IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Cissey BELLO
28/11/2025

Outline

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

Executive Summary

- Data collection methodology:
  - Data was collected Using the SpaceX API and Web Scraping Launch Records from Wikipedia
- Perform data wrangling
  - Data wrangling was performed by loading the raw dataset, identifying missing values, categorizing feature and analyzing outcome patterns
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Normalize and Split Data, Tune the parameters to fine the best (Grid search), Train the model and Evaluation (Confusion Matrix)

Introduction

- **Project background and context**

The commercial space industry is rapidly evolving, with private companies making space travel and satellite deployment more accessible and affordable. Among the leading innovators is SpaceX, whose Falcon 9 rocket drastically lowers launch costs through its ability to reuse the first-stage booster. Since this booster represents the most expensive part of the rocket, its recovery significantly impacts the economics of a mission.

- **Problems you want to find answers**
  - How does SpaceX achieve lower-cost launches?
  - What factors influence first-stage landing success?
  - Can we predict whether the Falcon 9 first stage will land?
  - How can predicted landing outcomes help estimate launch costs?

Section 1

Methodology

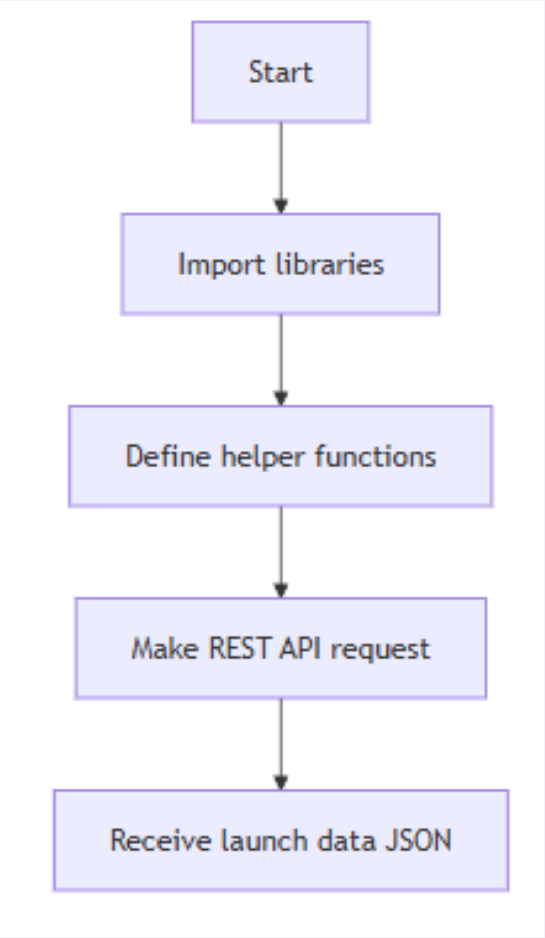
Methodology

Executive Summary

- Data collection methodology:
  - Data was collected Using the SpaceX API and Web Scraping Launch Records from Wikipedia
- Perform data wrangling
  - Data wrangling was performed by loading the raw dataset, identifying missing values, categorizing feature and analyzing outcome patterns
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Normalize and Split Data, Tune the parameters to fine the best (Grid search), Train the model and Evaluation (Confusion Matrix)

Data Collection – SpaceX API

To see SpaceX API notebook (GitHub) click here Link



Set up environment for REST API communication and data processing

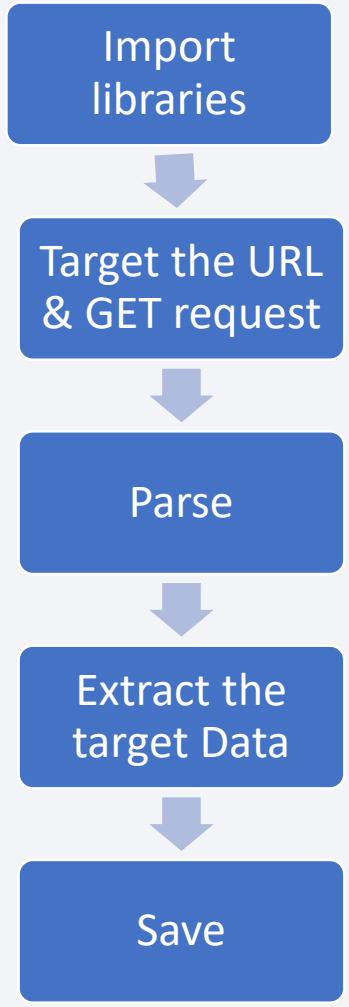
Prepare functions for extracting detailed rocket, payload, launch site, and core information via REST calls

Request bulk launch records from SpaceX API

Convert API response to pandas DataFrame for data manipulation

Data Collection - Scraping

To see Web scraping notebook (GitHub) click here Link



Set up environment for scraping communication and data processing

Target the Wikipedia URL and Make HTTP GET request (simulate browser)

Parse returned HTML for tables

Extract headers and initialize storage, Iterate over rows, clean and parse values

Store results and convert to DataFrame

Data Wrangling

To see Web Wrangling notebook (GitHub) click here Link

Summary of Data Wrangling Process (SpaceX Falcon 9 Dataset)

- **Installed and imported required libraries** (pandas, numpy) for data processing.
- **Loaded the raw dataset** using Pandas and previewed its structure.
- **Checked for and calculated missing values** in every feature to assess data completeness.
- **Classified columns** into categorical and numerical types for targeted analysis.
- **Explored launch sites and orbit types** using value counts to understand their distributions.
- **Examined landing outcomes,** categorizing them as "successful" or "unsuccessful."
- **Created a binary label** (Class) for each launch: 1 for successful landings and 0 for failures.

EDA with Data Visualization

To see EDA with Data Visualization notebook (GitHub) click here Link

- FlightNumber vs PayloadMass (sns.catplot, hue=Class)
  - To visualize how payload sizes vary over the launch timeline and whether success correlates with flight experience or payload mass.
- FlightNumber vs LaunchSite (sns.catplot, hue=Class)
  - To see launch-site activity over time and compare success rates per site across launches.
- PayloadMass vs LaunchSite (sns.catplot, hue=Class)
  - To inspect whether certain sites are used for heavier/lighter payloads and whether payload mass relates to success per site.
- Success rate by Orbit (sns.barplot on grouped mean Class)
  - To quantify and compare landing success rates across orbit types (aggregated summary rather than point-level plotting).
- FlightNumber vs Orbit (sns.catplot, hue=Class)
  - To check whether the relationship between flight experience (FlightNumber) and success differs by orbit type.
- PayloadMass vs Orbit (sns.catplot, hue=Class)
  - To explore whether payload mass distributions differ by orbit and whether heavier/lighter payloads show different success patterns per orbit.

EDA with SQL

To see EDA with SQL notebook (GitHub) click here Link

- Dropped and recreated SPACEXTABLE from SPACEXTBL excluding rows with NULL Date.
- Queried DISTINCT Launch_Site → CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, CCAFS SLC-40.
- Selected 5 rows WHERE Launch_Site LIKE 'CCA%' (sample records from CCA* sites).
- SUM(PAYLOAD_MASS__KG_) WHERE Customer='NASA (CRS)' → 45596.
- AVG(PAYLOAD_MASS__KG_) WHERE Booster_Version='NASA (CRS)' → 2928.4 (possible filtering mistake: Customer used in place of booster name).
- MIN(Date) WHERE Landing_Outcome LIKE '%ground pad%' → 2015-12-22.
- SELECT Booster_Version WHERE Landing_Outcome='Success (drone ship)' AND payload BETWEEN 4000 AND 6000 → four boosters (e.g., F9 FT B1022, B1026, B1021.2, B1031.2).
- COUNT(*) GROUP BY Mission_Outcome → mission counts (e.g., Success 98, Failure (in flight) 1, plus variants).
- SELECT Booster_Version WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(...)) → all boosters that carried the maximum payload.

Build an Interactive Map with Folium

To see Interactive Map with Folium notebook (GitHub) click here Link

- folium.Map
  - Centered on NASA Johnson Space Center (29.5596849, -95.0830972) with a chosen zoom level.
  - Purpose: provides the basemap and initial geographic context so all markers/circles are displayed in relation to a familiar reference point.
- Circle + Marker + Popup at NASA JSC
  - folium.Circle at the NASA coordinate (radius=1000, color='#d35400', fill=True).
  - folium.map.Marker with a DivIcon text label "NASA JSC".
  - Popup attached to the circle showing the site name.
  - Purpose: highlight the reference location, make it easily clickable (popup) and labeled on the map.
- For each launch site (looping over launch_sites_df) I added:
  - folium.Circle(site_coord, radius=1000, color='#ffff00', fill=True) with a folium.Popup(site_name) To give a visible area around each launch site

Build a Dashboard with Plotly Dash

To see Dashboard with Plotly Dash (GitHub) click here Link

- **Dropdown (site):** lets you choose All or a specific launch site to focus the dashboard and drive the charts for comparison or drill-down.
- **Pie chart**: for "All" shows total successes by site; for a specific site shows success vs failure — provides a quick high-level distribution.
- **Payload RangeSlider**: filters launches by payload mass to isolate effects of payload range on outcomes.
- **Scatter plot**: payload vs outcome (class), colored by booster category with hover showing site — reveals payload–outcome and booster-related patterns under current filters

Predictive Analysis (Classification)

To see Predictive Analysis notebook (GitHub) click here Link

Set up environment for modelling and evaluate the model

Build a function to evaluate the model prediction (Confusion matrix)

Normalize and Split Data (Train and Test)

Tune the parameters to fine the best (Grid search), and Train the model

Predict on the test data and Evaluate with Confusion Matrix

Results

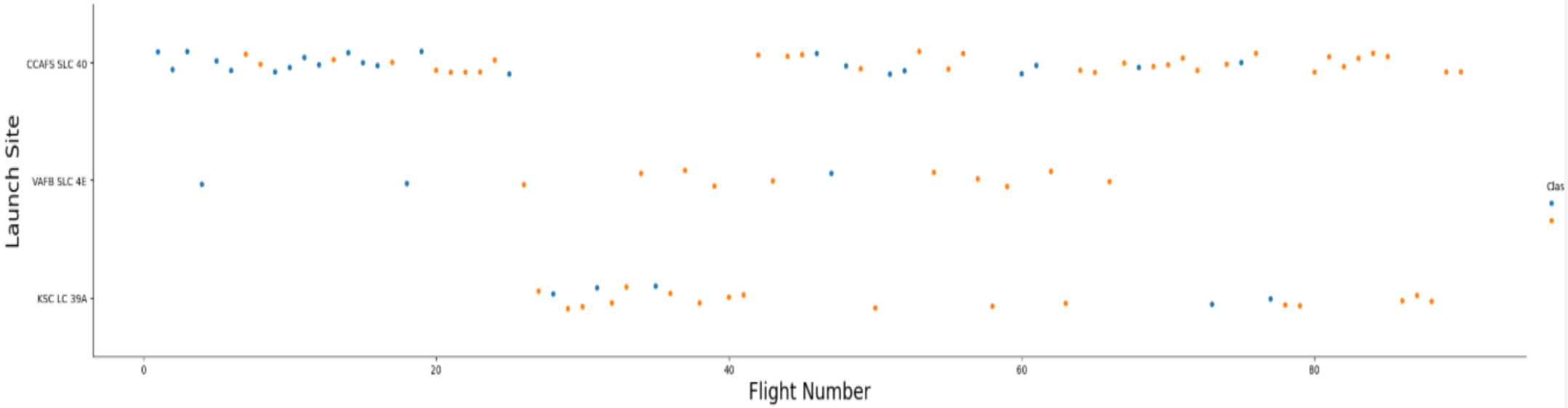
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

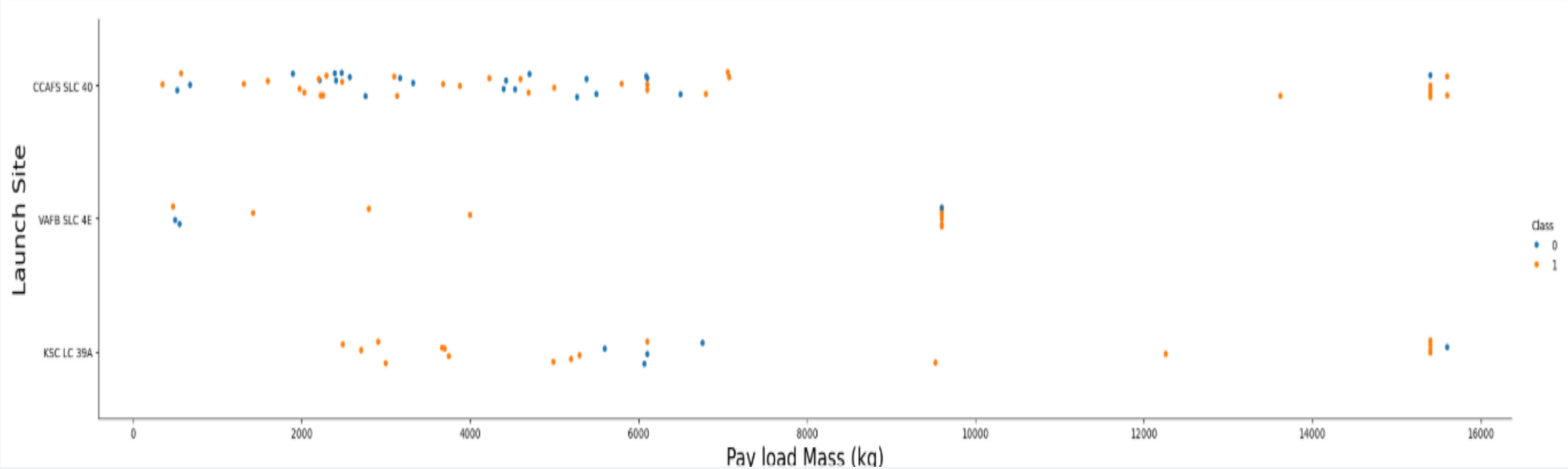
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



CCAFS SLC 40 has the most launches, spread across almost all flight numbers. Flight number can serve as a rough timeline. The plot shows: -Early flights are riskier. -Each launch site eventually reaches a high success rate.

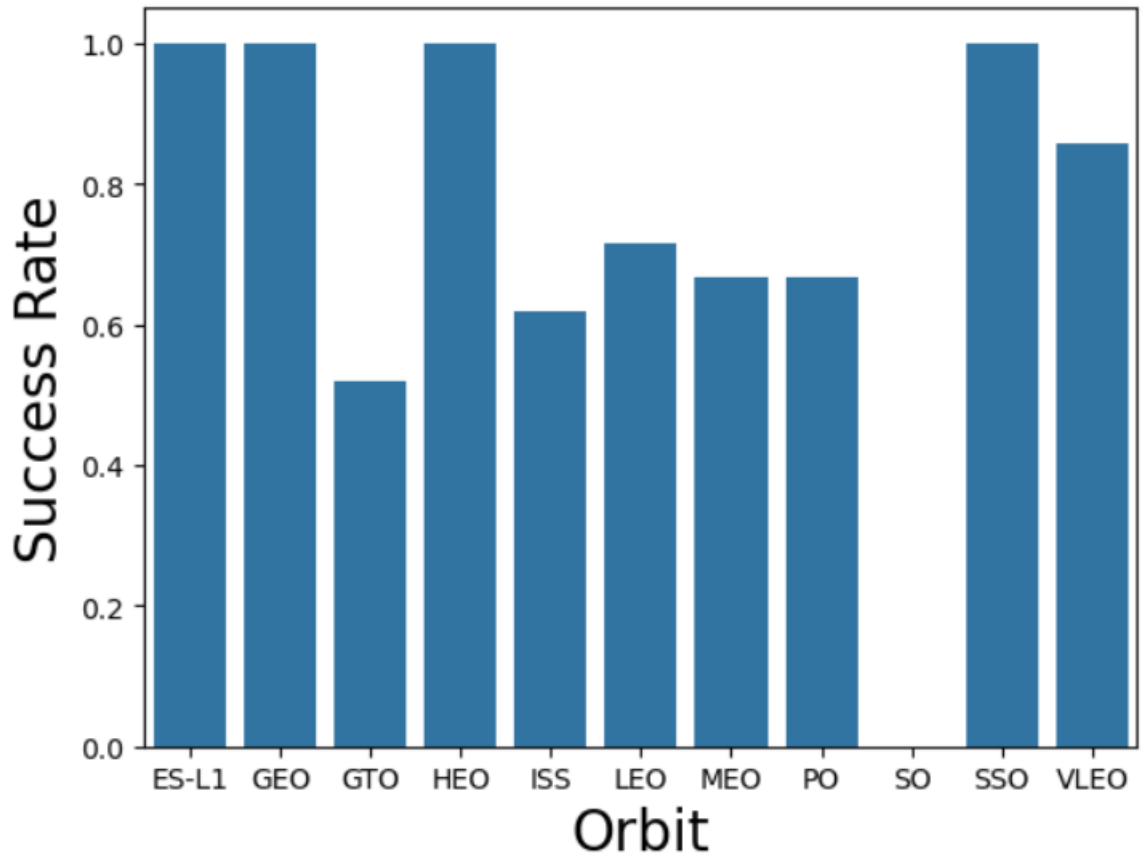
Payload vs. Launch Site

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Success Rate vs. Orbit Type

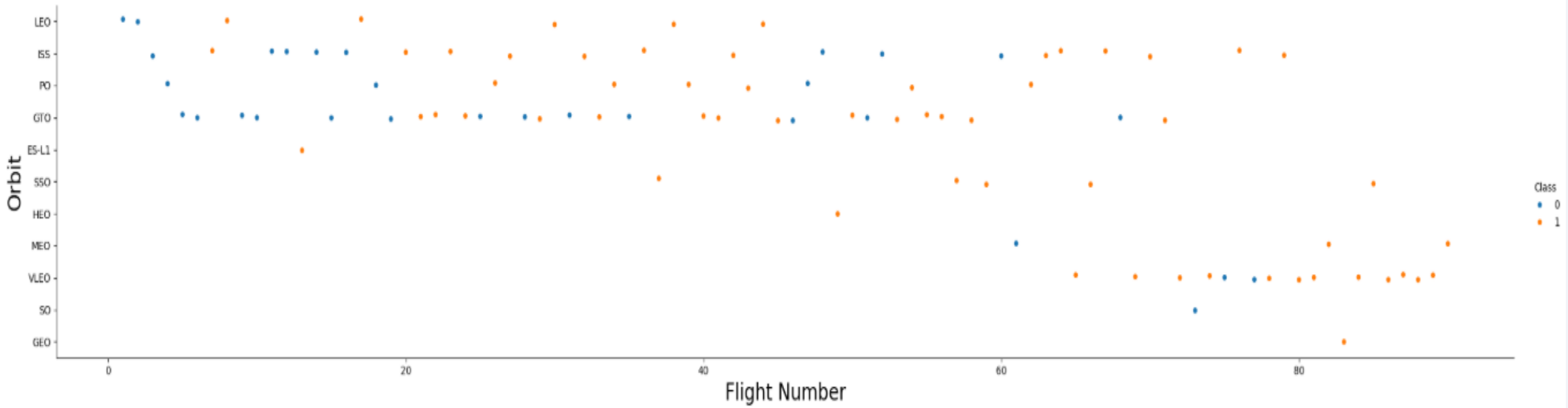
```
sns.barplot(data=success_rate, x="Orbit", y="Class")
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```



ES-L1, GEO, HEO, SSO have the highest sucess rate.

Flight Number vs. Orbit Type

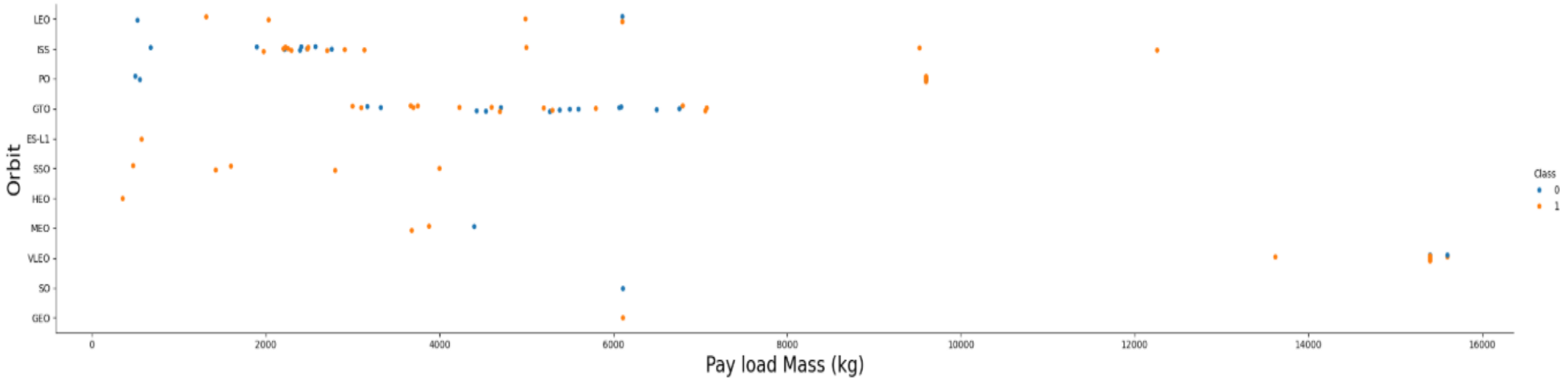
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



LEO orbit the 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.

Payload vs. Orbit Type

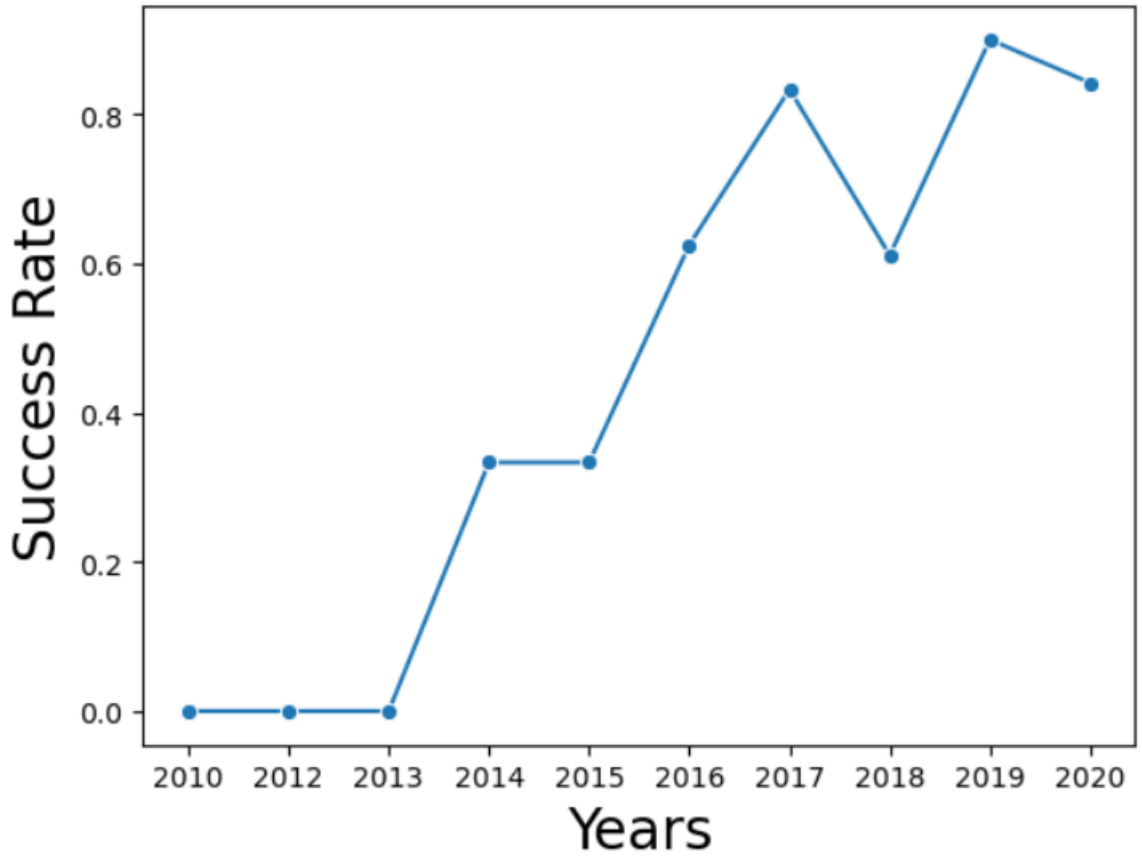
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend



You can observe that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing. 2018 and 2020 descreased but get the highest suceess rate in 2019

All Launch Site Names

```
%sql SELECT DISTINCT Launch_Site FROM SPACEXTABLE
```

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

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

- Distinct allow to retrieve unique Launch Site in The SPACEX Table

Launch Site Names Begin with 'CCA'

Display 5 records where launch sites begin with the string '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

Summarize the total payload Mass carried by boosters from NASA

To get the result use the function SUM and filter with customer = `NASA (CRS)´

Average Payload Mass by F9 v1.1

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass FROM SPACEXTABLE WHERE Booster_Version = 'NASA (CRS)';
```

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

Average_Payload_Mass
2928.4

First Successful Ground Landing Date

```
%sql SELECT MIN(Date) AS First_Ground_Pad_Success FROM SPACEXTABLE WHERE Landing_Outcome LIKE '%ground pad%';
```

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

First_Ground_Pad_Success
2015-12-22

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

```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 60
```

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

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
]: %sql SELECT Mission_Outcome, COUNT(*) AS Total_Count FROM SPACEXTABLE GROUP BY Mission_Outcome;
```

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

]:

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

Boosters Carried Maximum Payload

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

```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
```

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

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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
%sql SELECT substr("Date", 6, 2) AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site"FROM "SPACEXTABLE"
WHERE "Landing_Outcome" = 'Failure (drone ship)'AND substr("Date", 1, 4) = '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

- 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

```
%sql SELECT "Landing_Outcome", COUNT(*) AS Outcome_Count FROM "SPACEXTABLE" WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing_Outcome" ORDER BY Outcome_Count DESC;
```

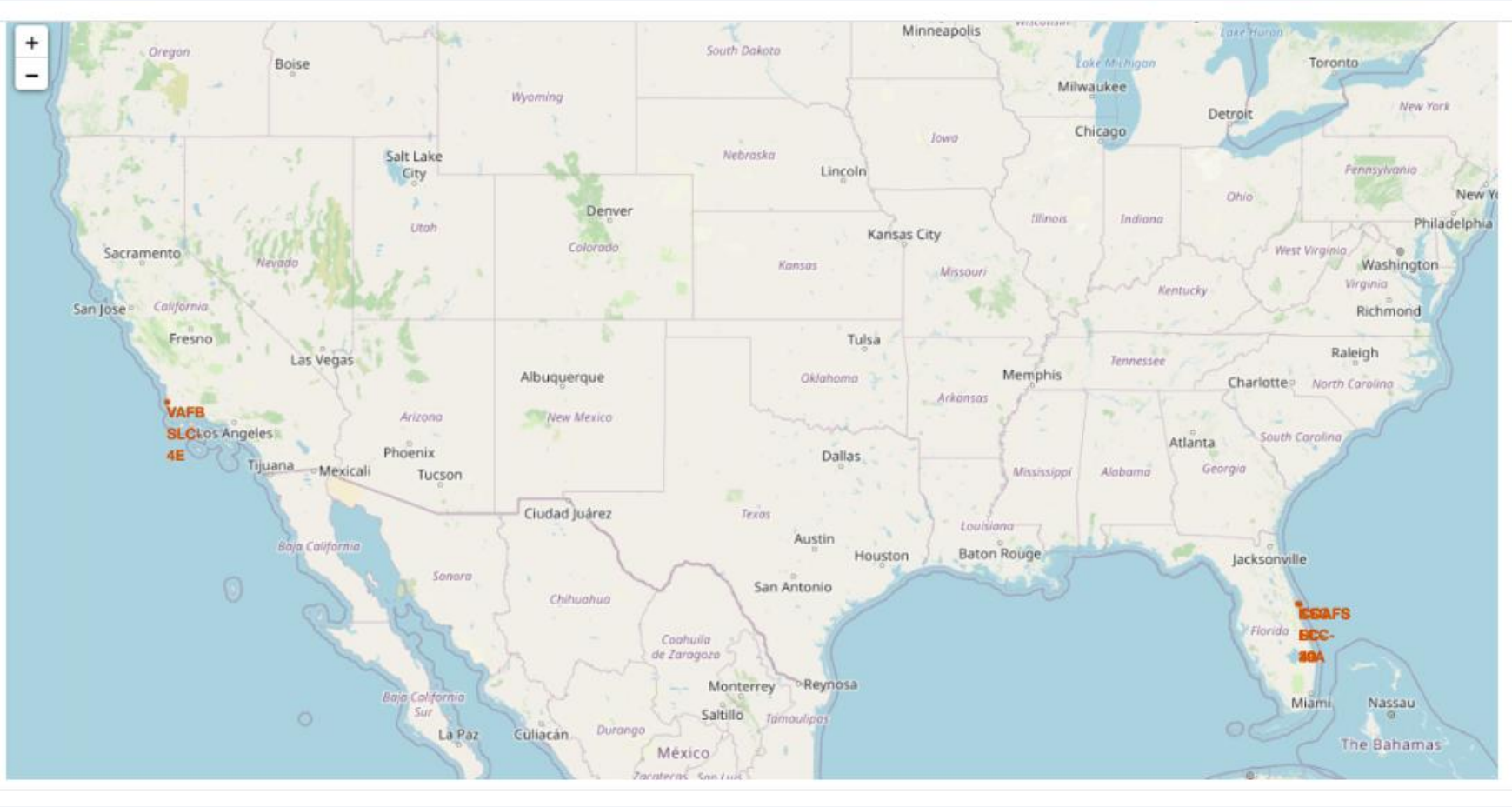
```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome	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

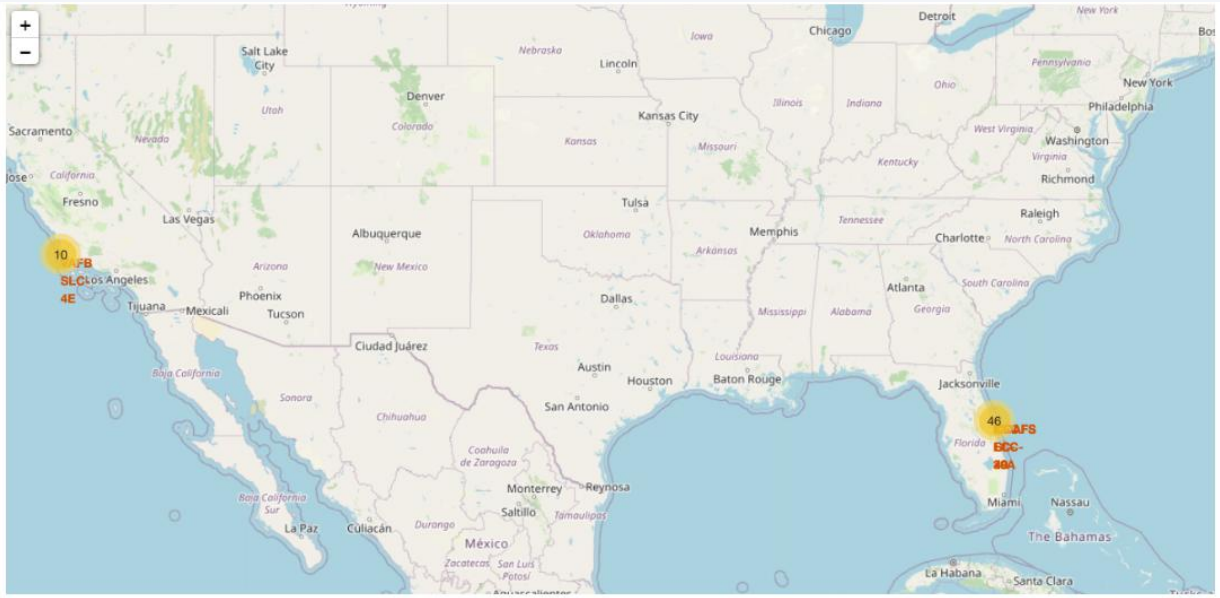
Launch Sites Proximities Analysis

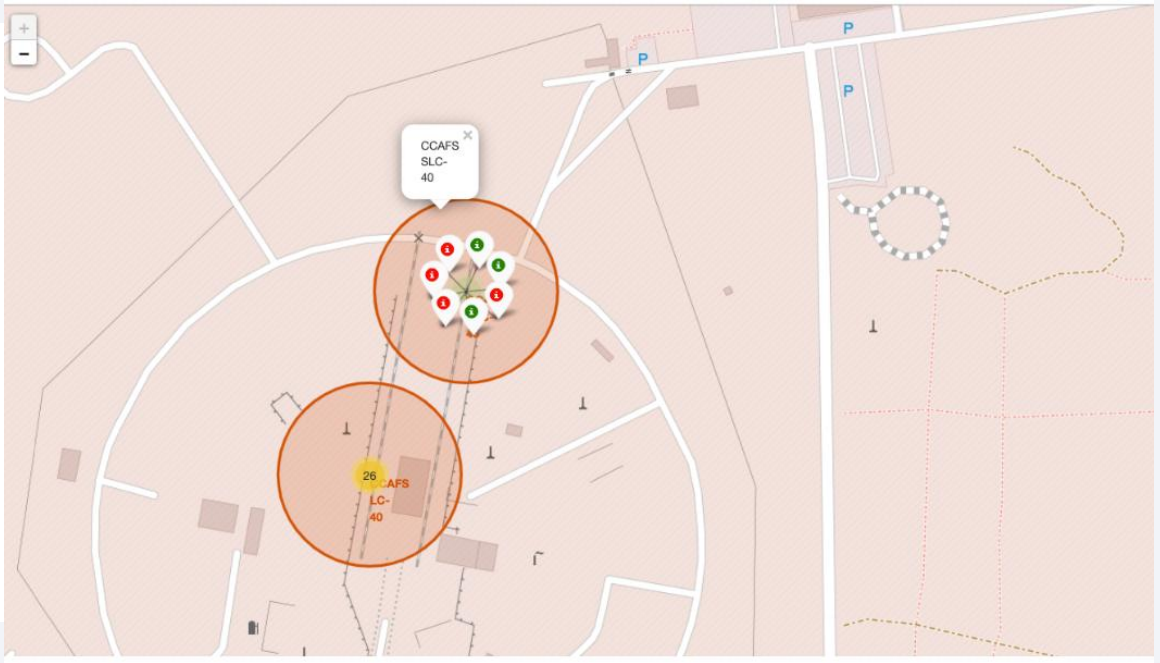
Launch sites 'location



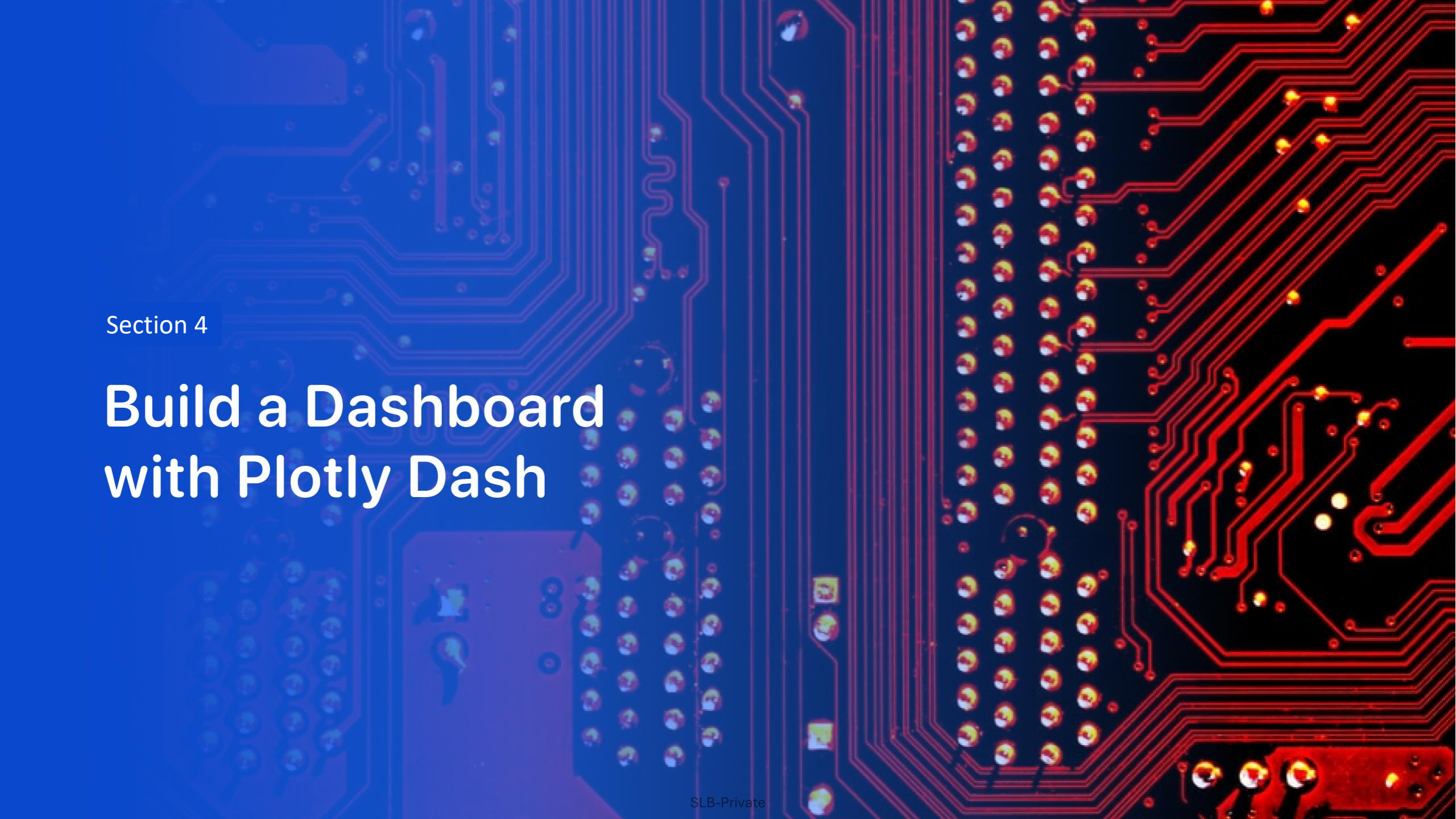
Only VAFB SLC-4E site is on the west coast of US. The rest is on the East coast

Success/failed launches for each site





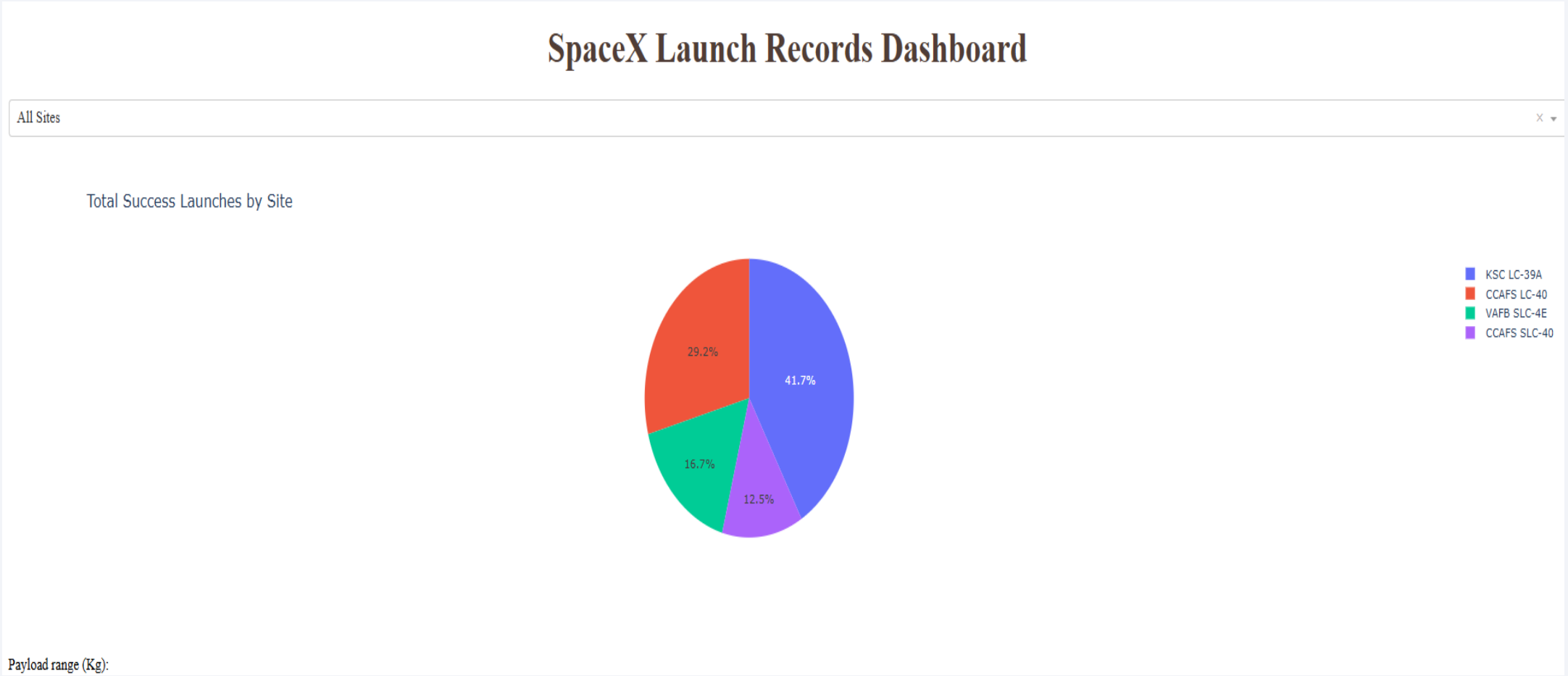
Distances between a launch site to its proximities



Section 4

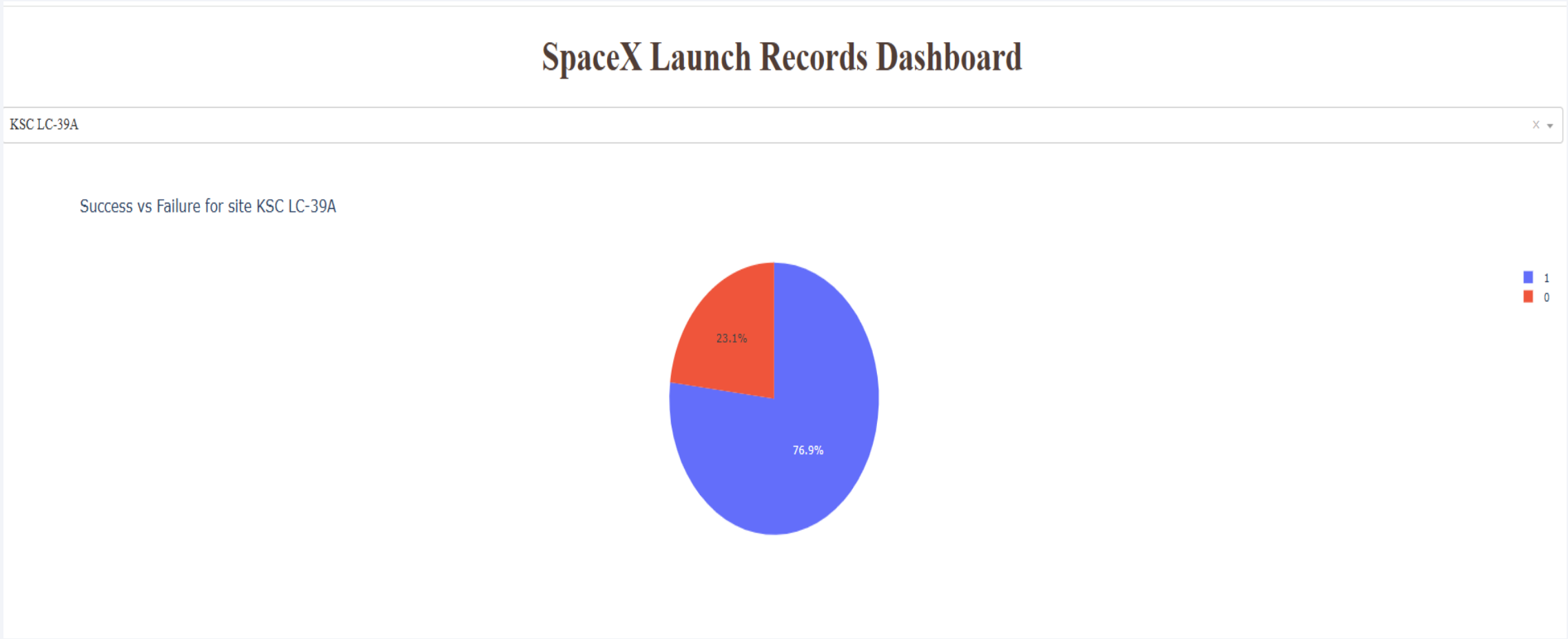
Build a Dashboard with Plotly Dash

Launch success count for all sites



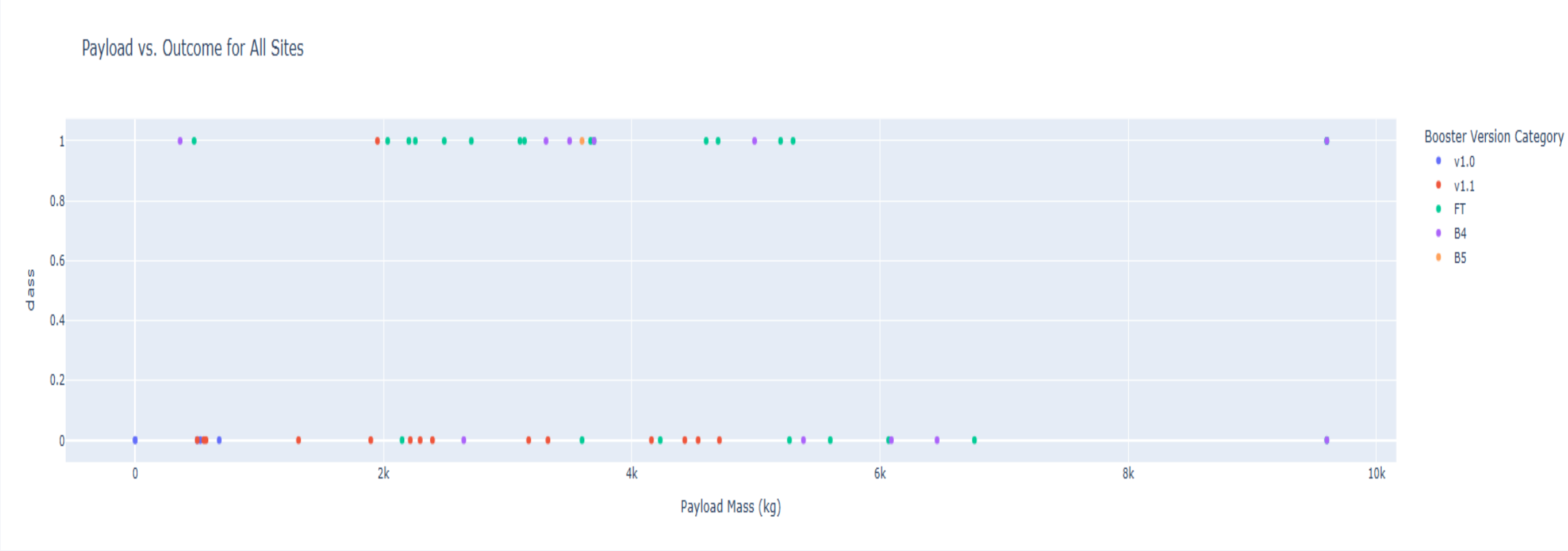
KSC LC 39A is the best Launch site with the success rate

Success vs Failure for site KSC LC 39A



Success rate is 76.9% is the highest of all sites. That means 1 failed over 5 Launch

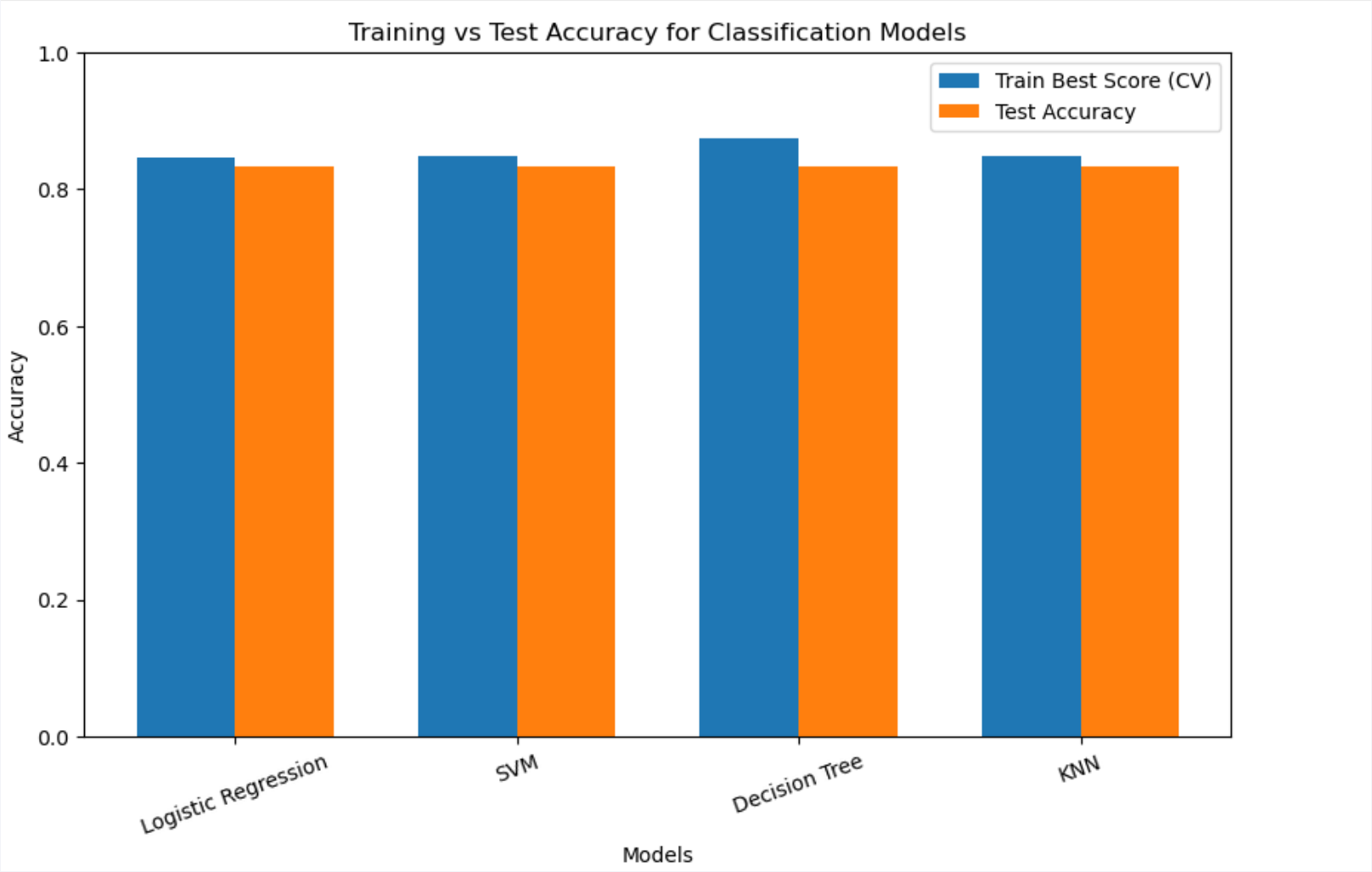
Payload vs. Launch Outcome for all sites



Section 5

Predictive Analysis (Classification)

Classification Accuracy

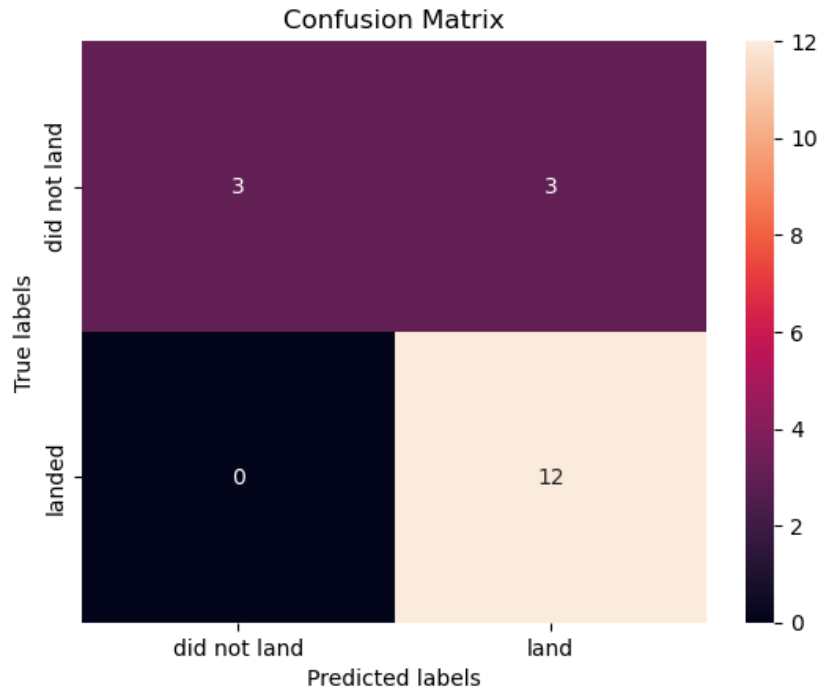


	Model	Train Best Score (CV)	Test Accuracy
0	Logistic Regression	0.846429	0.833333
1	SVM	0.848214	0.833333
2	Decision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333

Logistic Regression is the nearest predict score compare train to Test data

Confusion Matrix

- The Best performing model(s) with accuracy 0.8333333333333334: ['Logistic Regression', 'SVM', 'Decision Tree', 'KNN']



Examining the confusion matrix, we see that logistic regression can distinguish between the different classes. We see that the problem is false positives.

Overview:

True Postive - 12 (True label is landed, Predicted label is also landed)

False Postive - 3 (True label is not landed, Predicted label is landed)

Conclusions

- 4 disctint Launch sites
- ES-L1, GEO, HEO and SSO Orbits got 100% success rate
- KSC LC 39A is the best Launch site with the success rate
- Logistic retrogression give the best model prediction between train and test data. It can be use for the predict the next launches

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

```
Best performing model(s) with accuracy 0.8333333333333334: ['Logistic Regression', 'SVM', 'Decision Tree', 'KNN']
```

```
accuracy_table = pd.DataFrame({
    'Model': ['Logistic Regression', 'SVM', 'Decision Tree', 'KNN'],
    'Train Best Score (CV)': [
        logreg_cv.best_score_,
        svm_cv.best_score_,
        tree_cv.best_score_,
        knn_cv.best_score_
    ],
    'Test Accuracy': [
        accuracy_logreg,
        accuracy_svm,
        accuracy_tree,
        accuracy_knn
    ]
})

accuracy_table
```

	Model	Train Best Score (CV)	Test Accuracy
0	Logistic Regression	0.846429	0.833333
1	SVM	0.848214	0.833333
2	Decision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333

Thank you!