Winning Space Race with Data Science

Pedro Salvado
17th Octobre 2022

Outline

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

Executive Summary

- **Summary of methodologies**
  - **Data Collection through API**
  - **Web Scraping for data collection**
  - **Data Wrangling**
  - **Exploratory Data Analysis (EDA)**:
    - With SQL
    - Data Visualization
    - Visual Analysis with Folium
    - Visual Analysis with Dash Plotly
  - **Machine Learning:**
    - Classification
    - Cross Validation

.

- **Summary of all results**
  - EDA results
  - Predictive Classification and comparison between algorithms

Introduction

- **Project background and context**
  - 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. The goal of the project is to predict if the first stage will land successfully.
- **Problems you want to find answers**
  - What factors influence the landing success of the first stage;
  - If there are any set of conditions that influence the landing of the first stage

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected through SPACEX API and web scraping from Wikipedia
- Perform data wrangling
  - Correct missing data and OneHot Encoding and StandardScaler
- 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

- **Methodologies:**
  - Data collection was done using the get request method to the SPACEX API.
  - Then decoded the response through json() function and then created a Pandas dataframe through json_normalize() method;
  - After a first glance at the data there was a need to correct some missing data, ‘Payloadmass’, using the mean of overall values and fill the missing values with that value.
  - Web Scraping to get the Falcon9 records.

Data Collection – SpaceX API

- Get request method to the API.
- Create Dataframe

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

```
response = requests.get(spacex_url)
```

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

```
static_json_url='https://cf-courses-data.s3.us.cloud-objec
response = requests.get(static_json_url)
```

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

Data Collection - Scraping

- Web scrap to wikipedia to get the Falcon 9 launch records. BeautifulSoup helped with identifying table – table headers and table cells

Link: https://github.com/pmgsalvado/Coursera---SpaceX-/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)
```

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content)
```

```
# 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')
```

```
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
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a li
rows = first_launch_table.find_all('th')
for row in rows:
  name = extract_column_from_header(row)
  if name != None and len(name) > 0:
    column_names.append(name)
```

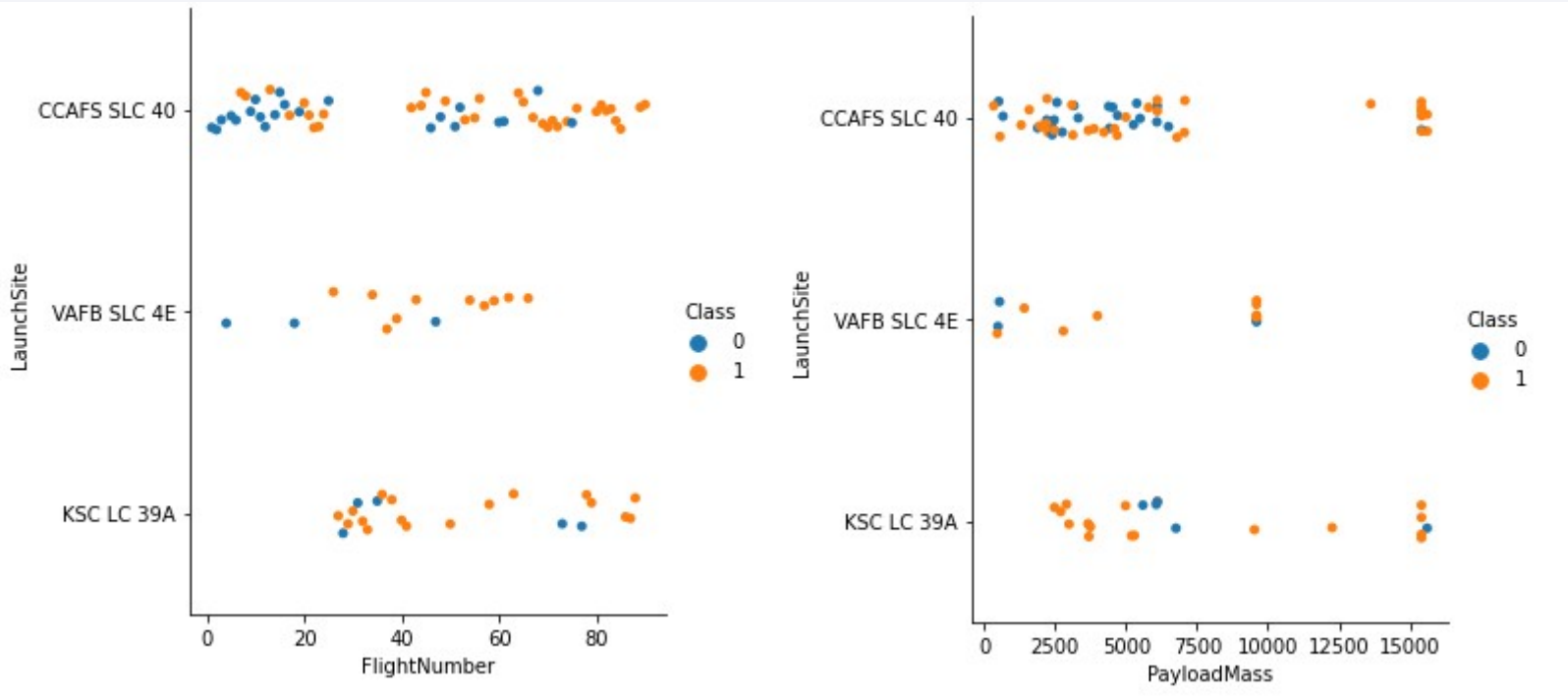
Data Wrangling

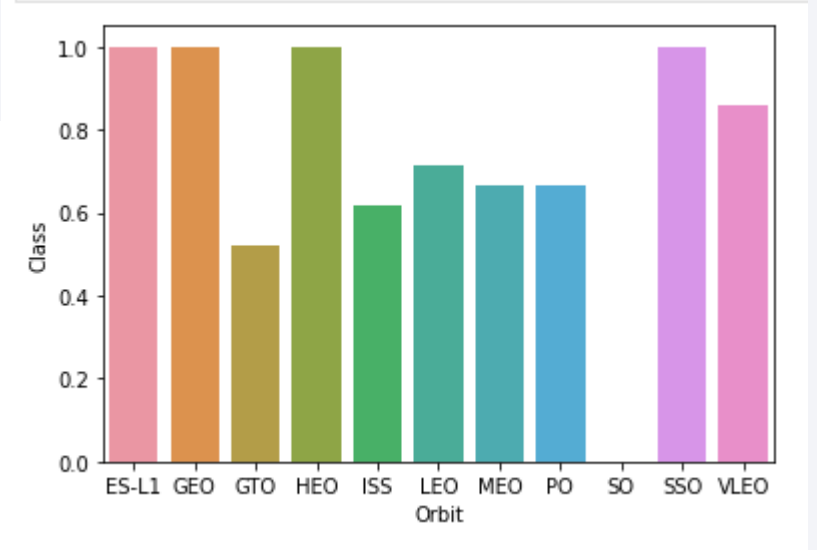
The first step was to perform Exploratory Data Analysis (EDA) to correct missing values and identify the attributes.
We identified the number of launches by site, what were the orbits and the number of launches by orbit.
We then created a new column, Class, with success and unsuccess landing.

Link:
https://github.com/pmgsalvado/Coursera---SpaceX-/blob/main/Data%20Wrangling%20EDA.ipynb

EDA with Data Visualization

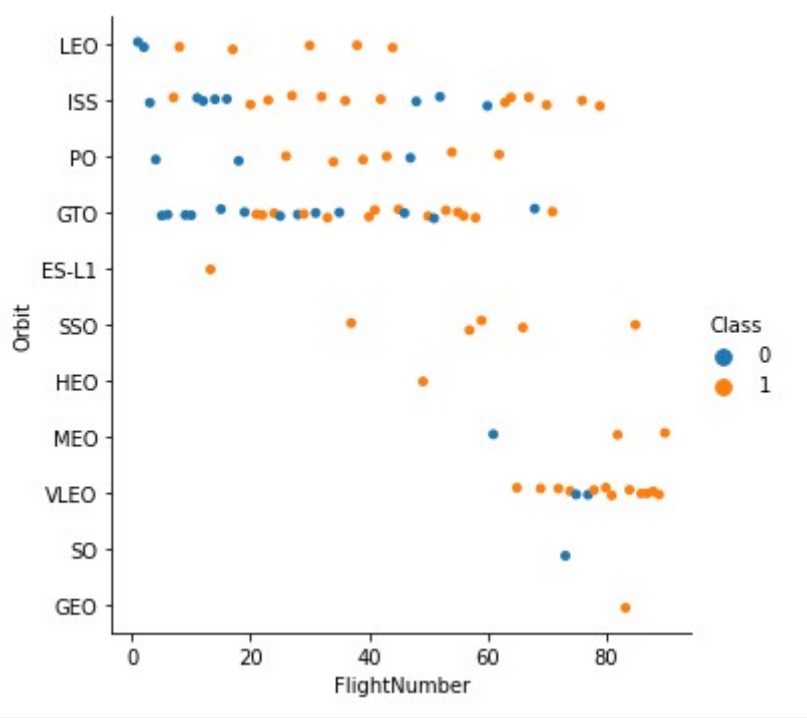
We visualized the relationship between FlightNumber and LaunchSite, between the Payload and LaunchSite, SuccessRate and Orbit and PayLoad and OrbitType. (each with distinction Success/ unsucess)

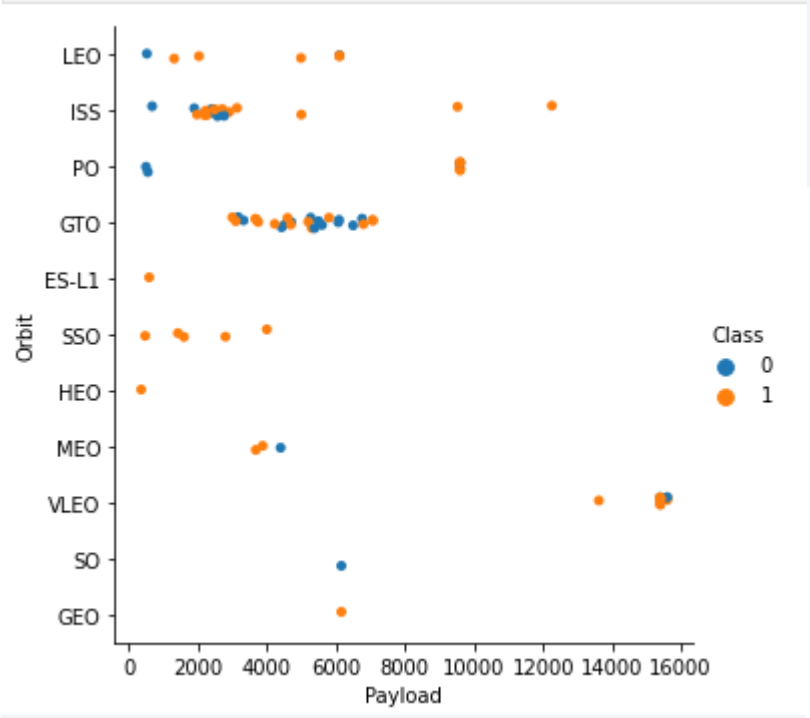


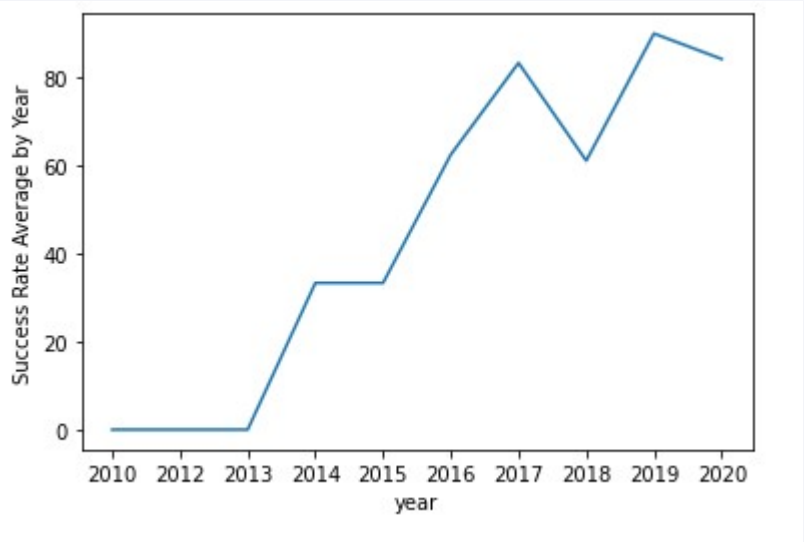


EDA with Data Visualization

We visualized the relationship between FlightNumber and Otbit, between the Payload and Orbit and the evolution of the success over the years.







EDA with SQL

- The dataset was loaded to Sqlite
- After we applied EDA with SQL:
  - Find the unique names of the launch site
  - Find the overall max Pay Load Mass
  - The average Payload mass of Falcon 9 v1.1
  - The date of the first successful landing in a ground pad
  - The total successful and non successful outcomes

  Link:

  https://github.com/pmgsalvado/Coursera---SpaceX-/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- All the Launch Sites were marked in the map with circles and markers and their designation.
- To each Launch Site we added each Launch and it’s landing outcome, as markers clusters (color different by outcome)
- And distance from Launch Site to POI’s was calculated:
  - Coastal Line
  - City Centers
  - Etc.

  Link:

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

Build a Dashboard with Plotly Dash

- An interactive dashboard was created using Plotly Dash
- Pie Charts were created to show the landing success of each launch site, and after was added the Payload Parameter to show how the success rate changed.

  Link:

  https://github.com/pmgsalvado/Coursera---SpaceX-/blob/main/lab_dash.py

Predictive Analysis (Classification)

- The data was loaded with Pandas.
  - X – was already onehot encoded
  - Y – class (Success(‘1’)/No Success(‘0’))

- X matrix was Standardized
- After the data we split the data between train and test.
- In order to choose from different algorithms and parameters we used GridSearchCV to find the best model
- We found the best performing classification model.

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

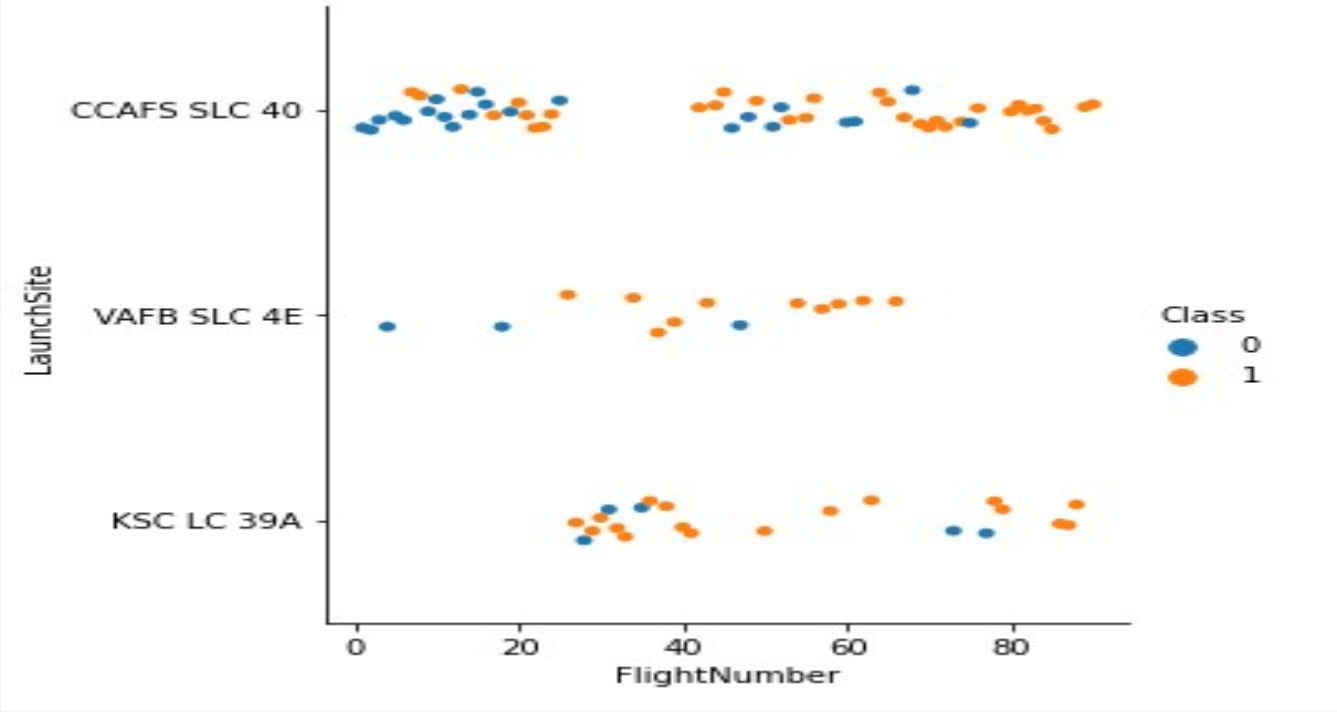
Results

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

Section 2

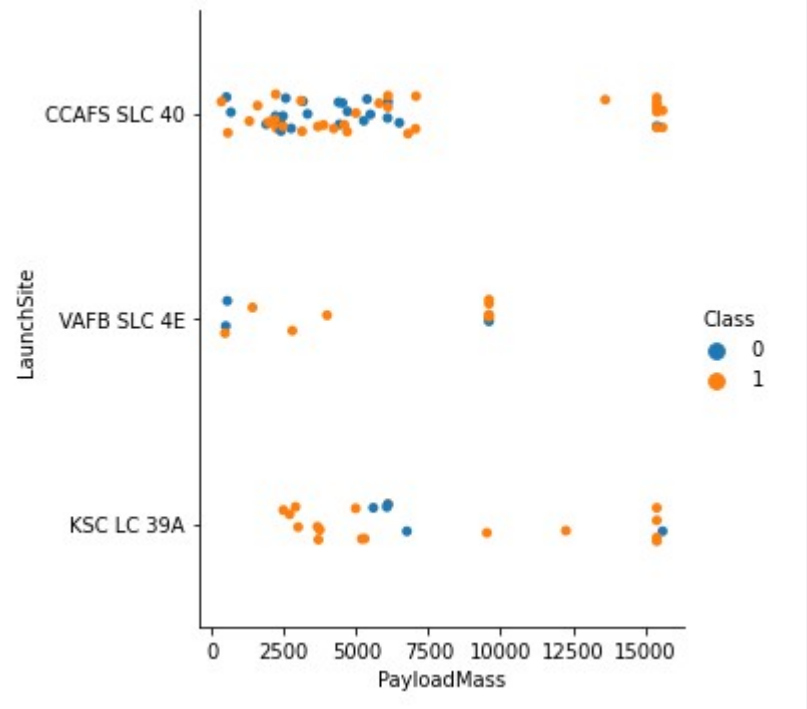
Insights drawn from EDA

Flight Number vs. Launch Site



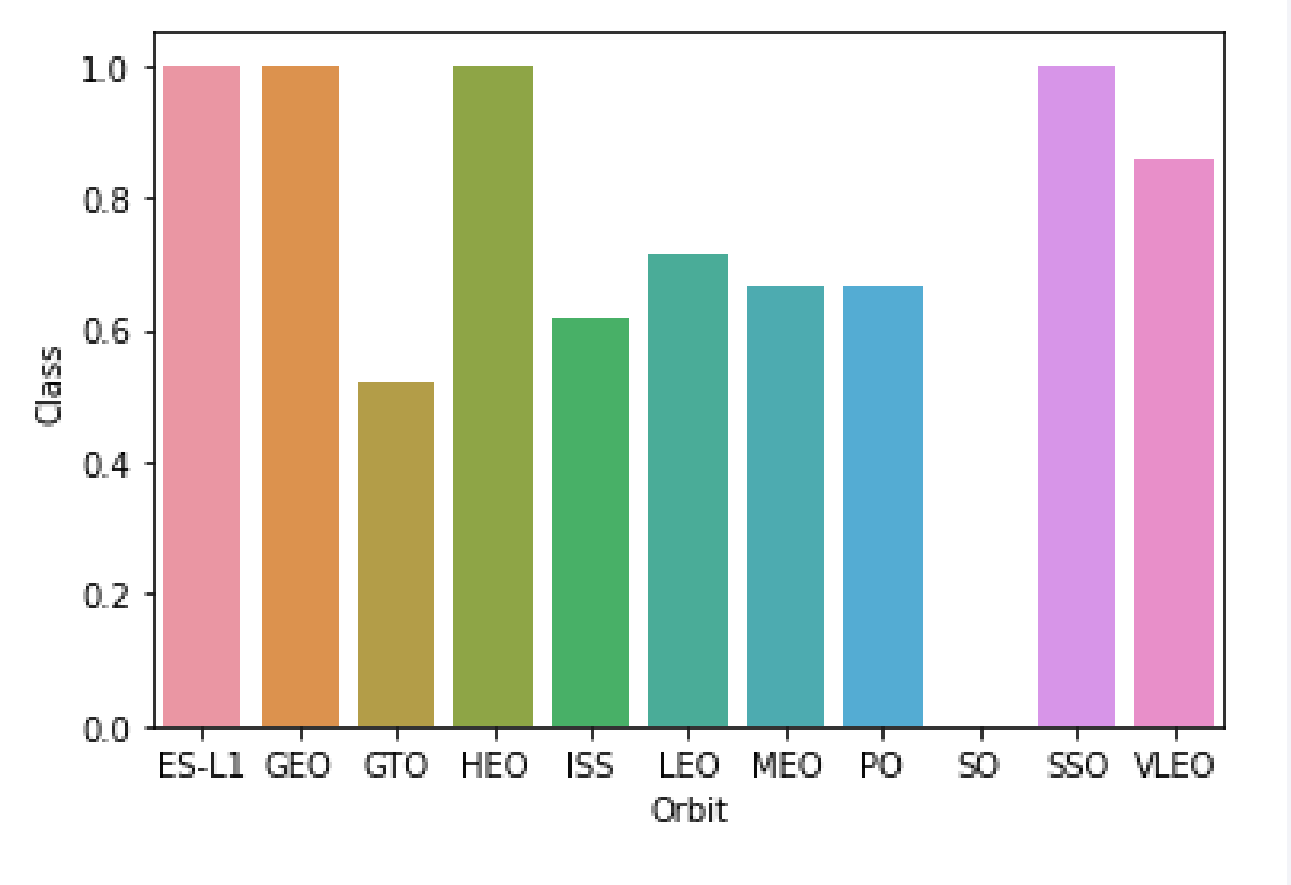
Clearly the FlightNumber, has some influence on the overall success of the mission

Payload vs. Launch Site



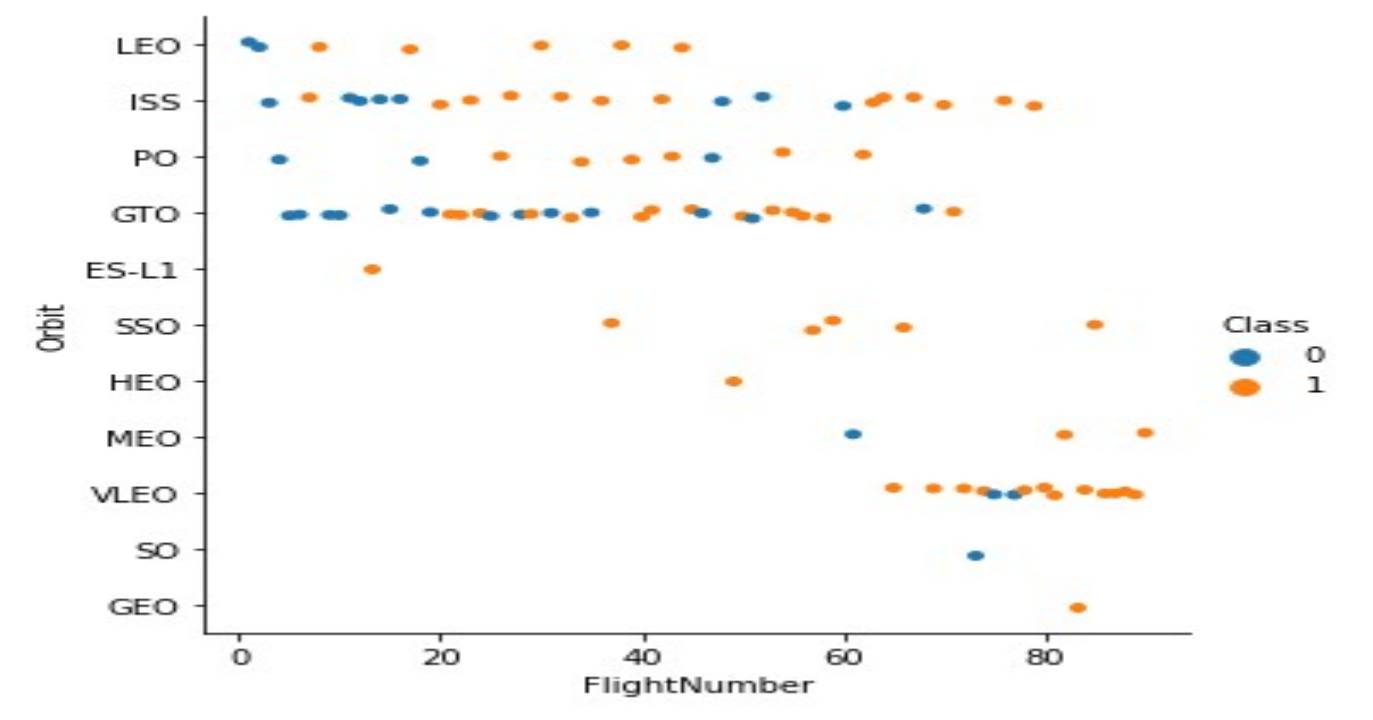
- We can see that for Payload mass above 10000 VAFB SLC 4E site there are no launches.

Success Rate vs. Orbit Type



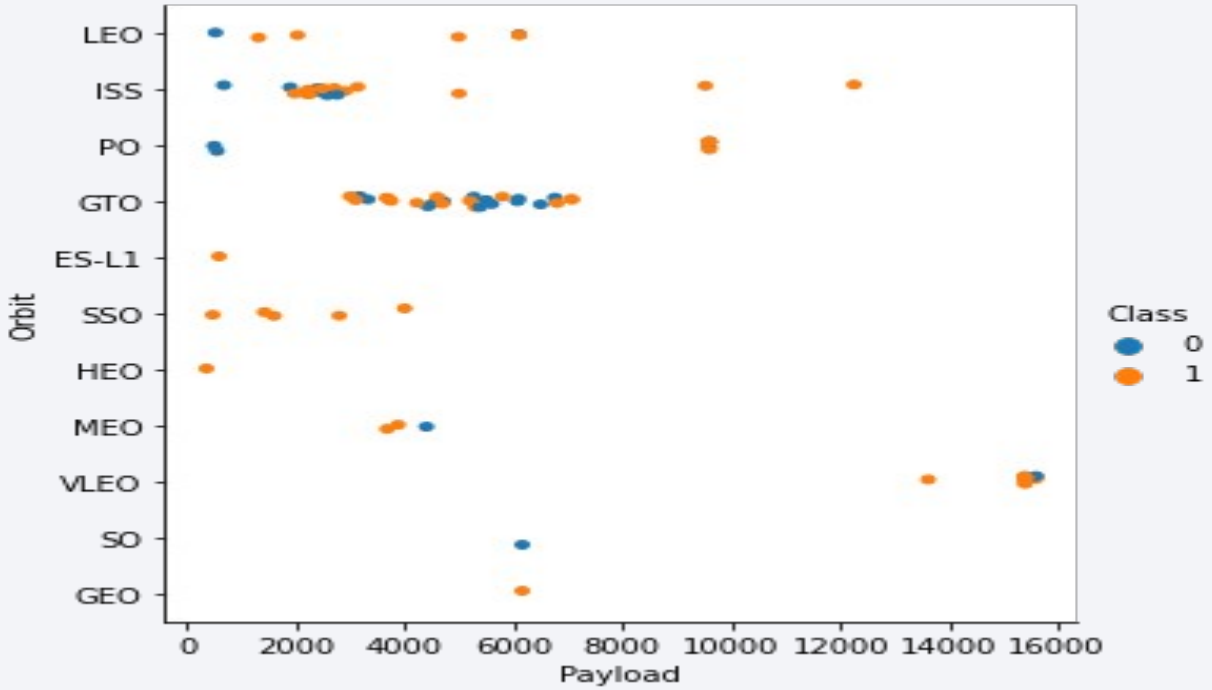
- ES-L1, GEO, HEO, SSO have higher success rate

Flight Number vs. Orbit Type



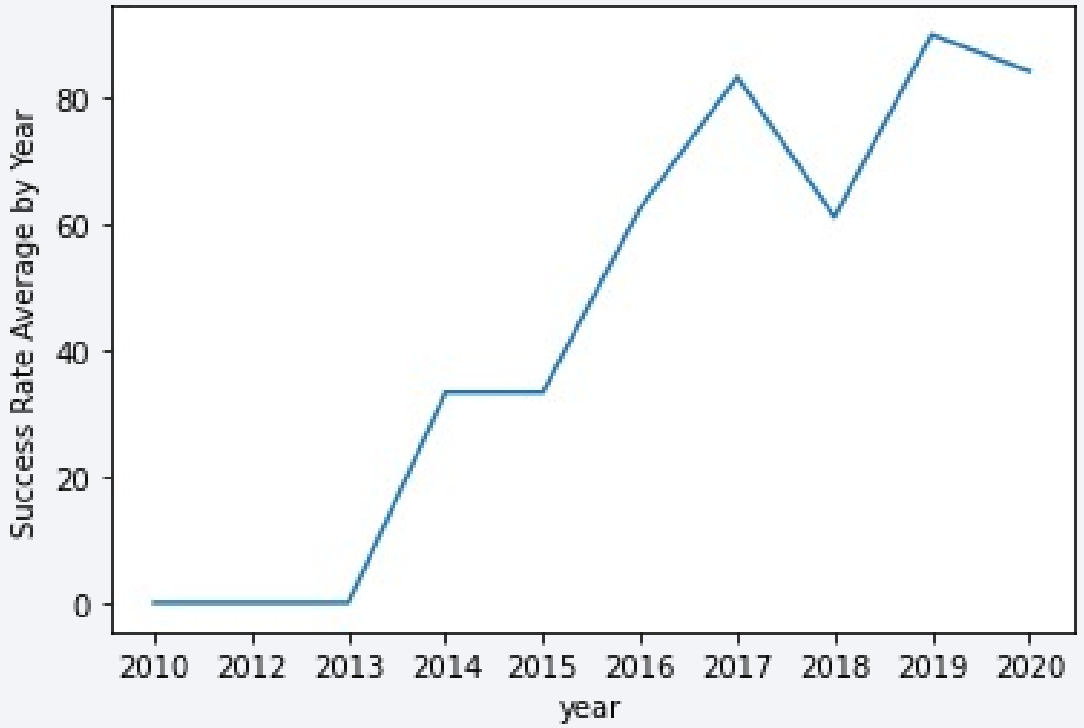
- For the LEO orbit, we can see there is a clear relationship between the Flight Number and the Success Landing

Payload vs. Orbit Type



- For higher Payload, PO, LEO, ISS have higher success

Launch Success Yearly Trend



- The success rate increase over the time. Maybe experience and learning from errors had some part in the overall success

All Launch Site Names

```
%sql SELECT DISTINCT(Launch_Site) FROM SPACEXTBL
```

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

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

Launch Site Names Begin with 'CCA'

```
%sql SELECT Launch_Site FROM SPACEXTBL WHERE Launch_site LIKE 'CCA%' LIMIT 5
```

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL
```

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

SUM(PAYLOAD_MASS__KG_)
619967

Average Payload Mass by F9 v1.1

```
%sql SELECT Booster_Version, AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version like 'F9 v1.1'
```

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

Booster_Version	AVG(PAYLOAD_MASS__KG_)
F9 v1.1	2928.4

First Successful Ground Landing Date

```
%sql SELECT MIN(Date) AS DATE FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Succ%'
```

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

DATE
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE "Landing _Outcome" = 'Success (drone ship)'
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) FROM SPACEXTBL GROUP BY Mission_Outcome
```

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

Mission_Outcome	COUNT(Mission_Outcome)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

```
%sql SELECT SUBSTR(Date,4,2) AS "MONTH" , "Landing _Outcome"  AS "OUTCOME", Booster_Version, Launch_Site FROM SPACEXTBL
 WHERE SUBSTR(Date,7,5) = '2015' AND "Landing _Outcome" like 'Failure%'
```

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

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

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

```
%sql SELECT Date FROM SPACEXTBL WHERE "Landing _Outcome" like 'Succes%' AND Date BETWEEN '04-06-2010' AND '20-03
```

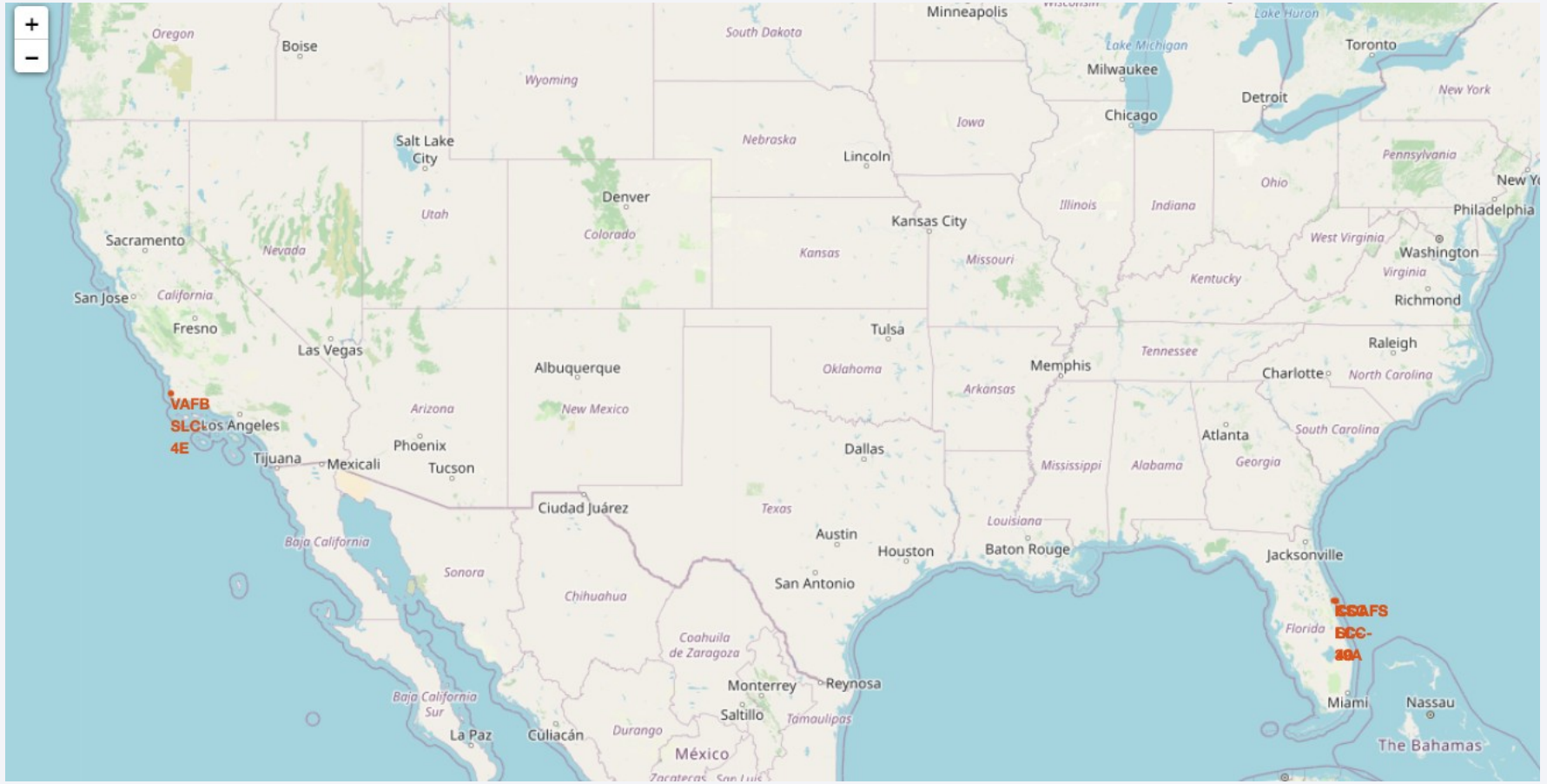
```
 * sqlite:///my_data1.db
Done.
```

Date
19-02-2017
18-10-2020
18-08-2020
18-07-2016
18-04-2018

Section 3

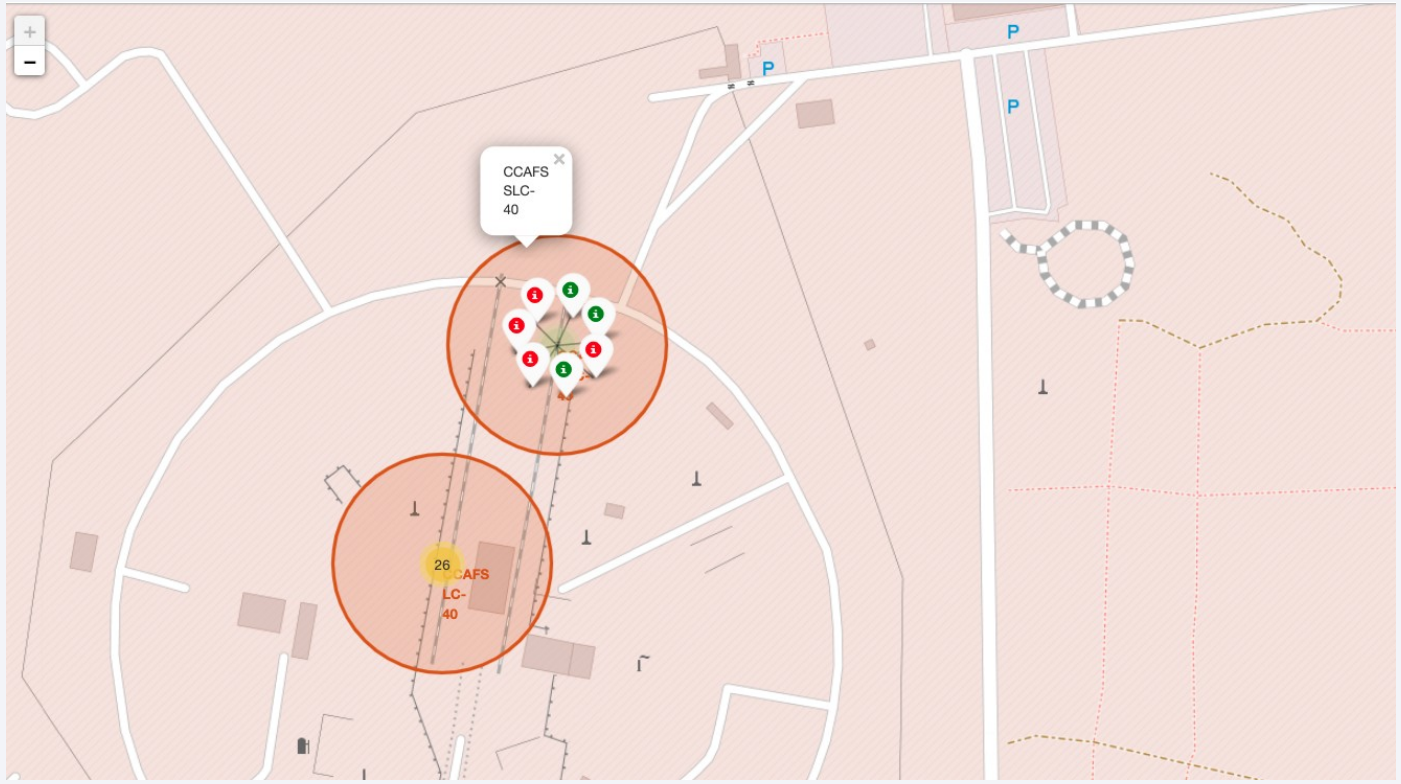
Launch Sites Proximities Analysis

All Launch Sites



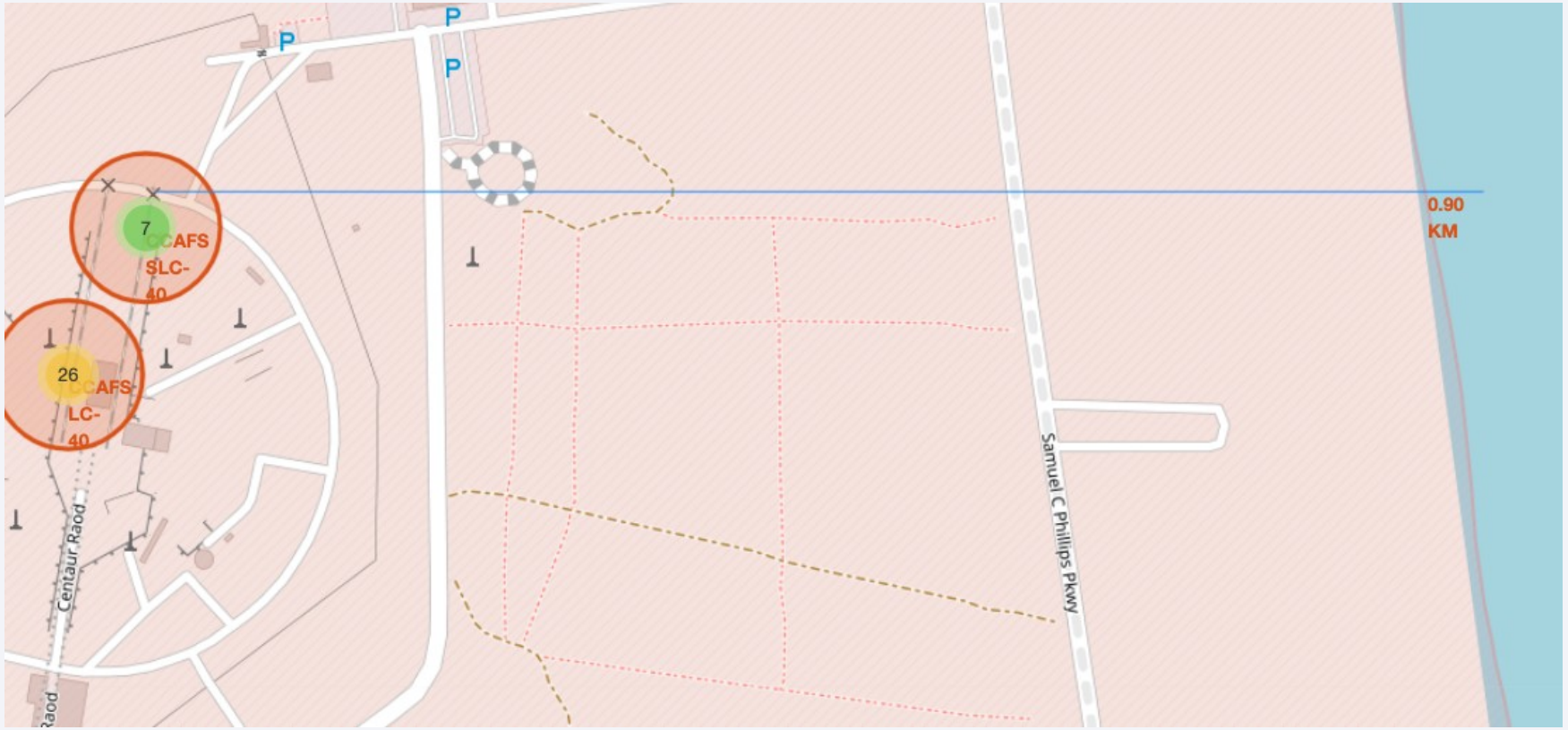
All Launch Sites are in the U.S.A and near the coast

Launch Site with markers

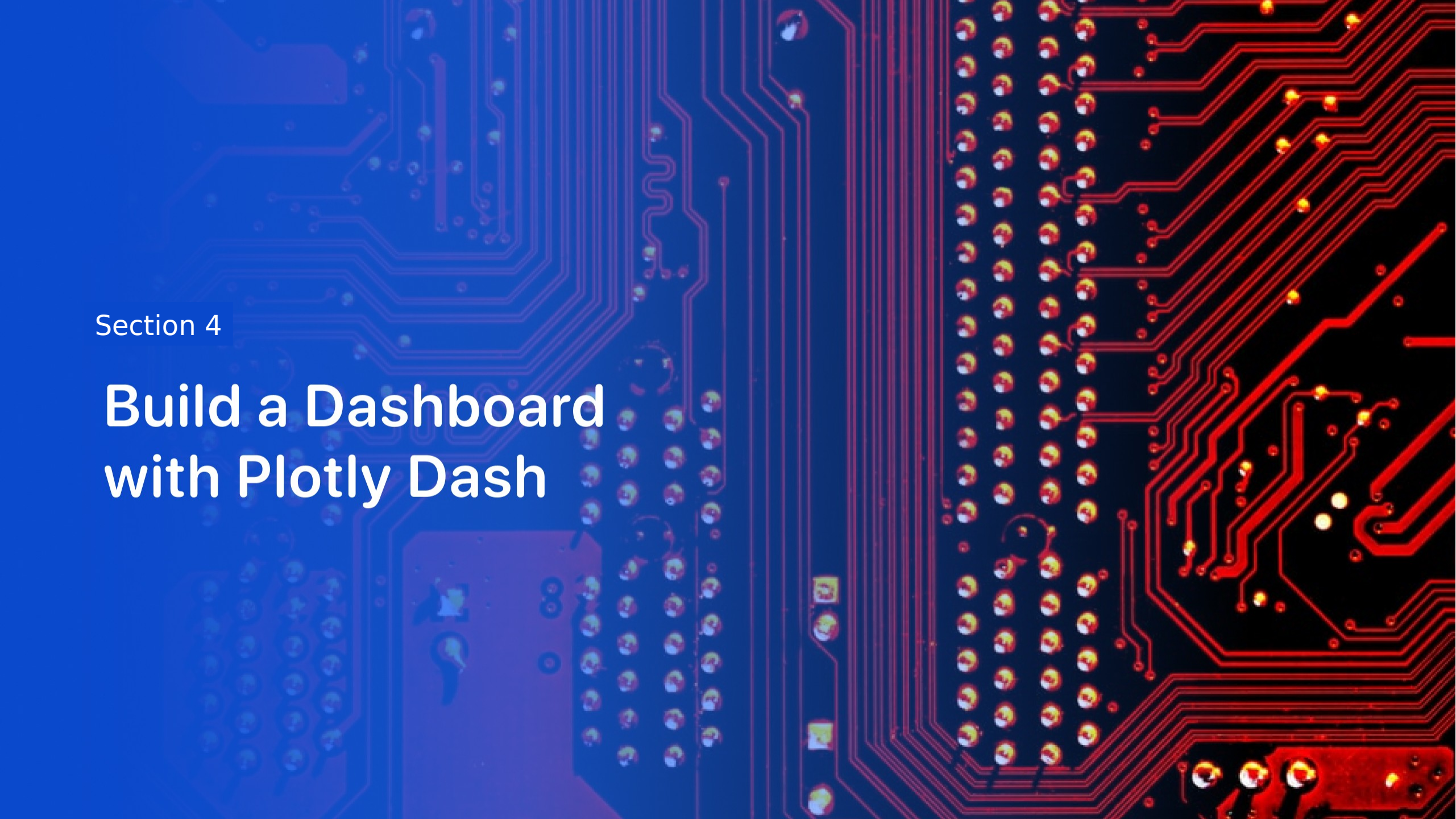


Each marker cluster enables to show on the map the number of launches and their outcome.

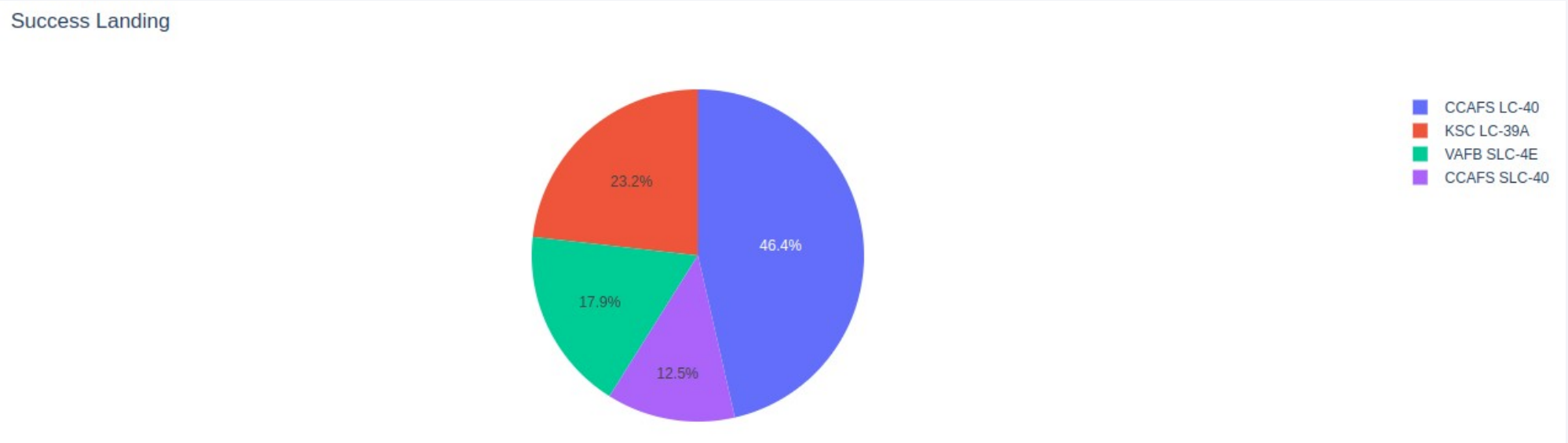
Launch Site proximities



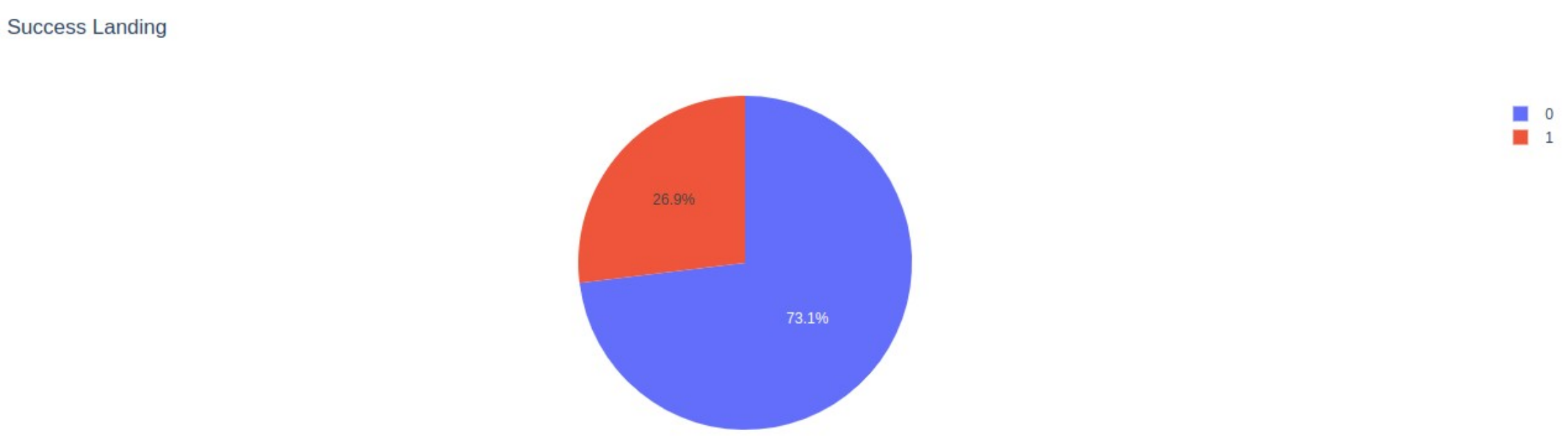
Every launch site is near the coast and far from city centers and far from railways and highways


Section 4
Build a Dashboard with Plotly Dash

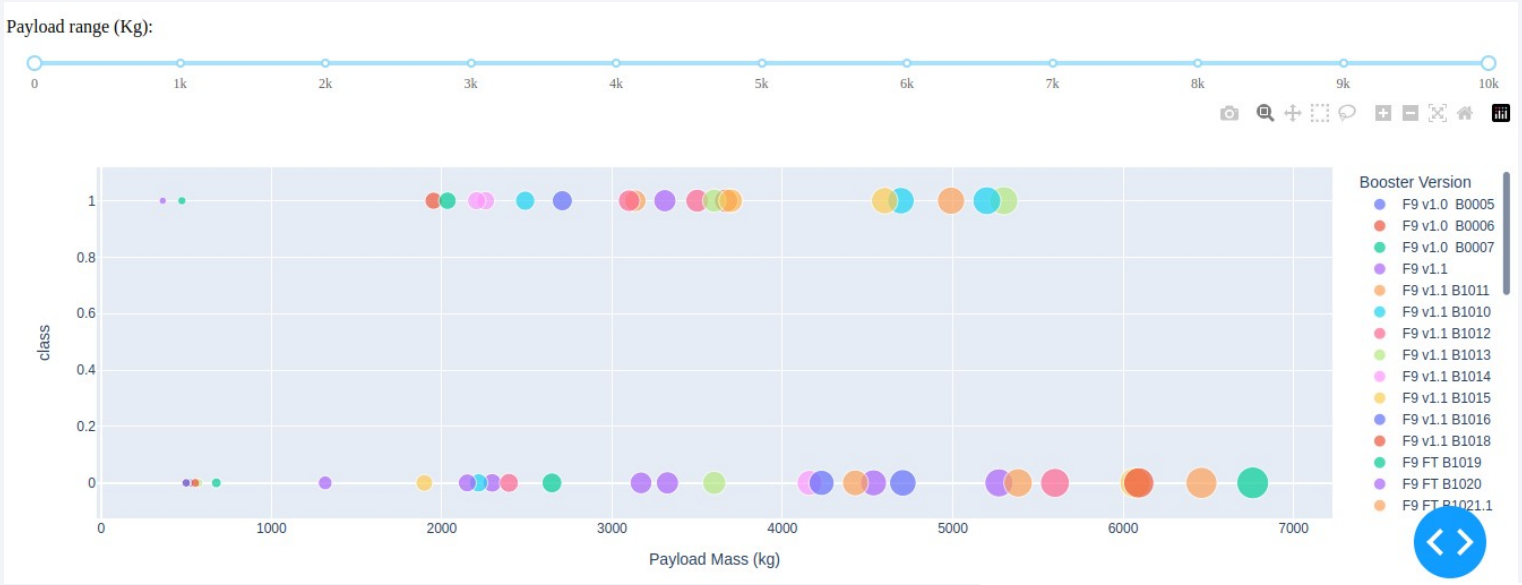
Pie Chart – Success Count per Launch site



CCAFS LC-40 Success Ratio (Pie Chart)

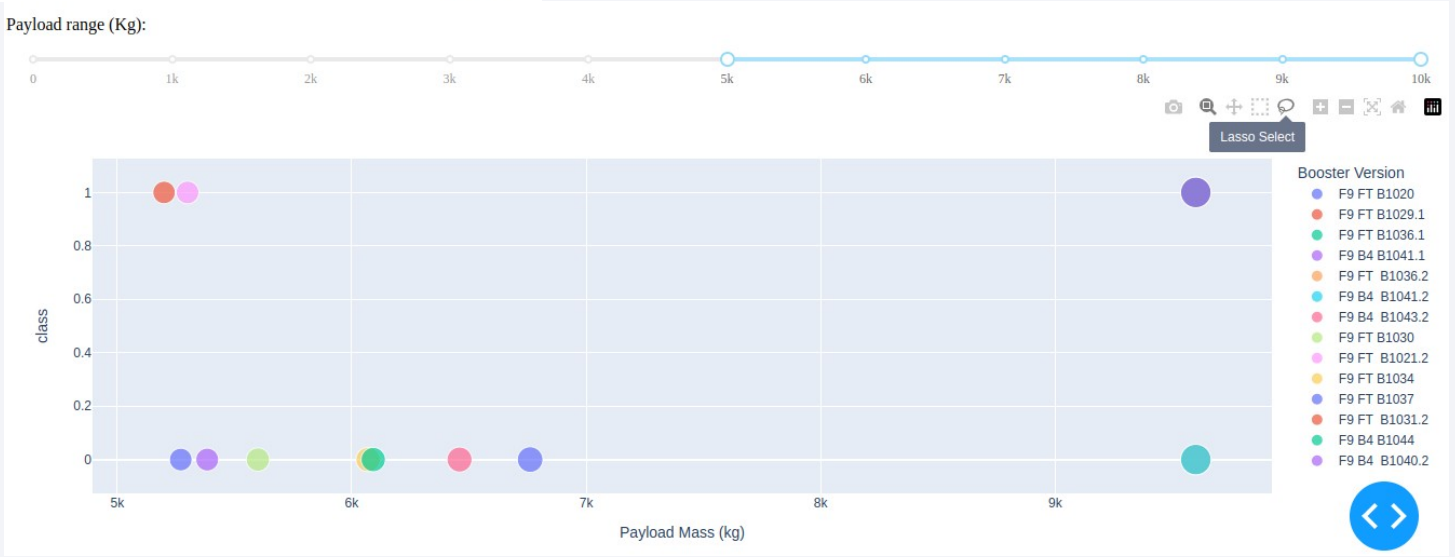


Scatter Plot by Payload range



Scatter plot for all sites for all Payload Range

Scatter plot for all site for a Payload Range of 5000Kg to 1000Kg



Section 5

Predictive Analysis (Classification)

Classification Accuracy

The model with the best accuracy was the Decision Tree

```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.9444444444444444
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

With following parameters obtained through GridSearchCV

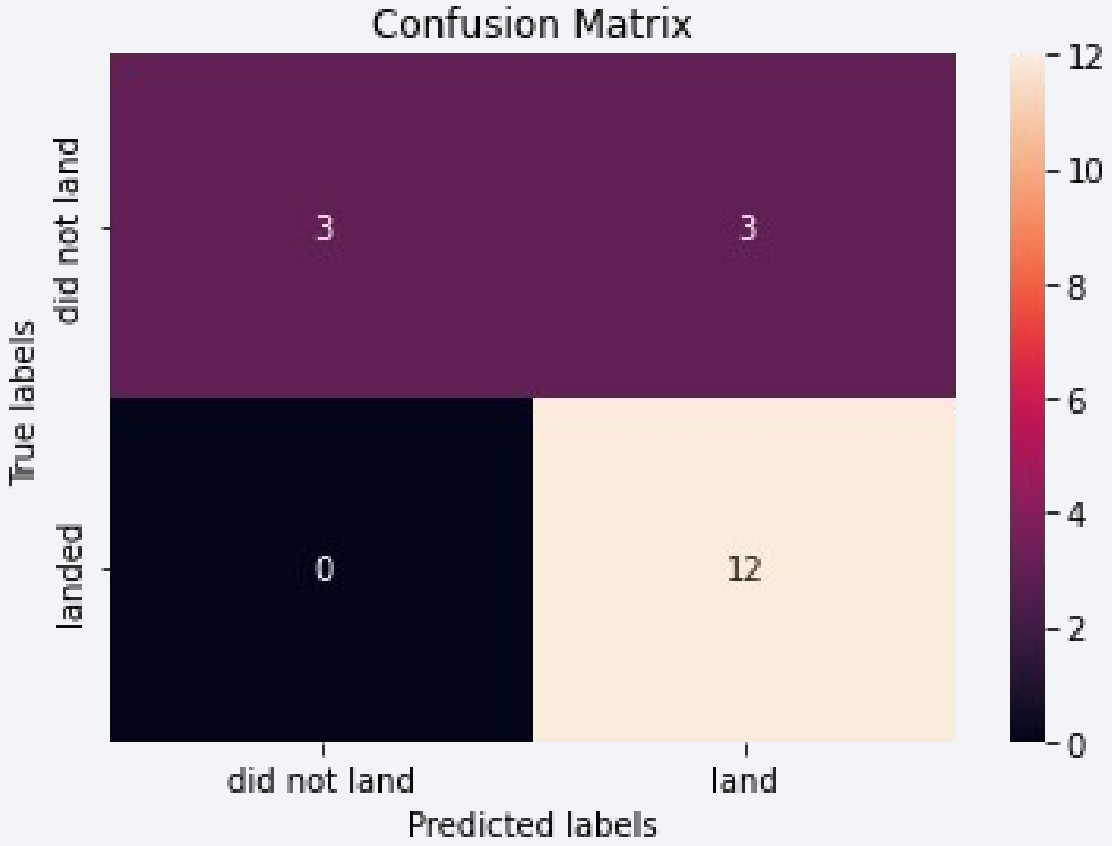
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 6, 'max_features': 'sqrt', 'min_
samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
accuracy : 0.875
```

```
tree_cv.score(X_test, Y_test)
```

```
0.9444444444444444
```

Confusion Matrix



We can see that the model correctly classified 12 landing, however it classified 3 as landed but in fact didn’t land and 3 correctly classified as didn’t land

Conclusions

We can conclude that the rate of success increased from 2013 until today.

The orbits ES-L1, GEO, HEO, SSO have higher success rate

The CCAFS LC-40 has the highest Landing Success Ratio

The Decision Tree is the algorithm with the best performance

Thank you!