IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

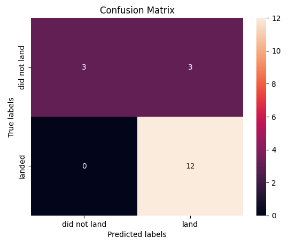
Adrian Llop
15th June 2024

Outline

Executive Summary

Introduction

Methodology



Results

Conclusion

Appendix

Executive Summary

Summary of methodologies

- **Data Collection**
  - SpaceX API
  - Web scraping
- **Data Wrangling**
- **Exploratory Data Analysis (EDA) with Data Visualization**
- **EDA with SQL**
- **Interactive Analytics: Folium and Plotly Dash**
- **Predictive Analysis:**
  - Logic Regression, SVM, Decision Tree and KNN

Summary of results

- **KSC LC-39A has the highest success ratio.**
- **Payload below 8000 Kg for most of the launches.**
- **A 50 to 70 % success is achieved in the most targeted orbits.**
- **Successful outcomes have been growing over time.**
- **We are able to explore critical locations nearby the launching infrastructure with the Dashboard.**
- **KNN was chosen to predict the launch outcome with around 83% accuracy.**

Introduction

Space X has achieved to reuse the first stage module in rockets reducing the cost per launch to $62M versus $165M from other companies that do not reuse it. We want to develop a model to predict whether a launch outcome will be successful or not provided certain features, to use this information against other companies' bid in future projects.

Section 1

Methodology

Methodology

- **Data Collection**
  - SpaceX API
  - Web scraping
- **Data Wrangling**
- **Exploratory Data Analysis (EDA) with Data Visualization**
- **EDA with SQL**
- **Interactive Analytics: Folium and Plotly Dash**
- **Predictive Analysis:**
  - Logic Regression, SVM, Decision Tree and KNN

Data Collection

- **We explored two strategies for data collection: Web Scraping and SPACE X API request.**
- **The data was stored in a data frame including the columns of interest, such as launch sites, outcomes, orbits, etc.**
- **The data was cleaned ensuring consistency. In particular, null values in Payloads were substituted by the Payloads mean.**
- **An additional column called 'Class' was appended in the data frame to transform the failed and successful outcomes into numeric values: 0 and 1 respectively.**

Data Collection – SpaceX API

Request and Parse the Data from SPACE X using GET REQUEST

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

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

```python
response.status_code
```

```
200
```

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

```python
# Use json_normalize meethod to convert the json result into a dataframe
json_response = requests.get(static_json_url)

# Decode the response content as JSON
data = json_response.json()

# Convert the JSON data into a Pandas DataFrame
df = pd.json_normalize(data)
```

Store target data in a Data Frame

```python
# Show the head of the dataframe
df_launch.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1A	167.743129	9.047721
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2A	167.743129	9.047721
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2C	167.743129	9.047721
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3C	167.743129	9.047721
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857

Filter Data to obtain only Falcon 9 Launches and substitute missing values from Payload by its mean

data to a new dataframe called `data_falcon9`.

```python
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = df_launch[df_launch['BoosterVersion']=='Falcon 9']
```

Now that we have removed some values we should reset the FlgihtNumber column

```python
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

[LINK TO REPOSITORY]

Data Collection – Scraping

HTTP GET request to Falcon 9 Wiki and create BeautifulSoup from HTML response

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

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

html = requests.get(static_url)
```

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

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

Print the page title to verify if the `BeautifulSoup` object was created properly

```python
# Use soup.title attribute
soup.title
```

Get relevant column names

```html
<tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Version,<br/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0"><a href="#cite_note-booster-11">[b]</a></sup>
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_note-Dragon-12">[c]</a></sup>
</th>
<th scope="col">Payload mass
</th>
<th scope="col">Orbit
</th>
<th scope="col">Customer
</th>
<th scope="col">Launch<br/>outcome
</th>
<th scope="col"><a href="/wiki/Falcon_9_first-stage_landing_tests" title="Falcon 9 first-stage landing tests">Booster<br/>landing</a>
```

Create Data Frame by parsing the HTML tables

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10
...	...	...	...	...	...	...	...	...	...	...	...

[LINK TO REPOSITORY](#)

Data Wrangling

Explore number of launches on each site

```
# Apply value_counts() on column LaunchSite
df.value_counts('LaunchSite')

LaunchSite
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
dtype: int64
```

Explore the occurrence of each orbit and of mission outcome of the orbits

```
landing_outcomes = df.value_counts('Outcome')
landing_outcomes

Outcome
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
dtype: int64
```

```
landing_outcomes = df.value_counts('Outcome')
landing_outcomes

Outcome
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
dtype: int64
```

Create a landing outcome label called 'Class':
0 – Landing failed
1 - Successful

tcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0

[LINK TO REPOSITORY]

EDA with Data Visualization

LINK TO REPOSITORY

We visualize the relation between the launch success and other target features.
In particular, we report the following trends:

Successful launches related with the launch site

- Flight Number vs Launch Site scattering
- Payload vs Launch Site scattering

Successful launches related with the orbit type

- Success Rate vs Orbit Type bar plot
- Flight Number vs Orbit Type scattering
- Payload vs Orbit Type

Launch Success Yearly Trend

EDA with SQL

[LINK TO REPOSITORY]

EDA was also conducted with SQL. Here we summarize the investigated queries to the SCPACEXTABLE:

- **All launch site names:**
  SELECT DISTINCT(Launch_Site) **FROM** SPACEXTABLE;

- **List 5 records of launch site names beginning with 'CCA':**
  SELECT * **FROM** SPACEXTABLE **WHERE** Launch_Site **LIKE** 'CAA%' **LIMIT** 5;

- **Total Payload Mass from NASA (CRS):**
  SELECT SUM(PAYLOAD_MASS__KG_) **FROM** SPACEXTABLE **WHERE** Customer **LIKE** 'NASA (CRS)';

- **Average Payload Mass by F9 v1.1:**
  SELECT AVG(PAYLOAD_MASS__KG_) **AS** Mean_mass **FROM** SPACEXTABLE **WHERE** Booster_Version **LIKE** 'F9 v1.1';

- **First successful ground landing date:**
  SELECT DATE **FROM** SPACEXTABLE **WHERE** Landing_Outcome **LIKE** 'Success (ground pad)' **ORDER BY** Date **ASC LIMIT** 1;

EDA with SQL

- **Successful drone ship landing with payload between 4000 and 6000:**
  SELECT DISTINCT(Booster_Version) FROM SPACEXTABLE WHERE Landing_Outcome LIKE ‘Success (drone ship)’ AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;

- **Total Number of successful and failure mission outcomes:**
  SELECT Landing_Outcome, COUNT(Landing_Outcome) FROM SPACEXTABLE WHERE Landing_Outcome LIKE ‘Success%’ OR Landing_Outcome LIKE ‘Failure%’ GROUP BY Landing_Outcome;

- **List the names of the booster which have carried the maximum payload mass:**
  SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);

- **Rank the count of landing outcomes between 2010-06-04 and 2017/03-20 in descending order:**
  SELECT Landing_Outcome, COUNT(*) AS Count FROM SPACEXTABLE WHERE Date BETWEEN ‘2010-06-04’ AND ‘2017-03-20’ GROUP BY Landing_Outcome ORDER BY Count DESC;

EDA with SQL

- **List the name of the month when there was a failed landing outcome in drone ship during 2015. Add their respective booster versions, and launch site names:**

```
SELECT
    CASE substr(Date,6,2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END AS Month, Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTABLE WHERE Landing_Outcome = 'Failure (drone ship)' AND substr(Date,0,5) = '2015';
```

Build an interactive map with Folium

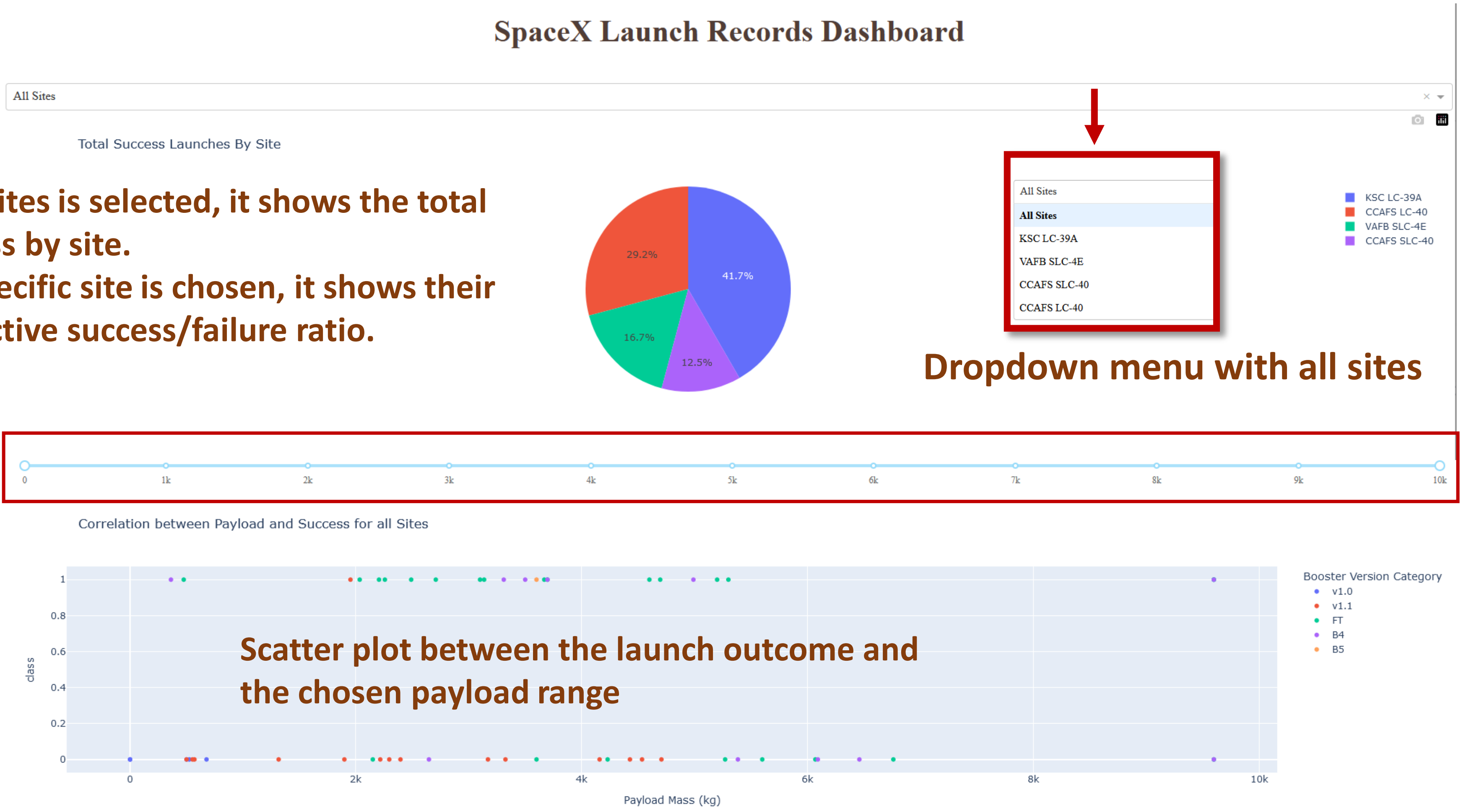
LINK TO REPOSITORY

The interactive map with Folium contains:

- **Circles** indicating the location of the launch sites.
- **Cluster of markers** at each location indicating the launch outcome with a **color code:** green if it was successful and red if unsuccessful.
- **Visualization of the coordinates (longitude and latitude)** of a given point the mouse is pointing.
- **Estimate distances between launch sites and other locations** of interests such as coastlines and railways.
- **Plot lines with their respective labels** indicating the distance between the launch sites and another selected location.

Interactive Dashboard to explore success rates per Site and Payload

[LINK TO REPOSITORY]



Pie chart:

- If All sites is selected, it shows the total success by site.
- If a specific site is chosen, it shows their respective success/failure ratio.

Dropdown menu with all sites

Select payload range

Scatter plot between the launch outcome and the chosen payload range

Predictive Analysis (Classification)

LINK TO REPOSITORY

Split data between train (80%) and test (20%)

Train 4 algorithms:
- Logic Regression
- SVM
- Decision Tree
- KNN

Compare the predicted accuracy

Plot the confusion matrix

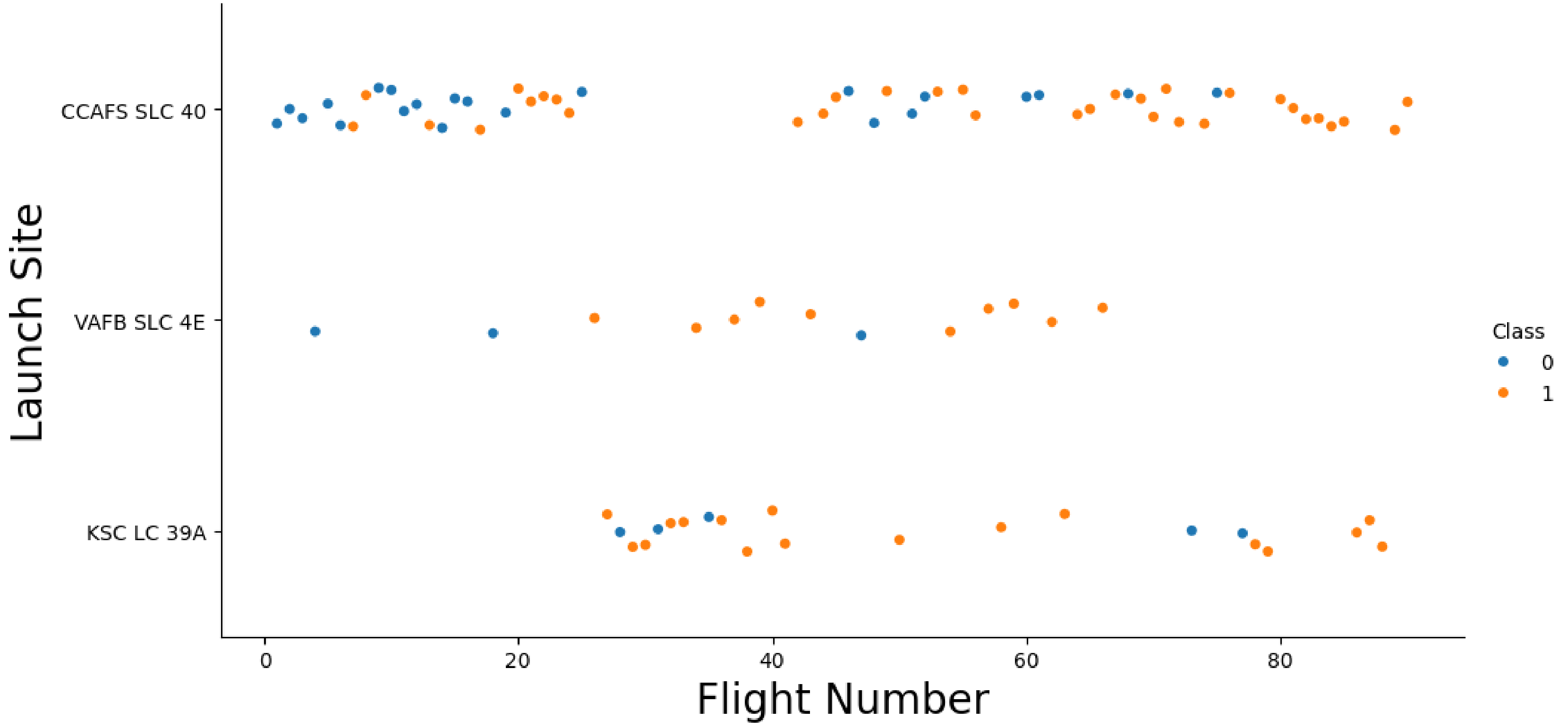
Results

- CCAFS SLC is the site with more launches attempted and KSC LC-39A has the highest success ratio.
- Overall, launches are executed with payload below 8000 Kg.
- When considering the orbits, success ratio ranges from 50 to 70 % in ISS, PO, GTO, VLEO. Further data needs to be collected to assess the success ratio of other orbits.
- Successful outcomes have been growing over time.
- We are able to explore critical locations nearby the launching infrastructure with the Dashboard.
- KNN was chosen to predict the launch outcome with around 83% accuracy. Comparable values are obtained with other three candidates.

Section 2

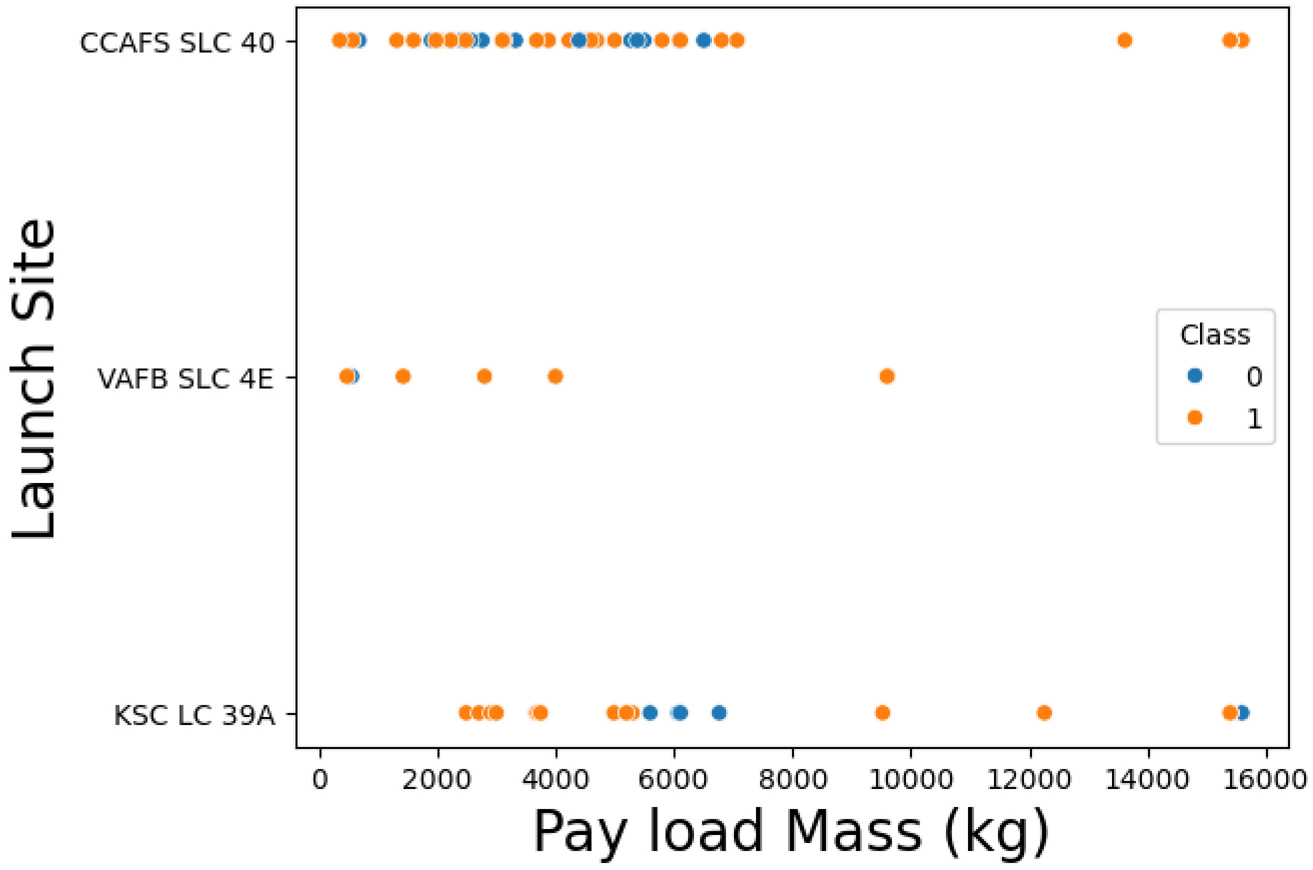
Insights drawn from EDA

Flight Number vs. Launch Site



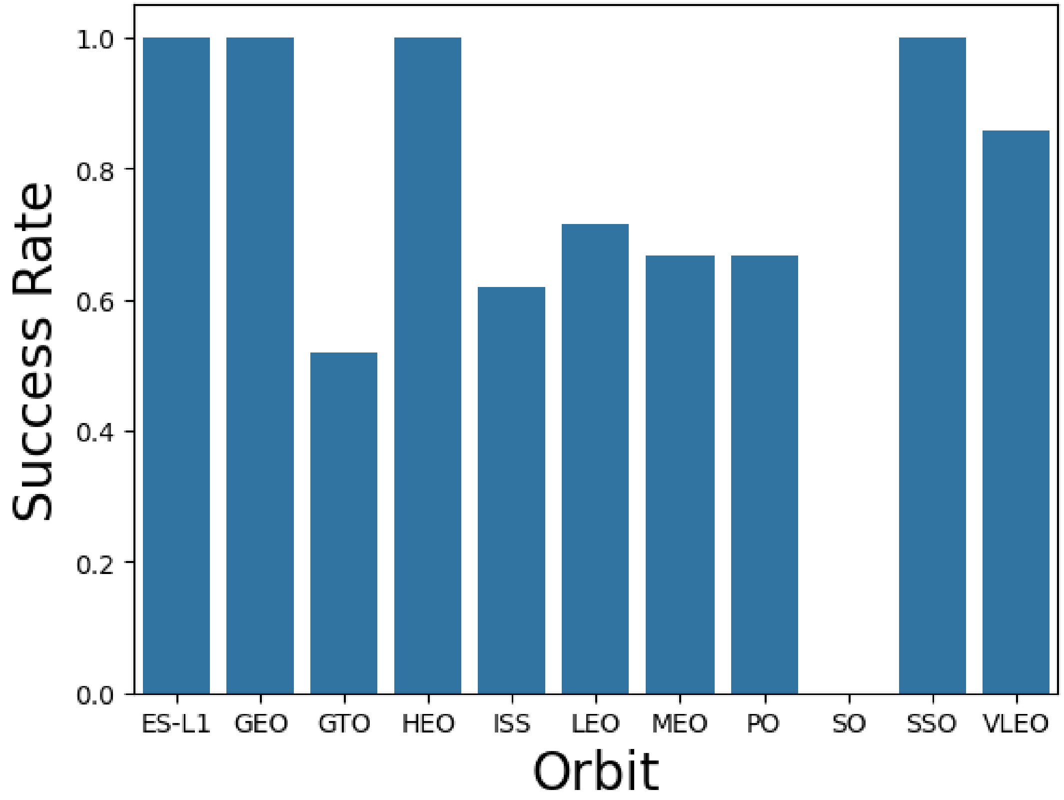
- **CCAFS SLC 40 is the site with more flights registered.**

Payload vs. Launch Site



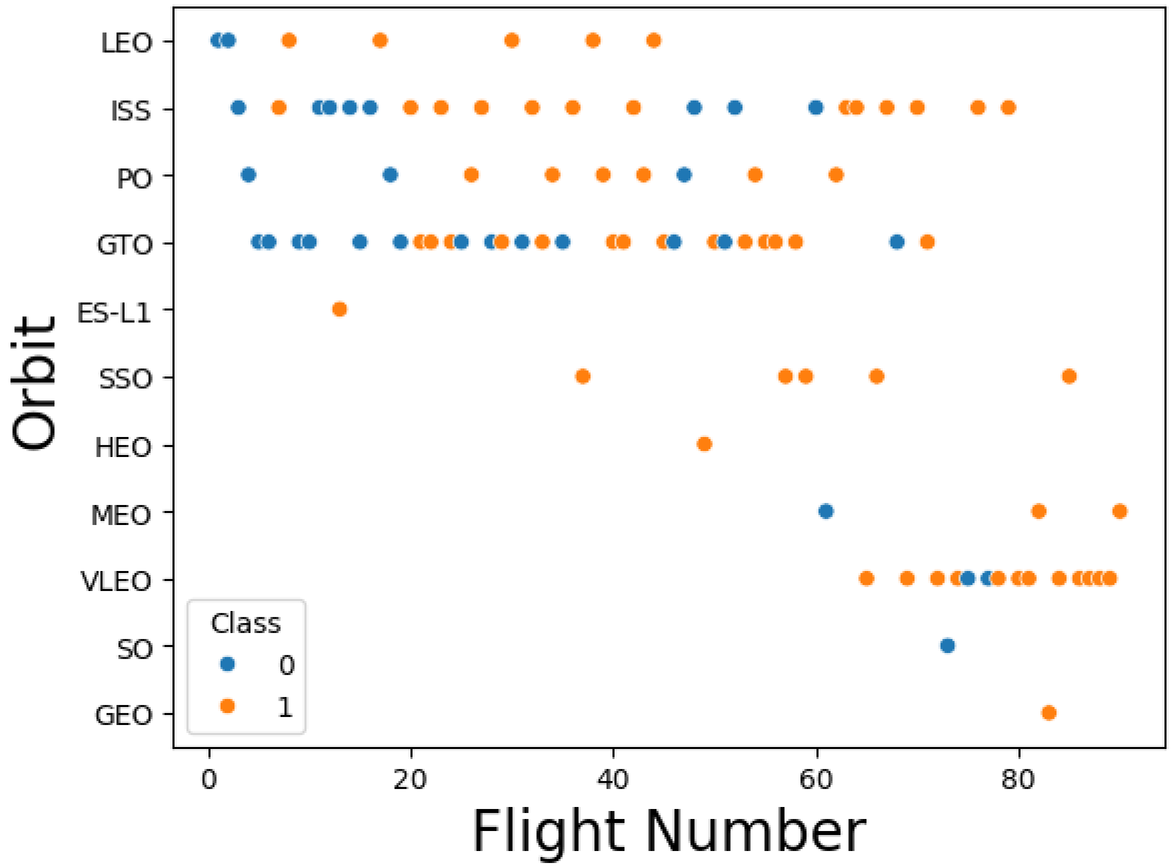
- **Most launches are executed with a payload mass below 8000 Kg.**
- **Most launches registered in CCAFS SLC 40 are below 8000 Kg.**
- **Launches executed in KSC LC 39A are between 2000- 16000 Kg.**
- **VAFB SLC 4E registers the least payload spread.**
- **A priory, considering the scatter distribution, the success does not seem to correlate with specific payload range or launch site.**

Success Rate vs. Orbit



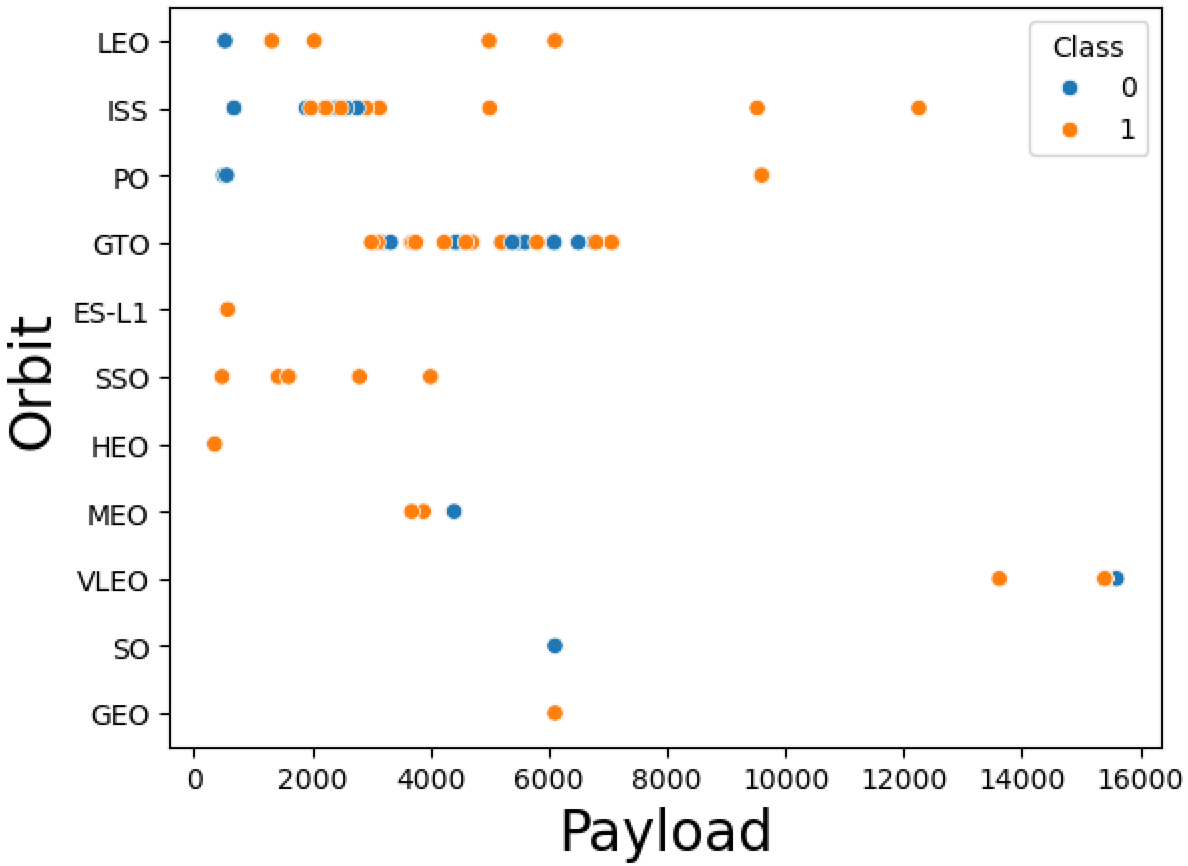
- **Most successful orbits are ES-L1, GEO, HEO, SSO and VLEO with success rate above 80%.**
- **Success rate ranging from 50 to 70% is registered for GTO, ISS, LEO, MEO and PO.**
- **A 0% is registered in launches to SO.**
- **To get the full picture, we need to contextualize this results with the number of launches executed to each orbit.**

Flight Number vs. Orbit



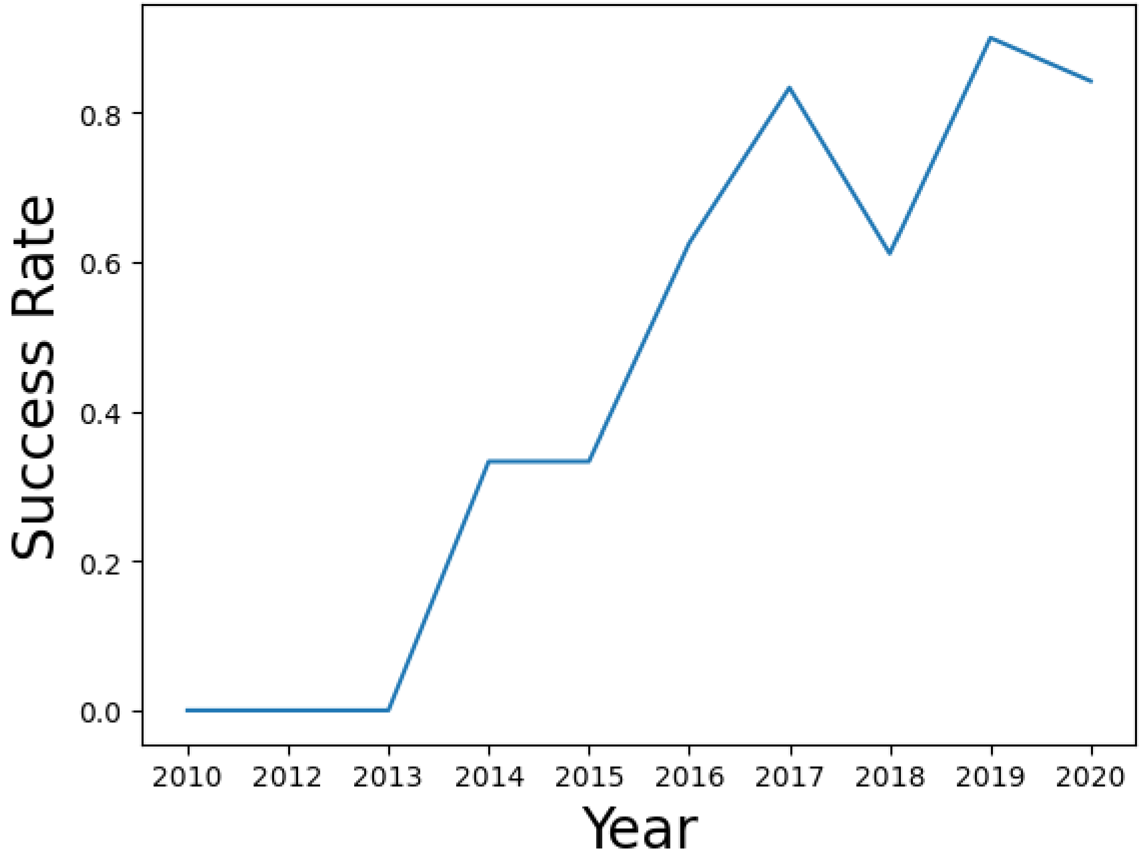
- **Most launches are registered to ISS, PO, GTO and VLEO orbits.**
- **Therefore, caution is advised when considering the success Rates presented in the previous slide regarding ES-L1, SSO, HEO, MEO, SO and GEO orbits. The number of launches executed to reach these orbits is too low to obtain a reliable metric.**

Payload vs. Orbit Type



- **Most of the launches are executed with less than 8000 Kg of payload.**
- **Launches to GTO and the ISS have had a wider range of payloads.**
- **Successful outcomes are detected in all kind of payloads.**
- **Unsuccessful outcomes are registered for LEO, ISS and PO orbits in payloads lower than 1000 Kg.**
- **Successful attempts are registered with the same payload range when reaching ES-L1, SSO and HEO.**

Launch Success Yearly Trend



- **Overall, the success rate has been growing over time**

All Launch Site Names

- **All launch site names can be obtained using DISTINCT:**

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

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

Launch Names Begin with ‘CCA’

- **List 5 records of launch site names beginning with ‘CCA’ using LIKE and LIMIT:**

```
%sql select * from SPACEXTABLE 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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- **Total Payload Mass from NASA (CRS) using SUM():**

```
%sql select SUM(PAYLOAD_MASS__KG_) AS Total_mass from SPACEXTABLE where Customer like 'NASA (CRS)'
```

Total_mass
45596

Average Payload Mass by F9 v1.1

- **Average Payload Mass by F9 v1.1 using AVG():**

```
%sql select AVG(PAYLOAD_MASS__KG_) AS Mean_mass from SPACEXTABLE where Booster_Version like 'F9 v1.1'
```

Mean_mass
2928.4

First Successful Ground Landing Date

- **First successful ground landing date using ORDER BY and ASC:**

```
%sql select Date from SPACEXTBL where Landing_outcome like 'Success (ground pad)' order by Date ASC limit 1
```

Date
2015-12-22

Successful Drone Ship Landing

- **Successful drone ship landing with payload between 4000 and 6000 using BETWEEN:**

```
%sql select distinct(Booster_Version) from spacextbl where Landing_outcome like 'Success (drone ship)' and payload_mass__kg_ between 4000 and 6000
```

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

Total Number of Successful and Failure Missions

- **Total Number of successful and failure mission outcomes using COUNT() and '%':**

```
%sql select Landing_Outcome, count(Landing_Outcome) from spacextbl where Landing_outcome LIKE 'Success%' OR Landing_outcome LIKE 'Failure%'  group by Landing_outcome
```

Landing_Outcome	count(Landing_Outcome)
Failure	3
Failure (drone ship)	5
Failure (parachute)	2
Success	38
Success (drone ship)	14
Success (ground pad)	9

Boosters Carried Maximum Payload

- **List the names of the booster which have carried the maximum payload mass using nested query:**

```
%sql select booster_version from SPACEXTBL where payload_mass__kg_=(select max(payload_mass__kg_) from SPACEXTBL)
```

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

2015 Launch Records

- **List the name of the month when there was a failed landing outcome in drone ship during 2015. Add their respective booster versions, and launch site names. We used CASE, WHEN…THEN…, END clauses and substr():**

```
%%sql
SELECT
    CASE substr(Date, 6, 2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END AS Month,
    Landing_Outcome,
    booster_version,
    launch_site
FROM
    SPACEXTBL
WHERE
    Landing_Outcome = 'Failure (drone ship)' AND
    substr(Date, 0, 5) = '2015';
```

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

Rank Landing Outcomes between 2010-06-04 and 2017-3-02

- **Rank the count of landing outcomes between 2010-06-04 and 2017/03-20 in descending order using COUNT(), GROUP BY, ORDER BY and DESC:**

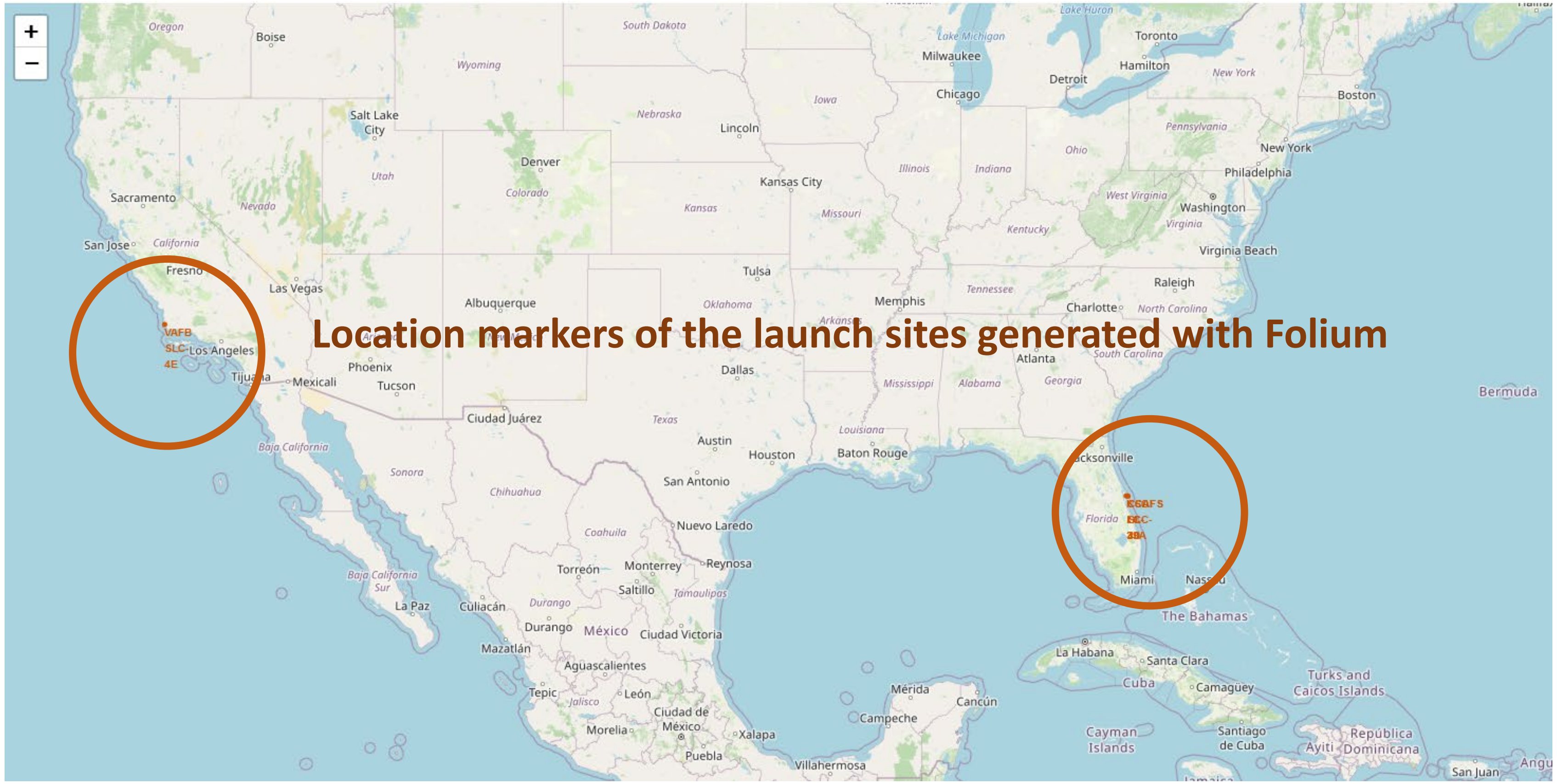
```
%%sql
SELECT
    Landing_Outcome,
    COUNT(*) as count
FROM
    SPACEXTBL
WHERE
    Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY
    Landing_Outcome
ORDER BY
    count DESC;
```

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

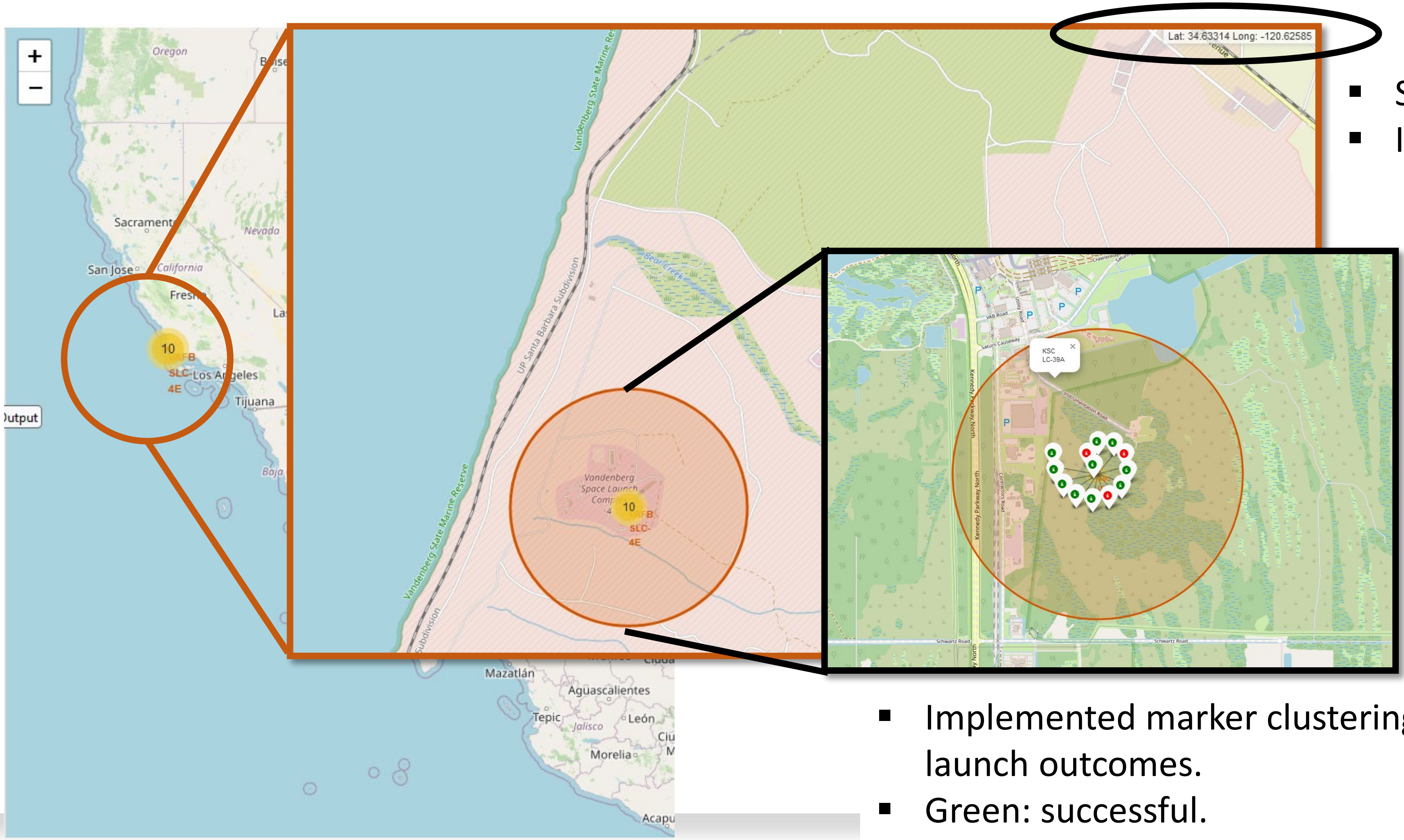
Section 3

Launch Sites Proximities Analysis

Folium Map: Site’s Location Markers



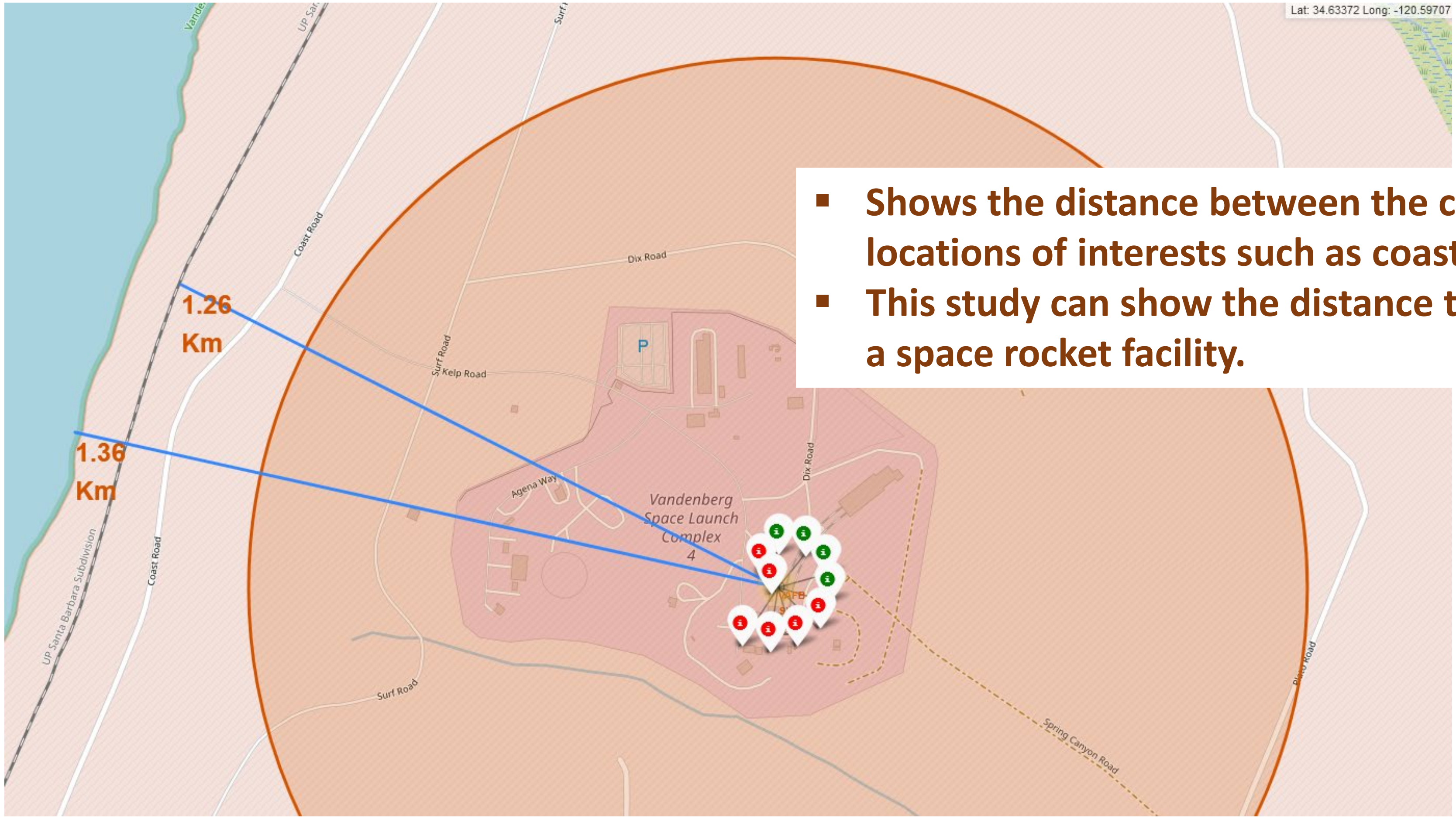
Folium Map: Clustering and Coordinates



- Shows Latitude and Longitude
- Interactive with mouse position

- Implemented marker clustering to represent launch outcomes.
- Green: successful.
- Red: Unsuccessful.

Folium Map: Plot Labeled Distance Lines

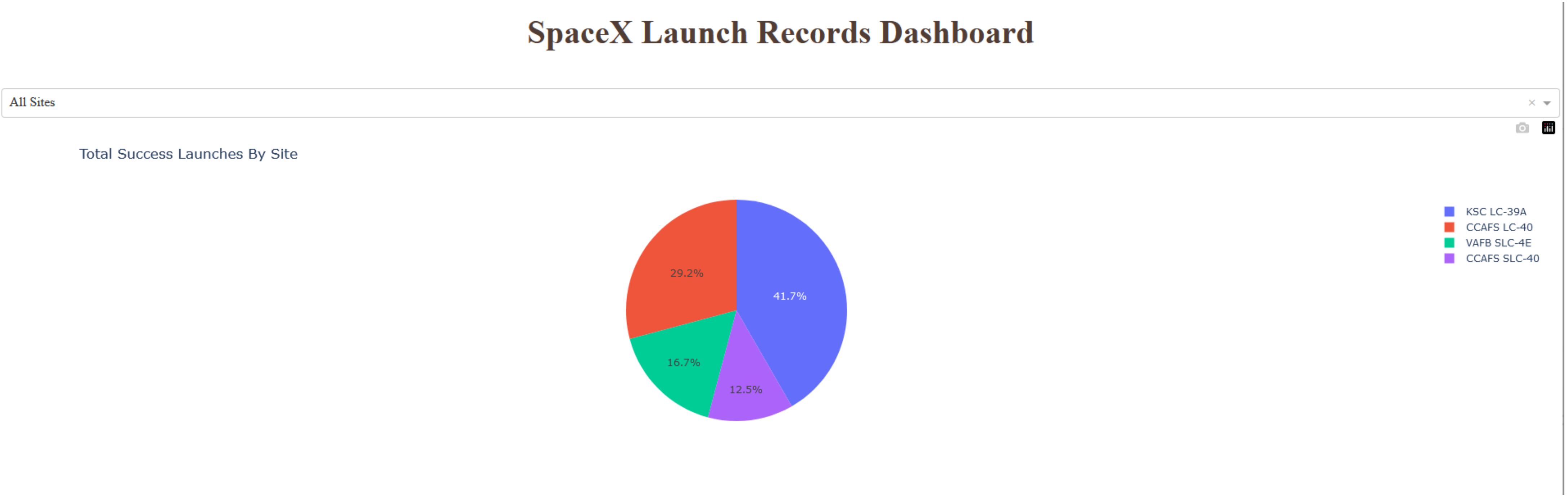


- **Shows the distance between the chosen launch site and other locations of interests such as coastline or railway.**
- **This study can show the distance to critical infrastructure from a space rocket facility.**

Section 4

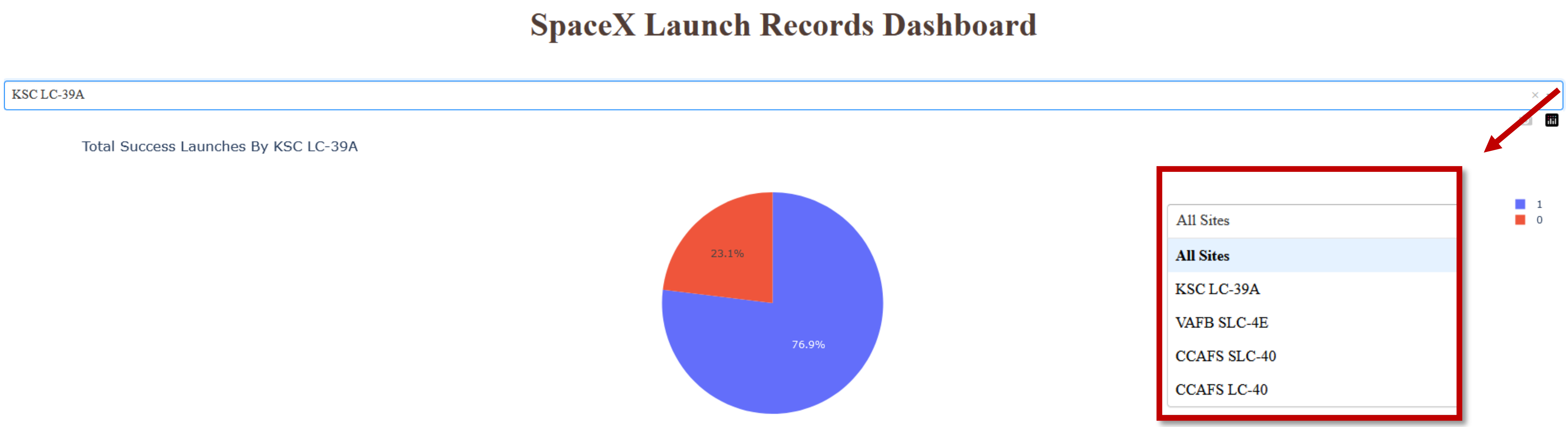
Build a Dashboard with Plotly Dash

Dashboard: All sites in a pie



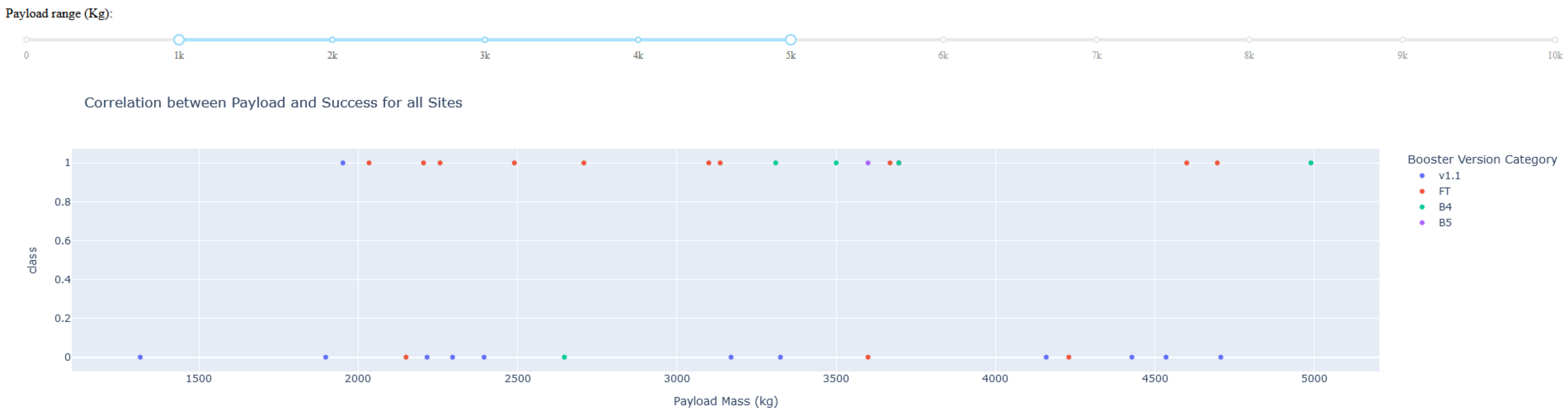
- **We can see the success rate of all Launch Sites**
- **KSC LC-39A is the most successful**

Dashboard: Success per site



- **We can choose from the dropdown list the available Launch Sites to get the specific success and failure proportion.**
- **As an example, we show the KSC LC-39A ratios.**

Dashboard: Range Payload

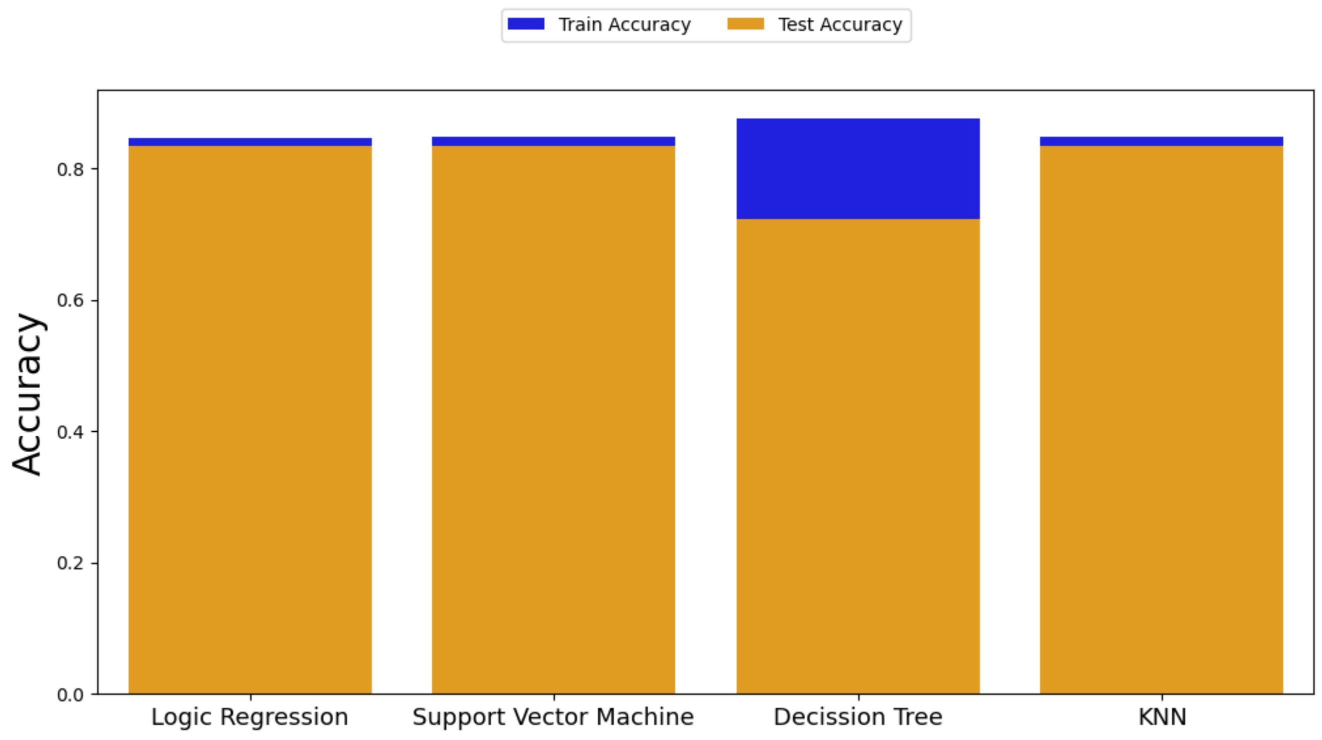


- **Additionally, we implemented a range slide to interactively visualize the outcome per payload range.**
- **The scatter can also be applied to specific sites chosen in the menu.**
- **The color dots correspond to the Booster Version Category.**

Section 5

Predictive Analysis (Classification)

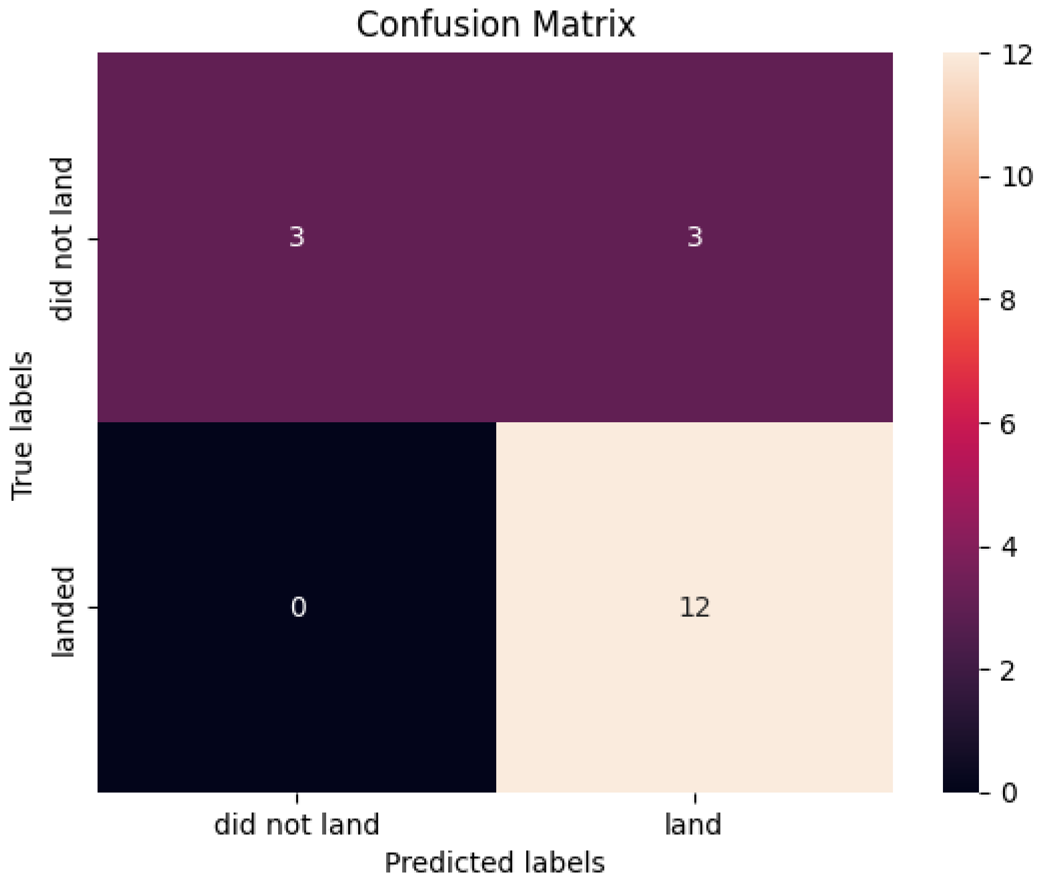
Classification Accuracy



- **The accuracy obtained with the test data is overall smaller than the one obtained with the trained data, as expected.**
- **Similar accuracy is found in Logic Regression, SVM and KNN algorithms.**
- **A critical accuracy difference between the trained and test data is observed with the Decision Tree algorithm.**

Note: An error occurred when compiling the cell code for the Decision Tree training from the template notebook. Apparently, the argument 'auto' in max_features() properties for Decision Tree is not recognised, and only 'log2' or 'sqrt' are allowed. Thus, the input ['auto','sqrt'] was replaced by ['log2','sqrt']. While the other three algorithms show consistent accuracy values, the Decision Tree shows a different accuracy every time it is executed.

Confusion Matrix



- **KNN algorithm was chosen because it presented a slight better accuracy in the decimal order than the Logic Regression and the SVM.**
- **The same confusion Matrix is obtained in the mentioned algorithms.**
- **It predicts the non-landings properly, but there is a margin of error for false positives.**

Conclusions

In this project, we...

- Collected data using API request and Web scraping.
- Prepared the data for supervised training.
- Built maps with data of launch sites.
- Developed a user-friendly dashboard.
- Performed EDA with Data visualization plots and SQL.
- Trained and compared several ML algorithms.

Some valuable insights are

- KSC LC-39A has the highest success ratio.
- Payload below 8000 Kg for most of the launches.
- A 50 to 70 % success is achieved in the most targeted orbits.
- Successful outcomes have been growing over time.
- We are able to explore critical locations nearby the launching infrastructure with the Dashboard.
- KNN was chosen to predict the launch outcome with around 83% accuracy.

Appendix

Click [HERE](#) to access the Public GitHub repository where all Notebook and Python files are stored. Alternatively, you can use this QR:

Scan me!

Thank you!