IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Edder Gutierrez>
<April 30th, 2025>

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection methodology by API and Web Scrapping:
  - Data wrangling
  - Exploratory Data Analysis (EDA) using visualization and SQL
  - Interactive visual analytics using Folium and Plotly Dash
  - Machine Learning Predictive Analysis
- Summary of all results
  - EDA Result
  - Interactive Dashboard
  - Predictive Analysis Result

Introduction

- Project background and context
    - The Goal is to determine the cost of a launch of the Falcon 9. 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
    - Predict if the Falcon 9 first stage will land successfully, to be able to determine the cost of a launch
    - What Factors determine if the rocket will land successfully
    - Iterate among various features that determine the successful landing

Section 1

Methodology

convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful

into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

Methodology

Executive Summary

- Data collection methodology:
    - By API: through a get request to the SpaceX API
    - By Web scraping: getting from Wikipedia the Falcon 9 and Falcon Heavy Launches Records
- Perform data wrangling
    - Convert the landing outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful by using One-hot encoding
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    - Find Hyperparameter for SVM, Classification Trees and Logistic Regression and calculates the Accuracy of each of them

Data Collection

- Data Collection methology.
  - By API: through a get request to the SpaceX API
  - Decoded the response as a JSON (Java Script Object Notation) using json() function and transforming into a Pandas data-frame using .json_normalize()
  - Checked for missing values by .isnull() function
  - Also using BeutifulSoup we performed web scrapping to Falcon 9 from Wikipedia. The Goal was to extract the records as HTML table and parse and convert to a Pandas data-frame for future analysis

Data Collection – SpaceX API

- Get request was used to the SpaceX API to data collection, clean and some data wrangling

- GitHub URL link:

https://github.com/e-spec/Applied-Data-Science/blob/main/1.spacex-API-data-collection.ipynb

Flowchart of SpaceX API

```
[39]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

We should see that the request was successfull with the 200 status response code

```
[40]: response=requests.get(static_json_url)
```

```
[41]: response.status_code
```

```
[41]: 200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
[42]: # Use json_normalize meethod to convert the json result into a dataframe
data=pd.json_normalize(response.json())
```

Task 3: Dealing with Missing Values

Calculate below the mean for the `PayloadMass` using the `.mean()`. Then use the mean and the `.replace()` function to replace `np.nan` values in the data with the mean you calculated.

```
[73]: # Calculate the mean value of PayloadMass column
mean_payload_mass = data_falcon9['PayloadMass'].mean()

# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan, mean_payload_mass, inplace=True)
```

Data Collection - Scraping

- Using BeatifuSoup we web scrap Falcon9 from Wikipedia

- GitHub URL:

- https://github.com/e-spec/Applied-Data-Science/blob/main/2.spacex-WebScrap-data-collection.ipynb

Flowchart of web scraping

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

[6]:

```python
# use requests.get() method with the provided static_url
# assign the response to a object
response=requests.get(static_url)
if response.status_code == 200:
    html_content = response.text
    print("Request successful!")
    #print(response.text)  # Print the HTML content of the page
else:
    print(f"Request failed with status code: {response.status_code}")
```

```
Request successful!
```

Create a `BeautifulSoup` object from the HTML `response`

[7]:

```python
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup=BeautifulSoup(html_content,'html.parser')
```

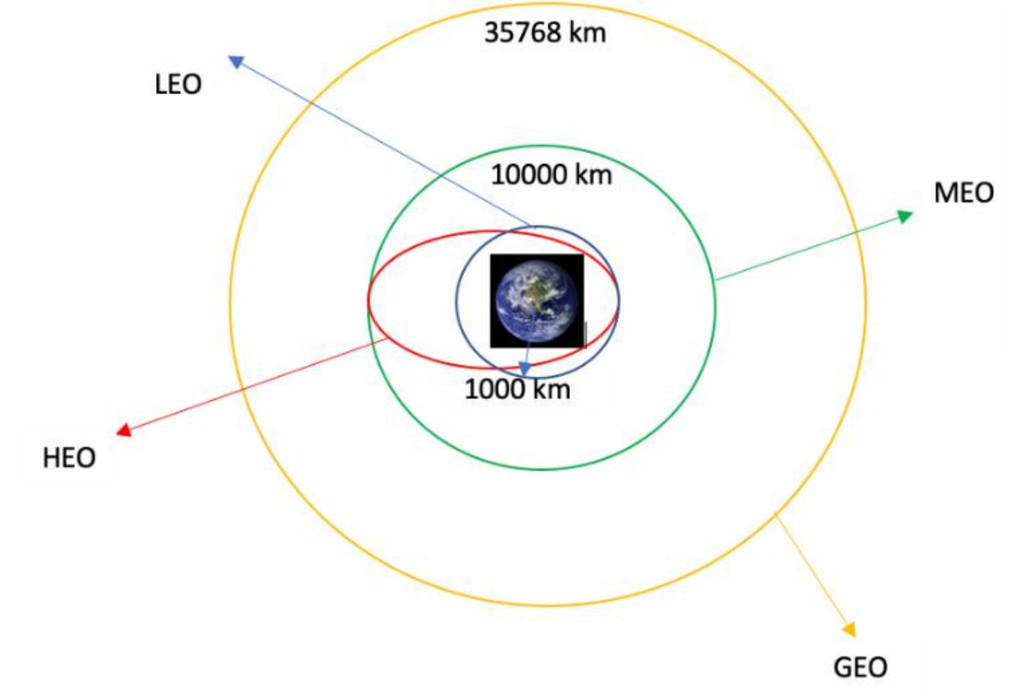
Next, we just need to iterate through the `<th>` elements and apply the provided `extract_column_from_header()` to extract column name one by one

[17]:

```python
# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
# Assuming first_launch_table is a BeautifulSoup object for the relevant table
column_names = []

# Find all <th> elements in the table
for th in first_launch_table.find_all('th'):
    # Apply the provided extract_column_from_header() function
    name = extract_column_from_header(th)

    # Check if the extracted name is non-empty and append it to the list
    if name is not None and len(name) > 0:
        column_names.append(name)

# Print the resulting list of column names
print("Extracted Column Names:")
print(column_names)
```

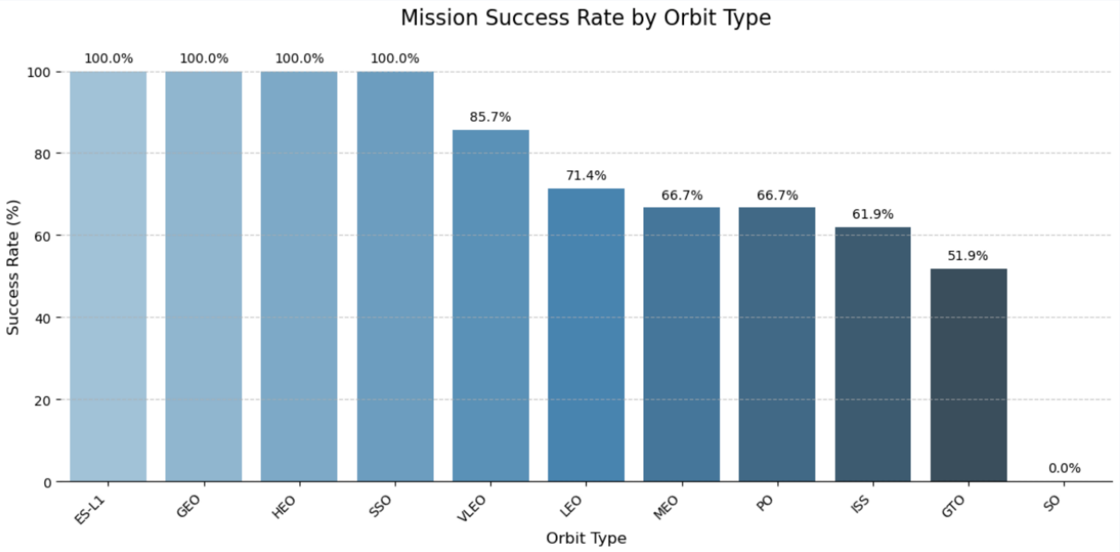
Data Wrangling

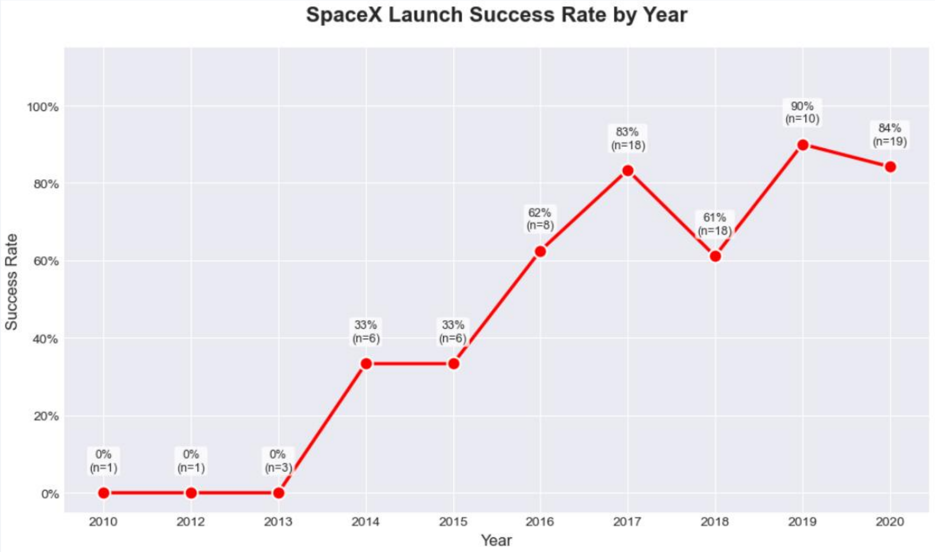
- EDA was done to determine the training labels
- The number of launched was calculated for each location and the number of orbits for each of them. See image for clarification
- We created a landing outcome label
- GitHub URL:
    - https://github.com/e-spec/Applied-Data-Science/blob/main/3.spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- We visualized the data to explore the relationship flight number and payload, launch site. Also Launch Site vs Payload.
- Success rate for each orbit type and the launch success yearly trend. See plots below:





- GitHub URL: https://github.com/e-spec/Applied-Data-Science/blob/main/5.SpaceX_%20EDA_with_Matplotlib_and_Pandas.ipynb

EDA with SQL

- SQL queries performed:
  - Display the names of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string 'CCA’
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version F9 v1.1
  - List the date when the first succesful landing outcome in ground pad was acheived.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - List the total number of successful and failure mission outcomes
  - List all the booster_versions that have carried the maximum payload mass. Use a subquery.
  - List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
  - 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.
- GitHub URL: https://github.com/e-spec/Applied-Data-Science/blob/main/4.SpaceX_%20EDA_with_SQL.ipynb

Build an Interactive Map with Folium

- All launch sites were added along with map objects such as markers, circles, successful and failure launched also were added to the folium map.
- Launch outcomes also added and using color labeled for high success rate.
- Added distances between a launch sites and items in the proximity to answer what is around the launch sites and make sure the launchings have certain distances away from cities.
- GitHub URL: https://github.com/e-spec/Applied-Data-Science/blob/main/6.%20Launch_site_location_FOLIUM.ipynb

Build a Dashboard with Plotly Dash

- Interactive dashboard was built with Plotly dash:
  - Pie charts showing the total number of launches
  - Plotted scatter graph showing the relationship with Outcome and Payload Mass (kg) for the different booster version categories
- This dashboard allows to see the following:
  - Largest Successful Launches Site: KSC LC-39A
  - Highest Launch Success Rate Site: KSC LC-39A
  - Payload Range with Highest Success Rate: (2000, 4000]
  - Payload Range with Lowest Success Rate: (6000, 8000]
  - Booster Version with Highest Success Rate: B5
- GitHub URL: https://github.com/e-spec/Applied-Data-Science/blob/main/7.%20Dashboard_Plotly.ipynb

Predictive Analysis (Classification)

- The data was loaded using pandas/numpy, transform (stabdardize) the data, and split into training and testing.
- Four different machine learning (SVM, Classification Trees, Logistic Regression and KNN) were built and tuned the hyperparameters using GridSearchCV
- We used accuracy as the metric and confusion matrix
- We evaluated the model and were able to see the one with best performance
- GitHub URL: https://github.com/e-spec/Applied-Data-Science/blob/main/8.SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

- Predictive analysis results

TASK 12

Find the method performs best:

[37]:
```
#All of them perform the same.
print(test_accuracy_log)
print(test_accuracy_svm)
print(test_accuracy_tree)
print(test_accuracy_knn)
```

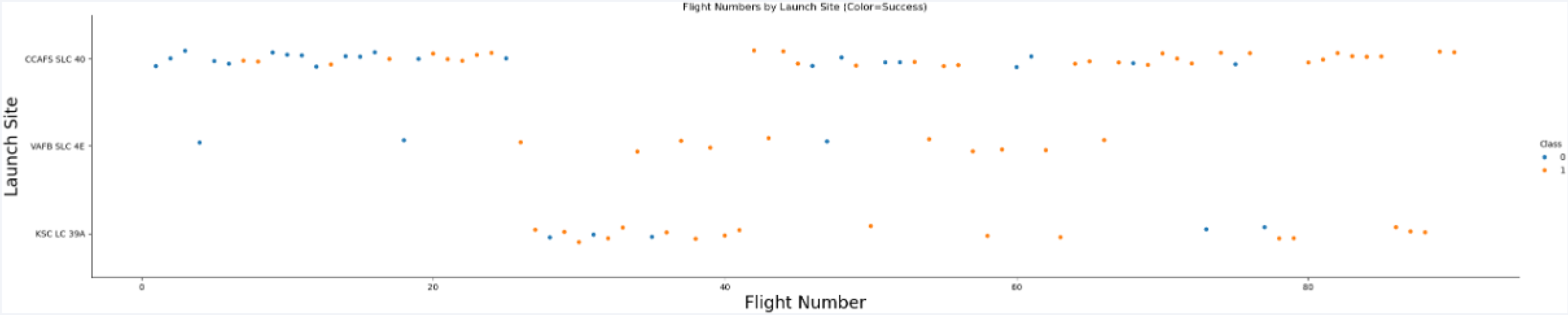
```
0.8333333333333334
0.8333333333333334
0.7777777777777778
0.8333333333333334
```

Section 2

Insights drawn from EDA

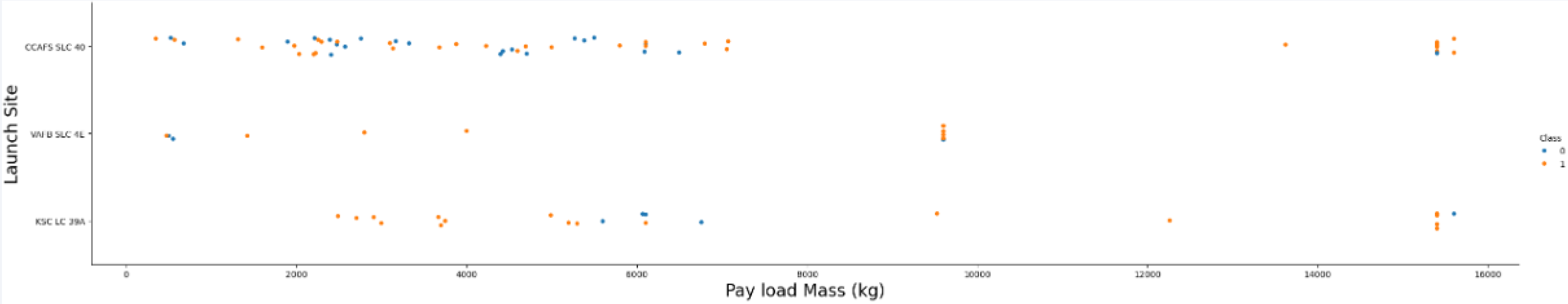
Flight Number vs. Launch Site

- The larger the flight amount at a specific launch site, the greater the success rate at that location



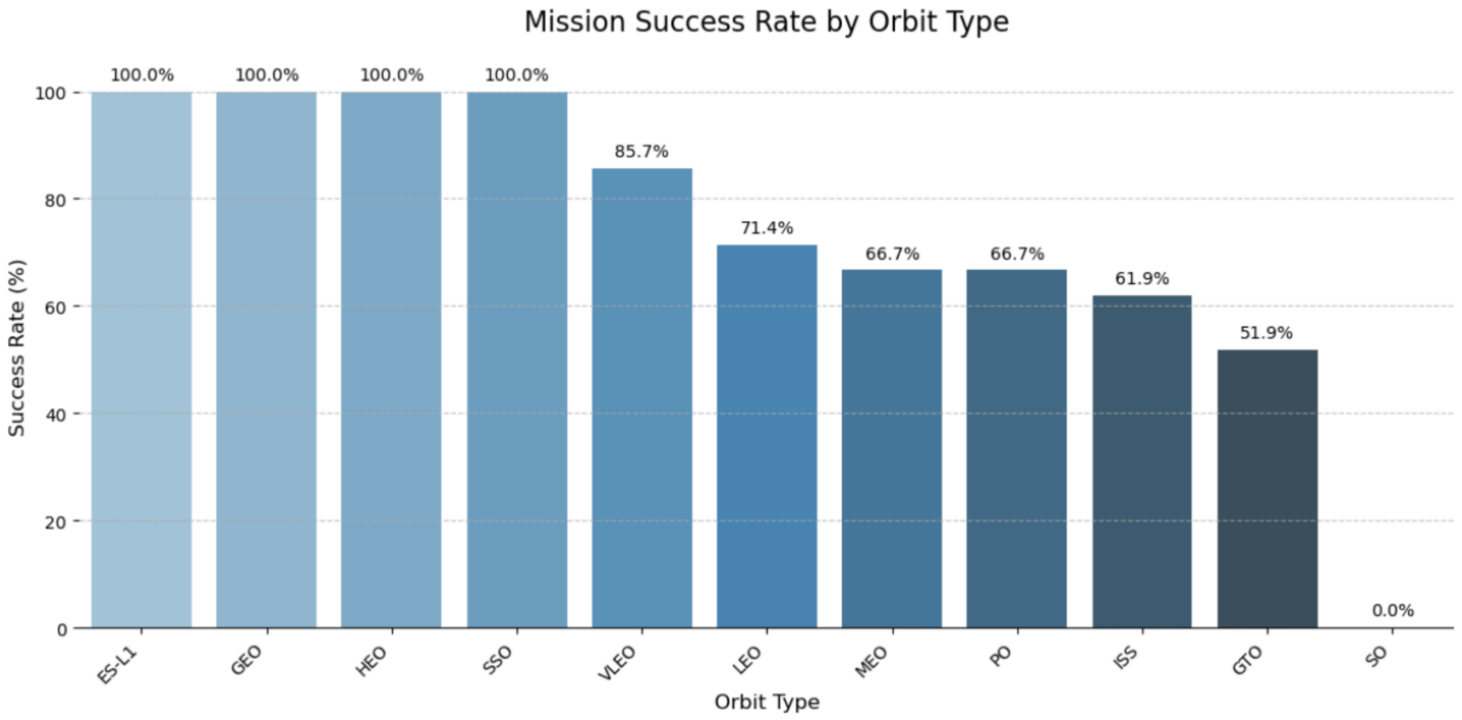
Payload vs. Launch Site

- The greater the payload mass the higher the success rate for the launch locations



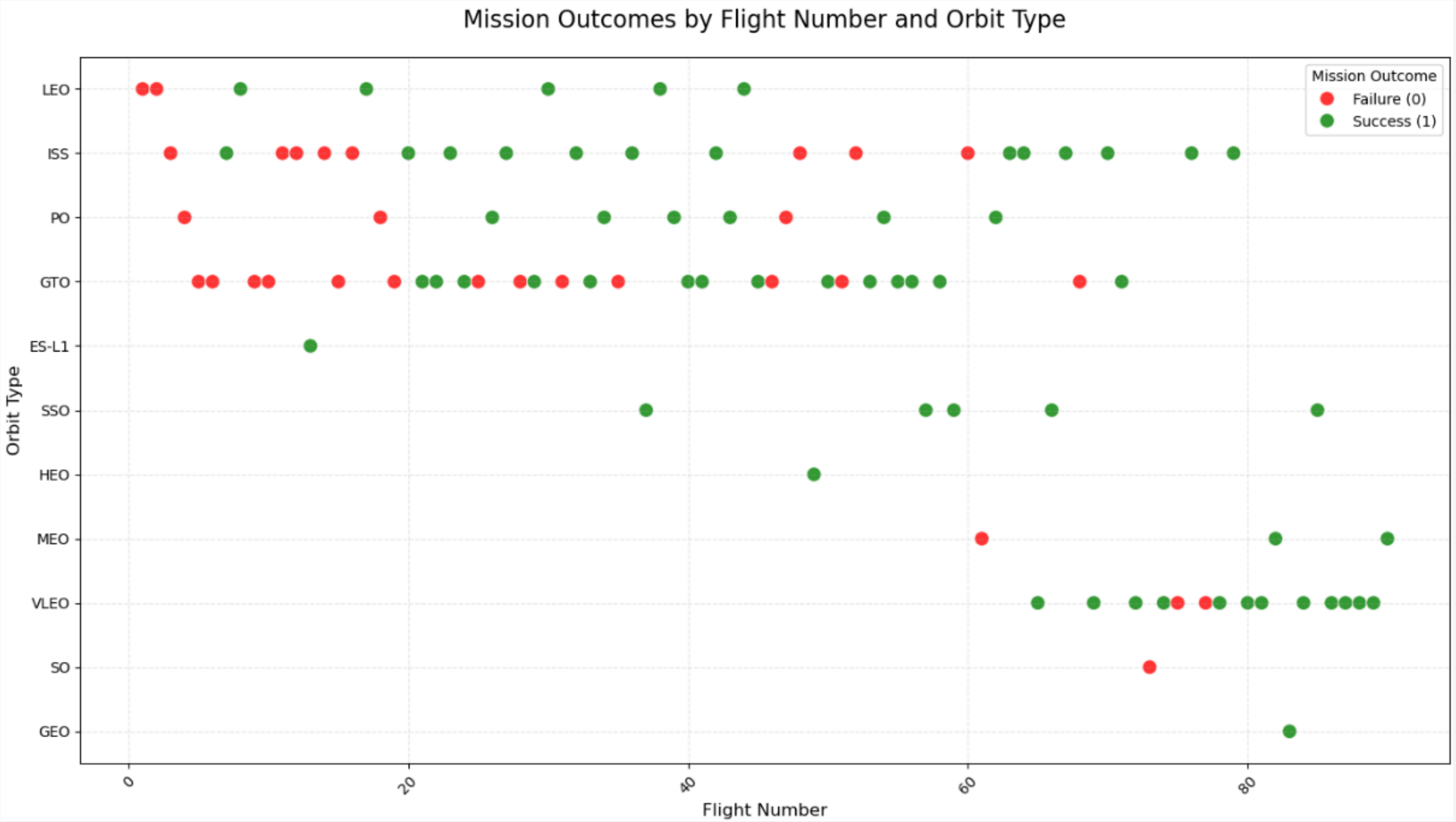
Success Rate vs. Orbit Type

- We can see that ES-L1, GEO, HEO, SSO have the most success rate



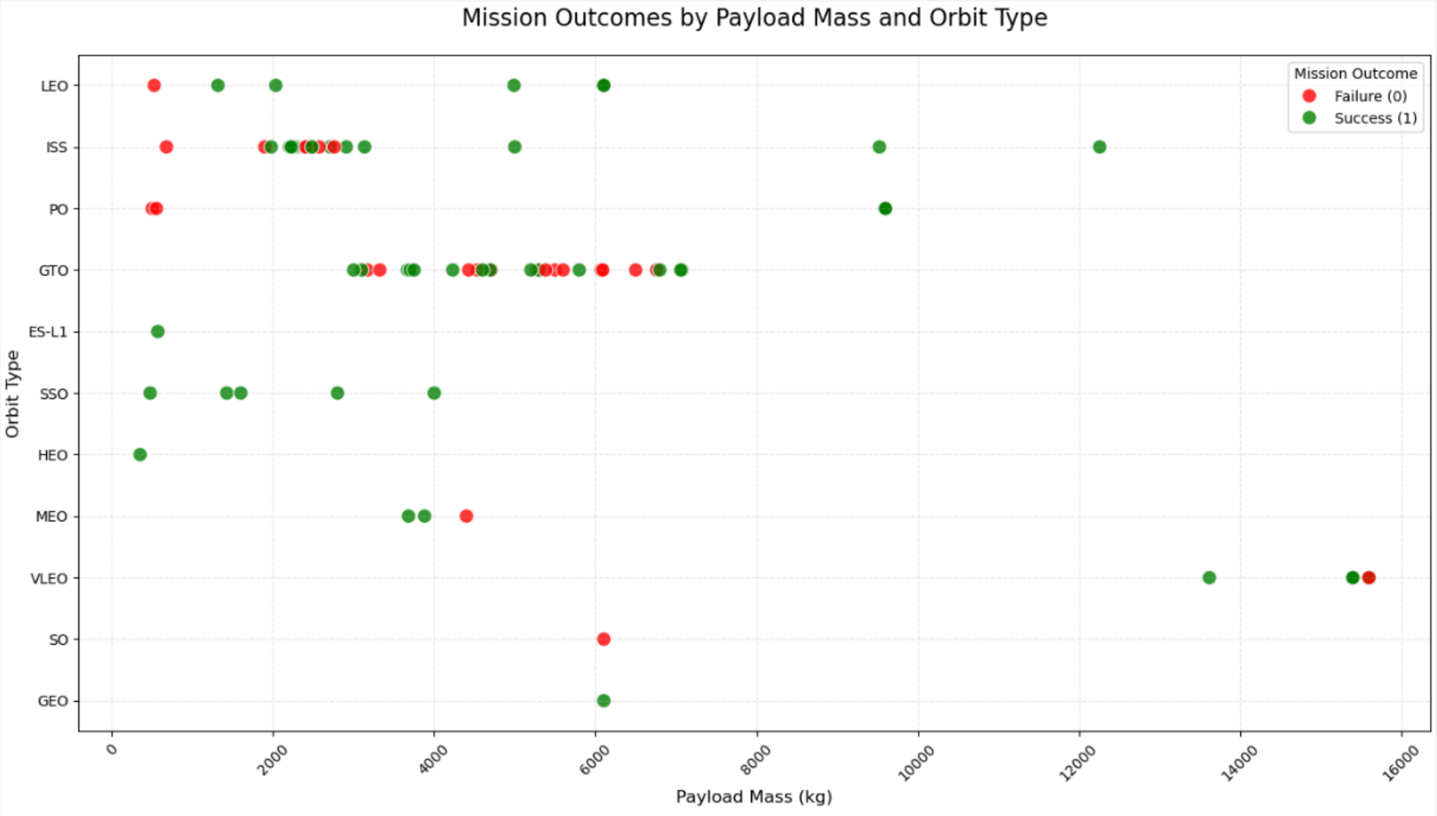
Flight Number vs. Orbit Type

- For LEO Orbit: the success rate is correlated with the number of flights, whereas for GEO there is not correlation



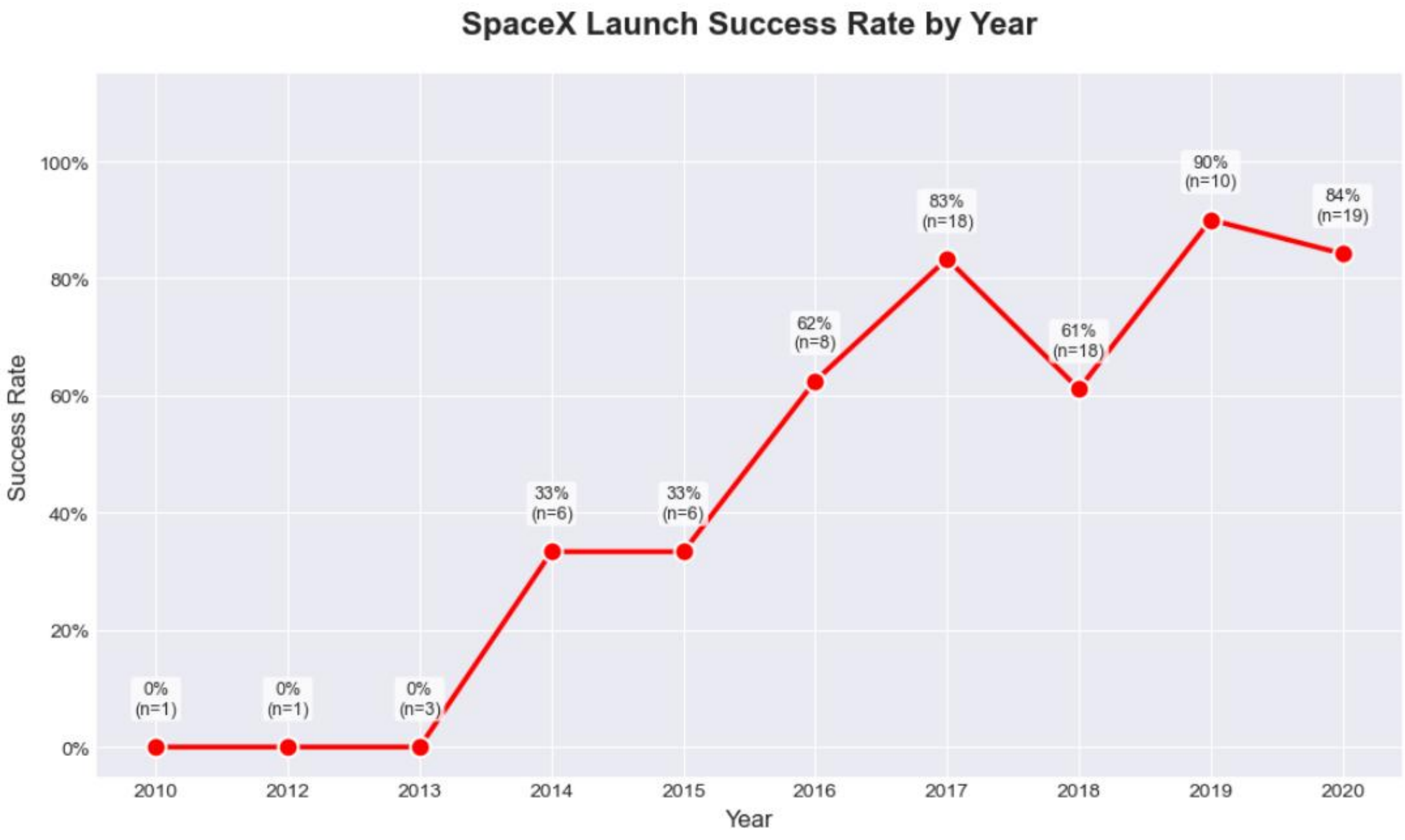
Payload vs. Orbit Type

- For heavy Payload Mass the Orbit LEO, ISS and PO has more successful landings



Launch Success Yearly Trend

- Since 2013 the success rate is constantly increasing



All Launch Site Names

DISTINCT key word allows to have the unique launch sites

```
[15]: %sql select distinct Launch_Site from SPACEXTABLE;
 * sqlite:///my_data1.db
Done.
```

[15]:

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

Launch Site Names Begin with 'CCA'

This query was used to find 5 records where launch sites begin with `CCA`

Task 2 ¶

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

[16]:
```
sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5;
```

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

[16]:

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

- This query calculate the total payload carried by boosters from NASA

Task 3

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

[34]:
```
#%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer='NASA (CRS)';
```

[39]:
```
%%sql SELECT SUM(PAYLOAD_MASS__KG_)
AS Total_Payload_Mass
FROM SPACEXTABLE
WHERE Customer
LIKE '%NASA (CRS)%';
```

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

[39]:

Total_Payload_Mass
48213

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

Task 4

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

[42]:
```
#%sql SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass_for_F9v1 FROM SPACEXTABLE WHERE Booster_Version LIKE '%F9 v1.1%';
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass_for_F9v1 FROM SPACEXTABLE WHERE Booster_Version='F9 v1.1';
```

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

[42]:

Average_Payload_Mass_for_F9v1
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad. This query was obtained by MIN function

```
[52]: %%sql SELECT MIN(Date)
      AS First_Successful_Landing
      FROM SPACEXTABLE
      WHERE Landing_Outcome = 'Success (ground pad)';

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

[52]:

First_Successful_Landing
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. This query was obtained combining WHERE and AND.

```
[54]: %%sql SELECT Booster_Version AS Boosters_with_success_in_drone_ship
      FROM SPACEXTABLE
      WHERE Landing_Outcome = 'Success (drone ship)'
      BETWEEN PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;

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

[54]:

Boosters_with_success_in_drone_ship
F9 v1.0 B0003
F9 v1.0 B0004
F9 v1.0 B0005
F9 v1.0 B0006
F9 v1.0 B0007
F9 v1.1 B1003

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes. This query obtained using COUNT and GROUPBY

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

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

[26]:

Mission_Outcome	TotalNo_Successful_and_Failure_Mission_Outcomes
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. Query obtained using a Subquery

Task 8

List all the booster_versions that have carried the maximum payload mass. Use a subquery.

[61]:
```
# The subquery gets the maximum payload mass from the table.
# The outer query finds all booster versions that carried that exact mass
```

[27]:
```
%%sql SELECT Booster_Version
FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTABLE
);
```

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

[27]:

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5

2015 Launch Records

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

Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

[63]:
```
# substr(Date, 6, 2) AS Month: gets the month from the date.
# substr(Date, 1, 4) = '2015': filters records from the year 2015.
# Landing_Outcome LIKE 'Failure (drone ship)': selects failed landings on drone ships.
```

[28]:
```
%%sql SELECT
    substr(Date, 6, 2) AS Month,
    Booster_Version,
    Launch_Site,
    Landing_Outcome
FROM SPACEXTABLE
WHERE substr(Date, 1, 4) = '2015'
  AND Landing_Outcome LIKE 'Failure (drone ship)';
```

 * sqlite:///my_data1.db
Done.

[28]:

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

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.

[66]:
```
# COUNT(*): counts how many times each Landing_Outcome occurred.
# WHERE Date BETWEEN ...: restricts the records to the given date range.
# GROUP BY Landing_Outcome: groups the data by type of landing outcome.
# ORDER BY Outcome_Count DESC: sorts from most frequent to least.
```

[29]:
```
%%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.
```

[29]:

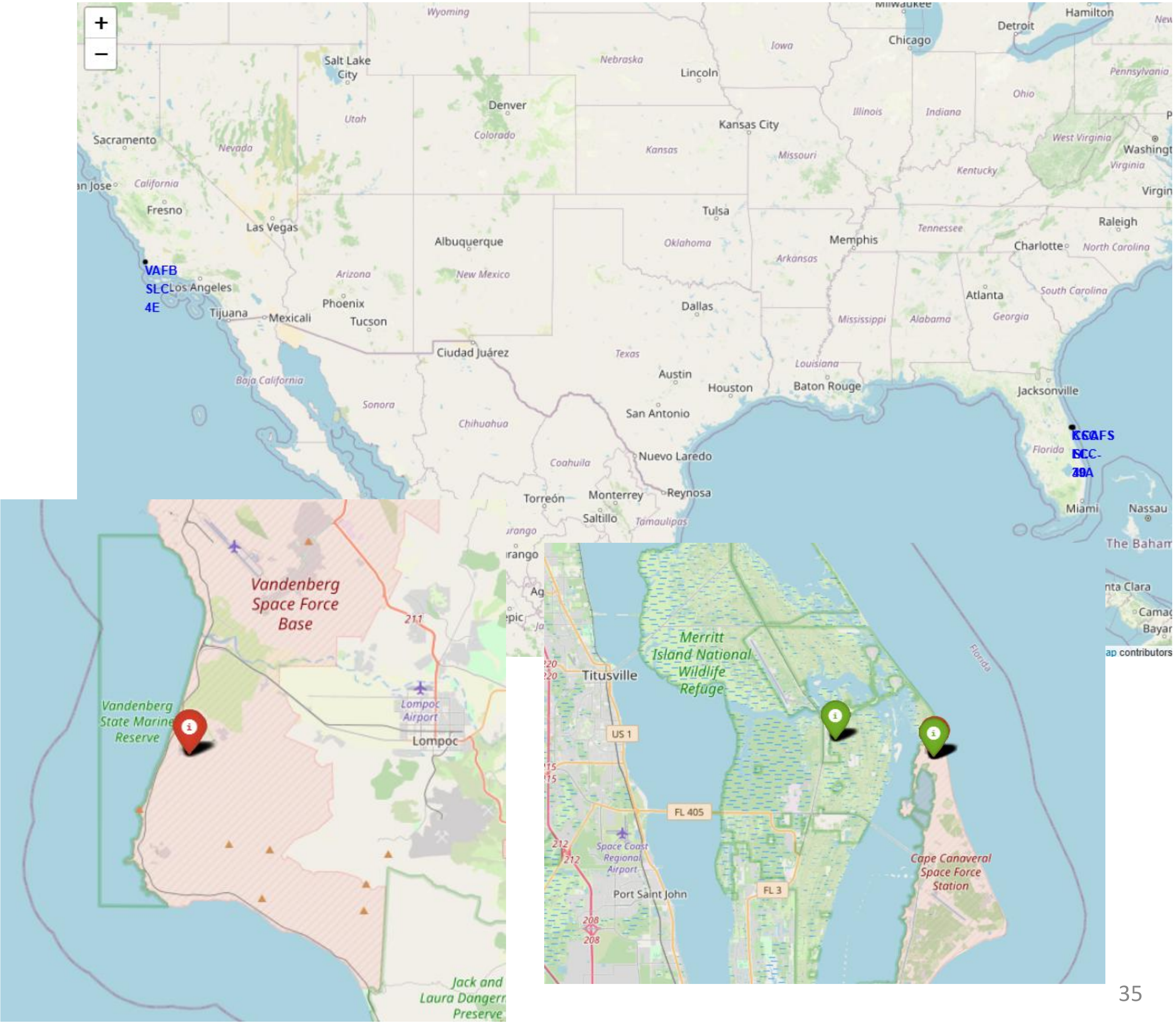
Landing_Outcome	Outcome_Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

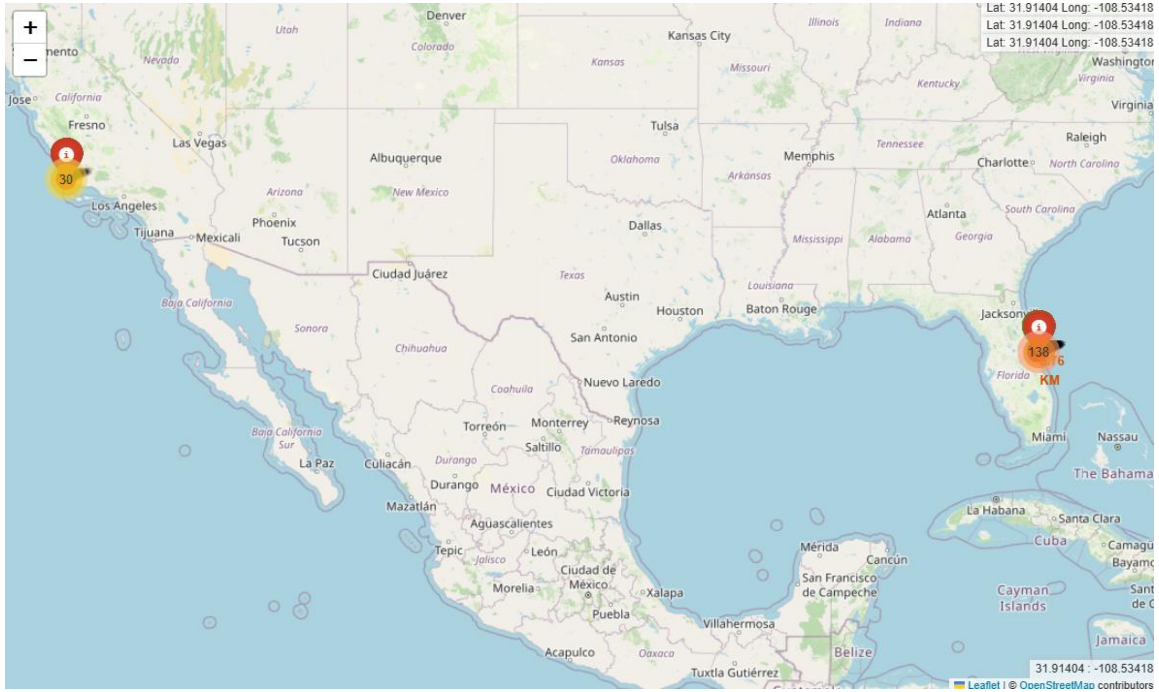
Launch Sites

- Launch sites are by the coast and in restricted areas

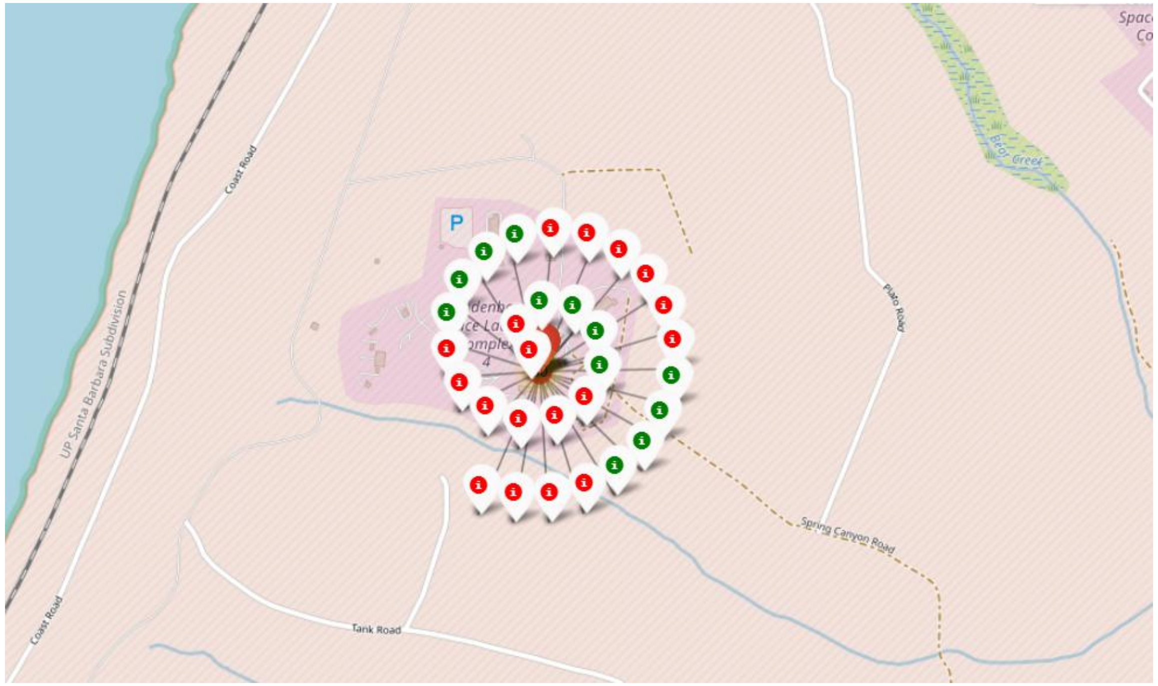


Success/Failed Launches

- Clusters for every launch site

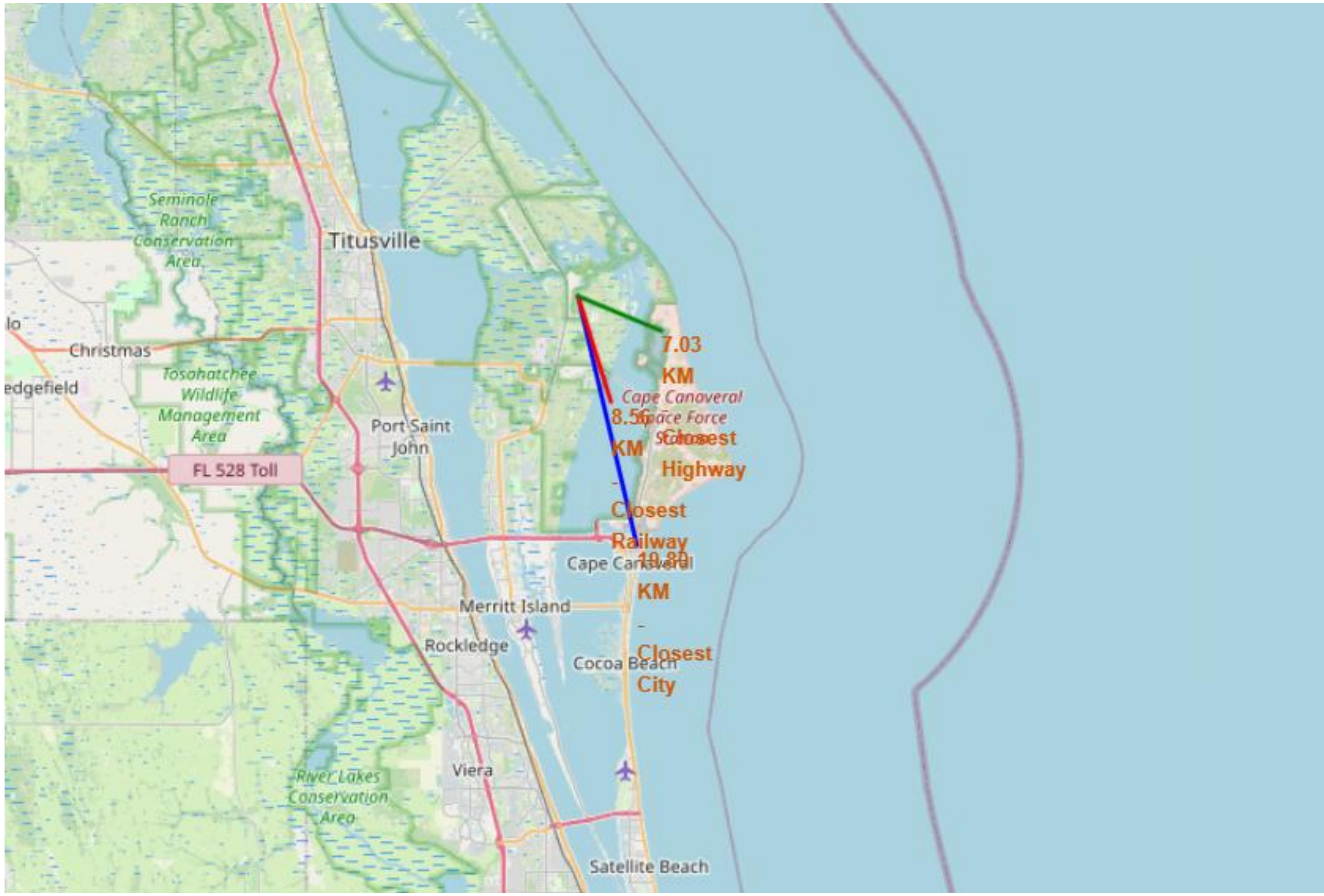


- Green Marker=Success
- Red Marker=Failure



Launch Sites and Proximities

- Near the launch sites we have railway, highway, coastal line. This secure a safe distance to cities/population

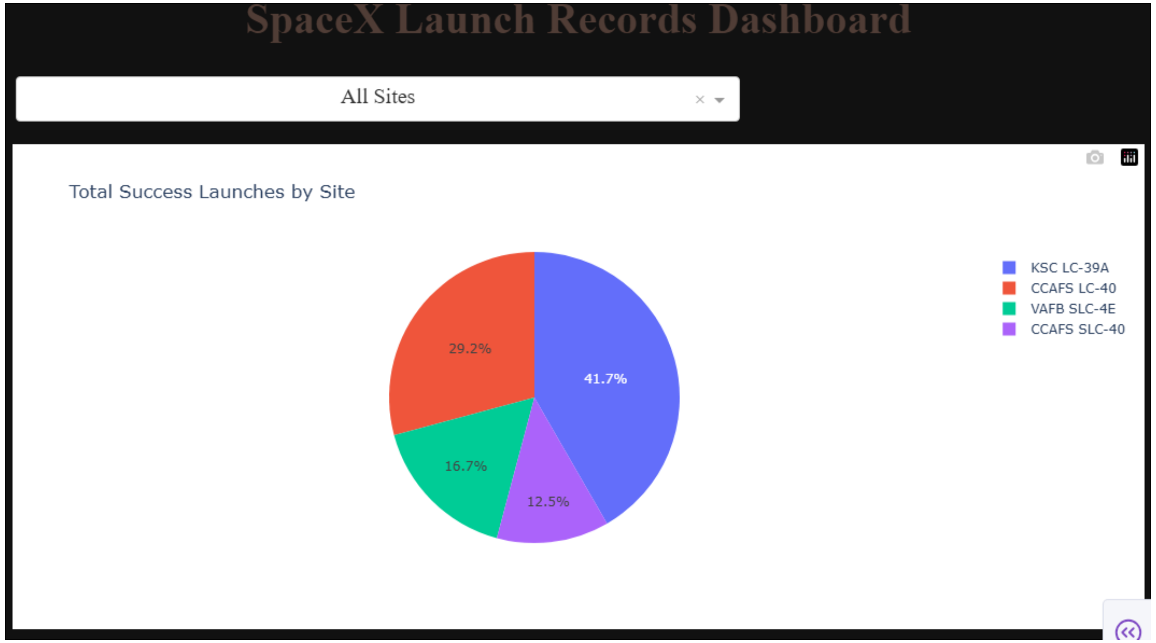


Section 4

Build a Dashboard with Plotly Dash

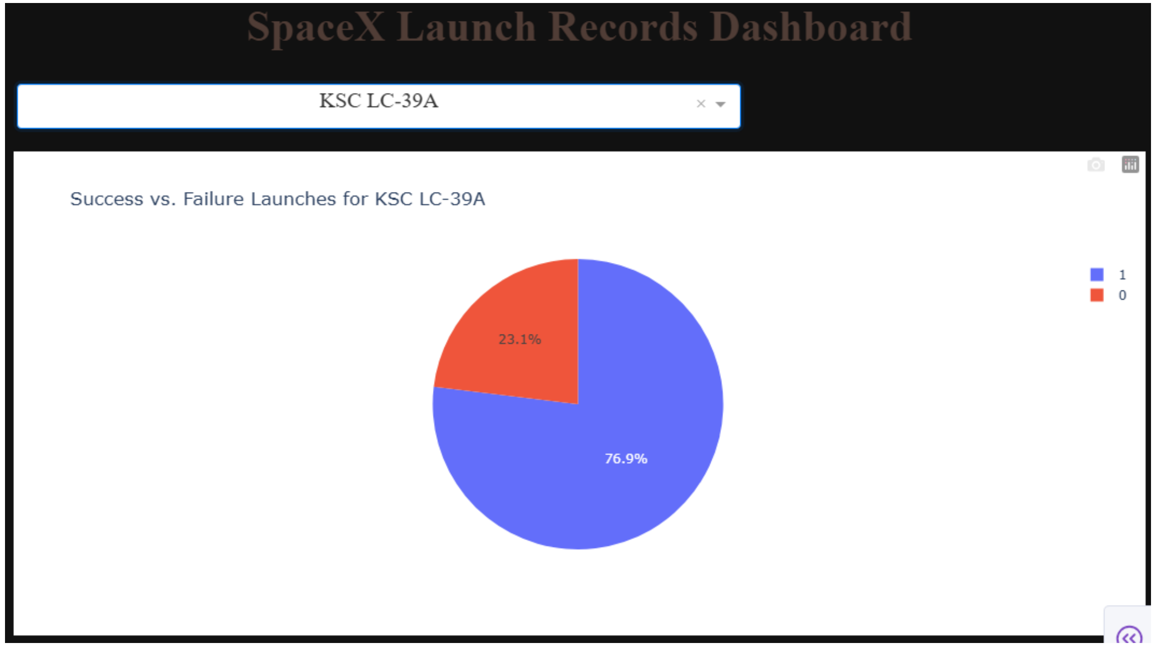
Total Success Launches by Site

- KSC LC-39A has the higher success followed by CCAFS LC-40



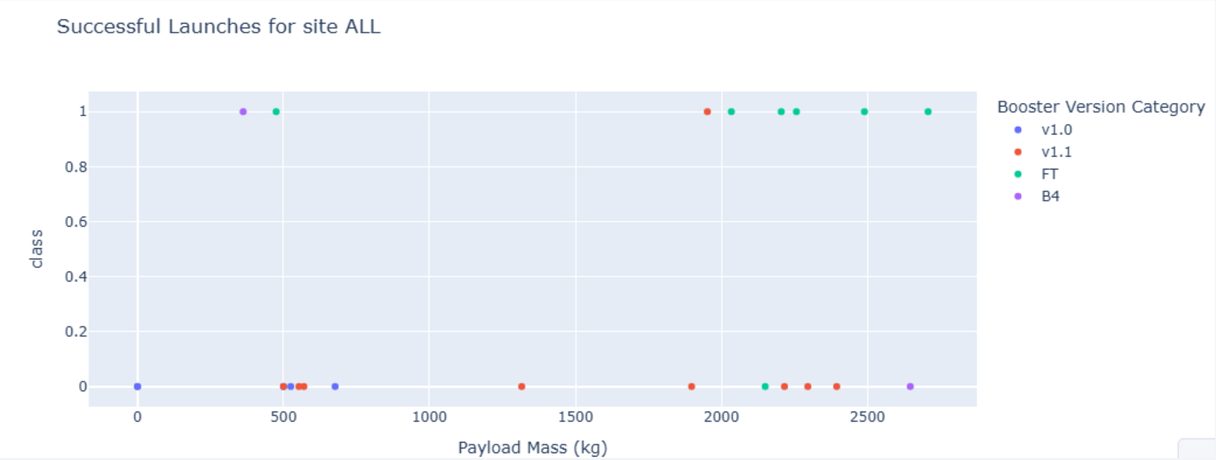
KSC LC-39A

- This is the site with the highest success rate

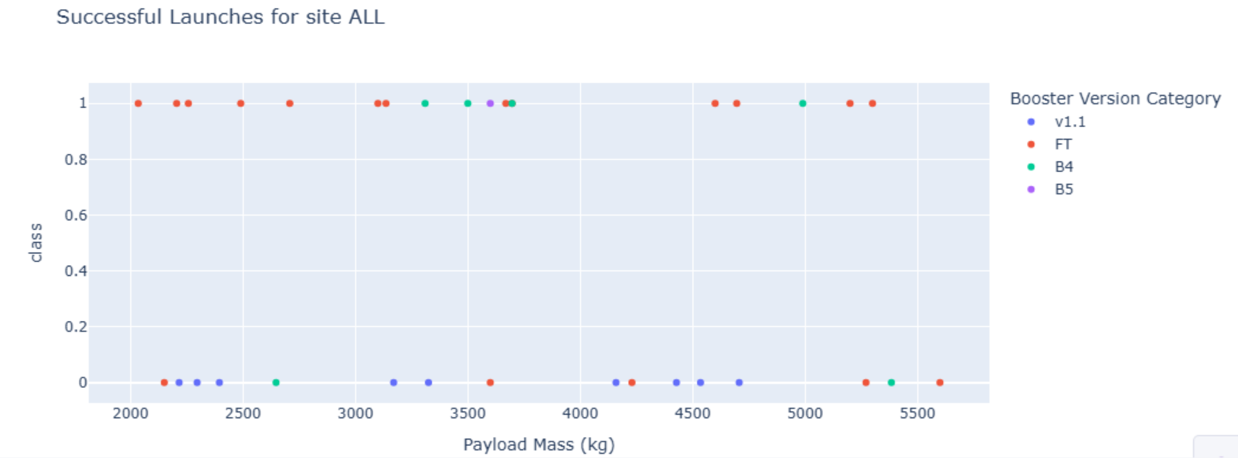


Payload Mass vs Launch Outcome

- 0-2500(kg) range with the most failed launches



- 2500-5000(kg) range with the most successful launches

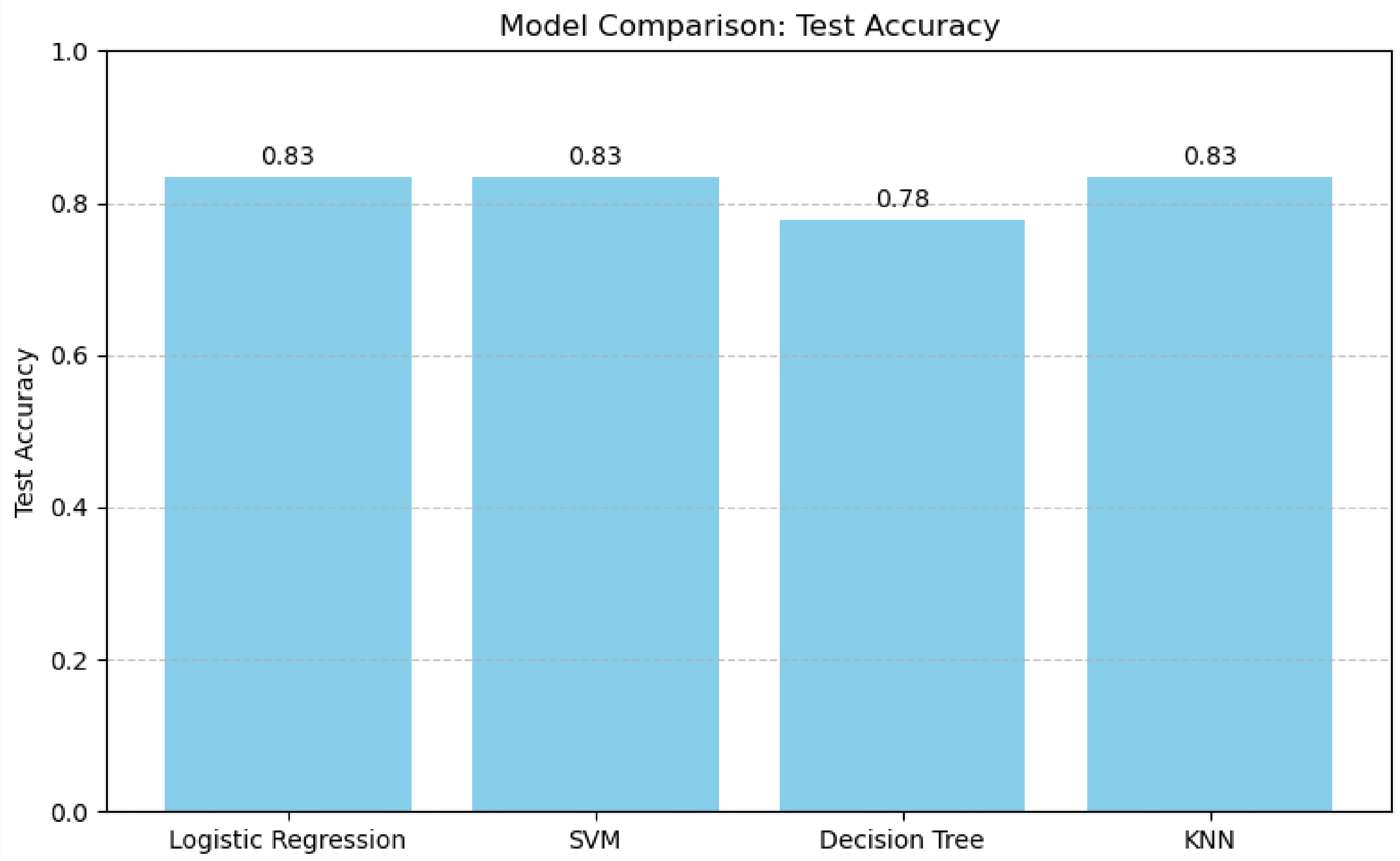


Section 5

Predictive Analysis (Classification)

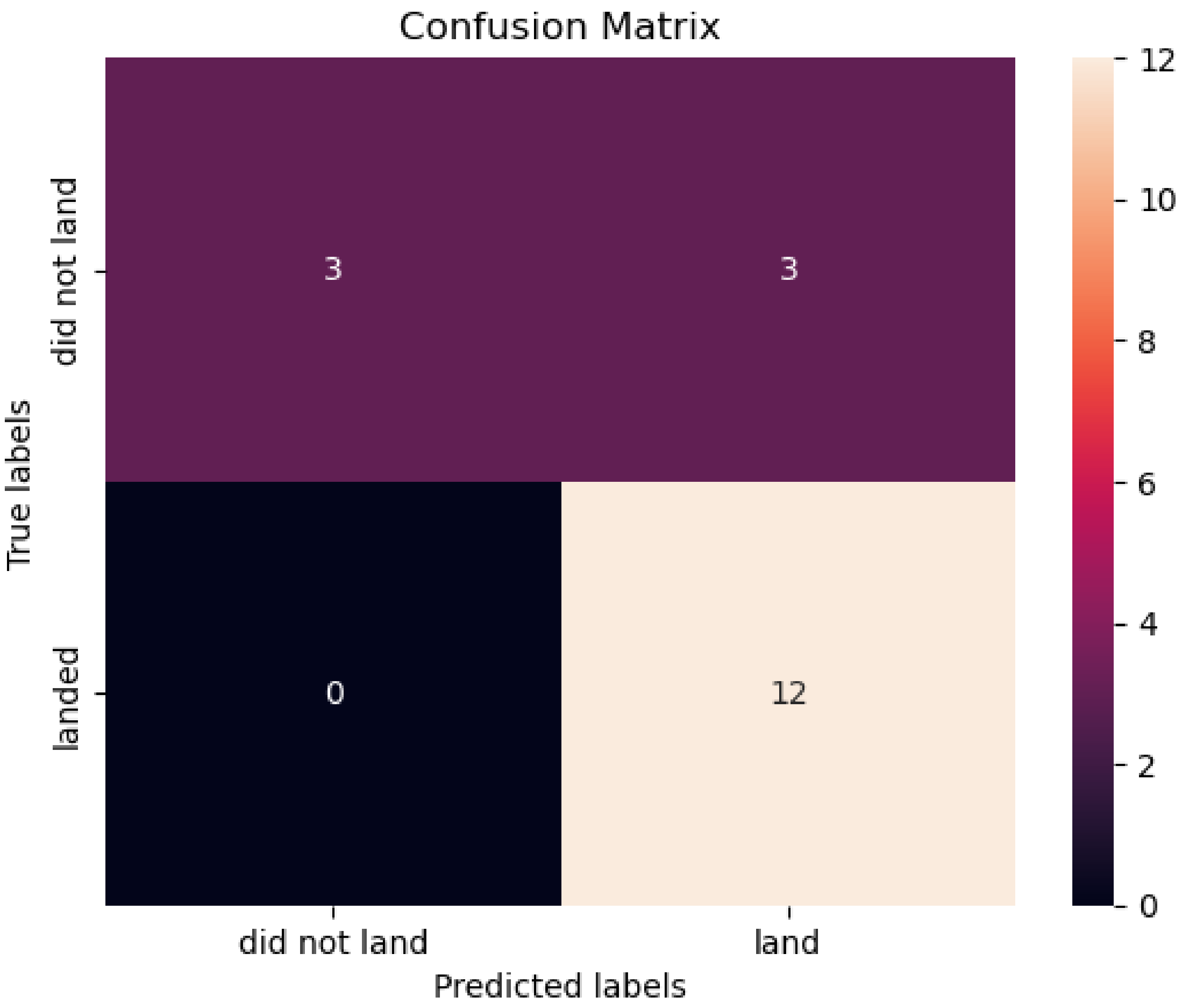
Classification Accuracy

- Accuracy of the models:



Confusion Matrix for Logistic, SVM and KNN:

- The model is very good at identifying when something landed (100% recall).
- It makes some mistakes when predicting things that did not land (3 false positives).
- Overall, the model is more cautious about missing actual landings than about incorrectly predicting landings.



Conclusions

- Logistic, SVM and KNN provided the same accuracy
- Using the Machine Learning we were able to predict if a launching will be successful and with it we were able to determine the launch cost.

Appendix

- Applied Data Science Capstone Github URL:
  - https://github.com/e-spec/Applied-Data-Science/tree/main

Thank you!