IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

OLUWATOSIN ADEOYE
02/10/2023

Outline

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

Executive Summary

Summary of methodologies

- Data collection
- Exploratory Data analysis
- Data Visualization
- SQL
- Interactive Map
- Dashboard with plotly
- Predictive analysis with Supervised machine learning

Summary of all results

- EDA results
- Predictive analysis result

Introduction

- **Project background and context**

Space X promotes Falcon 9 rocket launches on its website at a price of $62 million, while other providers charge over $165 million each. A significant portion of the cost difference is due to Space X's ability to recycle the initial stage. Consequently, if we can ascertain the success of the first stage landing, we can estimate the launch cost. This data is valuable for potential competitors looking to compete with Space X for rocket launch contracts. The project's objective is to develop a machine learning process for forecasting the first stage's landing outcome.

- **Problems you want to find answers**

What factors determine if the rocket will land successfully?

The interaction amongst various features that determine the success rate of a successful landing.

What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

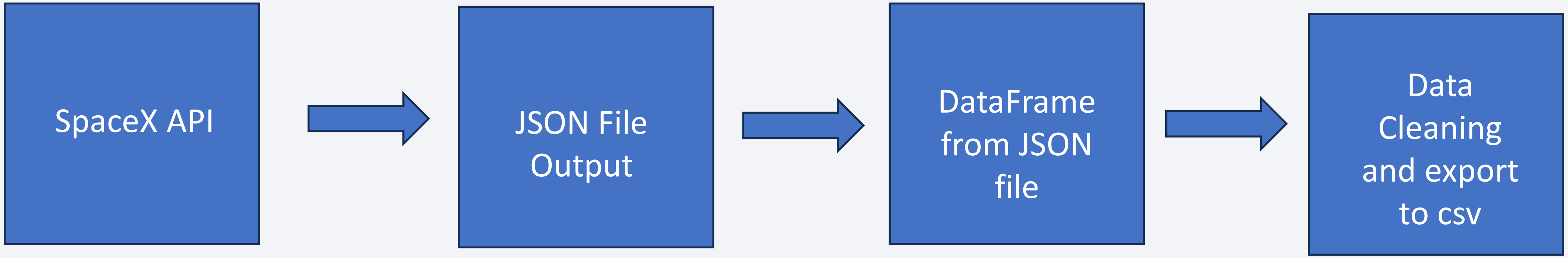
Methodology

Executive Summary

- Data collection methodology:
    - Web scraping from Wikipedia’
    - From SopaceX REST API
- Perform data wrangling
    - Replacing Nan values with mean
    - Dropping Unnecessary columns
    - One hot encoding of the categorical columns
- 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

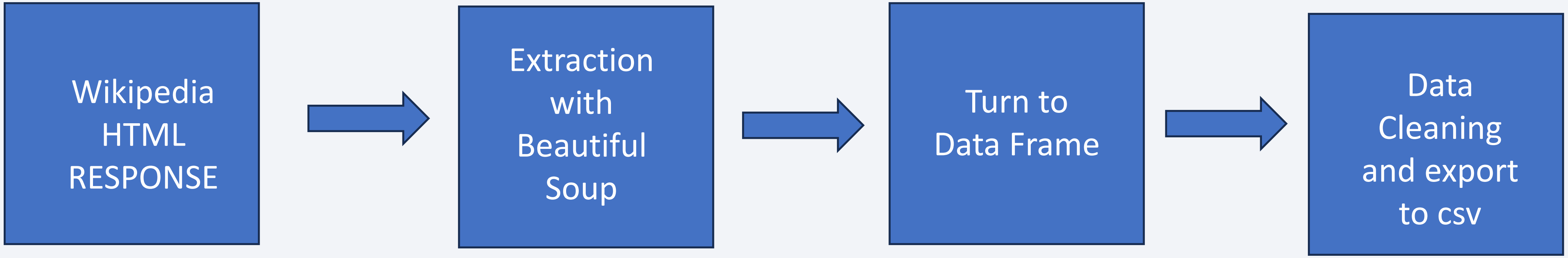
Data Collection

SpaceX API



SpaceX API → JSON File Output → DataFrame from JSON file → Data Cleaning and export to csv

Web scrapping

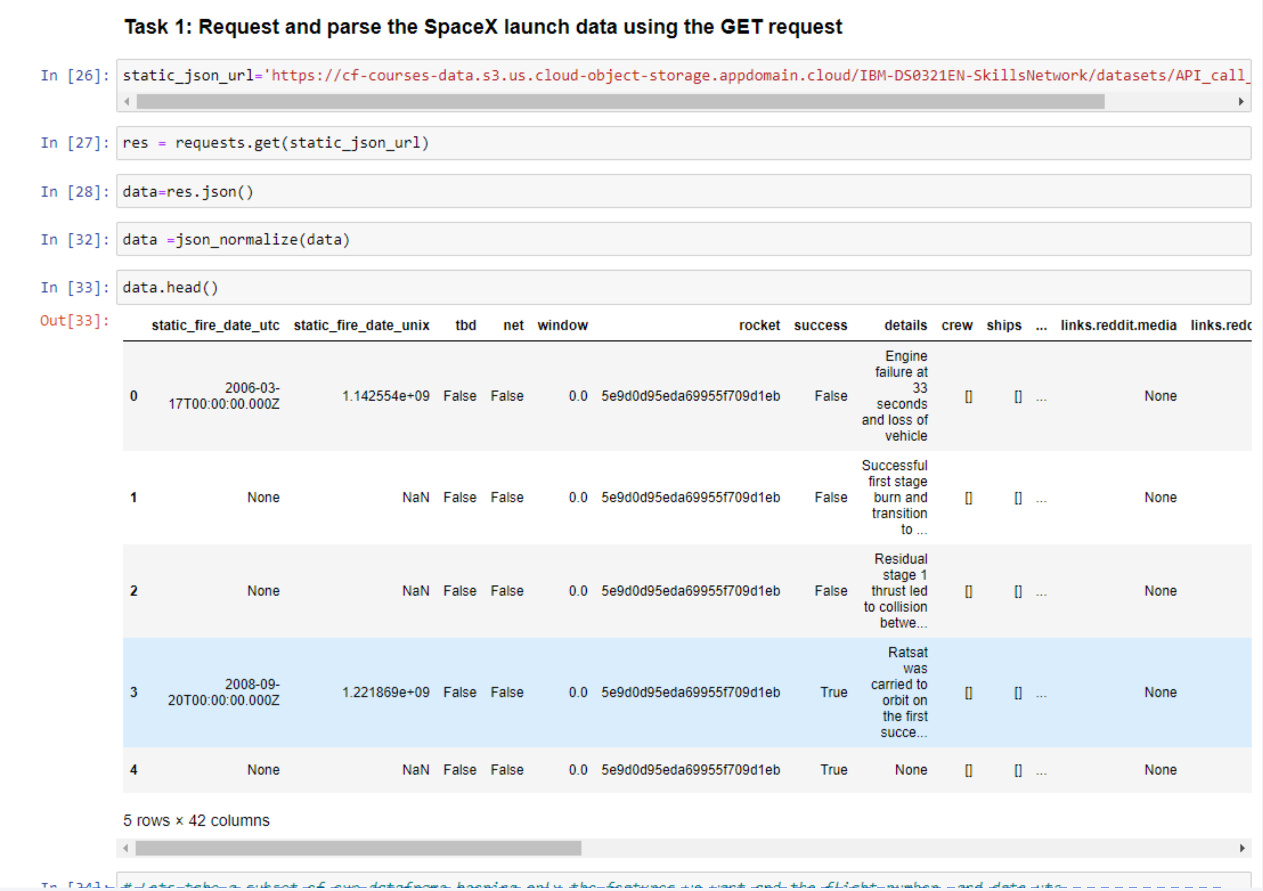


Wikipedia HTML RESPONSE → Extraction with Beautiful Soup → Turn to Data Frame → Data Cleaning and export to csv

Data Collection – SpaceX API

- Link to github

https://github.com/Lawlantosin/IBM_CAPSTONE_PROJECT



Task 1: Request and parse the SpaceX launch data using the GET request

```
In [26]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_

In [27]: res = requests.get(static_json_url)

In [28]: data=res.json()

In [32]: data =json_normalize(data)

In [33]: data.head()
```

Out[33]:

	static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	crew	ships	...	links.reddit.media	links.redc
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Engine failure at 33 seconds and loss of vehicle	[]	[]	...	None	
1	None	NaN	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Successful first stage burn and transition to ...	[]	[]	...	None	
2	None	NaN	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Residual stage 1 thrust led to collision betwe...	[]	[]	...	None	
3	2008-09-20T00:00:00.000Z	1.221869e+09	False	False	0.0	5e9d0d95eda69955f709d1eb	True	Ratsat was carried to orbit on the first succe...	[]	[]	...	None	
4	None	NaN	False	False	0.0	5e9d0d95eda69955f709d1eb	True	None	[]	[]	...	None	

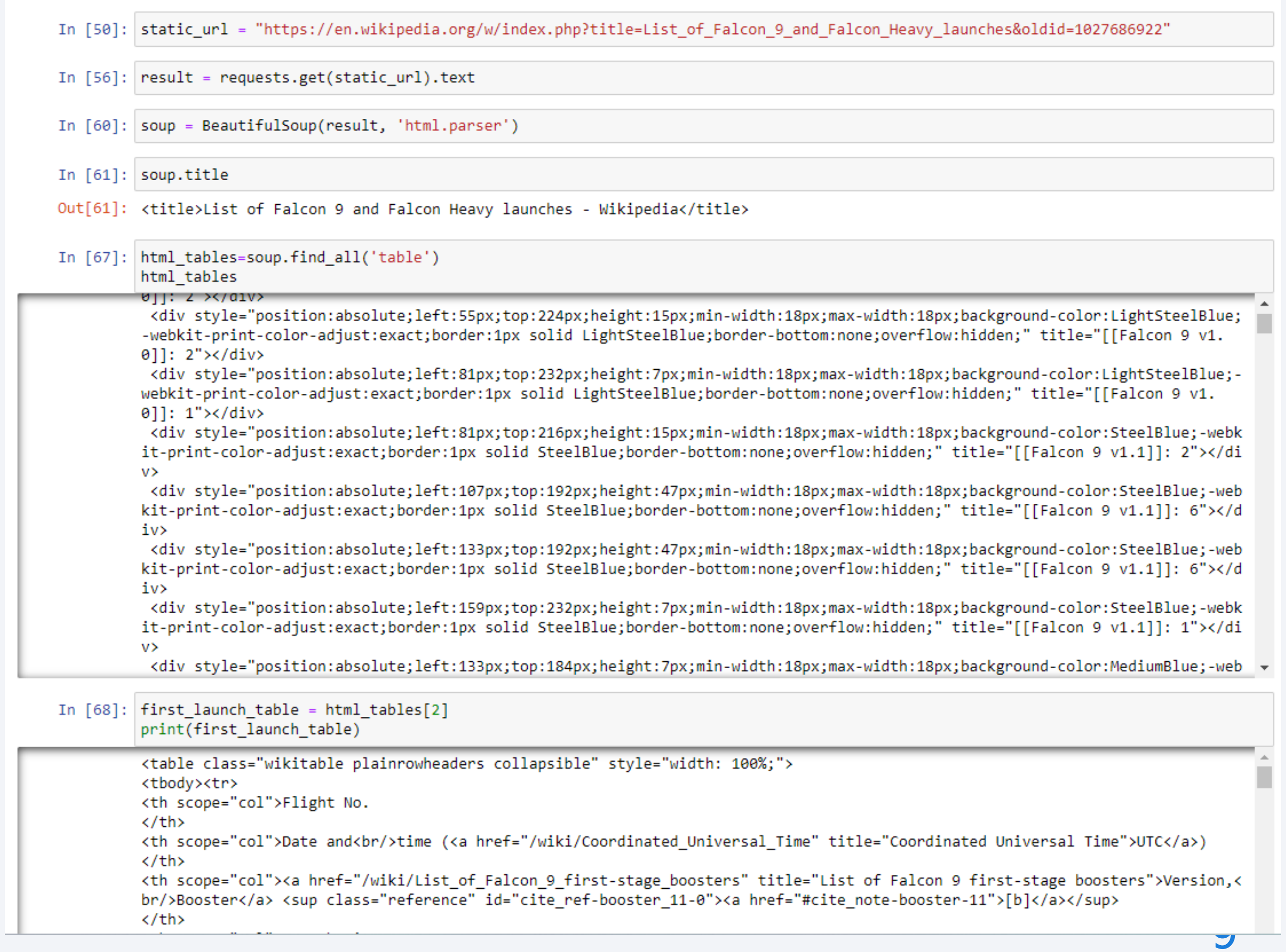
5 rows × 42 columns

```
In [34]: # Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
```

Data Collection - Scraping

- Link to github

https://github.com/Lawlantosin/IBM_CAPSTONE_PROJECT



```
In [50]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

In [56]: result = requests.get(static_url).text

In [60]: soup = BeautifulSoup(result, 'html.parser')

In [61]: soup.title

Out[61]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>

In [67]: html_tables=soup.find_all('table')
         html_tables

         0]]: 2"></div>
          <div style="position:absolute;left:55px;top:224px;height:15px;min-width:18px;max-width:18px;background-color:LightSteelBlue;
         -webkit-print-color-adjust:exact;border:1px solid LightSteelBlue;border-bottom:none;overflow:hidden;" title="[[Falcon 9 v1.
         0]]: 2"></div>
          <div style="position:absolute;left:81px;top:232px;height:7px;min-width:18px;max-width:18px;background-color:LightSteelBlue;-
         webkit-print-color-adjust:exact;border:1px solid LightSteelBlue;border-bottom:none;overflow:hidden;" title="[[Falcon 9 v1.
         0]]: 1"></div>
          <div style="position:absolute;left:81px;top:216px;height:15px;min-width:18px;max-width:18px;background-color:SteelBlue;-webk
         it-print-color-adjust:exact;border:1px solid SteelBlue;border-bottom:none;overflow:hidden;" title="[[Falcon 9 v1.1]]: 2"></di
         v>
          <div style="position:absolute;left:107px;top:192px;height:47px;min-width:18px;max-width:18px;background-color:SteelBlue;-web
         kit-print-color-adjust:exact;border:1px solid SteelBlue;border-bottom:none;overflow:hidden;" title="[[Falcon 9 v1.1]]: 6"></d
         iv>
          <div style="position:absolute;left:133px;top:192px;height:47px;min-width:18px;max-width:18px;background-color:SteelBlue;-web
         kit-print-color-adjust:exact;border:1px solid SteelBlue;border-bottom:none;overflow:hidden;" title="[[Falcon 9 v1.1]]: 6"></d
         iv>
          <div style="position:absolute;left:159px;top:232px;height:7px;min-width:18px;max-width:18px;background-color:SteelBlue;-webk
         it-print-color-adjust:exact;border:1px solid SteelBlue;border-bottom:none;overflow:hidden;" title="[[Falcon 9 v1.1]]: 1"></di
         v>
          <div style="position:absolute;left:133px;top:184px;height:7px;min-width:18px;max-width:18px;background-color:MediumBlue;-web

In [68]: first_launch_table = html_tables[2]
         print(first_launch_table)

         <table class="wikitable plainrowheaders collapsible" style="width: 100%;">
         <tbody><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>
```

Data Wrangling

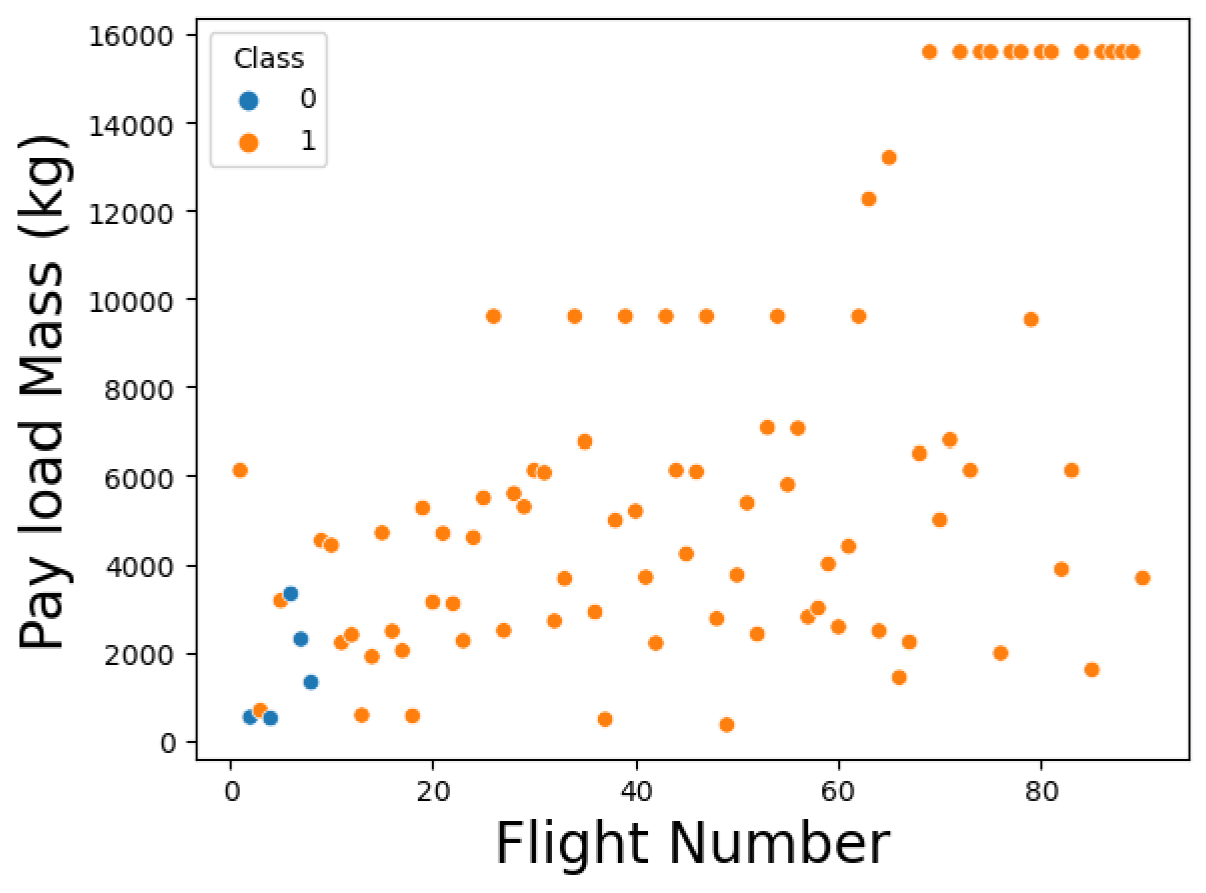
- Invalid columns were removed
- Only the dataframe containing falcon 9 was selected
- The null values in the payloadmass was replaced with its mean
- The null values in landing pad was left untouched
- The value_counts of launchsite, landing outcomes were checked
- The bad landing outcome were grouped in class 0 and the good was grouped in 1

https://github.com/Lawlantosin/IBM_CAPSTONE_PROJECT

EDA with Data Visualization

- The payload mass and the flight number has a positive correlation

- https://github.com/Lawlantosin/IBM_CAPSTONE_PROJECT



EDA with SQL

- The unique launch sites in the space mission
- Total payload mass carried by booseters launched by NASA (CRS)
- Names of the booster version that carried the maximum payload mass
- The average payload mass carried by booster version F9 v1.1
- The total number of successful and failure mission outcome.
- The failed landing outcomes in drone ship, their booster version and launch site names.

https://github.com/Lawlantosin/IBM_CAPSTONE_PROJECT

Build an Interactive Map with Folium

- Labeled all the launch sites and incorporated map elements like markers, circles, and lines to indicate launch outcomes (either success or failure) for each site on the Folium map.
- Assigned the launch outcomes to two classes: 0 for failure and 1 for success.
- By analyzing the marker clusters with their respective colors, we identified launch sites with a comparatively high success rate.

https://github.com/Lawlantosin/IBM_CAPSTONE_PROJECT

Build a Dashboard with Plotly Dash

- Built an interactive dashboard with Plotly dash
- Plotted a pie charts showing the total launches by a certain sites
- Plotted a scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

https://github.com/Lawlantosin/IBM_CAPSTONE_PROJECT

Predictive Analysis (Classification)

- Load the dataset
- Normalize the data using standard scaler
- Split the data into training and test sets
- Select the machine algorithm such as logistics regression, SVM, KNN and Decision Tree
- Set the parameters for each algorithm for Gridsearch cv
- Training Gridsearch Models on the training set
- Get the best parameters
- Compute Accuracy
- Plot confusion matrix
- Do Model comparison
- Select best model

https://github.com/Lawlantosin/IBM_CAPSTONE_PROJECT

Results

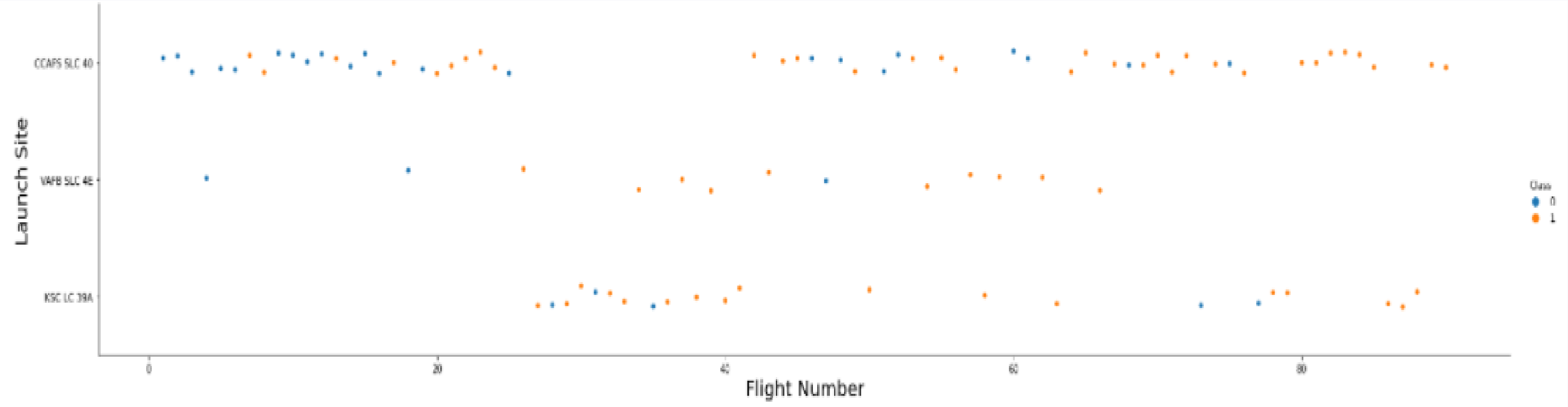
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

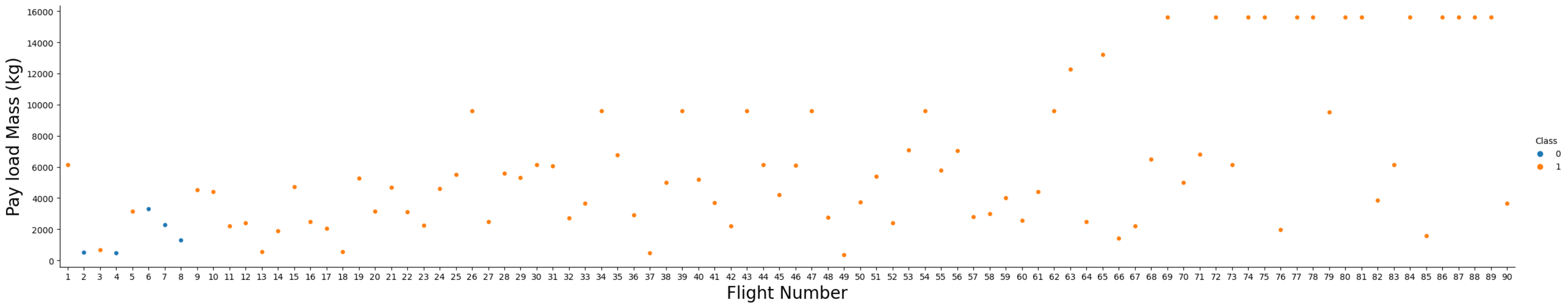
Flight Number vs. Launch Site

- As we can see from the plot, the larger the flight amount, the greater the success rate



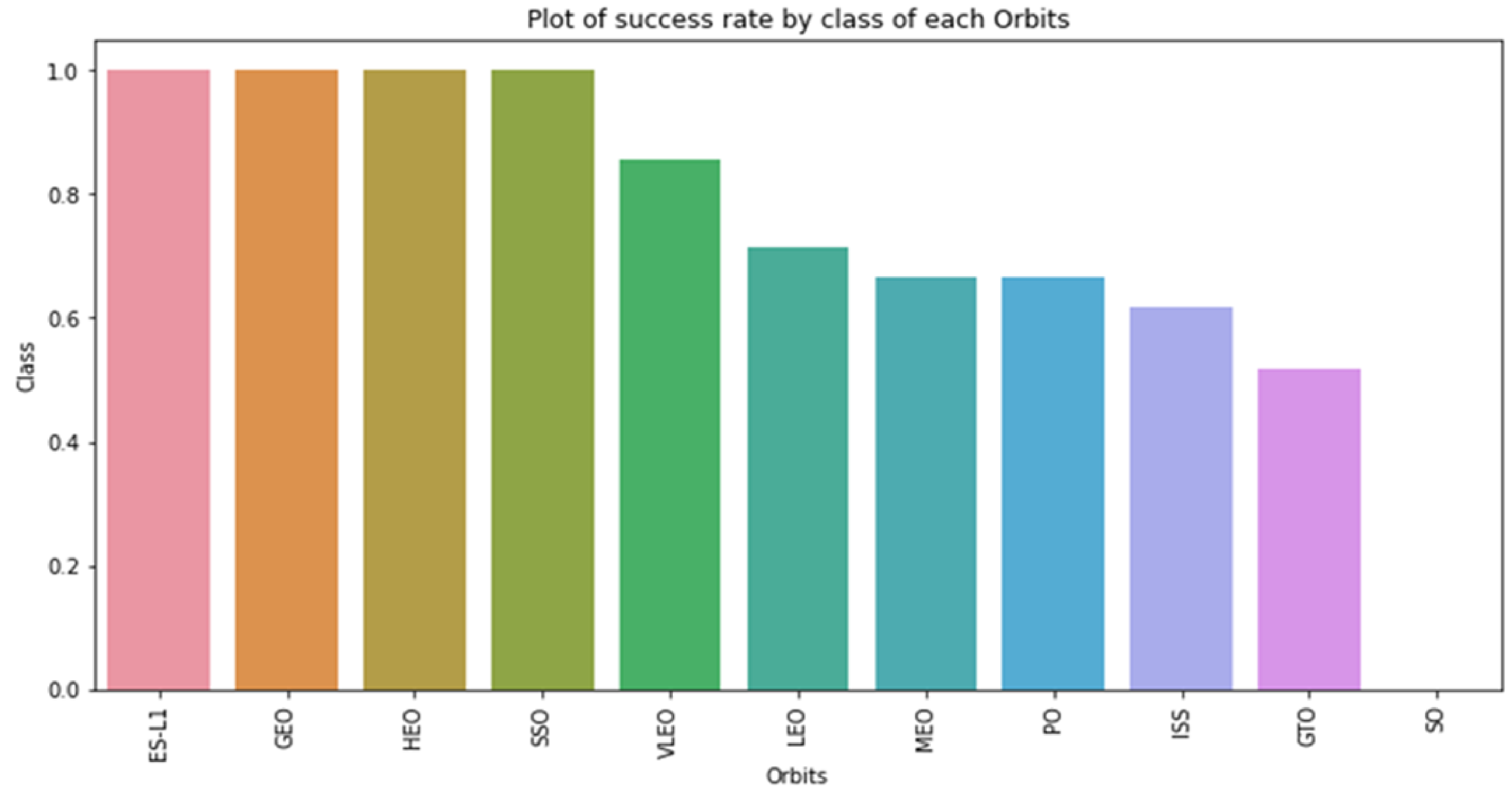
Payload vs. Launch Site

- The greater the payload mass the higher the success of the rocket



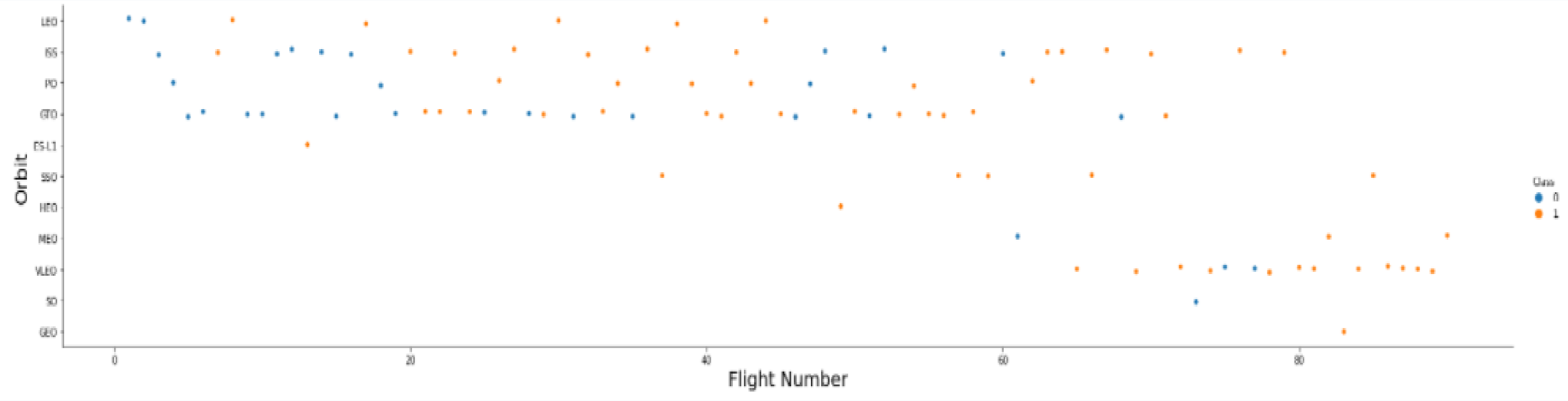
Success Rate vs. Orbit Type

- ES-L!, GEO, HEO, SSO and VLEO had the most success rate



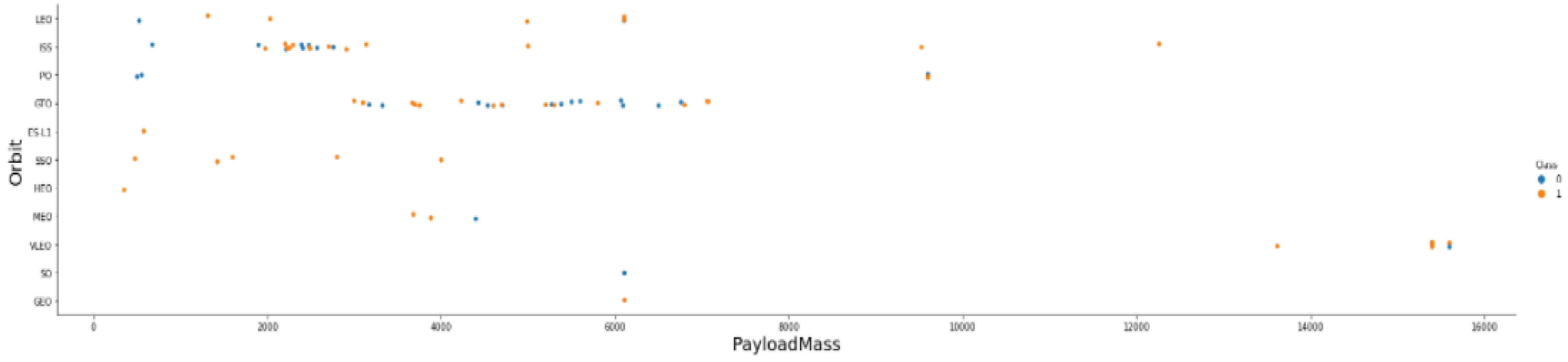
Flight Number vs. Orbit Type

- LEO orbit has a positive relationship while GTO orbit has no relationship



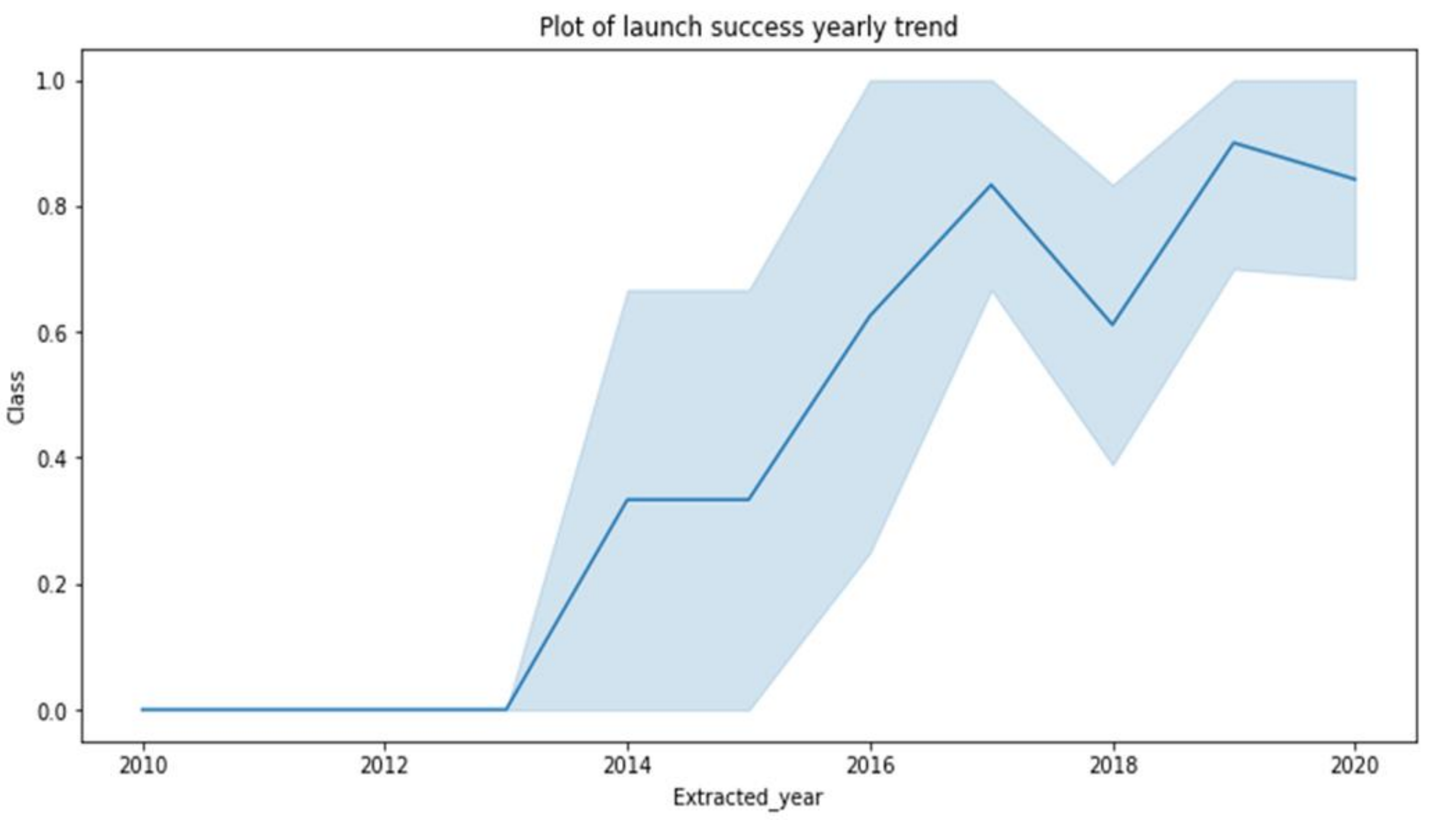
Payload vs. Orbit Type

- PO, LEO and ISS has more successful landing



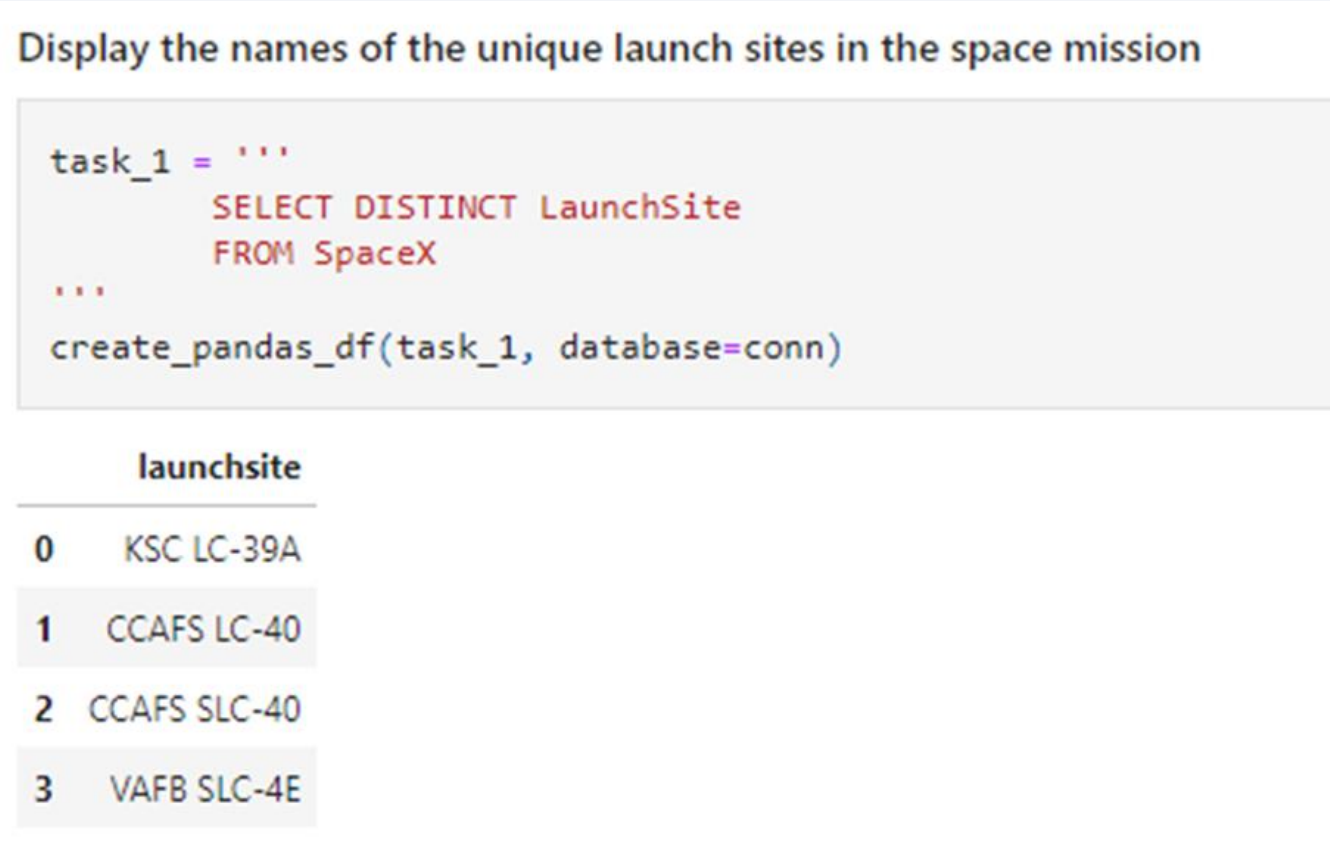
Launch Success Yearly Trend

- There was an upward progression since 2013 though there is a slight drop in 2018



All Launch Site Names

- The use of DISTINCT allows to remove duplicate Launch_site



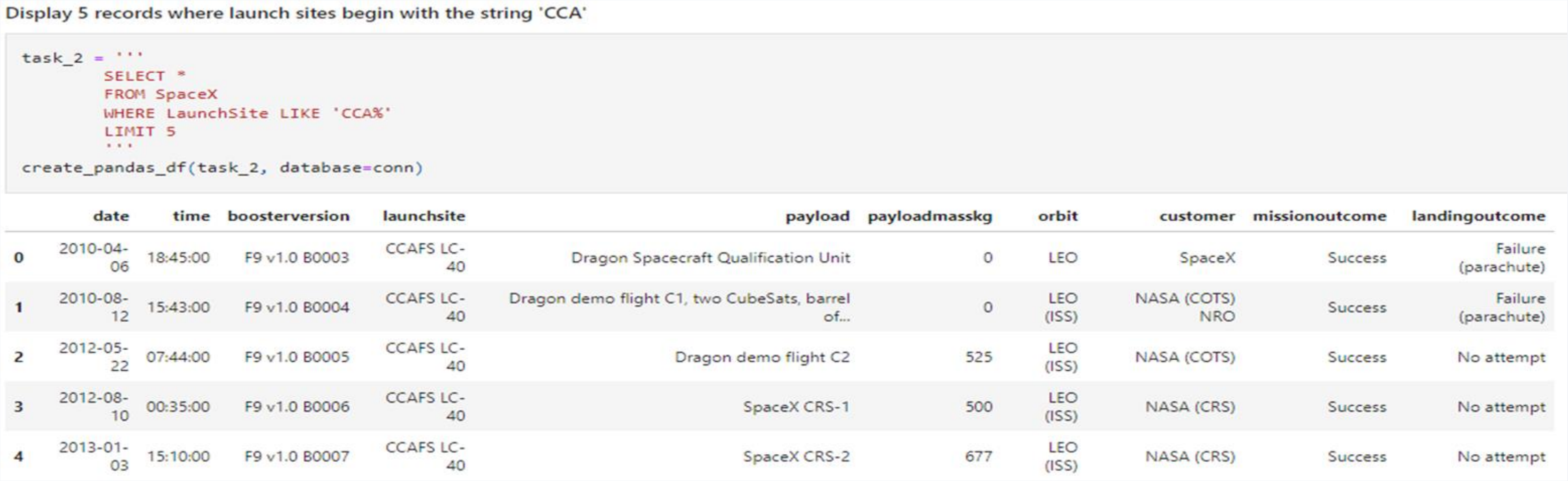
Display the names of the unique launch sites in the space mission

```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- Where clause followed by Like clause filter launch site that contains CCA and it was limited to 5 records using LIMIT

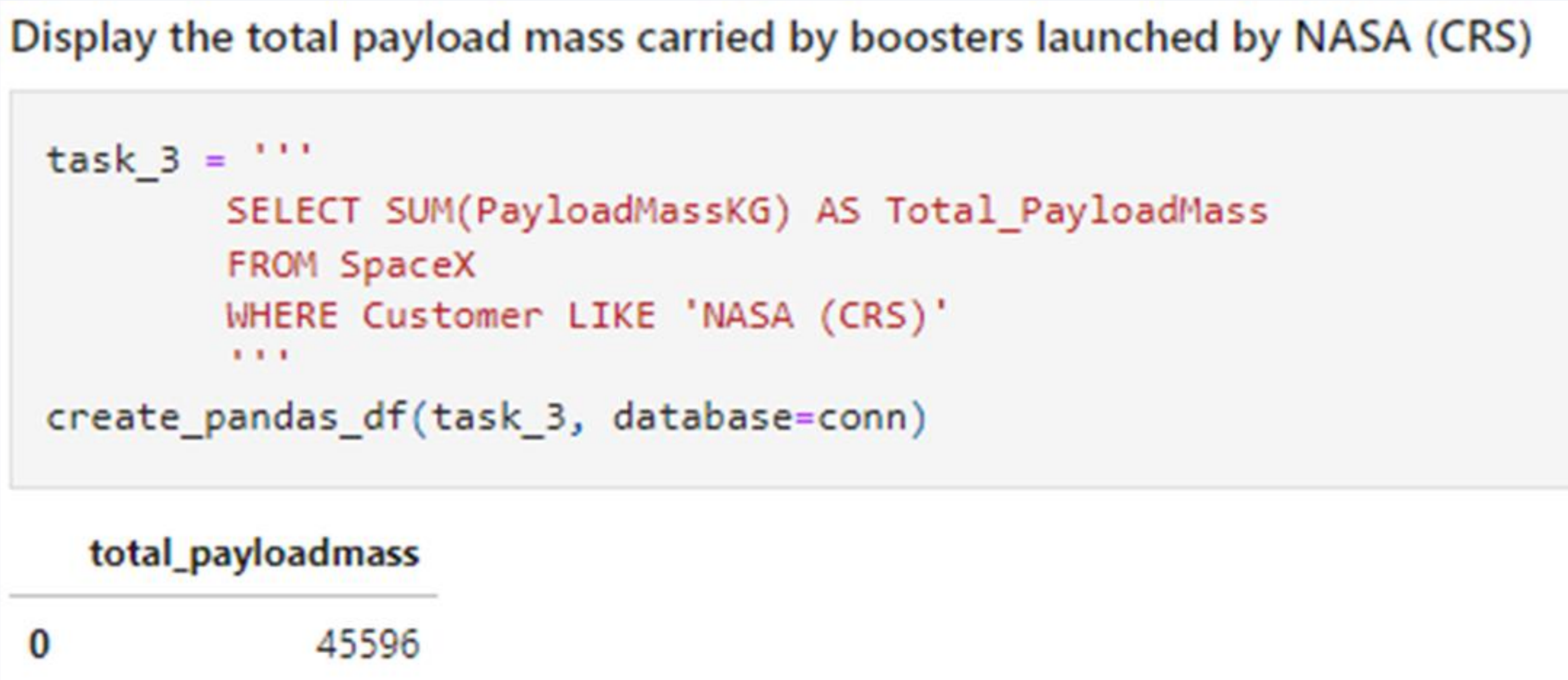


Display 5 records where launch sites begin with the string 'CCA'

```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

	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



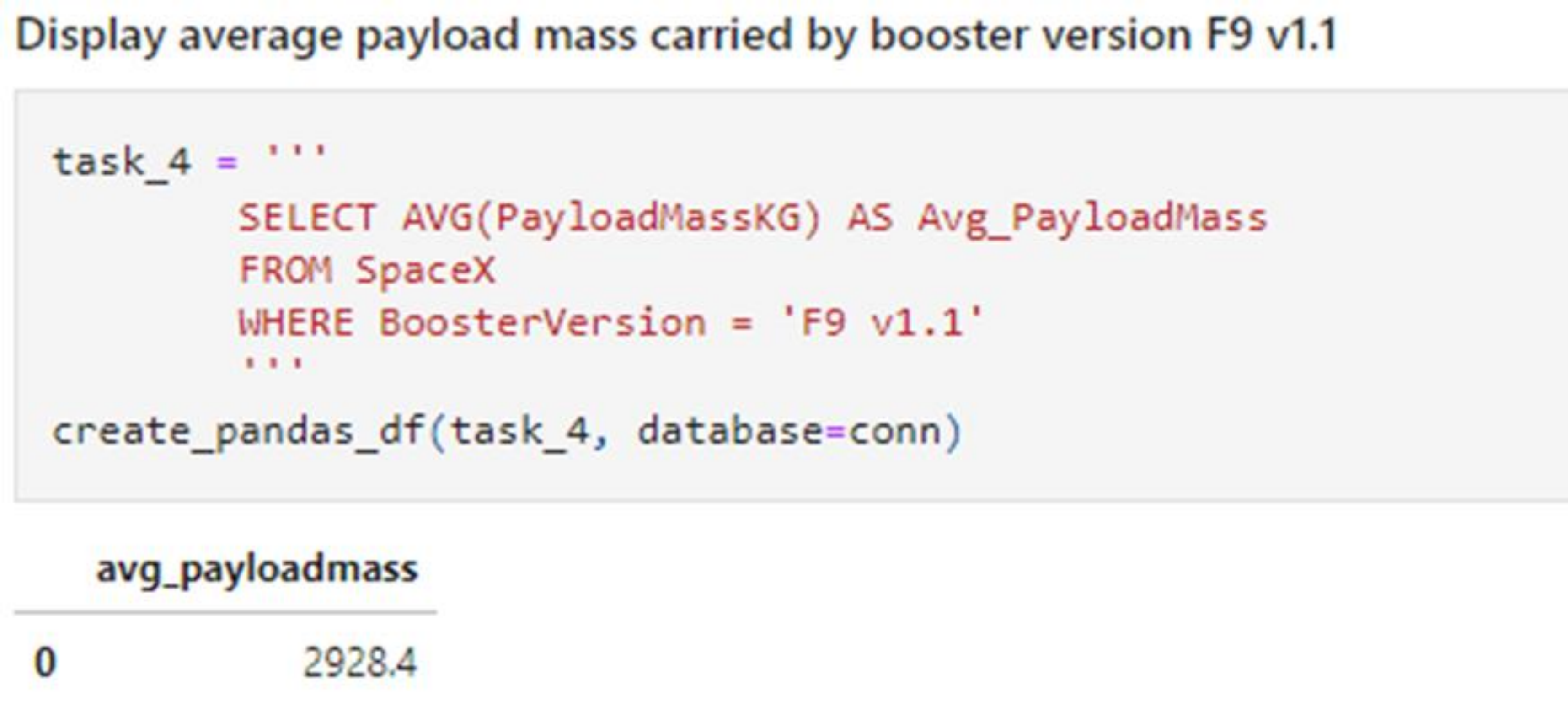
Display the total payload mass carried by boosters launched by NASA (CRS)

```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

```
   total_payloadmass
0              45596
```

Average Payload Mass by F9 v1.1

- The AVG agreegate function was used on the payloadmass columns to get the average



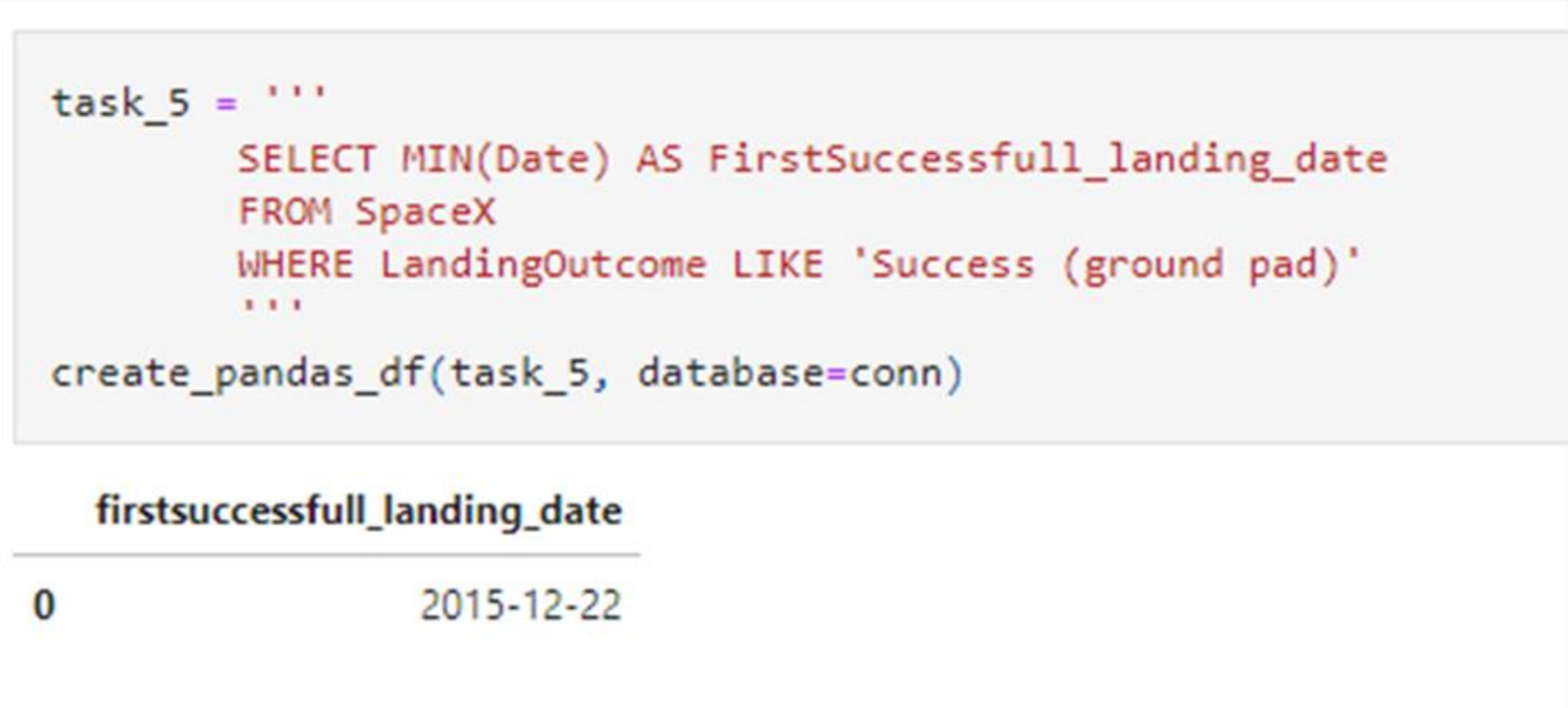
Display average payload mass carried by booster version F9 v1.1

```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- The minimum date was selected and filtered using the where clause on the landing outcome to Success

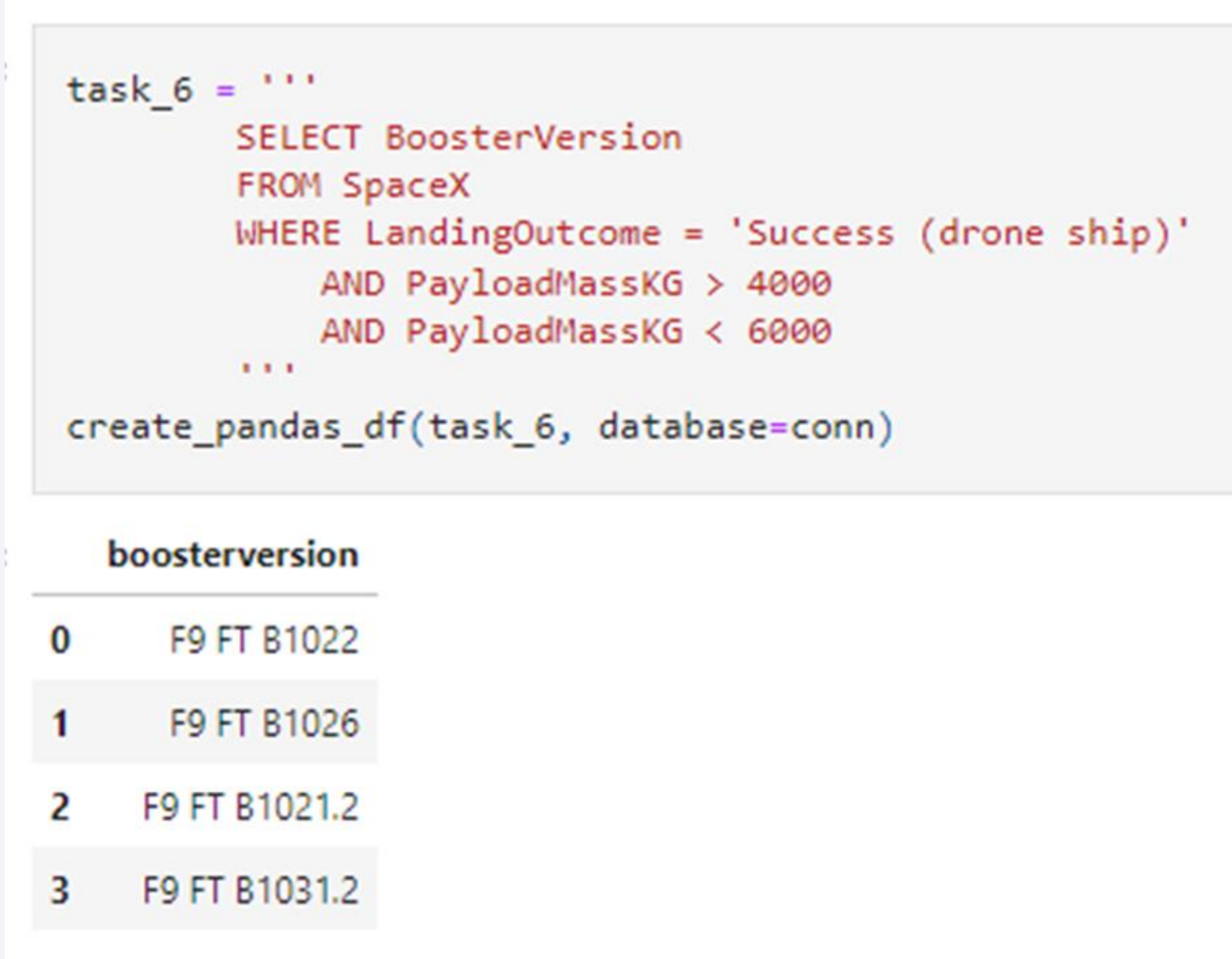


```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

```
   firstsuccessfull_landing_date
0                     2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- The where clause was used to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

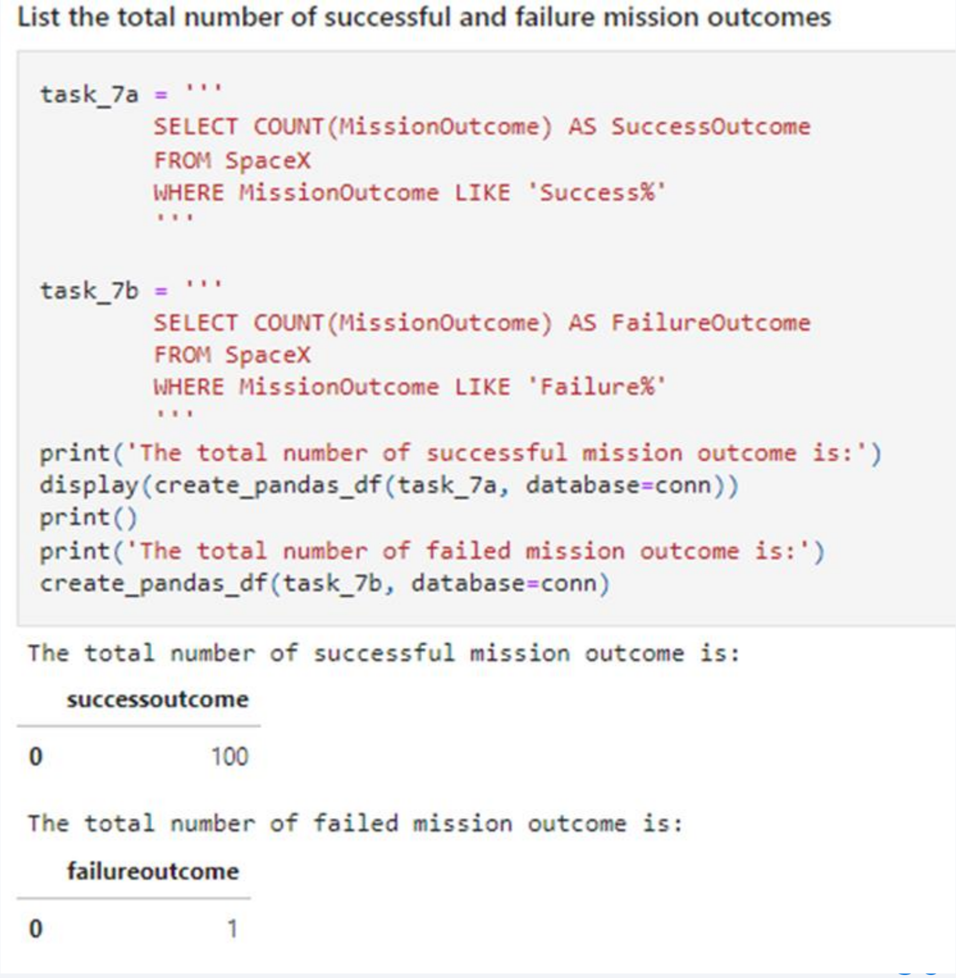


```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- wildcard like ‘%’ was used to filter for **WHERE** MissionOutcome was a success or a failure.



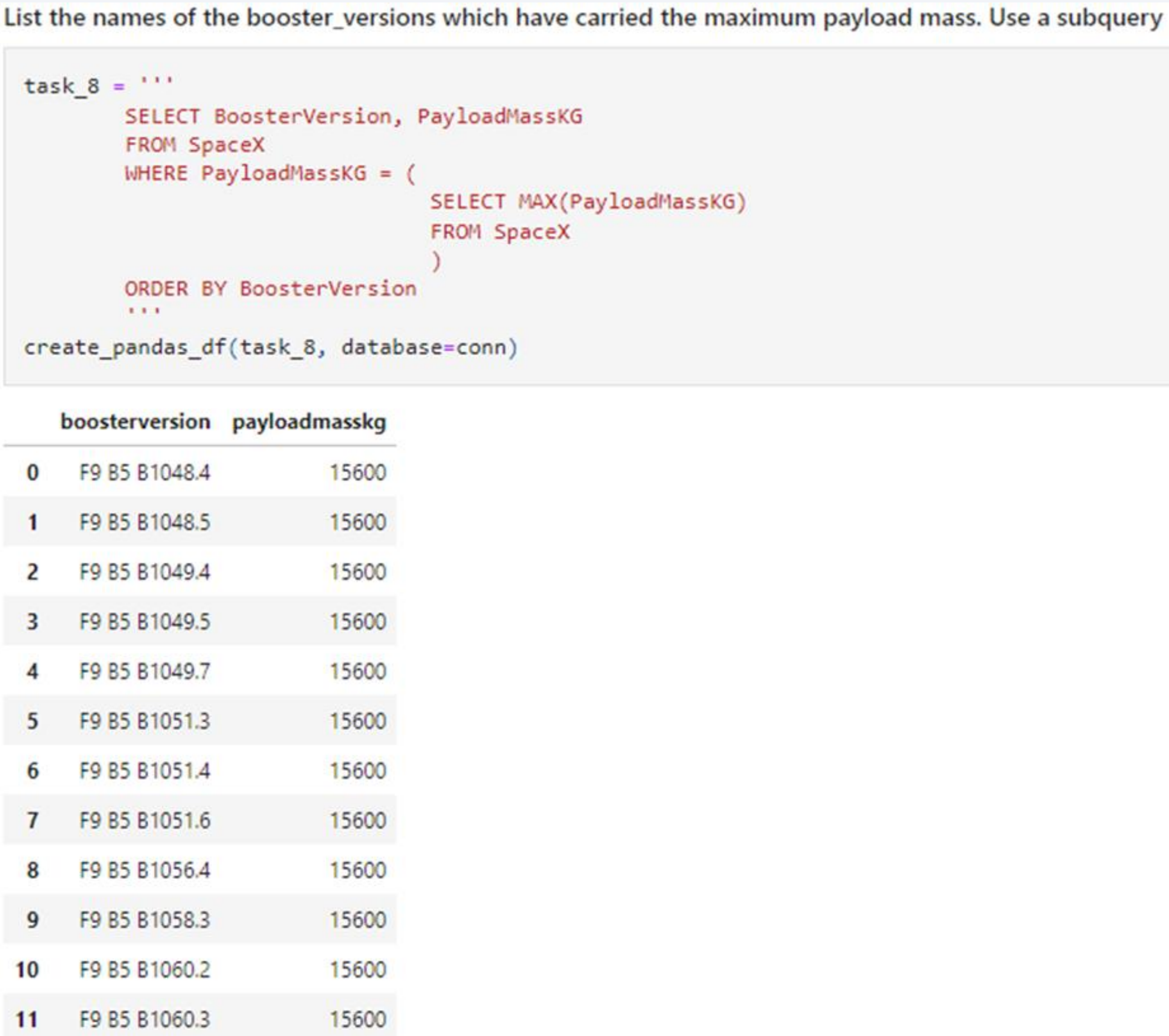
List the total number of successful and failure mission outcomes

```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

```
The total number of successful mission outcome is:
   successoutcome
0             100

The total number of failed mission outcome is:
   failureoutcome
0               1
```

Boosters Carried Maximum Payload

- We determined the booster that have carried the **maximum** payload using a subquery in the **WHERE** clause and the **MAX()** function.



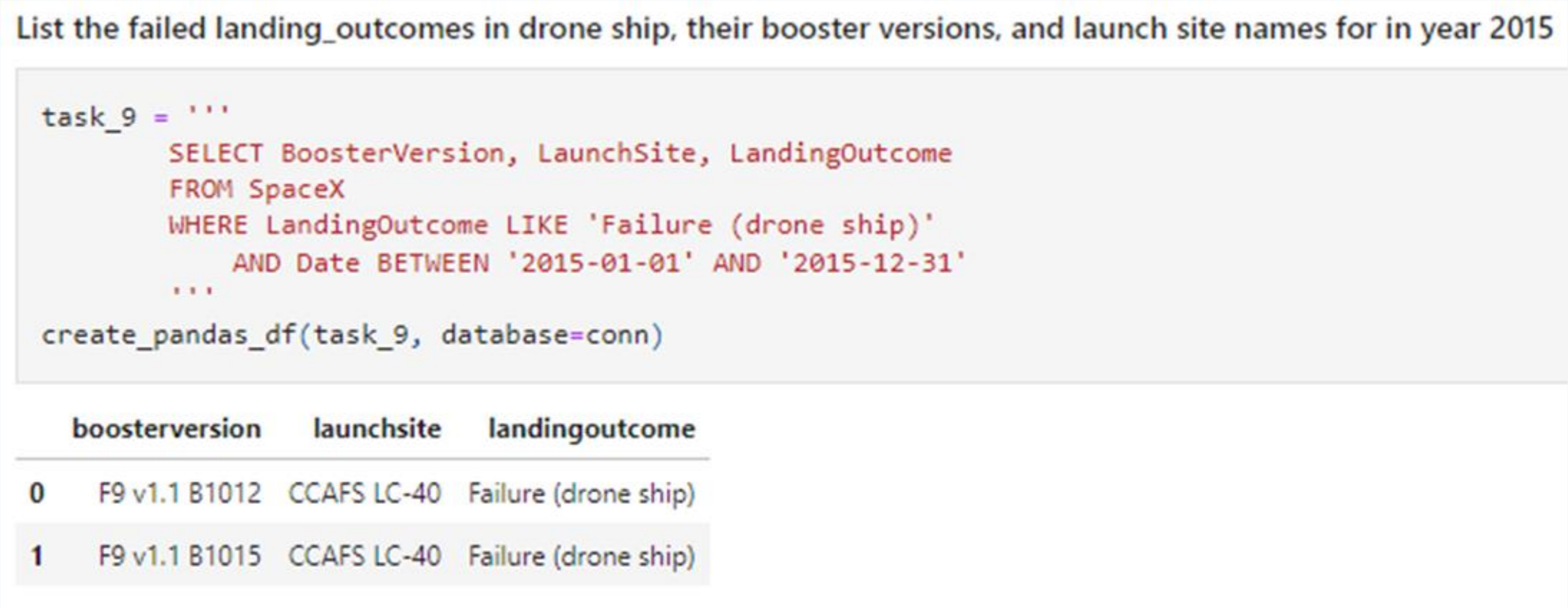
List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

- **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions was used to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015



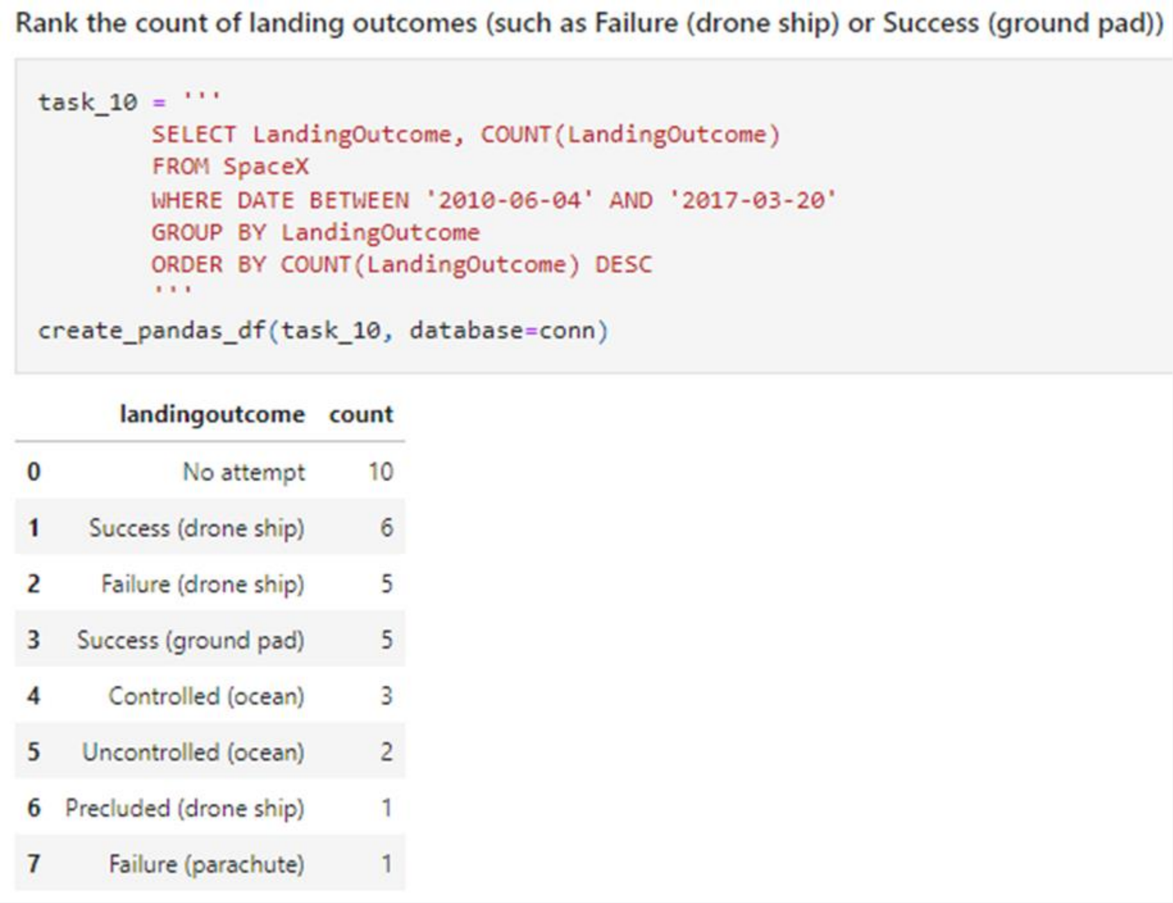
List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order



Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

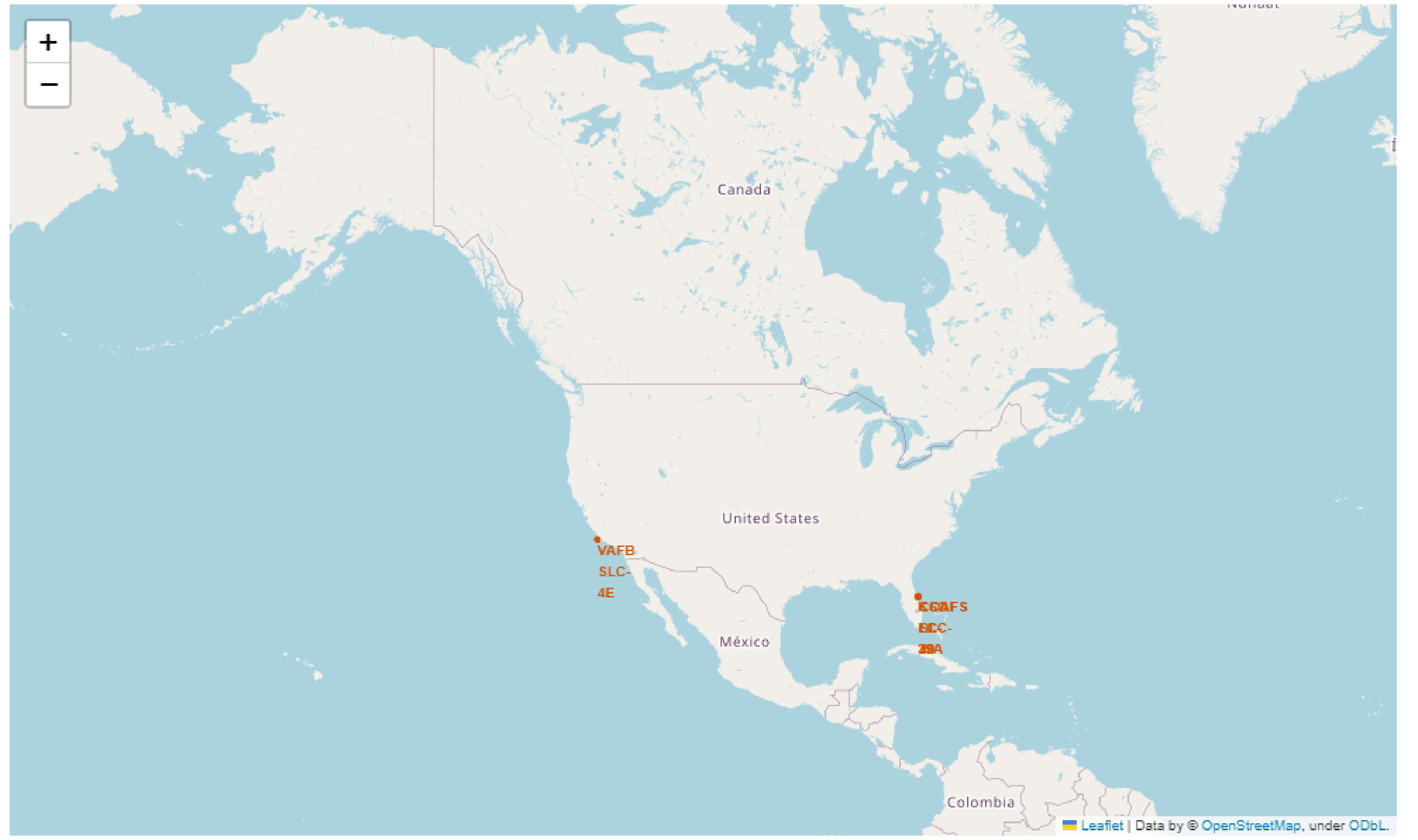
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Section 3

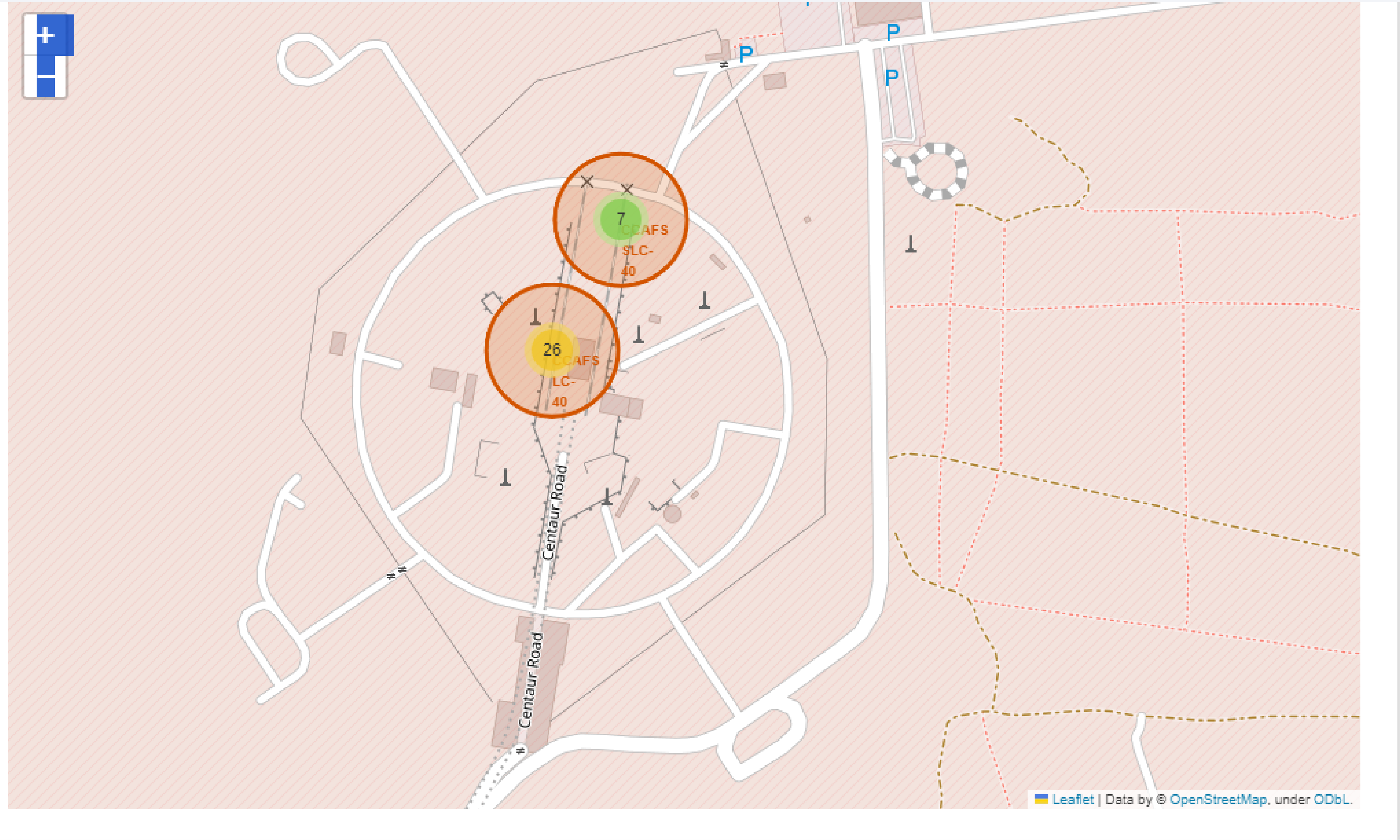
Launch Sites Proximities Analysis

GLOBAL MAP

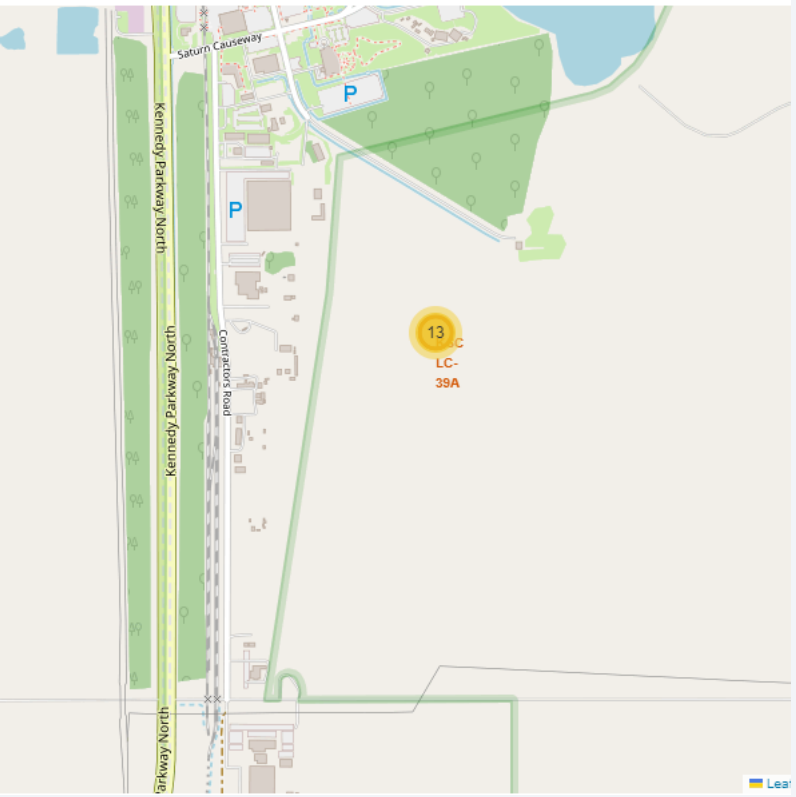
- It shows the map of the space X launch site in the united states

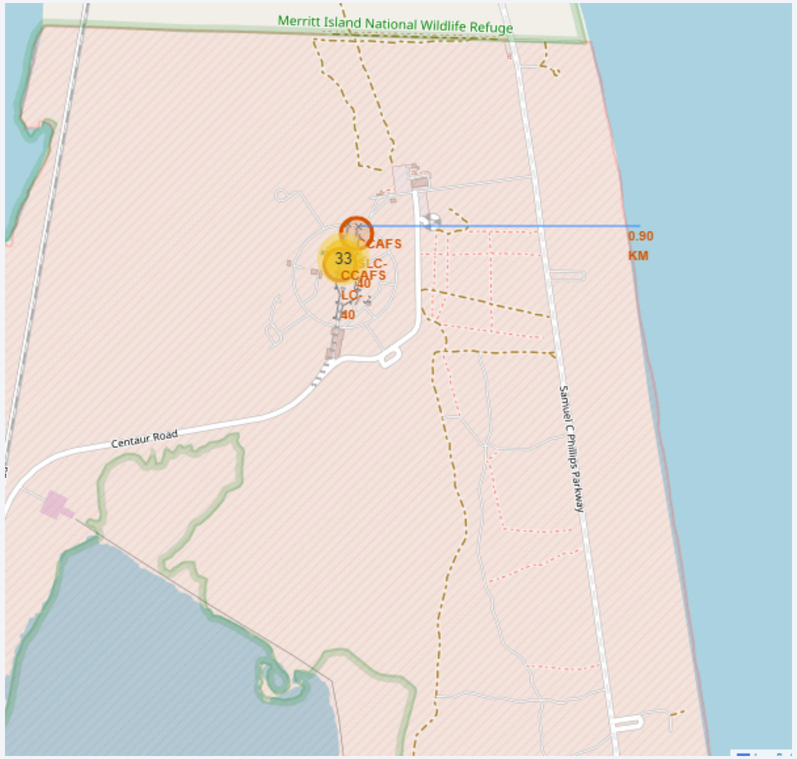


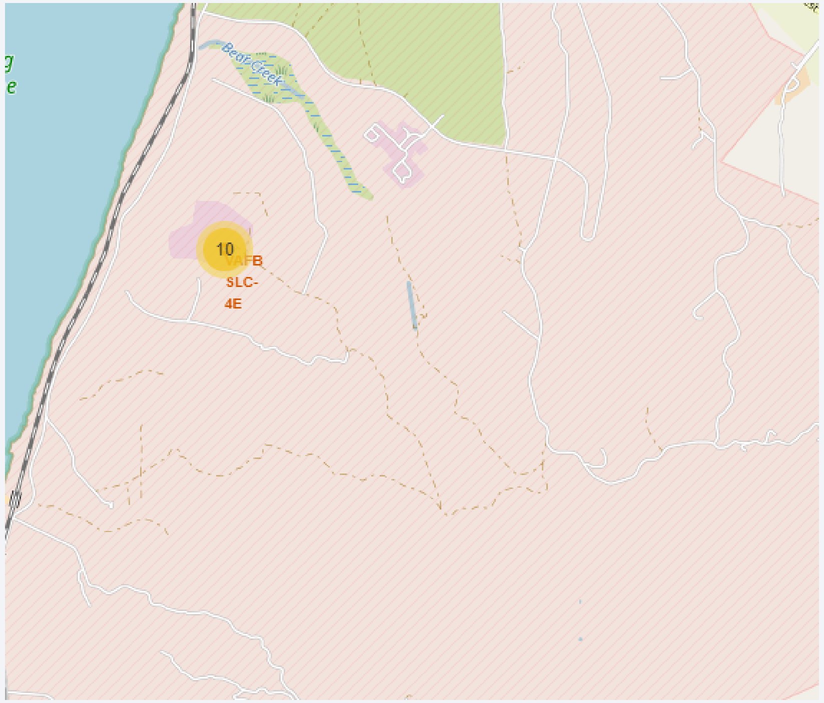
Launch sites with color labels

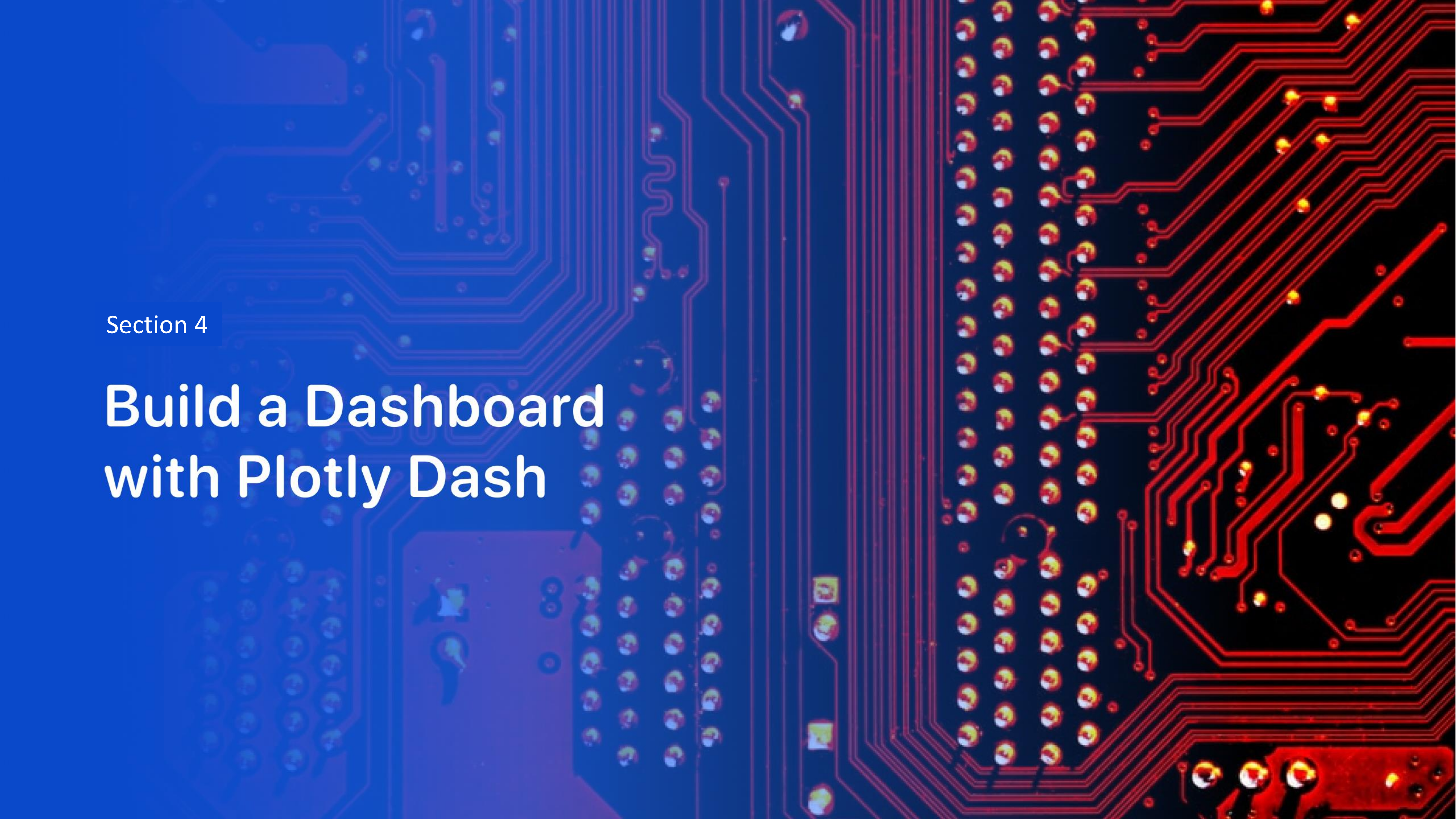


Launchsites showing proximities to landmarks




Merritt Island National Wildlife Refuge
CCAFS
33
SLC-
CCAFS
40
LC-
40
0.90
KM
Centaur Road
Samuel C Phillips Parkway

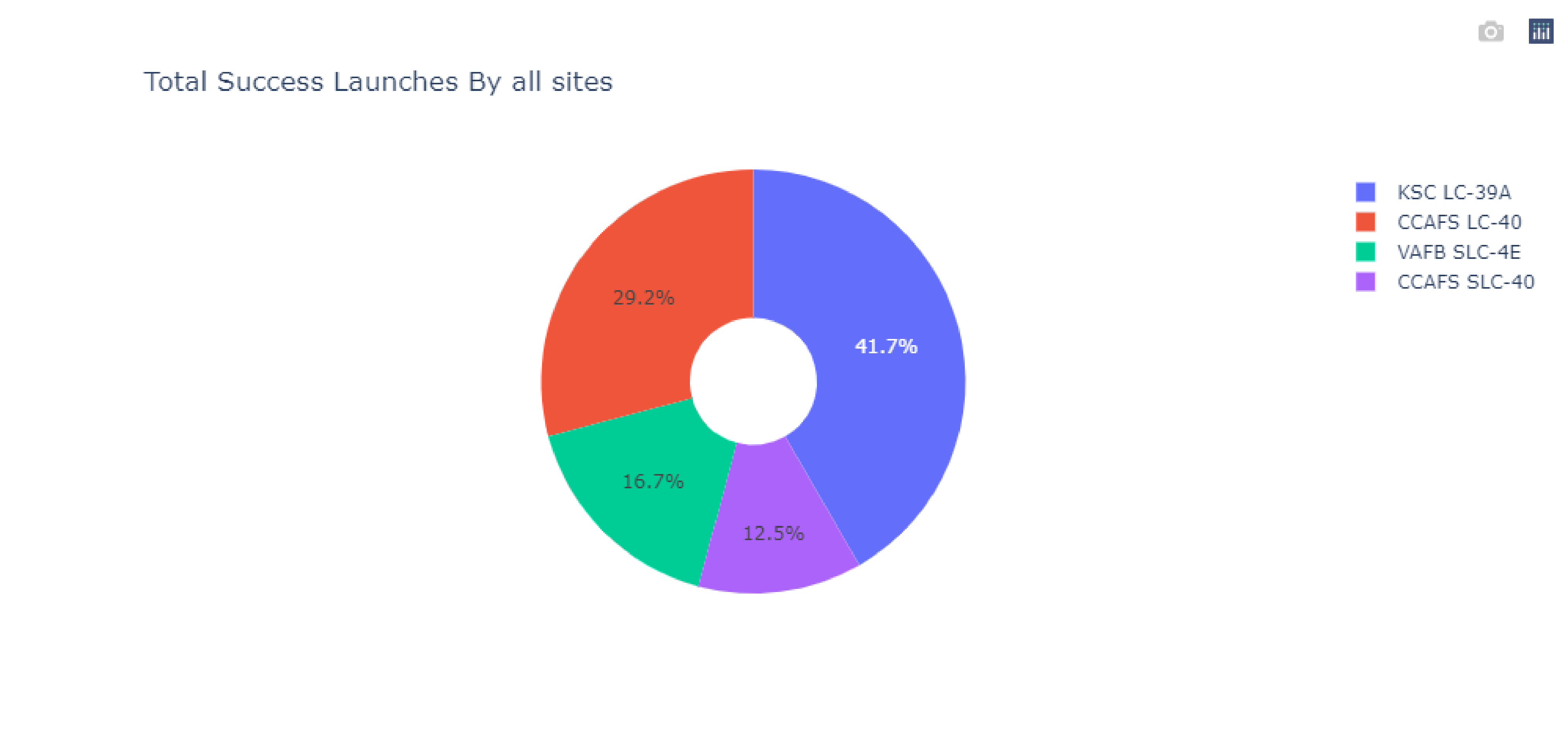




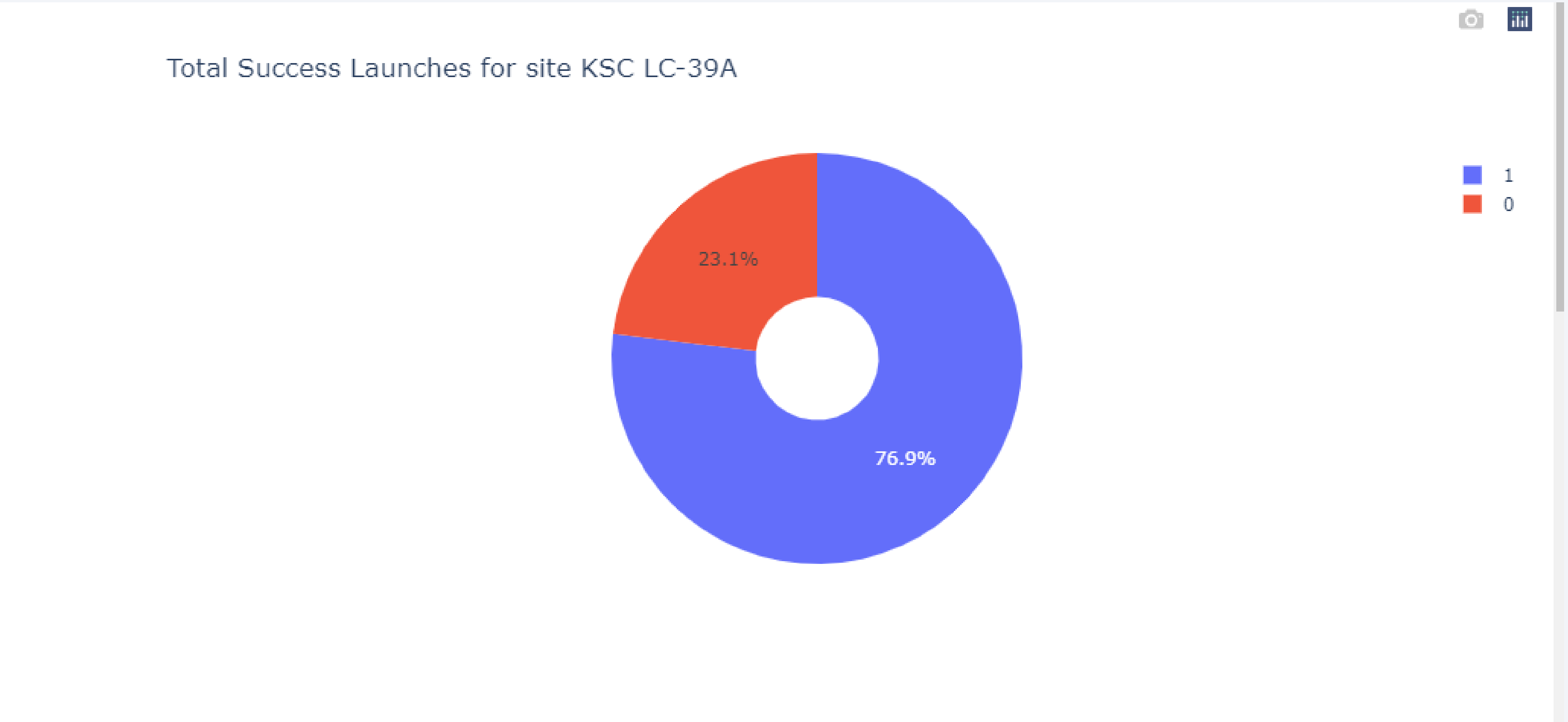
Section 4

Build a Dashboard with Plotly Dash

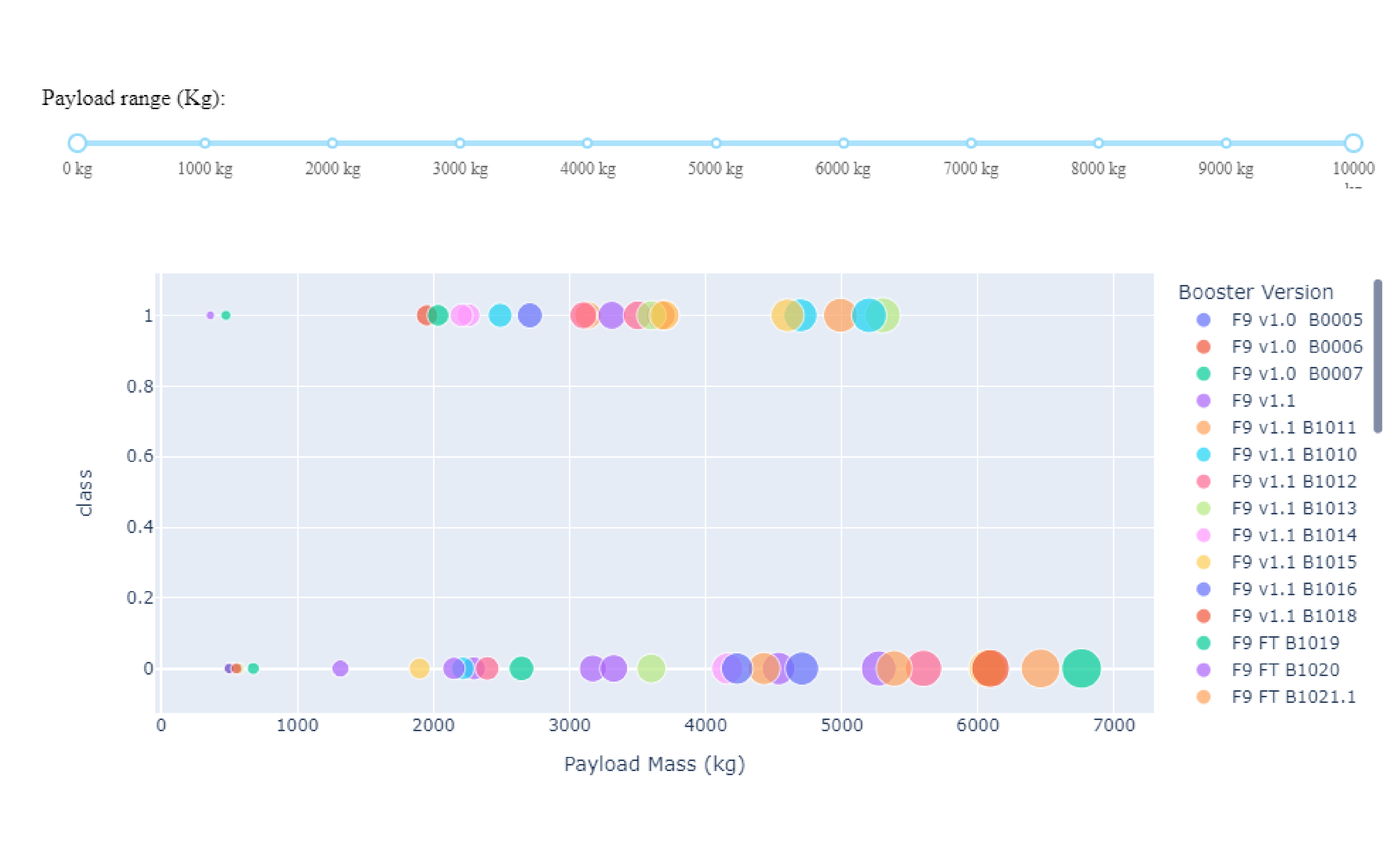
Pie chart showing the success percentage achieved by each launch site



piechart for the launch site with highest launch success ratio



Payload vs. Launch Outcome scatter plot for all sites

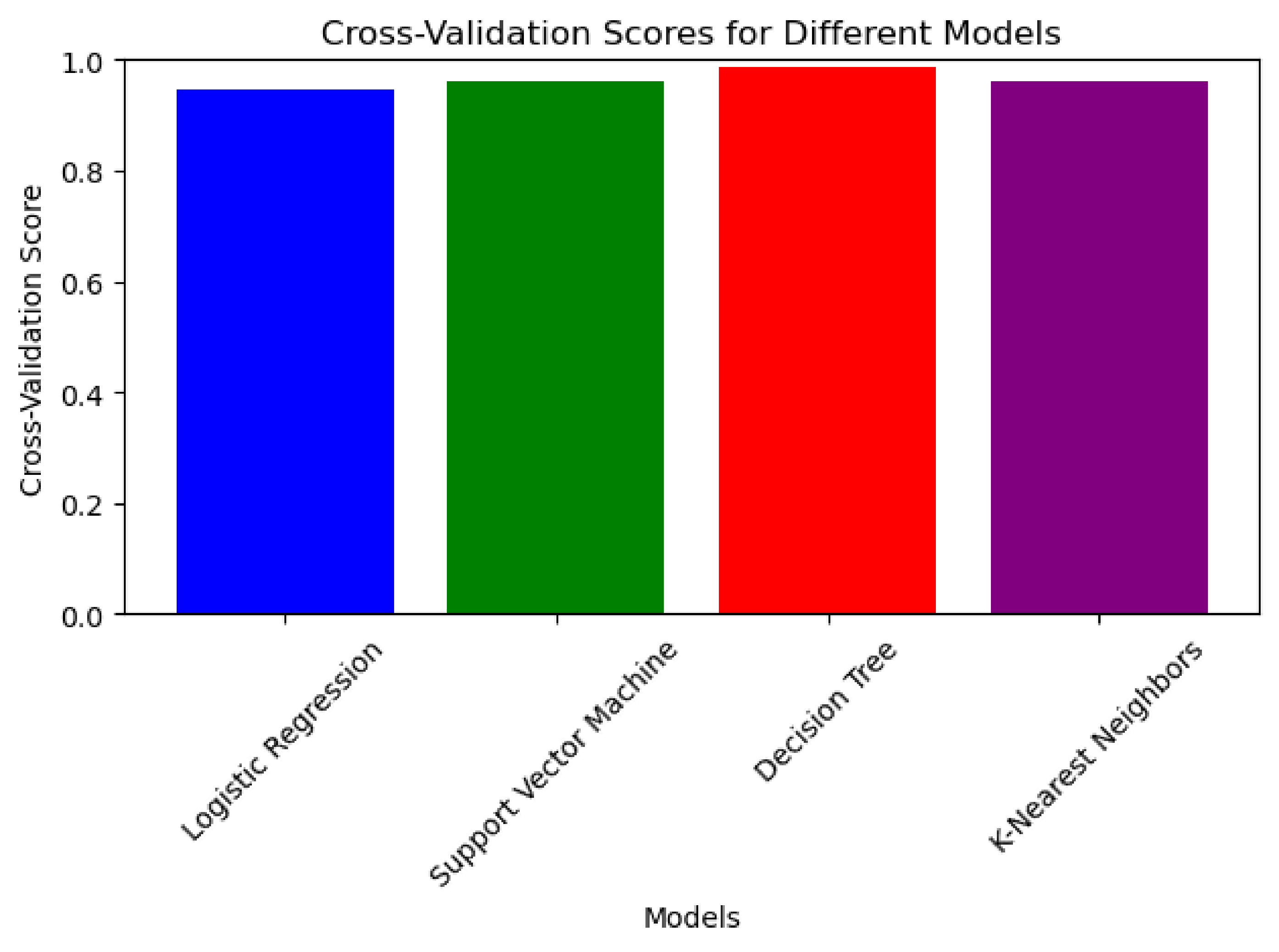


Section 5

Predictive Analysis (Classification)

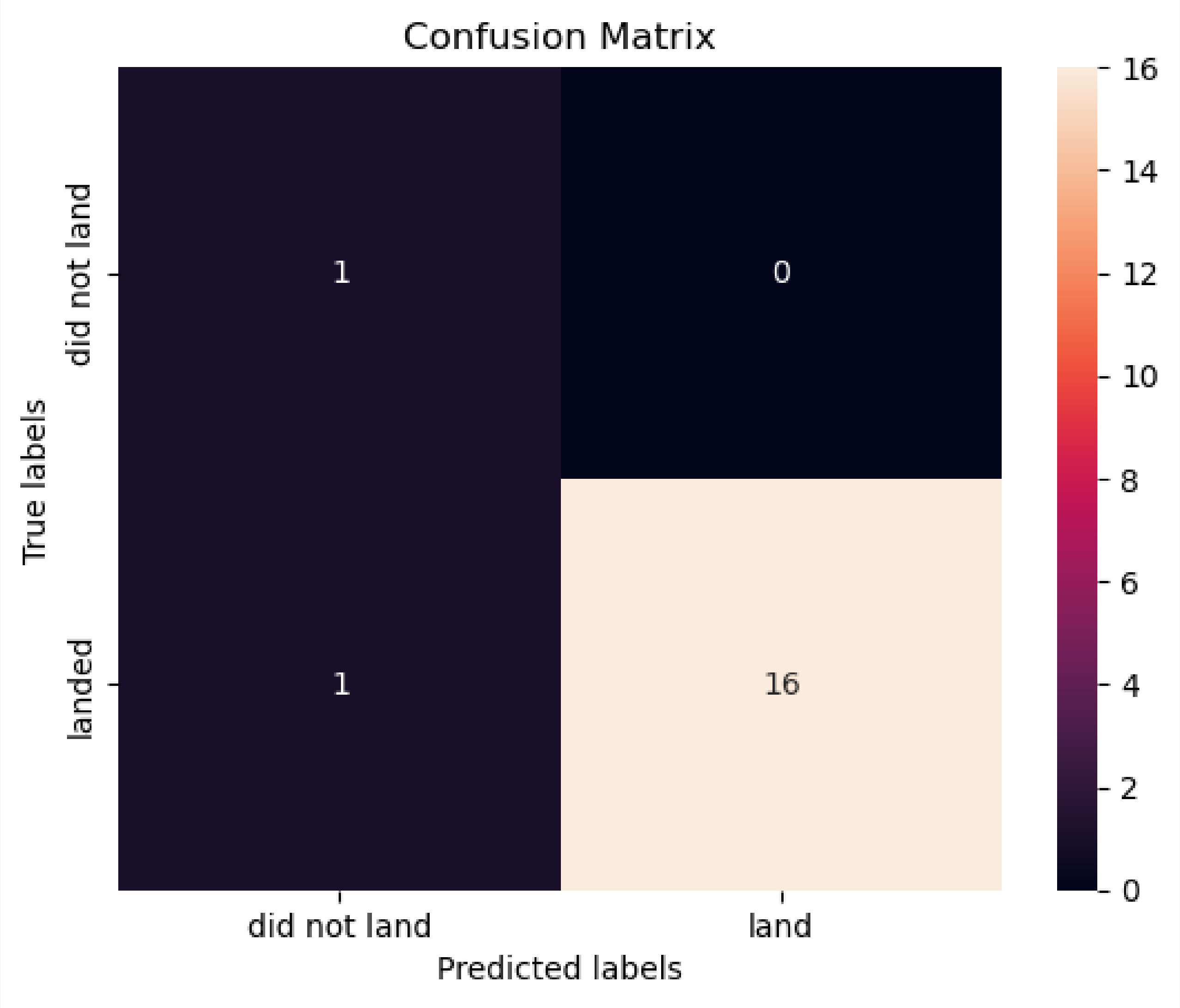
Classification Accuracy

- Decision tree Classifier has the highest accuracy of 0.986



Confusion Matrix

- It has a true negative of 1 and True Positive of 16



Conclusions

- KSC LX-39A had the most successful launches than any other launching sites
- There is a continuous progression in the launching success since 2013 to 2020
- From the scatterplot, we can see that the larger the flight amount at a launch site, the greater the success rate.
- From the bar chart, Orbit ES-L1, GEO, HEO, SSO, and VLEO had the most success launching rate
- The Decision tree classifier is the best algorithm with the accuracy of 0.986

Thank you!