IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sayedehtahereh Vakily

8 November 8 2024

Outline

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

Executive Summary

- **Summary of methodologies**
  - SpaceX Data Collection using SpaceX API
  - SpaceX Data Collection with Web Scraping
  - SpaceX Data Wrangling
  - SpaceX Exploratory Data Analysis using SQL
  - Space-X EDA DataViz Using Python Pandas and Matplotlib
  - Space-X Launch Sites Analysis with Folium-Interactive Visual Analytics and PlotyDash
  - SpaceX Machine Learning Landing Prediction
- **Summary of all results**
  - EDA results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis(Classification)

Introduction

- **Project background and context**

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

- **Problems you want to find answers**

In this capstone, we will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

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.

Data was first collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API. This was done by first defining a series helper functions that would help in the use of the API to extract information using identification numbers in the launch data and then requesting rocket launch data from the SpaceX API url.

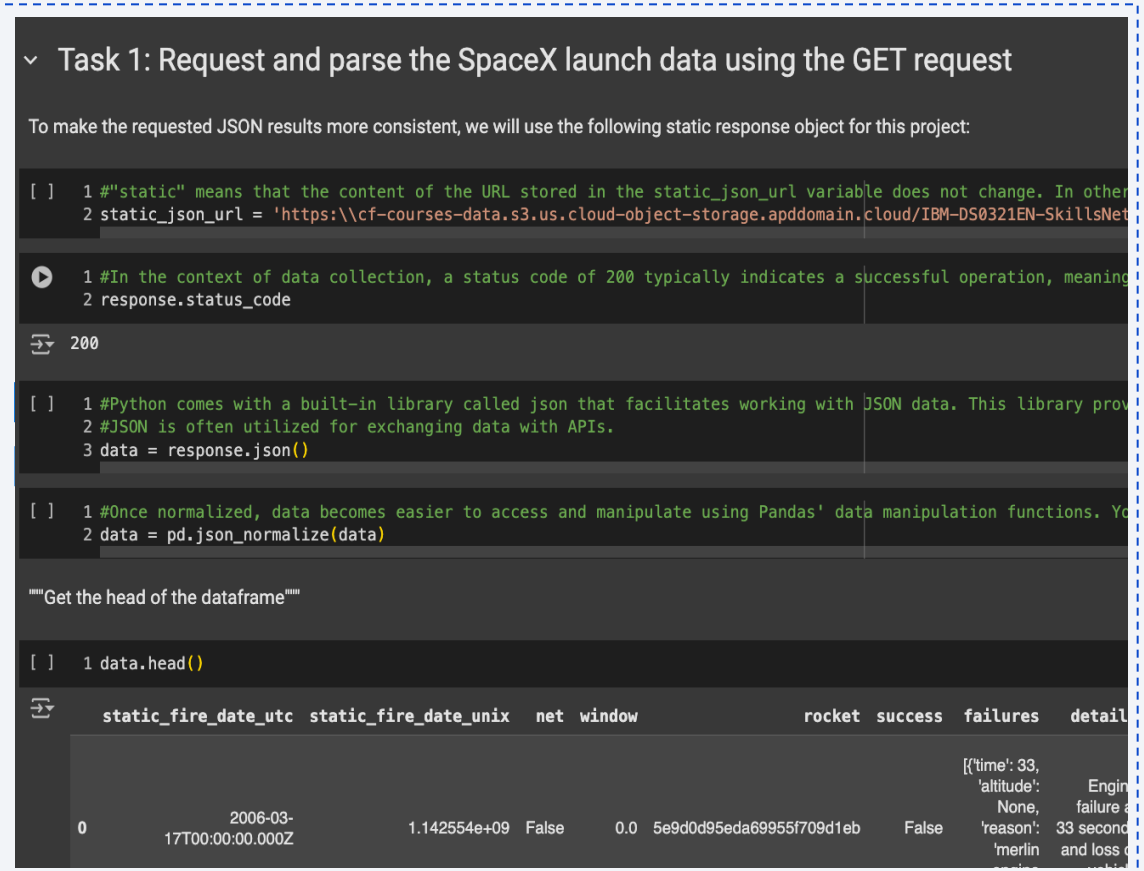
Finally to make the requested JSON results more consistent, the SpaceX launch data was requested and parsed using the GET request and then decoded the response content as a Jsonresult which was then converted into a Pandas data frame.

Also performed web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launchesof the launch records are stored in a HTML.

Data Collection – SpaceX API

- Data was collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API then requested and parsed the SpaceX launch data using the GET request and decoded the response content as a result which was then converted into a Pandas data frame
- Following is a GITHUB URL link for the spacex API call notebook.

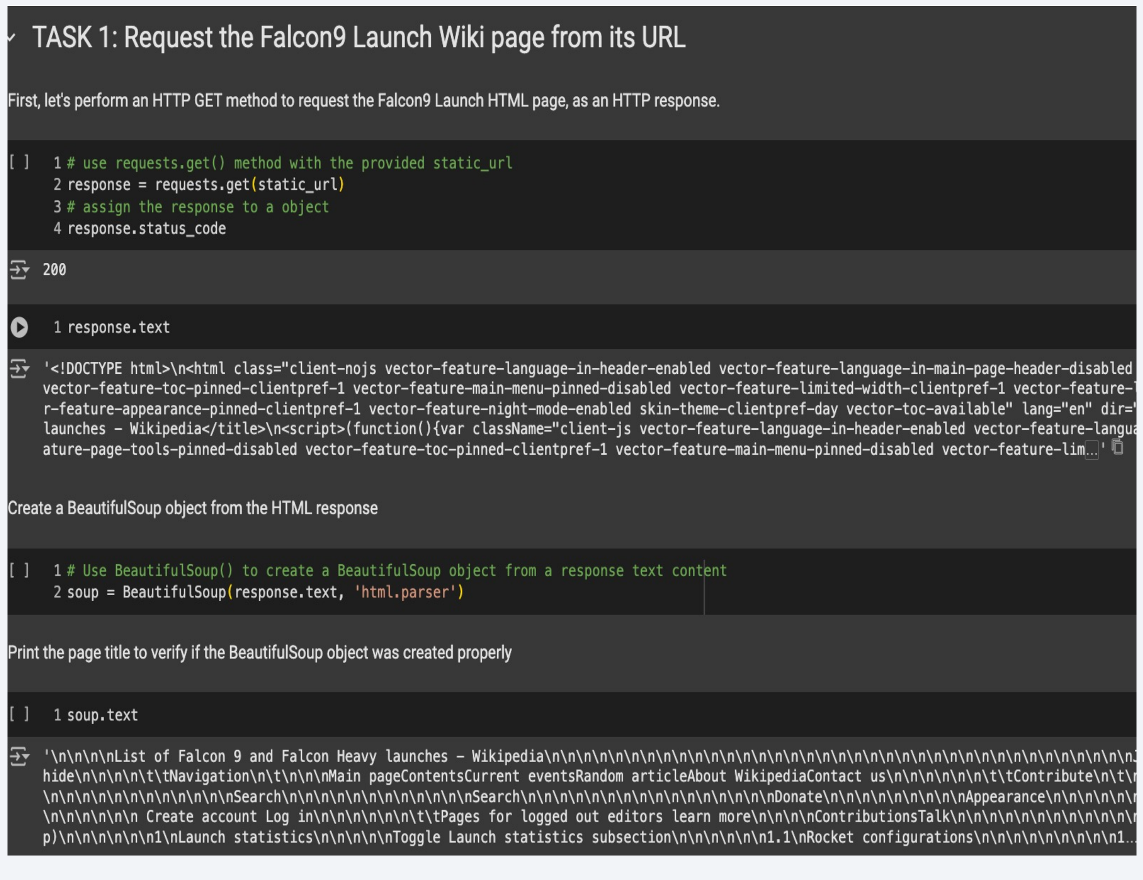
https://github.com/Stive88/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/main/data_collection_api_lab_stive_.py



Data Collection - Scraping

- To collect Falcon 9 historical launch records from Wikipedia using BeautifulSoup and request, to extract the Falcon 9 launch records from the HTML table of the Wikipedia page, then created a pandas data frame by parsing the launch HTML.
- Following is a GitHub URL link for the Spacex API call notebook.

https://github.com/Stive88/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/main/web_scraping_stive.py



Data Wrangling

- After obtaining and creating a Pandas DF from the collected data, data was filtered using the BoosterVersioncolumn to only keep the Falcon 9 launches, then dealt with the missing data values in the LandingPad and PayloadMasscolumns.
- Also performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
- Following is a GITHUB URL link for the spacex API call notebook.

https://github.com/Stive88/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/main/data_wrangling_stive.py

```
1 # Apply value_counts() on column LaunchSite
2 df['LaunchSite'].value_counts()
```

LaunchSite	count
CCAFS SLC 40	55
KSC LC 39A	22
VAFB SLC 4E	13

dtype: int64

```
[ ]  1 df['Longitude'].value_counts()
```

Longitude	count
-80.577366	55
-80.603956	22
-120.610829	13

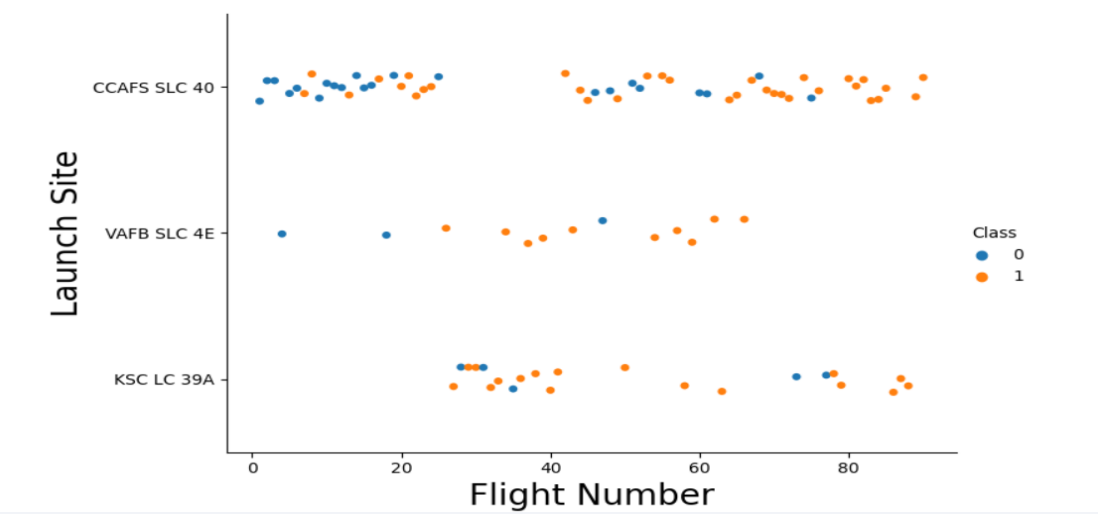
dtype: int64

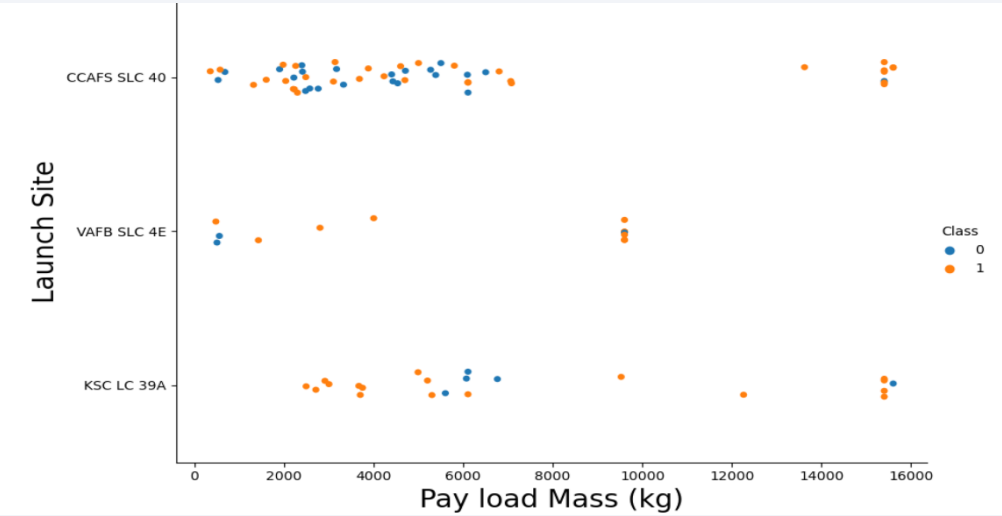
EDA with Data Visualization

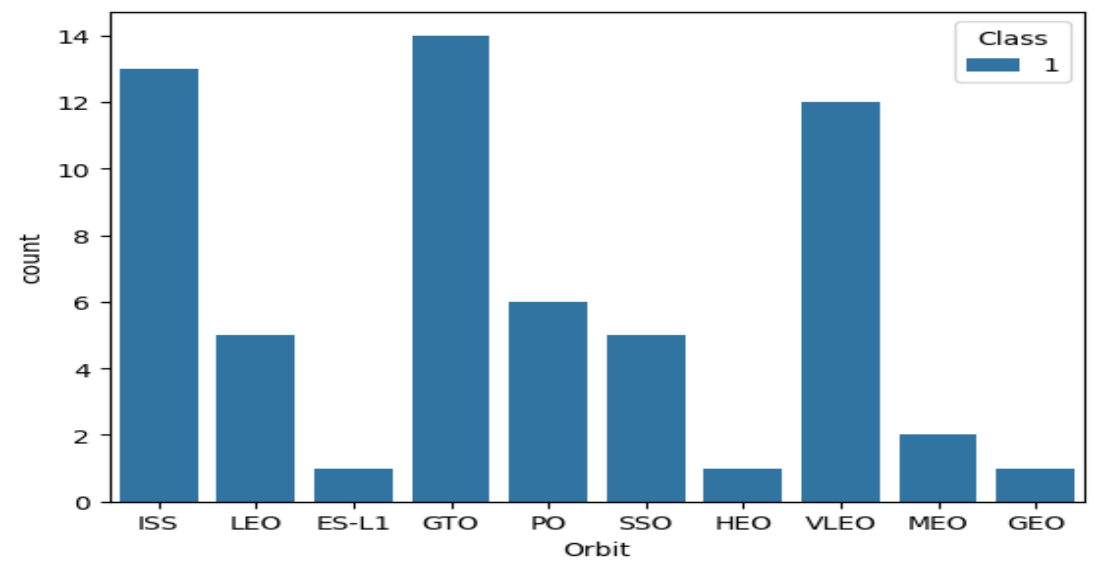
- Performed data Analysis and Feature Engineering using Pandas and Matplotlib.i.e.
- Exploratory Data Analysis
- Preparing Data Feature Engineering
- Used scatter plots to Visualize the relationship between Flight Number and Launch Site, Payload and Launch Site, FlightNumberand Orbit type, Payload and Orbit type.
- Used Bar chart to Visualize the relationship between success rate of each orbit type
- Line plot to Visualize the launch success yearly trend.
- Following is a GitHub link for EDA with Data Visualization notebook

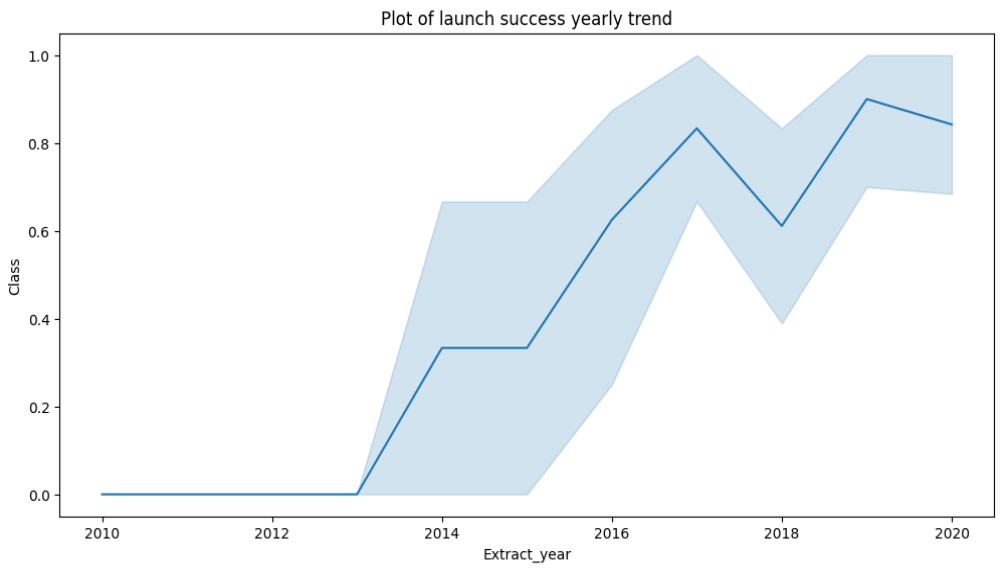
https://github.com/Stive88/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/main/eda_with_visualization_stive.py

EDA with Data Visualization









EDA with SQL

- Following tasks were performed in EDA with SQL

1. Understand the spacex dataset

2. Load the dataset into the corresponding table in db2 database

3. Execute the SQL queries to answer assignment questions.

Task 1

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

Task 2

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

```
%sql select * from SPACEXTBL where Launch_site like 'CCA%' limit 5
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Task 3

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) || ' kg' AS Total_Payload_Mass FROM SPACEXTBL WHERE Customer = 'NASA (CRS)';
```

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

Total_Payload_Mass
45596 kg

Task 4

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

```
%sql select avg(PAYLOAD_MASS__KG_) || ' kg'as Average_Payload_Mass from SPACEXTBL where Booster_Version like 'F9 v1.1'
```

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

Average_Payload_Mass
2928.4 kg

EDA with SQL

- Following tasks were performed in EDA with SQL

1. Understand the spacex dataset

2. Load the dataset into the corresponding table in db2 database

3. Execute the SQL queries to answer assignment questions.

Following is a GitHub link for a EDA with SQL notebook.

https://github.com/Stive88/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/main/exploratory_analysis_using_sql_stive.py

Task 5

List the date when the first succesful landing outcome in ground pad was acheived. Hint:Use min function

```
[ ]  1 %sql select min(Date) as First_Successful_Landing_Date from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
     2
```

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

First_Successful_Landing_Date
2015-12-22

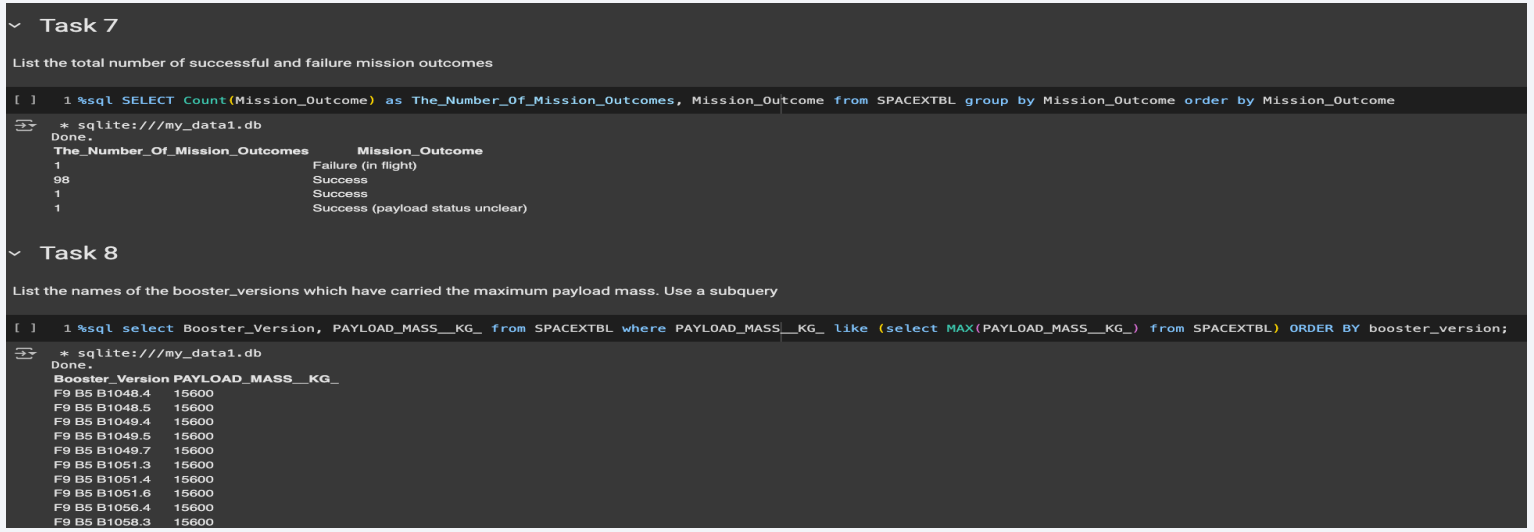
Task 6

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

```
 1 %sql select Booster_Version,PAYLOAD_MASS__KG_ , 'Success (drone ship)' from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000
```

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

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



Task 7

List the total number of successful and failure mission outcomes

```
[ ]  1 %sql SELECT Count(Mission_Outcome) as The_Number_Of_Mission_Outcomes, Mission_Outcome from SPACEXTBL group by Mission_Outcome order by Mission_Outcome
```

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

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

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
[ ]  1 %sql select Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTBL where PAYLOAD_MASS__KG_ like (select MAX(PAYLOAD_MASS__KG_) from SPACEXTBL) ORDER BY booster_version;
```

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

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

Build an Interactive Map with Folium

- Created folium map to marked all the launch sites, and created map objects such as markers, circles, lines to mark the success or failure of launches for each launch site.
- Created a launch set outcomes (failure=0 or success=1).
- Following is a GitHub link for a Interactive map with Folium,

https://github.com/Stive88/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/main/Interactive_visual_analytics_with_folium_stive.py

Build a Dashboard with Plotly Dash

- Built an interactive dashboard application with Plotly dash:
  - Add a Launch Site Drop-down Input Component
  - Add a callback function to render success-pie-chart based on selected site dropdown
  - Add a Range Slider to Select Payload
  - Add a callback function to render the success-payload-scatter-chart scatter plot
- Following is a GitHub link for a Dashboard with Plotly Dash

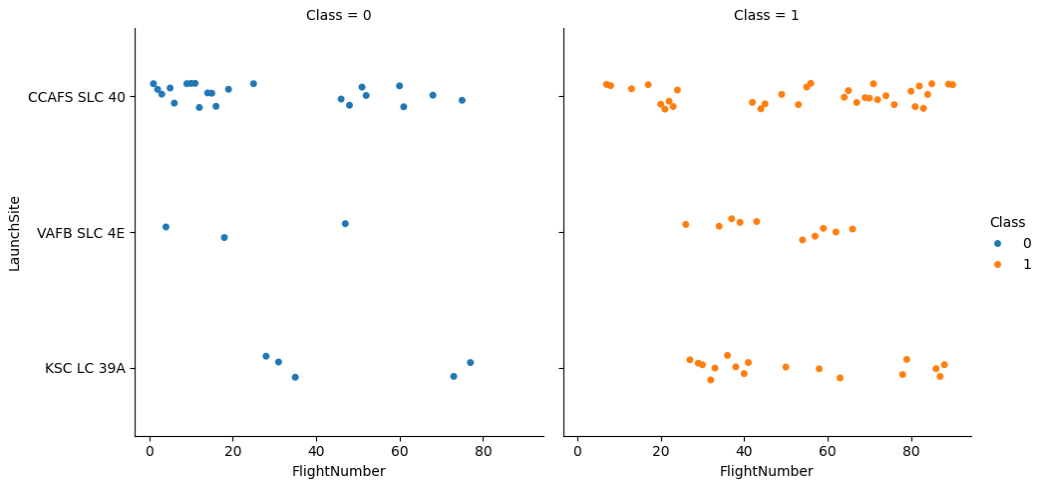
https://github.com/Stive88/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/main/spacex_launch_dash_stive.py

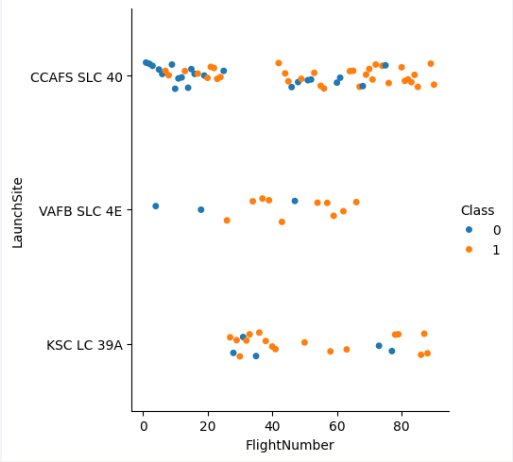
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

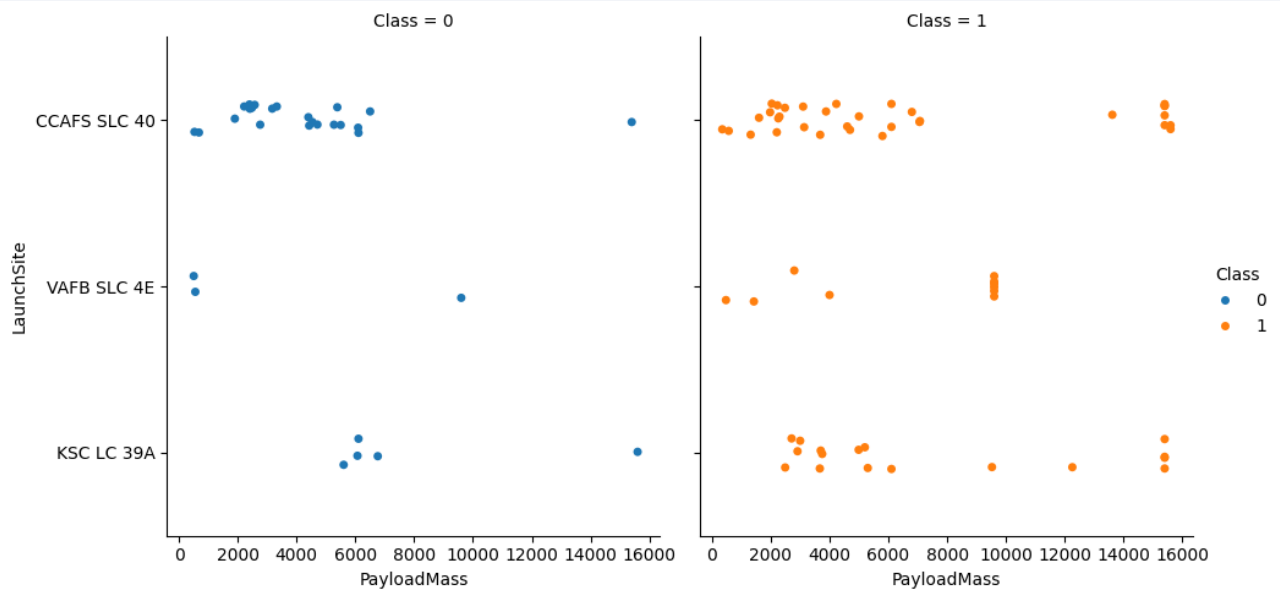
A Scatter plot for Flight number vs Launch site

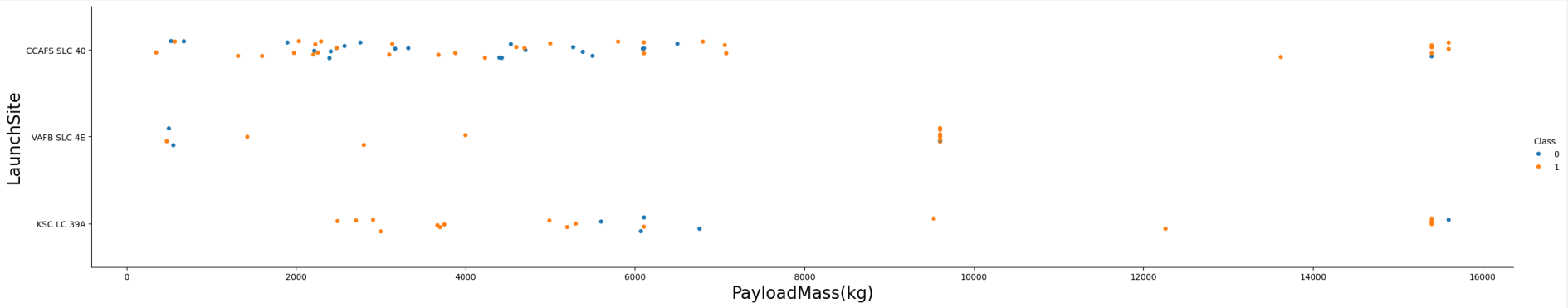




Payload vs. Launch Site

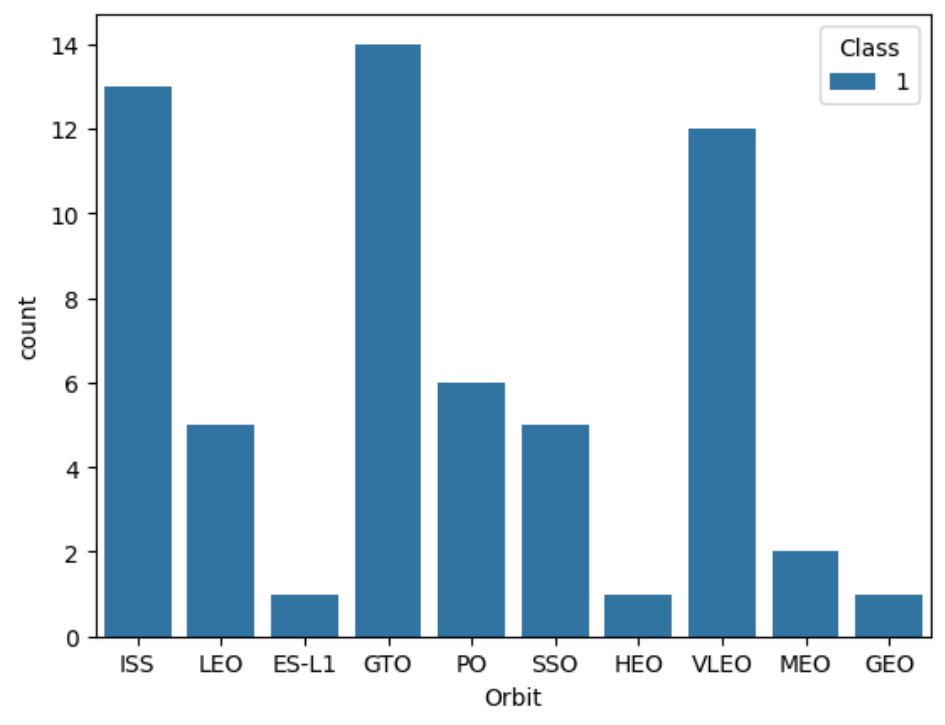
A Scatter plot for Payload mass vs Launch site

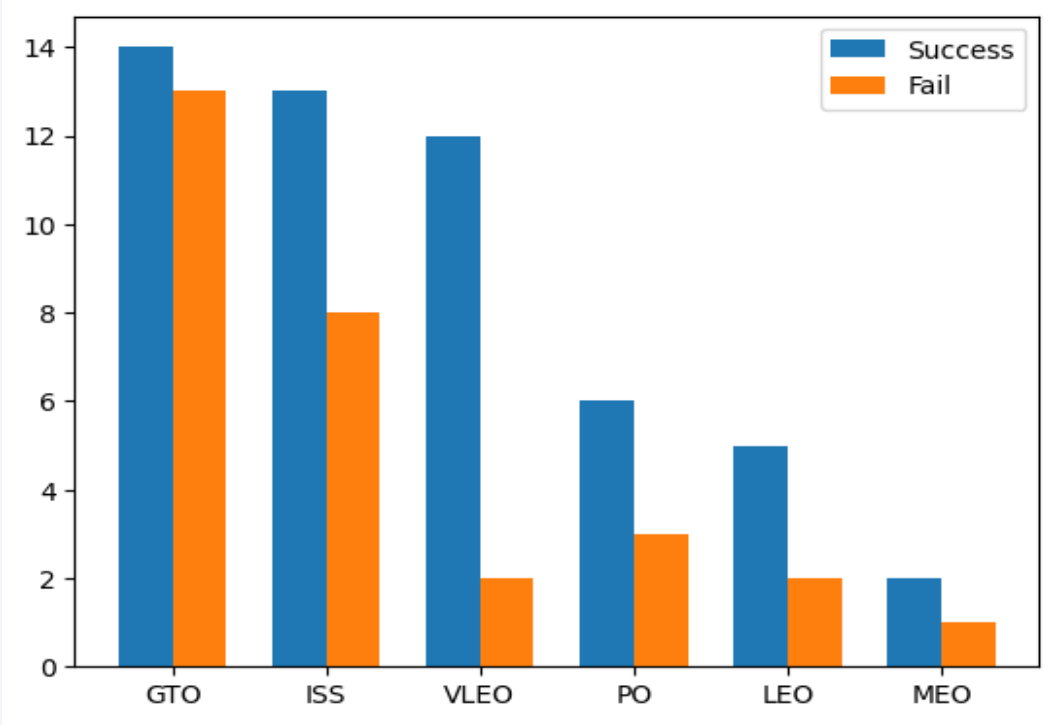




Success Rate vs. Orbit Type

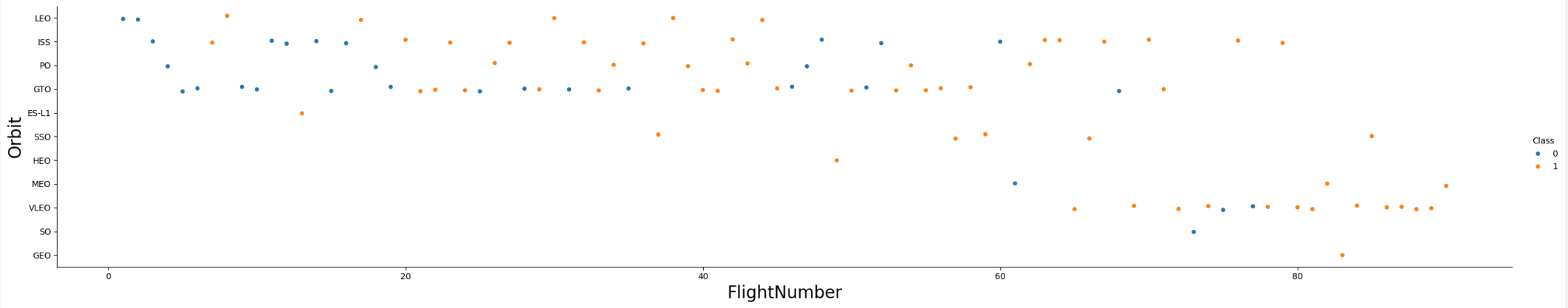
A Bar Chart for Success rate vs Orbit type





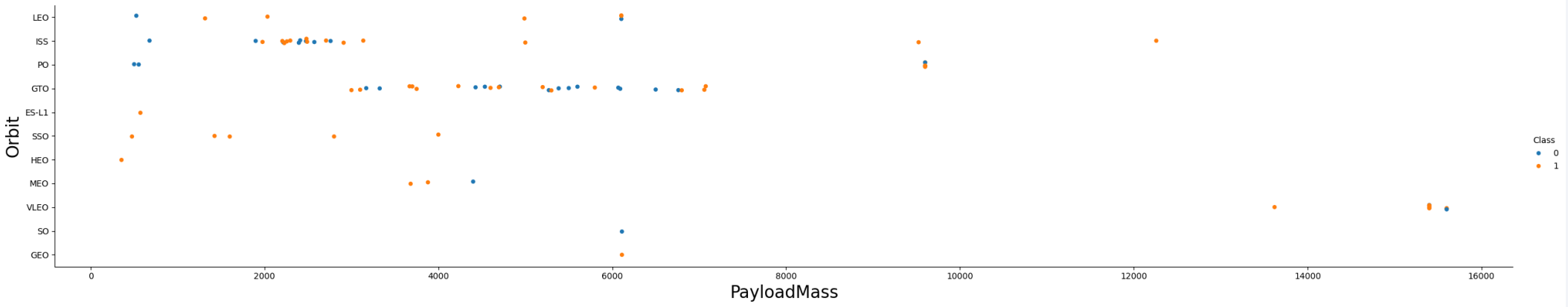
Flight Number vs. Orbit Type

A Scatter plot for Flight Number vs Orbit type



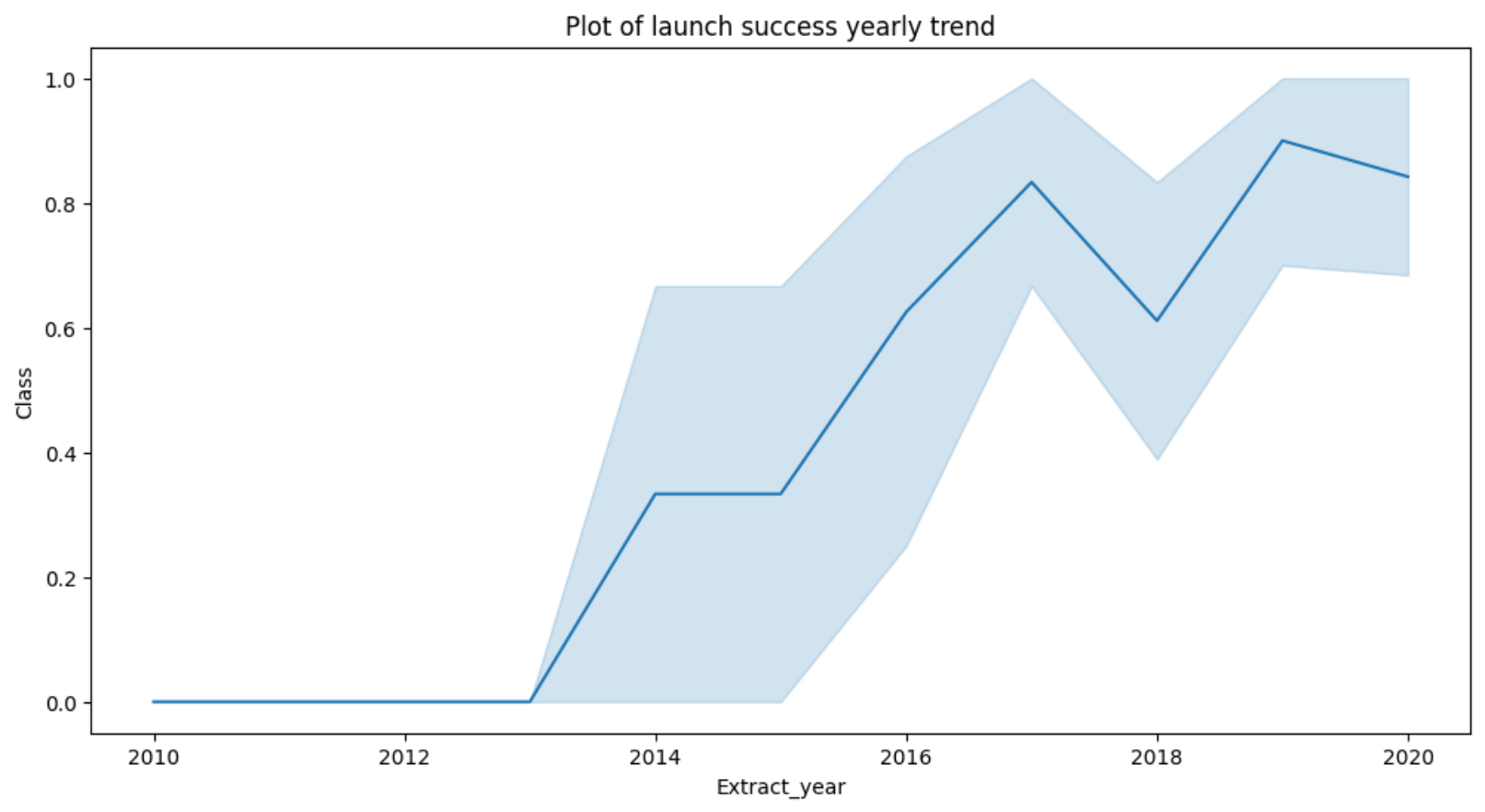
Payload vs. Orbit Type

A Scatter plot for Payload mass vs Orbit type



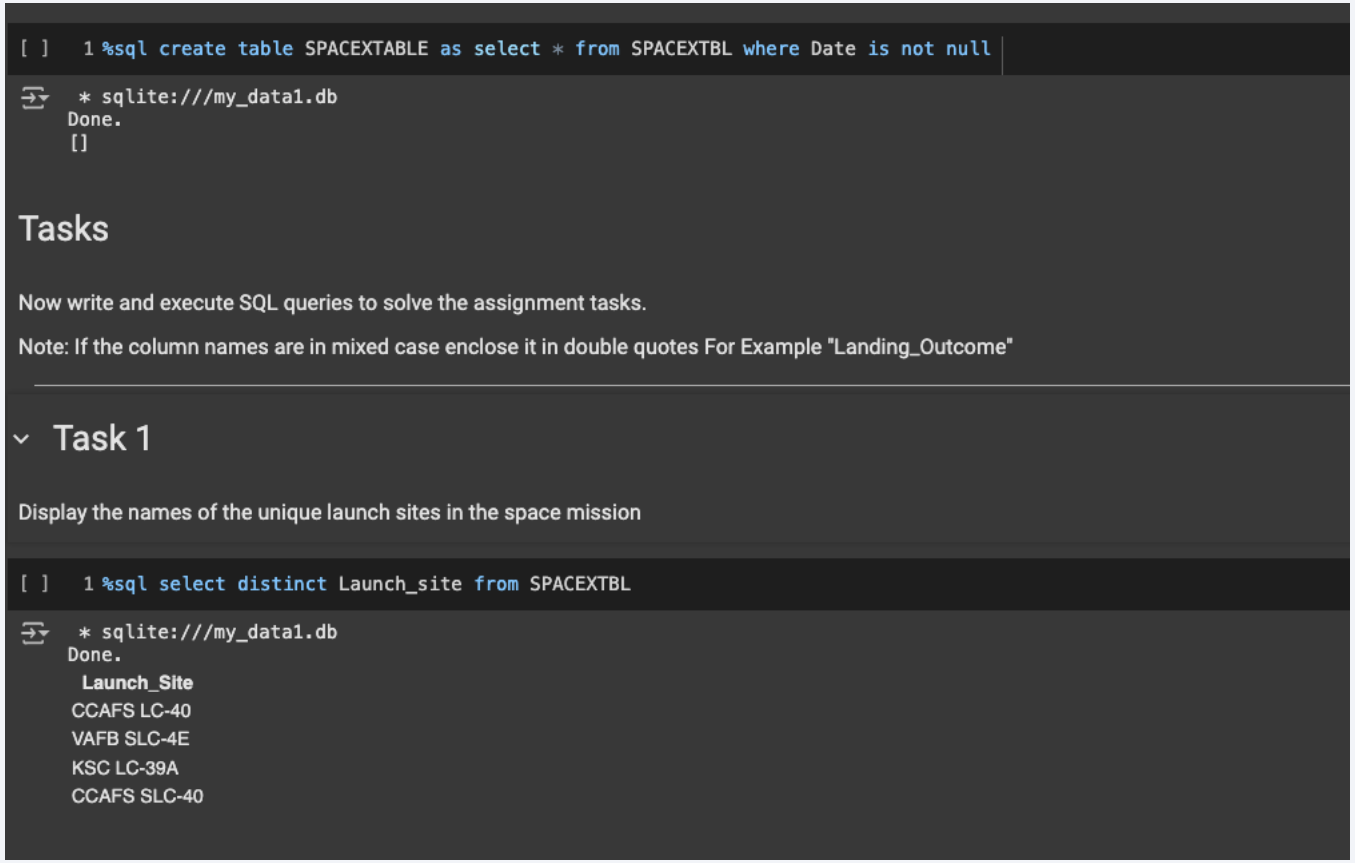
Launch Success Yearly Trend

A Line Chart for Launch Success yearly trend



All Launch Site Names

- Find the names of the unique launch sites
- Used the 'SELECT DISTINCT' statement to return only the unique launch sites from the 'LAUNCH_SITE' column of the SPACEXTBL table



Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

- Used ‘LIKE’ command with ‘%’ wildcard in ‘WHERE’ clause to select and display a table of all records where launch sites begin with the string 'CCA'

Task 2

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

```
%sql select * from SPACEXTBL where Launch_site like 'CCA%' limit 5
```

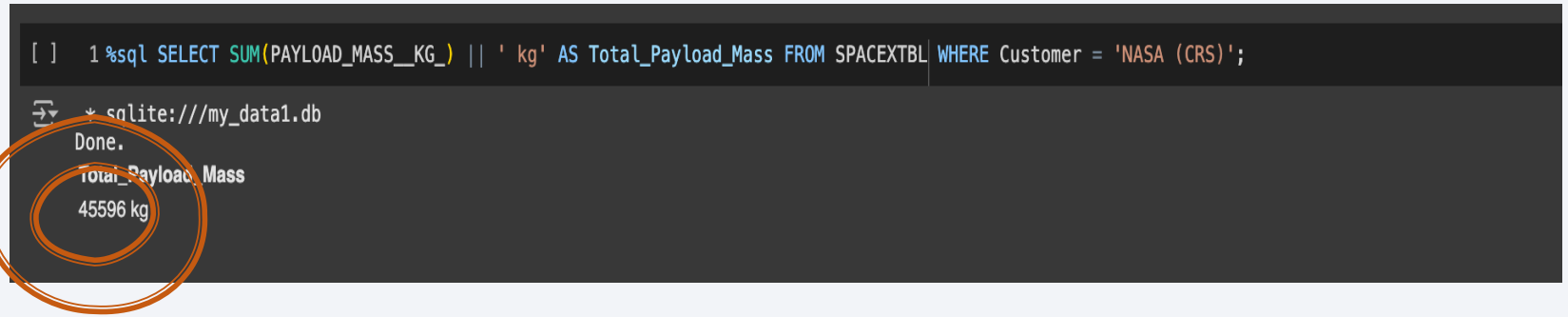
```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

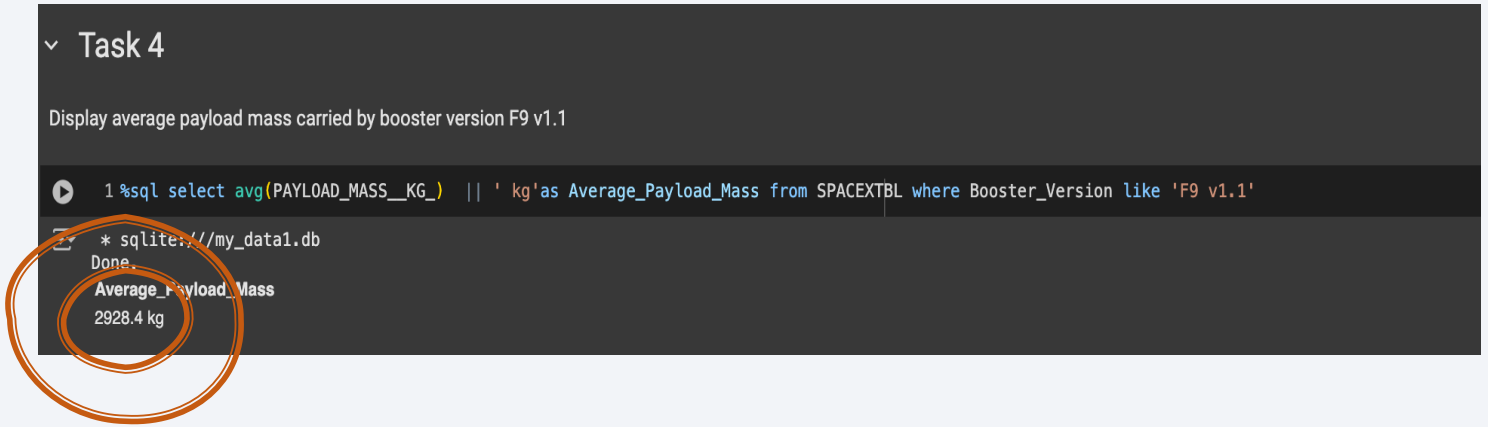
- Calculate the total payload carried by boosters from NASA

- Used the ‘SUM()’ function to return and display the total sum of the ‘PAYLOAD_MASS_KG’ column for Customer ‘NASA(CRS’)



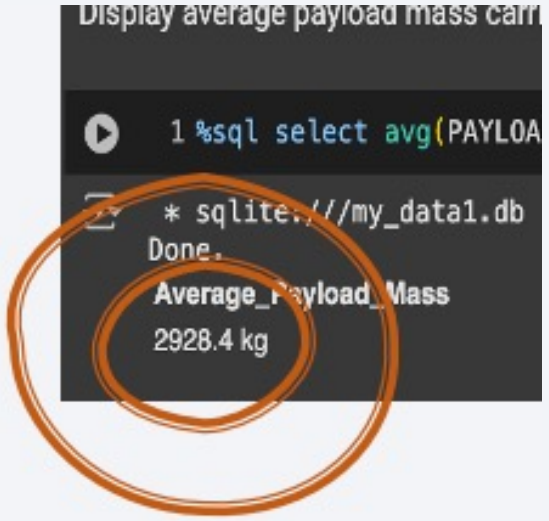
Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Used the 'AVG()' function to return and display the average payload mass carried by booster version F9 v1.1



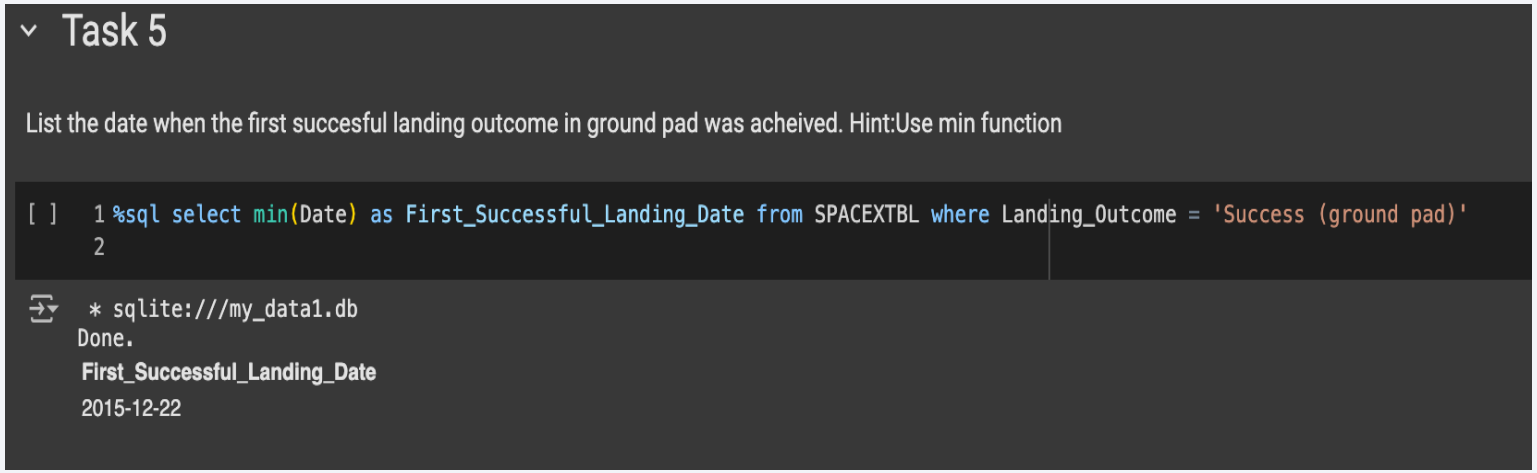
Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Used the ‘AVG()’ function to return and display the average payload mass carried by booster version F9 v1.1



First Successful Ground Landing Date

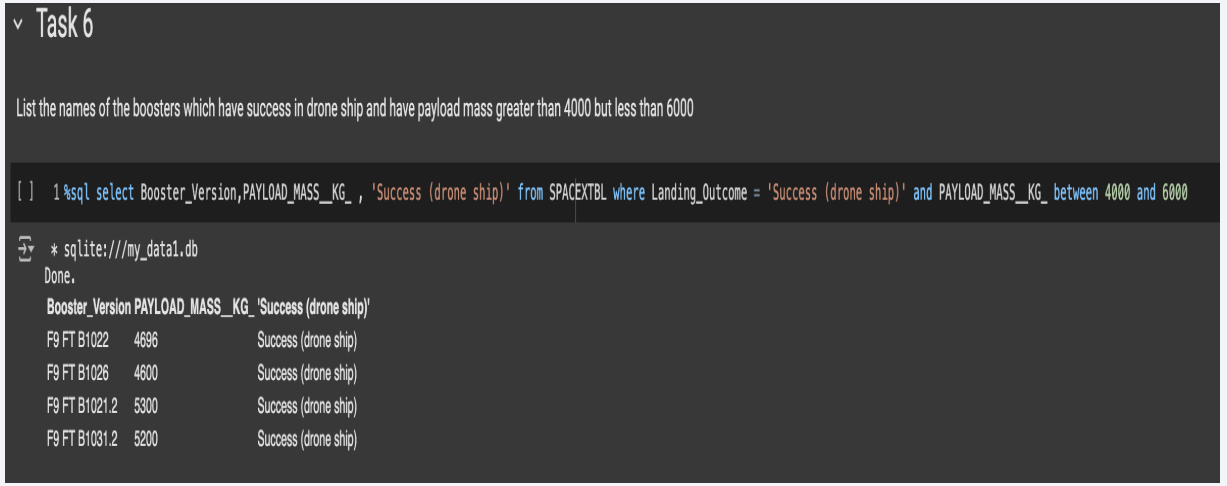
- Find the dates of the first successful landing outcome on the ground pad
- Used the 'MIN()' function to return and display the first (oldest) date when the first successful landing outcome on ground pad 'Success (ground pad)' happened.



Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

- Used the ‘Select Distinct’ statement to return and list the ‘unique’ names of boosters with operators >4000 and <6000 to only list boosters with payloads between 4000-6000 with landing outcome of ‘Success (drone ship)’.



Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

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

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

- Used the 'Select Distinct' statement to return and list the 'unique' names of boosters with operators >4000 and <6000 to only list boosters with payloads between 4000-6000 with landing outcome of 'Success (drone ship)'.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Used the 'COUNT()' together with the 'GROUP BY' statement to return the total number of mission outcomes

Task 7

List the total number of successful and failure mission outcomes

```
1 %sql SELECT Count(Mission_Outcome) as The_Number_Of_Mission_Outcomes, Mission_Outcome from SPACEXTBL group by Mission_Outcome order by Mission_Outcome
```

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

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

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

- Using a Subquery to return and pass the Max payload and used it list all the boosters that have carried the Max payload of 15600kgs

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
%sql select Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTBL where PAYLOAD_MASS__KG_ like (select MAX(PAYLOAD_MASS__KG_) from SPACEXTBL) ORDER BY booster_version;
```

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

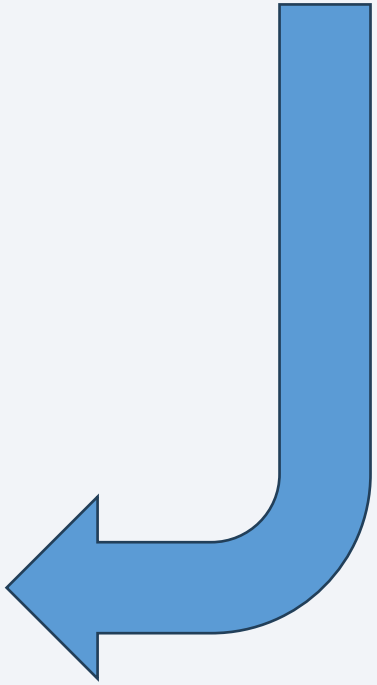
Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600

2015 Launch Records

```
%%sql
SELECT
    CASE SUBSTR(Date, 6, 2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END AS Month,
    Landing_Outcome,
    Booster_Version,
    Launch_Site
FROM
    SPACEXTBL
WHERE
    Landing_Outcome = 'Failure (drone ship)'
    AND SUBSTR(Date,0,5) = '2015';
```

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

Month	Landing_Outcome	Booster_Version	Launch_Site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40



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

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

Task 10

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

```
[ ]  1 %sql SELECT Landing_Outcome, COUNT(landing_outcome) AS Count, DATE FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY Count DESC;
```

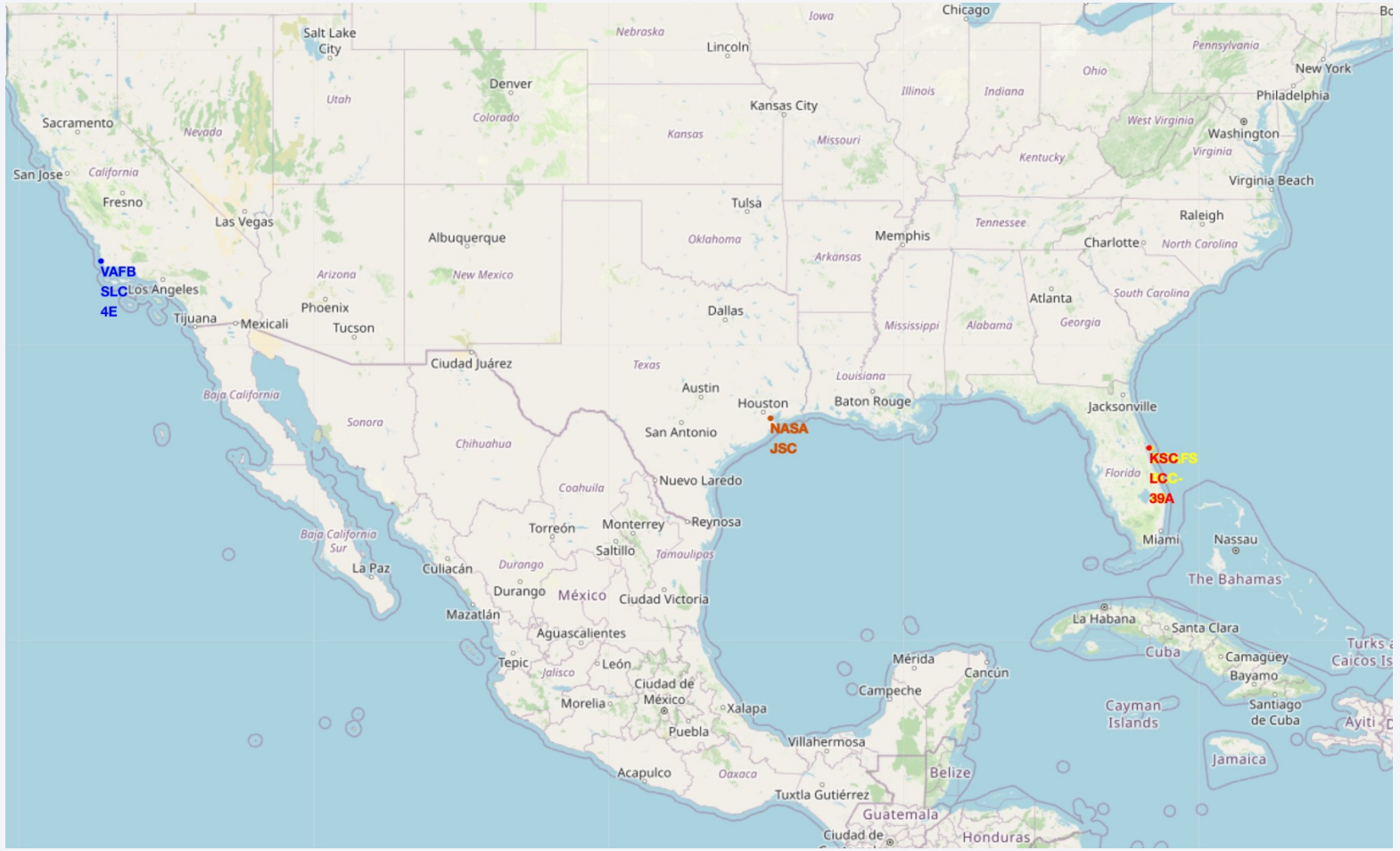
```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome	Count	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

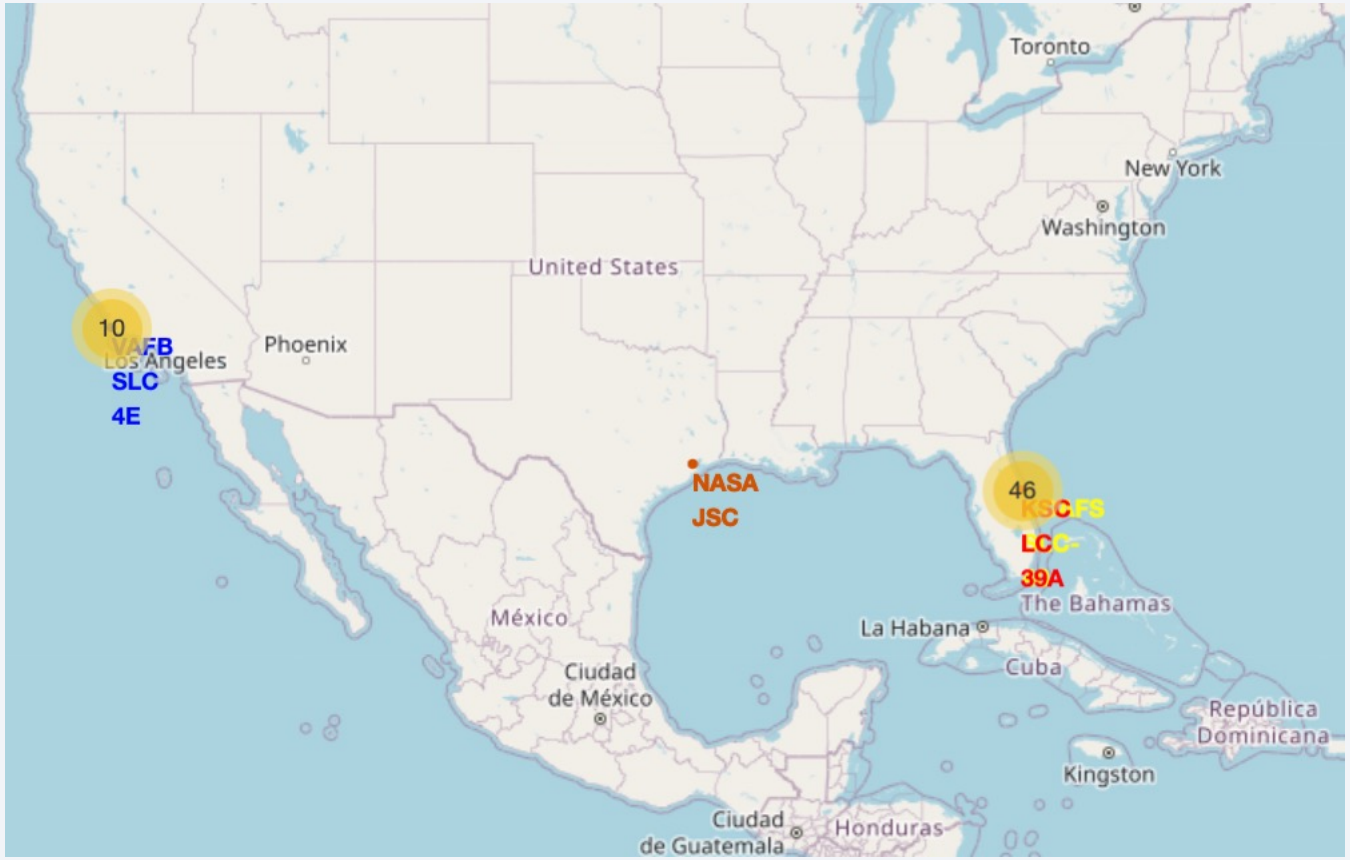
All Launch sites on the global map

All launch sites are in proximity to the Equator, (located southwards of the US map). All the launch sites are in very close proximity to the coast



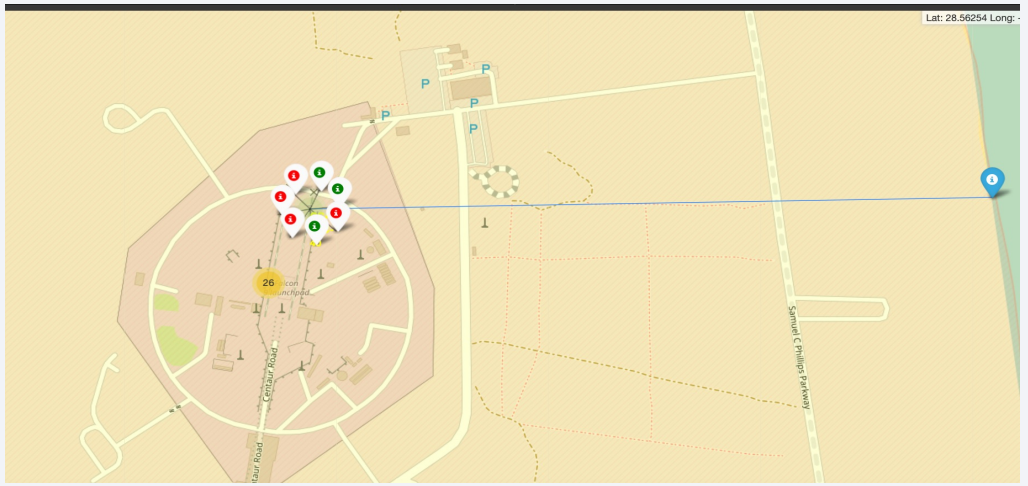
Success/failed Launches for each site on the map

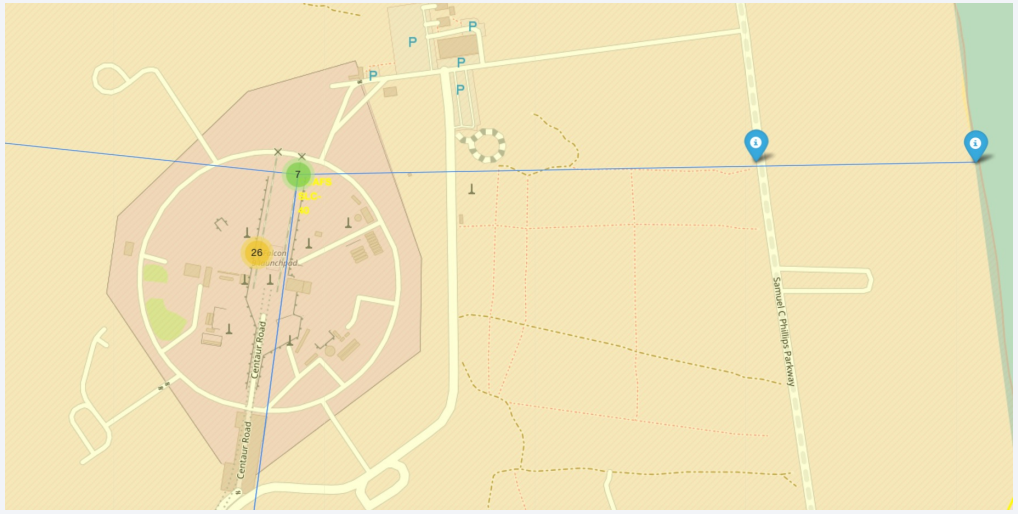
In the Eastern coast (Florida) Launch site KSC LC-39A has relatively high success rates compared to CCAFS SLC-40 & CCAFS LC-40.

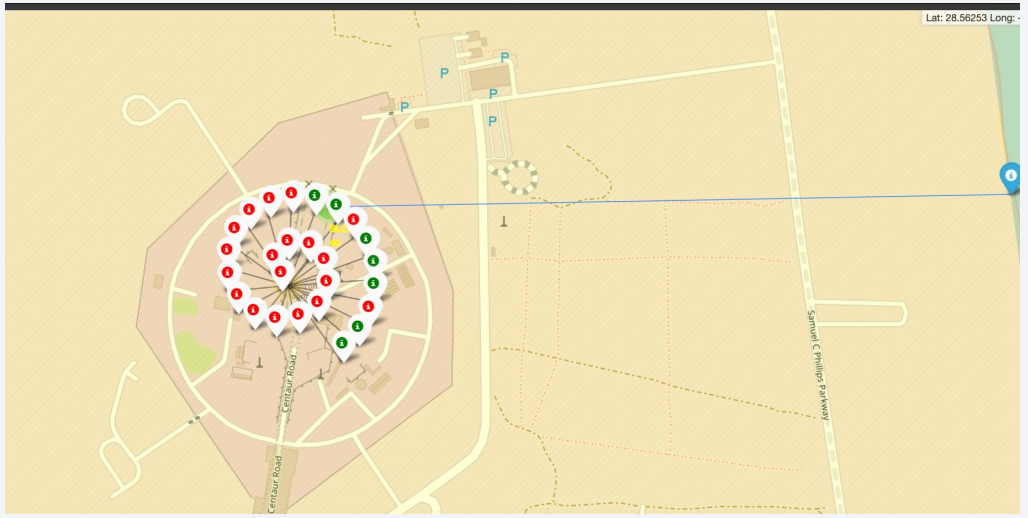


Success/failed Launches for each site on the map

In the West Coast (California) Launch site VAFB SLC-4E has relatively lower success rates 4/10 compared to KSC LC-39A launch site in the Eastern Coast of Florida.







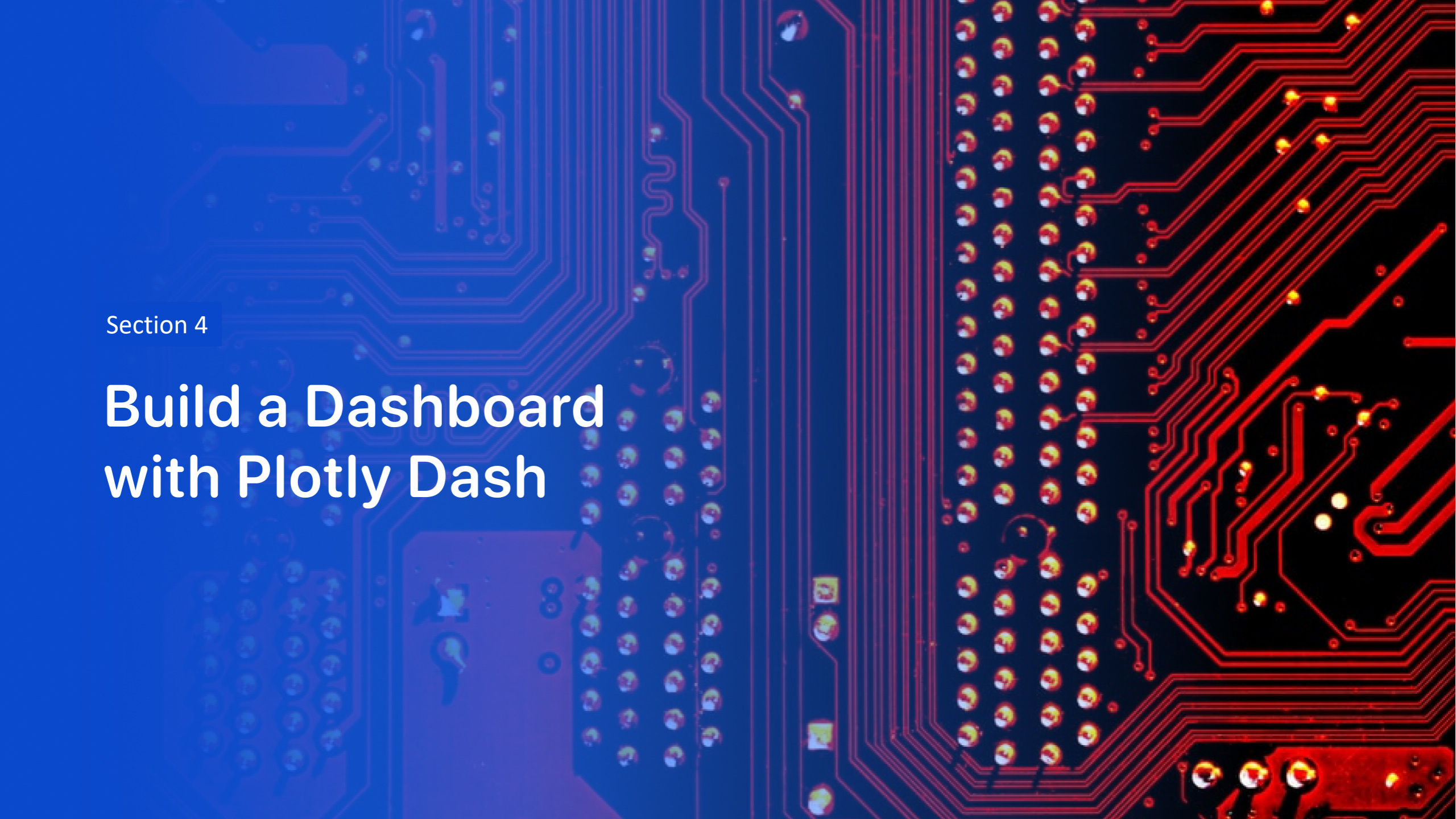
Distance between the launch sites and its proximities

```
# Create a marker with distance to a closest city, railway, highway, etc.
# Draw a line between the marker to the launch site
closest_city = 28.10473, -80.64531
closest_highway = 28.56335, -80.57085
closest_railroad = 28.56414, -80.58682
```

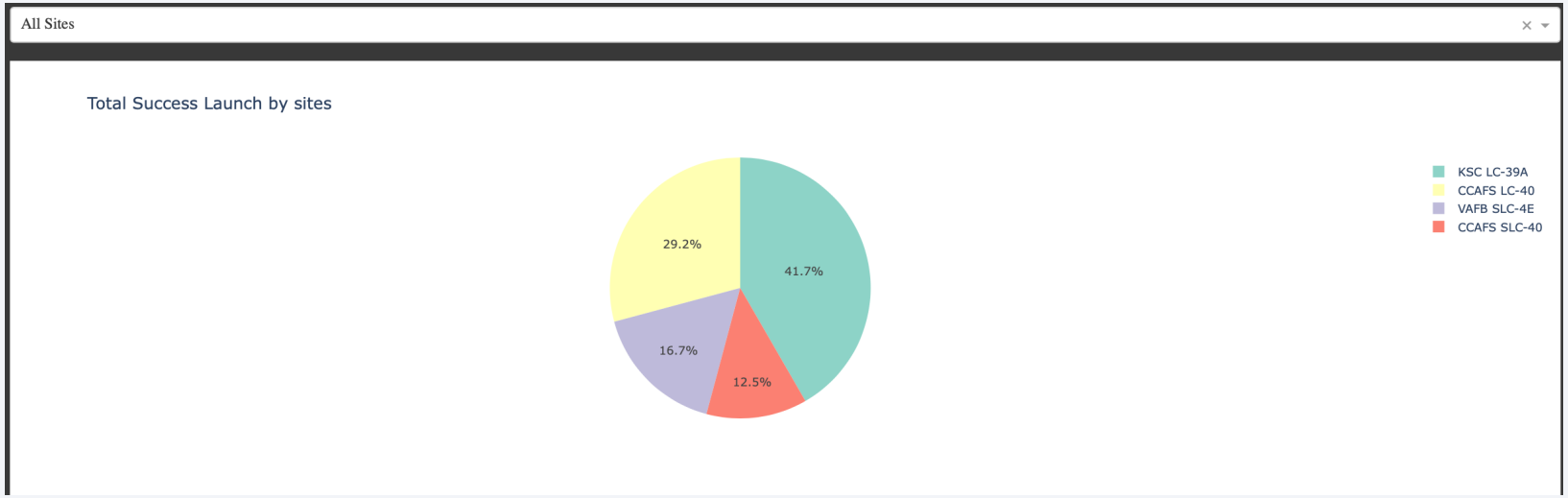
```
distance_city = calculate_distance(launch_site_lat, launch_site_lon, closest_city[0], close
print('Distance City:', distance_city, 'KM')
distance_highway = calculate_distance(launch_site_lat, launch_site_lon, closest_highway[0],
print('Distance Highway:' , distance_highway, 'KM')
distance_railroad = calculate_distance(launch_site_lat, launch_site_lon, closest_railroad[0]
print('Distance Railroad:' ,distance_railroad, 'KM')
```

```
Distance City: 51.43416999517233 KM
Distance Highway: 0.5834695366934144 KM
Distance Railroad: 0.9825312045582142 KM
```

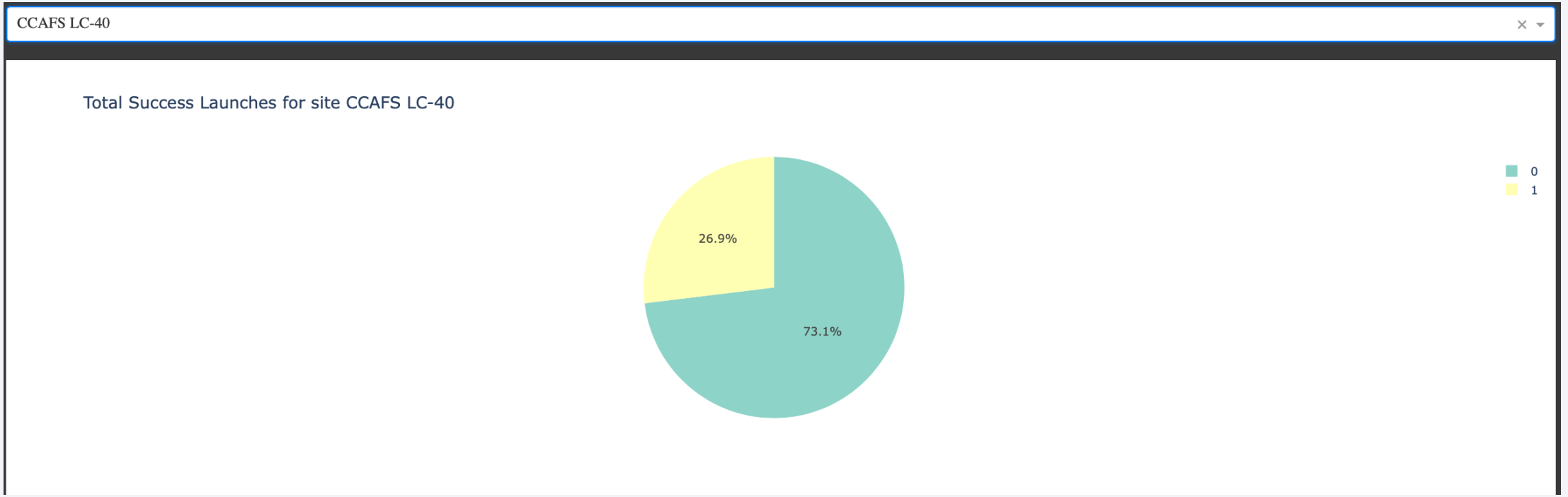
```
#Closest City Marker
folium.Marker(
    location=closest_city,
    popup='Closest City''<div style="font-size: 12px; color: #0000ff; text-align: center;"><
    icon=folium.Icon(color='blue', icon='info-sign')
).add_to(site_map)

#Closest City Line
coordinates = [[launch_site_lat, launch_site_lon], closest_city]
lines = folium.PolyLine(coordinates, weight=1)
site_map.add_child(lines)

#Closest Highway Marker
folium.Marker(
    location=closest_highway,
    popup='Closest Highway''<div style="font-size: 12px; color: #0000ff; text-align: center;
    icon=folium.Icon(color='blue', icon='info-sign')
).add_to(site_map)

#Closest Highway Line
coordinate = [[launch_site_lat, launch_site_lon], closest_highway]
lines = folium.PolyLine(coordinates, weight=1)
site_map.add_child(lines)

#Closest Railroad Marker
folium.Marker(
    location=closest_railroad,
    popup='Closest Railroad''<div style="font-size: 12px; color: #0000ff; text-align: center
    icon=folium.Icon(color='blue', icon='info-sign')
).add_to(site_map)

#Closest Railroad Line
coordinates = [[launch_site_lat, launch_site_lon], closest_railroad]
lines = folium.PolyLine(coordinates, weight=1)
site_map.add_child(lines)
```


Section 4
Build a Dashboard with Plotly Dash

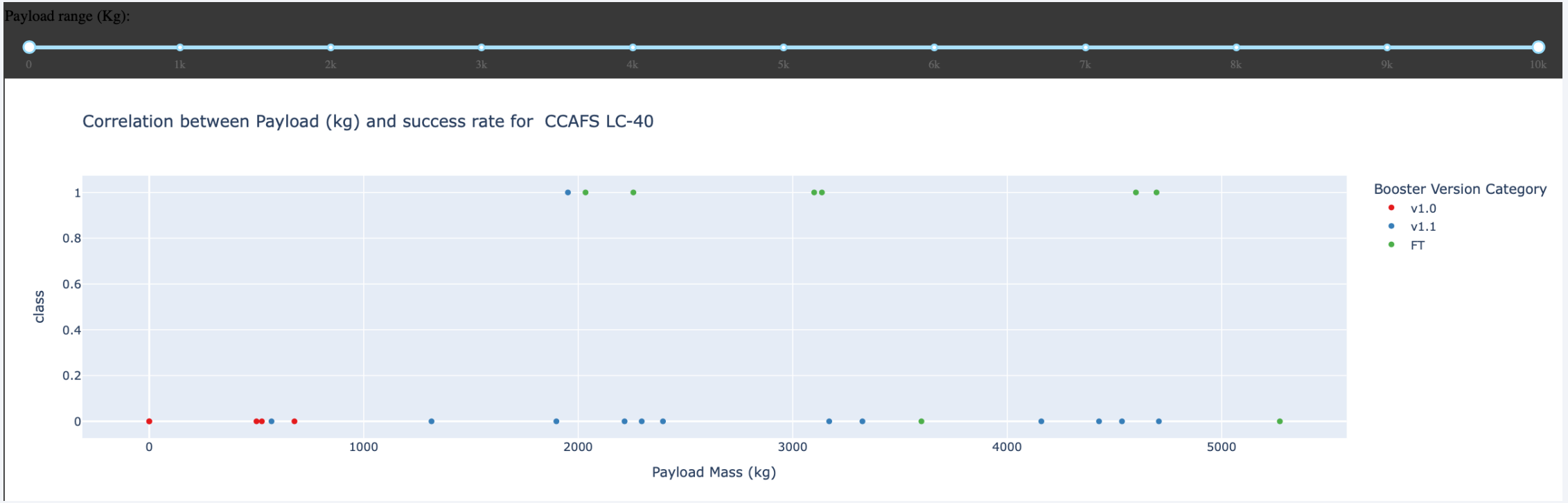
Launch success count for all the sites



Launch site with second highest launch success rate



Payload vs Launch outcome

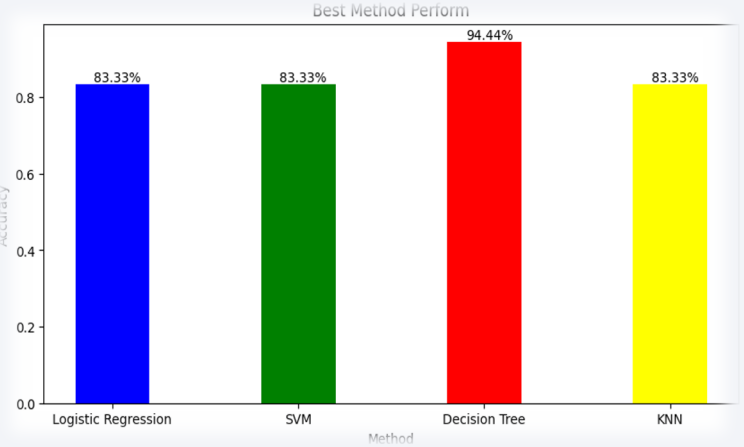


Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Find which model has the highest classification accuracy
- From the screenshot, we can conclude that not all models have similar accuracies and the decision tree model has the highest accuracy.

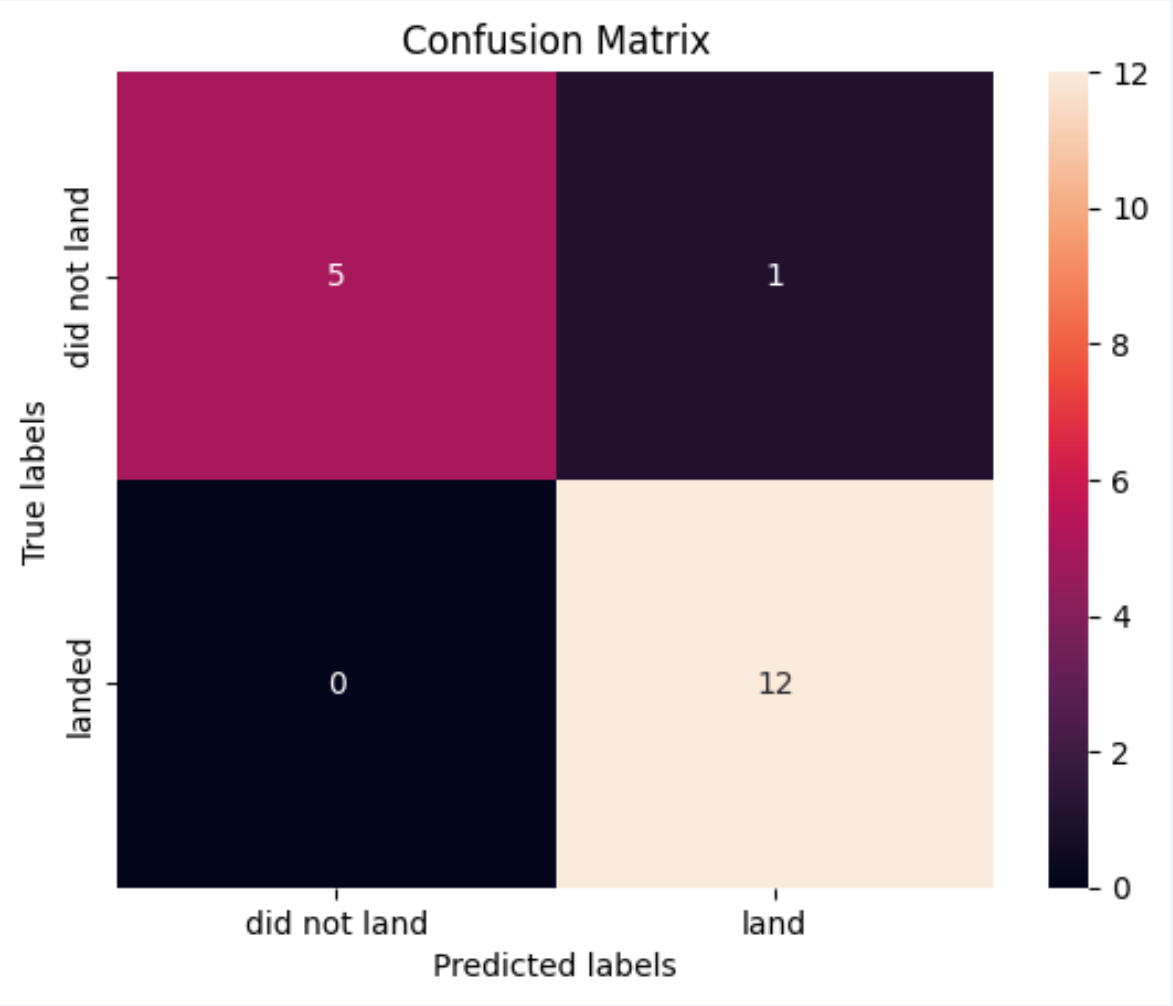



```
accuracy = []
Method = []

Method.append('Logistic Regression')
Method.append('SVM')
Method.append('Decision Tree')
Method.append('KNN')

accuracy.append(Score_log)
accuracy.append(Score_svm)
accuracy.append(score_tree)
accuracy.append(score_knn)

print (accuracy)
```

```
[0.8333333333333334, 0.8333333333333334, 0.9444444444444444, 0.8333333333333334]
['Logistic Regression', 'SVM', 'Decision Tree', 'KNN']
```

Confusion Matrix

- **A confusion matrix is a tool used to evaluate the performance of classification models.** It provides a clear picture of how well a model is able to correctly classify instances into different classes.
- **True Positive (TP):** 12 - The model correctly predicted 12 successful landings.
- **False Positive (FP):** 1 - The model incorrectly predicted 1 landing as successful when it was actually unsuccessful.
- **False Negative (FN):** 0 - The model did not incorrectly predict any successful landing as unsuccessful.
- **True Negative (TN):** 5 - The model correctly predicted 5 unsuccessful landings.
- The decision tree classifier performs well in predicting successful landings but struggles with predicting unsuccessful landings. It tends to misclassify unsuccessful landings as successful, resulting in a high number of false positives. This is a significant issue as it can lead to incorrect conclusions and decision-making.



Conclusions

- Different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E have a success rate of 77%.
- We can deduce that, as the flight number increases in each of the 3 launch sites, so does the success rate. For instance, the success rate for the VAFB SLC 4E launch site is 100% after Flight number 50. Both KSC LC 39A and CCAFS SLC 40 have 100% success rates after the 80th flight
- If you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).
- Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, with So orbit having the lowest success rate at ~50%. Orbit SO has a 0% success rate.

Conclusions

- LEO orbit Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit

- Finally the success rate since 2013 kept increasing till 2020.

Thank you!