Winning Space Race with Data Science

SpaceX Falcon 9 Landing Analysis

Amir Hakim

18th November 2022

Outline

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

Executive Summary

Summary of methodologies

- Data collection
- Data wrangling
- EDA with data visualization
- EDA with SQL
- Building an interactive map with Folium
- Building a Dashboard with Plotly Dash
- Predictive analysis (Classification)

Summary of all results

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

Introduction

Background

SpaceX revolutionized the spaceflight industry by making affordable spacelight a reality. SpaceX advertises Falcon 9 rocket launches on its website, costing 62 million dollars; other providers cost upwards of 165 million dollars each. These cost savings are made possible primarily because the first stage of Falcon 9 can be reused.This project aims to analyze past SpaceX launches that utilized the Falcon 9 rocket to understand the different attributes of the launches and their relationship with the success of Falcon 9's first stage to land. This is done through exploratory data analysis (EDA) and predictive analysis to determine the outcome of the landing.

Successful Landing

Unsuccessful Landing

Problems

- How the interaction between the different features of the launch would impact the success rate of a landing.
- What are the best-operating conditions that SpaceX has to achieve to ensure a successful landing?

Section 1

Methodology

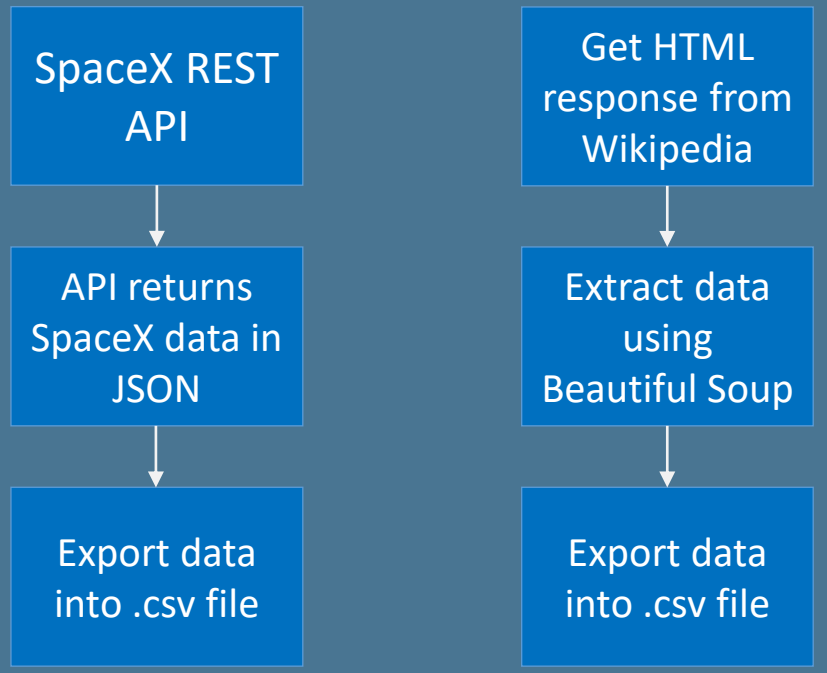
Overview

1. Collected the necessary data
    - SpaceX REST API
    - Web Scrapping
2. Performed data wrangling
    - One Hot Encoded the used dataset for Machine Learning training
    - Filter irrelevant columns and deal with null values
3. Perform exploratory data analysis (EDA) using visualization and SQL
4. Interactive maps with Folium for launch site analysis and an interactive dashboard with Plotly Dash
5. Did predictive analysis by building classification models to predict the outcome of the landing
    - Built, tuned, evaluated classification models

Data Collection Overview

The dataset was collected by:

1. SpaceX launch data is gathered by using SpaceX REST API.
   - This API gives us access to various data about the launches. To name a few, data about the rocket used, payload mass, and landing outcome.
   - This data is further used throughout other steps of this project and is also the test and training data on which the predictive models were trained and evaluated.
2. Another data source to obtain Falcon 9 launches by web scrapping on SpaceX launch Wikipedia.
   - This process was done using Beautiful Soup, a Python package for pulling data out of HTML files.
   - This dataset was not used in the other step as this web scrapping was done to refresh the understanding of web scrapping and Beautiful Soup.



Data Collection – SpaceX API

1. Getting response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)
```

2. Decoded the response in .json and was converted into a Pandas dataframe

```
data = pd.json_normalize(response.json())
```

3. Apply custom helper functions that will use the API to extract information using identification numbers in the launch data and append those data into a list

```
# Call getBoosterVersion
getBoosterVersion(data)

# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)
```

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}

# Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

4. Assign the extracted list to the specific keys of a dictionary and then the dictionary was converted to Pandas dataframe

```
data_falcon9 = df[df['BoosterVersion']!='Falcon 1']
```

5. Filtered out all rows of Falcon 1 data

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

data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

6. Replaced the payload mass missing values with mean values, exported dataframe into a .csv file

GitHub Notebook: Data Collection API

Data Collection – Web Scrapping

1. Getting response from HTML
2. Create BeautifulSoup object
3. All table elements were extracted within the Wikipedia page, and extract all columns element

```
response = requests.get(static_url).text
```

```
soup = BeautifulSoup(response, 'html.parser')
```

```
html_tables = soup.find_all("table")
column_names = []

# Apply find_all() function with `th` element on f
th_array = first_launch_table.find_all('th')
# Iterate each th element and apply the provided e
column name
for th_element in th_array:
    name = extract_column_from_header(th_element)
# Append the Non-empty column name (`if name is no
called column_names
    if (name != None and len(name)>0):
        column_names.append(name)
```

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with eac
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

4. Using column names as dictionary keys, custom functions were used to parse all the HTML columns and append the data into a list within a dictionary for their respective keys.

```
df=pd.DataFrame(launch_dict)
```

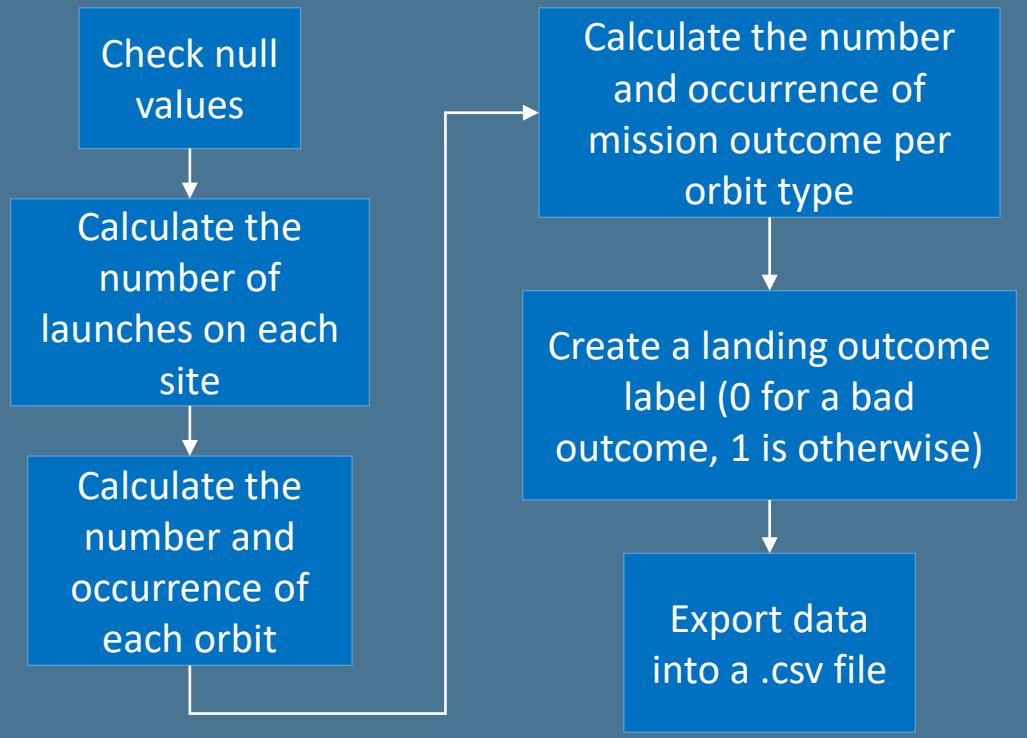
5. Dictionary was converted to dataframe

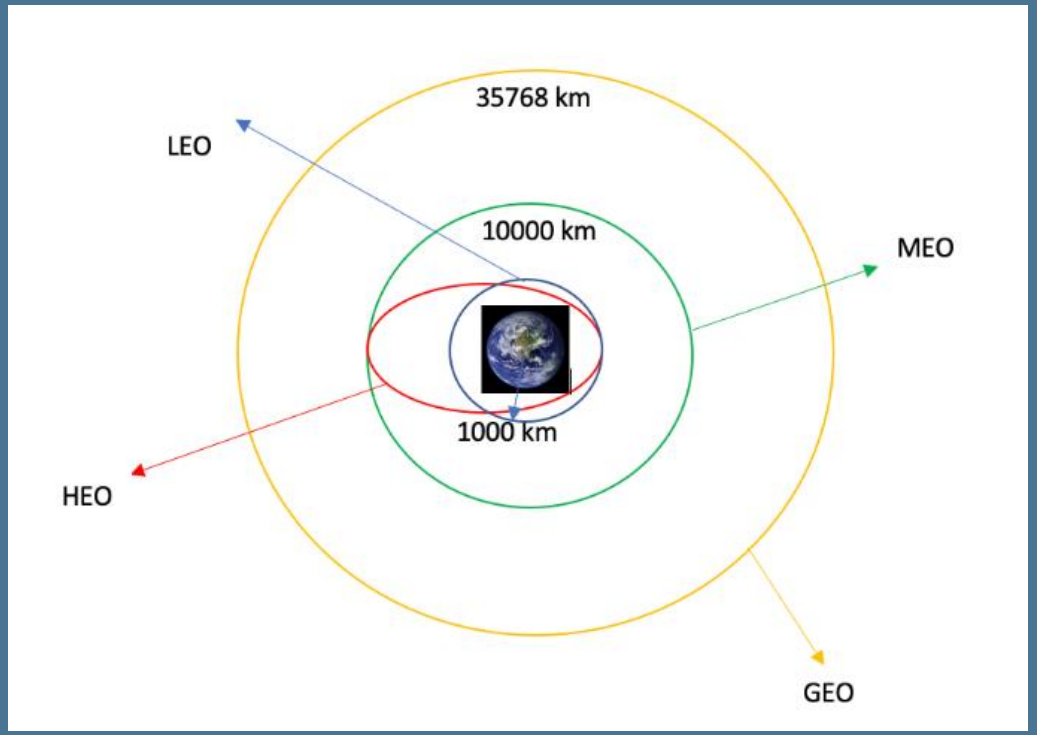
```
df.to_csv('spacex_web_scraped.csv', index=False)
```

6. Exported the dataframe into a .csv file

GitHub notebook: Data collection web scrapping Wikipedia page: List of Falcon 9 launches

Data Wrangling





Different types of orbit

GitHub notebook: Data Wrangling

EDA with Data Visualization

A few charts were plotted:

1. Scatter Plot: Show how much one variable affects the other
    - Flight number VS. Payload mass
    - Flight number VS. Launch site
    - Payload mass VS. Launch site
    - Flight number VS. Orbit
    - Payload mass VS. Orbit type
    - Orbit VS. Payload mass
2. Bar chart: A diagram in which the height of the rectangles represents the numerical values of different variables. Easy to compare sets of data between different groups.
    - Orbits VS. Success rate (mean)
3. Line chart: Line chart are useful to show the changes of data usually over time. This is useful to track changes over a short or long periods of time.
    - Years VS. Success rate (mean)

Preparing data for ML algorithms
(refer to cells under the Feature Engineering sections)

1. Choosing the relevant features for ML prediction
2. One hot encoded the dataset for categorical data
3. Cast all columns to be float64 data type
4. Export data as a .csv file

GitHub notebook: EDA with Data Visualization

EDA with SQL

SQL queries task:

1. Retrieved the names of the unique launch sites.
2. Retrieved 5 records where launch sites begin with the string 'CCA'.
3. Displayed the total payload mass carried by the boosters launched by the customer at NASA (CRS).
4. Displayed average payload mass carried by booster version F9 v1.1.
5. Founded the date when the first successful landing outcome on the ground pad was achieved.
6. List the names of the boosters which have successfully landed on a drone ship and have payload mass greater than 4000 and less than 6000.
7. Retrieved the total number of successful and failed mission outcomes.
8. Listed the names of the booster version which have carried the maximum payload mass.
9. Listed the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015.
10. Ranked the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the dates 2010-06-04 and 2017-03-20, in descending order.
11. Listed the total number of launches for each orbit type and whether it is a successful or failed mission.

GitHub notebook: EDA with SQL

Built Interactive Maps with Folium

- Locations of the four launch sites were marked on the map with object such as markers, circles, pop-up labels, lines, and clusters
- Clusters were used to show the number of launches that were successful or failed for each launch sites
  - Successful landing: Class 1: Green marker
  - Failed landing: Class 0: Red marker
- Calculated and draw a line for the distances between a launch site to its close proximities, the distance calculated:
  - Between the launch site and railways
  - Between the launch site and highways
  - Between the launch site and the coastline
  - Between the launch site and the closest city

GitHub notebook: Launch Sites Location Analysis

Interactive Dashboard with Plotly Dash

The interactive elements:

- Dropdown: choose data on any specific launch sites or all launch sites.
- Slider: to show the amount of success and failed landing for different payload mass ranges and booster versions.

The plots:

- Pie chart that shows the landing success in percentage for all the launch sites by default or if any specific launch site was chosen from the dropdown, the percentage between successful and failed landing for that specific launch site will be shown.
- Scatter plot that shows the successful and failed landing across different payload mass (the range can be set with the slider) for different booster versions for all or any specific launch sites (according to the dropdown)

Deploy

- As an extra step and to further enhance my knowledge of dashboarding, I have learned how to deploy the web-based dashboard by dash onto Heroku to host the dashboard for the viewing of others. However, the live website will only work **until 28th November 2022** because Heroku will no longer be a free platform.

GitHub link for source code: Interactive Dashboard URL for live website hosted on Heroku: Interactive Dashboard

Predictive Analysis (Classification)

Building the models

1. Used the one hot encoded dataset as the independent variable (the features used to predict landing class) and the column “Class” as the dependant variable (the label for the ML algorithm to predict) that categorized the landing outcome in binary as 1 for successful landing and 0 for failed landing
2. Loaded the independent variable (called X) into a Pandas dataframe and the “Class” label into a NumPy array (called Y)
3. Standardize and transform the independent data in X
4. Split both the variables X and Y into train and test datasets (X_train, X_test, Y_train, Y_test) and check the shape for each set
5. Used GridSeachCV to tune the parameters for all the tested ML algorithms (logistic regression, SVM, Decision tree, KNN)
6. Fit the datasets using the tuned parameter from GridSearchCV for each tested algorithm

Evaluating the model

1. The best training accuracy from GridSearchCV
2. Test accuracy using the test datasets
3. Confusion matrix
4. Comparing the test accuracy and confusion matrix for all the tested ML algorithms

GitHub notebook: SpaceX Machine Learning Prediction

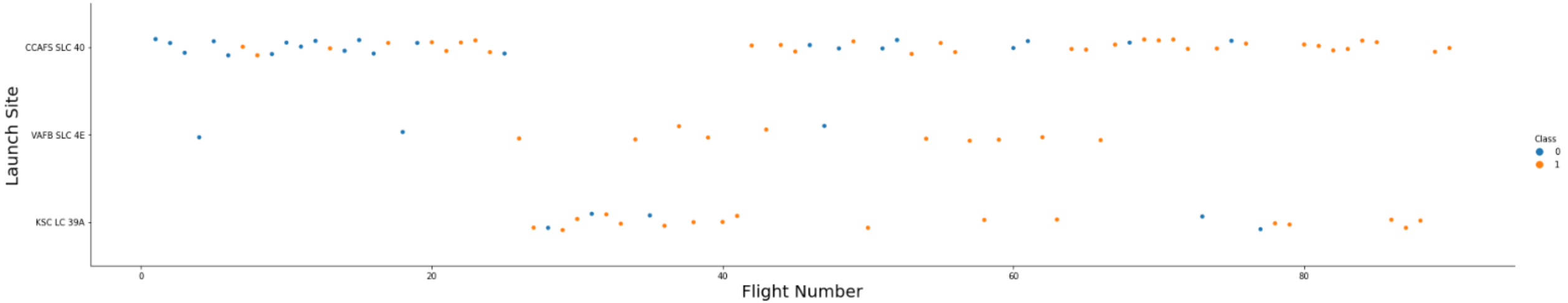
Results

1. Exploratory data analysis results
2. Interactive analytics demo in screenshots
3. Predictive analysis results

Section 2

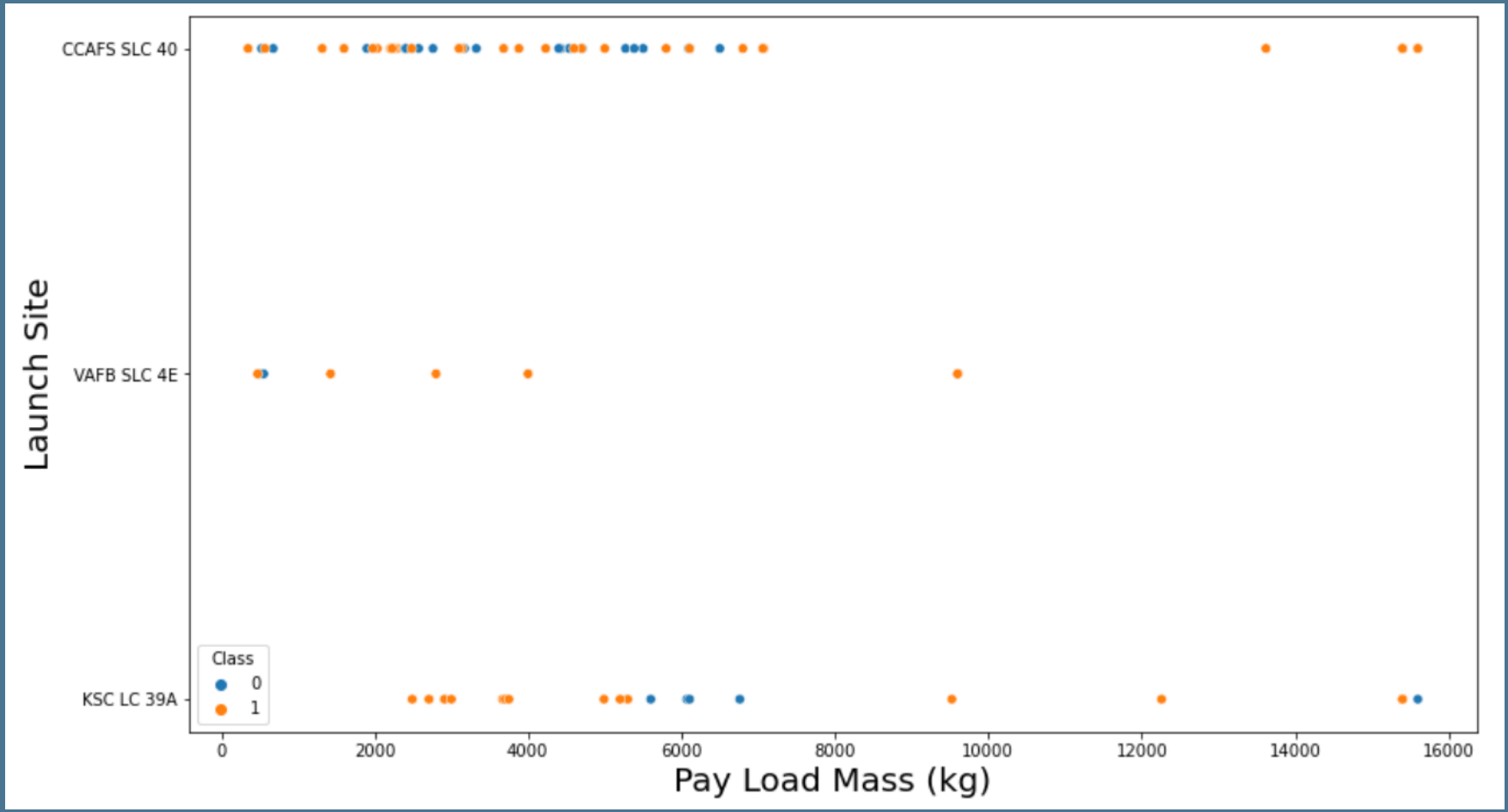
Insights drawn from EDA

Flight Number vs. Launch Site



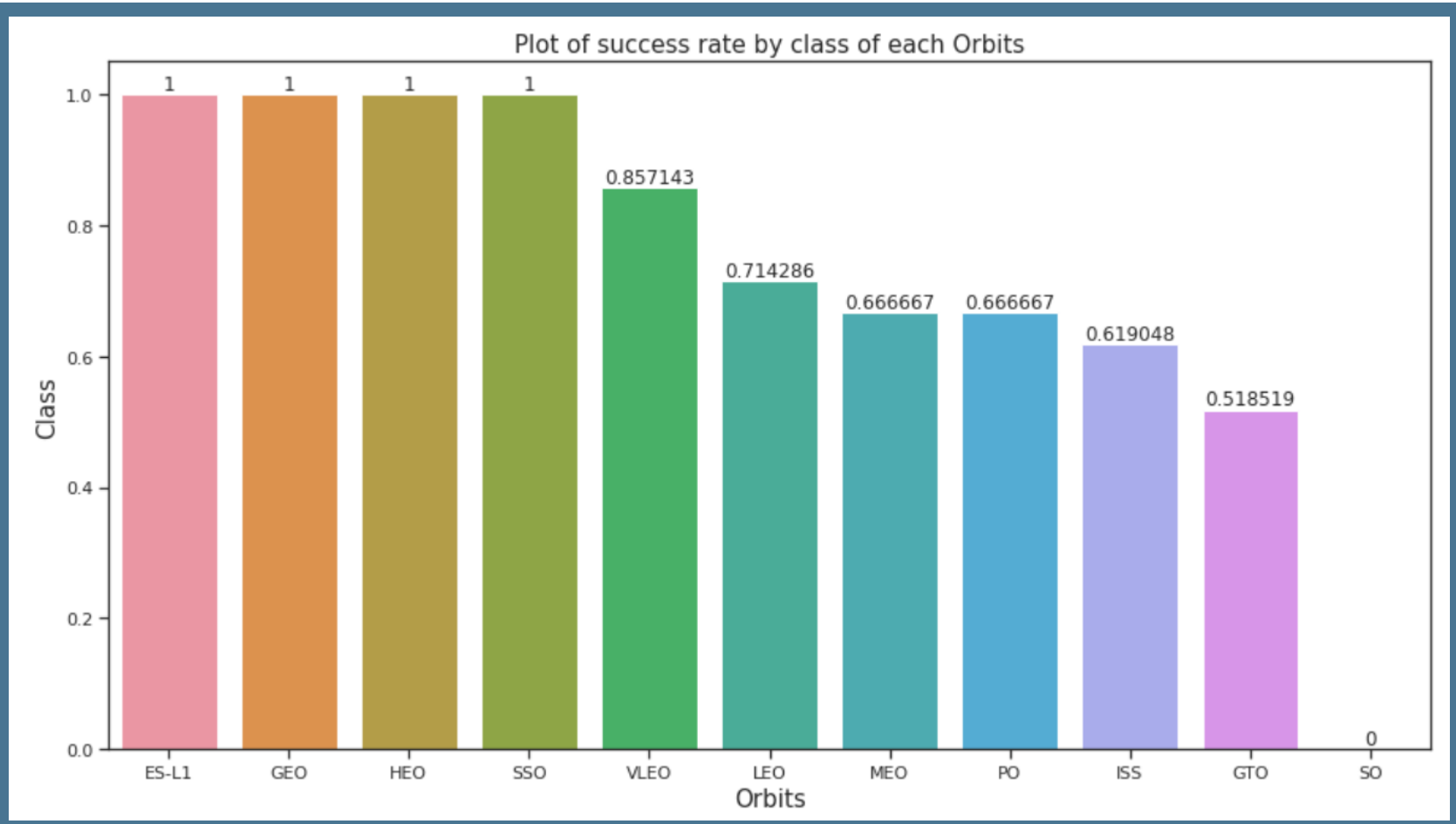
- From the plot, it shows that the higher the number of flights for a launch site, the greater the success rate for a landing

Payload Mass vs. Launch Site



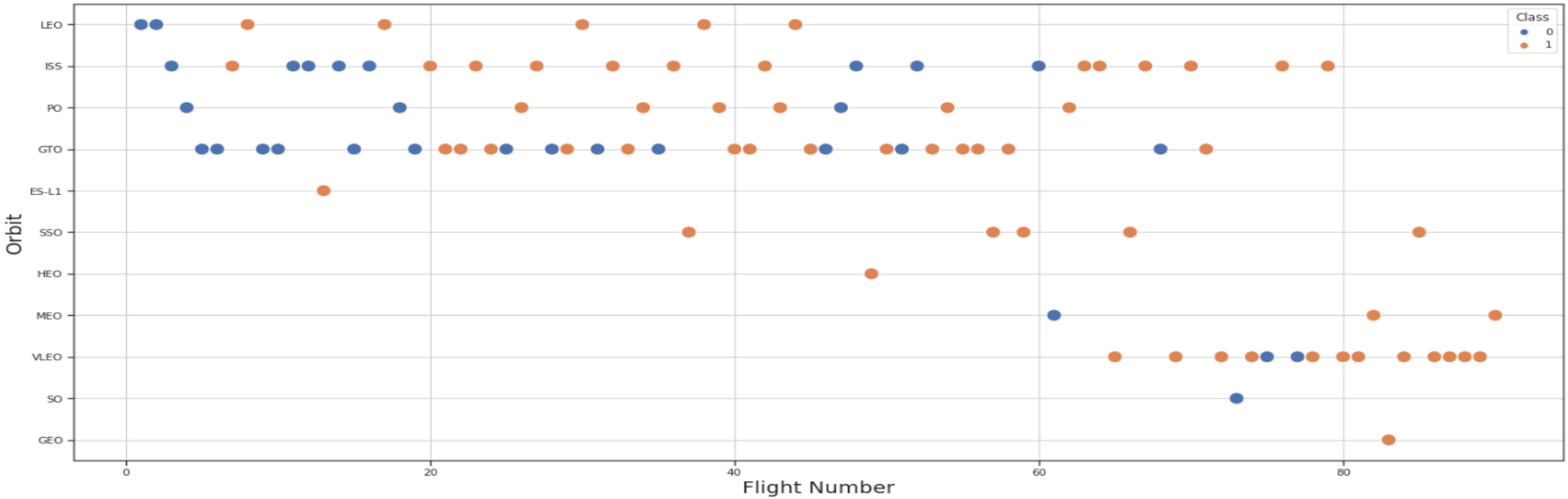
- For the CCAFS SLC 40, for payload mass above 7000kg, it tends to have a higher success rate
- For KSC LC 39A, for payload below around 5500kg, it also shows a higher success rate.
- For VAFB SLC 4E, there is no payload above 10000kg launched. Hence, finding any apparent trend for this launch site is hard.

Orbit vs. Success Rate



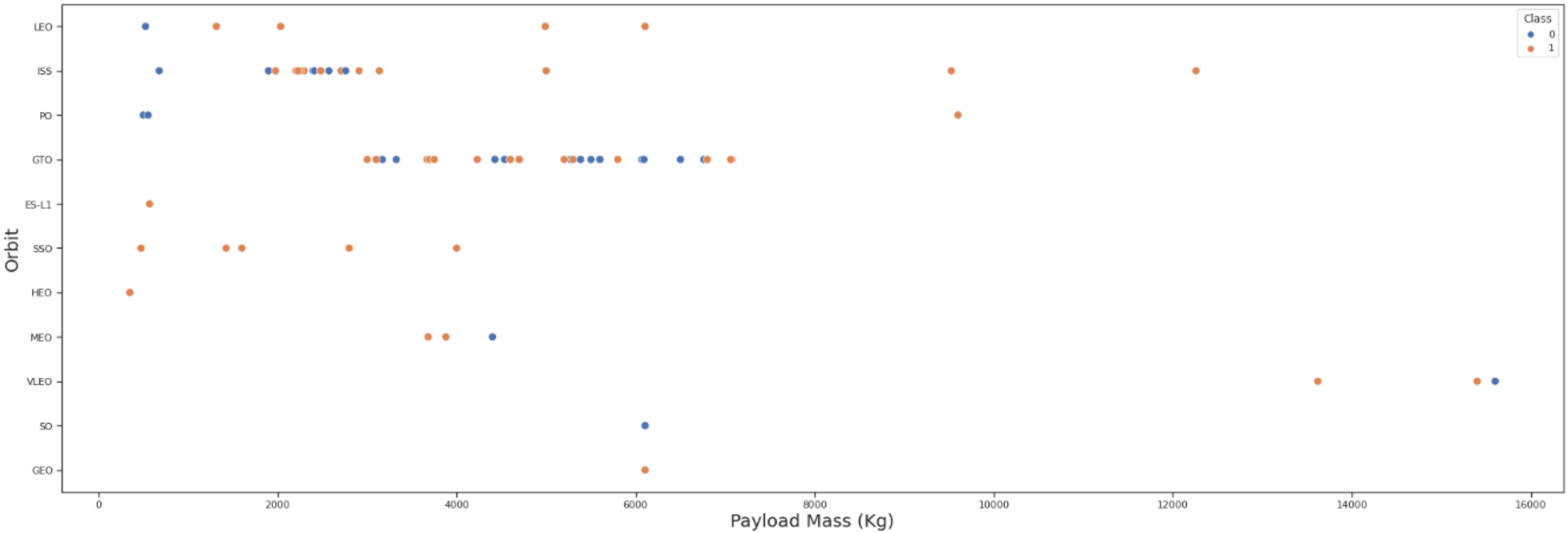
- For the orbit destination, ES-L1, GEO, HEO, and SSO have the best success rate.
- However, for ES-L1, HEO, and GEO, if we were to observe the next slide, there would be only one launch for each orbit destination.

Flight Number vs. Orbit



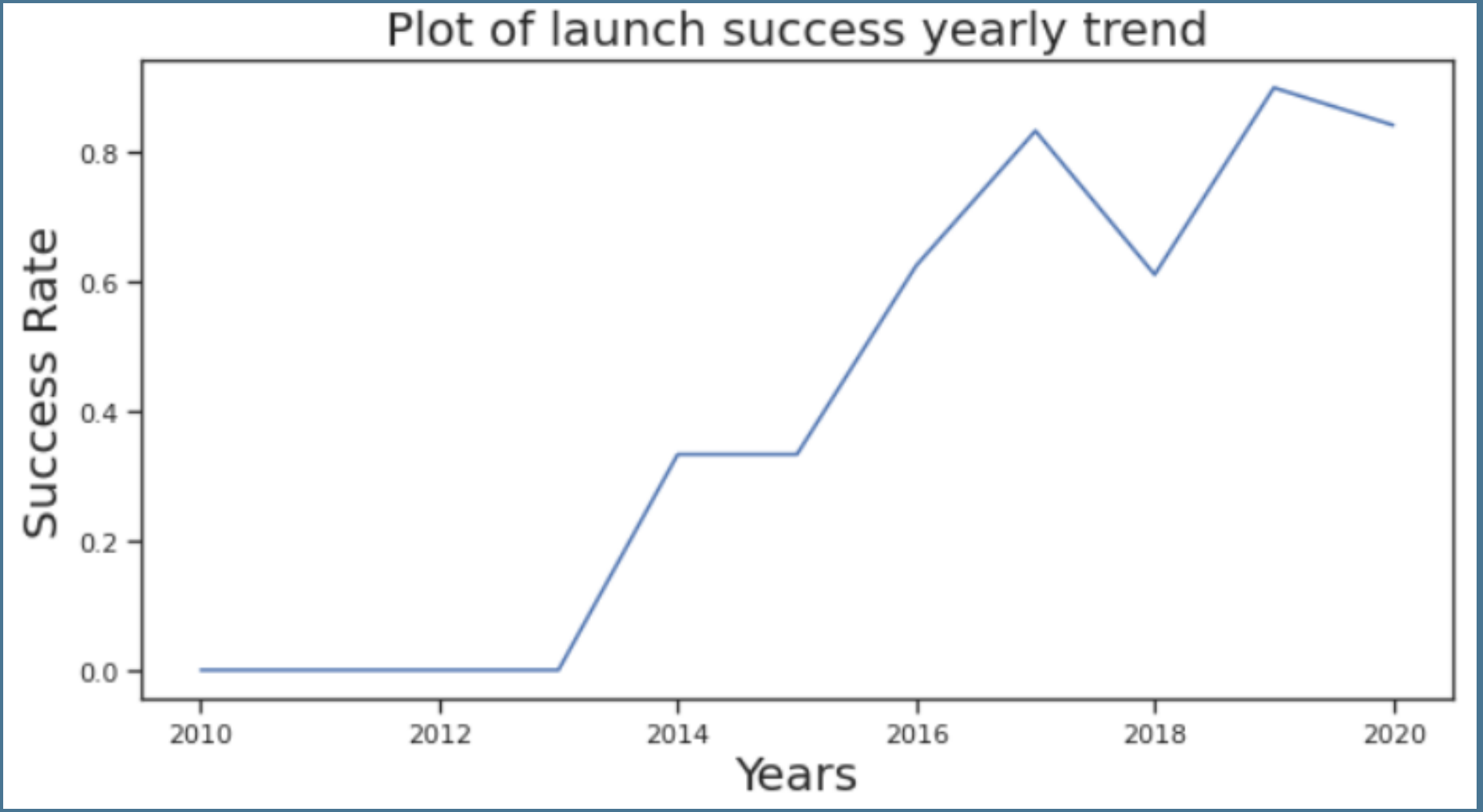
- Based on the previous slide, ES-L1, GEO, HEO, and SSO have the best success rate for the orbit destination.
- However, for ES-L1, HEO, and GEO, there is only one launch for each orbit destination. Thus, no clear relationship can be said about these orbits. The same can be said for GTO orbit, too, due to a series of failures and successes across all flight numbers.
- The success of LEO orbit does appear to be related to the number of flights due to having a higher success on the later number of a flights.

Payload Mass vs. Orbit



- As payload mass increase, the orbit PO, LEO, and ISS has an increased success rate.
- For GTO, we cannot distinguish this well as both positive landing rate, and negative landing(unsuccessful mission) are both present.

Launch Success Yearly Trend



- The success rate started to have a sharp increase in the year 2013 and kept increasing until 2020

All Launch Site Names

```
%sql SELECT DISTINCT launch_site FROM SPACEXTBL;
```

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

- The DISTINCT statement used in this query ensures that the query only returns unique launch site names from the launch_site column.

Display 5 Records where Launch Sites Begin with the String 'CCA'

```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

- This query used the LIKE operator to find the top 5 records for the launch site name witha string that started with ‘CCA’.

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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 Carried by Boosters Launched by NASA (CRS)

```
%sql SELECT SUM(payload_mass__kg_) AS "TOTAL PAYLOAD MASS BY NASA(CRS)" FROM SPACEXTBL WHERE customer = 'NASA (CRS)';
```

TOTAL PAYLOAD MASS BY NASA(CRS)
45596

- This query used the SUM function to total the amount of payload mass from the column payload_mass_kg_ launched by the customer ‘NASA (CRS)’.

Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(payload_mass__kg_) AS "average payload mass carried by booster version F9 v1.1"  FROM SPACEXTBL WHERE booster_version like 'F9 v1.1';
```

average payload mass carried by booster version F9 v1.1
2928

- This query used the AVG function to retrieve the average payload mass from column payload_mass_kg_ that is carried by the booster ‘F9 v1.1’.

First Successful Ground Landing Date

For this task two queries were tested that resulted in the same output

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad" FROM SPACEXTBL WHERE "Landing _Outcome" = 'Success (ground pad)';
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

- The first query used the MIN function on the launch date to retrieve the first date of which the condition of the “Landing_Outcome” is ‘success (ground pad)’.

```
%%sql
SELECT DATE AS "First Succesful Landing Outcome in Ground Pad"
FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Success (ground pad)' ORDER BY DATE ASC LIMIT 1;
```

- The second query only select the launch date where the “Landing_Outcome” condition is ‘success (ground pad)’ and only retrieved the first record by using ORDER BY DATE in ascending (to make sure the first date is the first record) and limit it to display 1 record only.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql
SELECT BOOSTER_VERSION FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

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

- This query was used to retrieve the booster version that successfully landed on a drone ship and also carried a payload mass between 4000kg and 6000kg.

Total Number of Successful and Failure Mission Outcomes

```
%%sql
SELECT mission_outcome, COUNT(*) AS "NUMBER OF LAUNCH" FROM SPACEXTBL GROUP BY mission_outcome;
```

```
 * ibm_db_sa://zzp37610:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.a
Done.
```

mission_outcome	NUMBER OF LAUNCH
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- This query used the COUNT function to count the number of launches for each mission outcome. We can see that there is one success and unclear payload status.

```
%sql SELECT COUNT(mission_outcome) AS "Total Mission" FROM SPACEXTBL;
```

```
 * ibm_db_sa://zzp37610:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2i
Done.
```

Total Mission
101

- This query used the COUNT function to count the total success and failure mission_outcome together.

```
%%sql
SELECT SUM(case when mission_outcome LIKE '%Success%' then 1 else 0 end) AS "Successful Mission",sum(case when mission_outcome LIKE '%Failure%' then 1
```

```
 * ibm_db_sa://zzp37610:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/BLUDB
Done.
```

Successful Mission	Failure Mission
100	1

- This last query is used to sum up, the amount of success and failure missions separately by using the LIKE operator on the string in mission_outcome column that consists of ‘success’ or ‘failure’.

Boosters Carried Maximum Payload

```
%%sql
SELECT DISTINCT booster_version, payload_mass__kg_ FROM SPACEXTBL
WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL);
```

```
 * ibm_db_sa://zzp37610:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l0
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

- This query was used to retrieve distinct booster versions and payload mass with the condition that the booster carried the maximum payload mass. This query also used sub-query as its condition to find the maximum payload mass.

Failed Landing Outcomes on Drone Ship for the Year 2015

```
%sql SELECT DATE, "Landing _Outcome",BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE DATE LIKE '2015-%' AND "Landing _Outcome" LIKE 'Failure (drone ship)';
```

```
 * ibm_db_sa://zzp37610:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/BLUDB
Done.
```

]:

DATE	Landing _Outcome	booster_version	launch_site
2015-01-10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- This query retrieved the date, landing outcome, booster version, and launch site for launch in the year 2015 and has an outcome of ‘Failure (drone ship)’

Landing Outcomes between the Date 2010-06-04 and 2017-03-20, in Descending Order

```
%%sql
SELECT "Landing _Outcome", COUNT("Landing _Outcome") AS "TOTAL LAUNCH" FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing _Outcome"
ORDER BY COUNT("Landing _Outcome") DESC
```

Landing _Outcome	TOTAL LAUNCH
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

- This query was used to retrieve the number of launches that resulted in each landing outcome between 2010-06-04 and 2017-03-20. The GROUP BY statement is to group the count of each landing_outcome, and then ORDER BY was used to rank the landing_outcome columns based on the number of each total launch in descending order.

Total Number of Launch for each Orbit Type and their Outcome

```
%%sql
SELECT orbit AS "Orbit", count(*) AS "Total Launch", SUM(case when mission_outcome LIKE '%Success%' then 1 else 0 end) AS "Successful Mission",
sum(case when mission_outcome LIKE '%Failure%' then 1 else 0 end) AS "Failure Mission" FROM SPACEXTBL
GROUP BY orbit
ORDER BY SUM(case when mission_outcome LIKE '%Success%' then 1 else 0 end) DESC ;
```

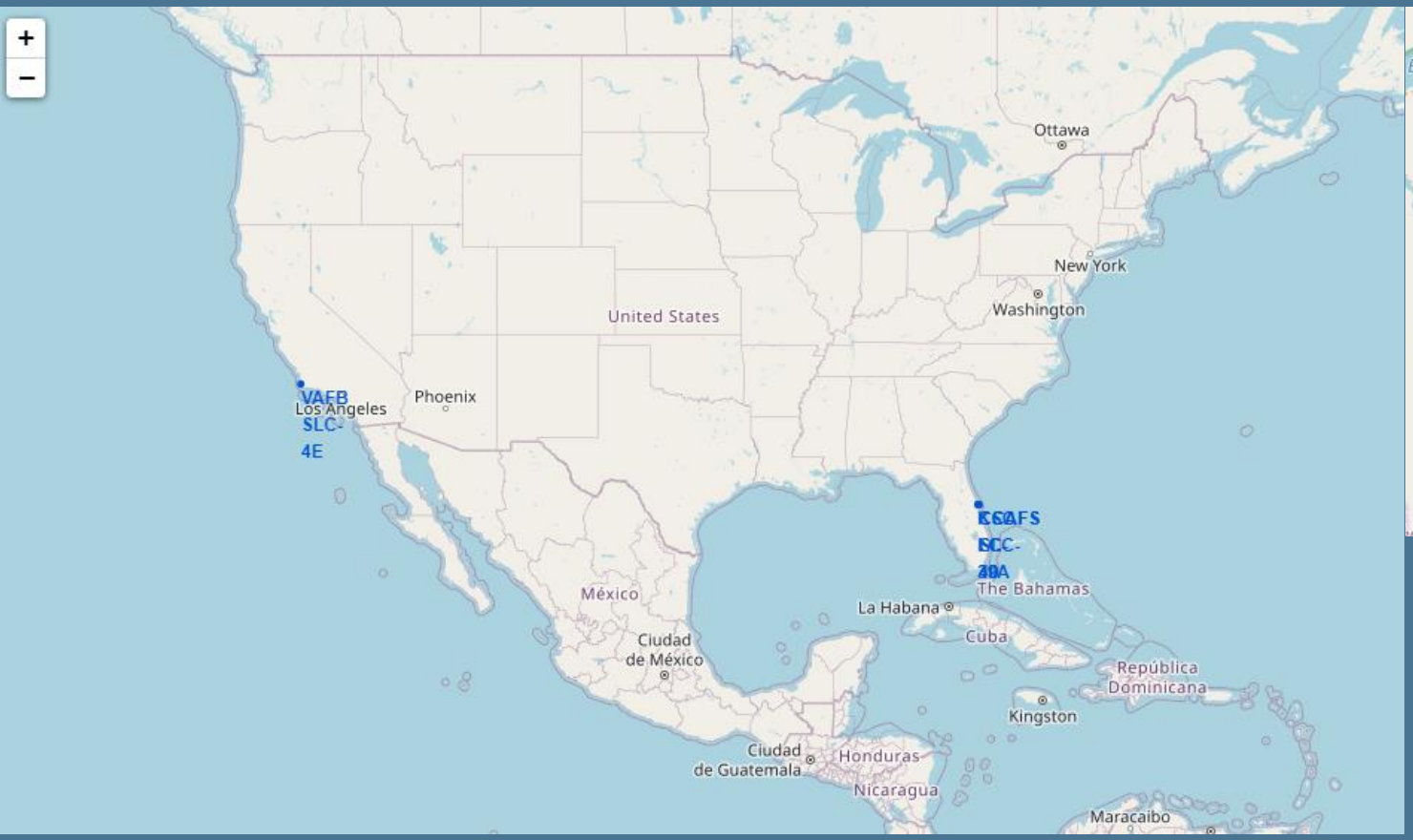
Orbit	Total Launch	Successful Mission	Failure Mission
GTO	30	30	0
LEO	25	25	0
LEO (ISS)	26	25	1
Polar LEO	8	8	0
SSO	6	6	0
MEO	3	3	0
HEO	2	2	0
Sub-orbital	1	1	0

- This query retrieved the total number of launches, successes, and failed mission for each orbit destination and then ordered it by descending order of success mission.

Section 3

Launch Sites Proximities Analysis

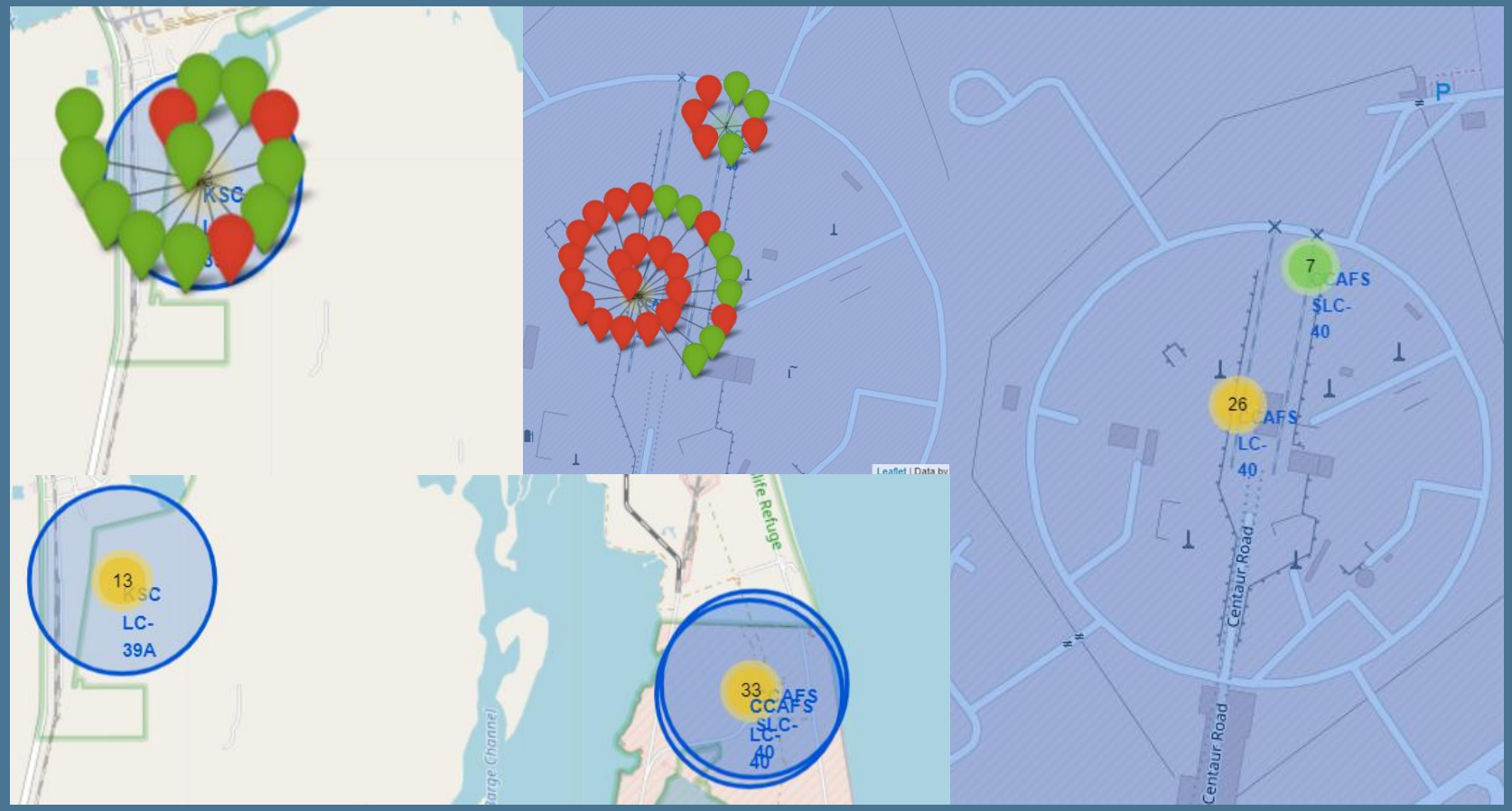
Marked Launch Sites



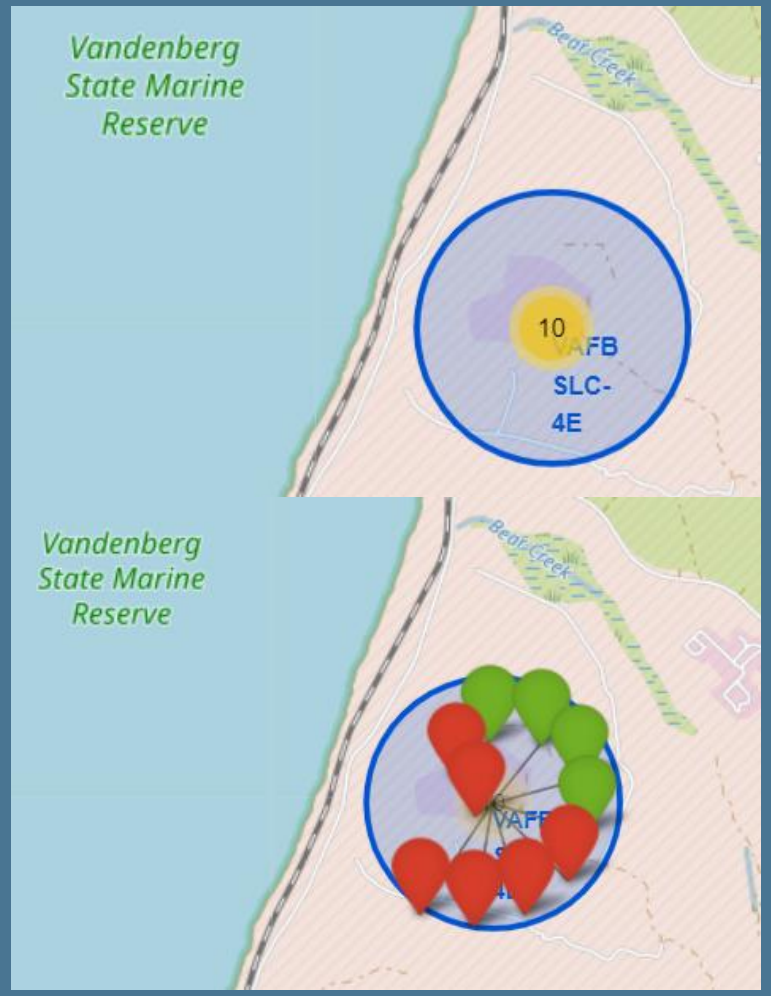
- All four launch sites were marked with a circle object and labeled for the names of the launch sites.
- All launch sites are located on the United States coasts of Florida and California.

Clusters for Success and Failed Landings

Florida launch sites



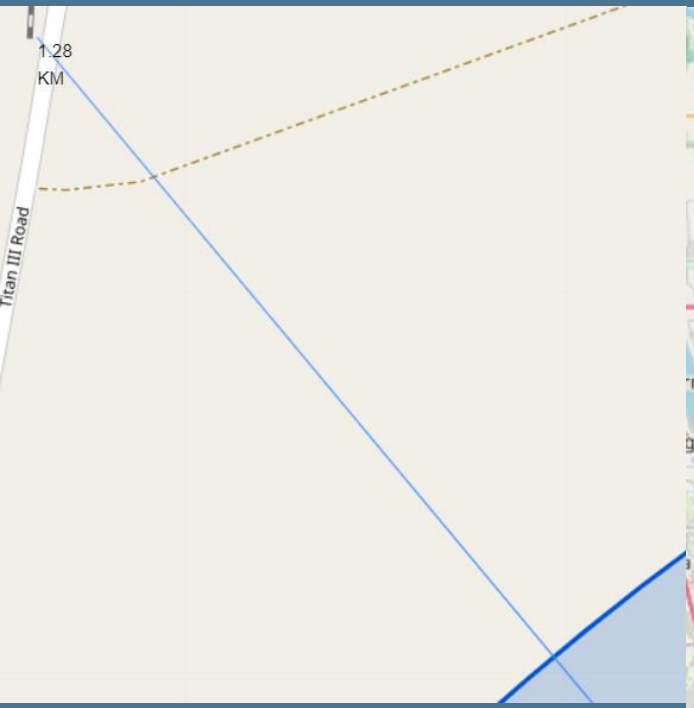
LA launch site



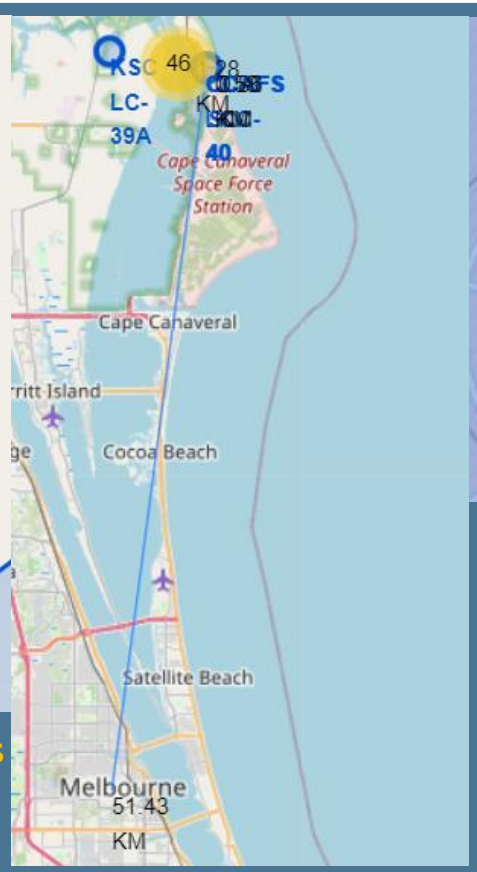
Clusters were used to show the number of launches for each site and the number of launch that were successful or failed for each launch site

- Successful landing: Class 1: Green marker
- Failed landing: Class 0: Red marker

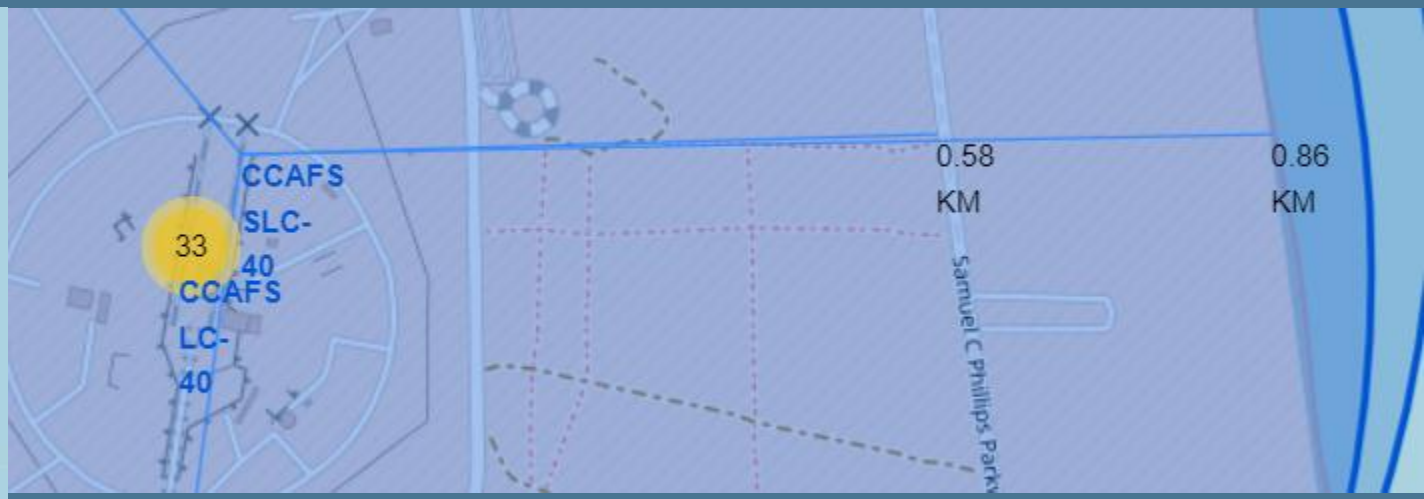
Proximities Analysis



Launch sites to the nearest highways



Launch sites to the nearest city



Launch sites to the nearest railroad and coastline

```
distance_highway = 0.5834695366934144  km
distance_railroad = 1.2845344718142522  km
distance_city = 51.43416999517233  km
```

```
print(distance_coastline,' km')
```

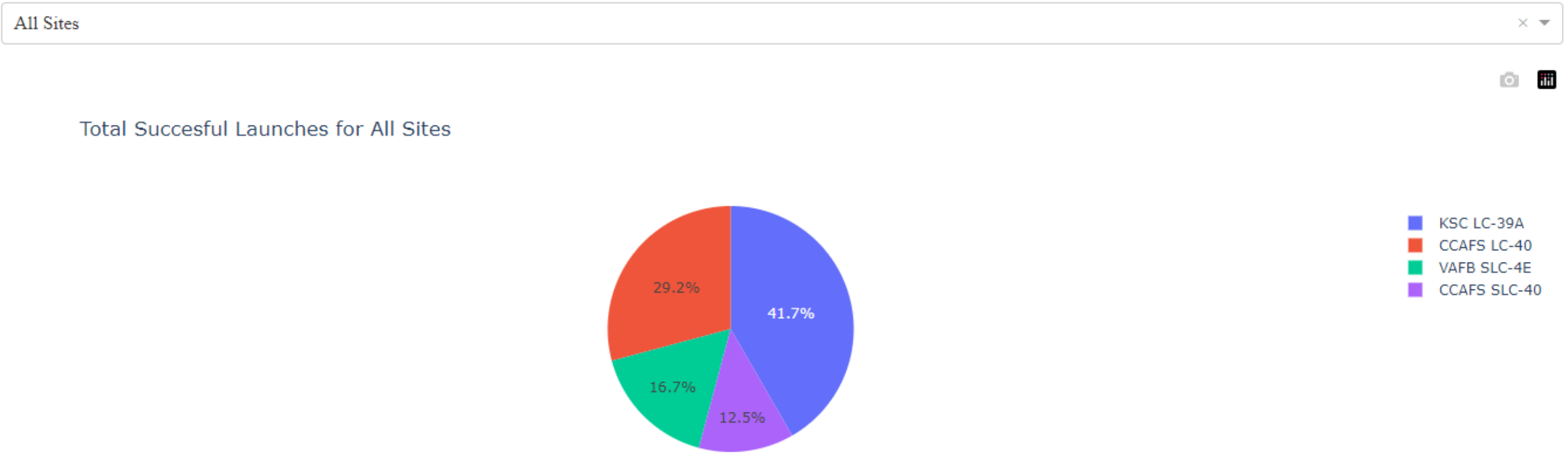
```
0.8627671182499878  km
```

- Lines were added between the the launch site and proximities in question
- Are launch sites in close proximity to railways? YES
- Are launch sites in close proximity to highways? YES
- Are launch sites in close proximity to coastline? YES
- Do launch sites keep certain distance away from cities? NO

Section 4

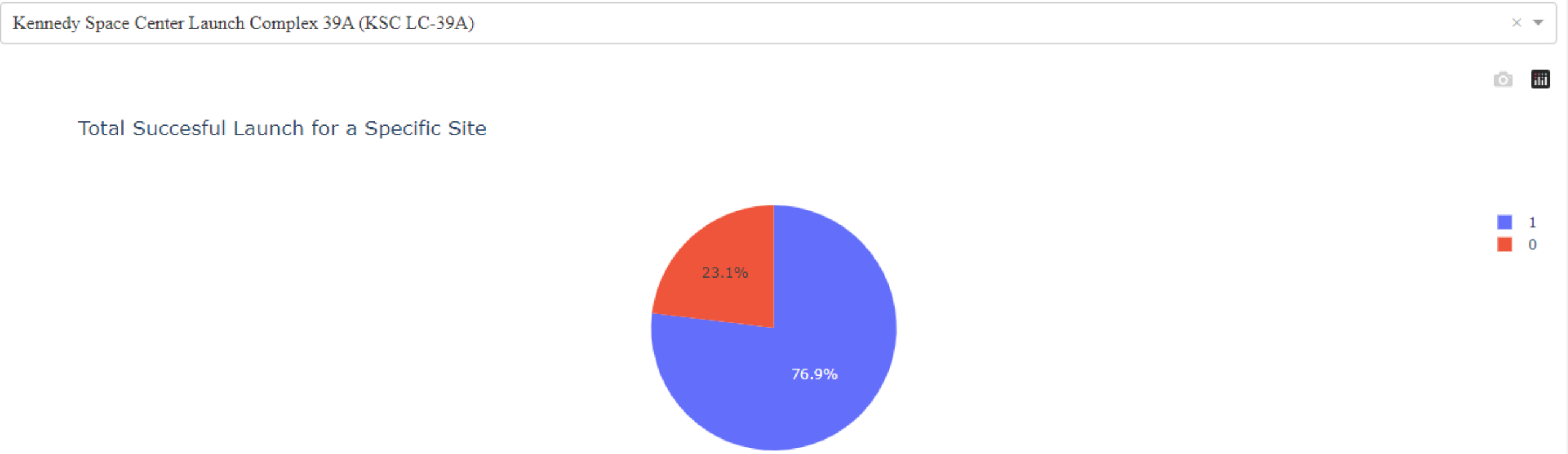
Dashboard with Plotly Dash

Success Percentages for all Launch Sites

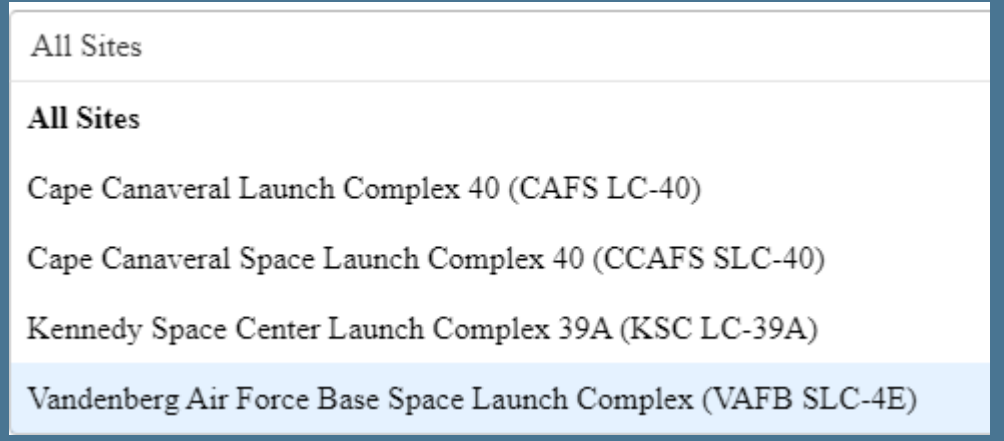


- From this pie chart, we can see that Kennedy Space Center Launch Complex 39A (KSC LC-39A) had the highest share of a successful landing
- Currently, the data for all launch sites are shown together, as no specific launch sites were chosen from the dropdown menu.

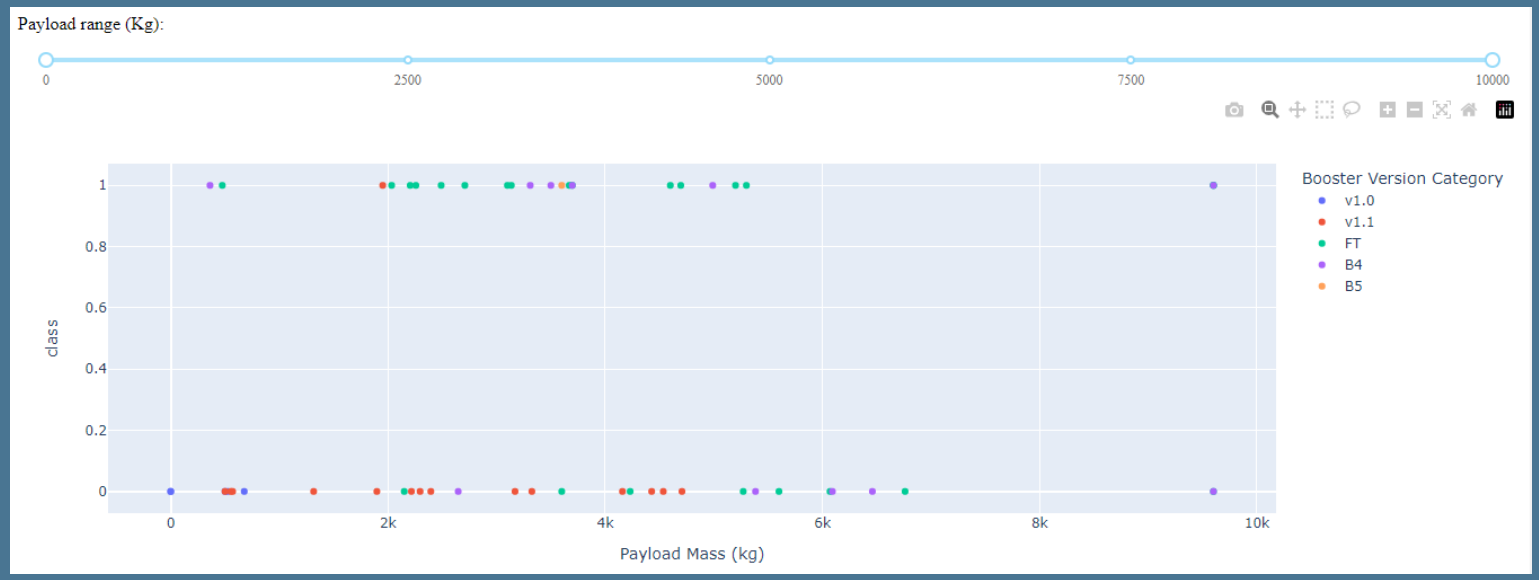
Percentages for Launch Site with Most Success Ratio



- From this pie chart, Kennedy Space Center Launch Complex 39A (KSC LC-39A) managed to achieve almost 77% of success (class 1) and 23% of landing (class 0) for Falcon 9 first-stage landing
- User can select data for all launch sites using the dropdown menu above the piechart.



Payload VS. Class (outcome)



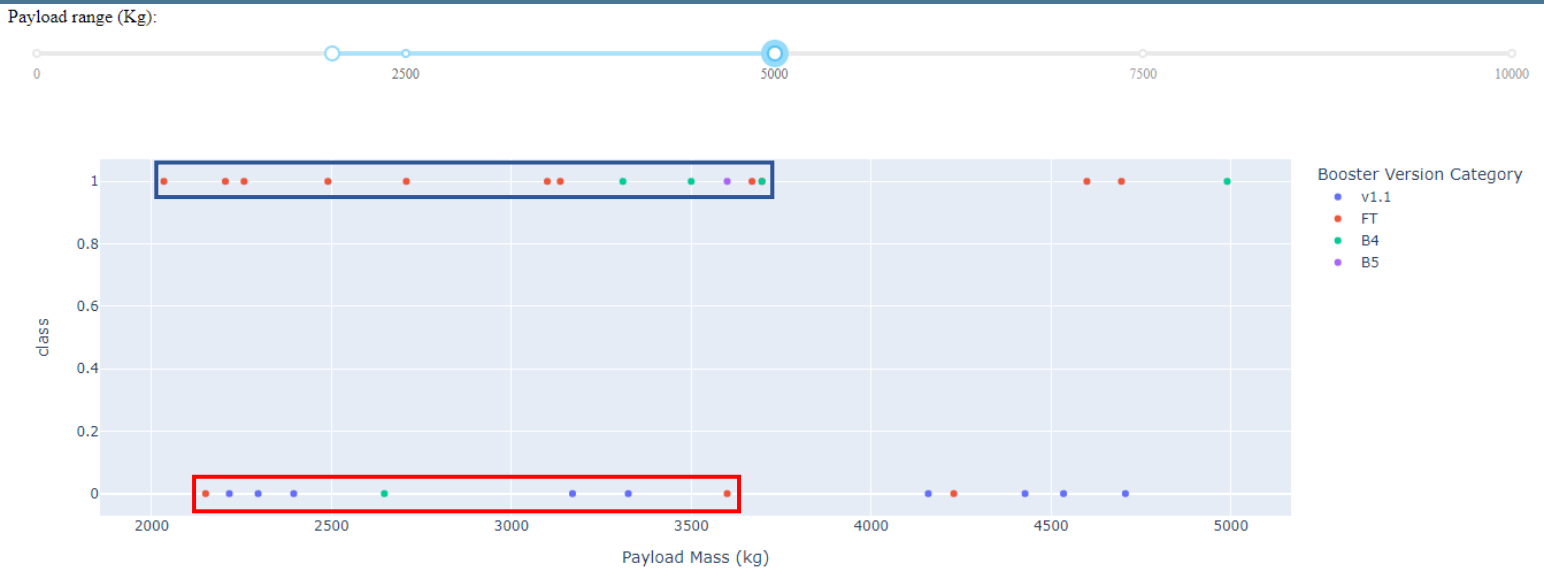
- Scatterplot of payload vs. class for all launch sites.
- From this particular plot, we could see the individual success and failed landing alongside what booster versions were used for all launch sites
- For this plot, another element exist to adjust the payload range using the slider from 0-10,000 kg.

- If a specific launch site were chosen from the dropdown, this will show the data points for only that launch site.
- From the plot on the bottom left, we could see that for launch site KSC LC-39A has 10 successes and 3 failed landings, and all 3 failed landings were using the FT booster version.

Adjusting the Payload Range Slider for Payload VS. Class Plot

1. Data for all launch sites across all payload range

2. Data for all launch sites across all payload range of 2k-5k Kg

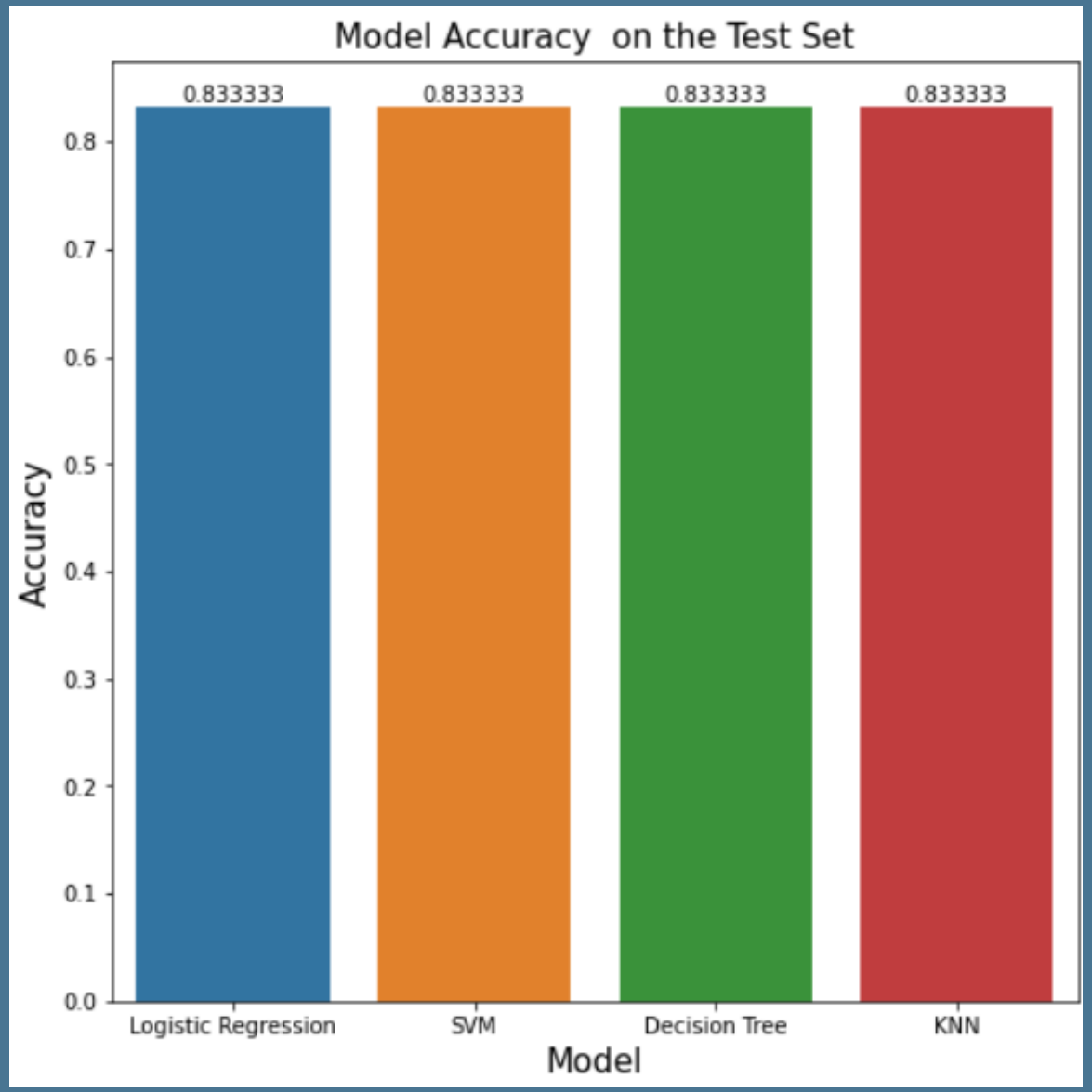


- From the first plot, we could see that most of the successes are between 2k-4k Kg of payload mass as that range is the densest with success datapoints.
- To have a clearer view, the range slider was adjusted to show the range between the 2k-5k Kg, as shown on the second plot.
- From the second plot, we can count 12 successes and 8 failures between 2k-5k Kg payload range.
- Out of 12 successes, 8 were using the FT booster version, and out of 8 failed landings, 5 launches were using v1.1 as their booster version.
- This shows that for payload range between 2k-4k Kg, the FT version has resulted in the best chances for a successful landing

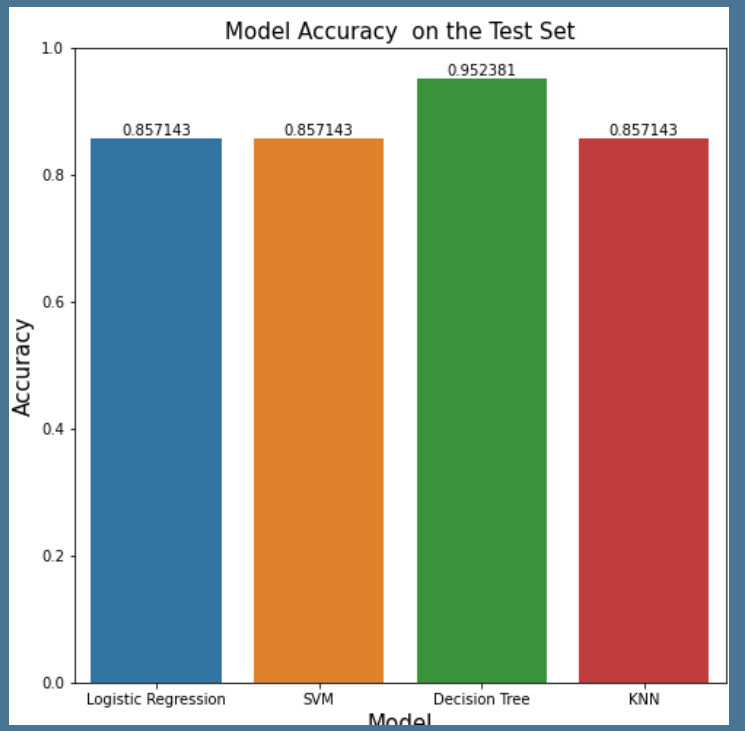
Section 5

Predictive Analysis (Classification)

Classification Accuracy

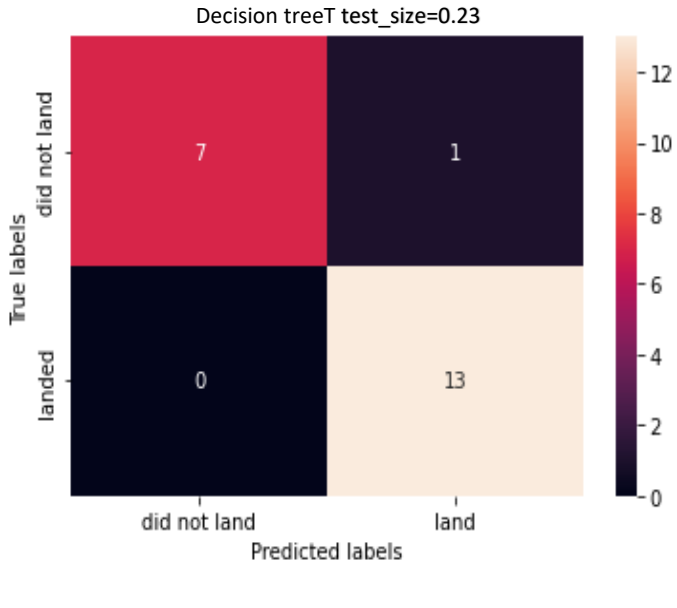


- All tested algorithms resulted in the same accuracy on the test set.
- Results for all methods are the same because of the small size of the dataset and even smaller test set. The size of the test set is 20% of the entire dataset. With this result, it is hard to find the best-performing model.
- Further testing was done by trial and error on the test set size. Changing the test set size to ~23% increased test samples from 18 (for 20%) to 21, and the accuracy for the decision tree algorithm_increased from 0.833 to 0.952 (14% increase).

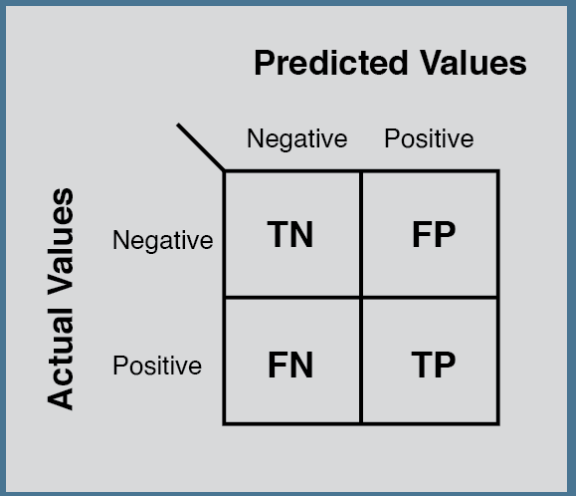


GitHub notebook: Decision Tree (best performance)

Confusion Matrix



- Based on the previous slides, with the default test set size value given for this project which is 20% of the dataset, all algorithms resulted in the same test accuracy and confusion matrix output. Hence, deciding the best performing model for this project is challenging.
- Further testing through trial and error on the test set size managed to increase the accuracy score on the test dataset and improve the confusion matrix output.
- Based on the first confusion matrix (test size 0.2), we could see that the model predicted correctly for all 12 rockets that landed. However, out of 6 rockets that did not land, 3 were predicted to land.
- Based on the confusion matrix, we could see the improvement when increasing the test set size to 23% in the sense that there all 12 rockets that landed were still predicted to land, and only 1 rocket that did not land was predicted to land compared to 3



Conclusion

- Under the default test_size value given for this project, all tested algorithms performed similarly for both test accuracy score and confusion matrix output.
- When using test_size 23%, the decision tree algorithm got the highest accuracy and increased its performance by 14% compared to the model using the default test size value.
- FT boosters are the best booster version for payload between 2k-4k Kg
- Kennedy Space Center Launch Complex 39A (KSC LC-39A) had the highest share of successful landing
- For the orbit destination, ES-L1, GEO, HEO, and SSO have the best success rate.

Thank You