IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Kristóf Bassa
2024.03.07.

Outline

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

Executive Summary

- Summary of methodologies:
  - Problem: creating predictions and visualisations for SPACE Y future lanches
  - Collecting data from SPACE X public API
  - Data wrangling with python (Exploratory Data Analysis)
  - Model development (models as: logistic regression, support vector machine,
    decision tree classifier, K nearest neighbours)
- Summary of all results:
  Best accuracy method is sigmoid

Introduction

- SpaceX, founded by Elon Musk in 2002, is a private aerospace manufacturer and space transportation company
- Dataset of Space X's lunch missions
- Analyzing the success rate of Space X missions
- Predicting the outcome of future Space Y launches
- Understanding relationships between importanat variables
- Evaluating the performance

Section 1

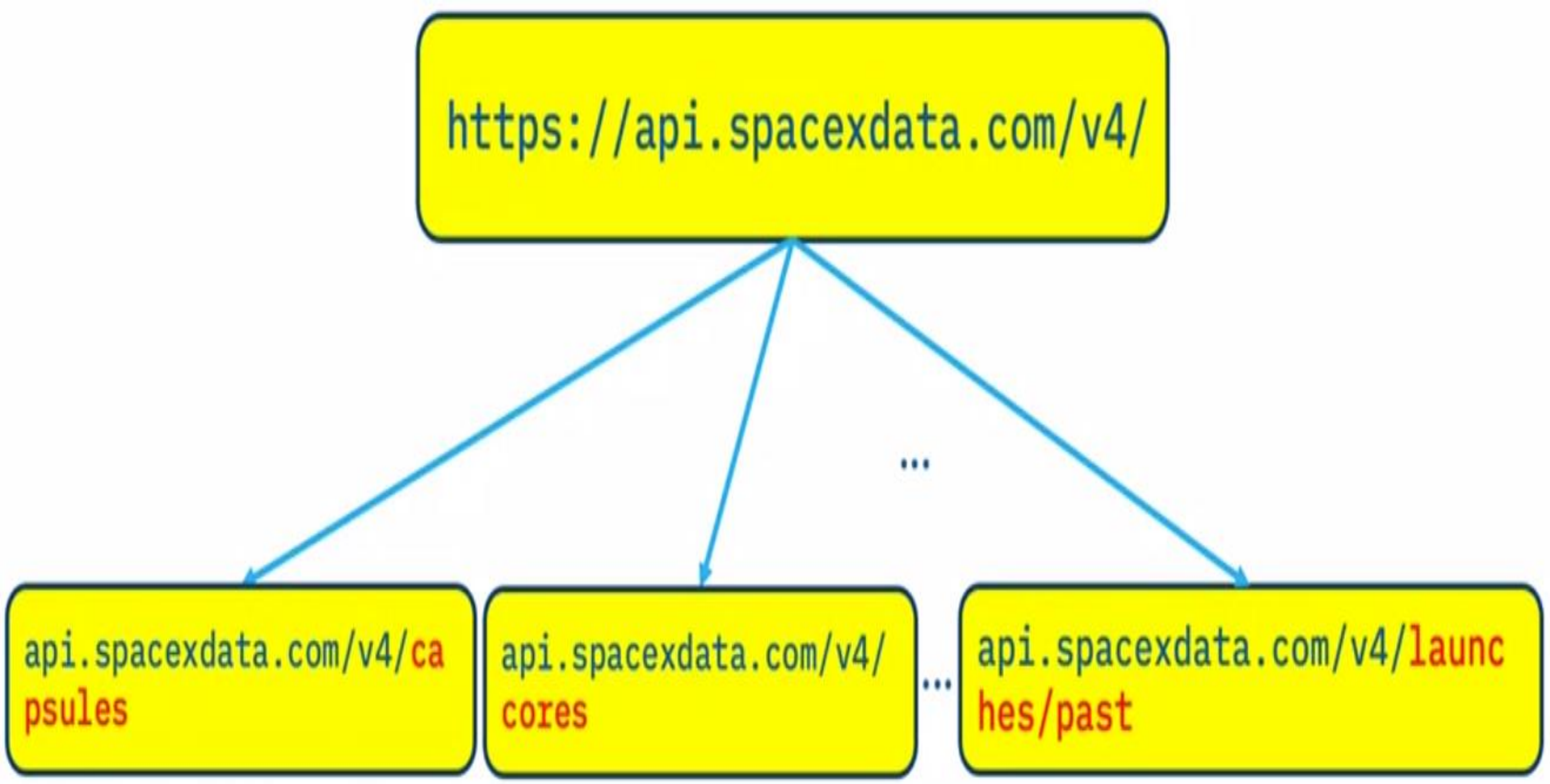
Methodology

Methodology

Executive Summary

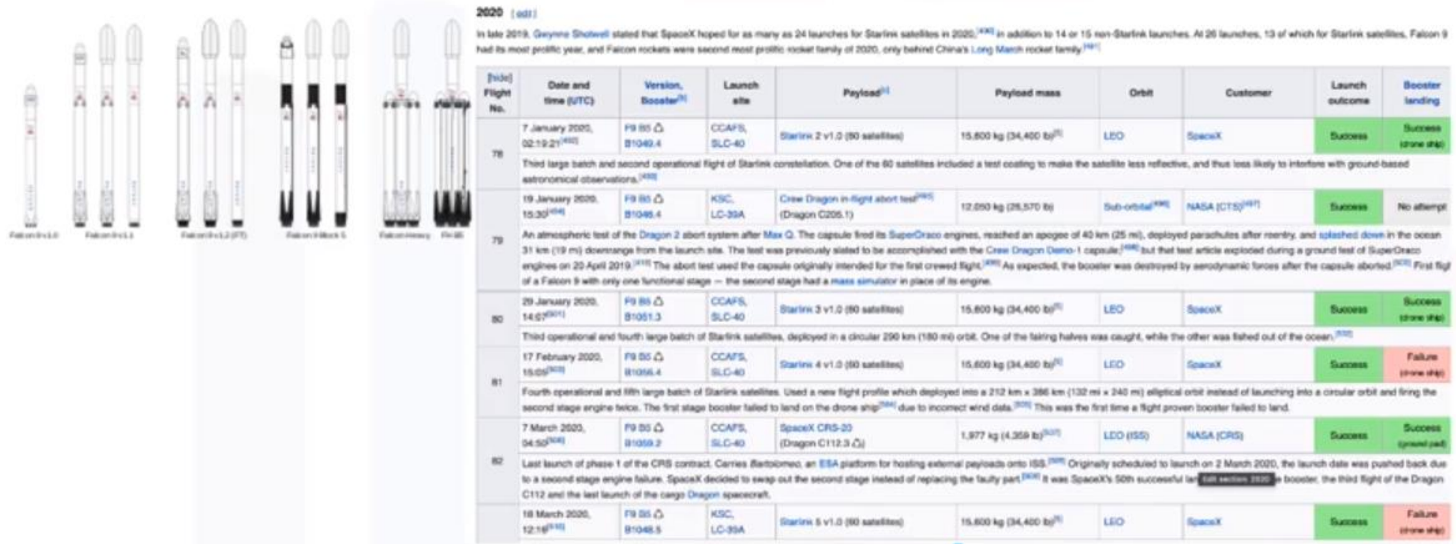
- Data collection methodology:
  - API – Space X REST API
- Perform data wrangling
  - Json_normalize function – generating a Pandas Dataframe
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logistic regression, Decision trees, Support Vector Machines (SVM), K-Nearest Neighbors (KNN)

Data Collection – SpaceX API



https://github.com/kristofbassa/spaceY.git

Data Collection - Scraping



Web scraping with BeautifulSoup

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1A	167.743129	9.047721
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2A	167.743129	9.047721
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2C	167.743129	9.047721
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3C	167.743129	9.047721
4	6	2010-06-04	Falcon 9	NaN	LEO	CCAFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857

https://github.com/kristofbassa/spaceY.git

Data Wrangling

- Data Preprocessing:

- Split the data into training and testing sets to evaluate model performance.
- Perform feature selection to identify the most relevant features for modeling.
- Address class imbalance issues if present by oversampling, undersampling, or using techniques like SMOTE.

EDA with Data Visualization

- Scatter Plot (Flight Number vs. Payload Mass): Used to visualize the relationship between the flight number and payload mass, aiming to understand any trends or patterns over time.
- Scatter Plot with Hue (Flight Number vs. Launch Site with Outcome as Hue): Utilized to explore the relationship between the flight number and launch site while incorporating the outcome of the launches as a hue, helping to identify any correlation between launch sites and mission outcomes.
- Scatter Plot with Hue (Payload Mass vs. Launch Site with Outcome as Hue): Employed to investigate the relationship between payload mass and launch site while considering the outcome of the launches as a hue, aiming to discern any impact of payload mass on mission success across different launch sites.
- Bar Chart (Success Rate of Each Orbit Type): Created to visualize the success rate of each orbit type, facilitating comparison and identification of trends in mission success across different orbit types.

EDA with SQL

- Queries:
- create table SPACEXTABLE as select * from SPACEXTBL where Date is not null
- select launch_site from SPACEXTABLE group by launch_site
- select * from SPACEXTABLE where launch_site like 'CCA%' limit 5
- select sum(PAYLOAD_MASS__KG_) as 'Total payload mass' from SPACEXTABLE where customer = 'NASA (CRS)'
- select avg(PAYLOAD_MASS__KG_) as 'Total payload mass' from SPACEXTABLE where booster_version like 'F9 v1.1%'
- select
- sum(case when Mission_Outcome = 'Success' then 1 else 0 end) as success, sum(case when Mission_Outcome = 'Failure' then 1 else 0 end) as failure from SPACEXTABLE

Build an Interactive Map with Folium

- Markers: Added markers to indicate the location of launch sites on the map. Each marker represents a specific launch site used by SpaceX.
- Circles: Added circles to visualize the range of landing pads associated with each launch site. The circles represent the landing area where the rocket's first stage attempts to land after launch.
- Lines: Added lines to connect launch sites with their corresponding landing pads, providing a visual representation of the launch trajectory and landing area.
- The purpose of adding these objects is to provide a comprehensive view of SpaceX's launch operations and infrastructure. By visualizing the launch sites, landing pads, and launch trajectories on the map, stakeholders can gain insights into the geographic distribution of SpaceX's activities and understand the proximity of launch sites to potential landing areas.

Build a Dashboard with Plotly Dash

- Line Chart: Showing the trend of SpaceX mission outcomes over time, allowing users to visualize the success rate of missions.
- Bar Chart: Displaying the distribution of mission outcomes (success/failure) across different launch sites, providing insights into the performance of each site.
- Scatter Plot: Illustrating the relationship between payload mass and mission outcome, helping users understand how payload mass affects mission success.
- Pie Chart: Presenting the distribution of mission outcomes for different orbit types, allowing users to compare success rates across orbits.

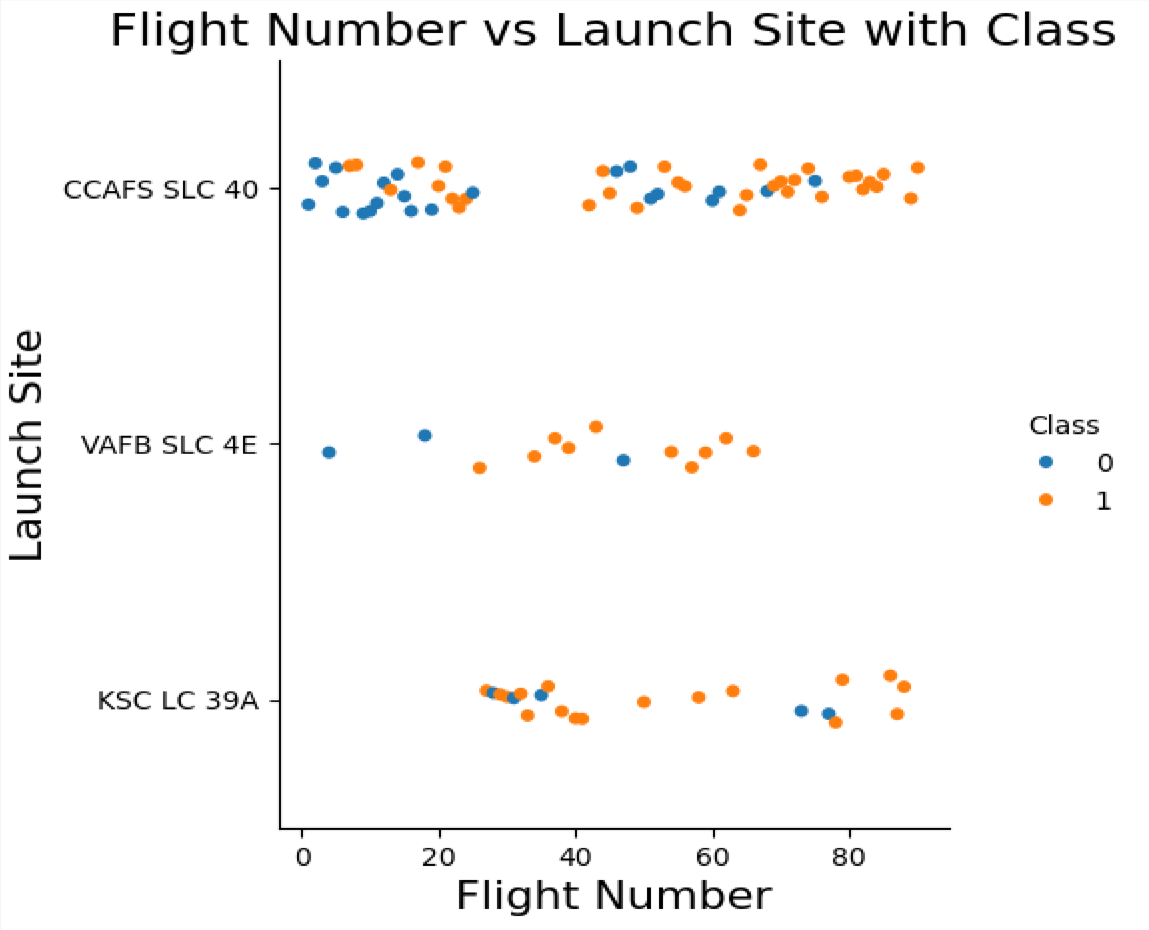
Predictive Analysis (Classification)

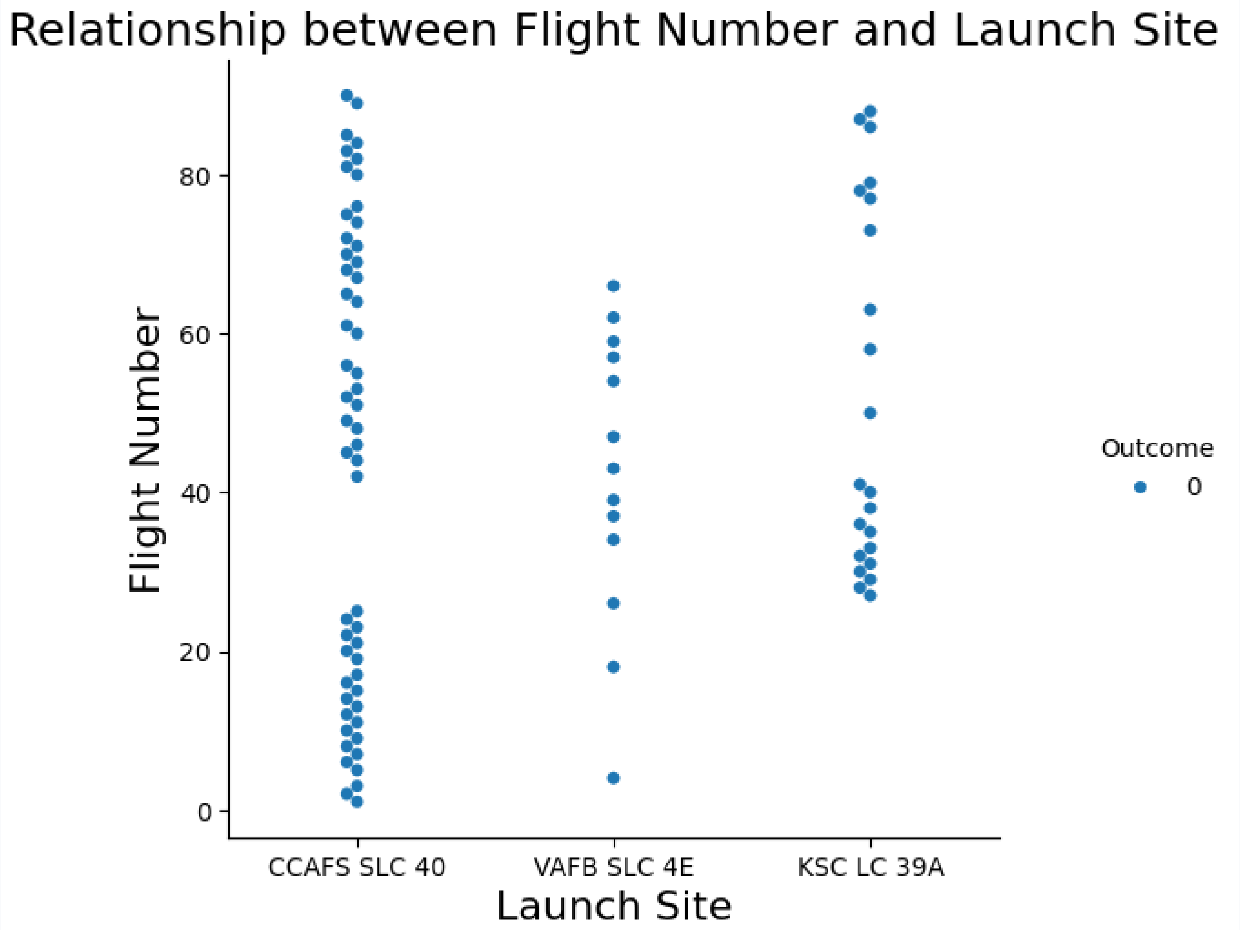
- Summarize how you built, evaluated, improved, and found the best performing classification model
- You need present your model development process using key phrases and flowchart
- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

Section 2

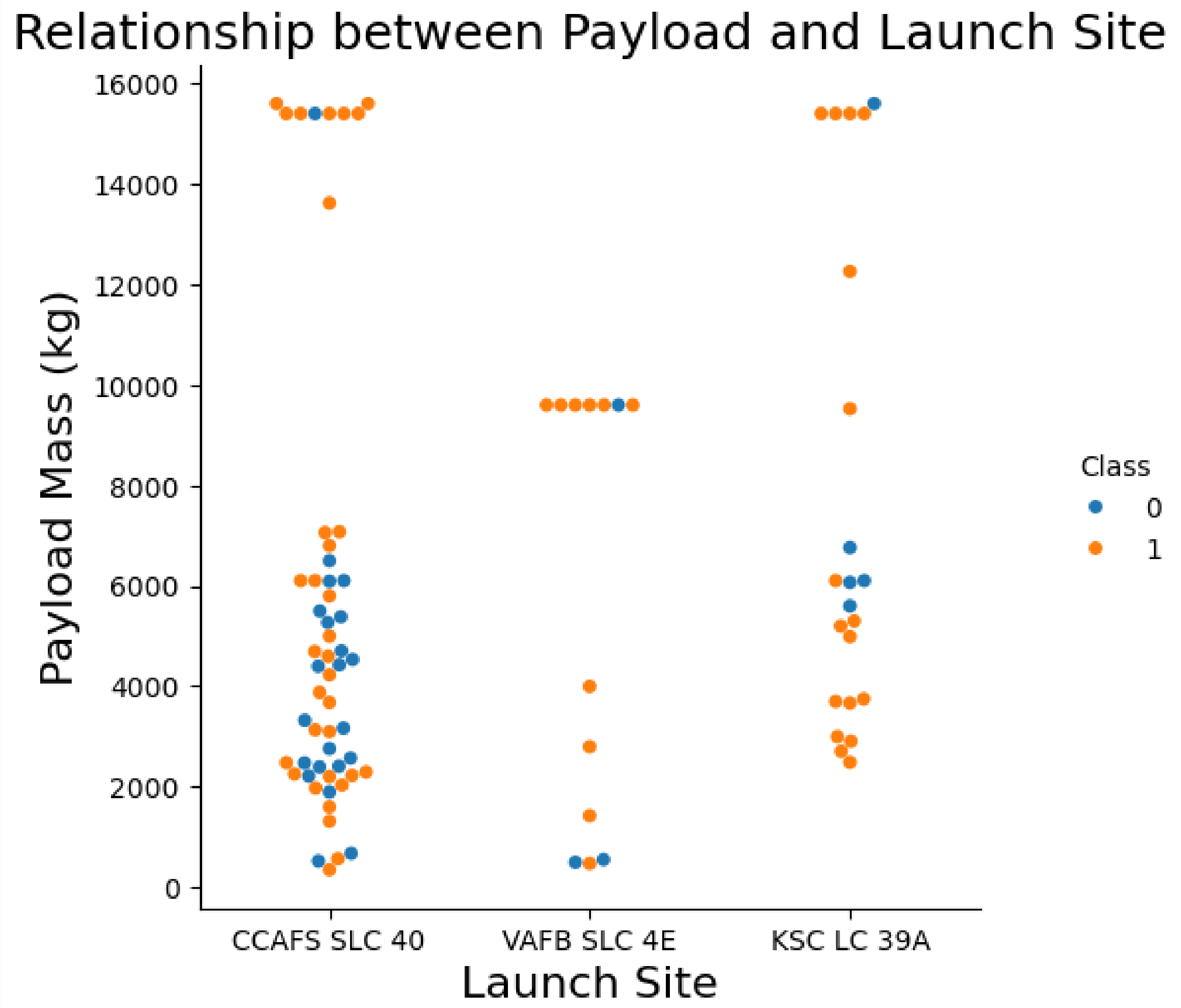
Insights drawn from EDA

Flight Number vs. Launch Site

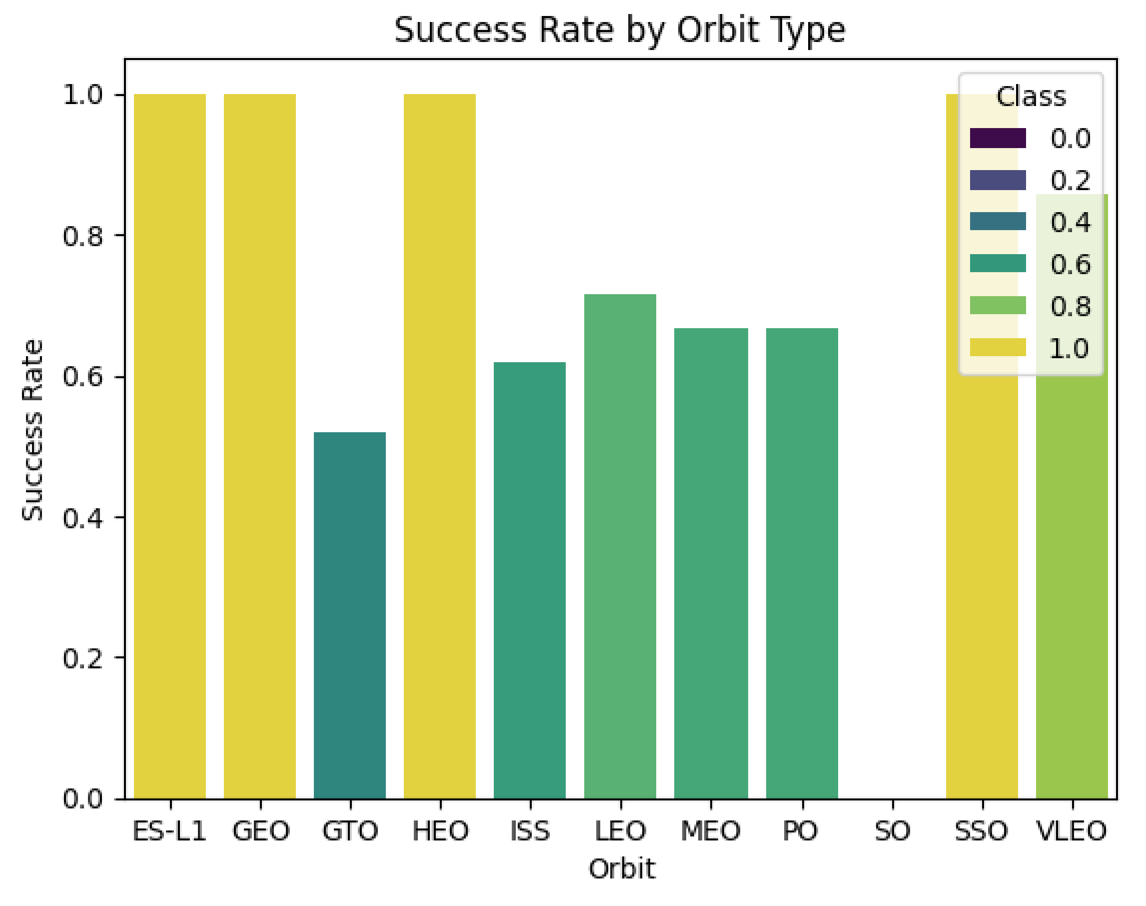


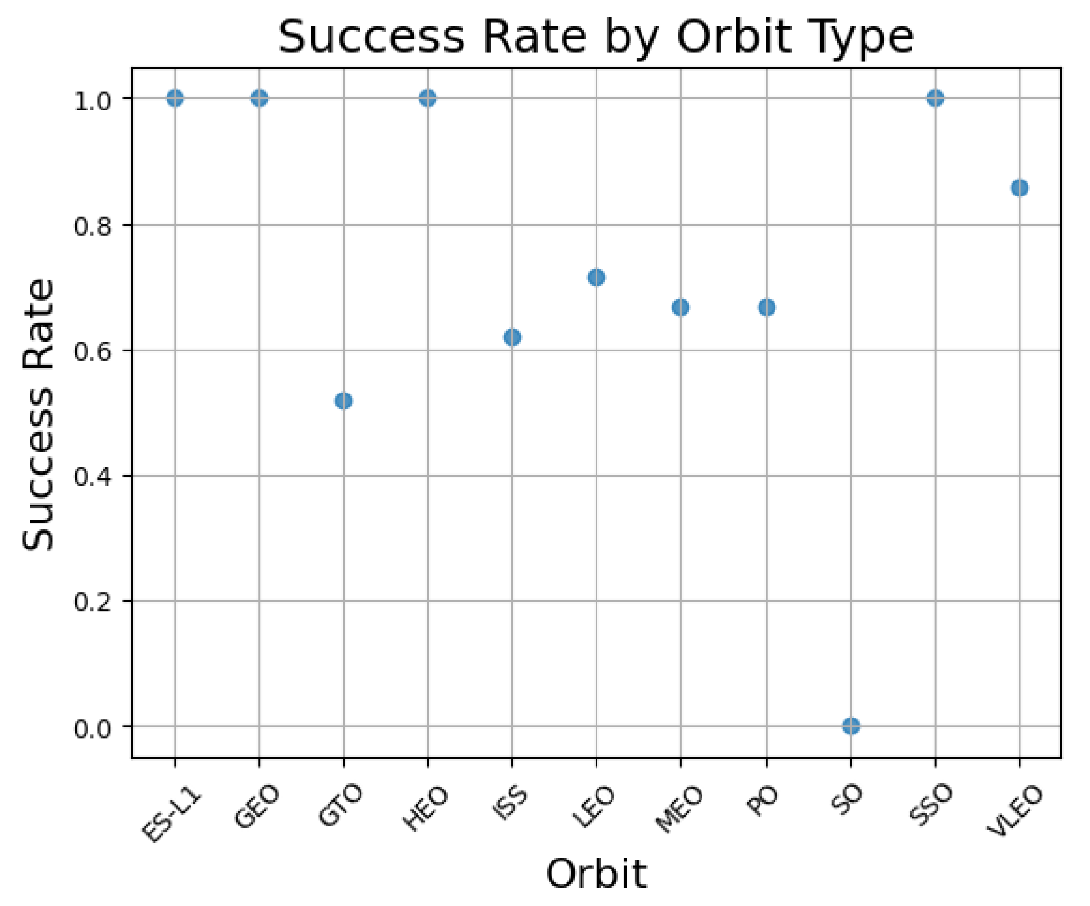


Payload vs. Launch Site

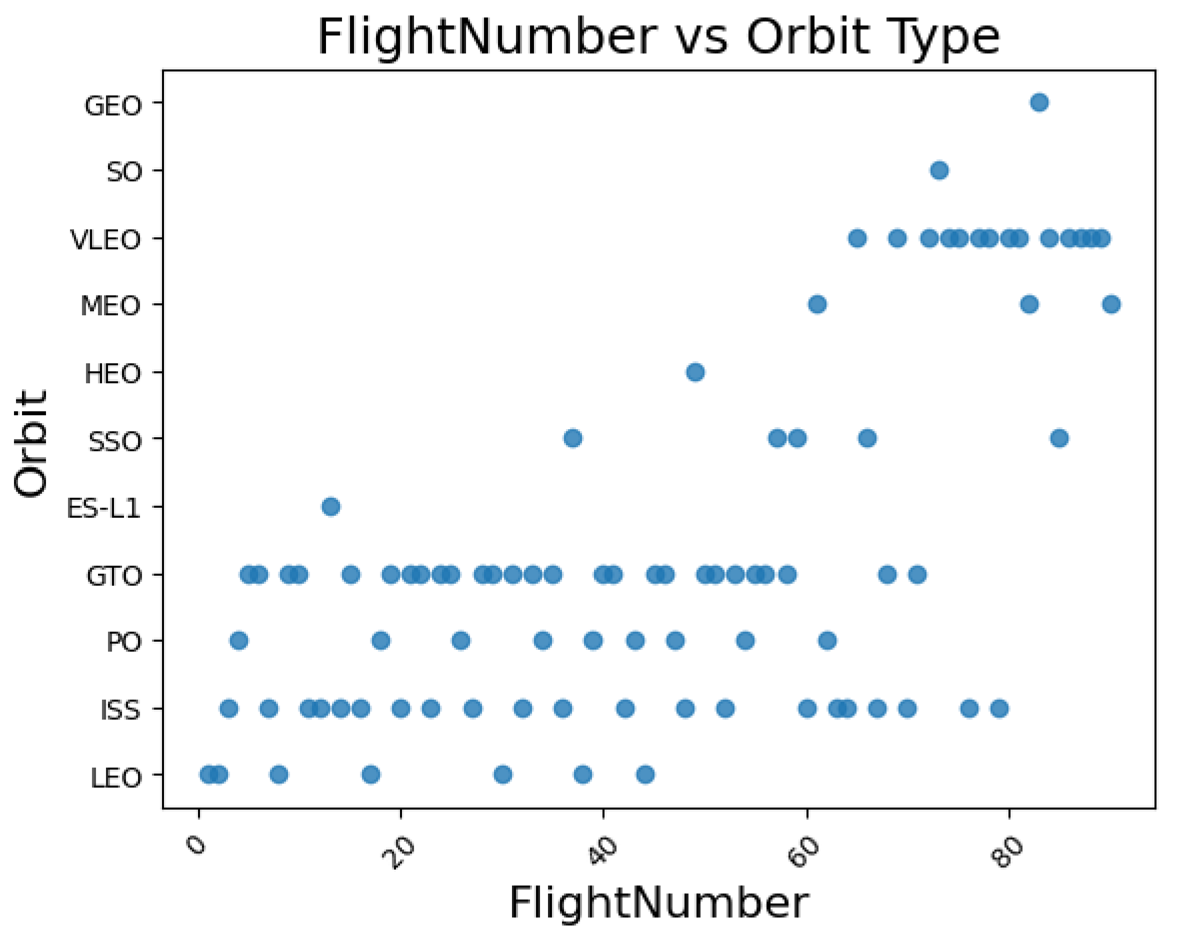


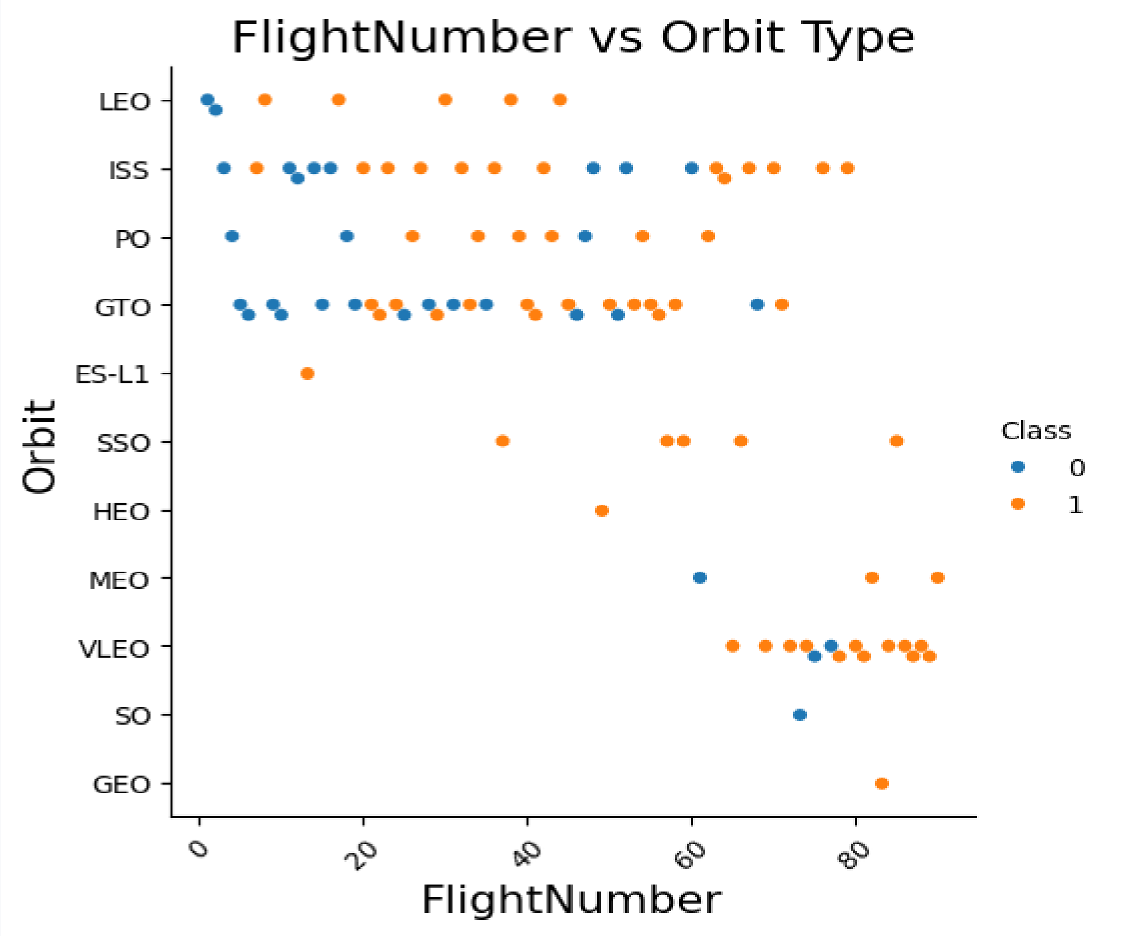
Success Rate vs. Orbit Type



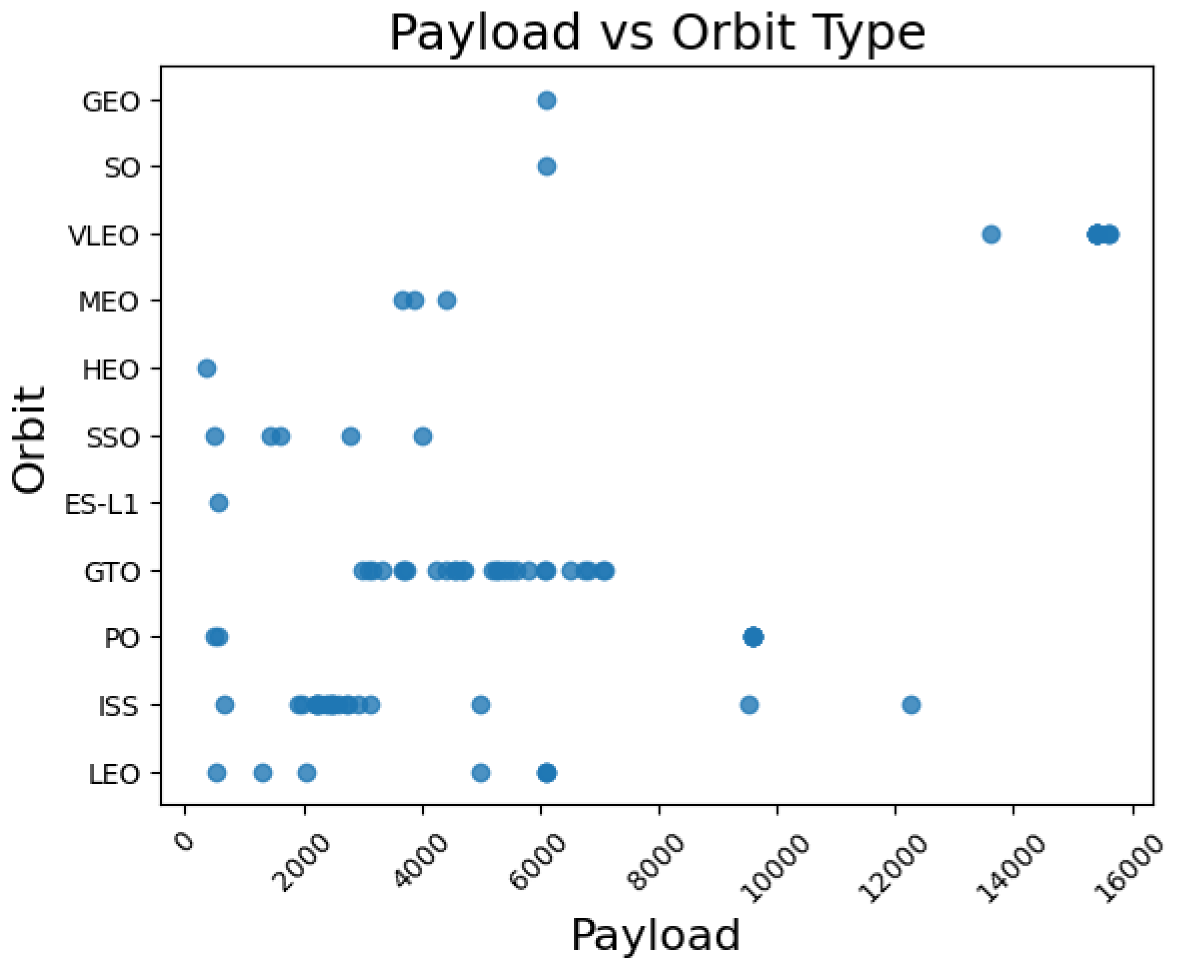


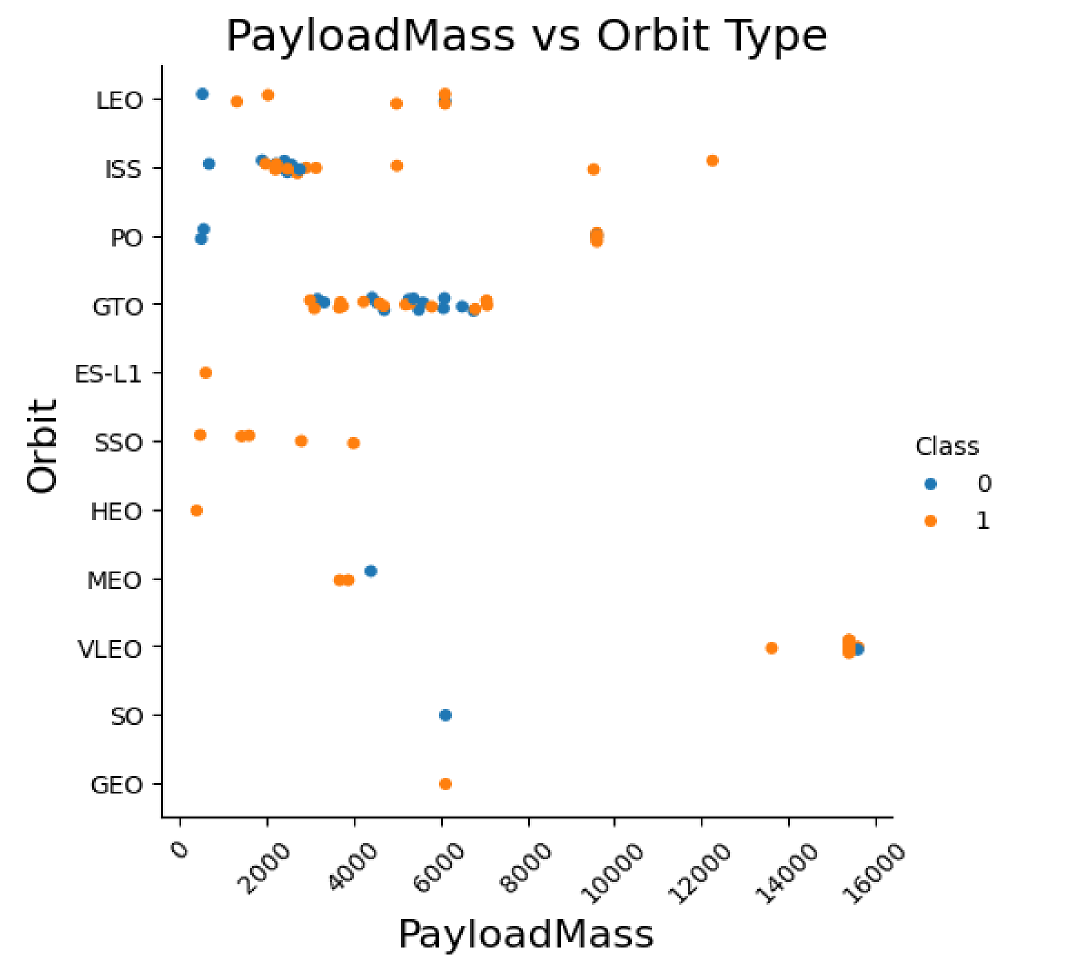
Flight Number vs. Orbit Type



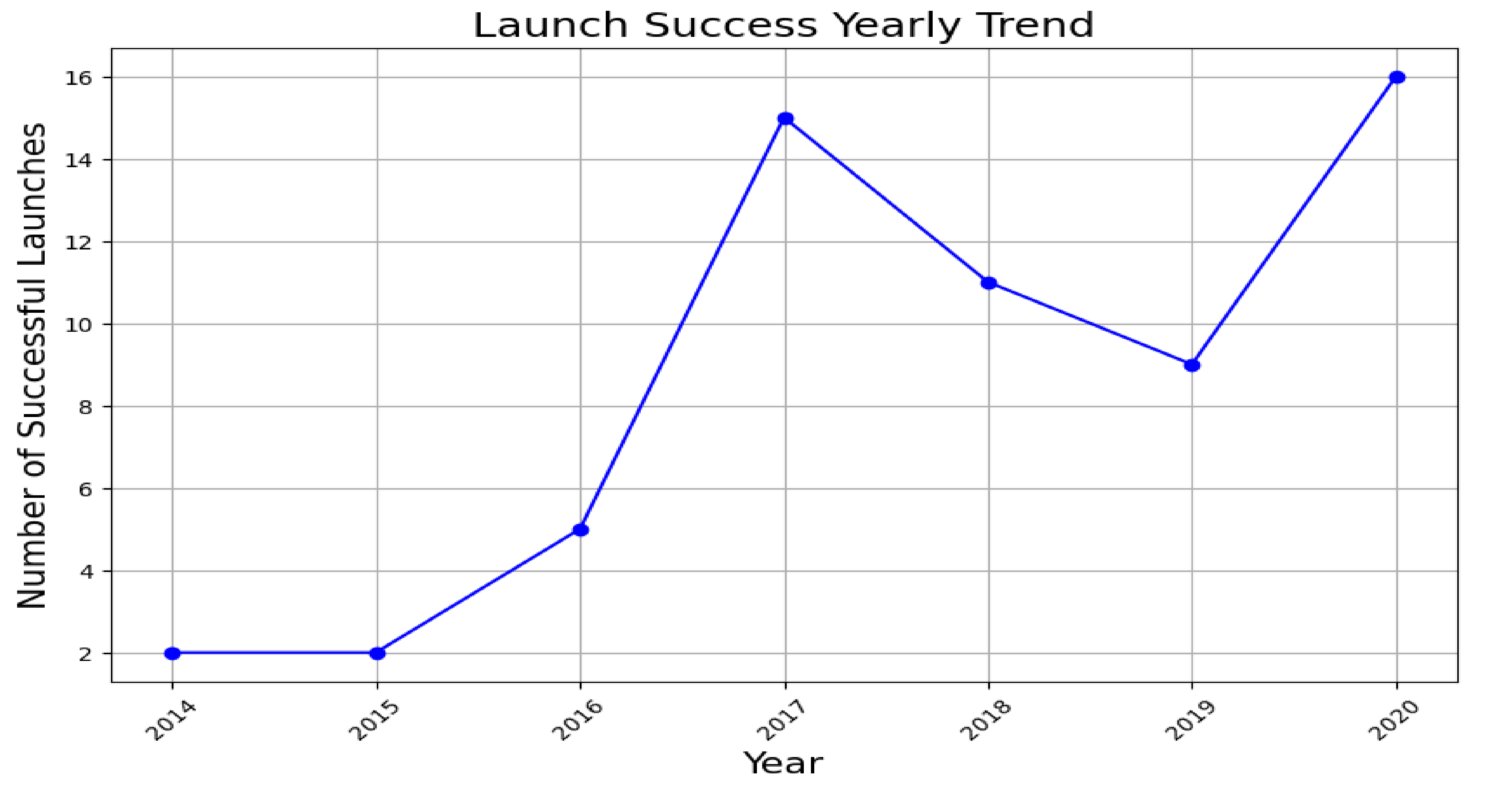


Payload vs. Orbit Type





Launch Success Yearly Trend



All Launch Site Names

```
%sql select launch_site from SPACEXTABLE group by launch_site
```

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

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

Launch Site Names Begin with 'CCA'

```
%sql select * from SPACEXTABLE 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

```
%sql select sum(PAYLOAD_MASS__KG_) as 'Total payload mass' from SPACEXTABLE where customer = 'NASA (CRS)'
```

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

Total payload mass
45596

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as 'Total payload mass' from SPACEXTABLE where booster_version like 'F9 v1.1%'
```

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

Total payload mass
2534.6666666666665

First Successful Ground Landing Date

```
%sql select min(date) from SPACEXTABLE where Landing_Outcome like 'Success (ground%'
```

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

min(date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select Booster_Version from SPACEXTABLE where Landing_Outcome like 'Success (drone%' and PAYLOAD_MASS__KG_ between 4000 and 6000 group by Booster_Version
```

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

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

Total Number of Successful and Failure Mission Outcomes

```
%%sql
select
sum(case when Mission_Outcome = 'Success' then 1 else 0 end) as success, sum(case when Mission_Outcome = 'Failure' then 1 else 0 end) as failure from SPACEXTABLE
```

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

success	failure
98	0

Boosters Carried Maximum Payload

```
%%sql
select booster_version, PAYLOAD_MASS__KG_ from SPACEXTABLE where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTABLE) group by booster_version, PAYLOAD_MASS__KG_
```

```
 * 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
substr(Date, 6,2) month,
Landing_Outcome,
booster_version,
launch_site
from SPACEXTABLE
where Landing_Outcome = 'Failure (drone ship)' and date like '2015%'
```

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

month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

```
%sql select Landing_Outcome, count(*) from SPACEXTABLE where date between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by 2 desc
```

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

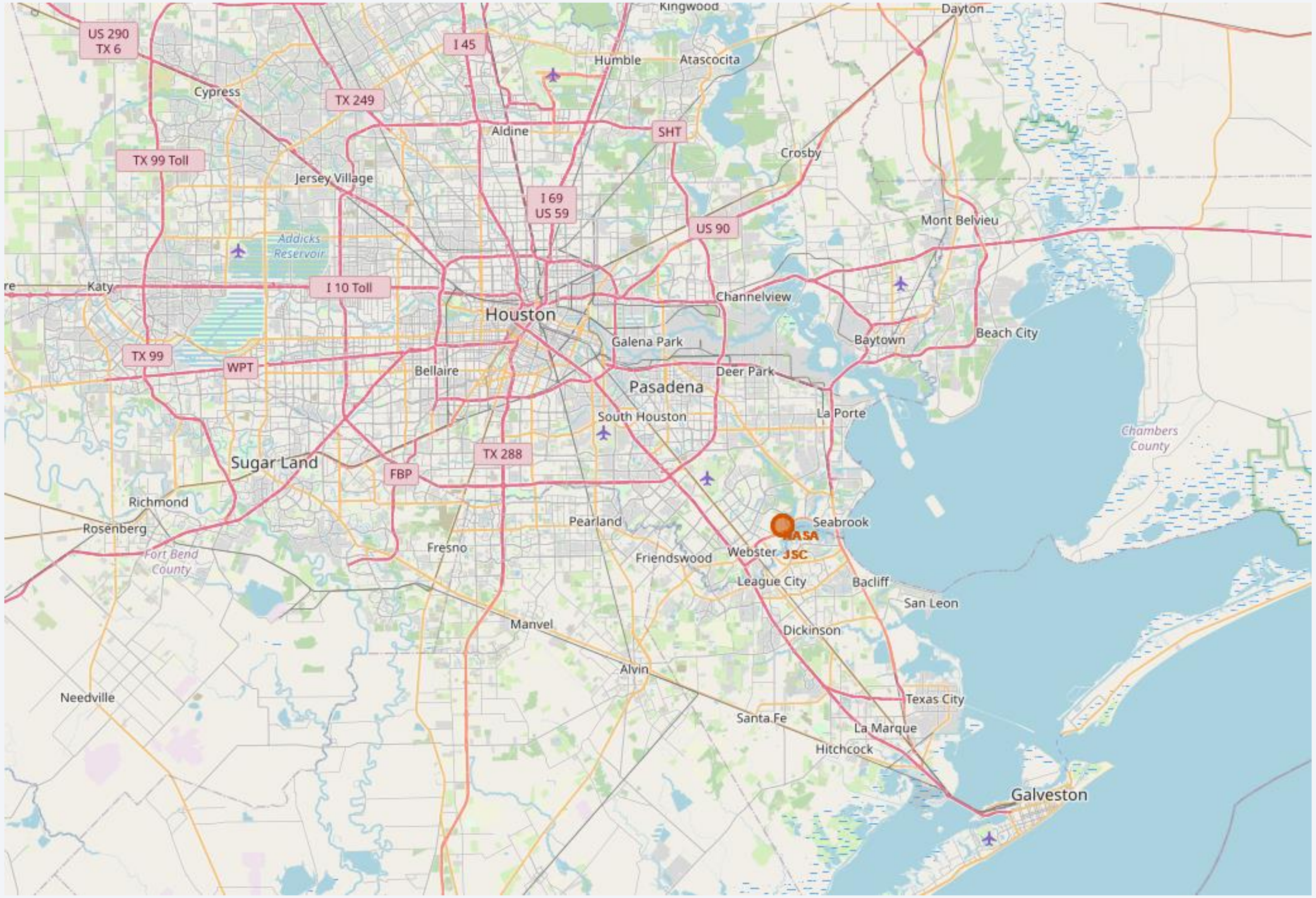
Landing_Outcome	count(*)
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

Launch Sites Locations Analysis with Folium

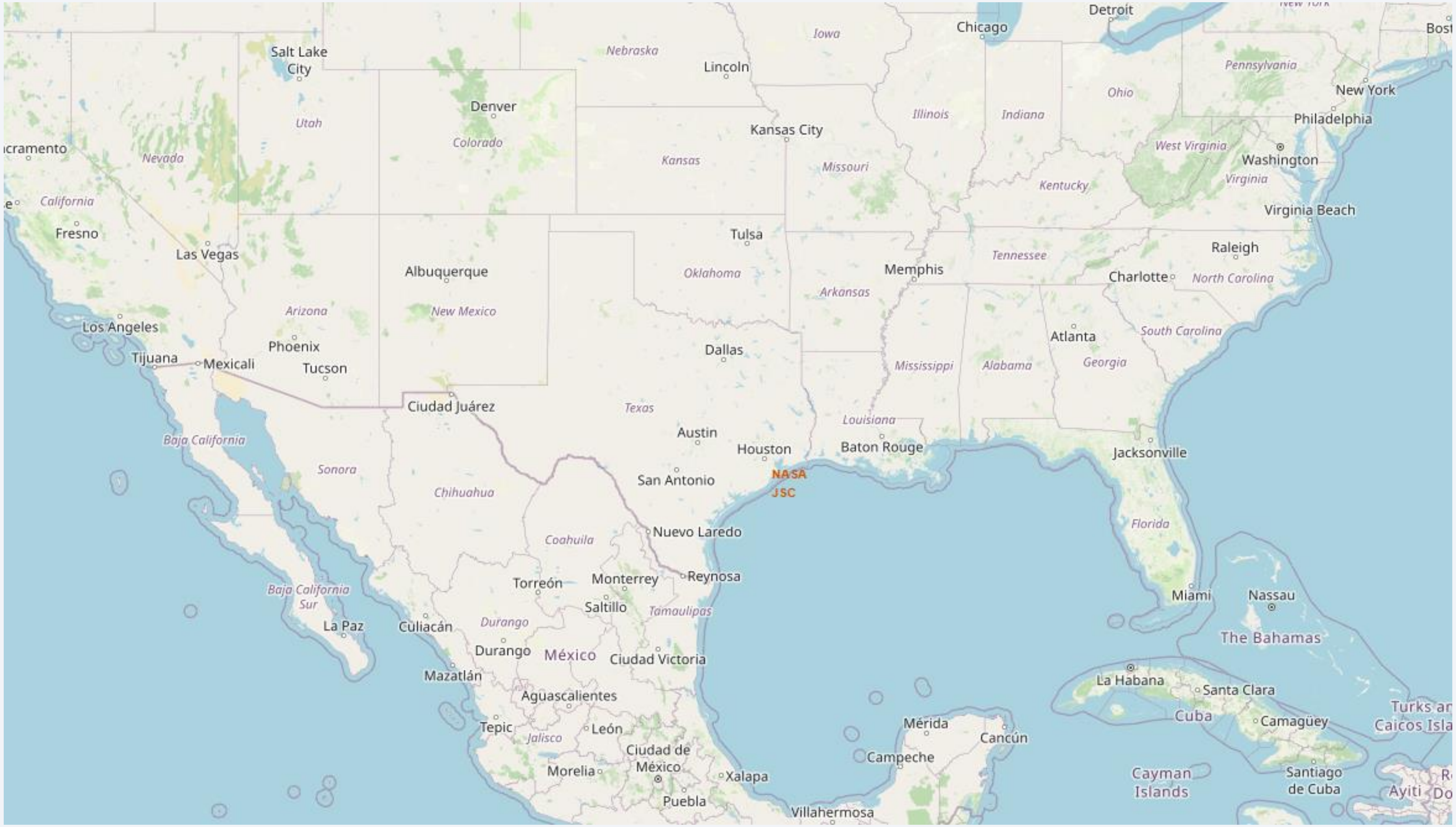
Launch sites are marked with airplanes



<Folium Map Screenshot 2>

- Replace <Folium map screenshot 2> title with an appropriate title
- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map
- Explain the important elements and findings on the screenshot

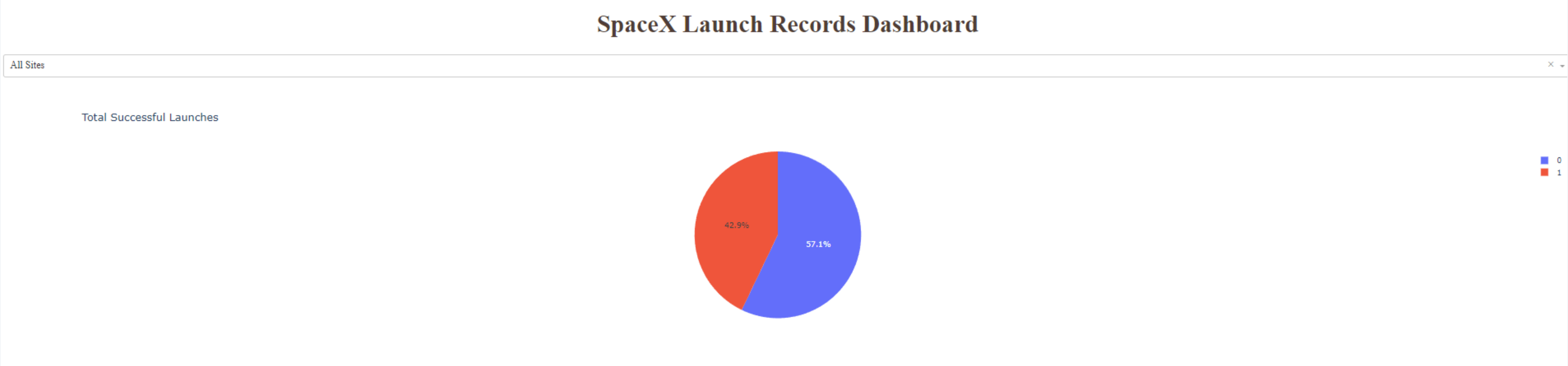
<Folium Map Screenshot 3>



Section 4

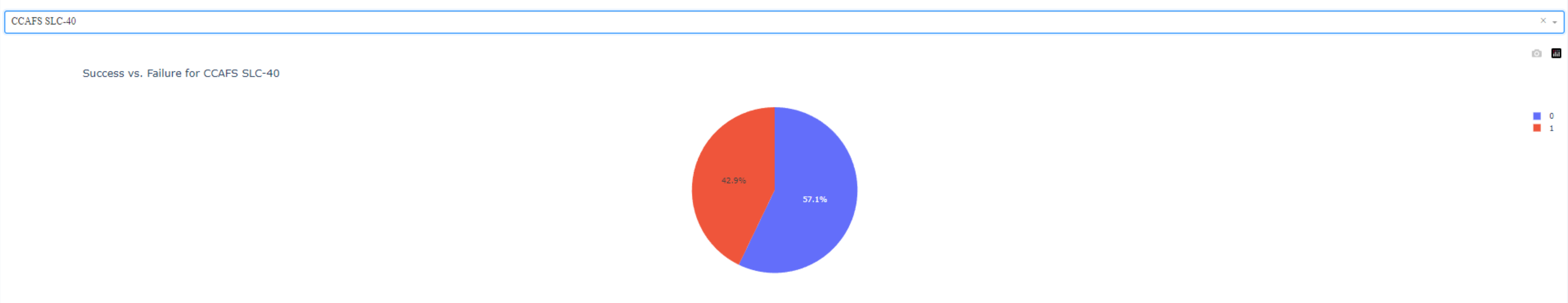
Build a Dashboard with Plotly Dash

Dashboard Application with Plotly Dash



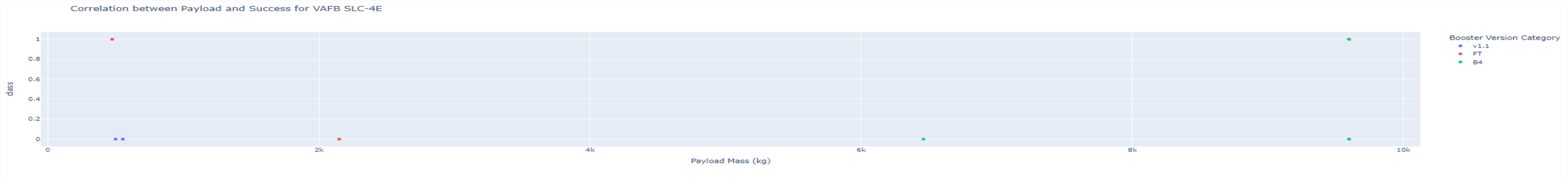
In 57.1% launches finished with a bad outcome like the booster did not land ☹

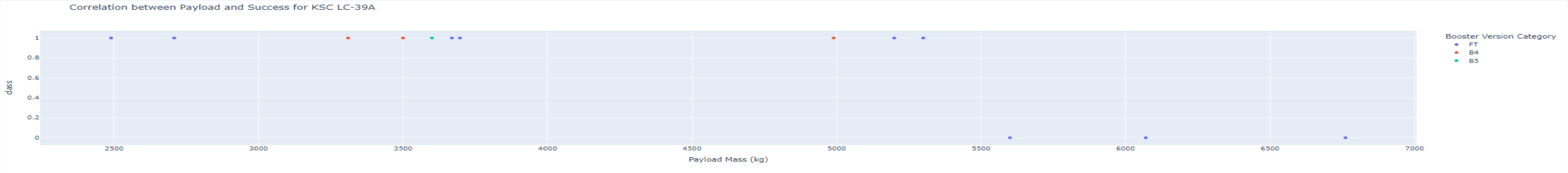
Piechart of the most successful ratio

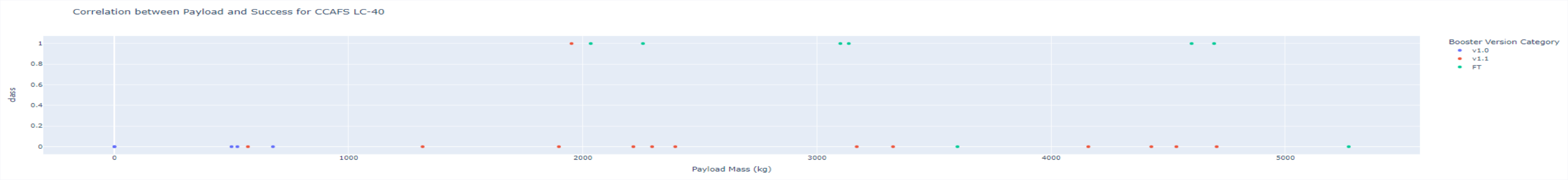


- The most successful site was CCAFS SLC-40

<Dashboard Screenshot 3>



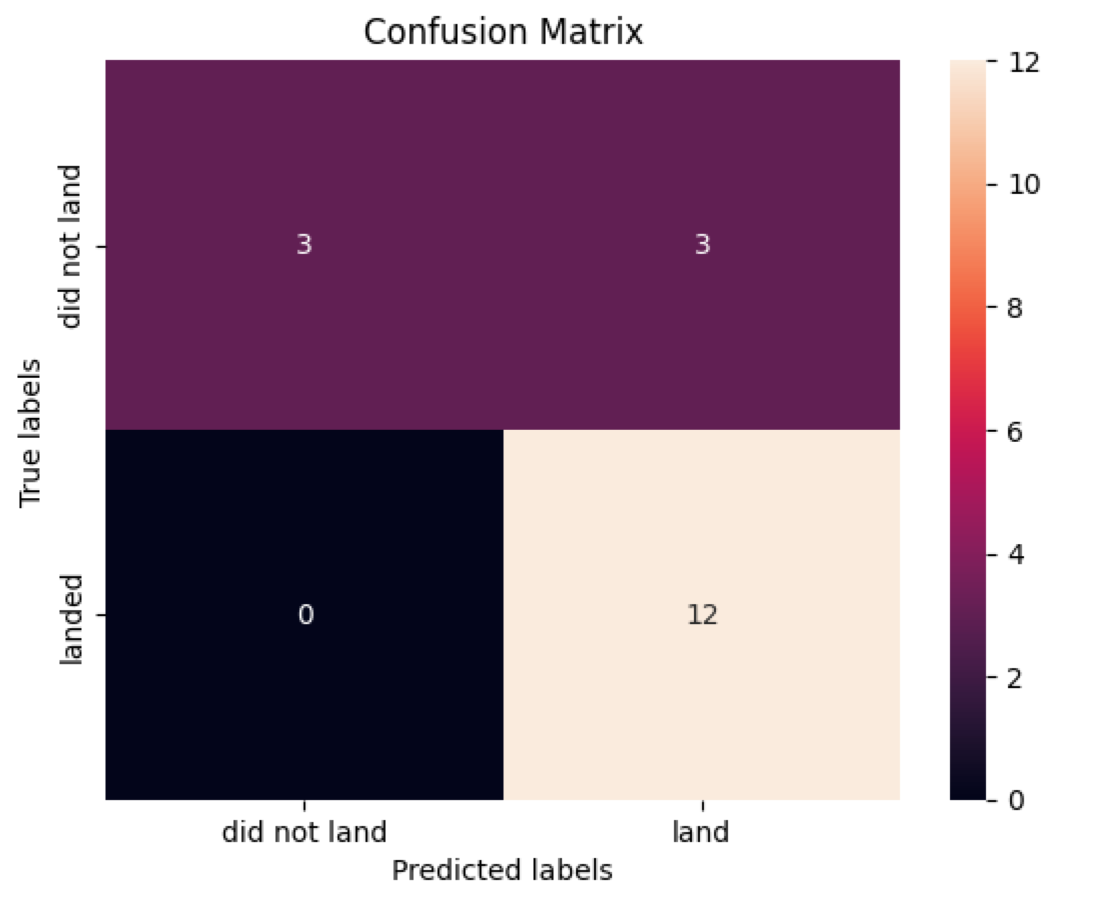




Section 5

Predictive Analysis (Classification)

Confusion Matrix



The best performing method is Sigmoid, because:

```
tree_accuracy = tree_cv.score(X_test, Y_test)
svm_accuracy = svm_cv.score(X_test, Y_test)
knn_accuracy = knn_cv.score(X_test, Y_test)

# Accuracies
print("Accuracy for Decision Tree:", tree_accuracy)
print("Accuracy for Support Vector Machine:", svm_accuracy)
print("Accuracy for K-Nearest Neighbors:", knn_accuracy)

# Find the method with the highest accuracy
best_method = max(tree_accuracy, svm_accuracy, knn_accuracy)

if best_method == tree_accuracy:
    print("Decision Tree performs best.")
elif best_method == svm_accuracy:
    print("Support Vector Machine performs best.")
else:
    print("K-Nearest Neighbors performs best.")


best_params_svm = svm_cv.best_params_

# Extract the best kernel
best_kernel = best_params_svm['kernel']

print("Best kernel for Support Vector Machines:", best_kernel)
```

```
Accuracy for Decision Tree: 0.8888888888888888
Accuracy for Support Vector Machine: 0.8333333333333334
Accuracy for K-Nearest Neighbors: 0.8333333333333334
Decision Tree performs best.
Best kernel for Support Vector Machines: sigmoid
```

Thank you!