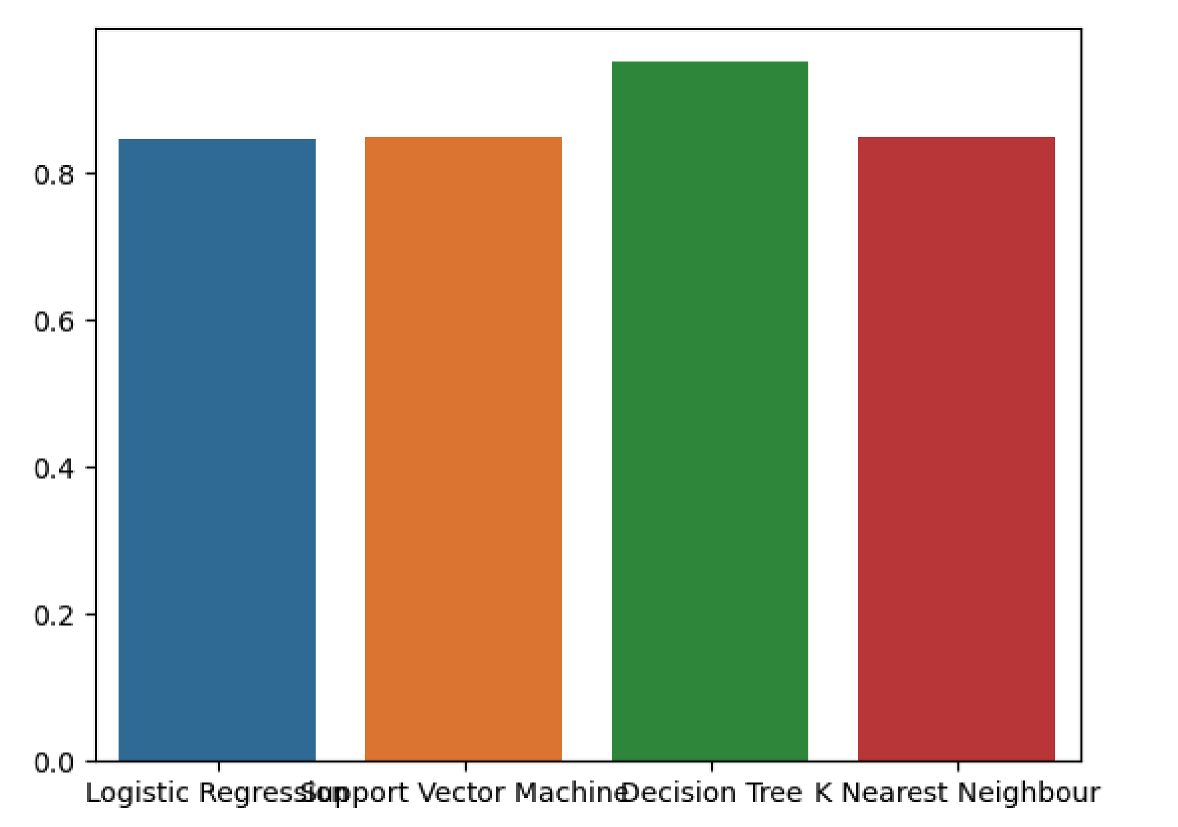
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

Section 5

Predictive Analysis (Classification)

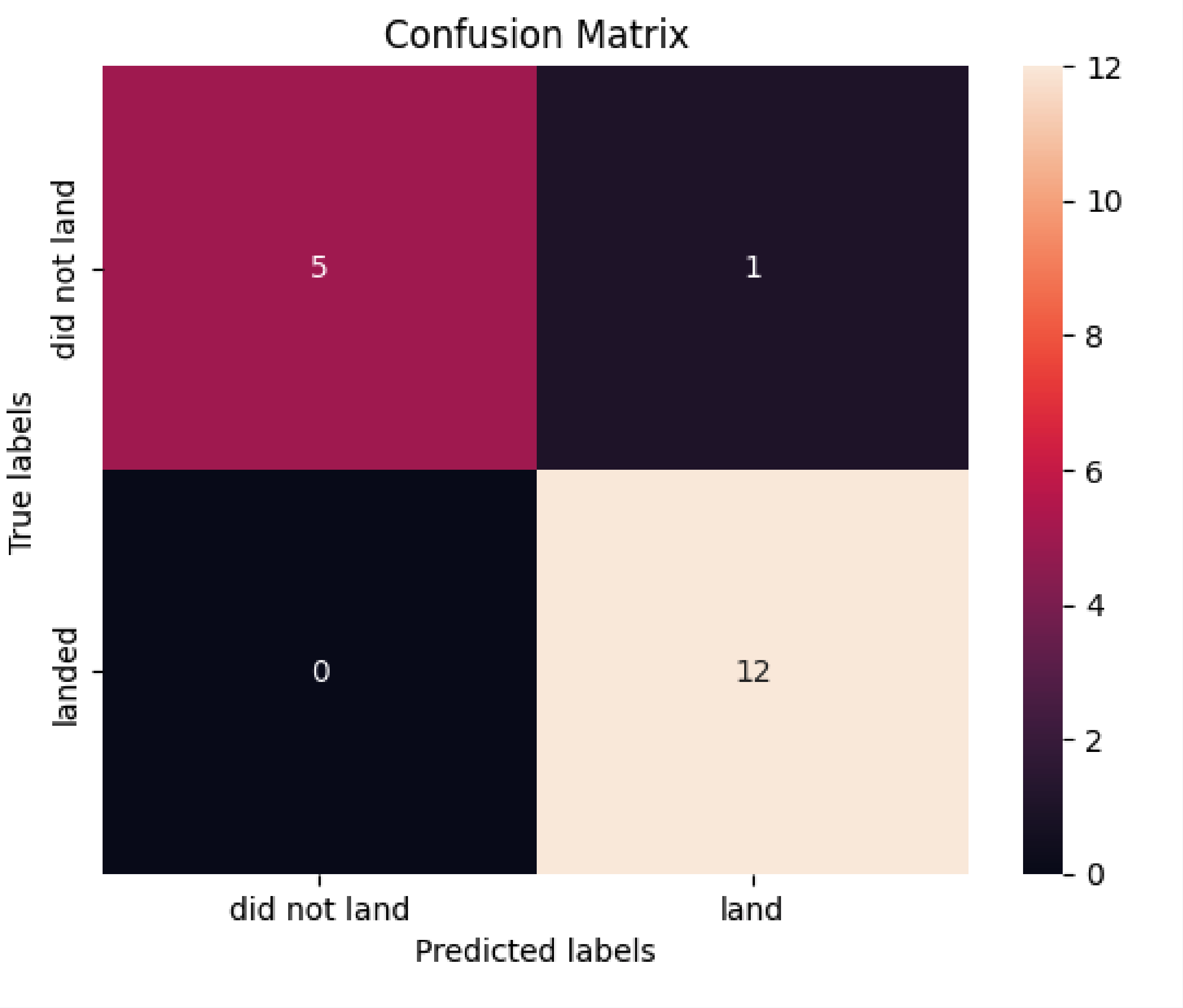
Classification Accuracy

- This bar plot shows the accuracy result for each of our prediction models
- We can see that the Decision Tree Model is the highest with a 95% accuracy.



Confusion Matrix

- This graph shows the confusion matrix for the Decision Tree model



Conclusions

- We can conclude that using a higher payload Mass has a higher success rate.
- Launch rates have gradually improved since 2013, peaking in 2019.
- Geocentric orbits such as GEO, HEO, VLEO had the highest success rate.
- KSC LC-39A has the greatest number of successful launches when compared to other launch sites.
- The Decision Tree model has the greatest prediction rate, at 95%.

Thank you!