IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Kyaw Zin Oo
22 August 2025

Outline

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

Introduction

- Project background and context

  SpaceX advertises Falcon 9 rocket launches at a cost of $62 million, compared to over $165 million charged by other providers. The significant cost advantage comes from SpaceX's ability to reuse the first stage of the rocket. Therefore, predicting whether the first stage will land successfully is key to estimating launch costs. This information is also valuable for competitors seeking to bid against SpaceX. The objective of this project is to develop a machine learning pipeline that can predict the likelihood of a successful first-stage landing.

- 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:
    - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
    - One-hot encoding was applied to categorical features
- 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

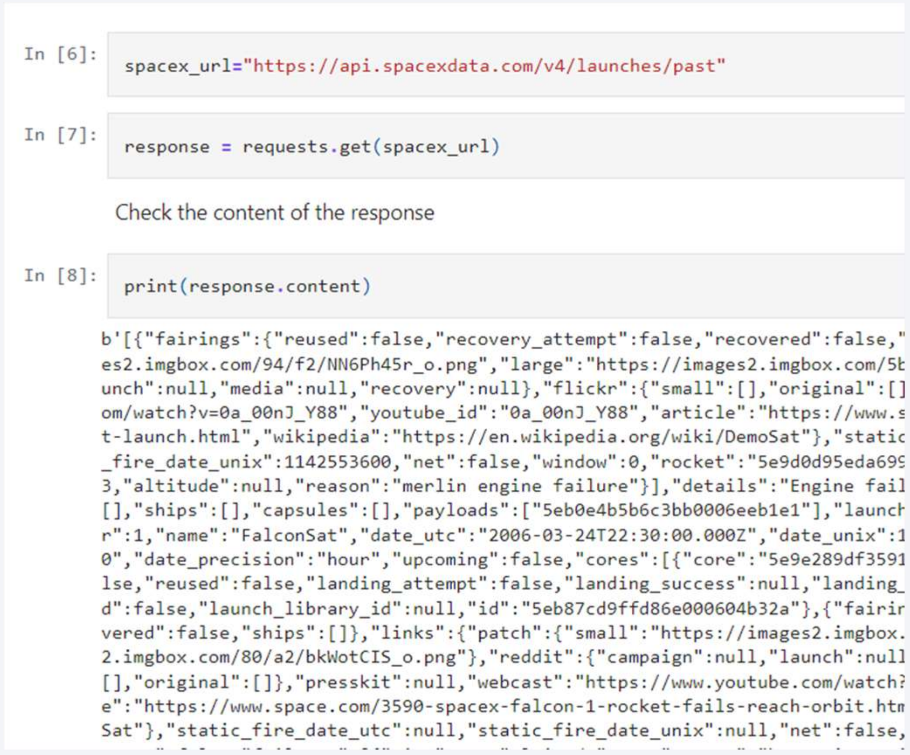
- Describe how data sets were collected.

  Data was collected through a GET request to the SpaceX API. The response was decoded into JSON format using the .json() function and converted into a Pandas DataFrame with .json_normalize(). The dataset was then cleaned, checked for missing values, and corrected where necessary.

  In addition, web scraping was performed on Wikipedia using BeautifulSoup to gather Falcon 9 launch records. The launch data was extracted from the HTML tables, parsed, and converted into a Pandas DataFrame for further analysis.

Data Collection – SpaceX API

- We collected data using a GET request to the SpaceX API, then cleaned, wrangled, and formatted the dataset to prepare it for analysis.
- The link to the notebook is https://github.com/kyawzo/test_repo/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



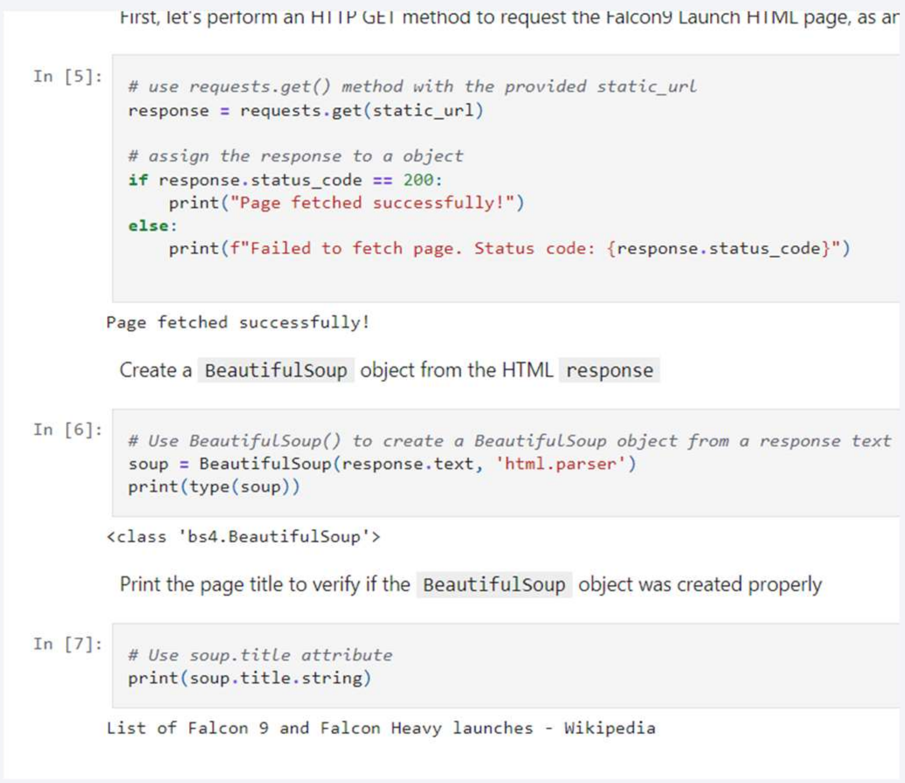
```
In [6]:  spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]:  response = requests.get(spacex_url)
```

Check the content of the response

```
In [8]:  print(response.content)

b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"
es2.imgbox.com/94/f2/NN6Ph45r_o.png","large":"https://images2.imgbox.com/5b
unch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]
om/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.s
t-launch.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static
_fire_date_unix":1142553600,"net":false,"window":0,"rocket":"5e9d0d95eda699
3,"altitude":null,"reason":"merlin engine failure"}],"details":"Engine fail
[],"ships":[],"capsules":[],"payloads":["5eb0e4b5b6c3bb0006eeb1e1"],"launch
r":1,"name":"FalconSat","date_utc":"2006-03-24T22:30:00.000Z","date_unix":1
0","date_precision":"hour","upcoming":false,"cores":[{"core":"5e9e289df3591
lse,"reused":false,"landing_attempt":false,"landing_success":null,"landing_
d":false,"launch_library_id":null,"id":"5eb87cd9ffd86e000604b32a"},{"fairin
vered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.
2.imgbox.com/80/a2/bkWotCIS_o.png"},"reddit":{"campaign":null,"launch":null
[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?
e":"https://www.space.com/3590-spacex-falcon-1-rocket-fails-reach-orbit.htm
Sat"},"static_fire_date_utc":null,"static_fire_date_unix":null,"net":false,
```

Data Collection - Scraping

- We used web scraping with BeautifulSoup to extract Falcon 9 launch records, then parsed the table data and converted it into a pandas dataframe for analysis.
- The link to the notebook is https://github.com/kyawzo/test_repo/blob/main/jupyter-labs-webscraping.ipynb



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

```
In [5]:  # use requests.get() method with the provided static_url
         response = requests.get(static_url)

         # assign the response to a object
         if response.status_code == 200:
             print("Page fetched successfully!")
         else:
             print(f"Failed to fetch page. Status code: {response.status_code}")

         Page fetched successfully!
```

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

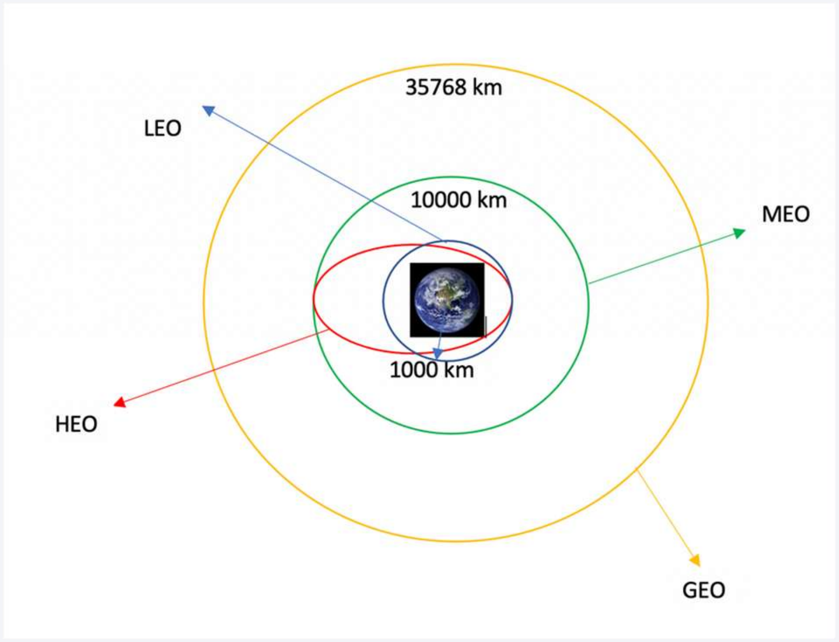
```
In [6]:  # Use BeautifulSoup() to create a BeautifulSoup object from a response text
         soup = BeautifulSoup(response.text, 'html.parser')
         print(type(soup))

         <class 'bs4.BeautifulSoup'>
```

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

```
In [7]:  # Use soup.title attribute
         print(soup.title.string)

         List of Falcon 9 and Falcon Heavy launches - Wikipedia
```

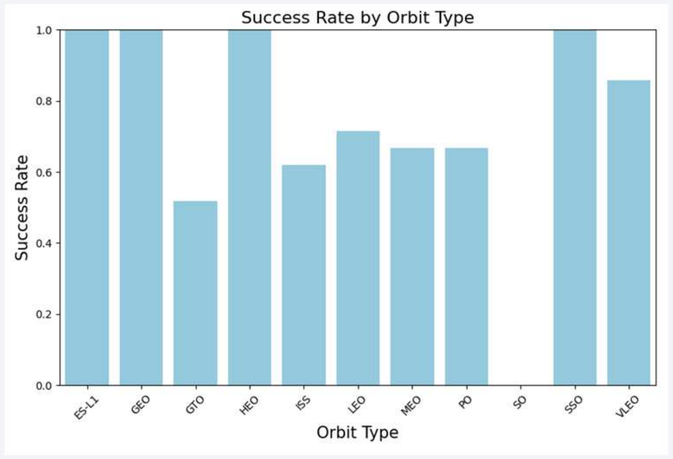
Data Wrangling

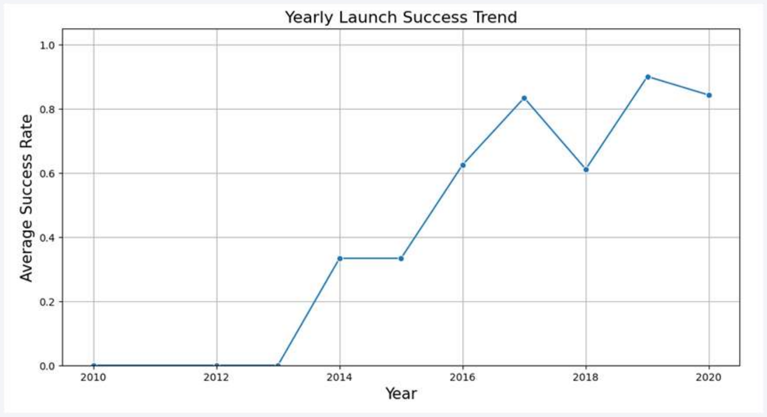
- We conducted exploratory data analysis (EDA) to define the training labels. This included calculating the number of launches at each site and analyzing the frequency and distribution of different orbits.
- We also created a landing outcome label from the outcome column and exported the processed results to a CSV file.
- The link to the notebook is https://github.com/kyawzo/test_repo/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- We analyzed the data through visualizations to examine the relationship between flight number and launch site, payload and launch site, the success rate of each orbit type, the association between flight number and orbit type, and the yearly trend of launch success.
- The link to the notebook is https://github.com/kyawzo/test_repo/blob/main/edadataviz.ipynb





EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database directly from the Jupyter notebook. Using SQL for exploratory data analysis (EDA), we generated insights by writing queries to answer questions such as:
  - What are the names of the unique launch sites in the missions?
  - What is the total payload mass carried by boosters launched under NASA (CRS)?
  - What is the average payload mass carried by the booster version F9 v1.1?
  - How many missions resulted in success and failure outcomes?
  - Which landings failed on drone ships, and what were their corresponding booster versions and launch sites?
- The link to the notebook is https://github.com/kyawzo/test_repo/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We visualized all launch sites on a Folium map, enhancing it with markers, circles, and lines to indicate the success or failure of launches at each site.
- Launch outcomes were classified as 0 (failure) or 1 (success). By using color-coded marker clusters, we were able to identify which launch sites demonstrated higher success rates.
- Additionally, we calculated distances from each launch site to nearby infrastructure and natural features. This allowed us to address questions such as:
    - Are the launch sites located near railways, highways, or coastlines?
    - Do launch sites maintain a certain distance from cities?
- The link to the notebook is https://github.com/kyawzo/test_repo/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- We developed an interactive dashboard using Plotly Dash. The dashboard includes:
  - Pie charts illustrating the total number of launches at each site.
  - Scatter plots visualizing the relationship between launch outcomes and payload mass (kg) across different booster versions.
- The link to the notebook is https://github.com/kyawzo/test_repo/blob/main/spacex-dash-app.py

Predictive Analysis (Classification)

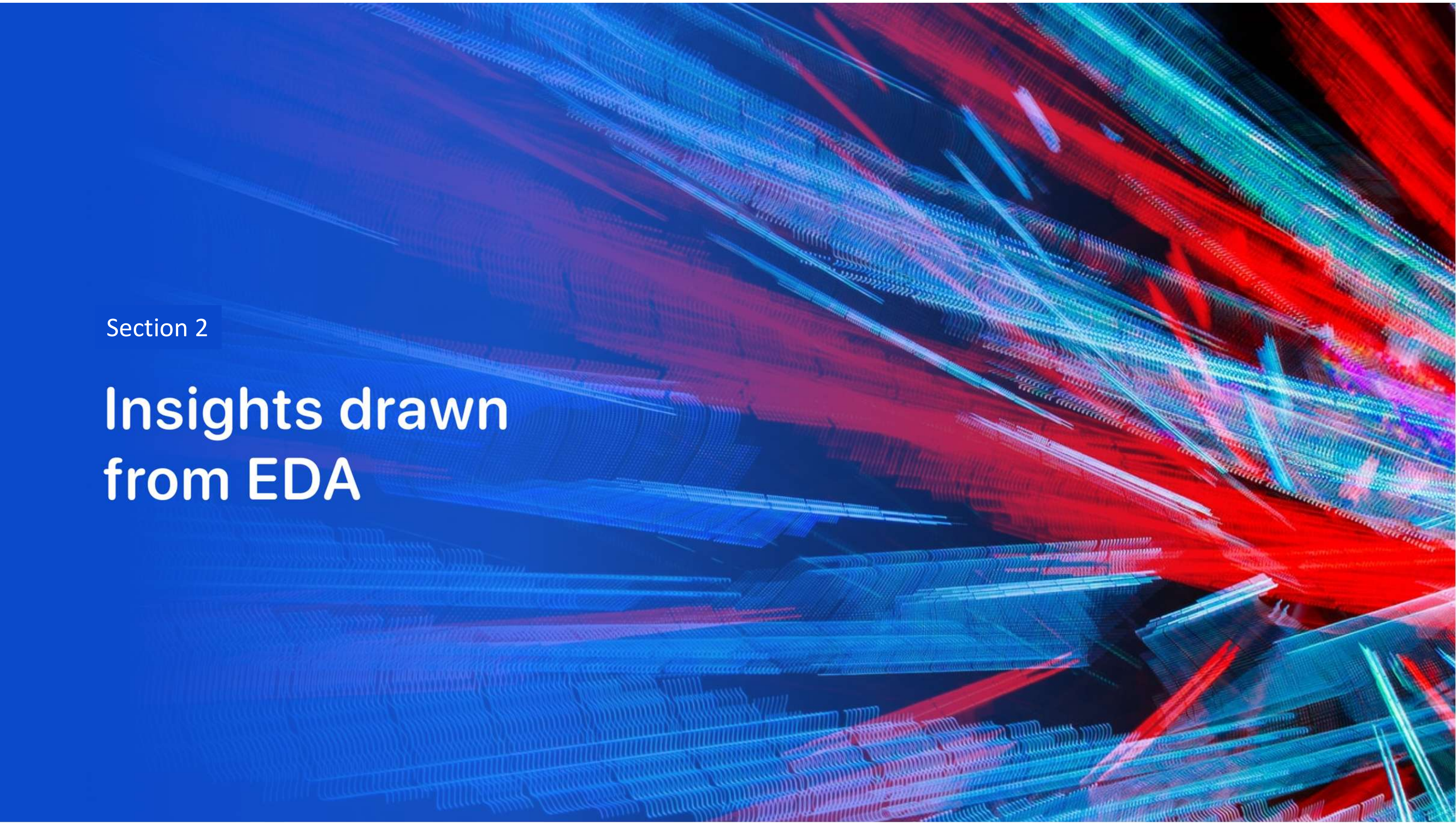
- We loaded the data using NumPy and Pandas, performed data transformations, and split it into training and testing sets.
- We built multiple machine learning models and tuned their hyperparameters using GridSearchCV.
- Model performance was evaluated using accuracy, and we further improved results through feature engineering and algorithm optimization.
- Finally, we identified the best-performing classification model.
- The link to the notebook is https://github.com/kyawzo/test_repo/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

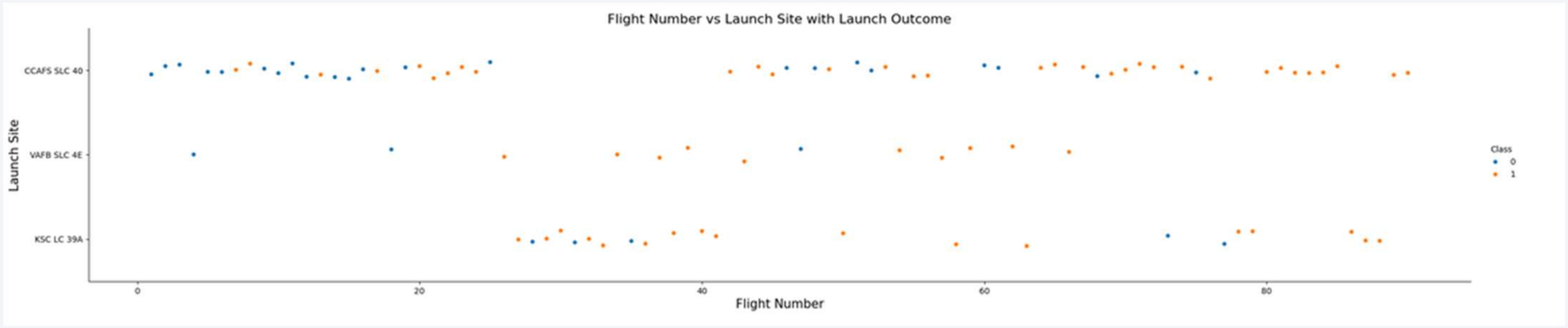
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

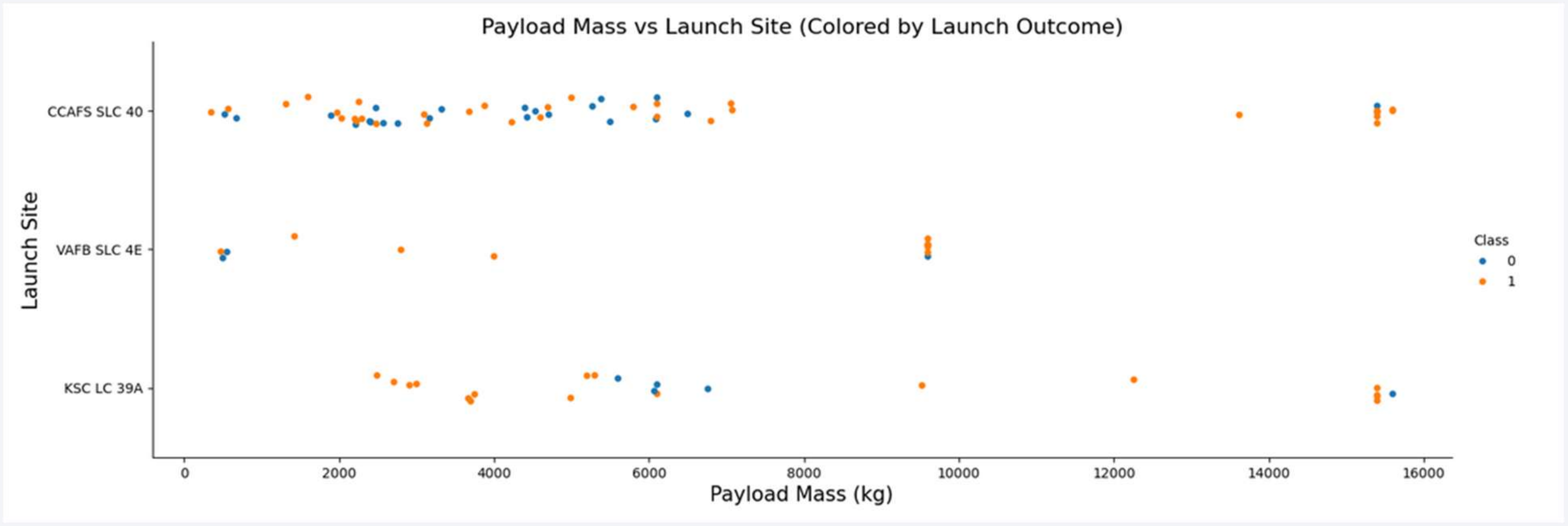


Flight Number vs. Launch Site



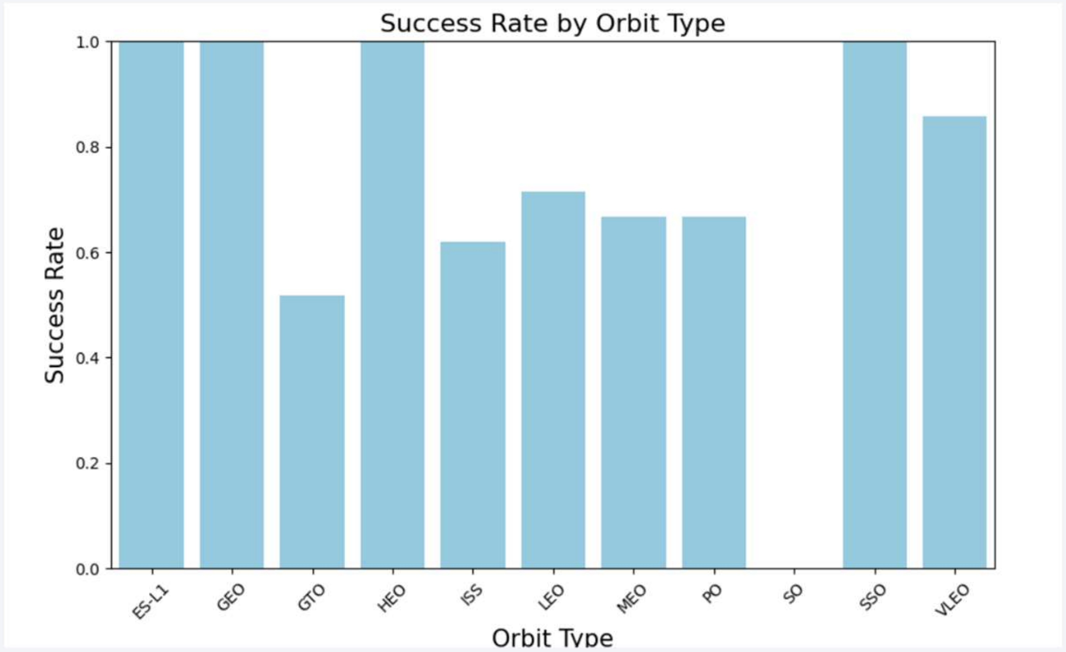
- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.

Payload vs. Launch Site



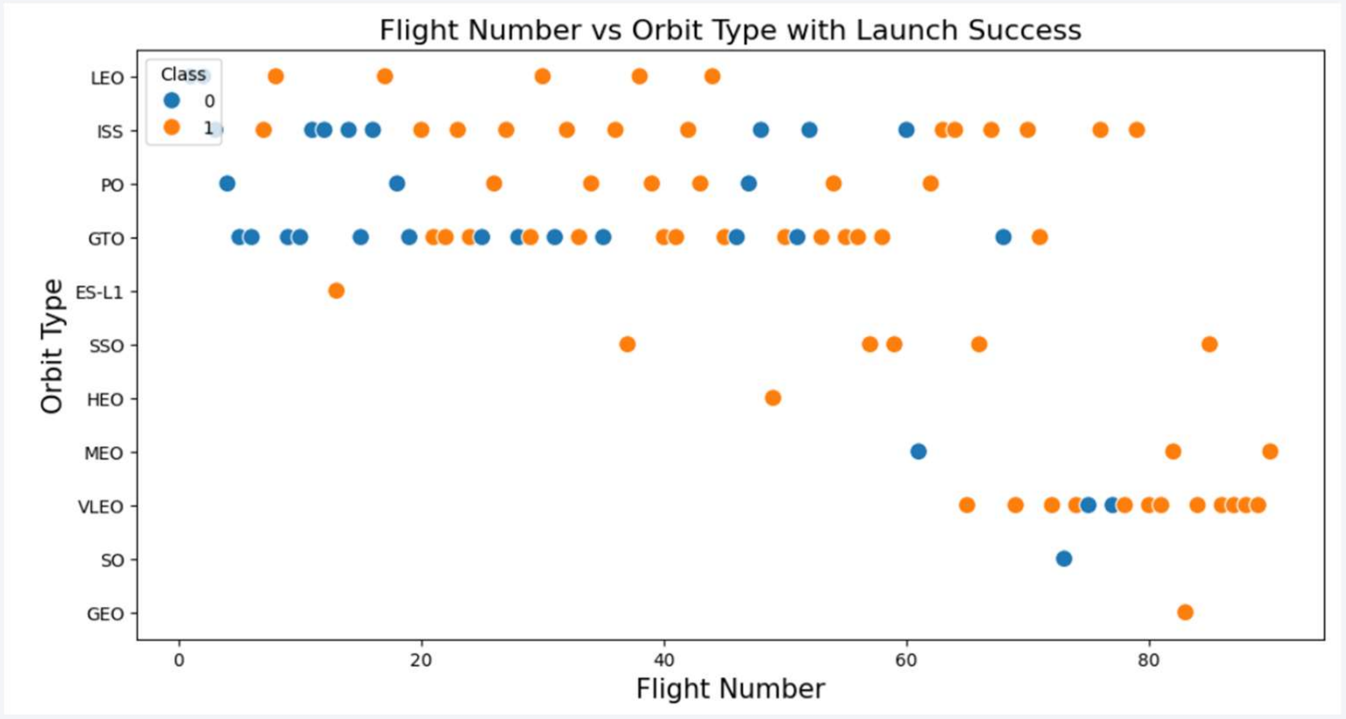
- The plot shows launches from three different sites: CCAFS SLC 40, VAFB SLC 4E, and KSC LC 39A.
- The majority of the launches appear to be successful (blue points) across all three sites.
- The plot reveals that failed launches (orange points) occurred at various payload masses from all three launch sites.

Success Rate vs. Orbit Type



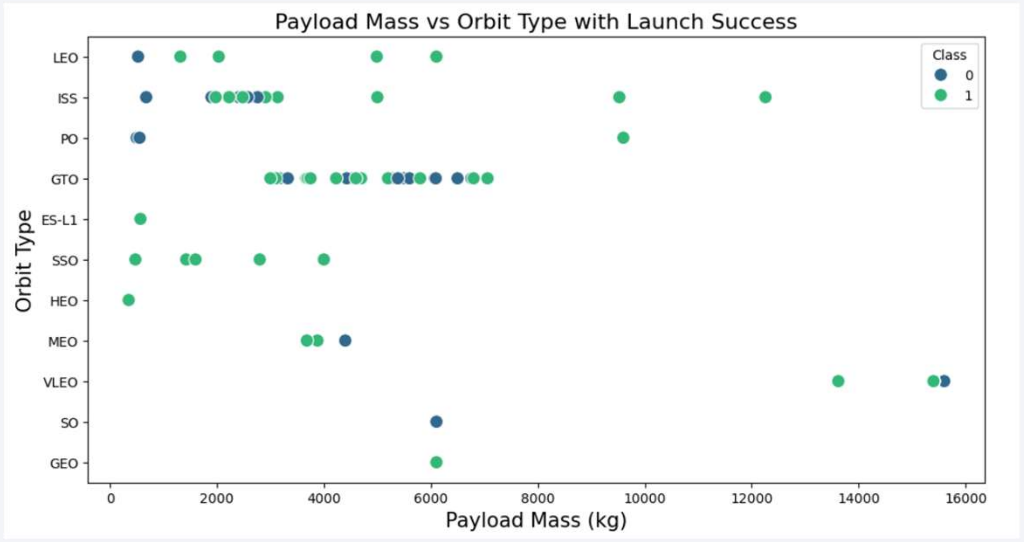
- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.

Flight Number vs. Orbit Type



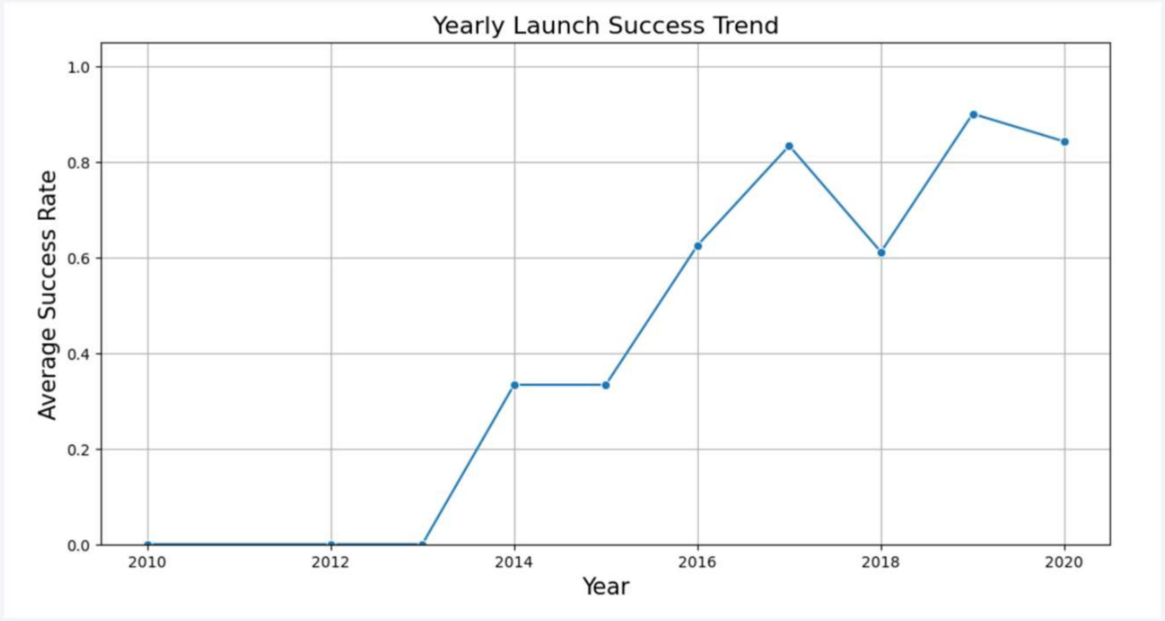
- The plot below shows the Flight Number vs. Orbit type.
- We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

Payload vs. Orbit Type



- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.

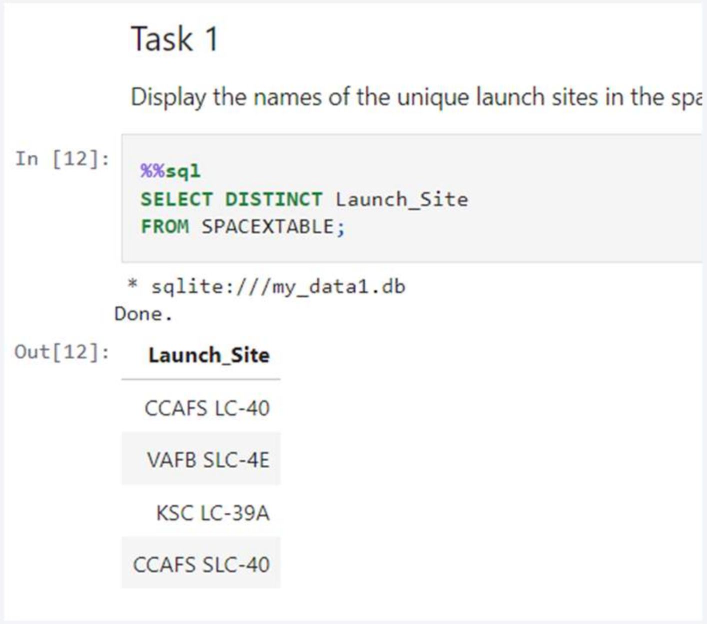
Launch Success Yearly Trend



- We can observe that the success rate since 2013 kept increasing till 2020

All Launch Site Names

- We used the key word DISTINCT to extract the unique launch sites from the SpaceX data.



Task 1

Display the names of the unique launch sites in the spa

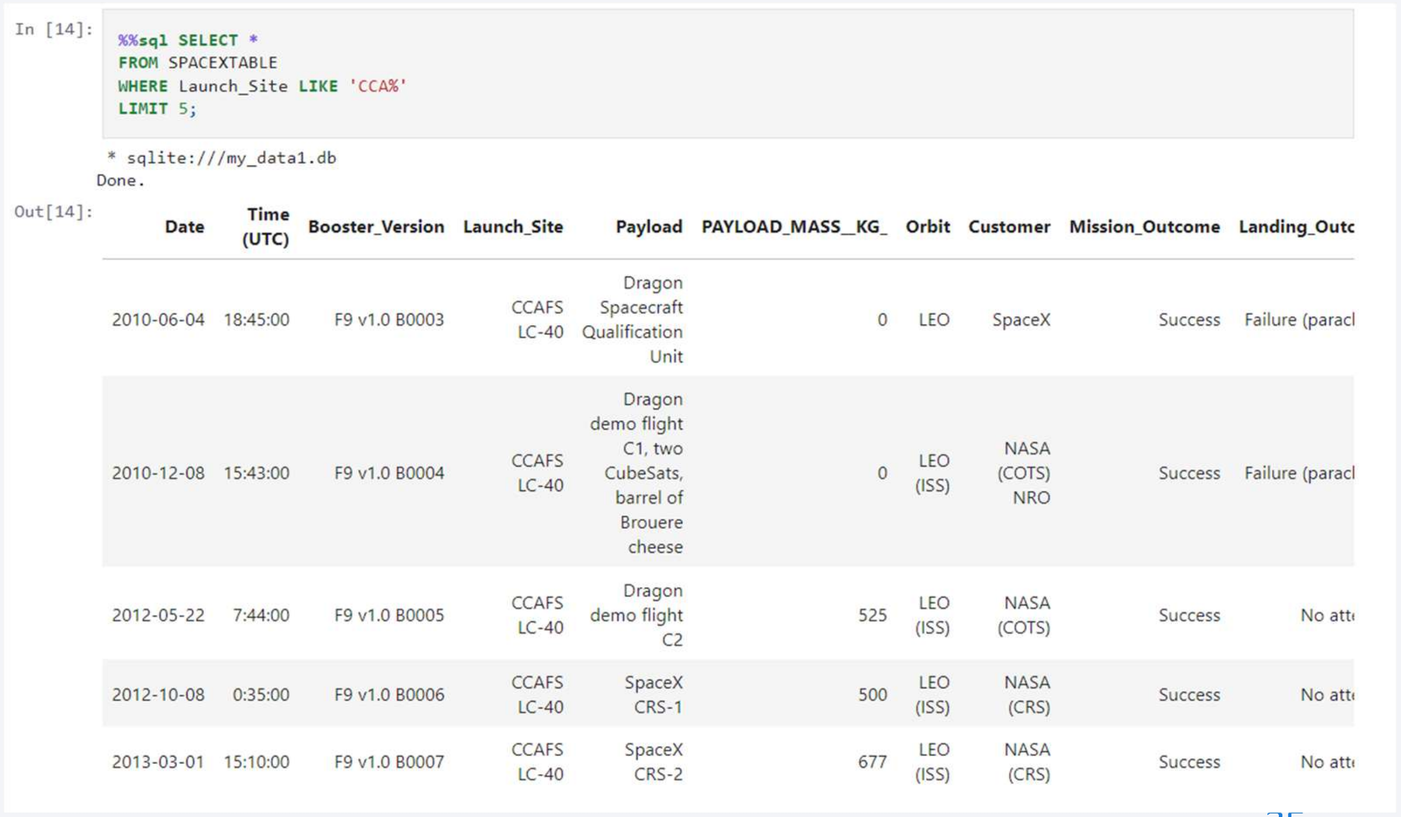
```
In [12]:  %%sql
          SELECT DISTINCT Launch_Site
          FROM SPACEXTABLE;

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

Out[12]:

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

Launch Site Names Begin with 'CCA'

- We used the key word LIMIT to extract 5 records and LIKE keyword to extract the Launch_Site start with “CCA”

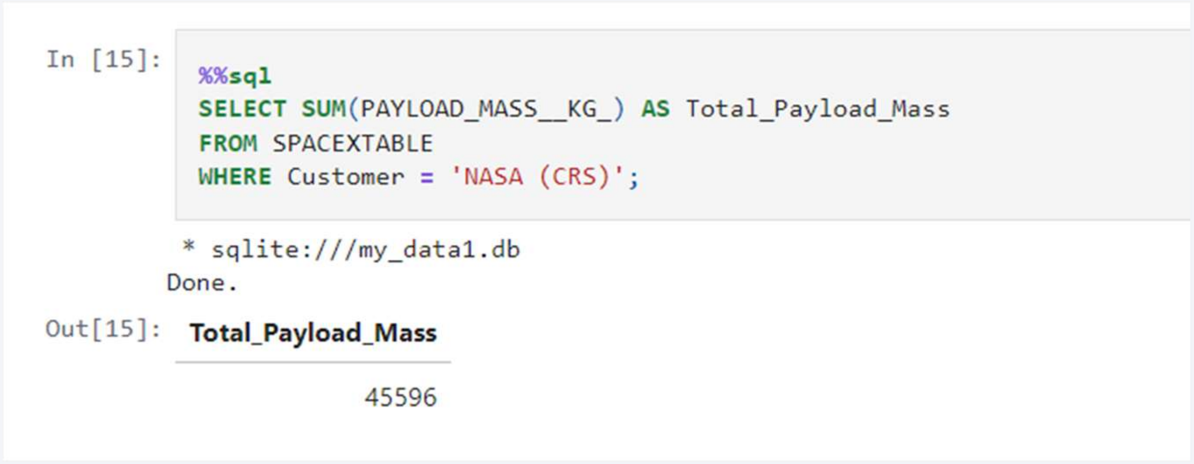


```
In [14]: %%sql SELECT *
         FROM SPACEXTABLE
         WHERE Launch_Site LIKE 'CCA%'
         LIMIT 5;

 * sqlite:///my_data1.db
Done.
Out[14]:
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outc
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (paracl
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 (paracl
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No att
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No att
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No att

Total Payload Mass

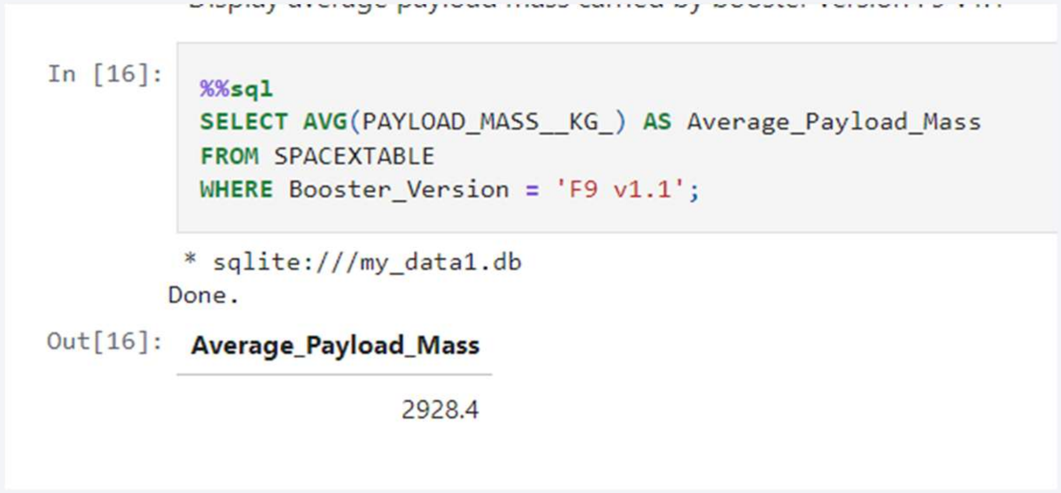
- Used SUM command to summarized the payload.



```
In [15]:  %%sql
          SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass
          FROM SPACEXTABLE
          WHERE Customer = 'NASA (CRS)';

           * sqlite:///my_data1.db
          Done.

Out[15]:  Total_Payload_Mass
                       45596
```

Average Payload Mass by F9 v1.1

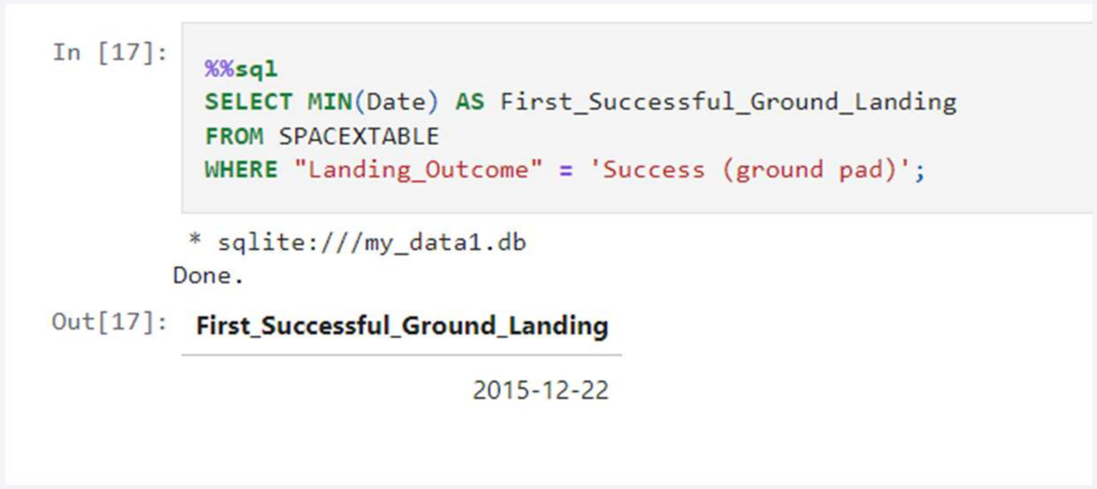
- Used AVG command to find the average of Payload Mass for booster version F9 v1.1



```
In [16]:  %%sql
          SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass
          FROM SPACEXTABLE
          WHERE Booster_Version = 'F9 v1.1';

 * sqlite:///my_data1.db
Done.

Out[16]:  Average_Payload_Mass
                        2928.4
```

First Successful Ground Landing Date

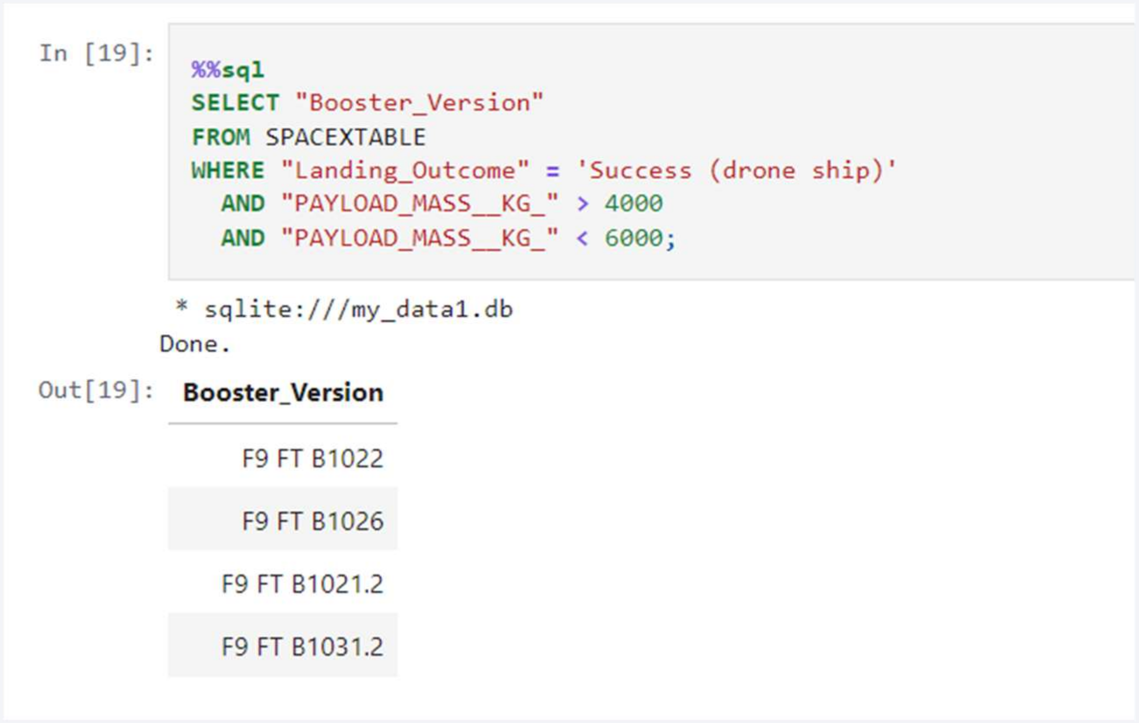
- Used MIN command to find the minimum of First Successful Ground Landing outcome for booster version F9 v1.1



```
In [17]:  %%sql
          SELECT MIN(Date) AS First_Successful_Ground_Landing
          FROM SPACEXTABLE
          WHERE "Landing_Outcome" = 'Success (ground pad)';

           * sqlite:///my_data1.db
          Done.

Out[17]:  First_Successful_Ground_Landing
                                2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- Used the WHERE clause 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



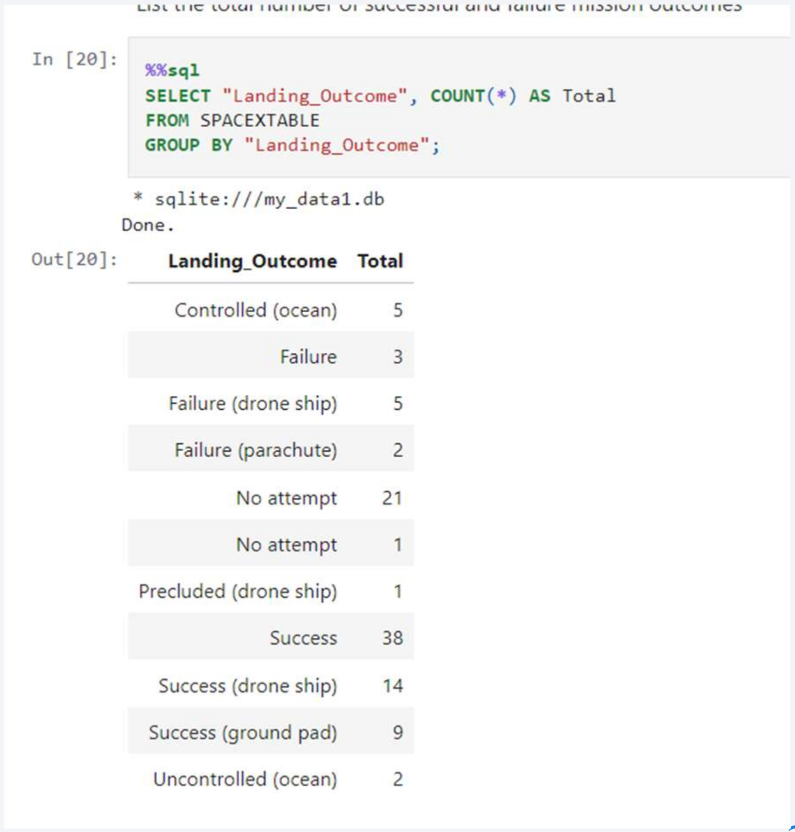
```
In [19]:  %%sql
          SELECT "Booster_Version"
          FROM SPACEXTABLE
          WHERE "Landing_Outcome" = 'Success (drone ship)'
            AND "PAYLOAD_MASS__KG_" > 4000
            AND "PAYLOAD_MASS__KG_" < 6000;

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

Out[19]:

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

Total Number of Successful and Failure Mission Outcomes

- Extract and count the different status of landing outcome.



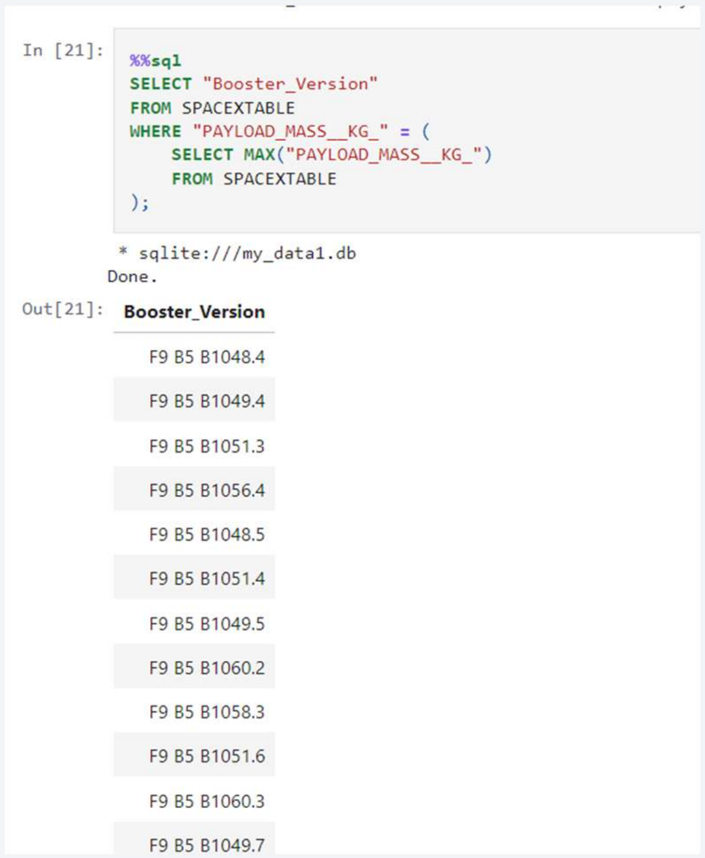
List the total number of successful and failure mission outcomes

```
In [20]:  %%sql
          SELECT "Landing_Outcome", COUNT(*) AS Total
          FROM SPACEXTABLE
          GROUP BY "Landing_Outcome";

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

```
Out[20]:      Landing_Outcome   Total
            Controlled (ocean)      5
                       Failure      3
          Failure (drone ship)      5
           Failure (parachute)      2
                    No attempt     21
                    No attempt      1
        Precluded (drone ship)      1
                       Success     38
          Success (drone ship)     14
         Success (ground pad)       9
          Uncontrolled (ocean)      2
```

Boosters Carried Maximum Payload

- Extracted the booster which have carried the maximum payload mass using subquery



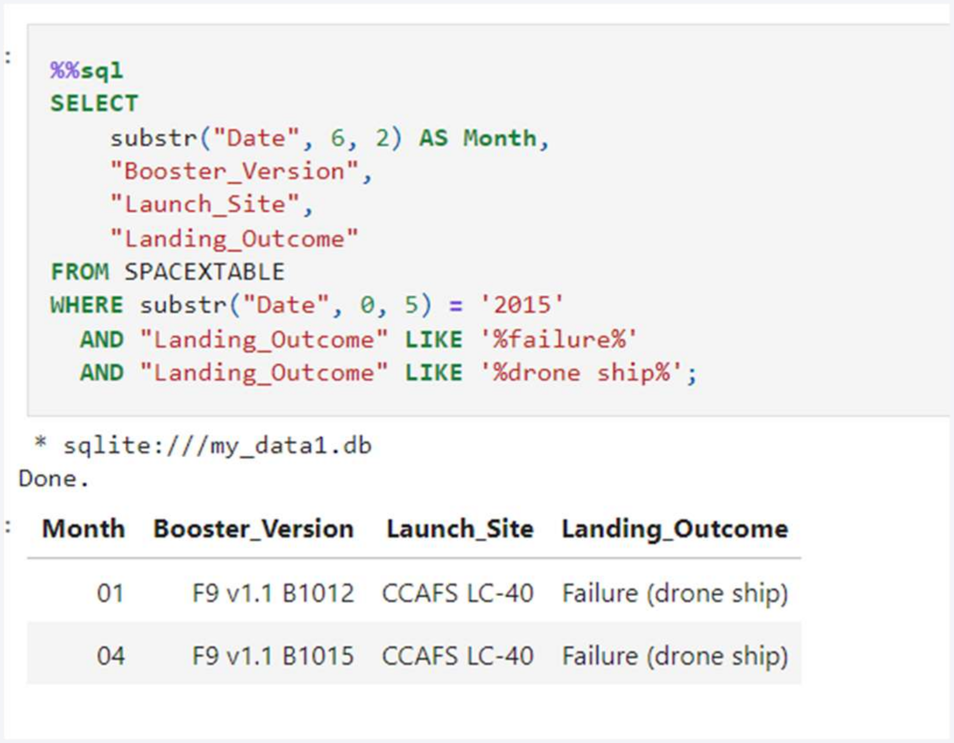
```
In [21]: %%sql
         SELECT "Booster_Version"
         FROM SPACEXTABLE
         WHERE "PAYLOAD_MASS__KG_" = (
             SELECT MAX("PAYLOAD_MASS__KG_")
             FROM SPACEXTABLE
         );

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

Out[21]:

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

- Extracted the list of failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015



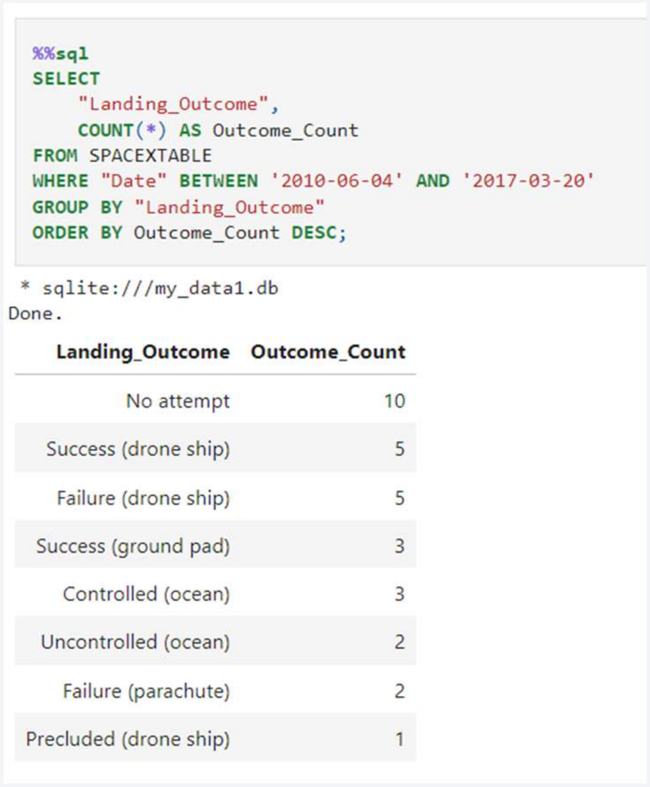
```
%%sql
SELECT
    substr("Date", 6, 2) AS Month,
    "Booster_Version",
    "Launch_Site",
    "Landing_Outcome"
FROM SPACEXTABLE
WHERE substr("Date", 0, 5) = '2015'
  AND "Landing_Outcome" LIKE '%failure%'
  AND "Landing_Outcome" LIKE '%drone ship%';
```

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

Month	Booster_Version	Launch_Site	Landing_Outcome
01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	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.



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

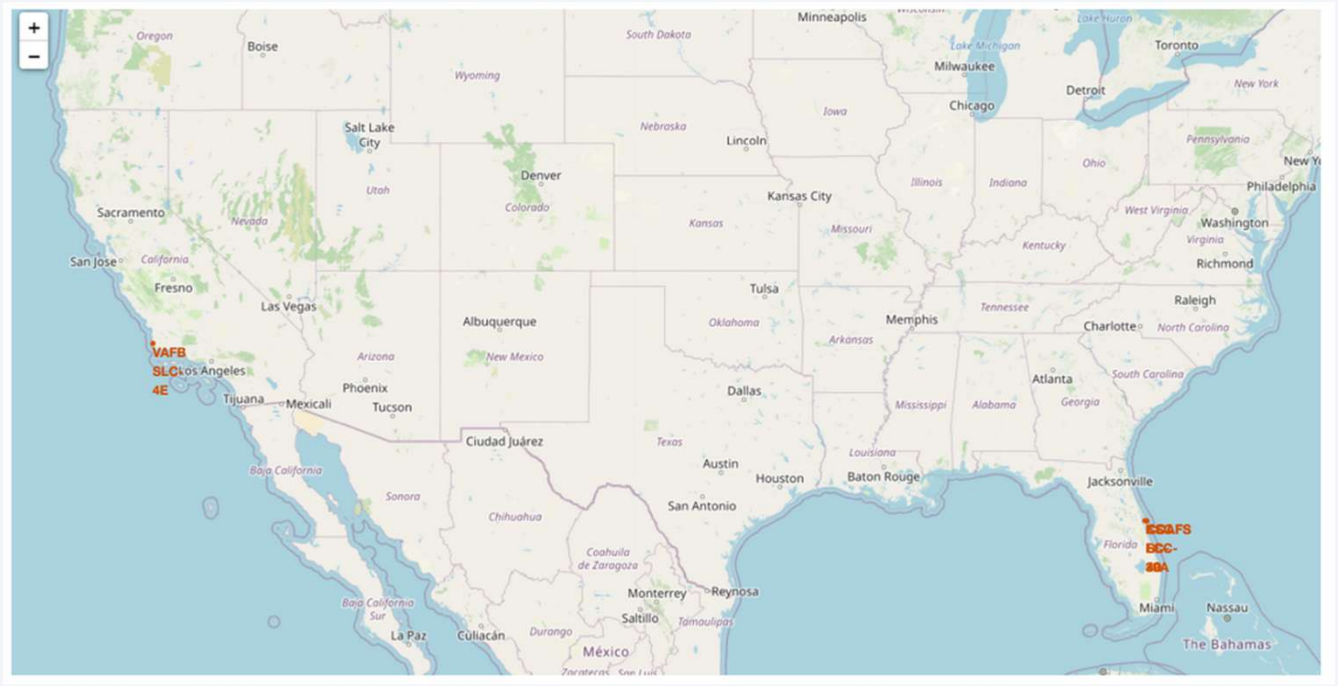
```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome	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

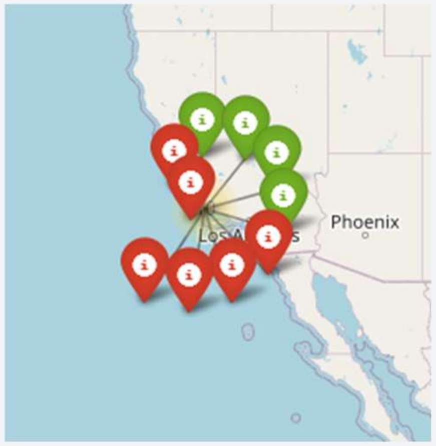
Launch Sites Proximities Analysis

All launch sites global map markers

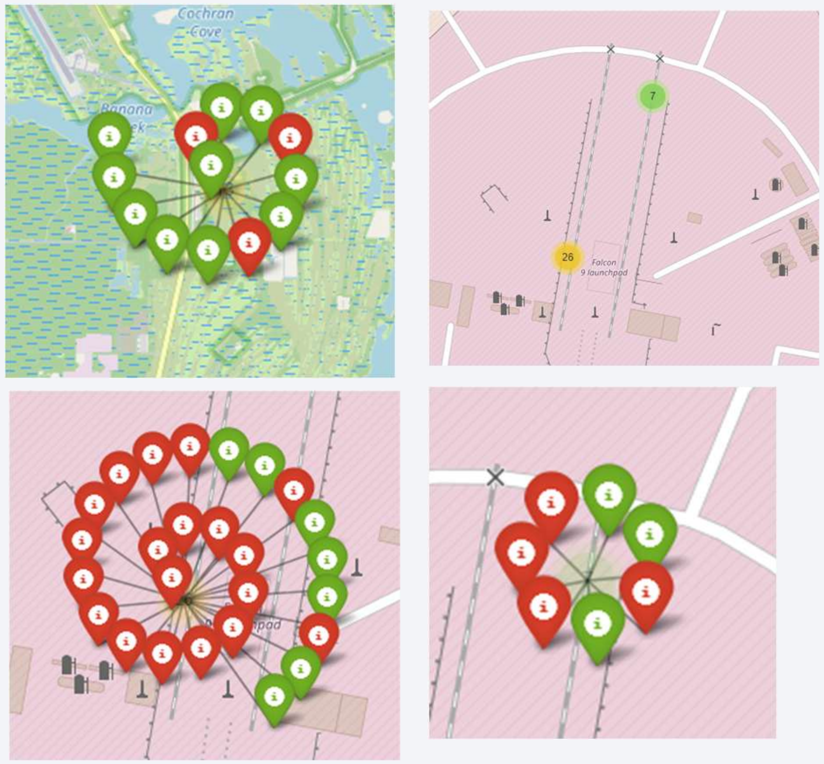


- We can see that the SpaceX launch sites are in the United States of America Florida and California.

Markers showing launch sites with color labels



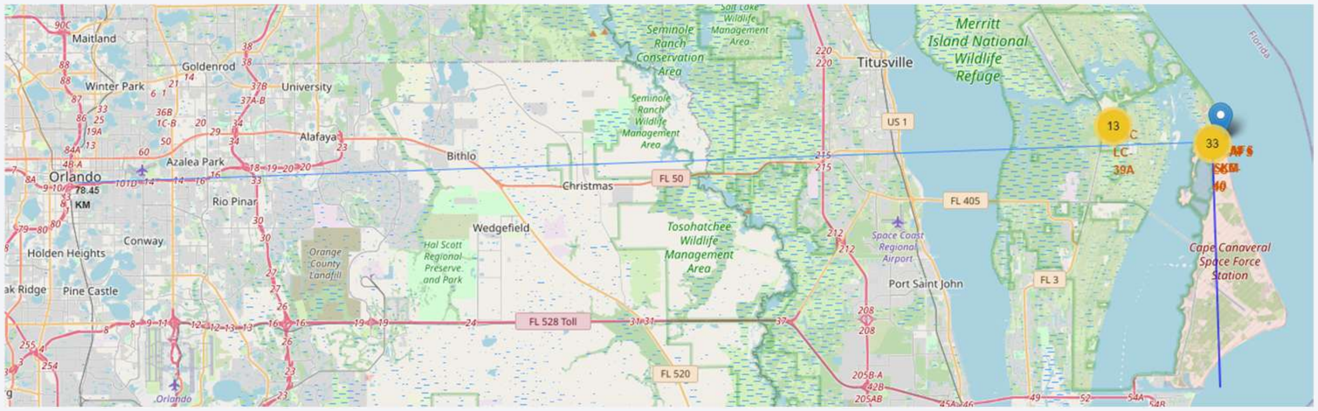
California Launch Site



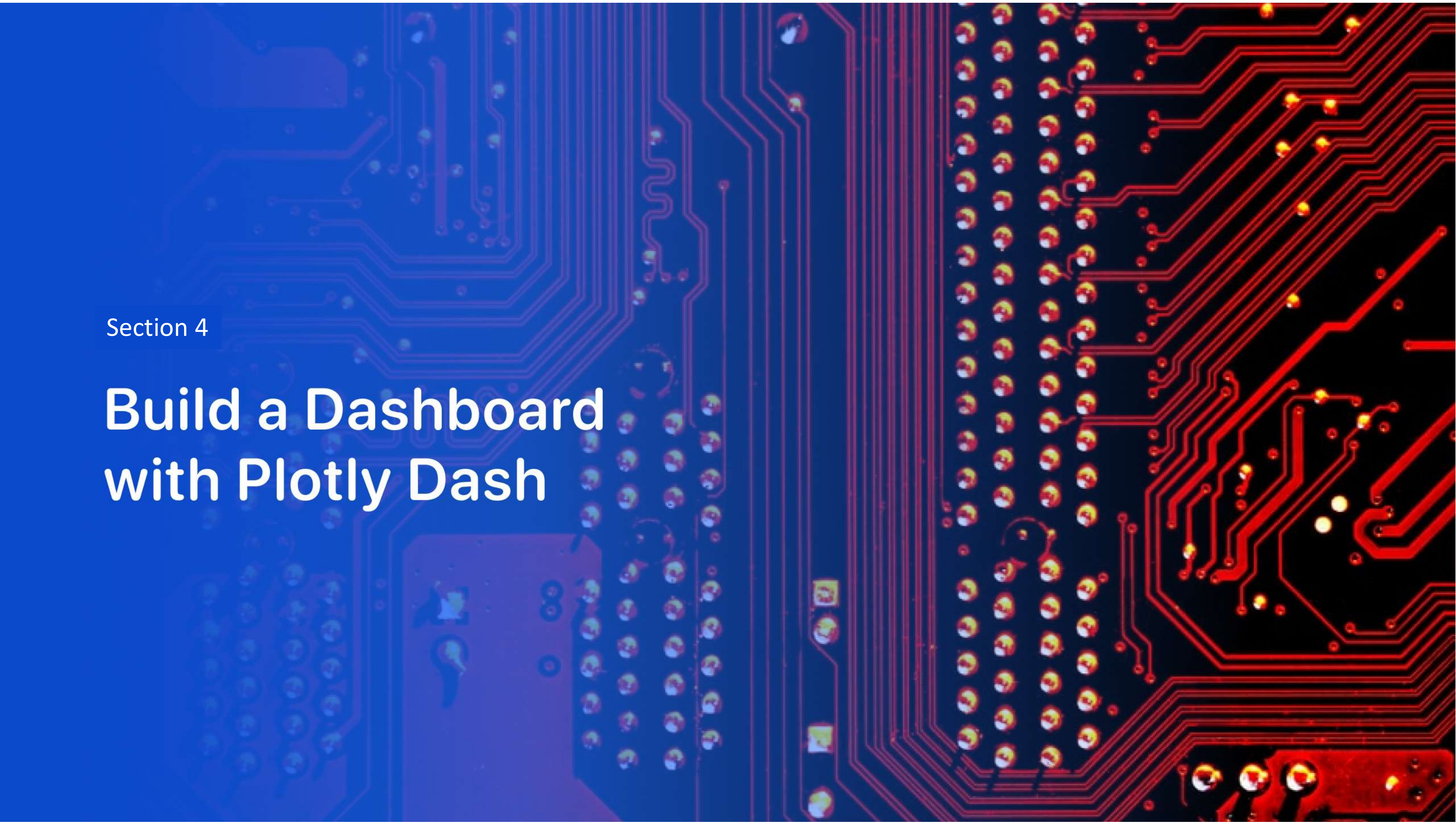
Florida Launch Site

- Green Marker shows successful Launches and Red Marker shows Failures

Launch Site distance to landmarks



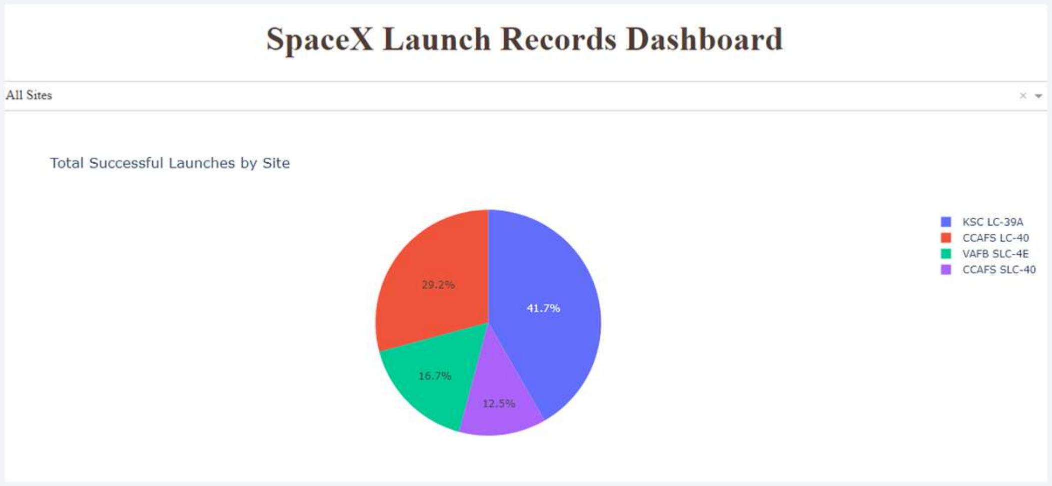
- Are launch sites in close proximity to railways? No.
- Are launch sites in close proximity to highways? No.
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 4

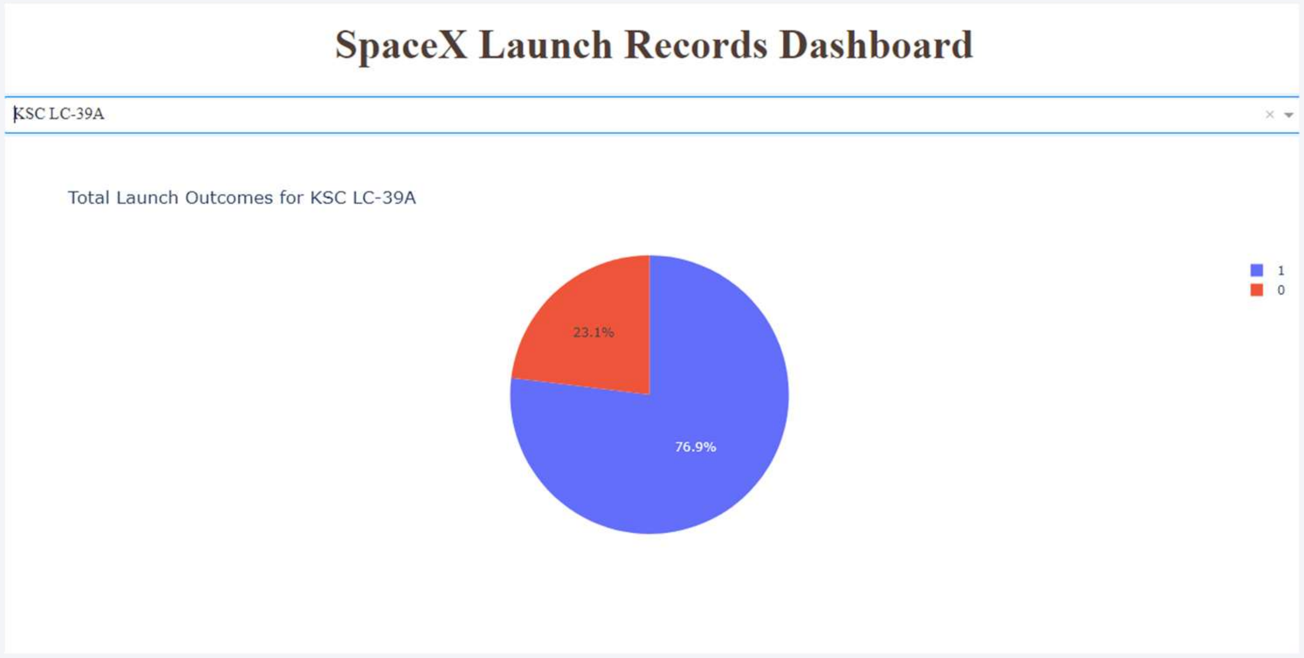
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



- Can see that KSC LC-39A had the most successful launches from all the sites

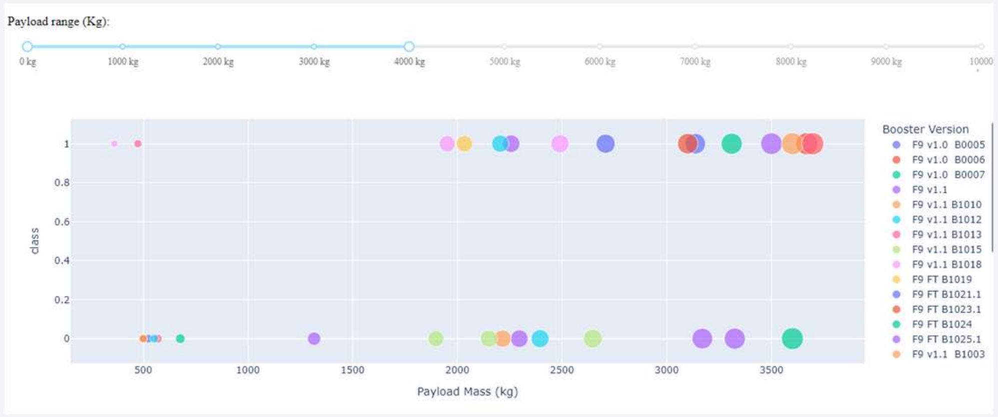
Pie chart showing the Launch site with the highest launch success ratio


SpaceX Launch Records Dashboard

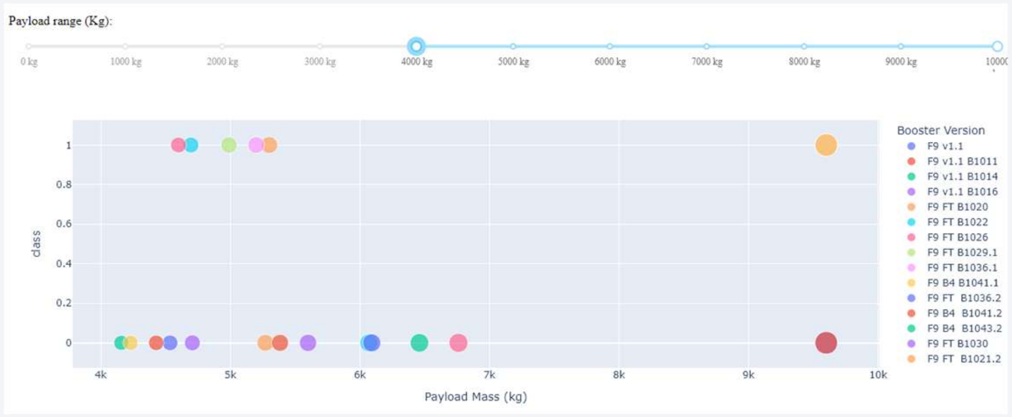
KSC LC-39A

- KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider



Low weighted payload 0kg-4000kg



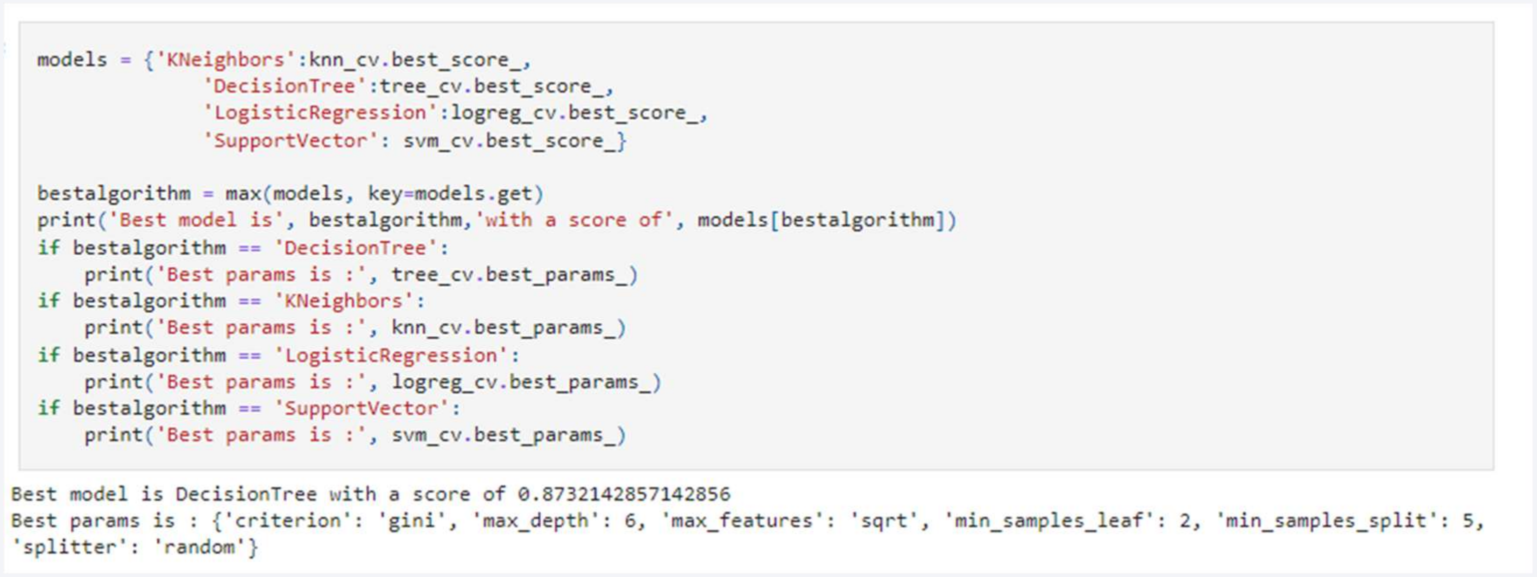
Heavy weighted payload 4000kg-10000kg

We can see the success rates for low weighted payloads is higher than the heavy weighted payloads.

Section 5

Predictive Analysis (Classification)

Classification Accuracy



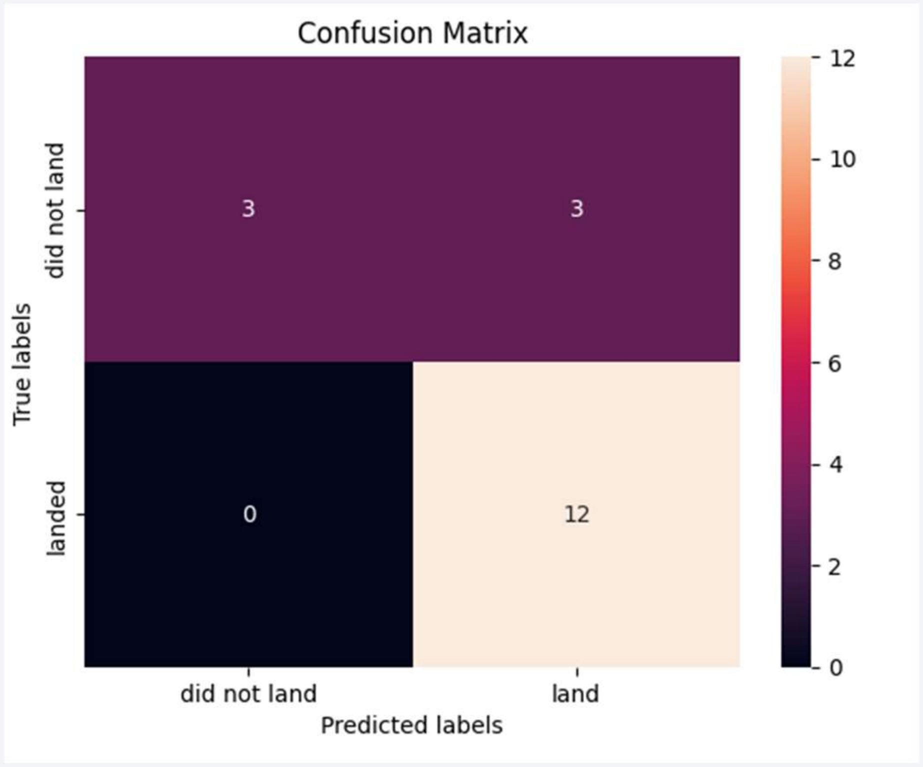
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5,
'splitter': 'random'}
```

- The decision tree classifier is the model with the highest classification accuracy

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

We can conclude the following:

- The larger the number of flights at a launch site, the higher the success rate at that site.
- Launch success rates increased steadily from 2013 to 2020.
- Orbits **ES-L1, GEO, HEO, SSO,** and **VLEO** achieved the highest success rates.
- **KSC LC-39A** had the most successful launches among all sites.
- The **Decision Tree Classifier** was the best-performing machine learning algorithm for this task.

Thank you!