IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vinicius Branco Scortegagna
April 24, 2022

Outline

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

Executive Summary

- In this project, a predictor model for the success of a rocket's first stage landing was produced based on the records provided by SpaceX.
- This was done using machine learning classification algorithms on a dataset that was obtained by cleaning the Falcon 9 launches data collected through SpaceX API and web scraping.
- Valuable insights were obtained in the process of exploratory data analysis, which was done using SQL queries and Python visualization techniques with graphs and maps, as well as an interactive dashboard.
- These insights include information about the best payload mass range, best booster version, best orbit and best launch site. Finally, an evaluation of the predictions made by the trained classification models was done, showing an accuracy of 83% in 3 out of 4 models.

Introduction

- Rocket launching is usually very expensive. However, SpaceX was able to minimize costs by reusing the first stage of their rockets. This can be done provided that the first stage is able to perform a successful landing.
- SpaceX showed, for example, that the total cost may be reduced from 165 million to 62 million dollars if they manage to make the first stage land safely.
- It is useful, therefore, to use machine learning techniques in order to predict if the first stage of a launch is more likely to land or not based on the records of past launches. Also, information retrieved by analysis of the data recorded by SpaceX may help increase its landing success rate and thus further improve money saving by learning from past experience.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
- Data Wrangling
- Exploratory data analysis (EDA) using visualization and SQL
- Interactive visual analytics using Folium and Plotly Dash
- Predictive analysis using classification models

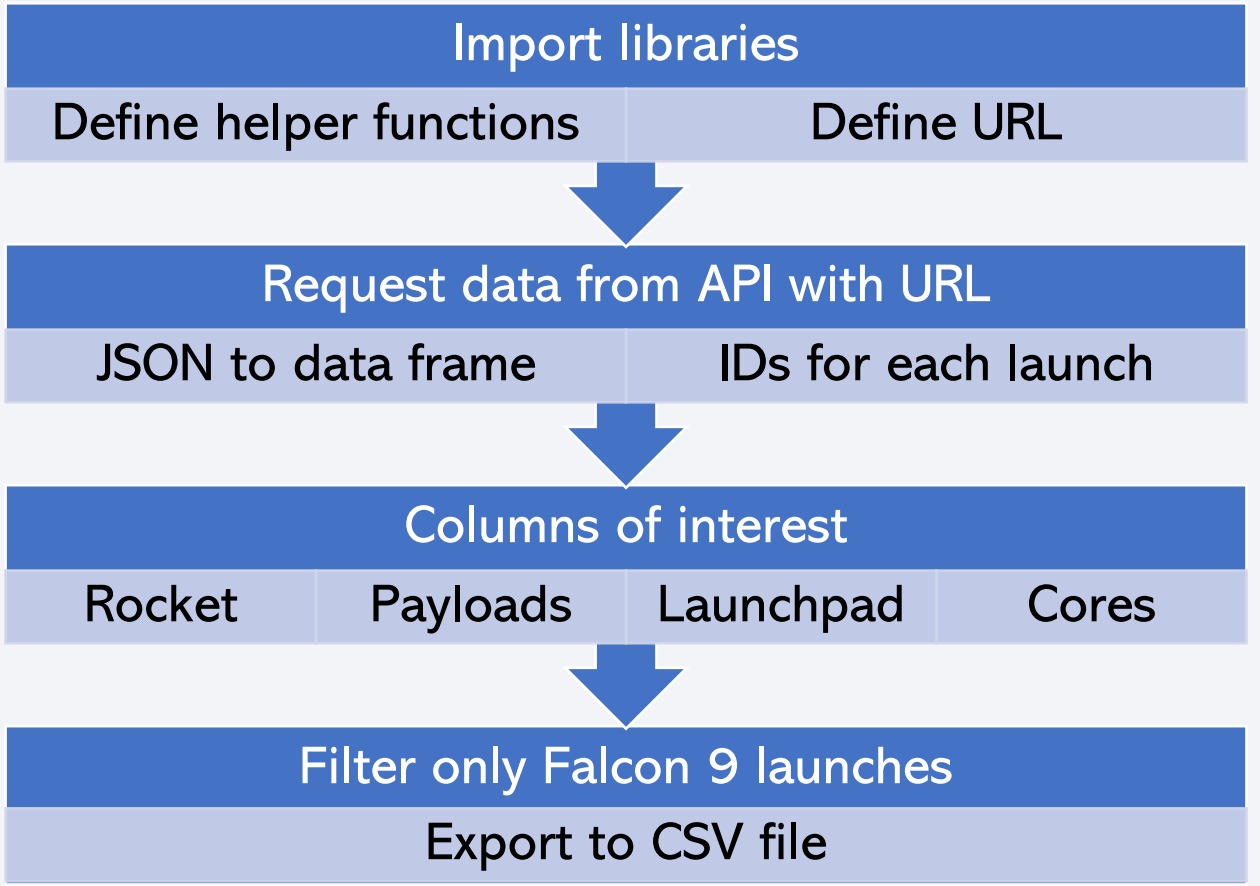
Data Collection

- The data sets were collected through SpaceX API and web scraping at Wikipedia's page "List of Falcon 9 and Falcon Heavy launches".

Request to SpaceX API	Construct data frame by reading JSON file	Clean the data and export to CSV
Request Wiki page from URL	Construct data frame by parsing HTML tables	Check the data and export to CSV

Data Collection – SpaceX API

- After importing the necessary libraries, such as requests, the get command is used with SpaceX' URL. The response is a JSON file, that was transformed to a data frame using Pandas.
- However, most of the data were only IDs. Then the API was requested again, using helper functions to get information about the launches from selected columns.
- Finally, the data frame was filtered to only include Falcon 9 launches.
- Check the Jupyter Notebook in GitHub.

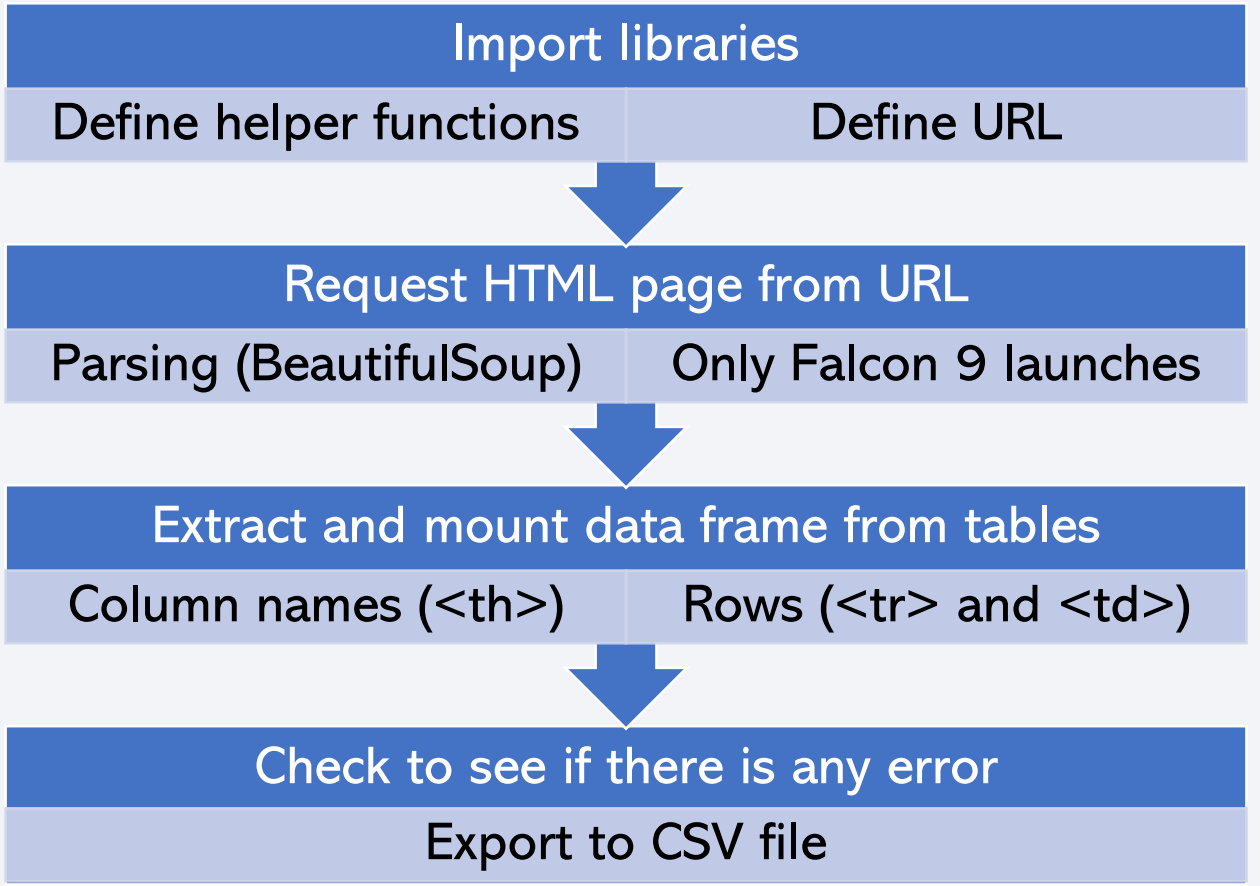


Data Collection – SpaceX API

Column	Helper Function	Informations Retrieved (new columns)
Rocket	getBoosterVersion	• Booster Version
Payloads	getPayloadData	• Payload Mass • Orbit
Launchpad	getLaunchSite	• Launch Site • Latitude • Longitude
Cores	getCoreData	• Outcome - *outcome of the landing* • Flights - *number of flights with that core* • Gridfins - *whether gridfins were used* • Reused - *whether the core is reused* • Legs - *whether legs were used* • Landing Pad - *the landing pad used* • Block - *number used to separate version of cores* • Reused Count - *number of times core has been reused* • Serial - *the serial of the core*

Data Collection – Scraping

- Similarly, in the web scraping process the get command is used to request the HTML page, which is parsed using BeautifulSoup and only the informations about Falcon 9 launches are extracted from the tables using (other) helper functions.
- Check the Jupyter Notebook in GitHub.



Data Wrangling

- Missing values:
  - First, 5 missing values of payload mass were filled by the mean;
  - Then, 26 missing values of landing pad were found and recognized to represent the situations when landing pads were not used.
- After checking column types, it was calculated:
  - The number of launches on each site;
  - The number and occurrence of each orbit;
  - The number and occurrence of mission outcome per orbit type
- Finally, the landing outcome label was created from Outcome column:
  - 0 for bad outcome;
  - 1 for successful landing.
- Check the Jupyter Notebook in GitHub.

EDA with SQL

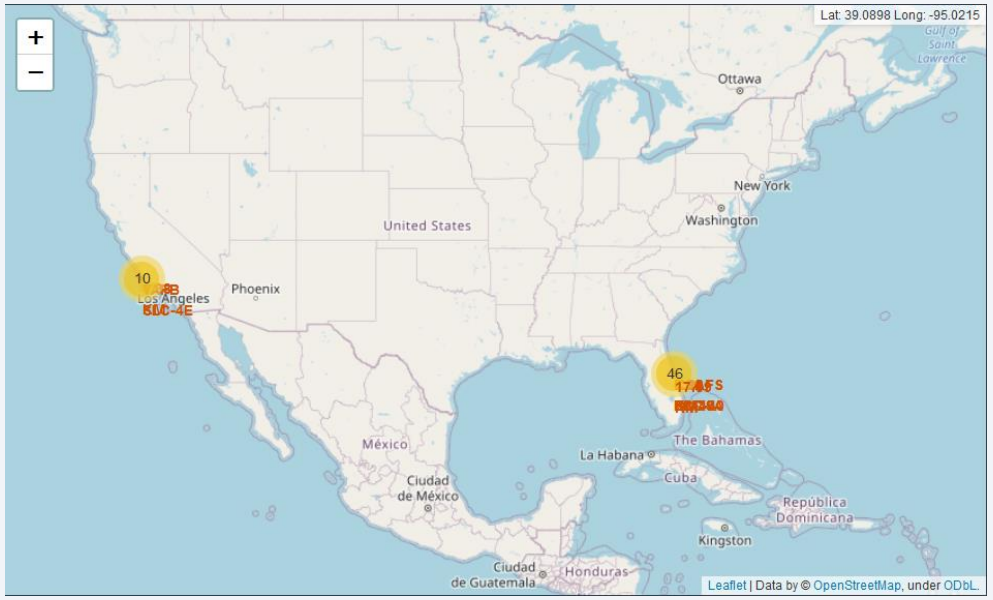
- Using SQL, it was displayed:
  - The names of the unique launch sites in the space mission;
  - The 5 records where launch sites begin with the string 'CCA';
  - The total payload mass carried by boosters launched by NASA (CRS);
  - The average payload mass carried by booster version F9 v1.1;
  - The date when the first successful landing outcome in ground pad was achieved;
  - The names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000;
  - The total number of successful and failure mission outcomes;
  - The names of the booster_versions which have carried the maximum payload mass;
  - The failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015;
  - The rank of 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;
- Check the Jupyter Notebook in GitHub.

EDA with Data Visualization

- In order to explore the data set and visualize possible relations between features, many graphs were plotted.
- Scatter plots with color indicating landing outcome:
  - Flight Number vs. Payload Mass;
  - Flight Number vs. Launch Site;
  - Payload Mass vs. Launch Site;
  - Flight Number vs. Orbit;
  - Payload Mass vs. Orbit.
- Bar plot with Success Rate for each Orbit;
- Line plot with Success Rate for each year.
- Check the Jupyter Notebook in GitHub.

Interactive Map with Folium

- Objects added to folium map:
  - A highlighted circle area with text label for each launch site, to mark its locations;
  - Marker clusters with launch outcomes for each site, to see which sites have high success rates;
  - Mouse position to get the coordinates of any point on the map;
  - Lines with calculated distances from launch sites to coastlines, railroads, highways and cities.



- [Check the Jupyter Notebook in GitHub].
- [Rendered Jupyter Notebook in nbviewer].

Dashboard with Plotly Dash

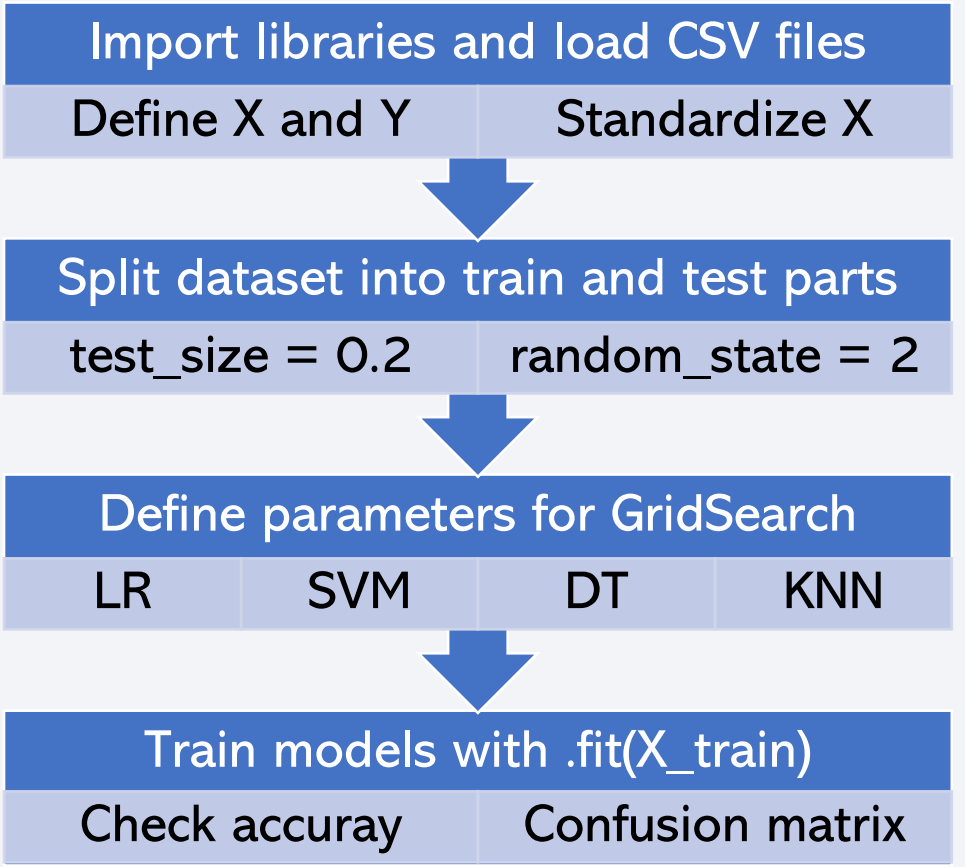
- In order to better analyze the data, an interactive dashboard was built, containing:
  - A dropdown menu to select all landing sites or a specific one;
  - A pie chart with the success rates for each site;
  - A slider bar to select the range of payload mass to display; and
  - A scatter plot with Payload Mass vs. Outcome (class 0 or 1), with color indicating Booster Version.
- Check the Python File in GitHub.

Features Engineering

- In the features engineering stage, it was defined that the predictor variables would be:
  - 'FlightNumber', 'PayloadMass', 'Orbit', 'LaunchSite', 'Flights', 'GridFins', 'Reused', 'Legs', 'LandingPad', 'Block', 'ReusedCount', 'Serial'.
- Dummy variables were created for the categorical columns:
  - 'Orbit', 'LaunchSite', 'LandingPad', 'Serial'.
- Finally, the entire data frame was cast to variable type float64.
- Check the Jupyter Notebook in GitHub (EDA with Data Visualization).

Predictive Analysis (Classification)

- After defining X to receive the predictor variables and Y to be the target variable, the predictor variables are standardized with Z-score scaling and the dataset is split into train and test sets.
- Four algorithms are used with GridSearch:
  - Logarithmic Regression (LR);
  - Support Vector Machine (SVM);
  - Decision Tree (DT); and
  - K-Nearest Neighbors (KNN).
- Finally, the 4 classification models with the best parameters found by GridSearch are evaluated checking its accuracy and confusion matrix
- Check the Jupyter Notebook in GitHub.



Predictive Analysis (Classification)

- Parameters used in GridSearch for the Logistic Regression model:

```
parameters_lr = {'C':np.arange(0.005, 1, 0.005),
                 'penalty':['l2'], # 'l1': lasso; 'l2': ridge.
                 'solver':['newton-cg','lbfgs']}
lr = LogisticRegression()
```

- Best parameters found by the Search:

```
LR tuned hyperparameters:  {'C': 0.015, 'penalty': 'l2', 'solver': 'newton-cg'}
LR training set accuracy:  0.8607142857142855
```

Predictive Analysis (Classification)

- Parameters used in GridSearch for the Support Vector Machine model:

```
parameters_svm = {'kernel': ('linear', 'rbf','poly','rbf', 'sigmoid'),
                  'C': np.logspace(-3, 3, 5),
                  'gamma': np.logspace(-3, 3, 5)}
svm = SVC()
```

- Best parameters found by the Search:

```
SVM tuned hyperparameters:  {'C': 1.0, 'gamma': 0.031622776601683793, 'kernel': 'sigmoid'}
SVM training set accuracy:  0.8482142857142856
```

Predictive Analysis (Classification)

- Parameters used in GridSearch for the Decision Tree model:

```
parameters_tree =
{'criterion': ['gini', 'entropy'],
 'splitter': ['best', 'random'],
 'max_depth': [2*n for n in range(1,10)],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10]}
tree = DecisionTreeClassifier()
```

- Best parameters found by the search:

```
DT tuned hyperparameters:
{'criterion': 'entropy',
 'max_depth': 10,
 'max_features': 'sqrt',
 'min_samples_leaf': 1,
 'min_samples_split': 10,
 'splitter': 'random'}
DT training set accuracy:  0.8892857142857142
```

Predictive Analysis (Classification)

- Parameters used in GridSearch for the K-Nearest Neighbors model:

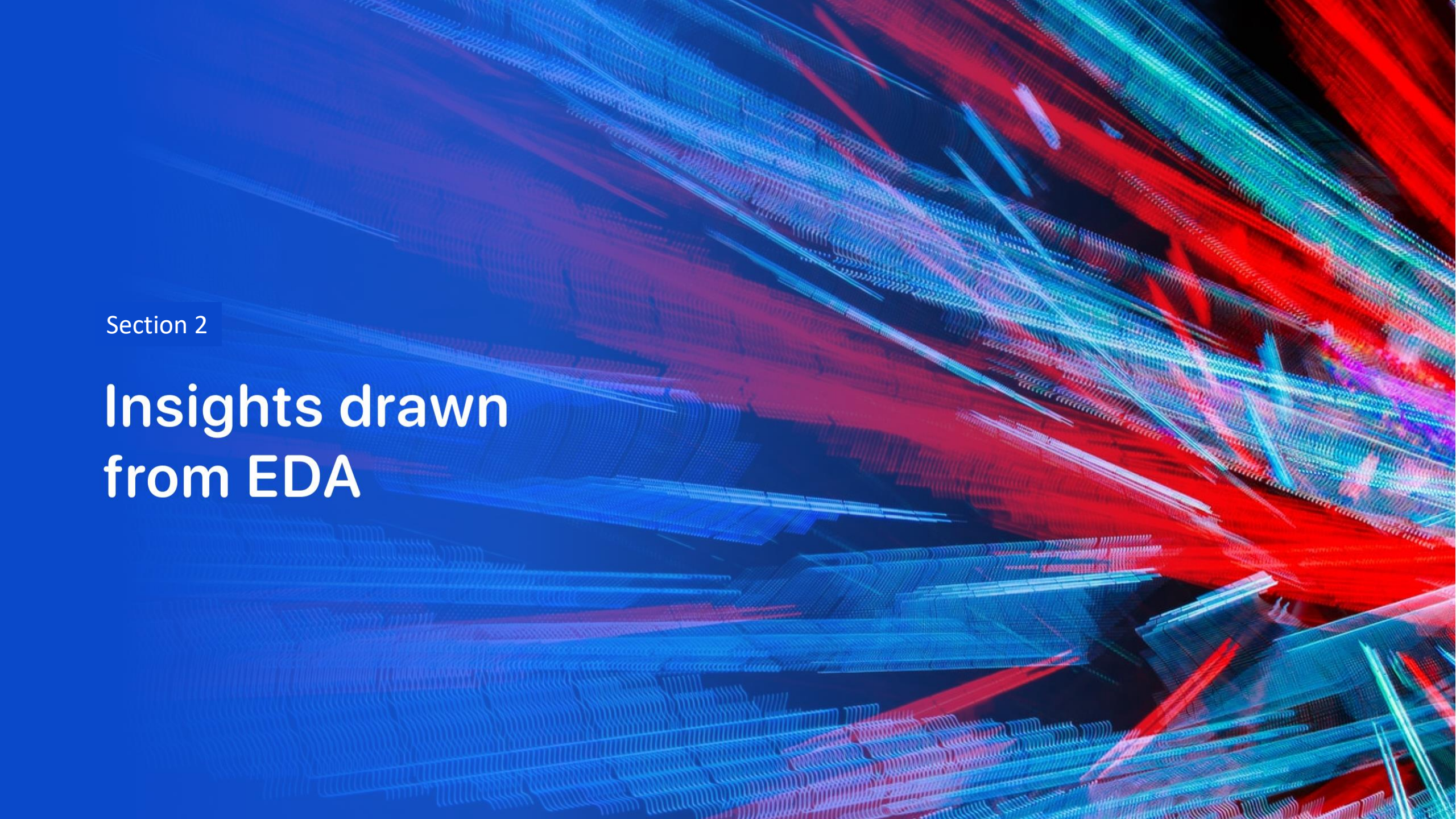
```
parameters_knn = {'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
                  'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
                  'p': [1,2]}
knn = KNeighborsClassifier()
```

- Best parameters found by the search:

```
KNN tuned hyperparameters:  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
KNN training set accuracy:  0.8482142857142858
```

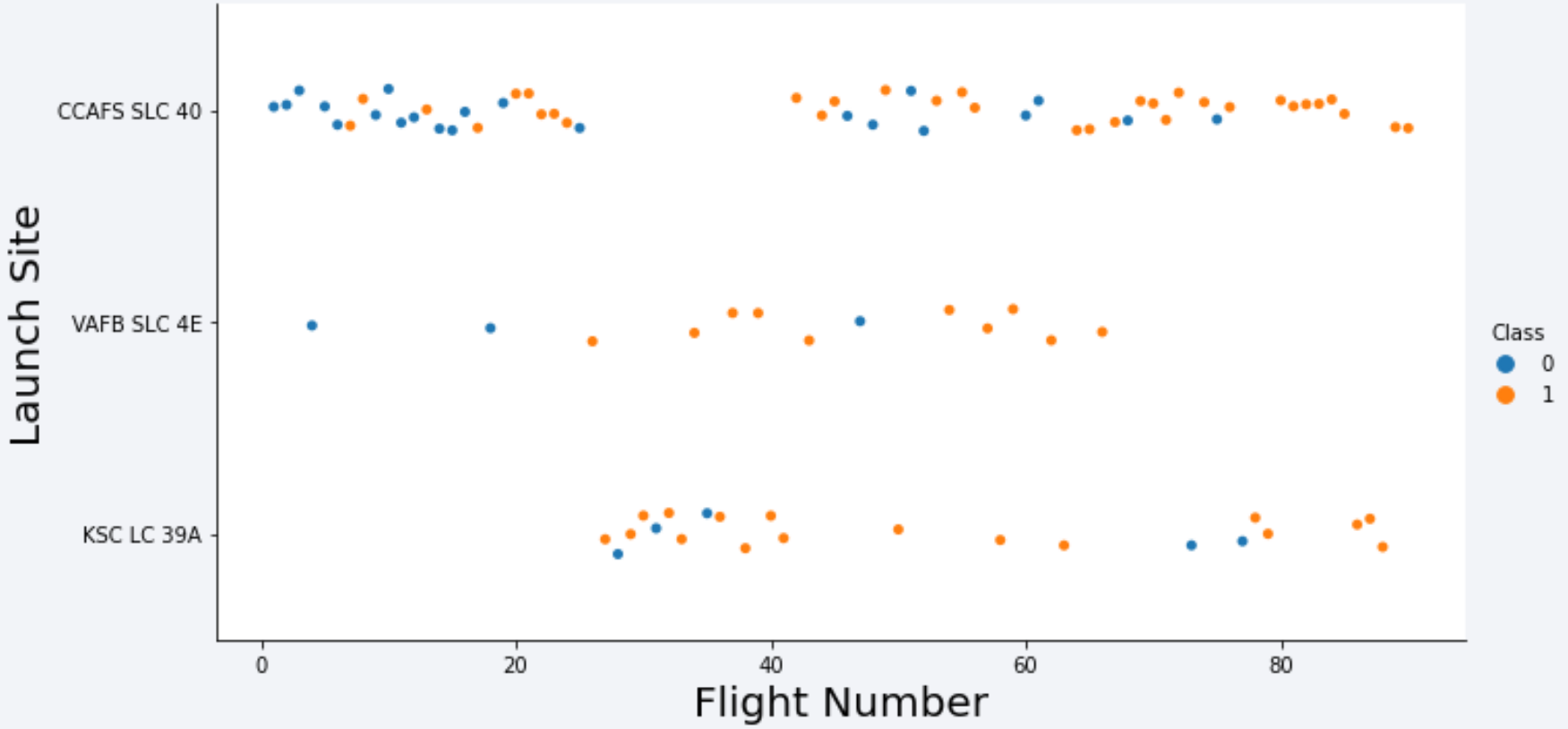
Results

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


Section 2
Insights drawn from EDA

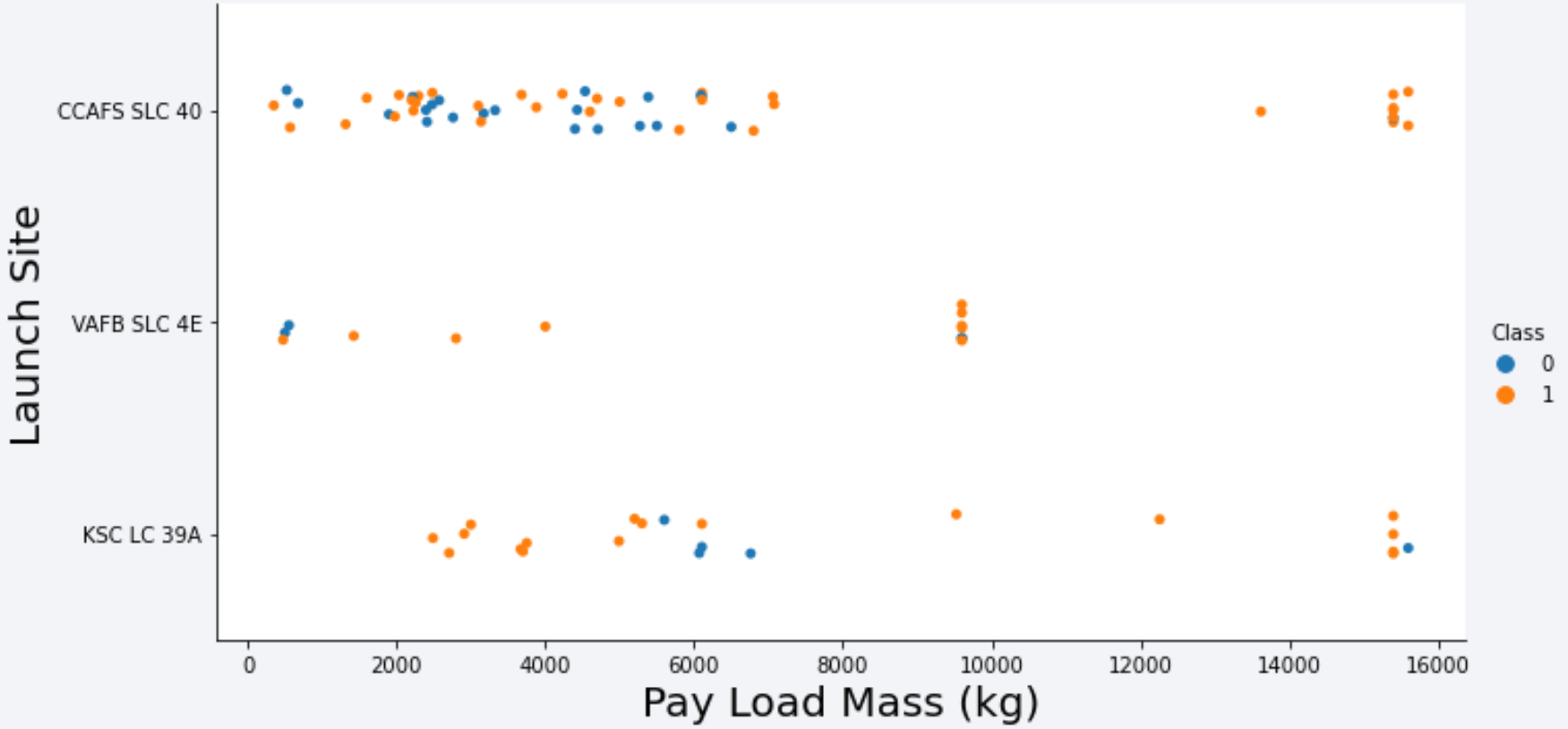
Flight Number vs. Launch Site

- The launch site CCAFS SLC 40 has a high number of failures, but was used much more times.



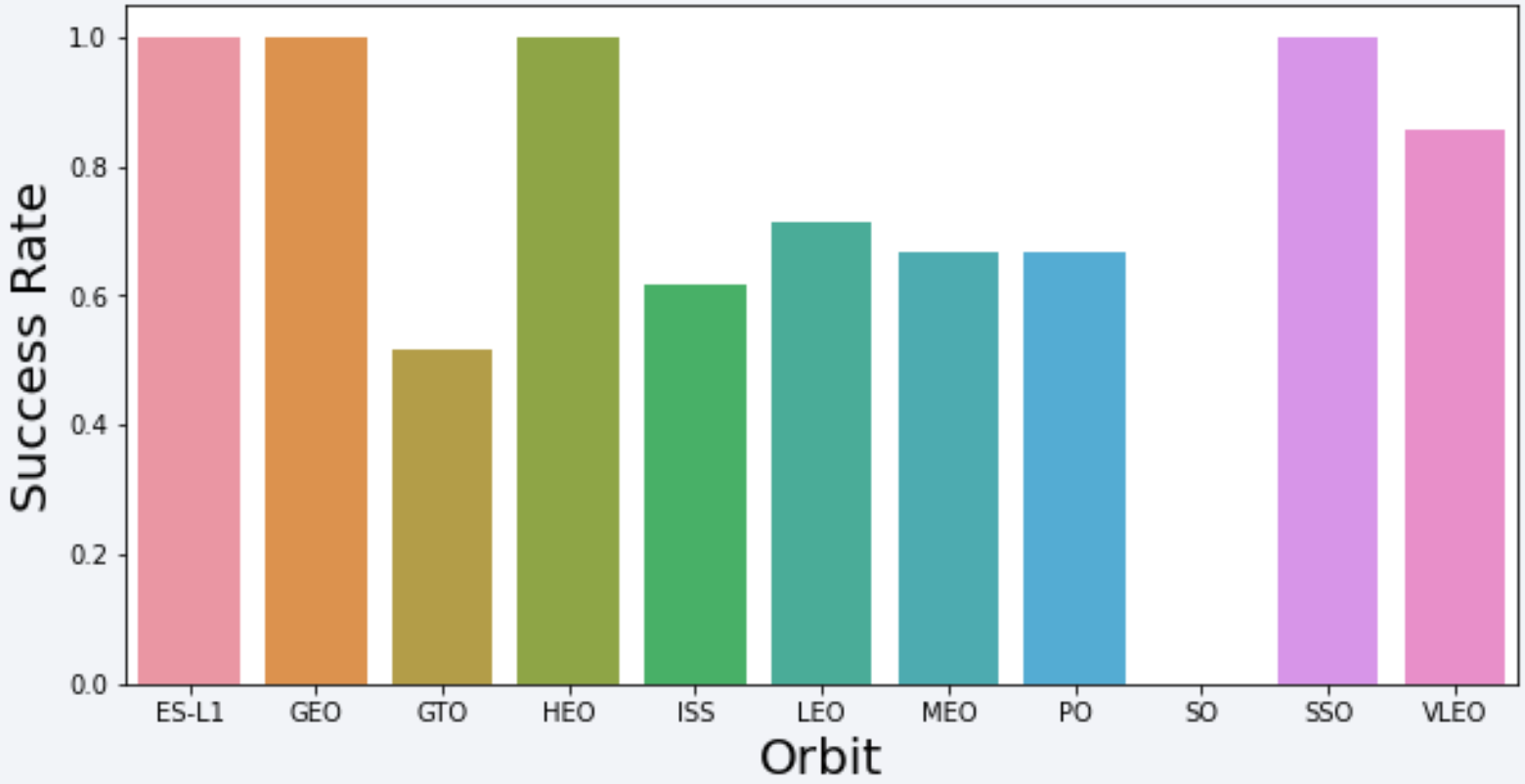
Payload vs. Launch Site

- For the VAFB-SLC launch site there are no rockets launched for heavy payload mass (greater than 10000).



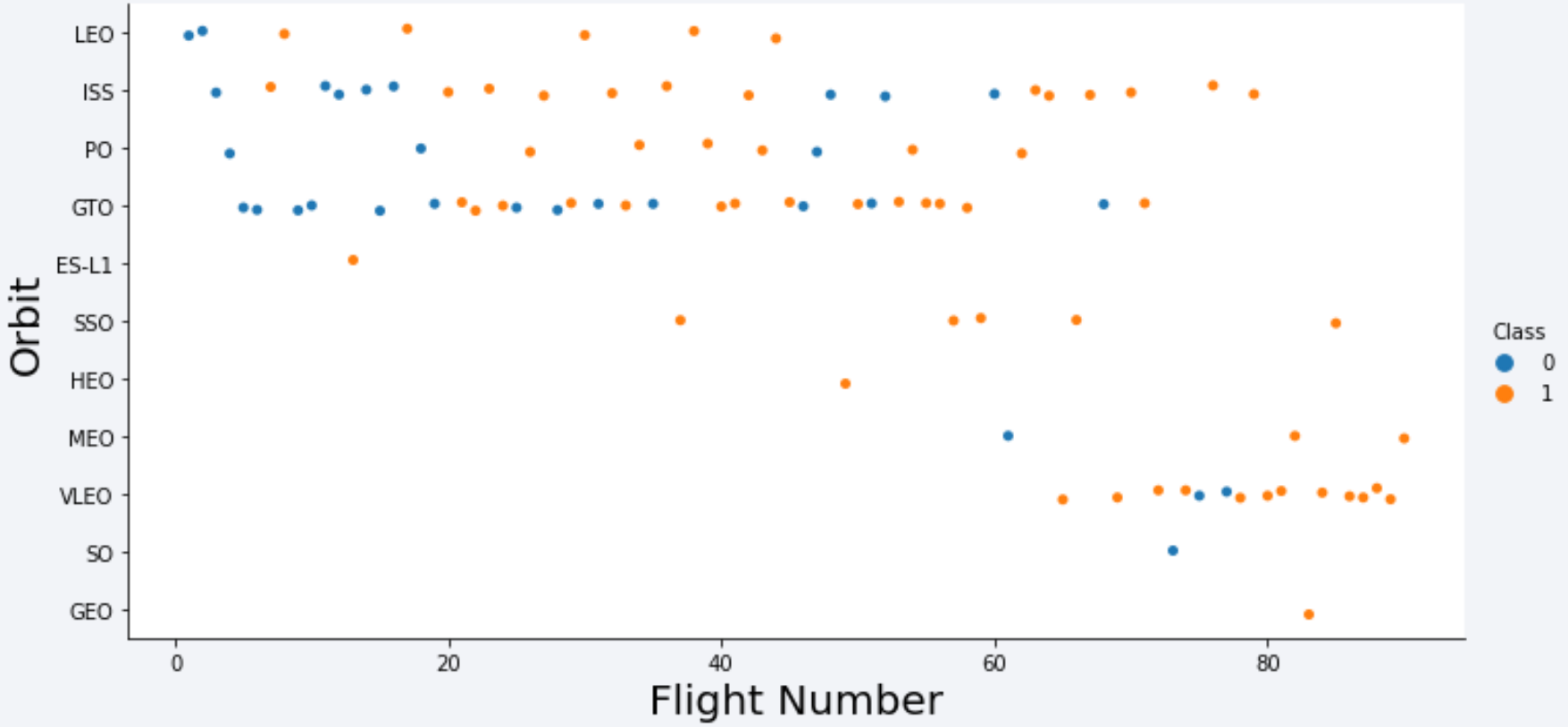
Success Rate vs. Orbit Type

- In 4 orbits the success rate is 100%: ES-L1, GEO, HEO and SSO.



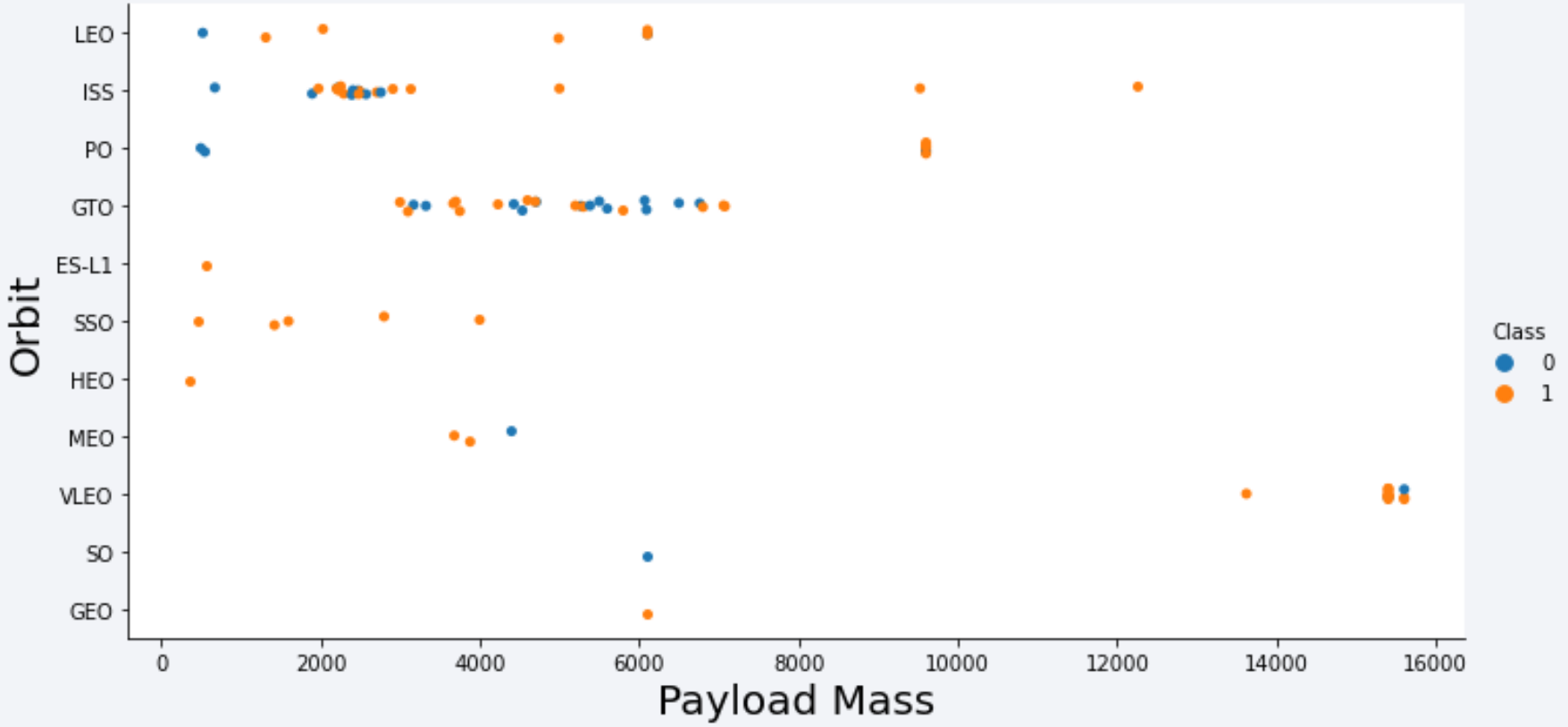
Flight Number vs. Orbit Type

- In the 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.



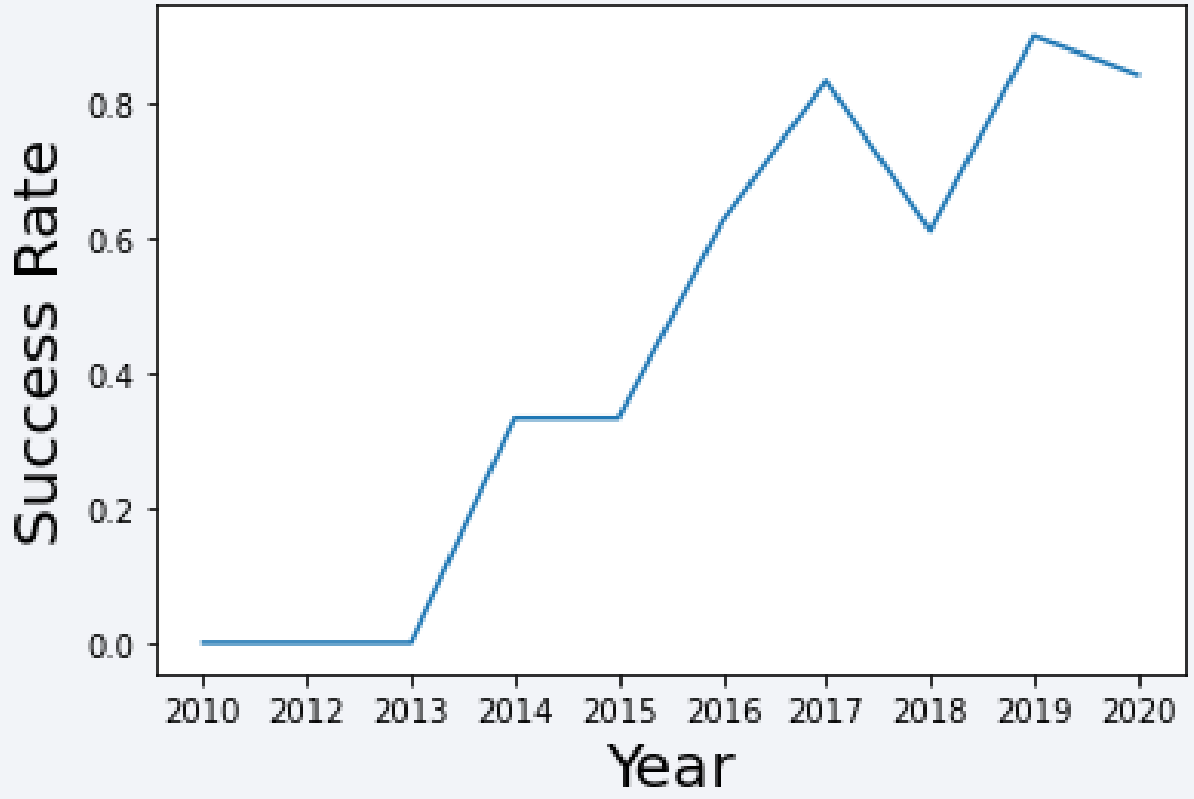
Payload vs. Orbit Type

- 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

- The overall success rate since 2013 kept increasing till 2020.



All Launch Site Names

- We may use the DISTINCT function to find the names of the unique launch sites:

```
%%sql
SELECT DISTINCT(launch_site) FROM spacextbl
```

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

Launch Site Names Begin with 'CCA'

- The LIKE command allows us to search for substrings using the percentage symbol:

```
%%sql
SELECT * FROM spacextbl
WHERE launch_site LIKE 'CCA%' LIMIT 5
```

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

- The SUM function allows us to calculate the total payload carried by boosters, and the WHERE command let us filter only those from NASA (CRS):

```
%%sql
SELECT SUM(payload_mass__kg_)
AS total_payload_mass_carried_by_boosters_launched_by_NASA_CRS
FROM spacextbl WHERE customer='NASA (CRS)'
```

total_payload_mass_carried_by_boosters_launched_by_nasa_crs
45596

Average Payload Mass by F9 v1.1

- The AVG function allows us to calculate the average payload mass carried by booster version F9 v1.1:

```
%%sql
SELECT AVG(payload_mass__kg_) AS average_payload_mass FROM spacextbl
WHERE booster_version LIKE 'F9 v1.1%'
```

average_payload_mass
2534

First Successful Ground Landing Date

- The MIN function may be used to find the dates of the first successful landing outcome on ground pad:

```
%%sql
SELECT MIN(date) AS date_of_the_first_successful_landing_in_ground_pad
FROM spacextbl
WHERE landing__outcome = 'Success (ground pad)'
```

date_of_the_first_successful_landing_in_ground_pad
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We may add two restrictions through command WHERE using logical AND together with BETWEEN command to 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 DISTINCT(booster_version) FROM spacextbl
WHERE landing__outcome = 'Success (drone ship)'
AND payload_mass__kg_ BETWEEN 4000 AND 6000
```

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

Total Number of Successful and Failure Mission Outcomes

- We may use the COUNT function together with a restriction to calculate the total number of successful and failure mission outcomes:

```
%%sql
SELECT COUNT(mission_outcome) AS number_of_successfuls
FROM spacextbl WHERE mission_outcome LIKE 'Success%'
```

number_of_successfuls
100

```
%%sql
SELECT COUNT(mission_outcome) AS number_of_failures
FROM spacextbl WHERE mission_outcome LIKE 'Failure%'
```

number_of_failures
1

Boosters Carried Maximum Payload

- We use the MAX function in a subquery inside the restriction of the WHERE command to list the names of the booster which have carried the maximum payload mass:

```
%%sql
SELECT DISTINCT(booster_version)
AS booster_versions_with_maximum_payload_mass FROM spacextbl
WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM spacextbl)
```

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

2015 Launch Records

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

```
%%sql
SELECT landing__outcome, booster_version, launch_site FROM spacextbl
WHERE landing__outcome = 'Failure (drone ship)' AND date LIKE '2015%'
```

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- Finally, we may use GROUP BY and ORDER BY commands, together with COUNT function and a BETWEEN command inside a WHERE restriction to rank the count of landing outcomes between 2010-06-04 and 2017-03-20, in descending order:

```
%%sql
SELECT landing__outcome, COUNT(landing__outcome) AS number_of_landings
FROM spacextbl WHERE date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY landing__outcome ORDER BY number_of_landings DESC
```

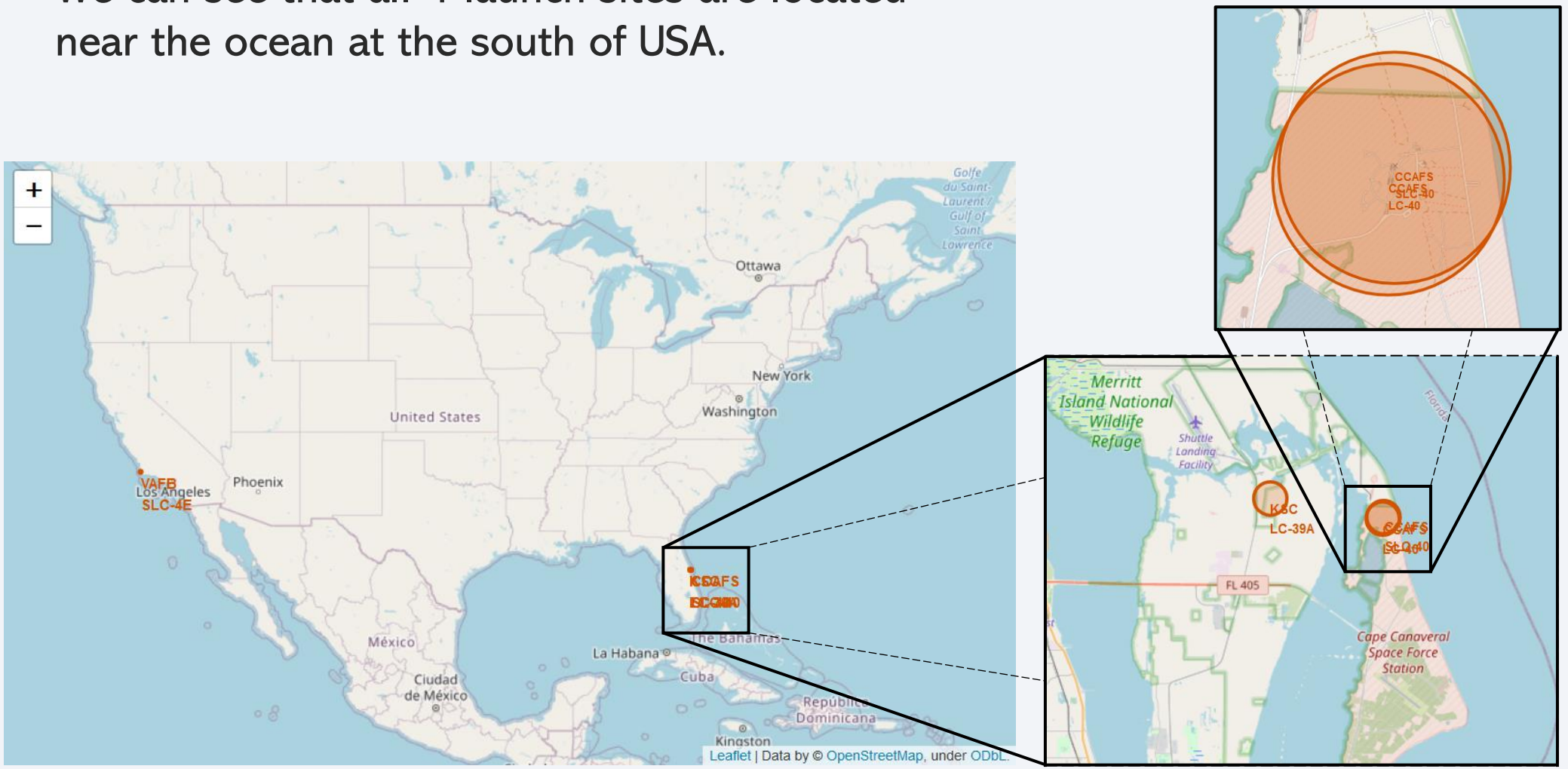
landing__outcome	number_of_landings
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

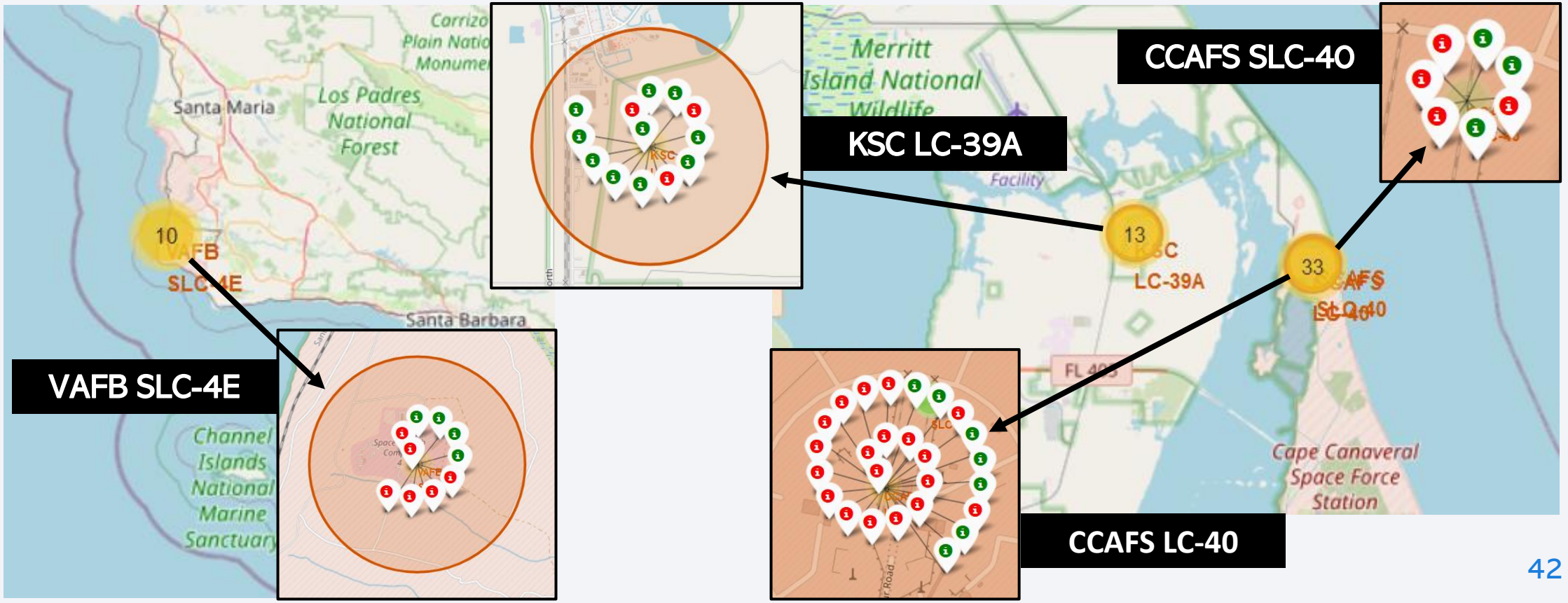
Launch Sites Locations

- We can see that all 4 launch sites are located near the ocean at the south of USA.



Launch Outcomes Marked in Color Labels

- Red for failure and green for success.
- The launching site with highest successful rate is clearly KSC LC-39A.

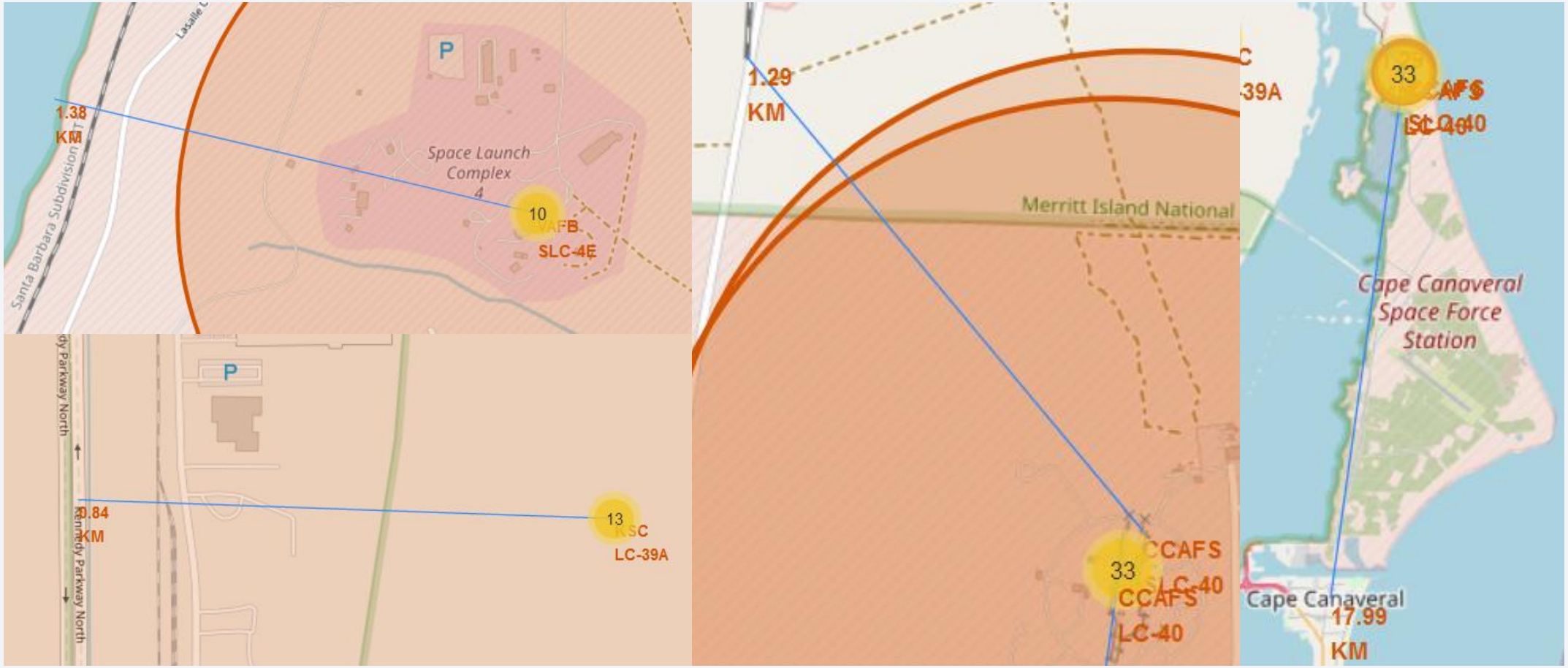


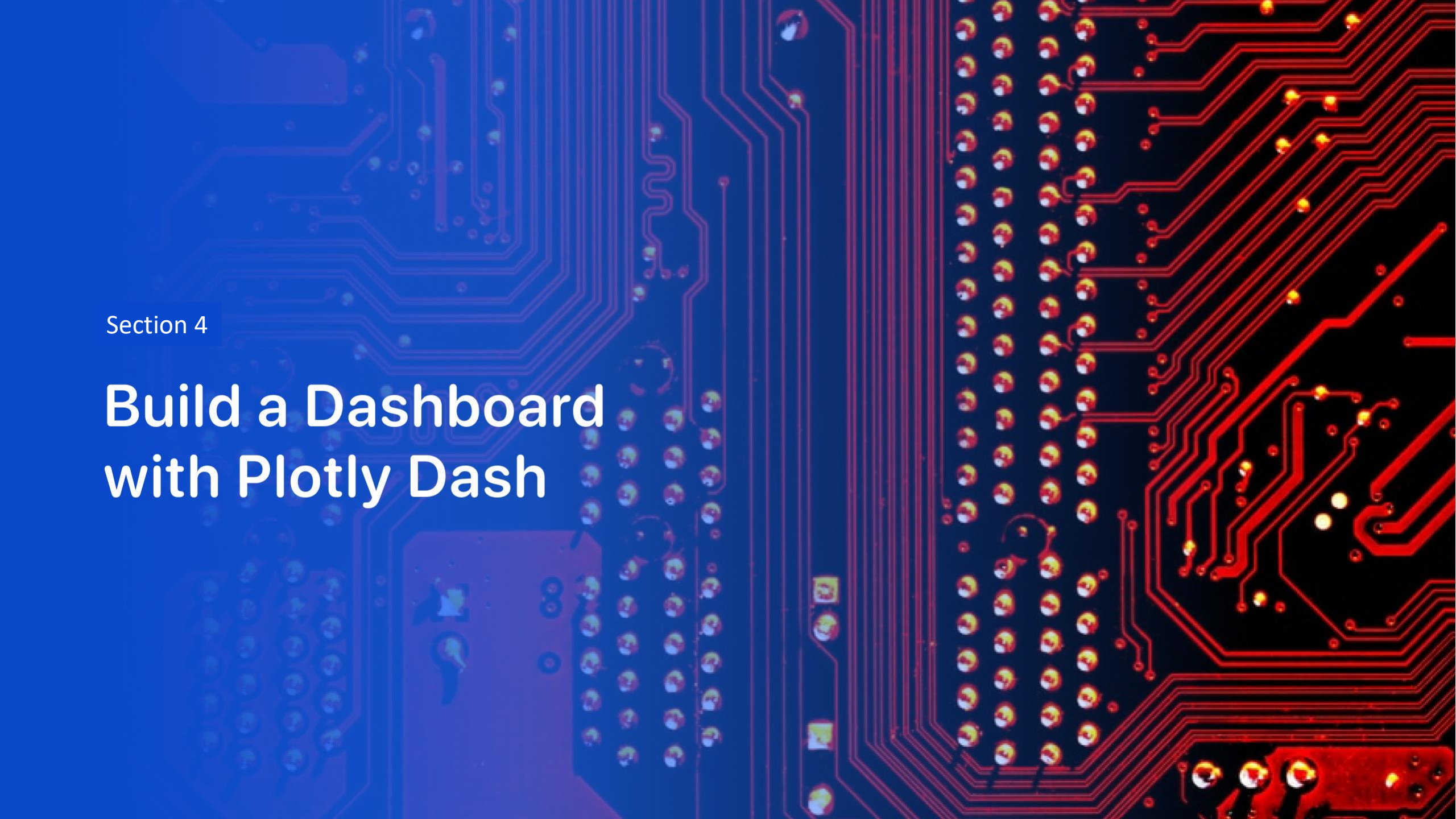
Distances from Launch Sites to its Proximities

- The launch sites are generally located near highways, coastlines and railroads, and away from cities. Some examples:
  - 1.4 km from VAFB SLC-4E to nearest coastline;
  - 0.8 km from KSC LC-39A to nearest highway;
  - 1.3 km from CCAFS SLC-40 to nearest railroad;
  - 18 km from CCAFS LC-40 to nearest city.

Distances from Launch Sites to its Proximities

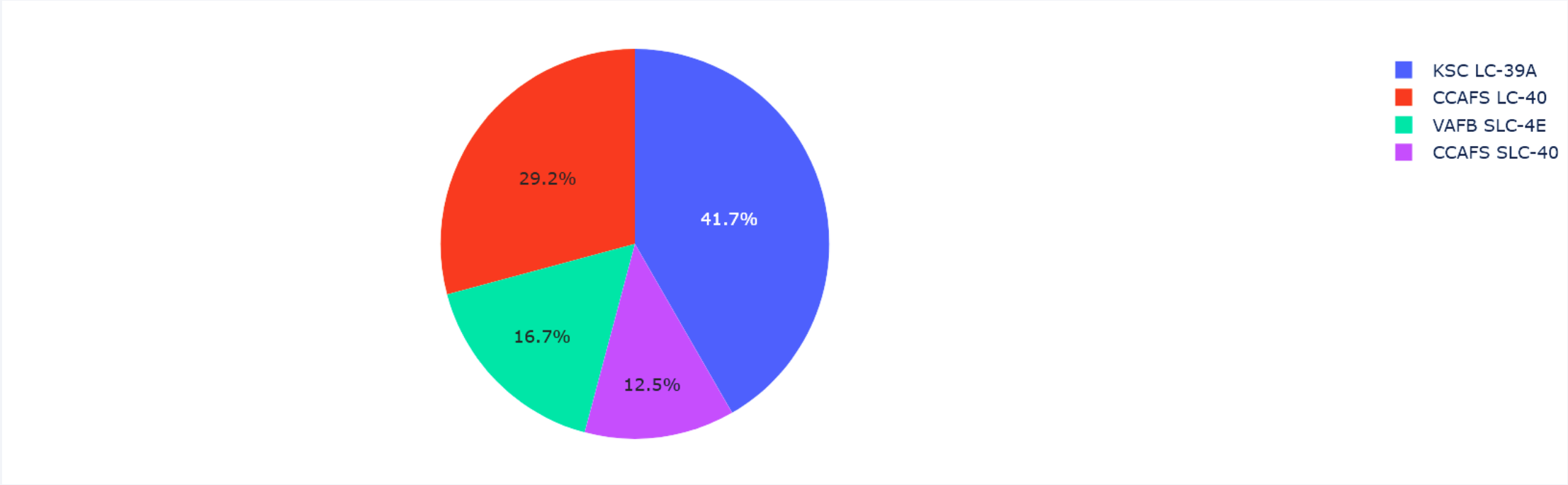
- Screenshots:




Section 4
Build a Dashboard with Plotly Dash

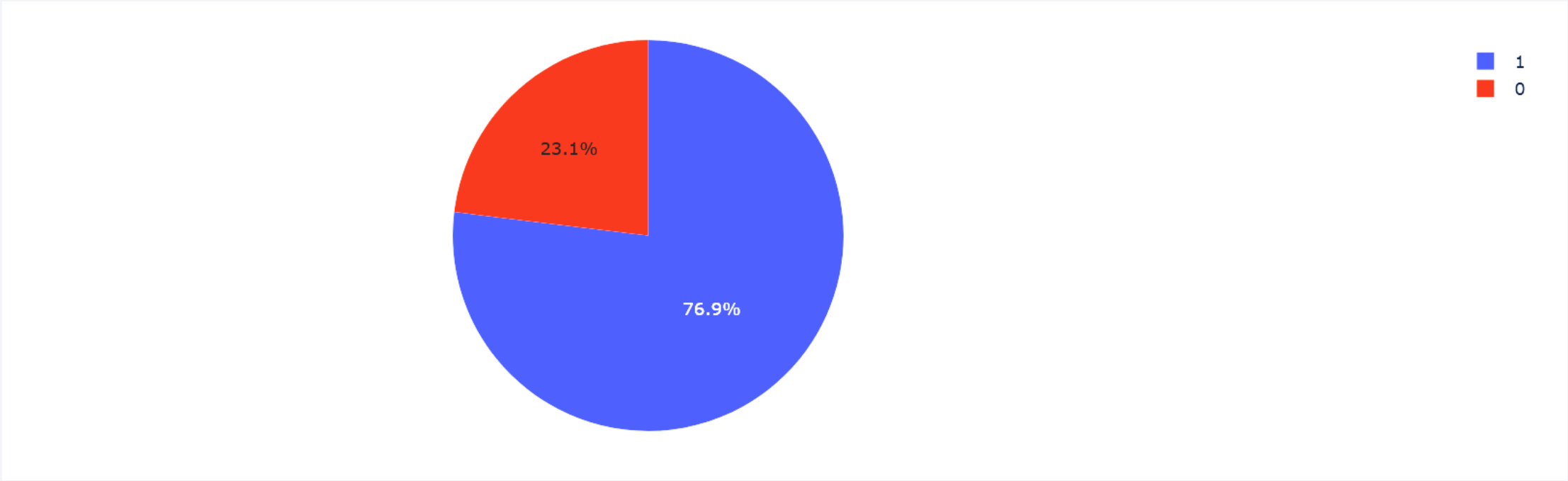
Total Success Launches by Site

- The launch site KSC LC-39A has the highest percentage of the total number of successful launches, confirming what was observed at the map.



Total Success Launches for Site KSC LC-39A

- In fact, site KSC LC-39A also has the highest successful rate considering each site individually, winning by far with 76.9%.



Correlation Between Payload and Success for All Sites

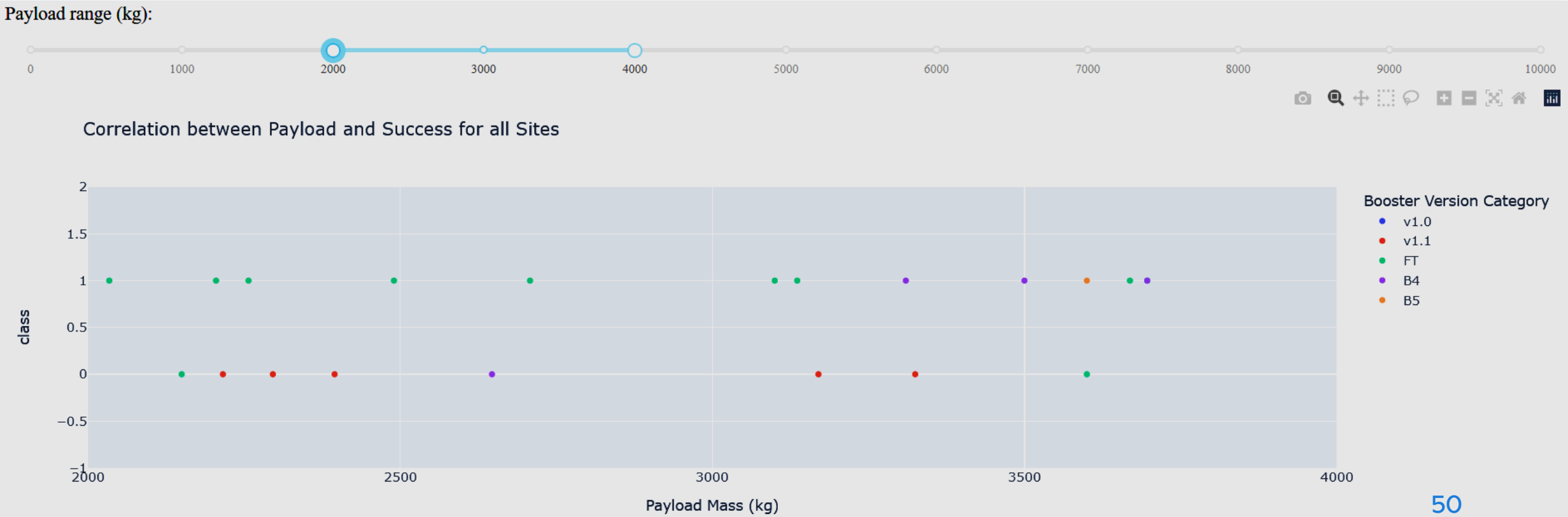
- Full range: 0 – 10,000 kg. Number of records: 56. Successful landings: 24.
- Success rate: 42.9%.

Correlation Between Payload and Success for All Sites

- Range: 0 – 2,000 kg. Number of records: 10. Successful landings: 3.
- Success rate: 30%.

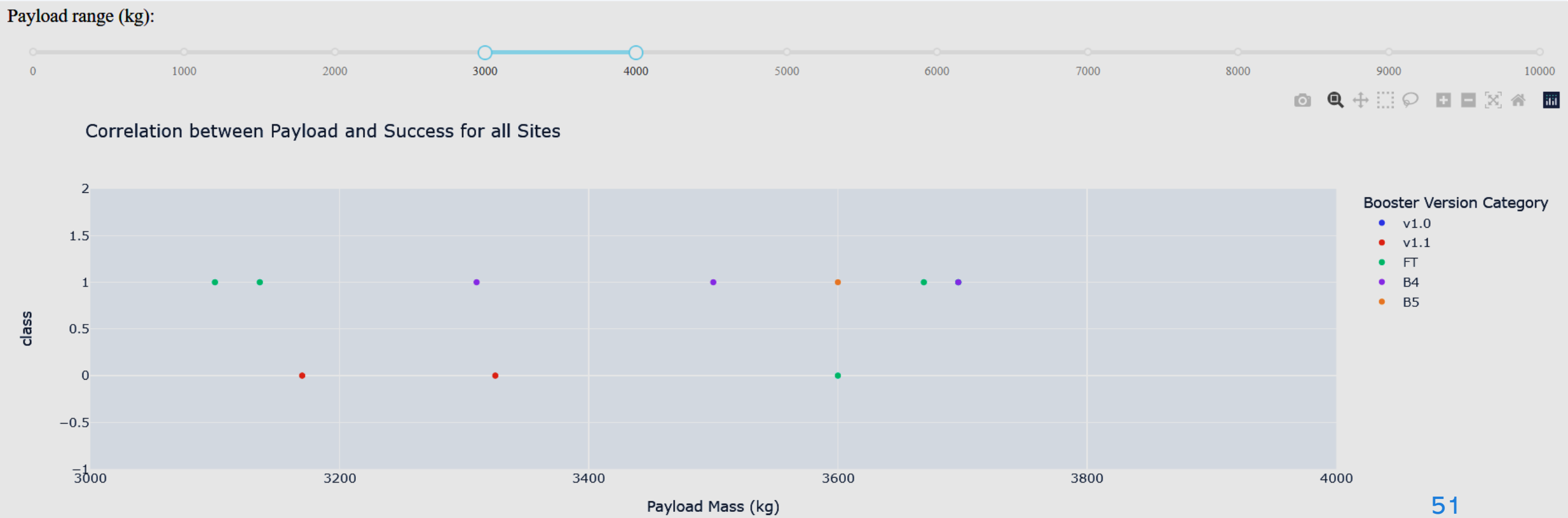
Correlation Between Payload and Success for All Sites

- Range: 2,000 – 4,000 kg. Number of records: 20. Successful landings: 12.
- Success rate: 60%.



Correlation Between Payload and Success for All Sites

- Range: 3,000 – 4,000 kg. Number of records: 10. Successful landings: 7.
- Success rate: 70%.

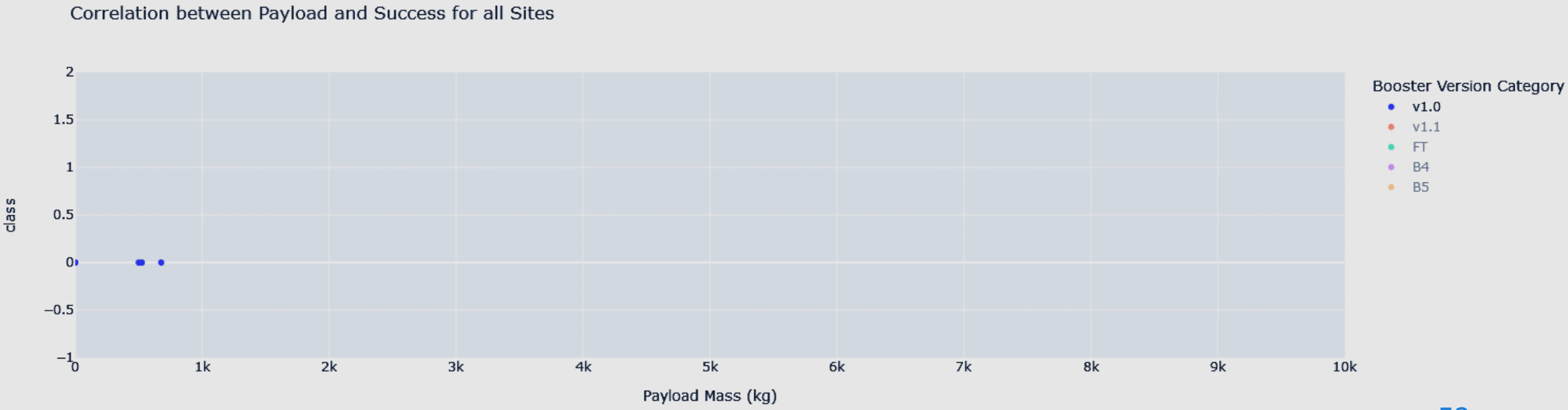


Correlation Between Payload and Success for All Sites

- Range: 5,000 – 10,000 kg. Number of records: 11. Successful landings: 3.
- Success rate: 27.3%.

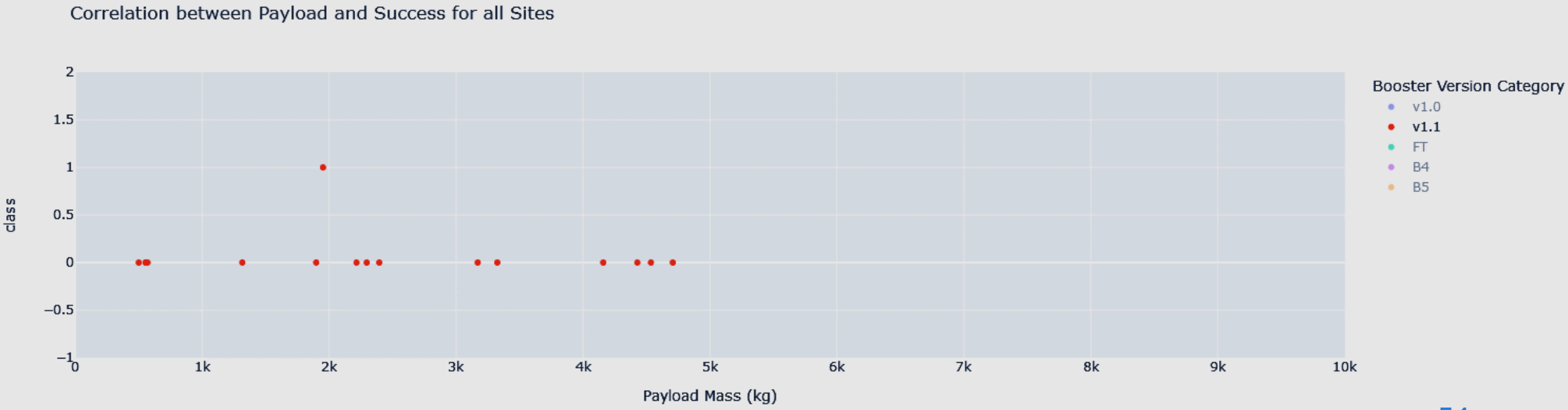
Success Rate by Booster Version

- Booster: v1.0. Number of records: 5. Successful landings: 0.
- Success rate: 0%.



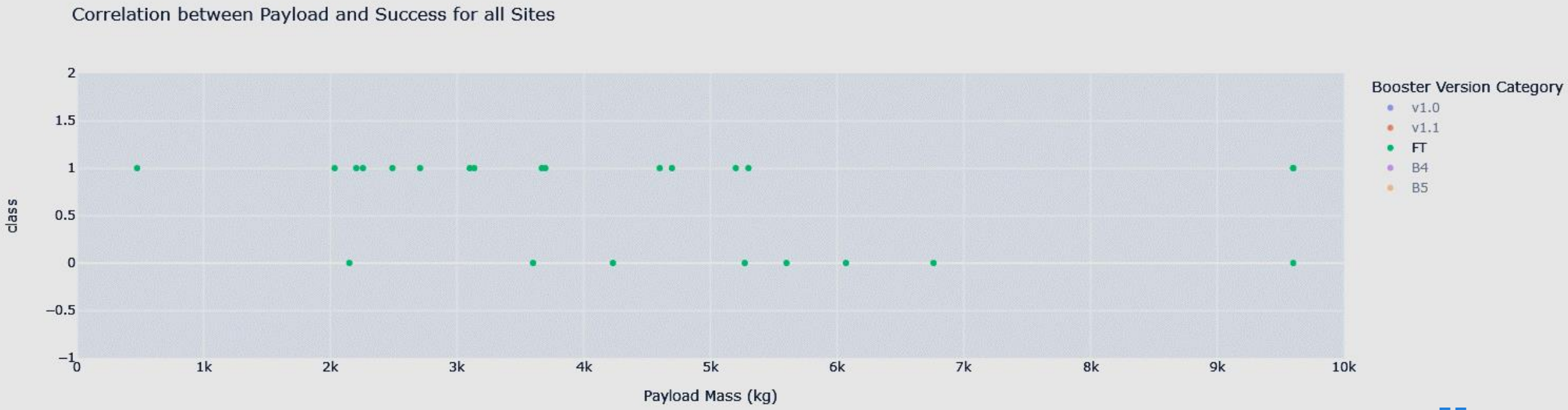
Success Rate by Booster Version

- Booster: v1.1. Number of records: 15. Successful landings: 1.
- Success rate: 6.7%.



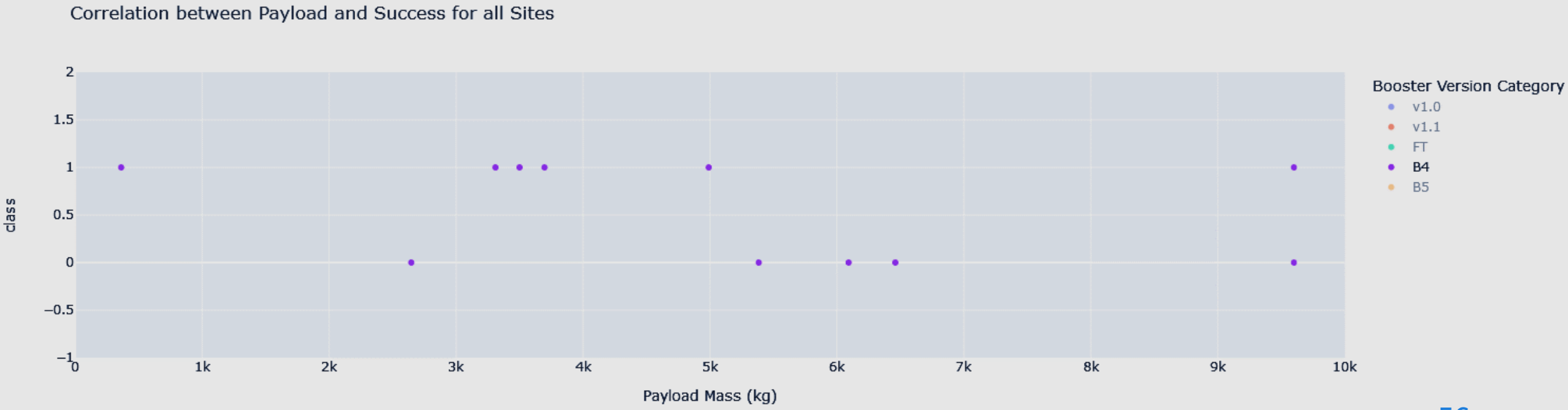
Success Rate by Booster Version

- Booster: FT. Number of records: 24. Successful landings: 16.
- Success rate: 66.7%.



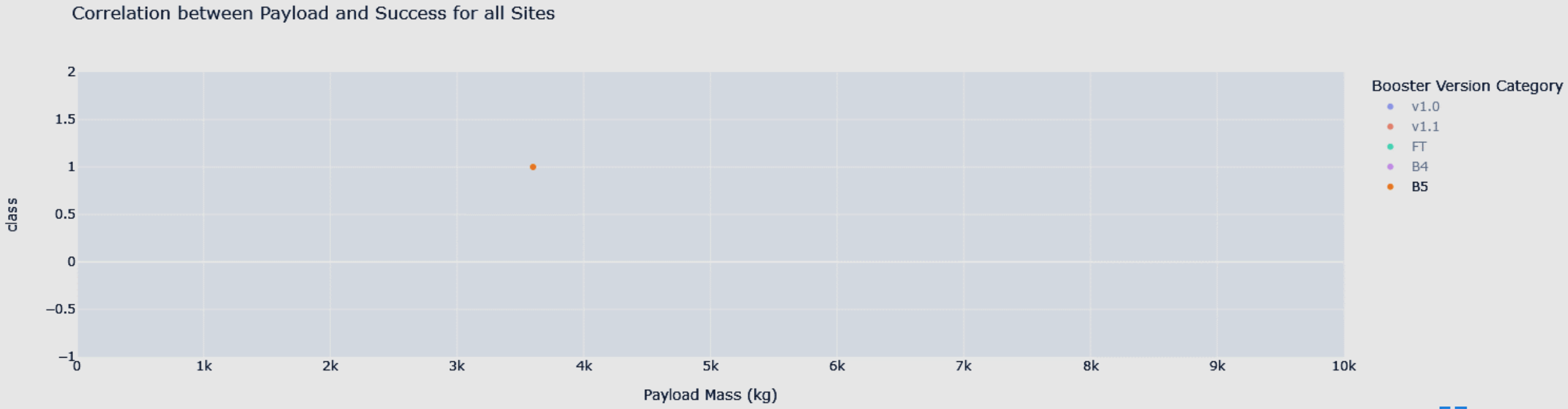
Success Rate by Booster Version

- Booster: B4. Number of records: 11. Successful landings: 6.
- Success rate: 54.5%.



Success Rate by Booster Version

- Booster: B5. Number of records: 1. Successful landings: 1.
- Success rate: 100%.



Answers about Payload Range and Booster Version

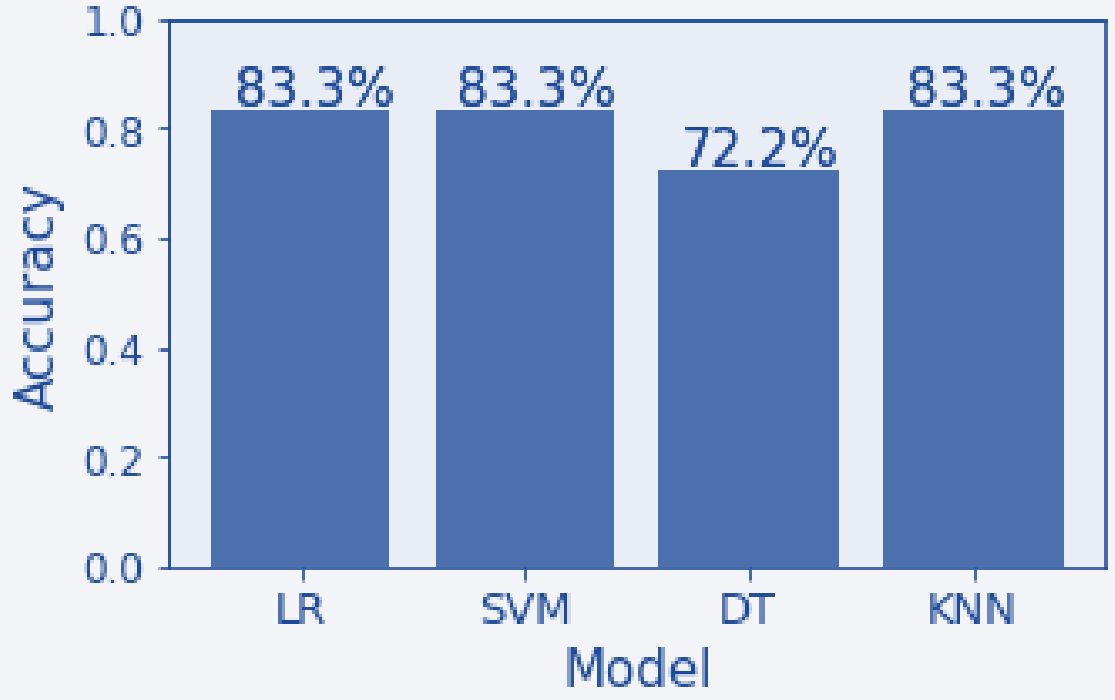
- Now we may answer some questions about the dataset:
  - Which payload range(s) has the highest launch success rate?
    - *Between 3,000 and 4,000 kg (70%).*
  - Which payload range(s) has the lowest launch success rate?
    - *Between 5,000 and 10,000 kg (27.3%).*
  - Which F9 Booster version has the highest launch success rate?
    - *FT (66.7%).*
    - *This is because B5 has been used only once, so must be disregarded.*

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The accuracy obtained over the test data was 83% for Logistic Regression, Support Vector Machine and K-Nearest Neighbors models, and 72% for the Decision Tree.



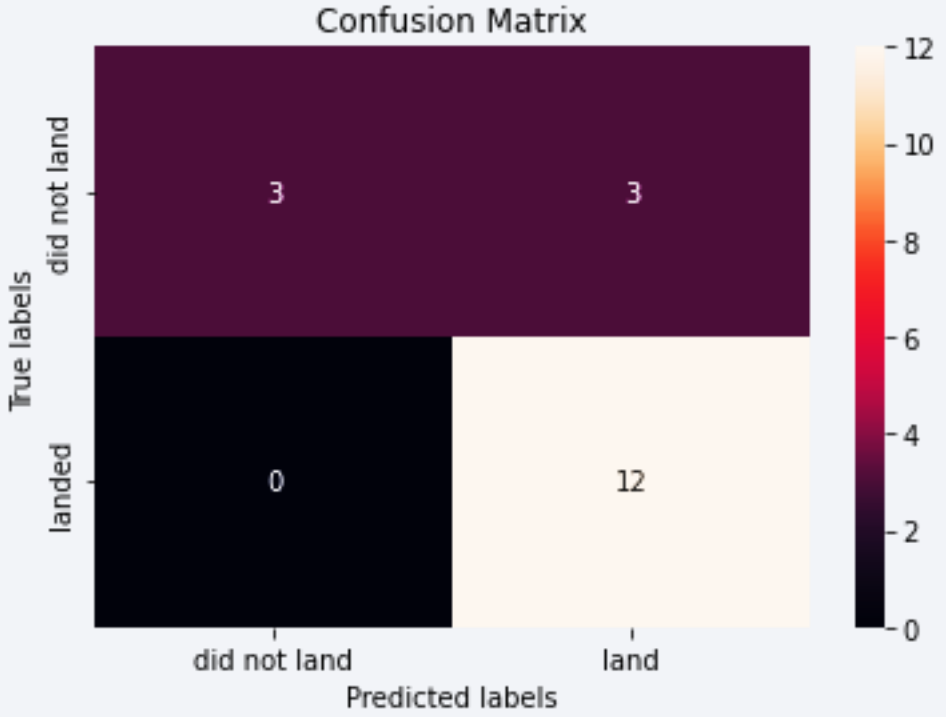
Confusion Matrix

- The confusion matrix is a way of organizing the results of binary classification algorithms. Its basic structure is given by:

Real Negatives	True Negatives	False Positives
Real Positives	False Negatives	True Positives
	Predicted Negatives	**Predicted Positives**

Confusion Matrix

- For LR, SVM and KNN models:



- For the Decision Tree:

Classification Report

- For LR, SVM and KNN models:

	precision	recall	f1-score	support
0	1.00	0.50	0.67	6
1	0.80	1.00	0.89	12
accuracy			0.83	18
macro avg	0.90	0.75	0.78	18
weighted avg	0.87	0.83	0.81	18

- For the Decision Tree:

	precision	recall	f1-score	support
0	0.60	0.50	0.55	6
1	0.77	0.83	0.80	12
accuracy			0.72	18
macro avg	0.68	0.67	0.67	18
weighted avg	0.71	0.72	0.72	18

Conclusions

- The information obtained at this study may be summarized as follows:
  - The best orbits are ES-L1, GEO, HEO and SSO.
  - The best launching site is KSC LC-39A.
  - The best payload mass range is between 3,000 and 4,000 kg.
  - The best booster version category is FT.
  - The best machine learning models for prediction are LR, SVM and KNN.
- Furthermore, SpaceX's annual success rate has been shown to be steadily increasing.
- Therefore, the information obtained here could help the company to maintain its growing performance.

Thank you!