IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

José Mera González
30 September 2025

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Methodology

- Data collection of historical records of Falcon 9 first-stage landings, sourced from Wikipedia and the SpaceX API.
- Classification: successful landings (1) versus failed landings (0), categorized in the "class" column (0: failed, 1: successful).
- Tools: IBM environment for Python.

Summary of Results

- The first landing attempt per orbit usually fails, but subsequent ones tend to succeed. This suggests strong data analysis by SpaceX, which has enabled the company to learn how to land the first-stage module with minimal cost. I assume they might include a few risky variables in the first landing attempt to tackle everything in one go or at least get close to it.
- The 0 class (failed landings) also includes non-recovery failures due to decisions made by the management team.

Introduction

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

Space Y and its magnate, Allon Musk—what a report on SpaceX's Falcon 9 first landings. Assuming that a higher rate of successful landings could help this new business snag a piece of the rocket-launching industry.

We want to dig into the rate of successful landings, plus their relationship to the payload charge, the landing place, and the destined orbit.

The insights we uncover could let us offer the market a cheaper product—not necessarily aiming to cover the full range of SpaceX's capabilities, just targeting the niches where Space Y stands to make the most profit.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data source Wikipedia and Space X API
- Perform data wrangling
  - Removed duplicates, normalized variables, built a structured table
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - I tunned the hyperparameters to improve the confussion matrix results.

Data Collection

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

The data sources were Wikipedia datasets and the SpaceX API. I ran Python scripts for automated extraction and cleaning, making sure landing outcomes were consistently classified by adding a new categorical column.
I removed duplicates and missing or inconsistent values, normalizing variables like dates to build a clean, structured table.
Used SQL queries and Python visualizations to dig into patterns in success rates across landing types.
I threw together Folium maps for geospatial landing distribution and Plotly Dash dashboards for dynamic filtering and success rate comparisons.
Built models (logistic regression, decision trees) to predict landing success. Did some hyperparameter tuning to optimize accuracy, evaluating everything with train/test splits and cross-validation.

Data Collection – SpaceX API

- I sent a GET request to the SpaceX API.
- Later, I parsed the JSON response into a structured format using the Python library pandas. Replaced NaN values with the column mean.
- And in the final step, I exported the cleaned DataFrame.

[Space X API](https://github.com/Glorfindiel/IBM-DS/blob/main/jupyter-labs-spacex-data-collection-api%20(1).ipynb)

https://github.com/Glorfindiel/IBM-DS/blob/main/jupyter-labs-spacex-data-collection-api%20(1).ipynb

Start

↓

Send GET request to SpaceX REST API (/launches)

↓

Receive JSON response

↓

Parse JSON into Pandas DataFrame

↓

Data Wrangling

- Handle missing PayloadMass (replace NaN with mean)

- Keep None for LandingPad

↓

Export Clean Dataset (CSV)

↓

Use for EDA, Visualization, and Modeling

Data Collection - Scraping

- I sent an HTTP GET request to retrieve Falcon 9 launch records from Wikipedia using BeautifulSoup.
- I identified and extracted HTML table header names.
- I parsed HTML table rows into a Python dictionary, then used pandas to convert it to a DataFrame.
- I exported the cleaned DataFrame to a CSV for later work.

Scrapping

https://github.com/Glorfindiel/IBM-DS/blob/main/jupyter-labs-webscraping_solucionado.ipynb

Start

↓

HTTP GET request → Wikipedia Falcon 9 Launch page

↓

Retrieve HTML Response

↓

BeautifulSoup parses HTML tables

↓

Extract column names (table headers)

↓

Parse table rows → Build dictionary

↓

Convert dictionary → Pandas DataFrame

↓

Export dataset (CSV)

↓

Use for further analysis

Data Wrangling

I did different calculations here, like:

- Counting the launches per site
- Counting the occurrence of each orbit
- Analyzing mission outcomes

And I converted the outcome into a binary label: 1 for successful, 0 for failed.

[Data Wrangling](https://github.com/Glorfindiel/IBM-DS/blob/main/labs-jupyter-spacex-Data%20wrangling_resuelto.ipynb)

https://github.com/Glorfindiel/IBM-DS/blob/main/labs-jupyter-spacex-Data%20wrangling_resuelto.ipynb

EDA with Data Visualization

- Scatter Plot (Flight Number vs Launch Site, colored by Class)
  To analyze if launch frequency by site influences landing success.
- Scatter Plot (Payload Mass vs Launch Site, colored by Class)
  To observe the effect of payload weight on success rates at different launch sites.
- Bar Chart (Success Rate by Orbit Type)
  To compare which orbits are more likely to result in successful landings.
- Scatter Plot (Flight Number vs Orbit Type, colored by Class)
  To study whether launch experience (higher flight number) improves landing success across orbits.
- Scatter Plot (Payload Mass vs Orbit Type, colored by Class)
  To assess the impact of payload mass on landing success across different orbit categories.
- Line Chart (Yearly Success Trend)
  To identify how success rates evolved over time, showing improvement in SpaceX reliability.

[EDA with Data Visualization](https://github.com/Glorfindiel/IBM-DS/blob/main/edadataviz_completo.ipynb)

https://github.com/Glorfindiel/IBM-DS/blob/main/edadataviz_completo.ipynb

EDA with SQL

- **Unique Launch Sites** → Query to display the distinct names of all launch sites.
- **Filter by Prefix** → Retrieved first 5 records where LaunchSite begins with "CCA".
- **Total Payload Mass by NASA (CRS)** → Calculated the sum of payloads carried by NASA (CRS) missions.
- **Average Payload Mass (F9 v1.1)** → Computed average payload mass for booster version **F9 v1.1**.
- **First Successful Ground Pad Landing** → Used MIN(Date) to find earliest successful landing on a ground pad.
- **Drone Ship Success with Medium Payloads** → Listed boosters with successful **ASDS landings** and payload mass between 4000–6000 kg.
- **Mission Outcomes Count** → Counted total number of **successes vs failures** in mission outcomes.
- **Max Payload Booster Versions** → Identified booster versions carrying the maximum payload using a subquery.
- **Failures on Drone Ship in 2015** → Extracted month, outcome, booster version, and site for failed ASDS landings in 2015.
- **Ranked Landing Outcomes (2010–2017)** → Ranked counts of different landing outcomes (success/failure, ground pad/drone ship) between 2010-06-04 and 2017-03-20.

EDA with SQL

https://github.com/Glorfindiel/IBM-DS/blob/main/jupyter-labs-eda-sql-coursera_sqllite_solucion.ipynb

Build an Interactive Map with Folium

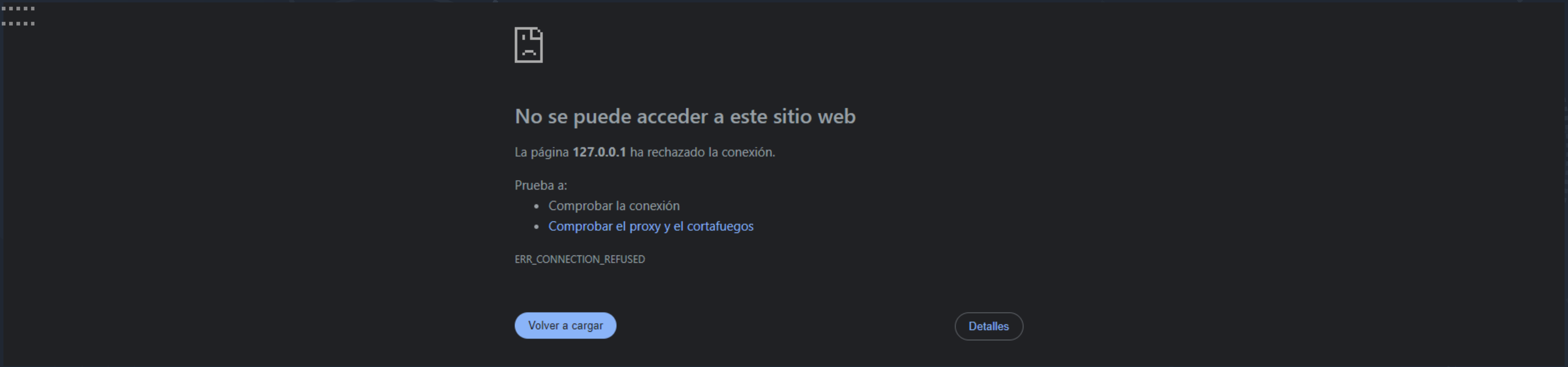
- **Circle & Marker for Launch Sites**
  → **Added circles and labeled markers at each launch site to visualize their geographical locations** on the map.
  → Purpose: Show spatial distribution of SpaceX sites and analyze proximity to coastlines/equator.
- **Colored Markers for Launch Outcomes**
  → **Green markers = successful landings, Red markers = failed landings.**
  → **Purpose: Distinguish outcomes directly on the map and detect location–success correlations.**
- **Marker Clusters**
  → **Grouped overlapping markers at the same coordinates.**
  → **Purpose: Simplify visualization when multiple launches occur at one site.**
- **Distance Lines (Launch Site → Proximities)**
  → **Added lines to measure distance from each launch site to relevant nearby features (e.g., coast).**
  → **Purpose: Evaluate whether proximity factors might influence launch outcomes**

[Build an Interactive Map with Folium](https://github.com/Glorfindiel/IBM-DS/blob/main/lab_jupyter_launch_site_location_completo.ipynb)

https://github.com/Glorfindiel/IBM-DS/blob/main/lab_jupyter_launch_site_location_completo.ipynb

Build a Dashboard with Plotly Dash

- Success Rate Pie Chart: Proportion, successful/failed landings, Falcon 9, launch site, dynamic updates
- Payload vs. Success Scatter Plot: Payload mass, landing success/failure, booster version, correlation visualization
- Launch Site Dropdown: Filter, specific sites, KSC, 76.9% success
- Payload Range Slider: Custom range, 0–9,600 kg, real-time filtering
- Build a Dashboard with Plotly Dash
- https://github.com/Glorfindiel/IBM-DS/blob/main/theia.tar
- My firewall is evil



Predictive Analysis (Classification)

- **Data Preparation**
  → **Created training labels (**Class column).
  → Standardized features (scaled inputs).
  → Split dataset into **training (80%)** and **testing (20%)**.
- **Model Building**
  → **Trained Logistic Regression**, **SVM**, **Decision Tree**, and **KNN** classifiers.
  → Applied **GridSearchCV (cv=10)** for hyperparameter tuning.
- **Evaluation**
  → **Measured validation accuracy** (via cross-validation).
  → Calculated **test accuracy** for each tuned model.
- **Improvement**
  → **Adjusted hyperparameters (regularization strength, kernel, depth, neighbors).**
  → **Compared models to avoid overfitting.**
- **Best Performing Model**
  → **Selected the classifier with highest test accuracy** as the final predictive model.

```
Best performing model: Logistic Regression - Test accuracy: 0.8333333333333334
```

[Predictive Analysis (Classification)](https://github.com/Glorfindiel/IBM-DS/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5_Completo.ipynb)

https://github.com/Glorfindiel/IBM-DS/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5_Completo.ipynb

Results

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

Section 2

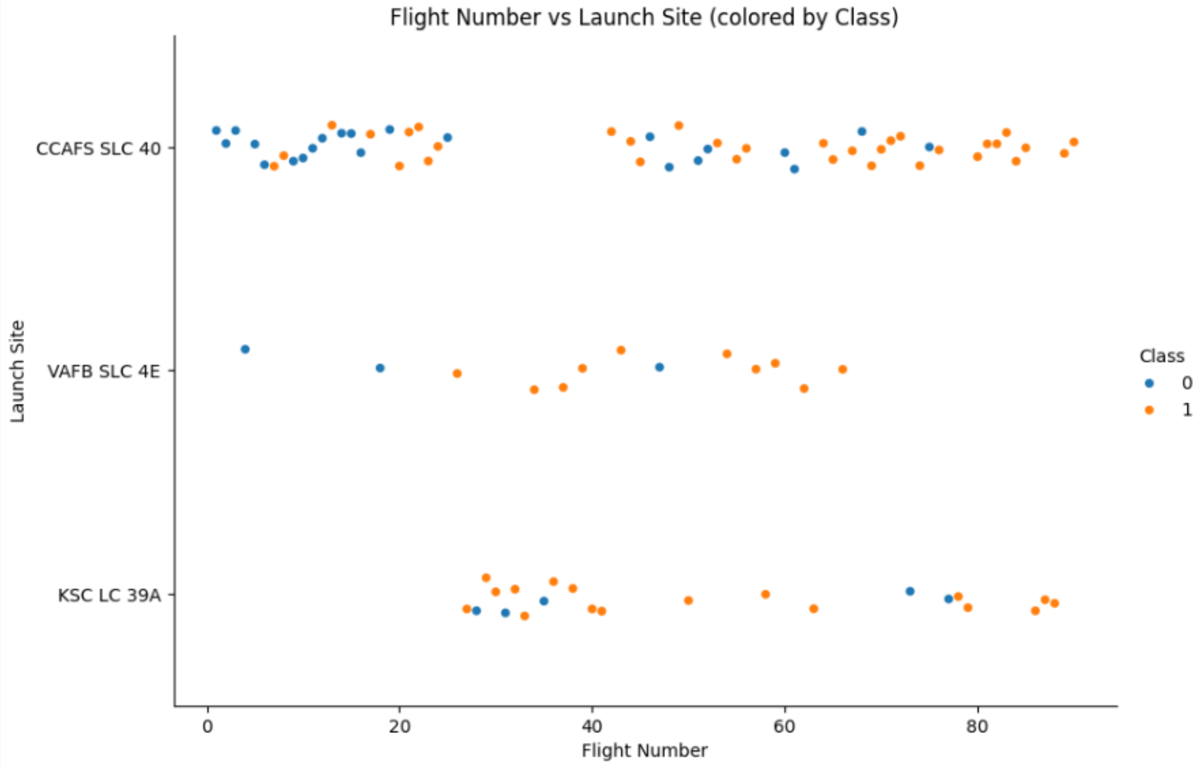
Insights drawn from EDA

Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site
- Show the screenshot of the scatter plot with explanations

As we can see in the scatterplot, before the twentieth launch, the failure rate was high, but SpaceX and its data team managed to boost efficiency, losing only 14 boosters before then.

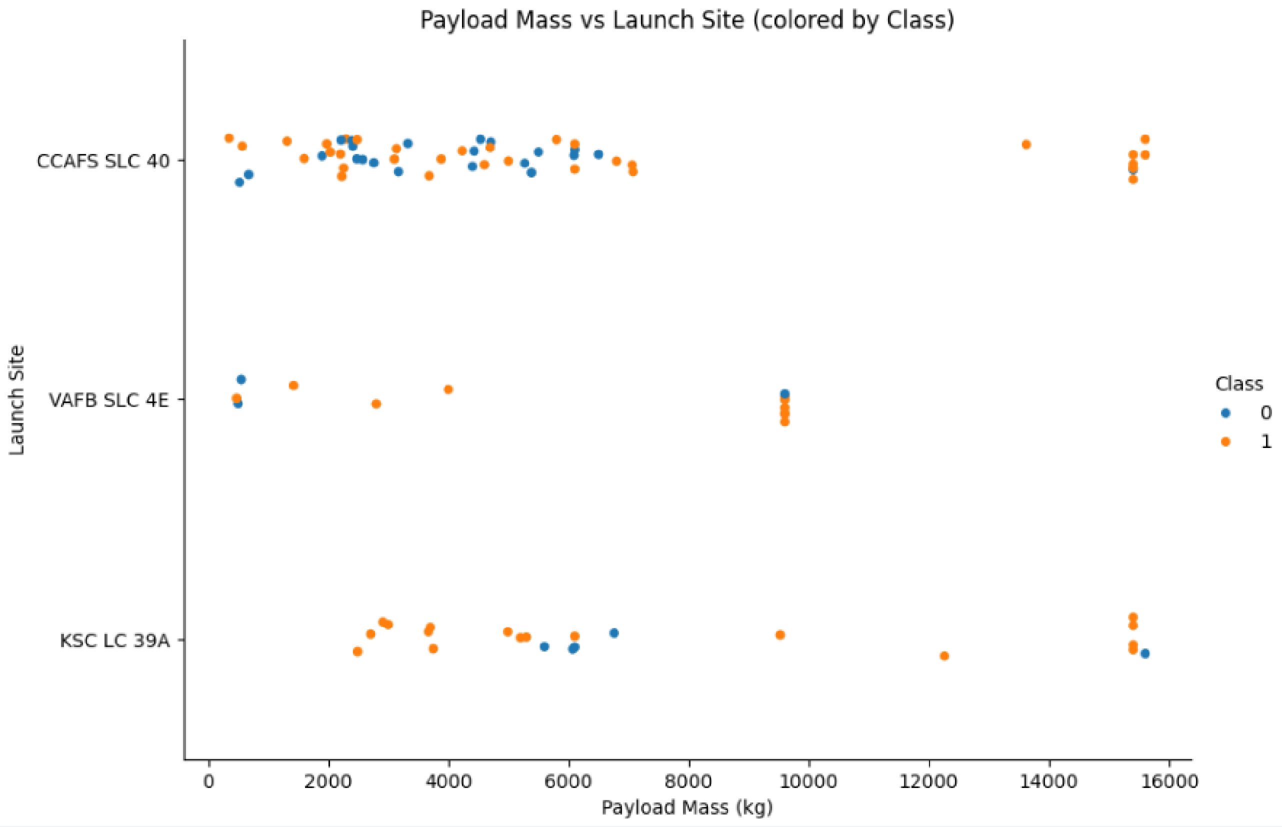
CCAFS SLC-40 has a high failure rate, but it seems to be their test bench. We require more info to clarify this point.



Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site
- Show the screenshot of the scatter plot with explanations

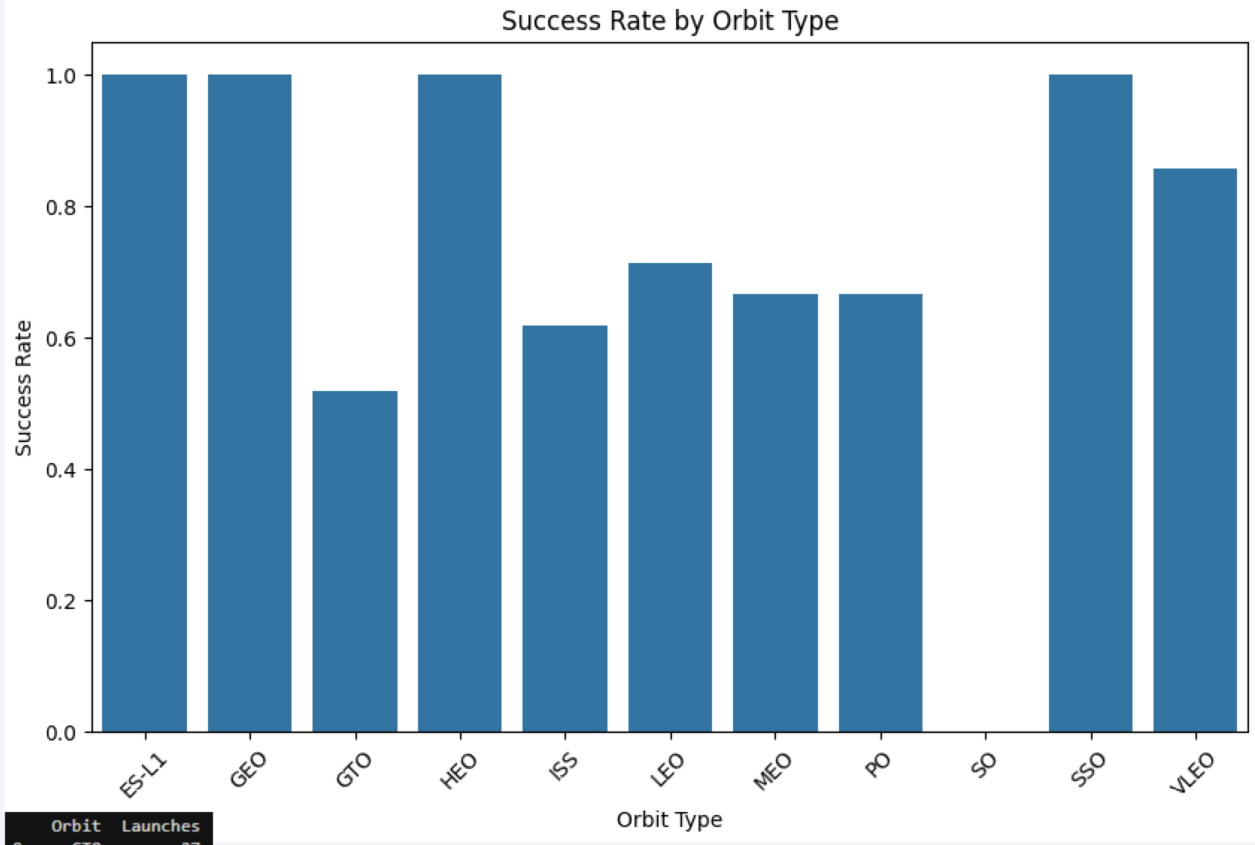
CCAFS SLC-40 seems, once again, the worst place to get a successful recovery. Also, lighter rockets seem to be more unstable and harder to recover than heavier ones. Maybe we need to gather environmental data, like wind, to check this point.



Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type
- Show the screenshot of the scatter plot with explanations

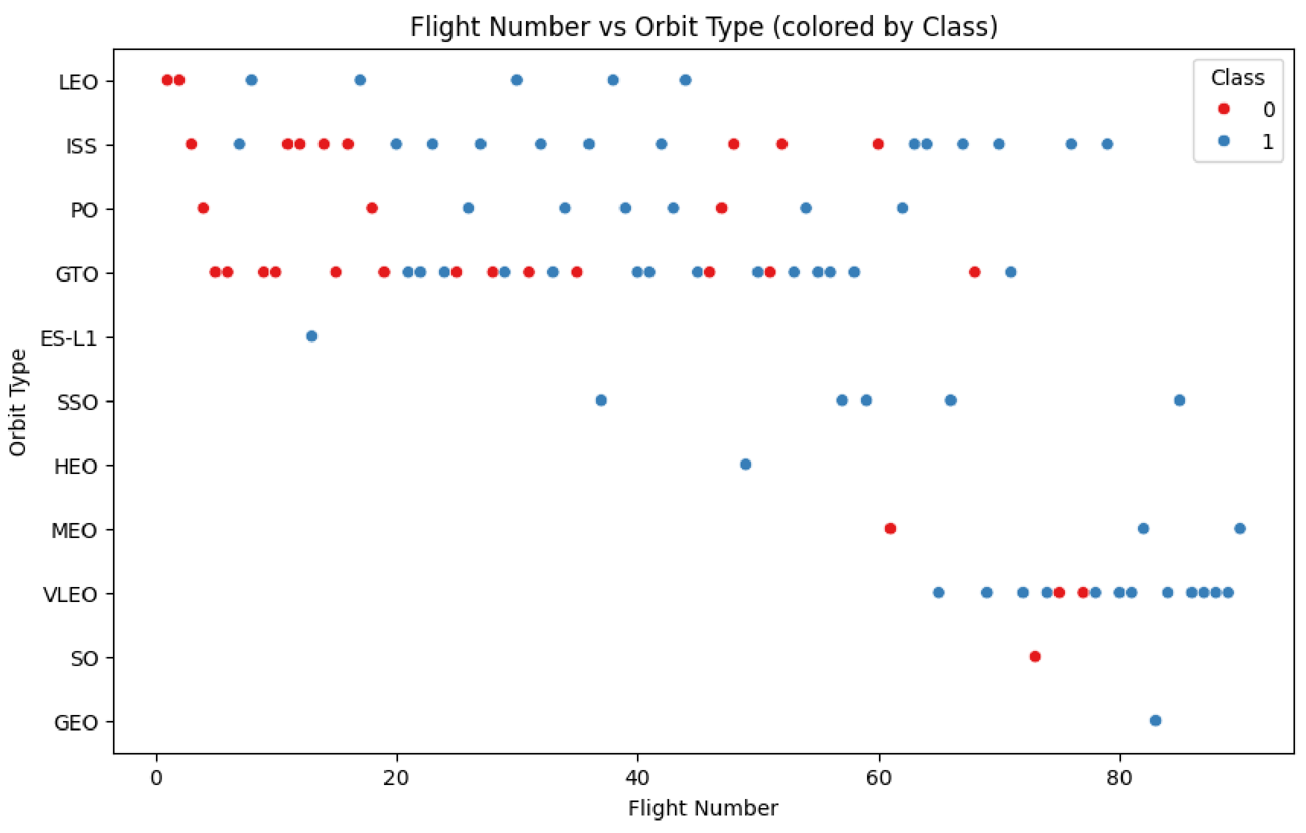
We can see that the orbits with more accumulated launches have higher failure rates, especially GTO, which is an intermediate orbit for placing a satellite in GEO. Others have only one launch, so I can’t say much more about them. But again, there’s a learning pattern underlying the failure-success rate.



	Orbit	Launches
0	GTO	27
1	ISS	21
2	VLEO	14
3	PO	9
4	LEO	7
5	SSO	5
6	MEO	3
7	ES-L1	1
8	HEO	1
9	SO	1
10	GEO	1

Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type
- Show the screenshot of the scatter plot with explanations

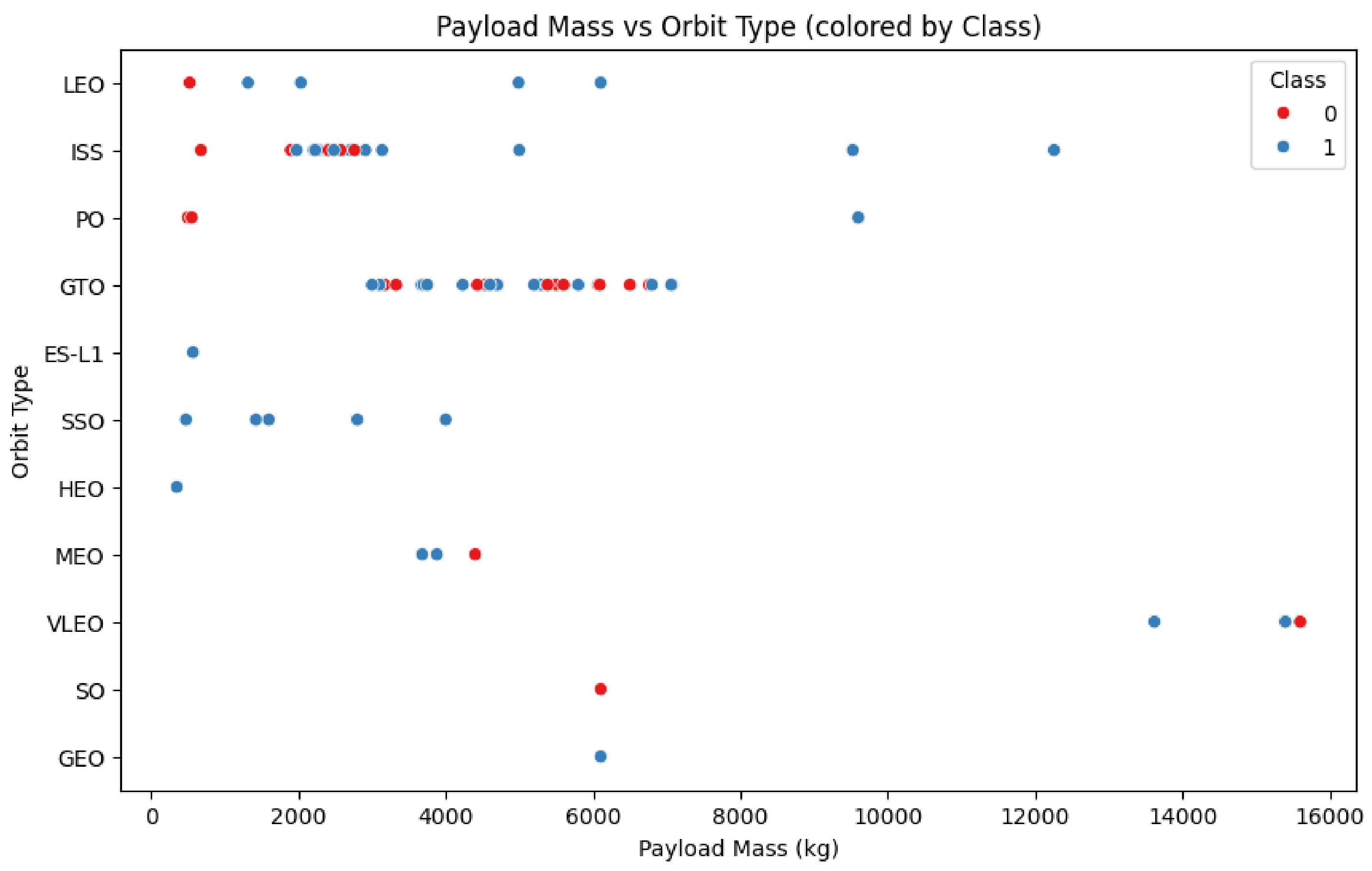


This scatter plot is more sensitive to the learning pattern behavior, showing high failure rates in the early stages, which are interesting because they reveal to Space Y the cost of investing in learning how to land. It also seems there's a second block of failures around the 50th launch—*maybe overconfidence or a change in something.* We need a detailed study at that point.

Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type
- Show the screenshot of the scatter plot with explanations

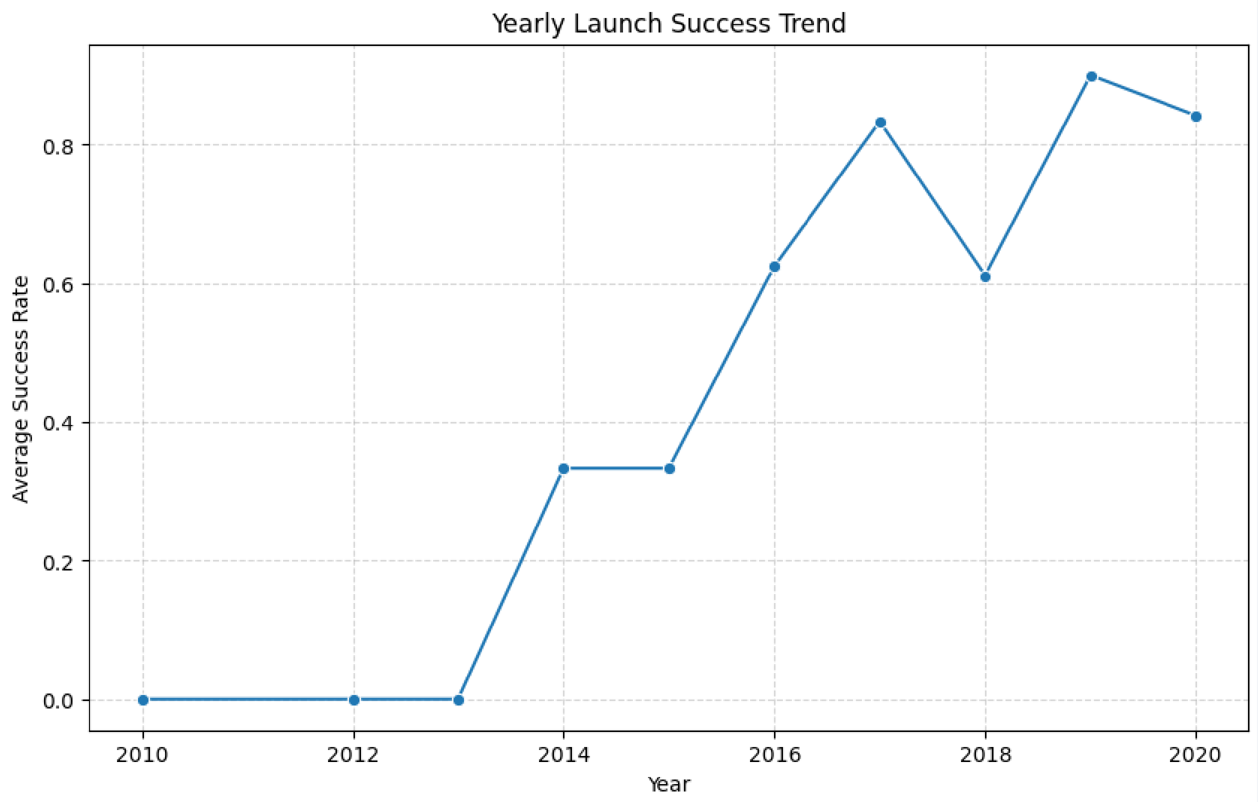
Now we can see how ISS and GTO concentrate most of the failures, usually with medium-mass payloads. LEO, ISS, and PO show the difficulty of controlling the landing with light-mass payloads. There's an outlier at VLEO, close to 16,000 kg, that failed. But it seems that higher mass makes recovery easier.



Launch Success Yearly Trend

- Show a line chart of yearly average success rate
- Show the screenshot of the scatter plot with explanations

2018 seems to be a year worth investigating in the future, as it shows an anomaly in the learning pattern on SpaceX landings. Also, we can see that SpaceX needed 5 years of training to get over 50% success.



All Launch Site Names

- Find the names of the unique launch sites
- Present your query result with a short explanation here

```
In [10]:  query = 'SELECT DISTINCT "Launch_Site" FROM SPACEXTBL'

          unique_launch_sites = pd.read_sql(query, con)
          print(unique_launch_sites)

     Launch_Site
0    CCAFS LC-40
1    VAFB SLC-4E
2     KSC LC-39A
3   CCAFS SLC-40
```

SpaceX uses four main sites to launch their rockets.

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Present your query result with a short explanation here

```
query = '''
SELECT *
FROM SPACEXTBL
WHERE "Launch_Site" LIKE 'CCA%'
LIMIT 5
'''

cca_launches = pd.read_sql(query, con)
print(cca_launches)
```

```
        Date Time (UTC) Booster_Version  Launch_Site  \
0  2010-06-04   18:45:00  F9 v1.0  B0003  CCAFS LC-40
1  2010-12-08   15:43:00  F9 v1.0  B0004  CCAFS LC-40
2  2012-05-22    7:44:00  F9 v1.0  B0005  CCAFS LC-40
3  2012-10-08    0:35:00  F9 v1.0  B0006  CCAFS LC-40
4  2013-03-01   15:10:00  F9 v1.0  B0007  CCAFS LC-40
```

With SQL, we can filter the data in many ways. One of them is to list 5 launch sites starting with "CCA".

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here

```
query = '''
SELECT SUM("Payload_Mass__kg_") AS Total_Payload
FROM SPACEXTBL
WHERE "Customer" = 'NASA (CRS)'
'''

total_payload = pd.read_sql(query, con)
print(total_payload)
```

```
   Total_Payload
0          45596
```

We can also do math operations, like asking how much payload was sent to space by SpaceX.

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here

```
# Consulta SQL para calcular el promedio de Payload Mass para la versión F9 v1.1
query = '''
SELECT AVG("Payload_Mass__kg_") AS Avg_Payload
FROM SPACEXTBL
WHERE "Booster_Version" = 'F9 v1.1'
'''

avg_payload = pd.read_sql(query, con)
print(avg_payload)
```

```
   Avg_Payload
0       2928.4
```

Or we can combine a math operation with a filter, like I showed in this task.

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here

```
# Consulta SQL para obtener la fecha del primer aterrizaje exitoso en ground pad
query = '''
SELECT MIN("Date") AS First_Successful_Ground_Landing
FROM SPACEXTBL
WHERE "Landing_Outcome" = 'Success (ground pad)'
'''

first_successful_landing = pd.read_sql(query, con)
print(first_successful_landing)
```

```
  First_Successful_Ground_Landing
0                      2015-12-22
```

Or search for specific information, like now. This date is important because it shows the 5-year learning path— a useful insight for securing funds for our space company.

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
- Present your query result with a short explanation here

```
# Consulta SQL para obtener los boosters exitosos en drone ship con Payload Mass entre 4000 y 6000
query = '''
SELECT "Booster_Version"
FROM SPACEXTBL
WHERE "Landing_Outcome" = 'Success (drone ship)'
  AND "Payload_Mass__kg_" > 4000
  AND "Payload_Mass__kg_" < 6000
'''

successful_boosters = pd.read_sql(query, con)
print(successful_boosters)
```

```
  Booster_Version
0     F9 FT B1022
1     F9 FT B1026
2  F9 FT  B1021.2
3  F9 FT  B1031.2
```

Now I asked a query about the boosters that performed a successful landing on a sea drone with a medium payload. That, as we saw, has not good numbers on safe landings.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here

```
# Consulta SQL para contar los resultados exitosos y fallidos
query = '''
SELECT "Landing_Outcome", COUNT(*) AS Count
FROM SPACEXTBL
GROUP BY "Landing_Outcome"
'''

mission_outcomes = pd.read_sql(query, con)
print(mission_outcomes)
```

```
          Landing_Outcome  Count
0      Controlled (ocean)      5
1                 Failure      3
2    Failure (drone ship)      5
3     Failure (parachute)      2
4              No attempt     21
5             No attempt       1
6   Precluded (drone ship)     1
7                 Success     38
8    Success (drone ship)     14
9     Success (ground pad)     9
10   Uncontrolled (ocean)      2
```

This information is very relevant, as it shows 21 failed landings where the cause was that management wasn't willing to attempt a landing. So, the SpaceX numbers should be recalculated, excluding these from the failures, and maybe by adding a class 2: "no attempt."

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here

```
# Consulta SQL para obtener los boosters con la máxima Payload Mass
query = '''
SELECT "Booster_Version", "Payload_Mass__kg_"
FROM SPACEXTBL
WHERE "Payload_Mass__kg_" = (
    SELECT MAX("Payload_Mass__kg_")
    FROM SPACEXTBL
)
'''

max_payload_boosters = pd.read_sql(query, con)
print(max_payload_boosters)
```

	Booster_Version	PAYLOAD_MASS__KG_
0	F9 B5 B1048.4	15600
1	F9 B5 B1049.4	15600
2	F9 B5 B1051.3	15600
3	F9 B5 B1056.4	15600
4	F9 B5 B1048.5	15600
5	F9 B5 B1051.4	15600
6	F9 B5 B1049.5	15600
7	F9 B5 B1060.2	15600
8	F9 B5 B1058.3	15600
9	F9 B5 B1051.6	15600
10	F9 B5 B1060.3	15600
11	F9 B5 B1049.7	15600

I'm considering a hypothesis that less mass means harder chances to get a safe landing. So, I filtered the max-loaded boosters for further investigation.

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- Present your query result with a short explanation here

```
# Consulta SQL para Task 9
query = '''
SELECT
    substr("Date", 6, 2) AS Month,
    "Landing_Outcome",
    "Booster_Version",
    "Launch_Site"
FROM SPACEXTBL
WHERE "Landing_Outcome" LIKE 'False Drone%'
  AND substr("Date", 0, 5) = '2015'
'''

failure_2015 = pd.read_sql(query, con)
print(failure_2015)
```

```
Empty DataFrame
Columns: [Month, Landing_Outcome, Booster_Version, Launch_Site]
Index: []
```

It's a difficult way of landing, so it provides good info.

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
- Present your query result with a short explanation here

```
# Consulta SQL para Task 10
query = '''
SELECT "Landing_Outcome", COUNT(*) AS Outcome_Count
FROM SPACEXTBL
WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing_Outcome"
ORDER BY Outcome_Count DESC
'''

landing_outcome_rank = pd.read_sql(query, con)
print(landing_outcome_rank)
```

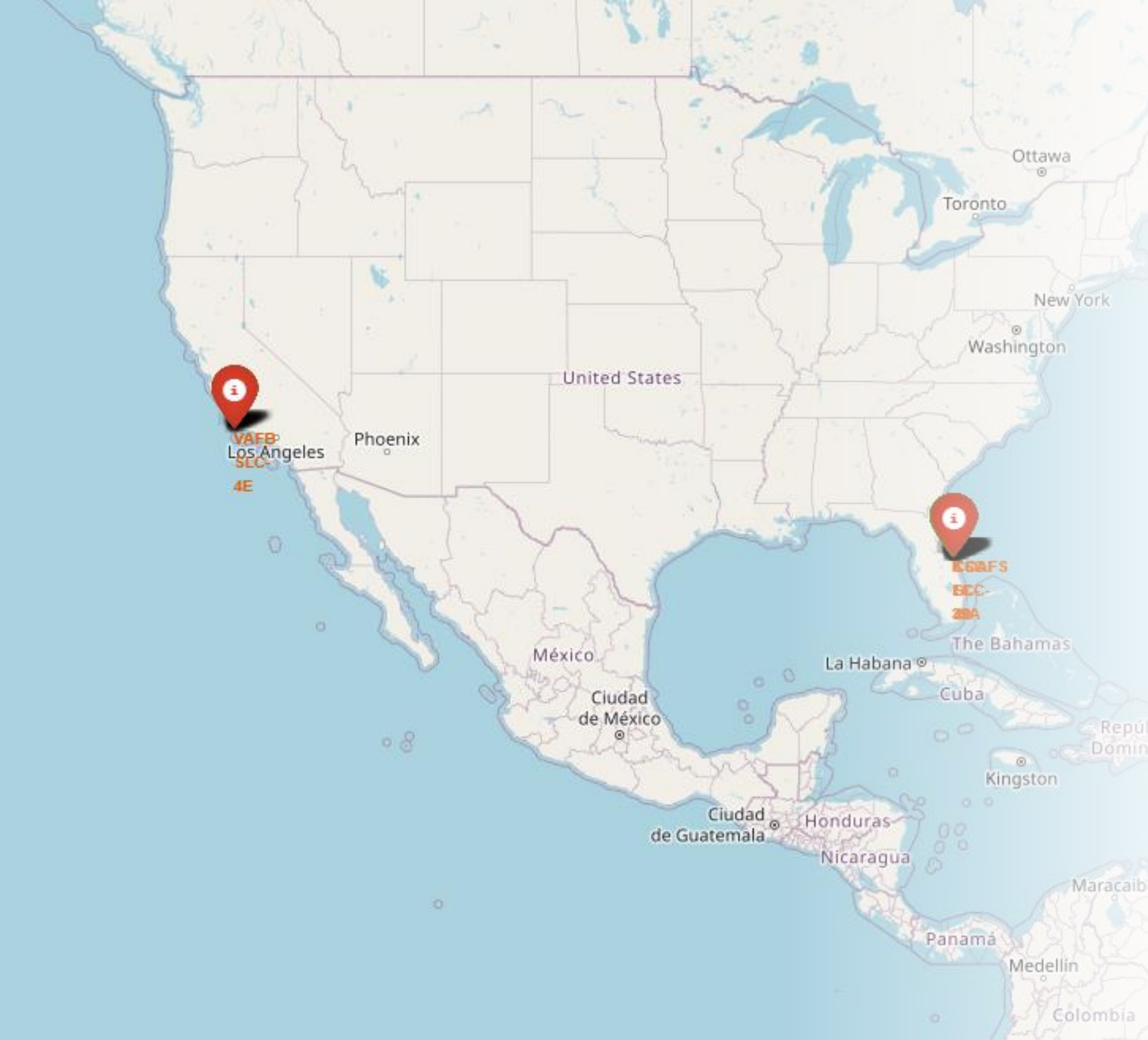
	Landing_Outcome	Outcome_Count
0	No attempt	10
1	Success (drone ship)	5
2	Failure (drone ship)	5
3	Success (ground pad)	3
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Failure (parachute)	2
7	Precluded (drone ship)	1

My first impression when I started working on this capstone was that landing in the ocean would be harder than on ground, excluding risk evaluations. But SpaceX managed to get a good success rate early on, and 2015 marked their first successful landing.

Section 3

Launch Sites Proximities Analysis

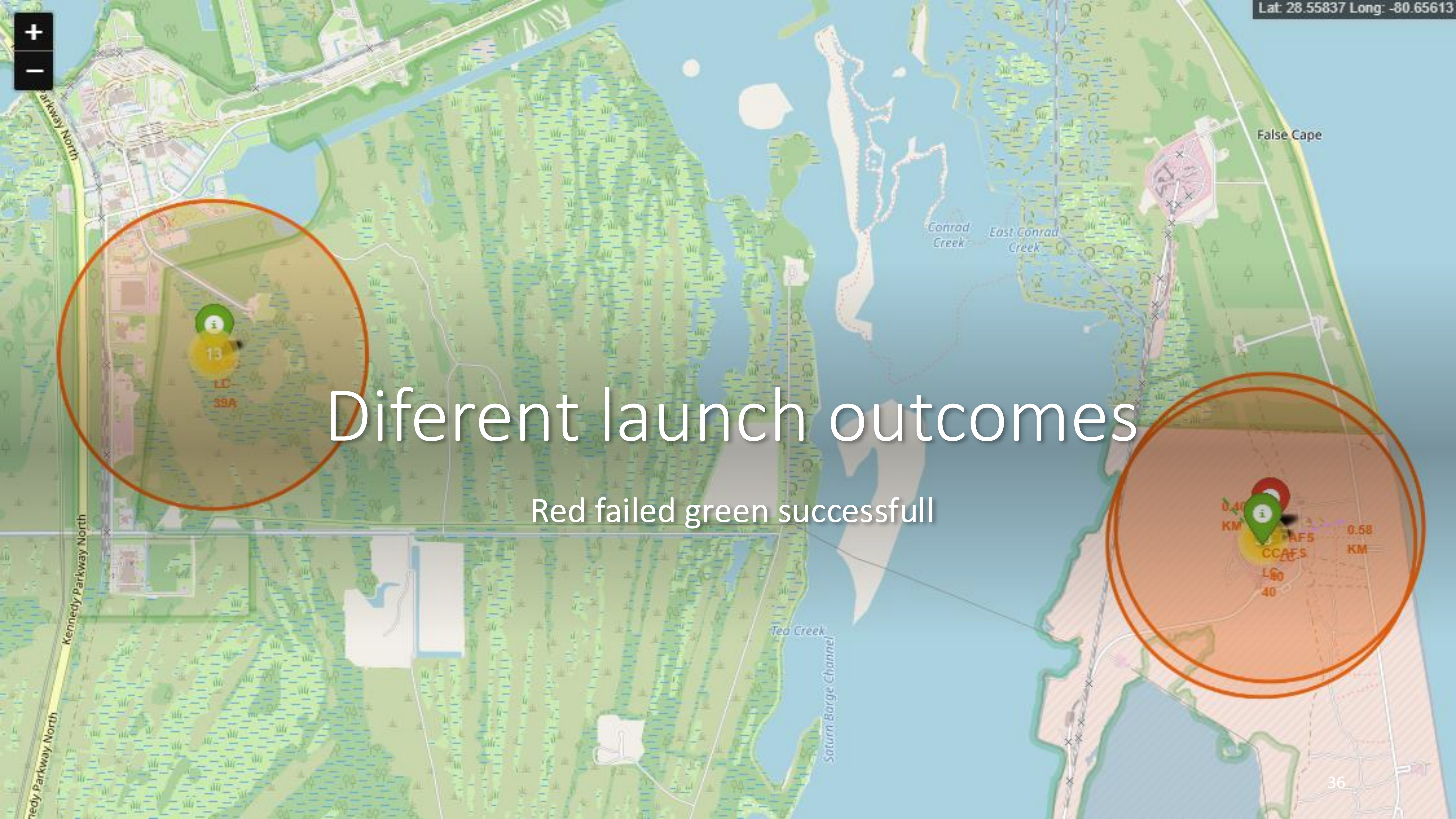
Launch Sites



- As we can see, there are two main regions for launching SpaceX rockets: the East Coast and the West Coast. Their geopositions are similar, but they're on different oceans with different conditions.

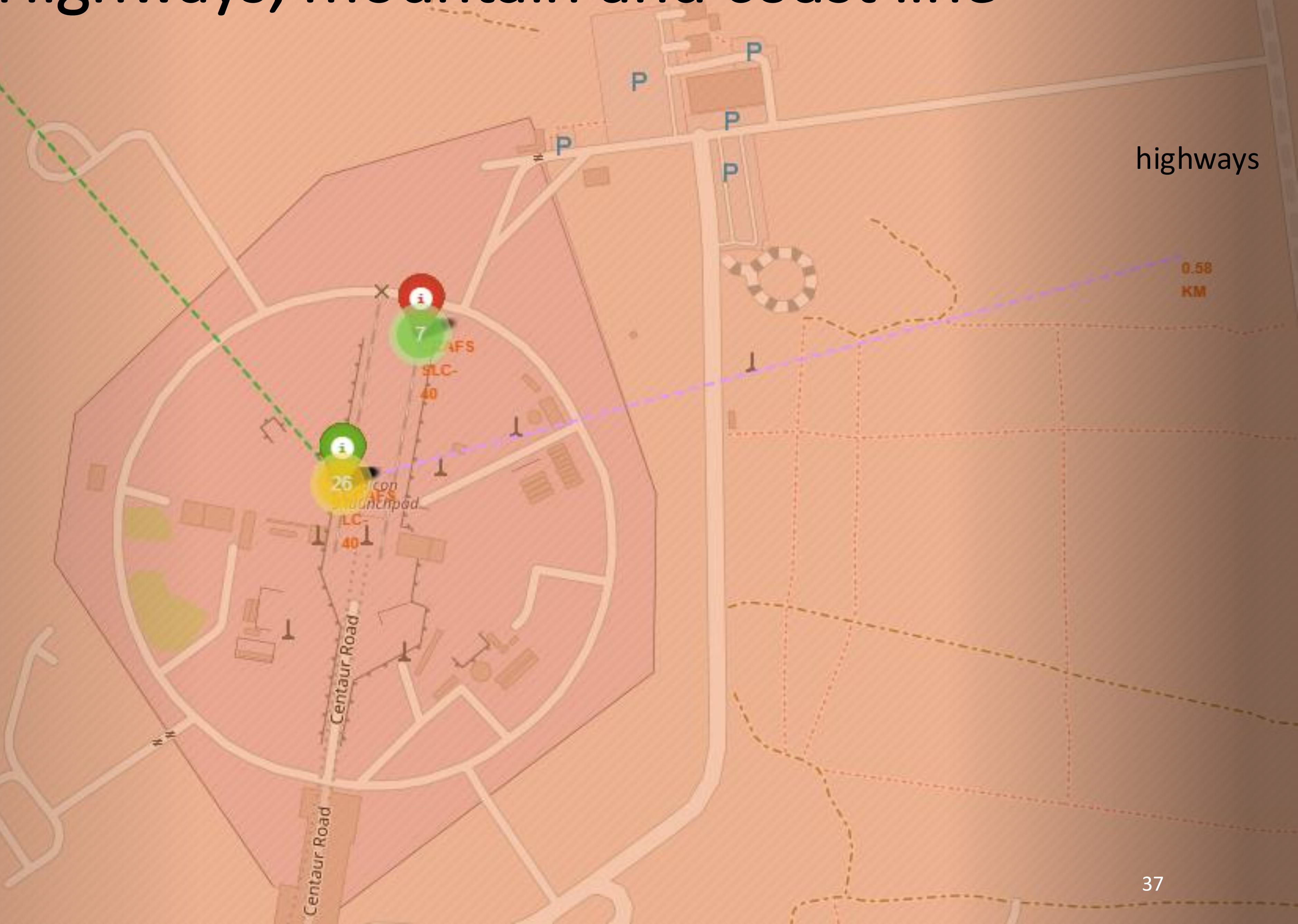
Diferent launch outcomes

Red failed green successfull



Distances to Highways, mountain and coast line

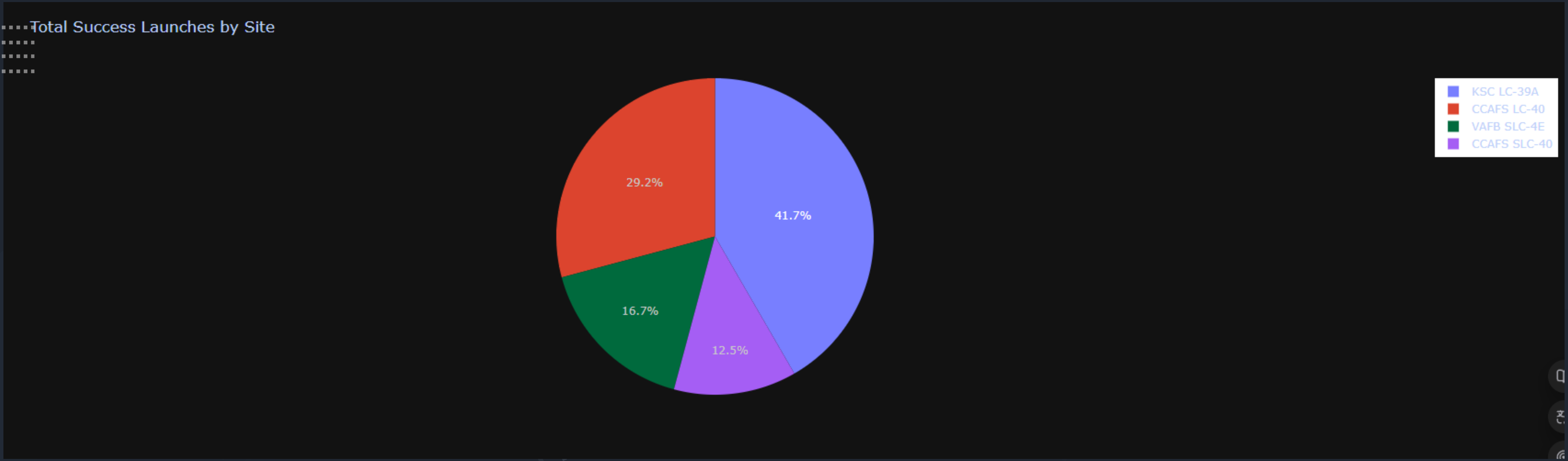
mountain peak



As we can see from the geospatial analysis in the capstone, SpaceX's launch sites are strategically placed with varying proximities to highways (for logistics and evacuations), mountains (which can influence weather and noise but are generally avoided for safety), and coastlines (key for over-ocean trajectories and drone ship recoveries).

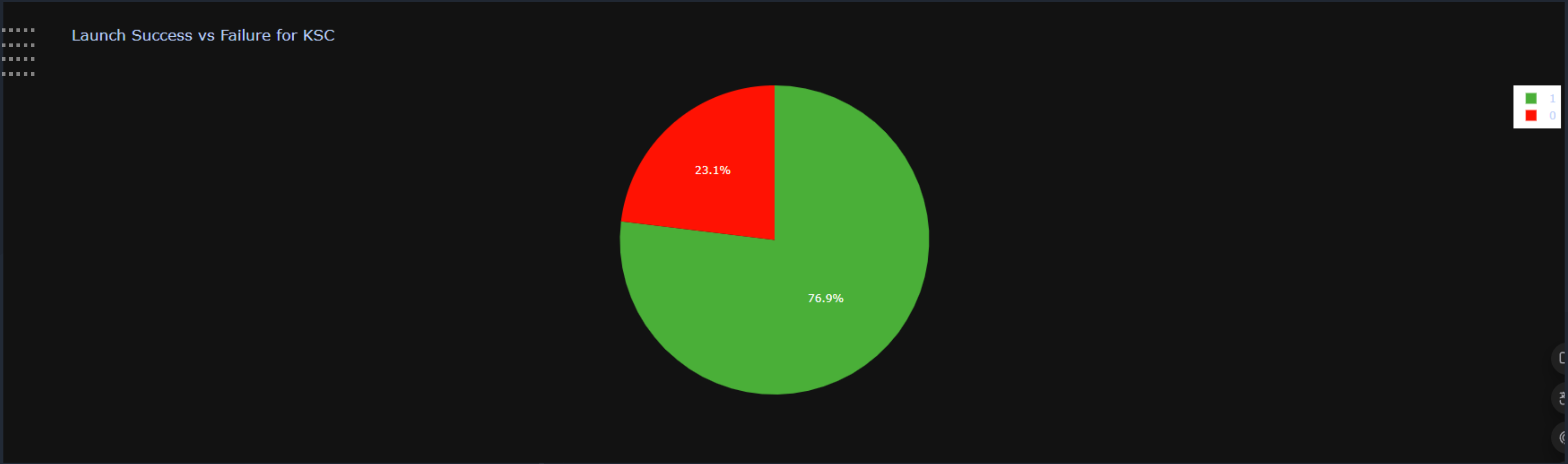
Section 4

Build a Dashboard with Plotly Dash



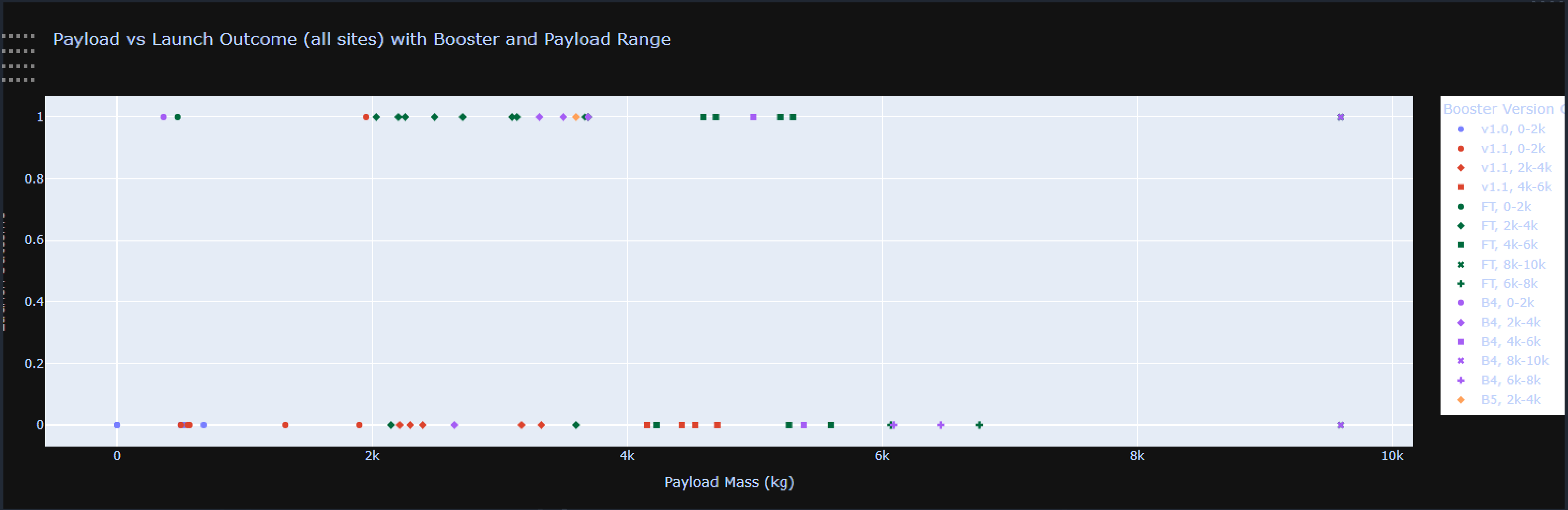
launch success count for all sites

- 41.7 % to KSC It is the best place to launch.
- CCAFS has the lower rate.
- 70% of the success come from only two places, East Coast.



launch site withhighest launch success ratio

- 76.9% success rate.
- In principle, the landing platforms for the first stage of the Falcon 9 located on the East Coast yield better results, and among them, with a success rate of 76.9, is KSC, a good place to start our SpaceY project.



Payload vs. Launch Outcome

with different all payload payload as symbols

- In principle, it seems that almost all types of payloads failed at least once, reinforcing the idea of the learning curve as a determining factor in the final outcome. It should be noted that not all failures are truly failures, as sometimes the recovery of the first stage is intentionally discarded, so it would be necessary to understand what factors determine such a decision

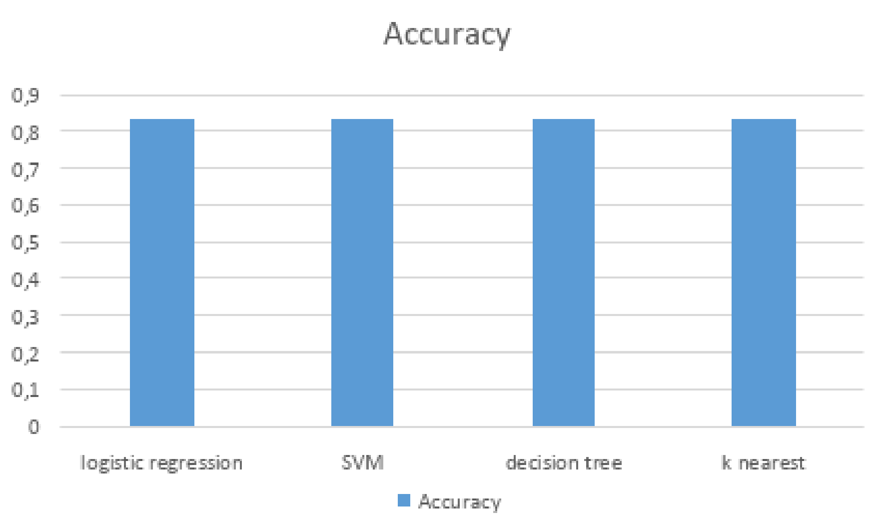
Section 5

Predictive Analysis (Classification)

Classification Accuracy

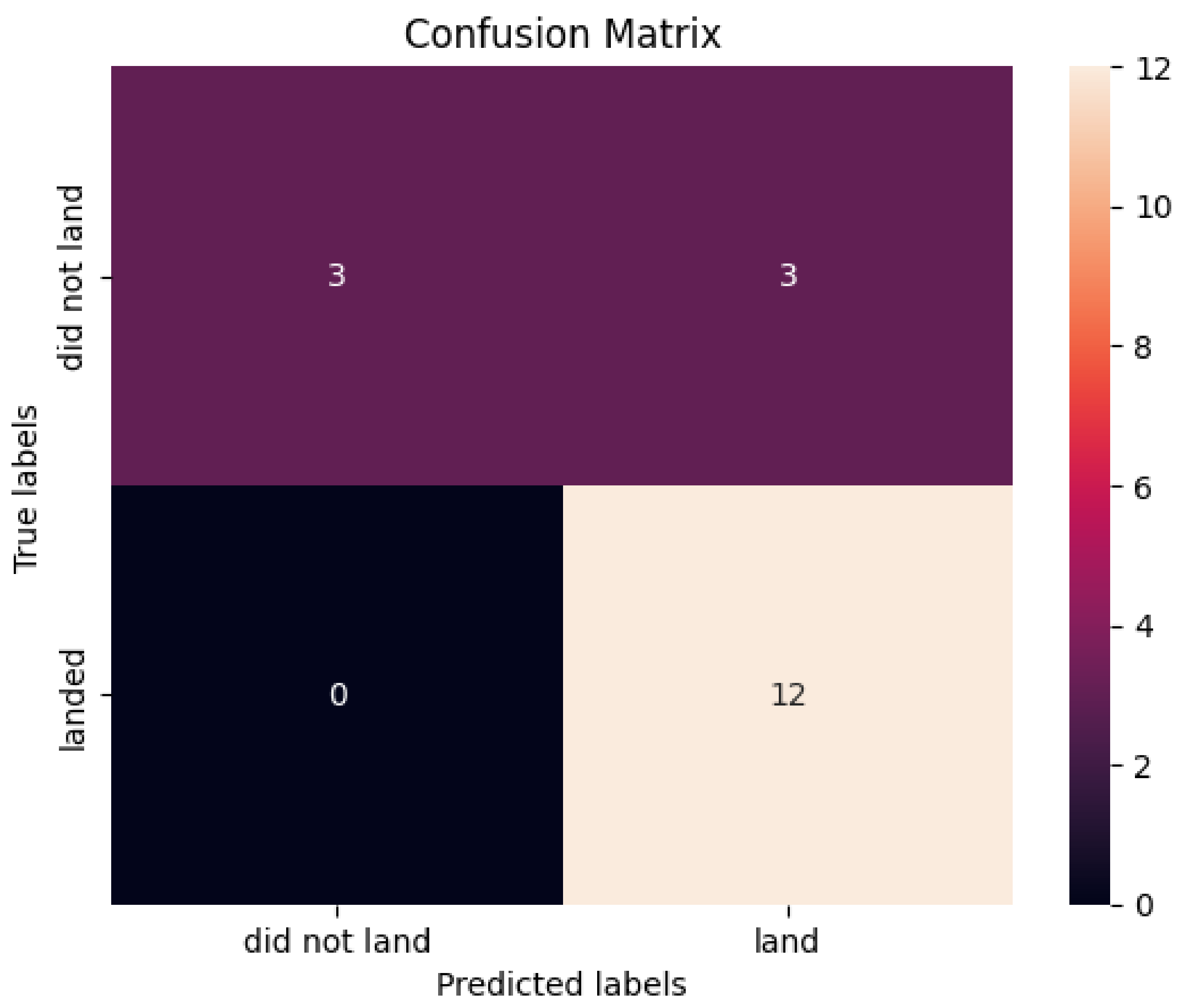
Logistic regression had the best performance after the validation test, while the decision tree scored higher on the validation data at 0.876—maybe despite the tree's drop in efficiency during the final test. I think deleting the early launches before 2015 will improve the model's performance, as those early failures are tied to the learning curve and not to the variables we're studying.

But that thought can be applied to the other models too, as they share the same score of 0.833.



Confusion Matrix

The confusion matrix shows that the model predicted the first-stage rocket would be recovered 12 times, while in reality, it wasn't recovered 3 times. All models produce the same matrix since, after 2015, landings are typically successful. That's why I think for Space Y, we should segment the model into two phases: one for the learning curve and another for once the first recovery is consolidated.



Conclusions

Point 1 The success of rocket recovery depends on the company's level of learning.

Point 2 The GTO and ISS orbits are the most complicated.

Point 3 Larger payloads have a higher likelihood of success.

Point 4 Landing at sea seems like the best option to start with.

… It would be necessary to provide a detailed explanation of why SpaceX's recovery success decreased in 2018.

Appendix

- The notebooks are stored in the cloud; for more information, follow the link
- https://github.com/Glorfindiel/IBM-DS/tree/main

Thank you!