IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Dennis de Mol
20-10-2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result from Machine Learning Lab

Introduction

SpaceX is a revolutionary company who has disrupt the space industry by offering a rocket launches specifically Falcon 9 as low as 62 million dollars; while other providers cost upward of 165 million dollar each. Most of this saving thanks to SpaceX astounding idea to reuse the first stage of the launch by re-land the rocket to be used on the next mission. Repeating this process will make the price down even further. As a data scientist of a startup rivaling SpaceX, the goal of this project is to create the machine learning pipeline to predict the landing outcome of the first stage in the future. This project is crucial in identifying the right price to bid against SpaceX for a rocket launch.

The problems included:

- Identifying all factors that influence the landing outcome.
- The relationship between each variables and how it is affecting the outcome.
- The best condition needed to increase the probability of successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collection occurred using SpaceX REST API and Wikipedia scraping
- Perform data wrangling
  - The data was one-hot encoded for categorical values
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- The data is collected using the SpaceX REST API. Using the get request and transforming via JSON files to a Pandas Dataframe.
- More data was gathered using webscraping the Wikipedia page of the launches. BeautifulSoup was used to extract the tables which were converted to again a Pandas Dataframe.

Data Collection – SpaceX API

- The URL is identified
- The JSON is converted to a Dataframe
- The data is changed to the correct formt

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

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

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have multiple payloads in a single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

- https://github.com/dennisdemol/Coursera/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

- Define Wikipedia URL
- Make Soup object
- Find table on page
- Extract table and process to Dataframe

- https://github.com/dennisdemol/Coursera/blob/main/jupyter-labs-webscraping.ipynb

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url).text
```

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
column_names = []

# 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
temp = first_launch_table.find_all('th')

for x in range(len(temp)):
    try:
        name = extract_column_from_header(temp[x])
        if (name is not None and len(name)>0):
            column_names.append(name)
    except:
        pass
```

```
headings = []
for key,values in dict(launch_dict).items():
    if key not in headings:
        headings.append(key)
    if values is None:
        del launch_dict[key]


def dict_list(dict_list, padel):
    lmax = 0
    for lname in dict_list.keys():
        lmax = max(lmax, len(dict_list[lname]))
    for lname in dict_list.keys():
        ll = len(dict_list[lname])
        if  ll <lmax:
            dict_list[lname] += [padel] * (lmax - ll)
    return dict_list

dict_list(launch_dict,0)

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

Data Wrangling

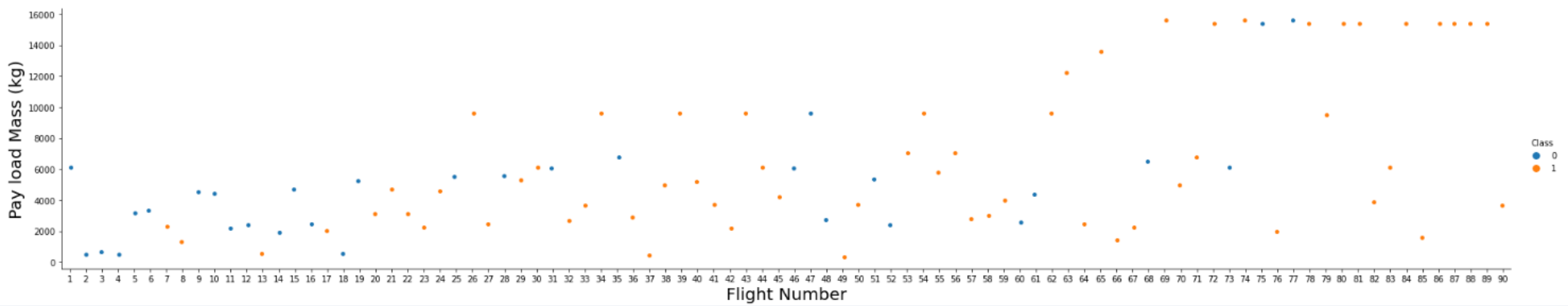
- By data wrangling the data is cleaned and organized. This is done such that the data can be used for the exploratory data analysis.
- In this lab the number of launches of each site were calculated, as well as the mission outcome per orbit type.
- Lastly, a landing outcome label is computed for easier processing

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

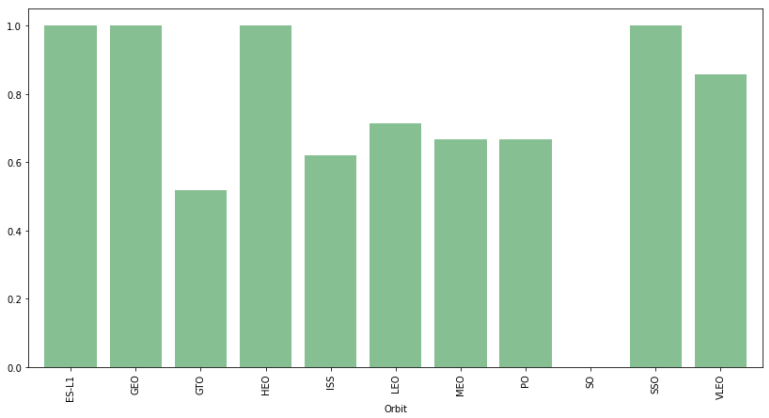
- https://github.com/dennisdemol/Coursera/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- For the EDA we were first interest in the change in payload over time



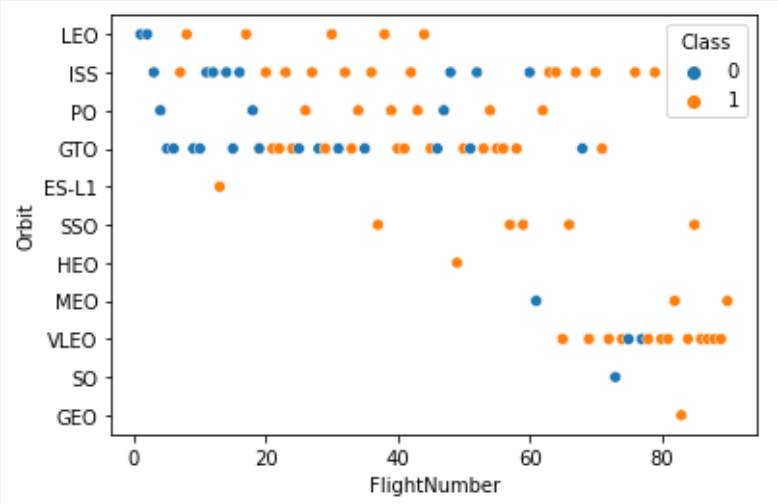
- Also, the success rate per orbit time is interesting to analyze.



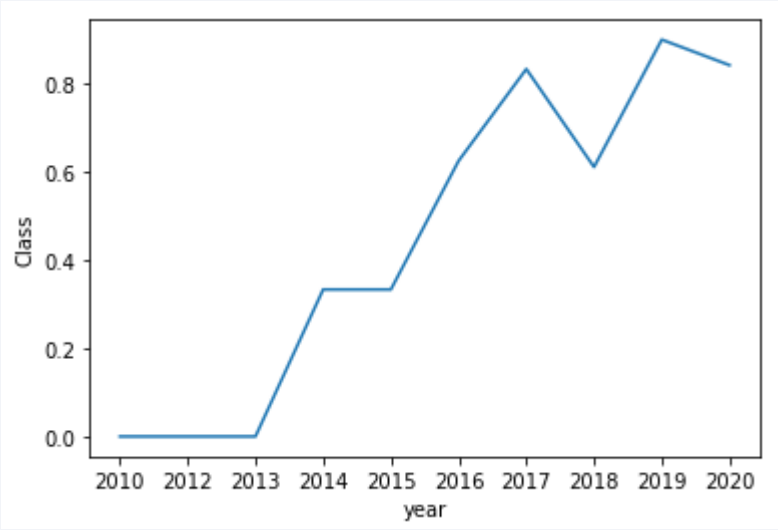
- https://github.com/dennisdemol/Coursera/blob/main/jupyter-labs-eda-dataviz.ipynb

EDA with Data Visualization

- Furthermore, the orbit type over time is analyzed.



- Finally, the success rate over time is plotted.



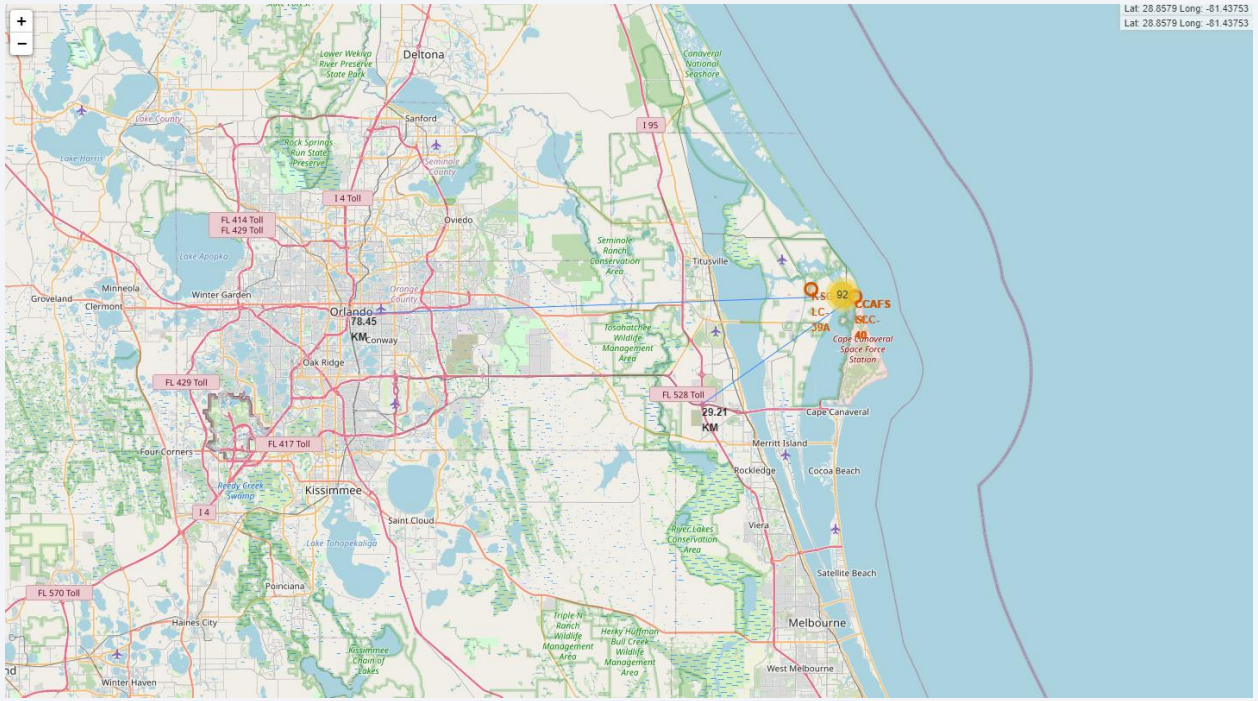
- https://github.com/dennisdemol/Coursera/blob/main/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- The names of the unique launch sites were displayed
- 5 launch sites starting with CCA were shown
- The total carried payload mass was computed
- The average payload mass of F9 v1.1 is computed
- The date of the first successful ground landing was shown
- The name of the drones with a mass between 4000 and 6000 which were successful
- Total number of mission fails and successes
- Name of boosters that carried the maximal mass
- Failed launches and the sites in 2015
- Rank the count of landing outcomes between 2 dates
- https://github.com/dennisdemol/Coursera/blob/main/jupyter-labs-exploratory_data_analysis.ipynb

Build an Interactive Map with Folium

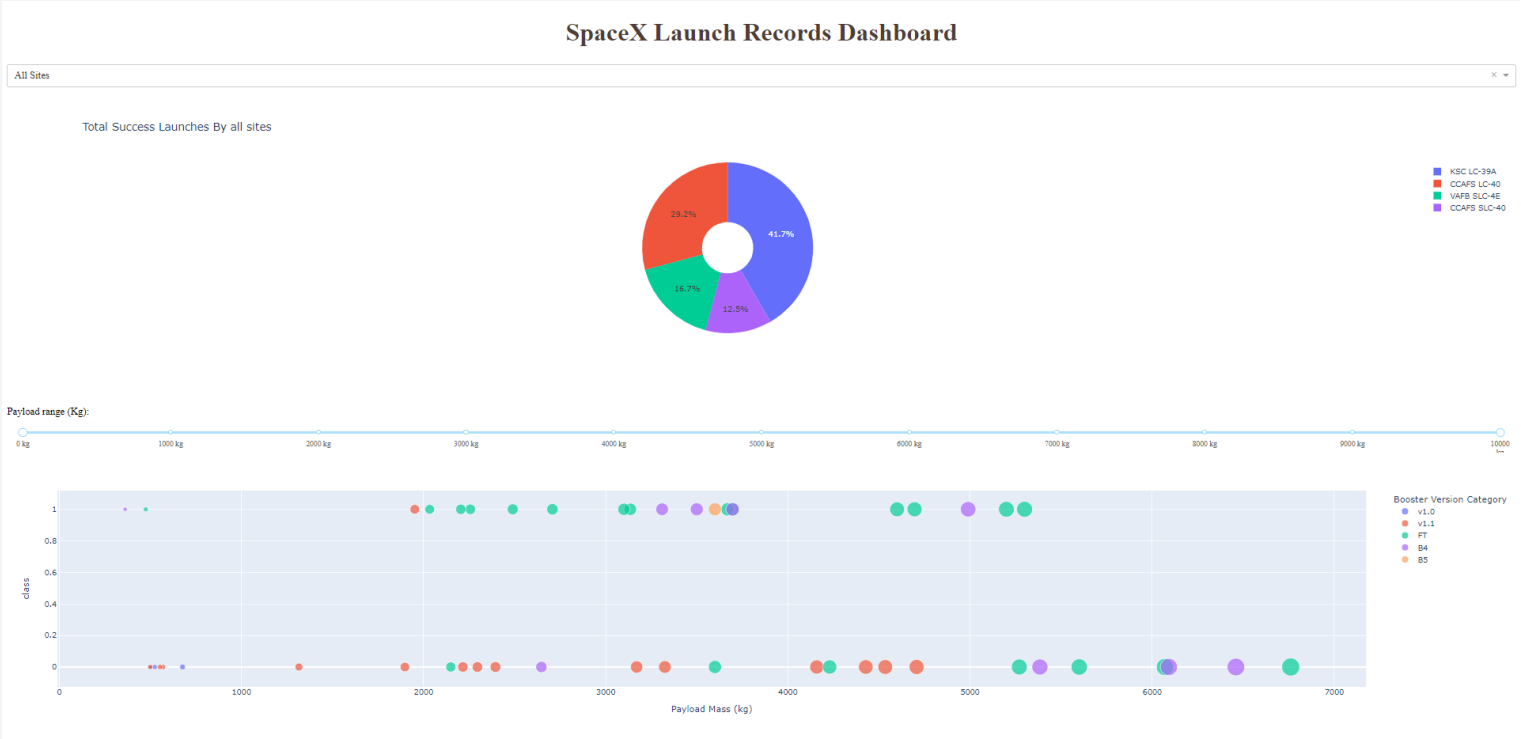
- On the folium map the launch sites were placed along with the number of successes and failures via the MarkerCluster. Furthermore, lines were drawn to the closest coastline point, highway intersection and city.



- https://github.com/dennisdemol/Coursera/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- From the dashboard the different launch sites can be selected. For each launch site the success percentage is presented along with a scatterplot of payload mass against success. The payload mass range of the scatterplot can be adjusted using a range slider.



- https://github.com/dennisdemol/Coursera/blob/main/spacex_dash.py

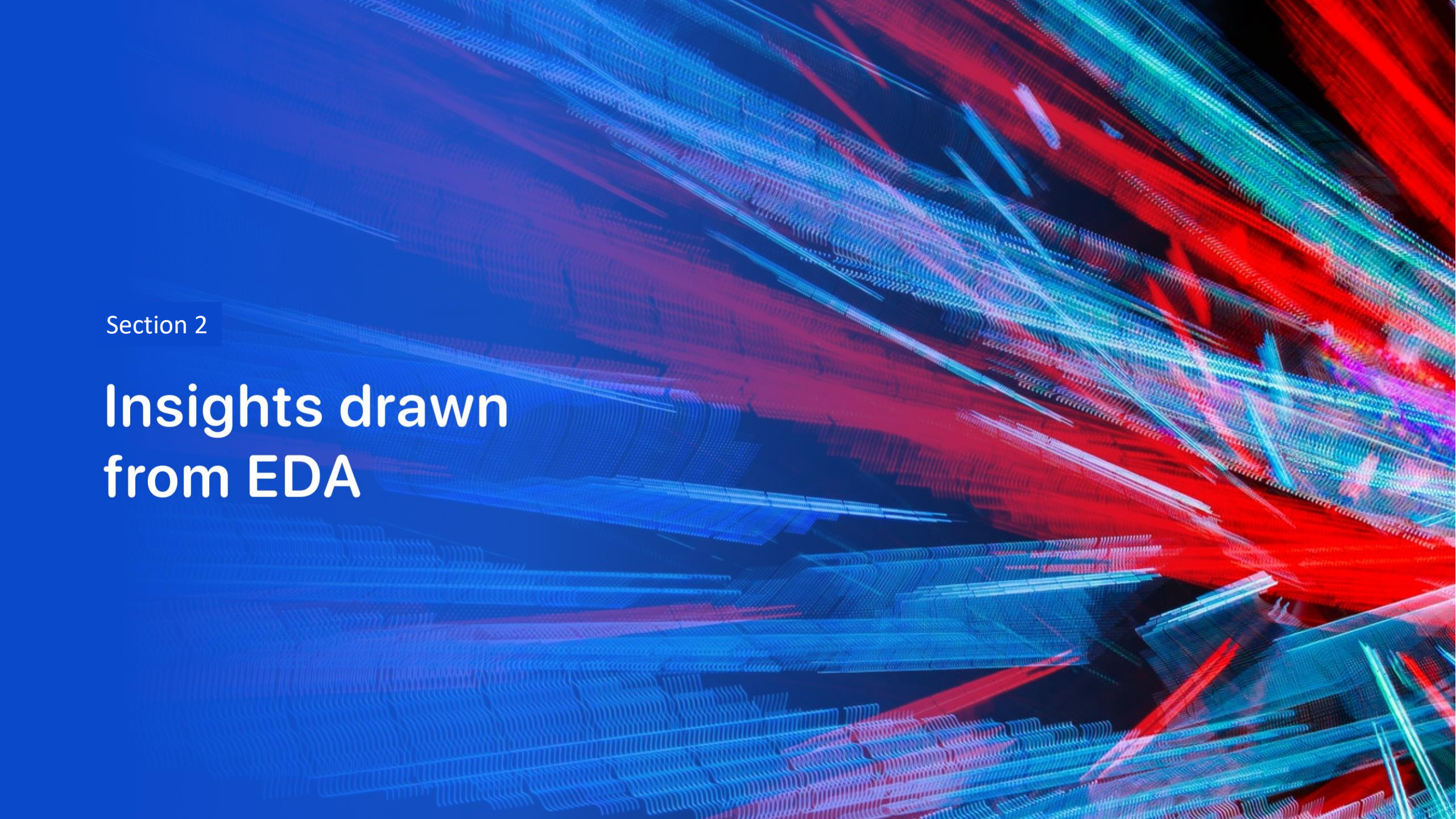
Predictive Analysis (Classification)

- To find the best performing classification model, different models were evaluated. These consisted of a logistic regression, support vector machine, decision tree classifier and K-nearest neighbours.
- The different models were trained on a training data set using cross validation for hyperparameter selection. When the best model was found based on the training data, the model was evaluated on the test set. The highest performing model is then selected as the final model.

- https://github.com/dennisdemol/Coursera/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

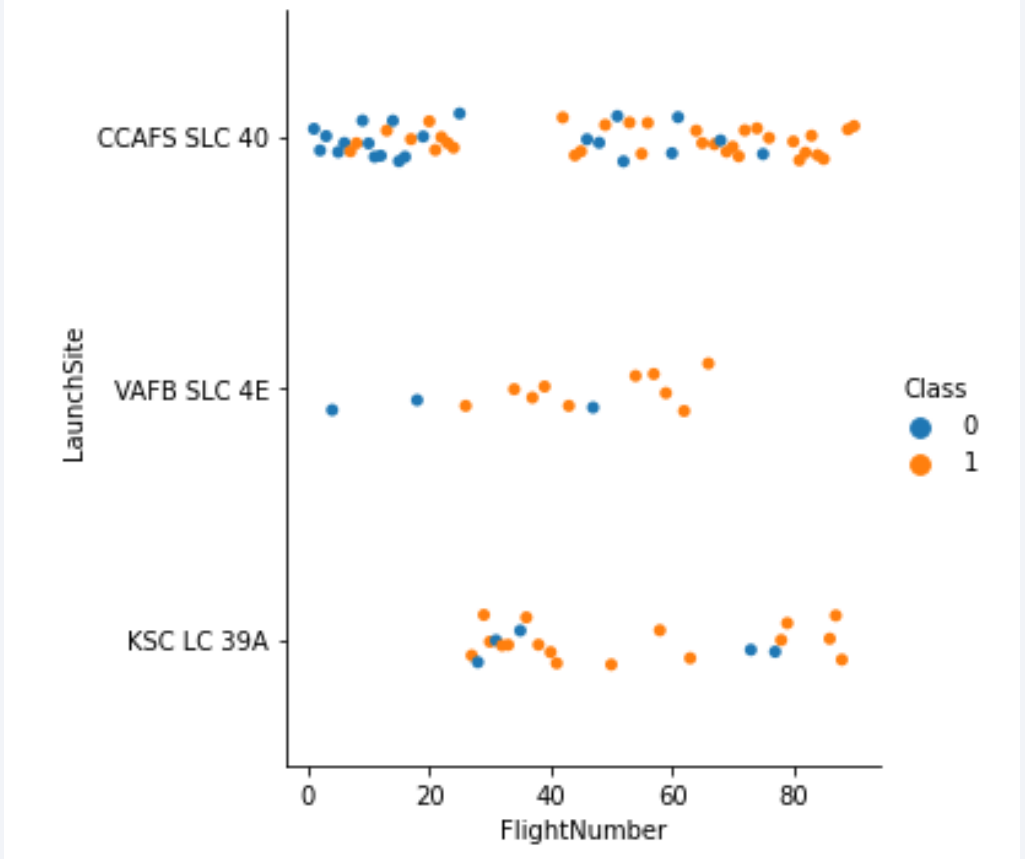
Results

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


Section 2
Insights drawn from EDA

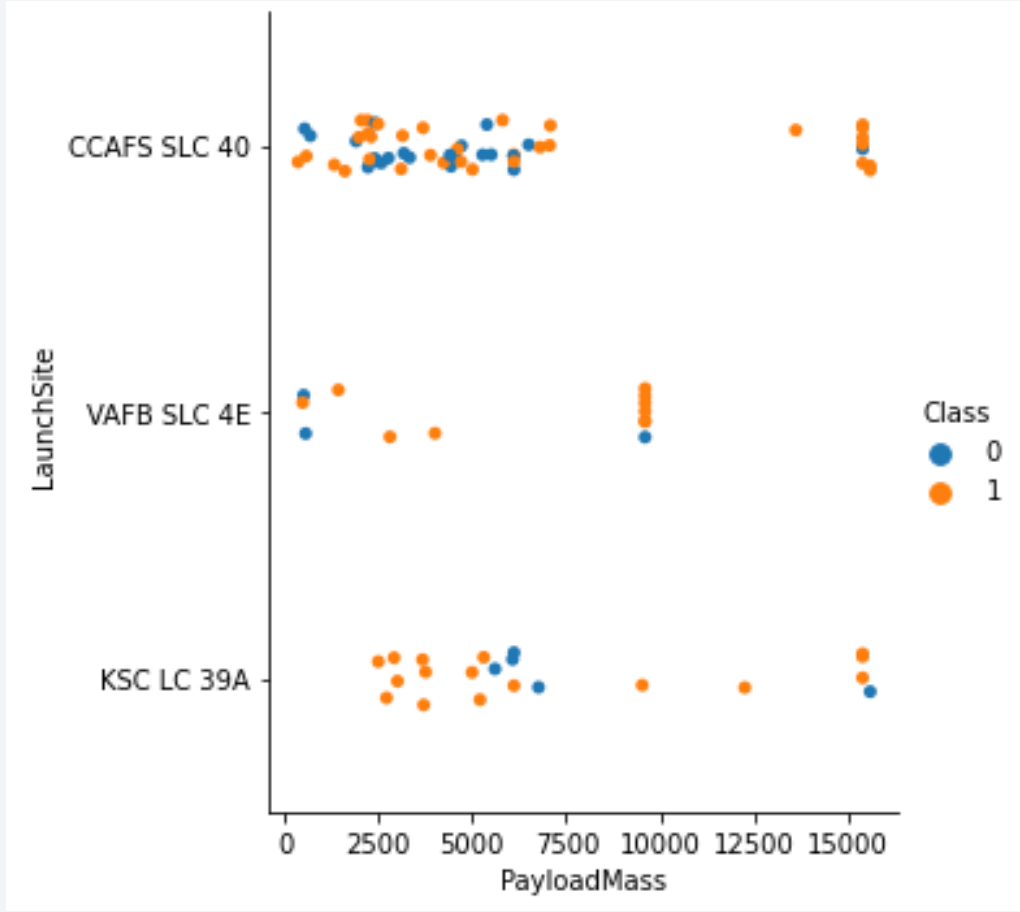
Flight Number vs. Launch Site

- Most launches were performed at ‘CCAFS SLC 40’
- ‘VAFB SLC 4E’ was out of use after flight 65
- ‘KSC LC 39A’ only in use since flight 25



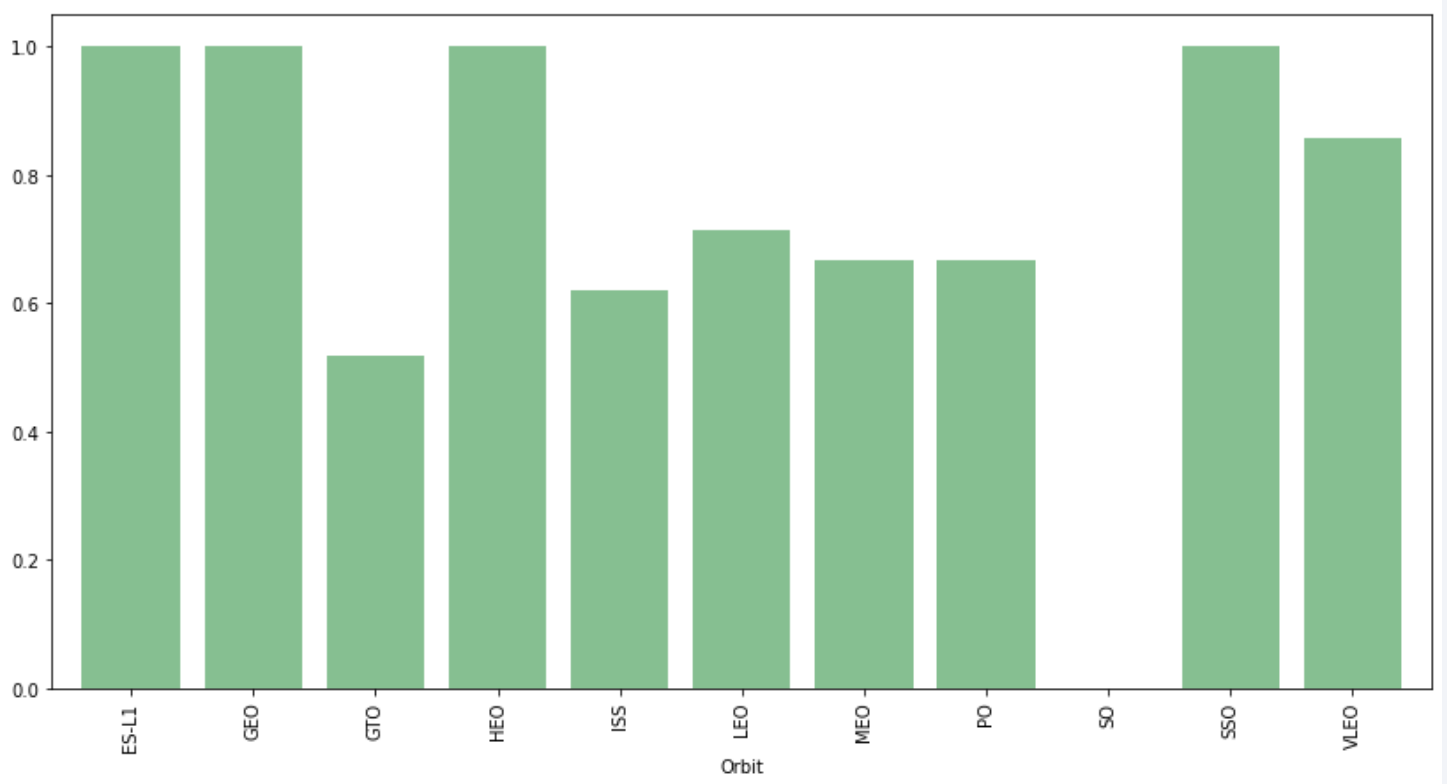
Payload vs. Launch Site

- Payload tests performed between mass of 0 and ~15000 kg.
- Many tests with payload mass of ~9000 kg at ‘VAFB SLC 4E’
- Many tests with payload mass of ~15000 kg at ‘CCAFS SLC 40’



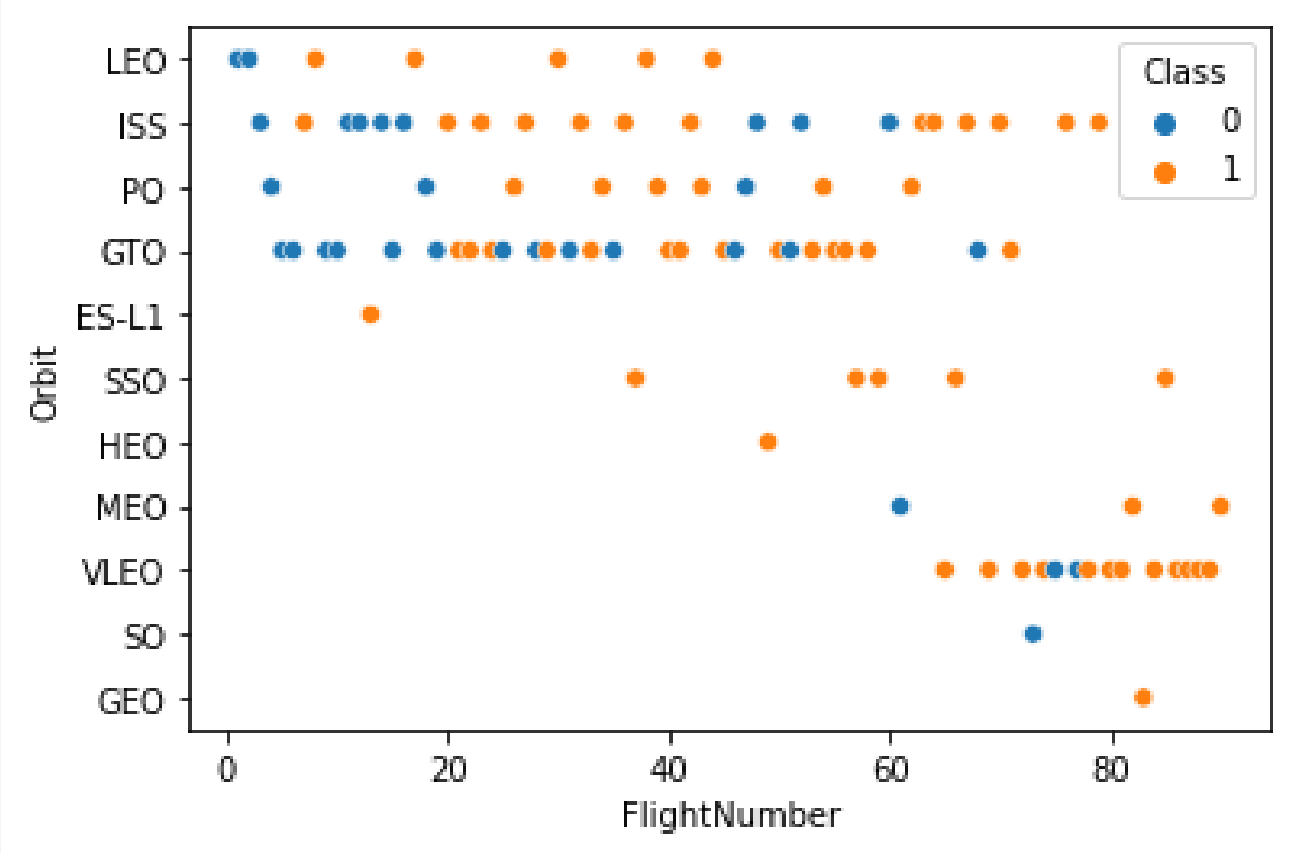
Success Rate vs. Orbit Type

- 0% success in SO
- 100% success in ES-L1, GEO, HEO and SSO



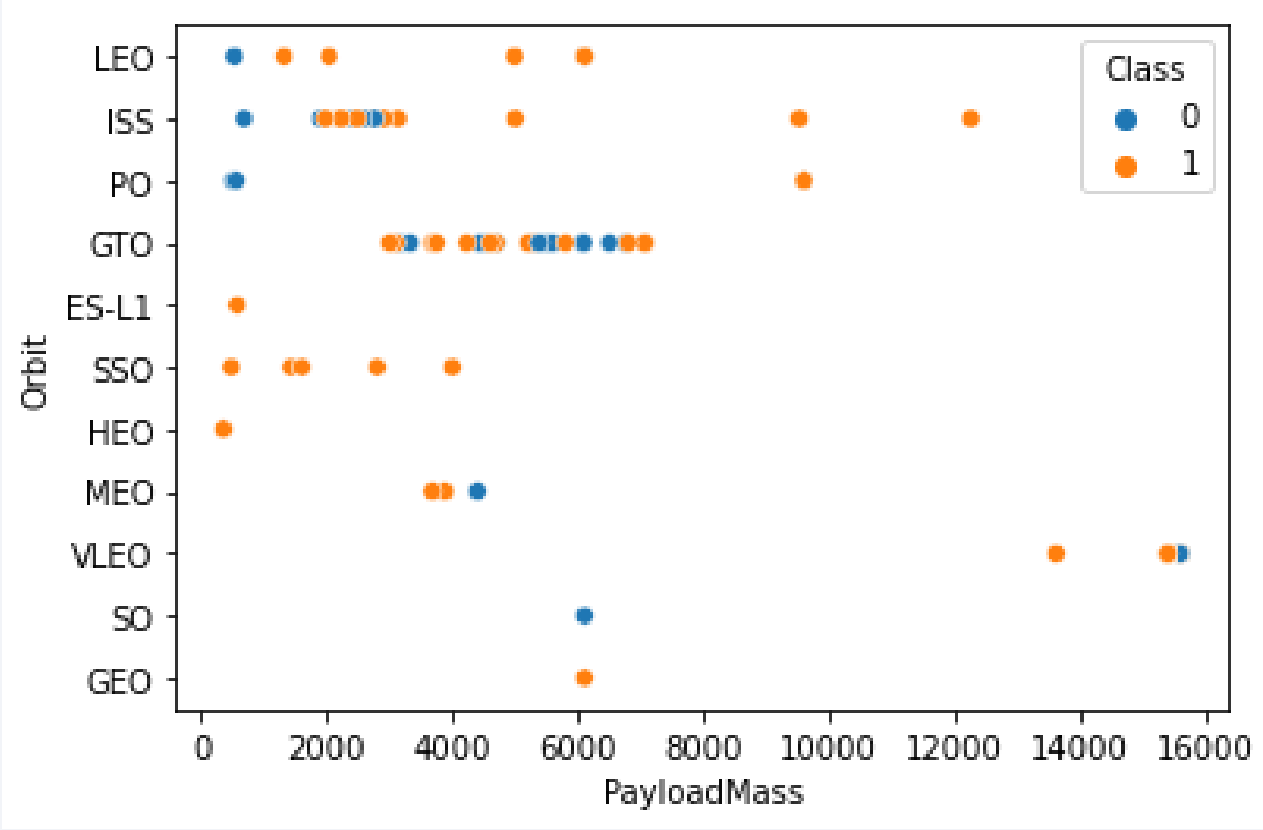
Flight Number vs. Orbit Type

- First flight in LEO
- Most flights in ISS, GTO and VLEO
- Only single flight in GEO



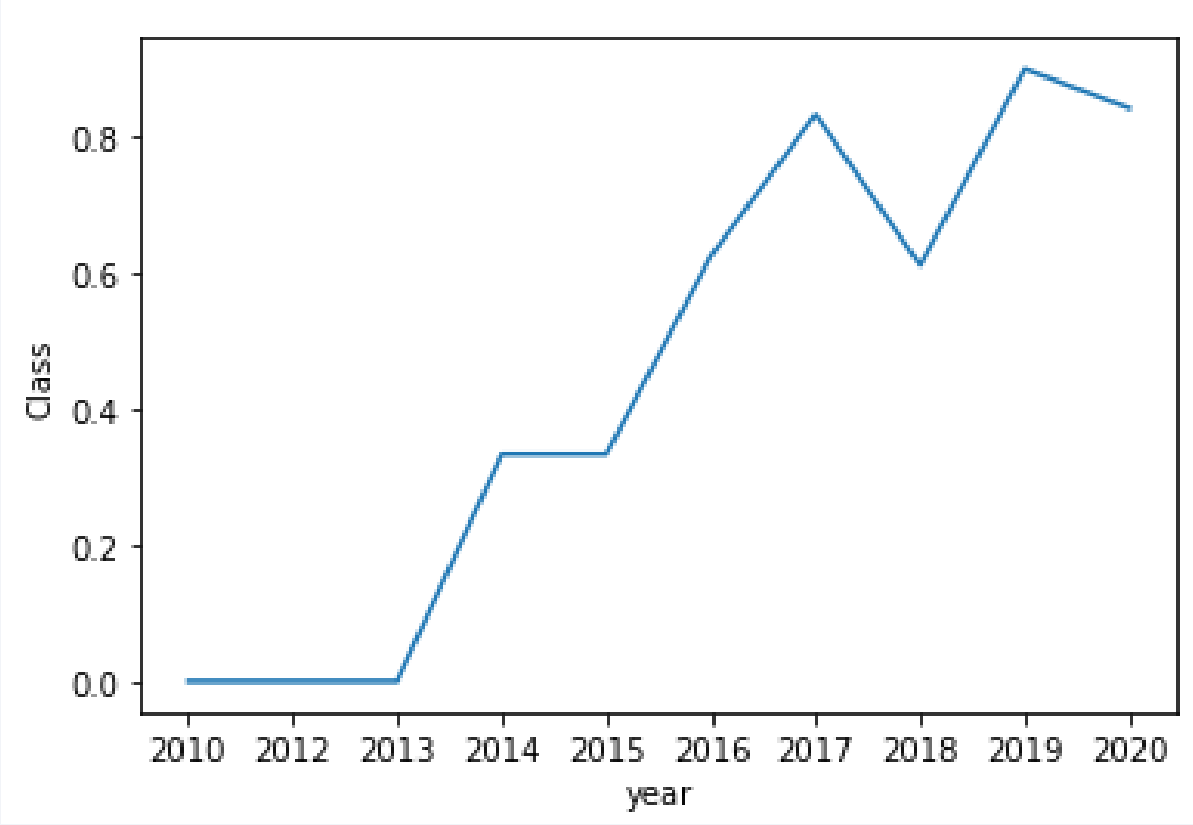
Payload vs. Orbit Type

- Highest payload mass for VLEO
- Many different payload masses in GTO



Launch Success Yearly Trend

- Success rate increased over time
- Success rate currently stable above 80%



All Launch Site Names

- Launch sites: CCAFS LC-40, CCAFS SLC-40, KSC LC-39A and VAFB SLC-4E

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;
```

- The query selects distinct launch sites names

Launch Site Names Begin with 'CCA'

```
%sql SELECT * FROM SpaceX WHERE LaunchSite LIKE 'CCA%' LIMIT 5
```

- The query selects full records where the LaunchSite name starts with ‘CCA’

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)
```

- The sum is calculated of all payload masses

- The result is 45596

Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

- The average value is calculated where the booster version is ‘F9 v1.1’
- The value is 2928

First Successful Ground Landing Date

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

- The query gives the lowest date where the ground landing was a success

- The result is 22-12-2015

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

- The query selects the booster version where the payload mass is between 4000 and 6000

- These versions are: F9 FT B1022, F9 FT B1026, F9 FT B1021,2 and F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

- Count the mission outcomes where it starts with ‘Success’
- The result is 100

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';
```

- Count the mission outcomes where it starts with ‘Failure’
- The result is 1

Boosters Carried Maximum Payload

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

- The query selects the booster versions where the payload was maximal.

- The versions are: 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

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

- The query selects the booster version and launch sites in 2015 where the landing is a failure

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

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

```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEX \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING__OUTCOME \
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

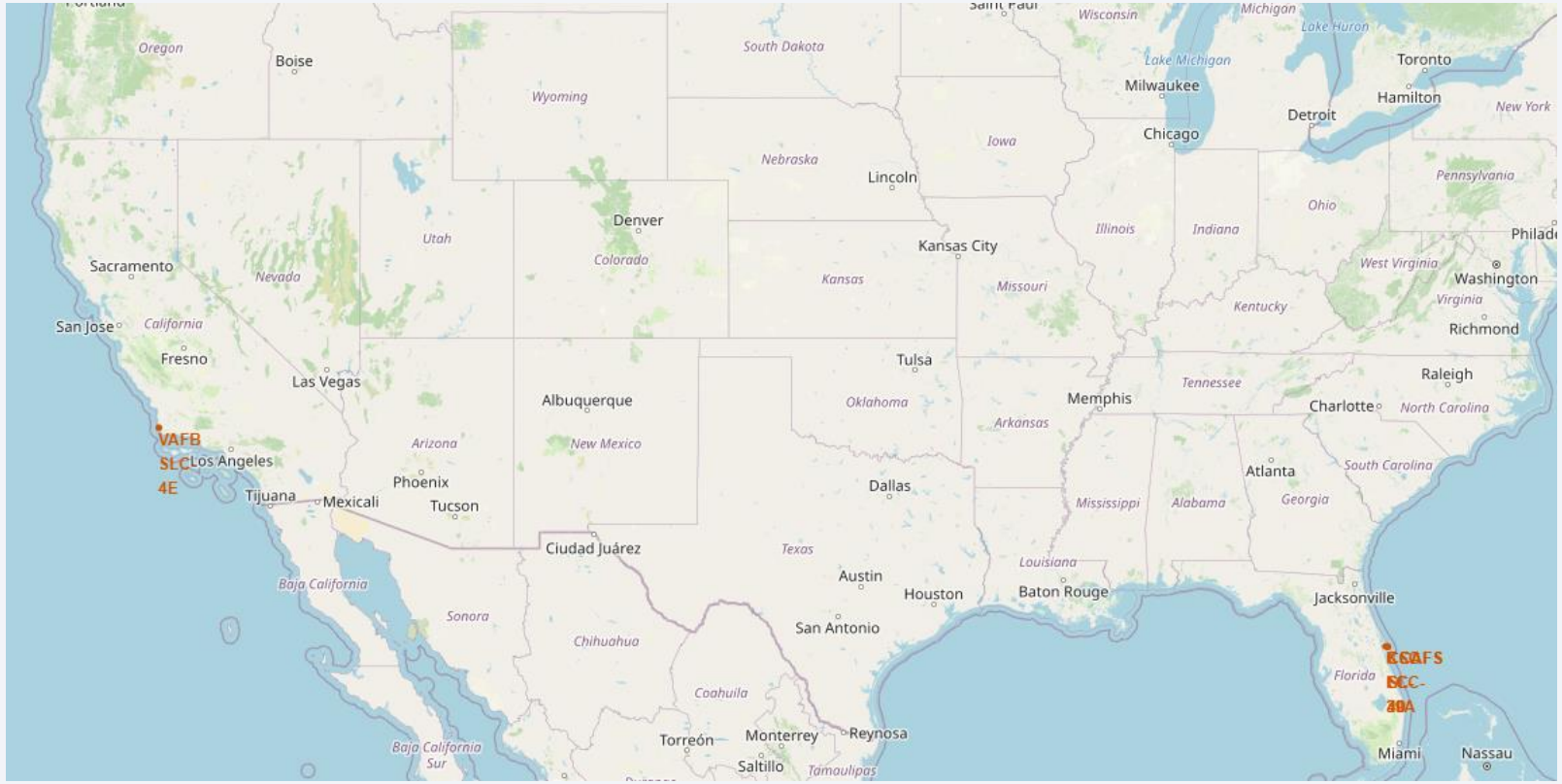
- The query selects all different landing outcomes along with the number they occurred. The occurrences are in descending order.

Landing Outcome	Total Count
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

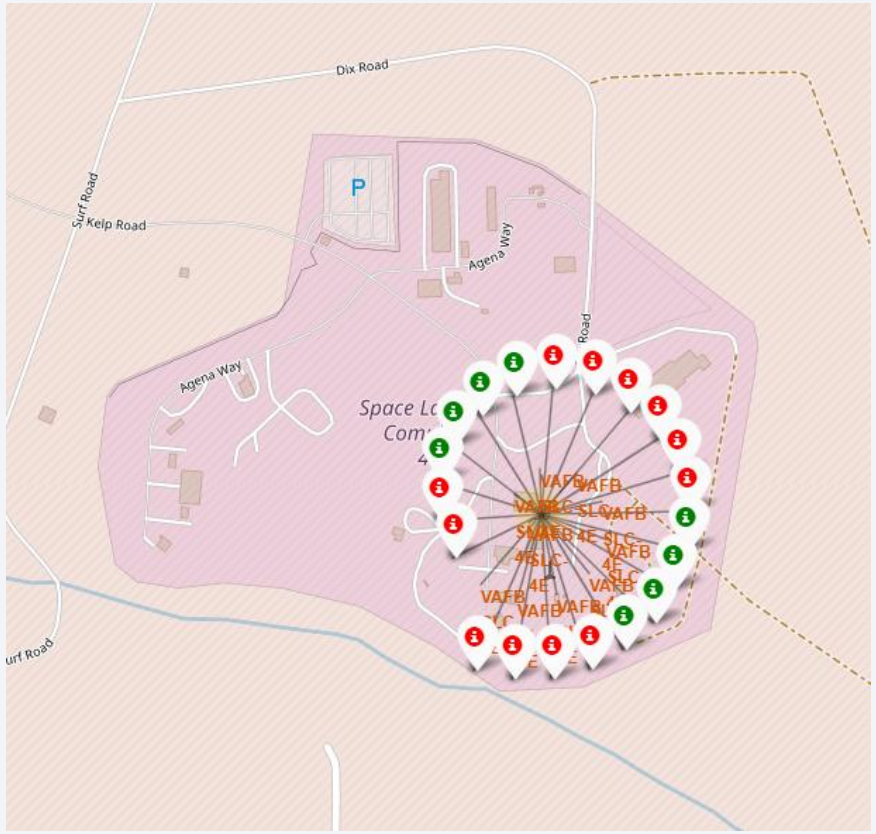
Launch Sites Proximities Analysis

Map of locations of takeoffs



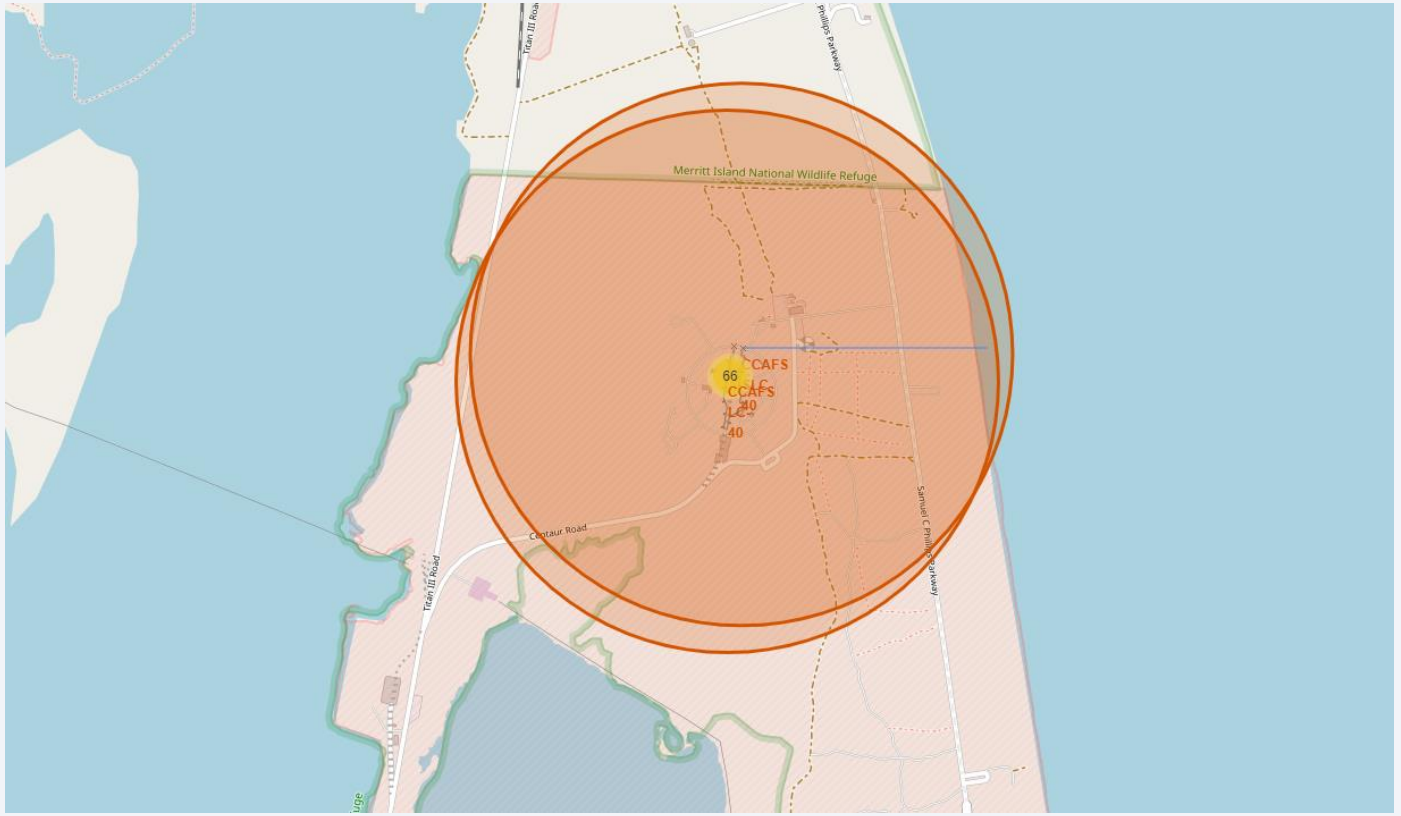
- The map shows the takeoff locations of the SpaceX rockets

Marker cluster of takeoff site

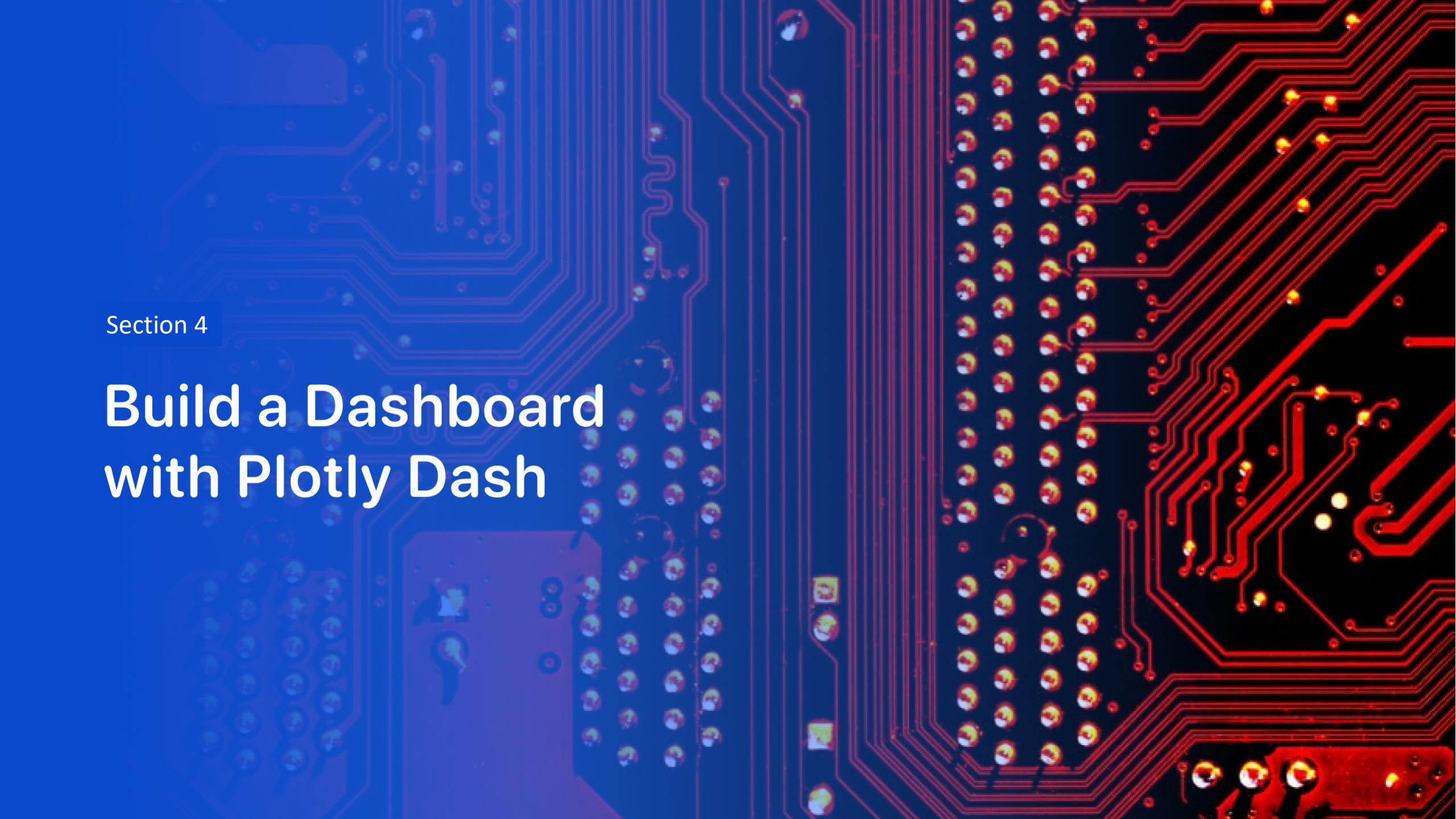


- Green labels mean successful, while red means failure

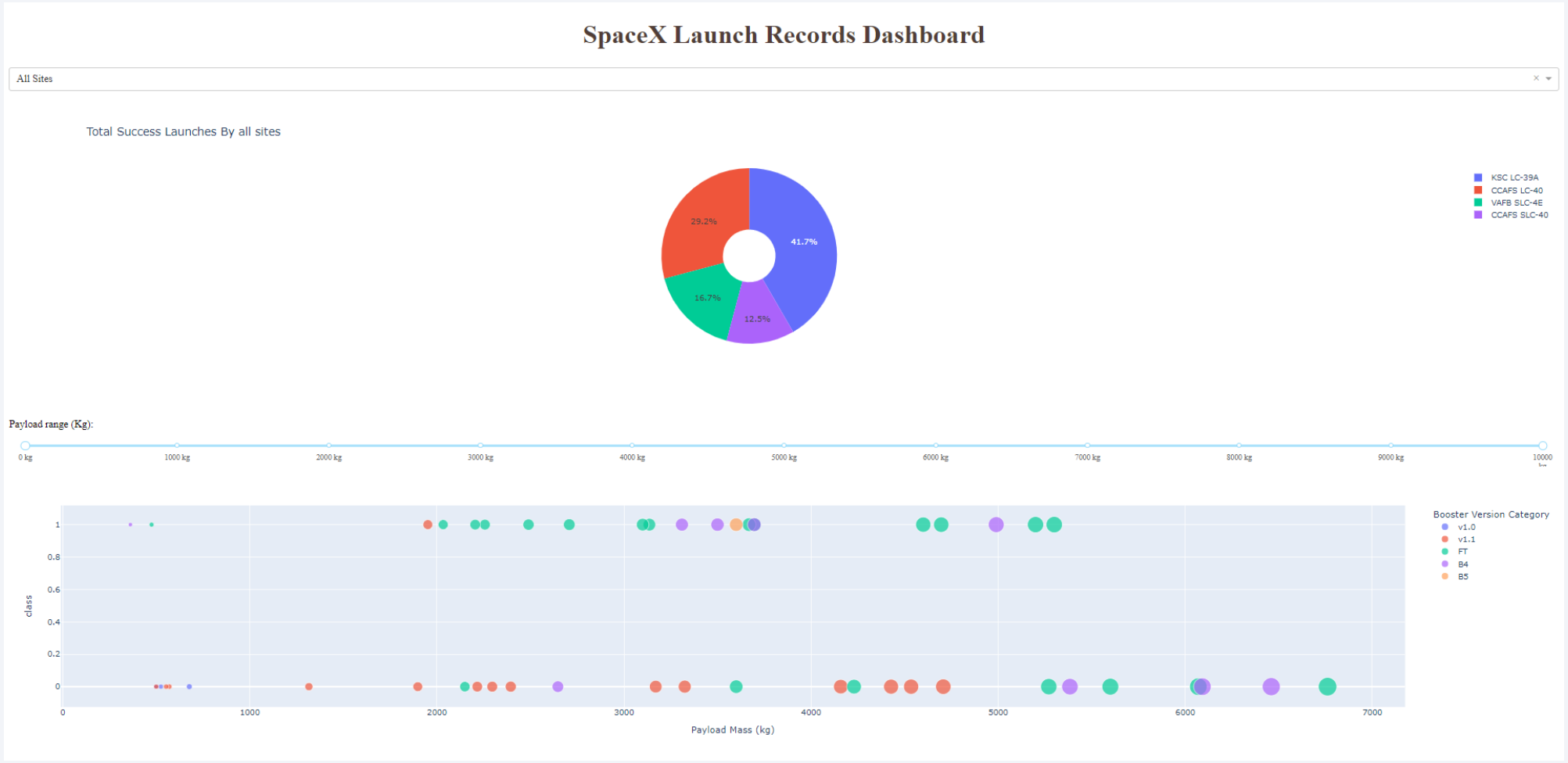
Distance to coastline



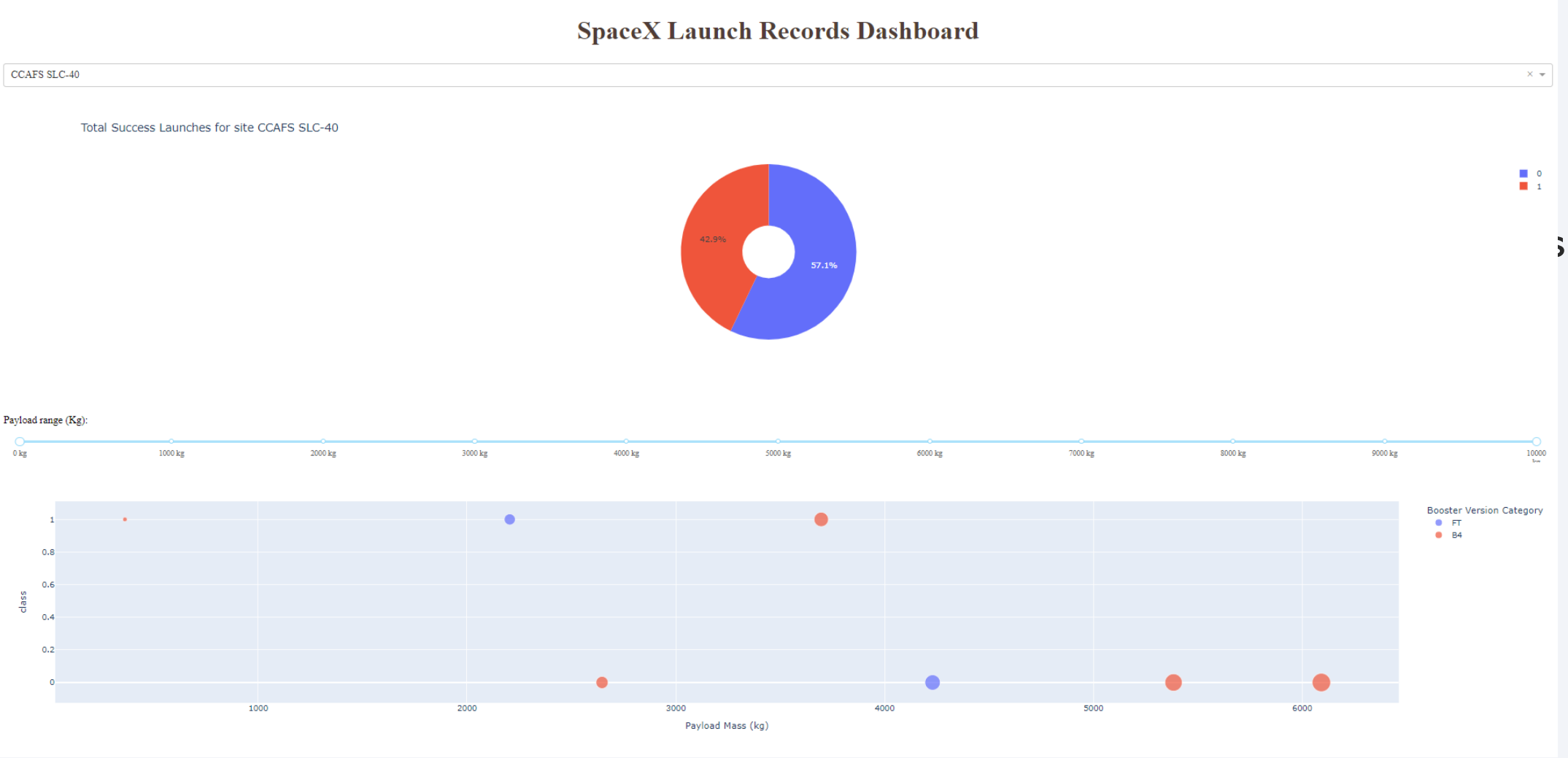
- The blue line indicates the line to the closest shoreline


Section 4
Build a Dashboard with Plotly Dash

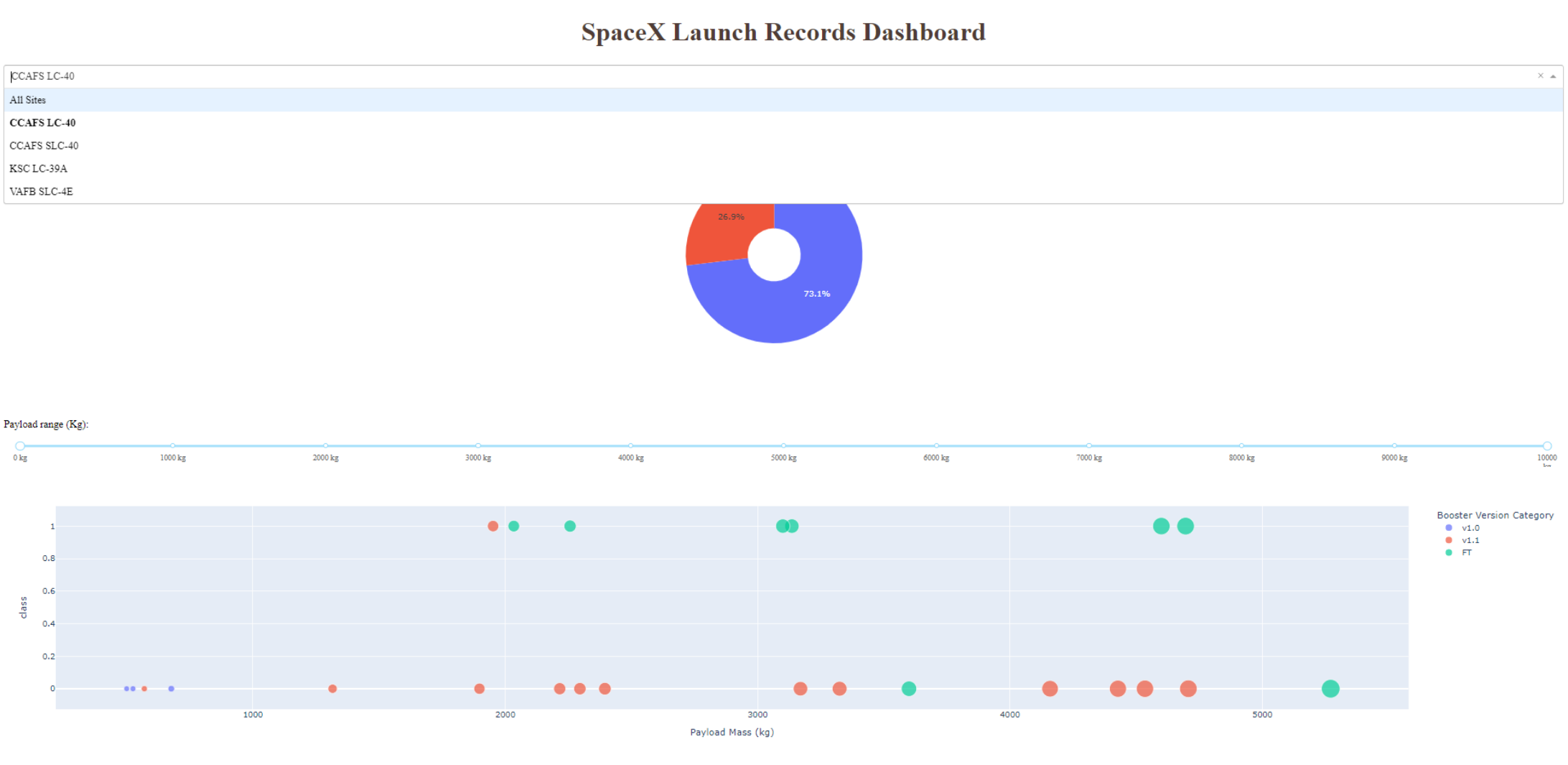
Dashboard showing all sites



Dashboard showing highest launch success site



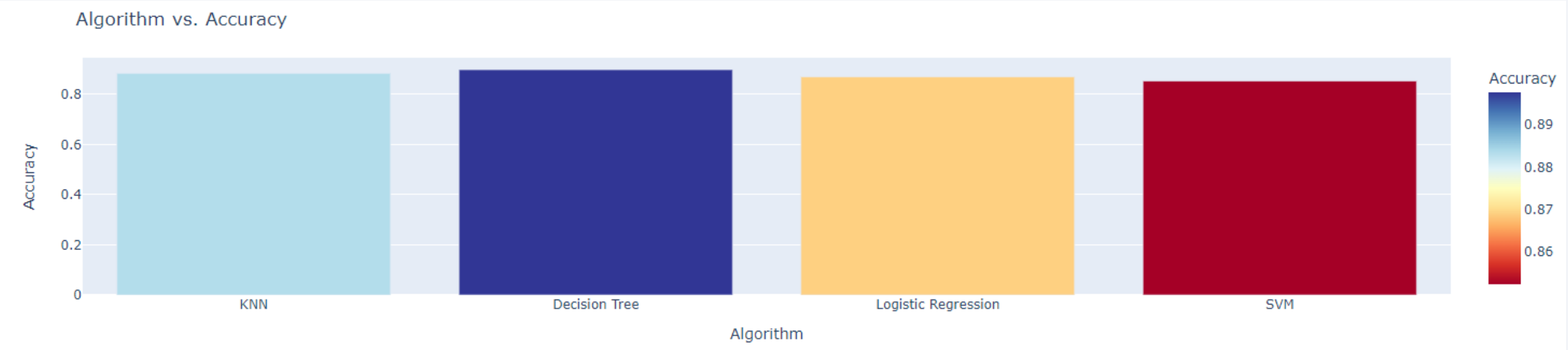
Dashboard for different site with all options shown



Section 5

Predictive Analysis (Classification)

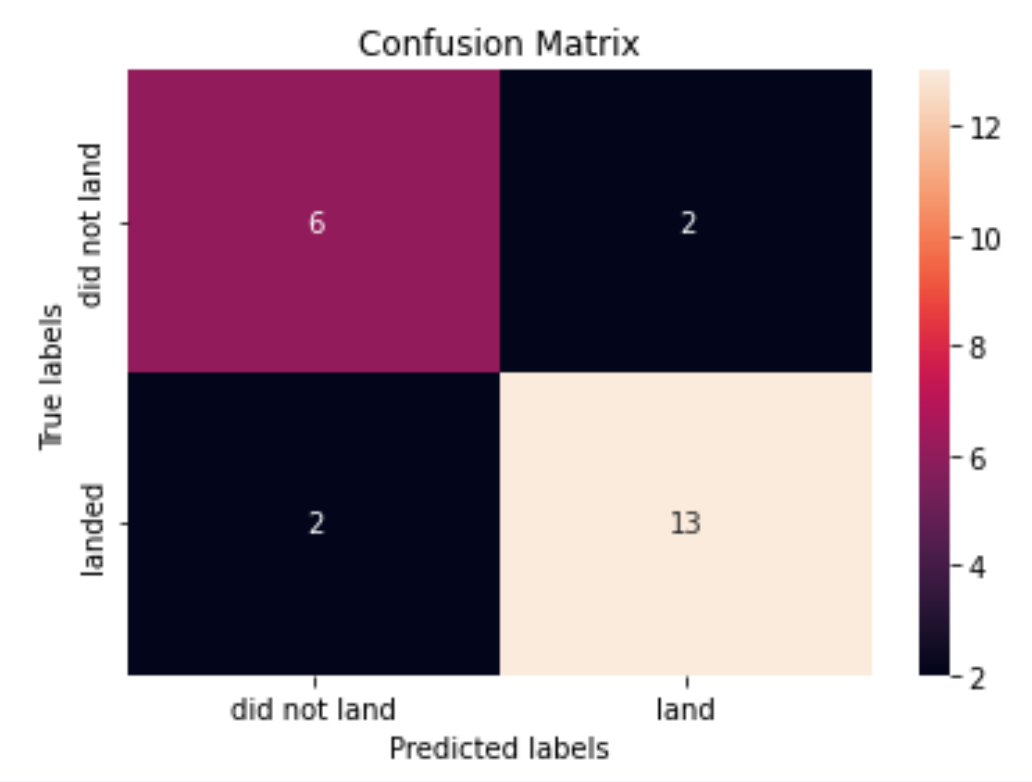
Classification Accuracy



- The decision tree algorithm is presenting the highest performance.

Confusion Matrix

- The confusion matrix shows 19 labels are correctly identified and 4 not.



Conclusions

- From the data we found that the success rate of the launches increased over time
- The decision tree model was best at correctly labeling the launches
- The takeoff locations were programmed on a Folium map for visual inspection

Thank you!