IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Igor Ngouagnia

12/08/2025

Outline

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

Executive Summary

- Summary of methodologies

  - Data Collection (API & Web Scraping)

  - Data Wrangling

  - Exploratory data analysis (EDA) using visualization and SQL

  - Interactive visual analytics using Folium and Plotly Dash

  - Predictive analysis using classification models

- Summary of all results

  - Exploratory Data Analysis results

  - Interactive analytics demo in screenshots

  - Predictive analytics results

Introduction

- Project background and context

Space X 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 Space X 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 space X for a rocket launch. So the goal of the project is to create a machine learning pipeline to predict if the first stage will land successfuly.

- Problems you want to find answers

  - Determine the parameters that influence the outcome of a rocket landing.
  - Use these parameters to establish a model that can predict whether a rocket will land or not.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected from 2 main sources

    - https://api.spacexdata.com/v4/rockets/ for SpaceX API

    - https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches for web scraping
- Perform data wrangling
  - After conducting some statistical analyses on the collected data, we created a landing outcome label based on the outcome (with 1 meaning the booster successfully landed and 0 meaning it was unsuccessful).

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - After collection, the data is normalized, divided into training and test datasets, and then evaluated using different classification models. Finally, we evaluated the accuracy of each model using the appropriate metrics.

Data Collection

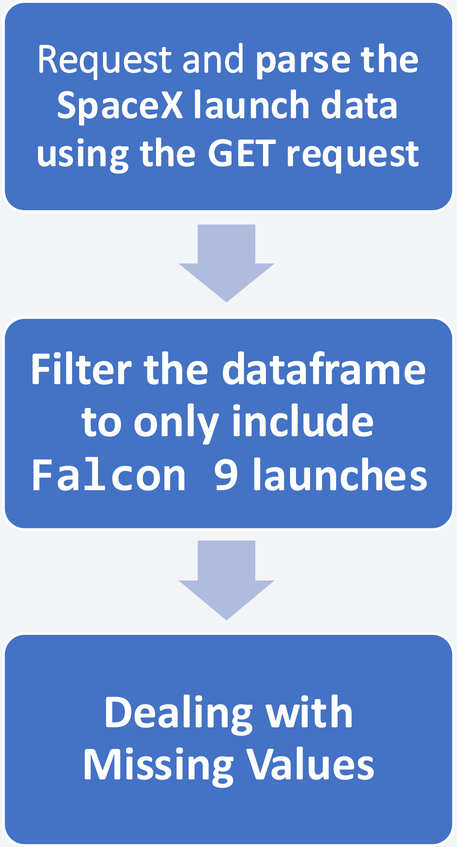
- Describe how data sets were collected.

The data sets were collected from SpaceX API (https://api.spacexdata.com/v4/rockets/) using a get request and from Wikipedia (https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches) using different web scraping technics. In this latter case, the table extracted was parsed and converted into a Pandas data frame using Pandas library.

Data Collection – SpaceX API

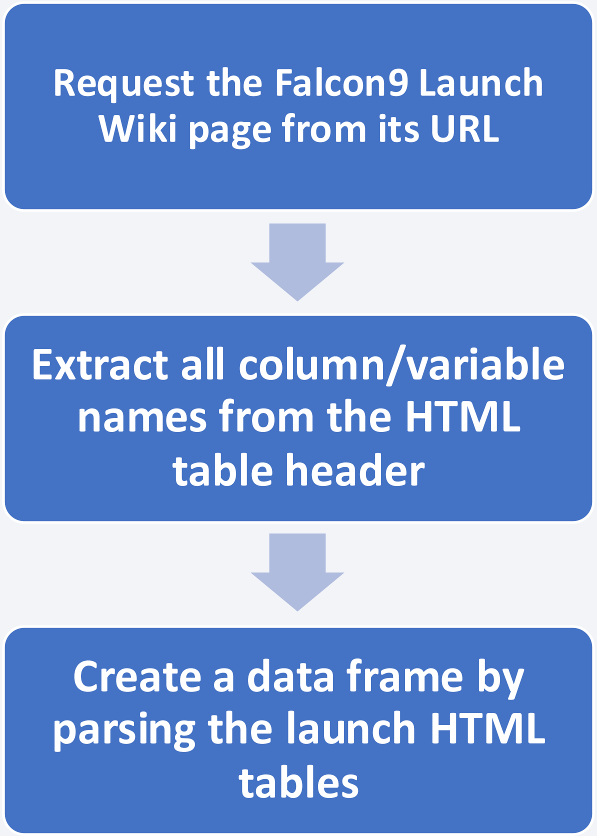
- The data was collected from SpaceX API according to this flowchart.

- Source code : https://github.com/igorngouagnia/Applied-Data-Science-Capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

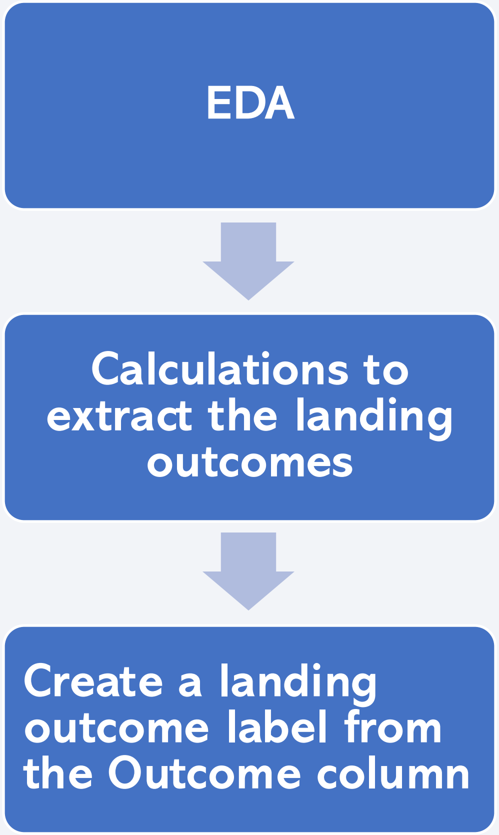
- The data was extracted from wikipedia and parsed according to this flowchart.
- Source code : https://github.com/igorngouagnia/Applied-Data-Science-Capstone/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

- Exploratory Data Analysis was performed on the data
- The number of launches on each site, the number and occurrence of each orbit and the number of occurence of mission outcome of the orbits were calculated
- A landing outcome label was created from the Outcome column

- Source code : https://github.com/igorngouagnia/Applied-Data-Science-Capstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- To explore the data, different types of graphs (scatter plots, bar charts) are used to visualize the relationships between pairs of features : FlightNumber vs PayloadMass, FlightNumber vs LaunchSite, Payload Mass vs Launch Site, success rate vs orbit, FlightNumber vs Orbit, Payload Mass vs Orbit, launch success yearly trend

- Source code : https://github.com/igorngouagnia/Applied-Data-Science-Capstone/blob/main/edadataviz.ipynb

EDA with SQL

- Summary of SQL queries performed

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000 kg
- List the total number of successful and failure mission outcomes
- List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function
- List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
- List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

- Source code : https://github.com/igorngouagnia/Applied-Data-Science-Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Map objects such as markers, circles, lines and marker clusters were created and added to the folium map
- Markers allow to indicate locations like launch sites
- Circles are used to highlight area around specific coordinates
- Marker clusters are a good way to simplify a map containing many markers having the same coordinate
- Lines are used to show the distance between two locations (coordinates)

- Source code : https://github.com/igorngouagnia/Applied-Data-Science-Capstone/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

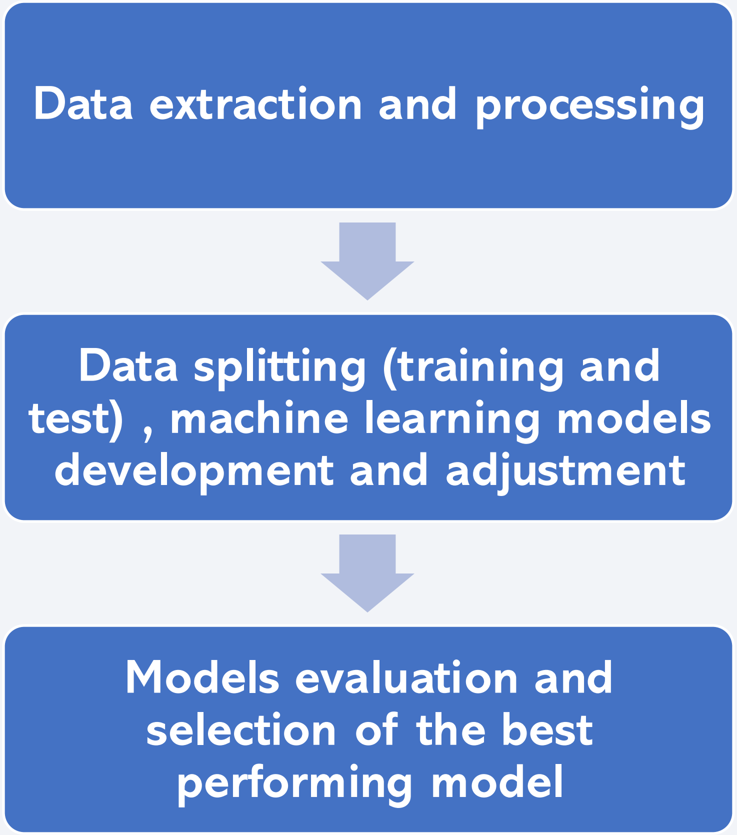
- We developed an interactive dashboard using Plotly dash.
- We created pie charts showing the total number of launches per site.
- We created a scatter plot showing the relationship between the outcome and the payload mass (in kg) for the different versions of the booster.

- Source code (completed Plotly Dash lab) : https://github.com/igorngouagnia/Applied-Data-Science-Capstone/blob/main/spacex-dash-app.py

Predictive Analysis (Classification)

- We imported the data using numpy and pandas libraries.
- Then we created the class column and normalized the data.
- Next, we divided the data into two parts: training and testing.
- We built different machine learning models and adjusted their hyperparameters using GridSearchCV.
- We found the method that performs best based on the accuracy metric calculated on the test data.

- **Source code** : https://github.com/igorngouagnia/Applied-Data-Science-Capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb



Results

- Exploratory data analysis results (a few results, among others)

  - There are four launch sites for the space mission

  - The total payload mass carried by boosters launched by NASA (CRS) is 45596 kg

  - The average payload mass carried by booster version F9 v1.1 is 2928.4 kg

  - The date when the first successful landing outcome in ground pad was achieved is 2015-12-22

  - The names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000 kg are : F9 FT B1022, F9 FT B1026, F9 FT B1021.2 and F9 FT B1031.2

  - Visualizing the launch success yearly trend, one observes that sucess rate since 2013 kept increasing till 2020

Results

- Interactive analytics demo in screenshots

With this interactive analysis, it was possible to identify that launch sites are generally located in safe areas, far from cities, such as near the sea. They are also located near highways and railroads, which facilitates the transport of needed people and equipment.

- Predictive analysis results

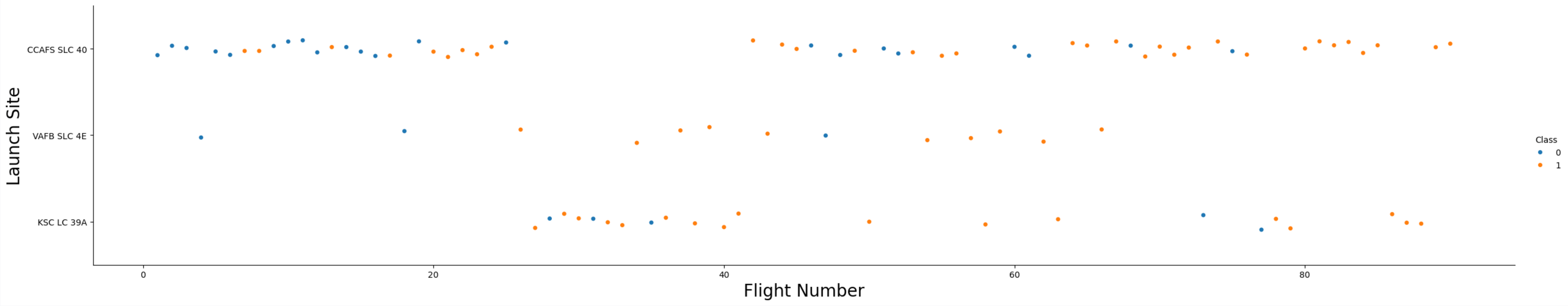
The evaluation of the different machine learning models shows that they are equivalent in terms of performance.

Section 2

Insights drawn from EDA

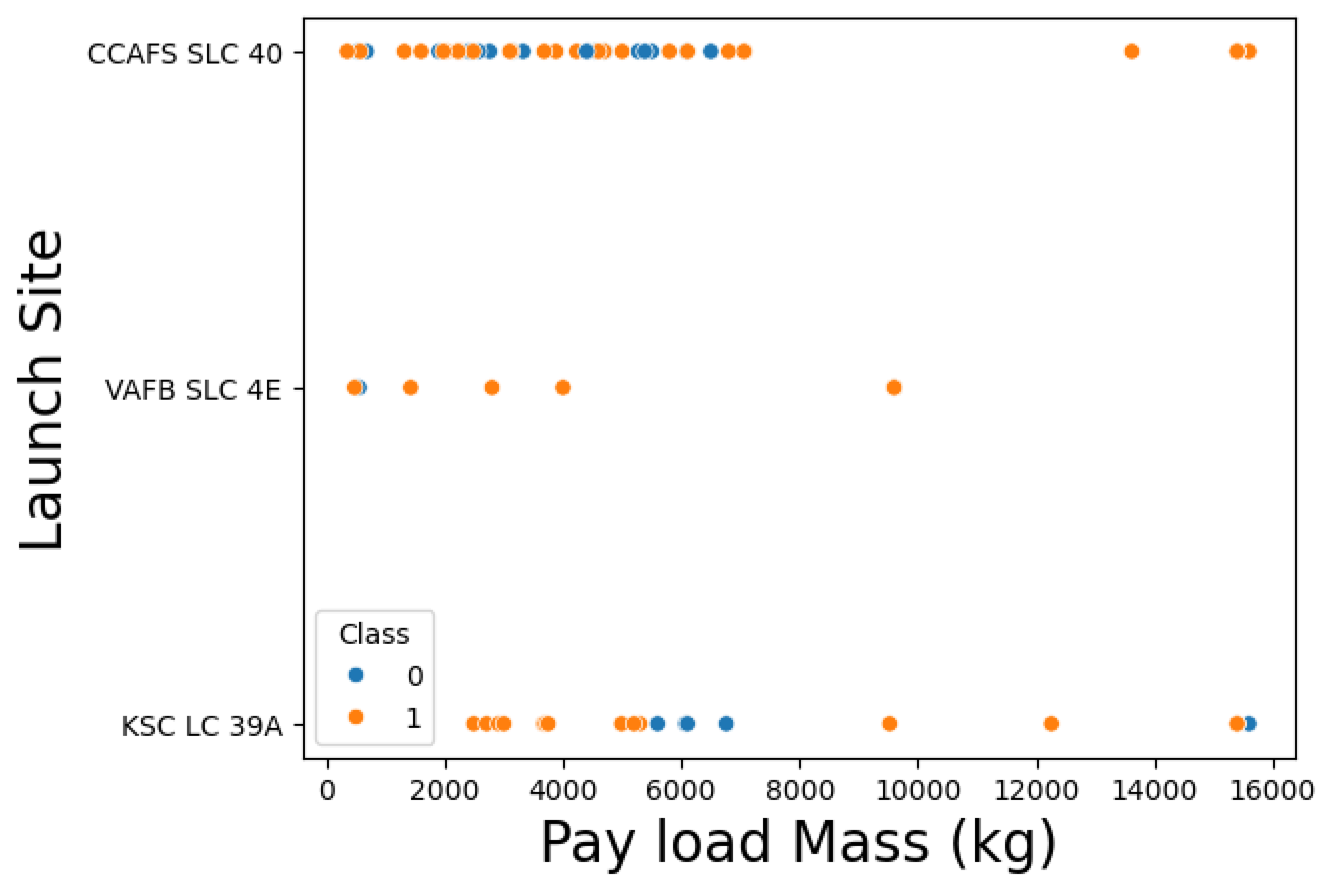
Flight Number vs. Launch Site

- As the flight number increases, the first stage is more likely to land successfully.
- For all launch sites, the probability of a successful outcome is higher. It is greater for VAFB SLC 4E and KSC LC 39A than for CCAFS SLC 40.



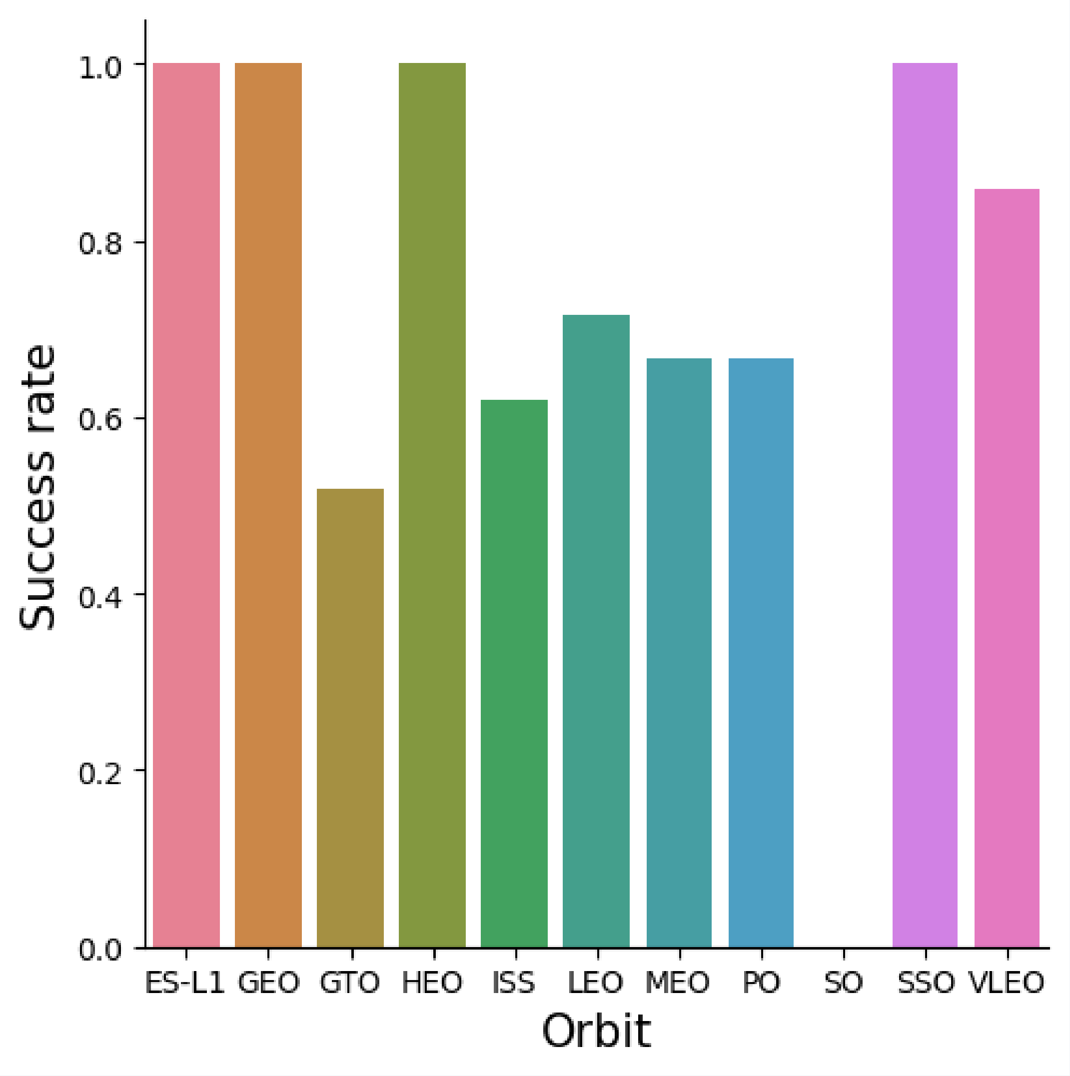
Payload vs. Launch Site

For the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000 kg).



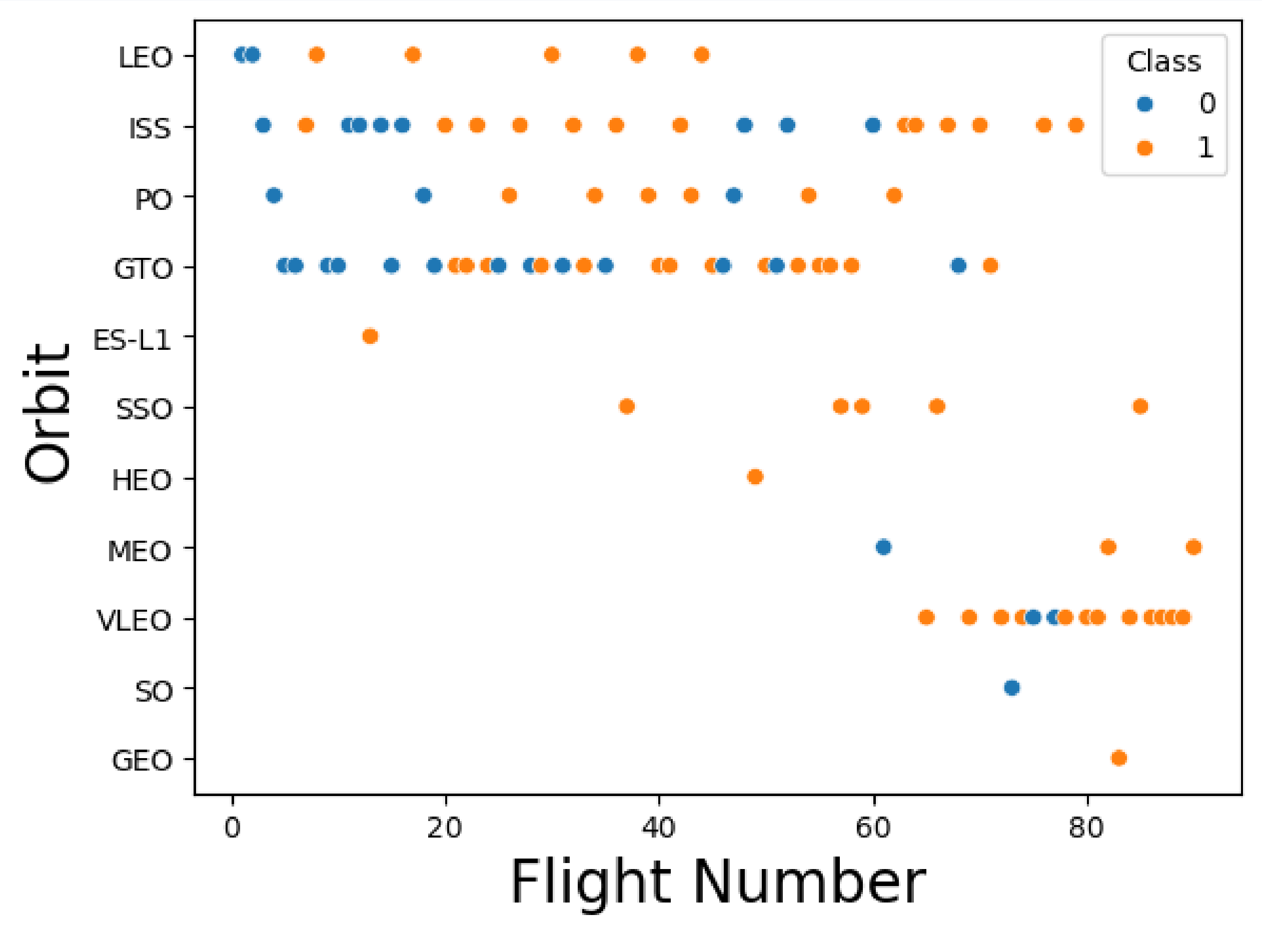
Success Rate vs. Orbit Type

The orbits with the highest success rates are ES-L1, GEO, HEO, and SSO.



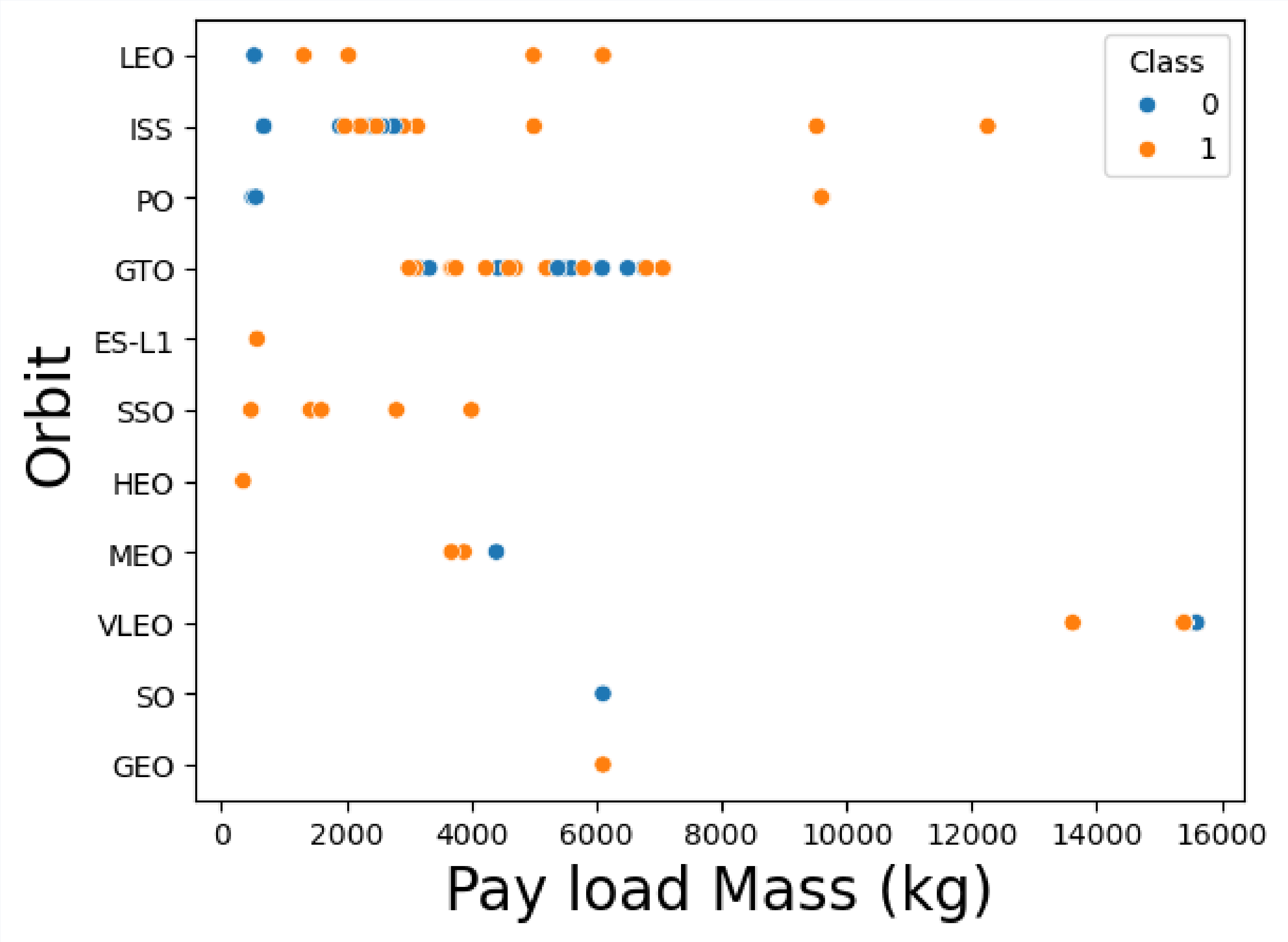
Flight Number vs. Orbit Type

In the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.



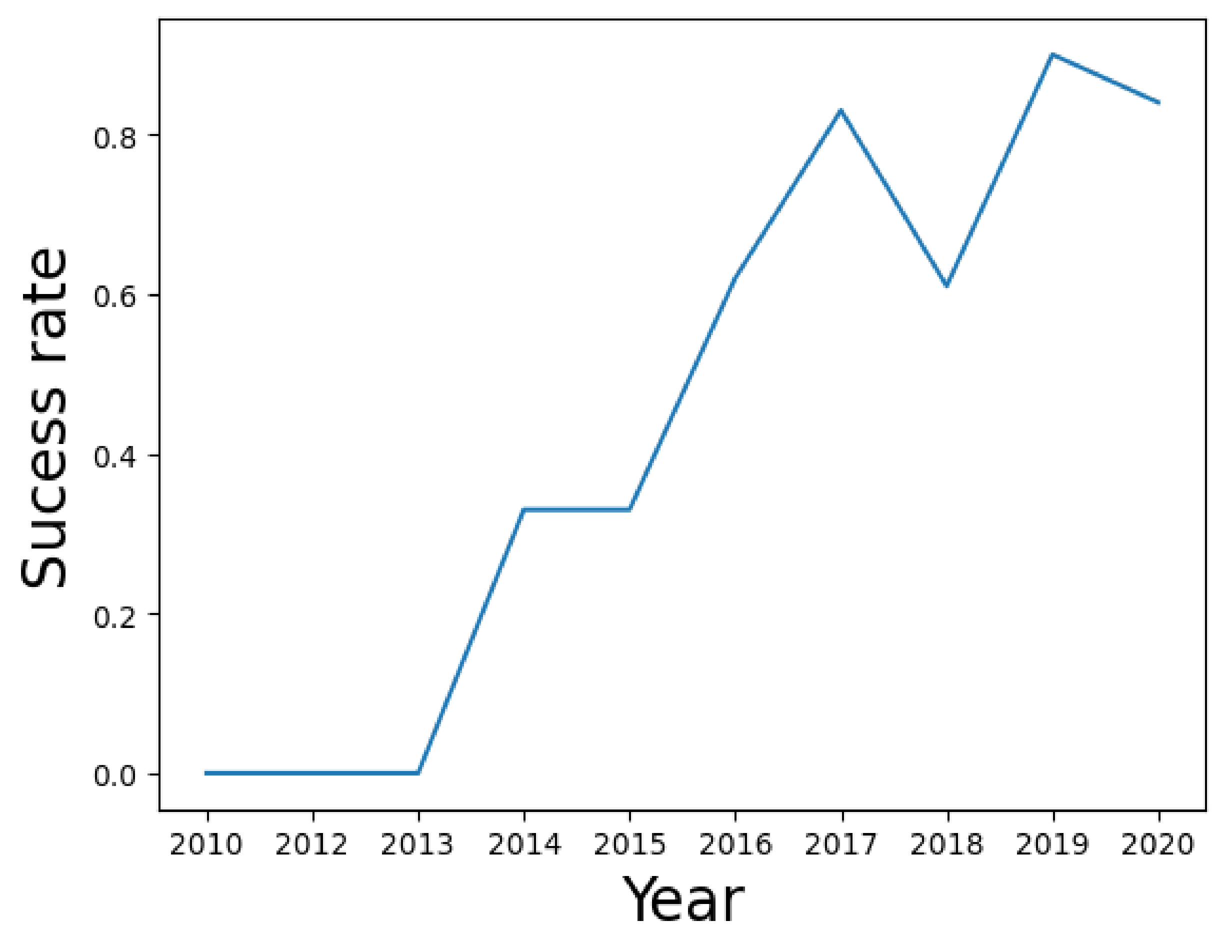
Payload vs. Orbit Type

- With heavy payloads, the successful landing or positive landing rate are more for Polar,LEO and ISS.
- For GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.



Launch Success Yearly Trend

The sucess rate since 2013 kept increasing till 2020



All Launch Site Names

The unique() function from the NumPy library is used to find the unique "launch sites" in the SpaceX data.

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

In [12]:
```
unique_launch=df["Launch_Site"].unique().tolist()
unique_launch
```

Out[12]:
```
['CCAFS LC-40', 'VAFB SLC-4E', 'KSC LC-39A', 'CCAFS SLC-40']
```

Launch Site Names Begin with 'CCA'

LIKE 'CCA%' selects records where the site name begin with CCA, and LIMIT 5 limits the display to 5 records.

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

```
In [13]:
query = """
    SELECT *
    FROM SPACEXTABLE
    WHERE launch_site LIKE 'CCA%'
    LIMIT 5
    """

cur.execute(query)
items = cur.fetchall()

for i in items:
    print(i)
```

```
('2010-06-04', '18:45:00', 'F9 v1.0  B0003', 'CCAFS LC-40', 'Dragon Spacecraft Qualification Unit', 0, 'LEO', 'Spa
ceX', 'Success', 'Failure (parachute)')
('2010-12-08', '15:43:00', 'F9 v1.0  B0004', 'CCAFS LC-40', 'Dragon demo flight C1, two CubeSats, barrel of Brouer
e 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 (COT
S)', 'Success', 'No attempt')
('2012-10-08', '0:35:00', 'F9 v1.0  B0006', 'CCAFS LC-40', 'SpaceX CRS-1', 500, 'LEO (ISS)', 'NASA (CRS)', 'Succes
s', 'No attempt')
('2013-03-01', '15:10:00', 'F9 v1.0  B0007', 'CCAFS LC-40', 'SpaceX CRS-2', 677, 'LEO (ISS)', 'NASA (CRS)', 'Succe
ss', 'No attempt')
```

Total Payload Mass

One finds 45596 kg

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

In [14]:

```
query = """
    SELECT customer, SUM(payload_mass__kg_)
    FROM SPACEXTABLE
    GROUP BY customer
    HAVING customer = 'NASA (CRS)'
    """

cur.execute(query)
items = cur.fetchall()

for i in items:
    print(i)
```

```
('NASA (CRS)', 45596)
```

Average Payload Mass by F9 v1.1

One finds 2928.4 kg

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

In [15]:

```
query = """
    SELECT booster_version, AVG(payload_mass__kg_)
    FROM SPACEXTABLE
    GROUP BY booster_version
    HAVING booster_version = 'F9 v1.1'
    """

cur.execute(query)
items = cur.fetchall()

for i in items:
    print(i)
```

```
('F9 v1.1', 2928.4)
```

First Successful Ground Landing Date

Using min function one finds 2015-12-22

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

In [16]:

```
query = """
    SELECT MIN(date), "Landing_Outcome"
    FROM SPACEXTABLE
    GROUP BY "Landing_Outcome"
    HAVING "Landing_Outcome" = 'Success (ground pad)'
    """

cur.execute(query)
items = cur.fetchall()

for i in items:
    print(i)
```

```
('2015-12-22', 'Success (ground pad)')
```

Successful Drone Ship Landing with Payload between 4000 and 6000

WHERE allows to filter boosters that landed successfully, while the AND condition allows to select the payload mass range we are interested in.

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [17]:

```
query = """
    SELECT booster_version, payload_mass__kg_, "Landing_Outcome"
    FROM SPACEXTABLE
    WHERE "Landing_Outcome" = 'Success (drone ship)'
        AND payload_mass__kg_ > 4000
        AND payload_mass__kg_ < 6000
    """

cur.execute(query)
items = cur.fetchall()

for i in items:
    print(i)
```

```
('F9 FT B1022', 4696, 'Success (drone ship)')
('F9 FT B1026', 4600, 'Success (drone ship)')
('F9 FT  B1021.2', 5300, 'Success (drone ship)')
('F9 FT  B1031.2', 5200, 'Success (drone ship)')
```

Total Number of Successful and Failure Mission Outcomes

We can see that there have been a total of 100 (98+1+1) successful missions and only one failed mission.

List the total number of successful and failure mission outcomes

In [18]:

```
query = """
    SELECT mission_outcome, COUNT(*)
    FROM SPACEXTABLE
    GROUP BY mission_outcome
    """

cur.execute(query)
items = cur.fetchall()

for i in items:
    print(i)
```

```
('Failure (in flight)', 1)
('Success', 98)
('Success ', 1)
('Success (payload status unclear)', 1)
```

Boosters Carried Maximum Payload

We use a subquery in the WHERE clause with the aggregate function MAX()

List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.

In [22]:

```
query = """
    SELECT booster_version
    FROM SPACEXTABLE
    WHERE payload_mass__kg_ = (
        SELECT MAX(payload_mass__kg_)
        FROM SPACEXTABLE
    )
    """

cur.execute(query)
items = cur.fetchall()

for i in items:
    print(i)
```

```
('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

We used substr(Date,6,2) as month to get the months and substr(Date,0,5)='2015' for year.

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

In [23]:

```
query = """
    SELECT date, substr(Date, 6, 2), launch_site, booster_version, "Landing_Outcome"
    FROM SPACEXTABLE
    WHERE substr(Date, 0, 5) = '2015'
      AND "Landing_Outcome" = 'Failure (drone ship)'
    """

cur.execute(query)
items = cur.fetchall()

for i in items:
    print(i)
```

```
('2015-01-10', '01', 'CCAFS LC-40', 'F9 v1.1 B1012', 'Failure (drone ship)')
('2015-04-14', '04', 'CCAFS LC-40', 'F9 v1.1 B1015', 'Failure (drone ship)')
```

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

- We first selected landing outcomes and extract the COUNT of landing outcomes from the data
- Using the WHERE clause, we filter for landing outcomes BETWEEN 2010-06-04 and 2017-03-20
- We then applied the GROUP BY clause to group the landing outcomes
- Finally we applied the ORDER BY clause to order the grouped landing outcomes in descending order.

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

In [34]:

```
%sql select LANDING_Outcome , count(*) As total_count from spacextbl where Date between '2010-06-04' and '2017-03
```

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

Out[34]:

Landing_Outcome	total_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

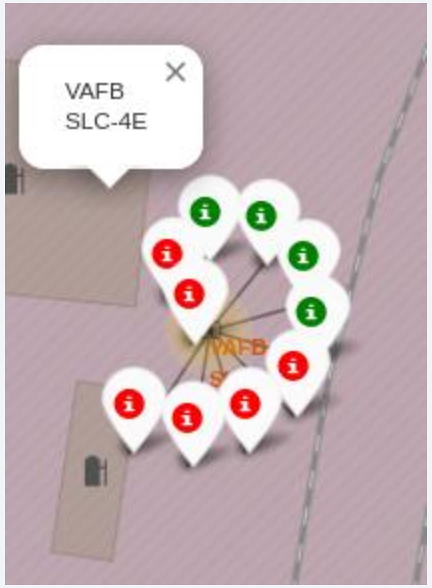
All launch sites location – Global map

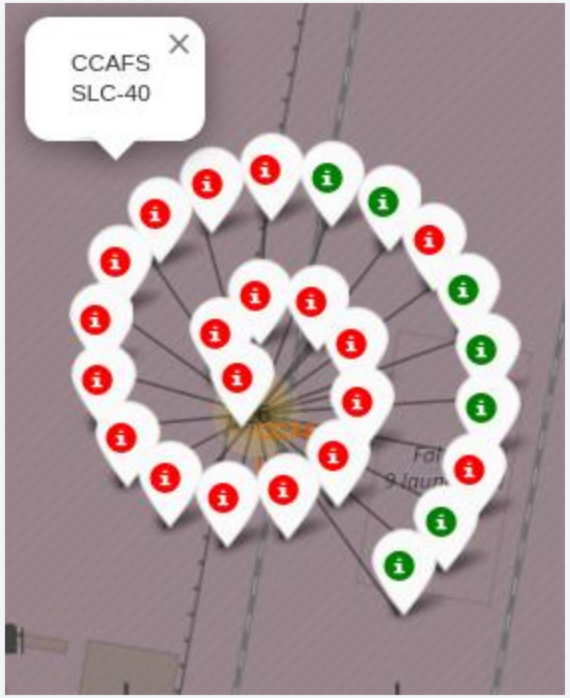
We can see that the launch sites are located on the coasts of the United States of America.

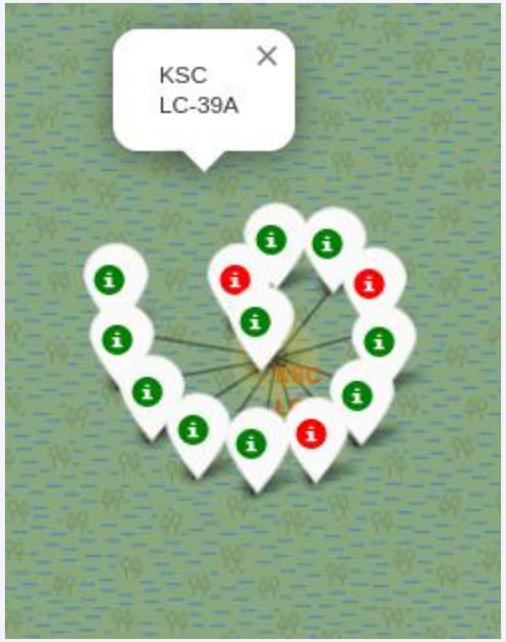
Color-labeled launch outcomes

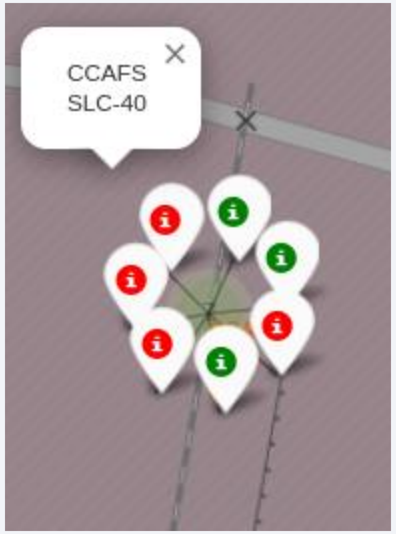
Green marker : successful laucnh   red marker : failed launch

The launch site KSC LC-39A has the highest success rate, while CCAFS SLC-40 seems to have the lowest success



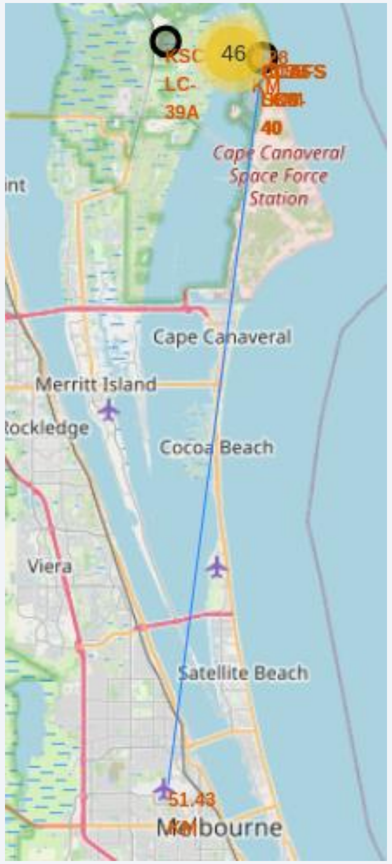






Distance from launch site to its proximities

We observe that the selected launch site (CCAFS SLC-40) is close to the coastline, a railway, and a highway, but is far away from cities.

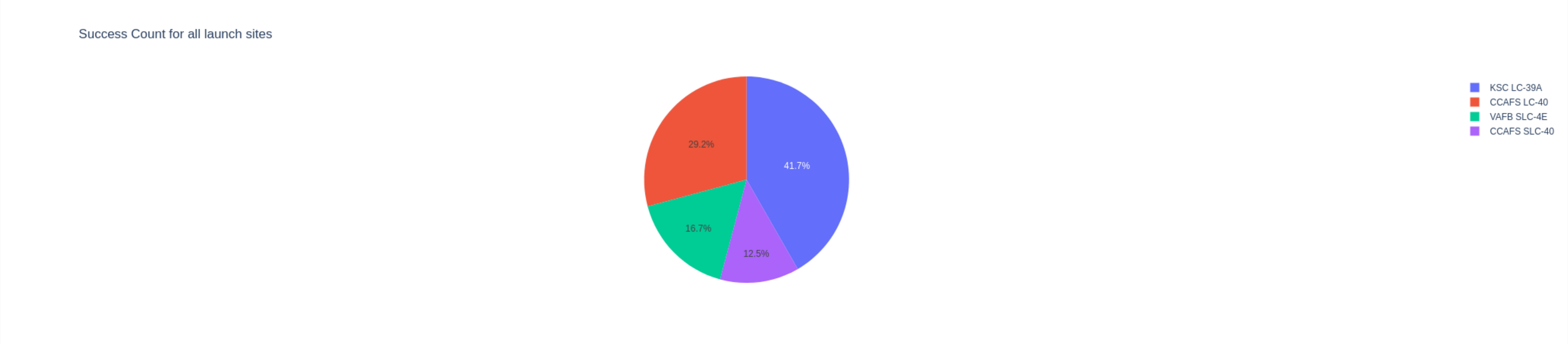


Section 4

Build a Dashboard with Plotly Dash

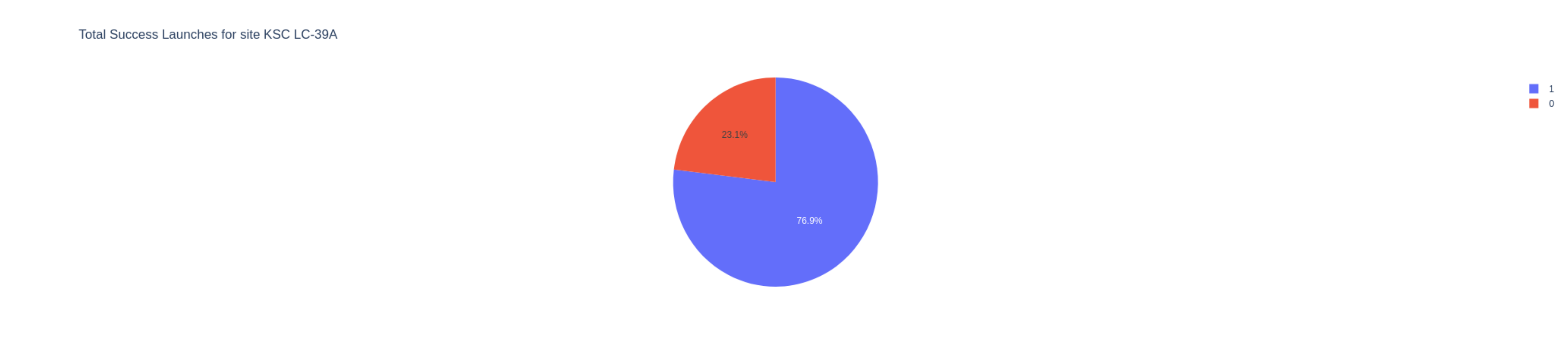
Launch success percentage of all sites

From all the sites, KSC LC-39A has the most successful launches



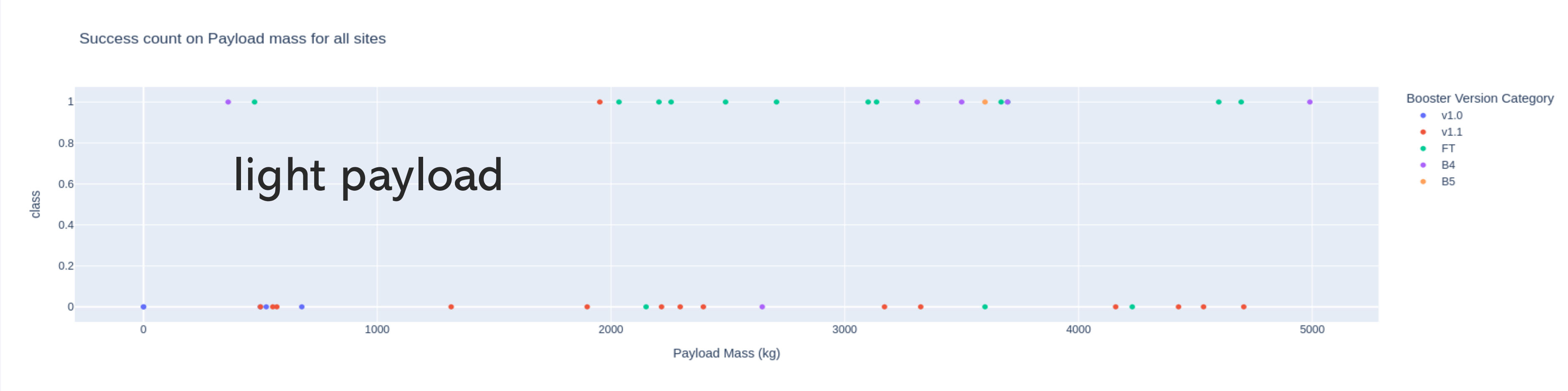
Launch site with the highest launch success ratio

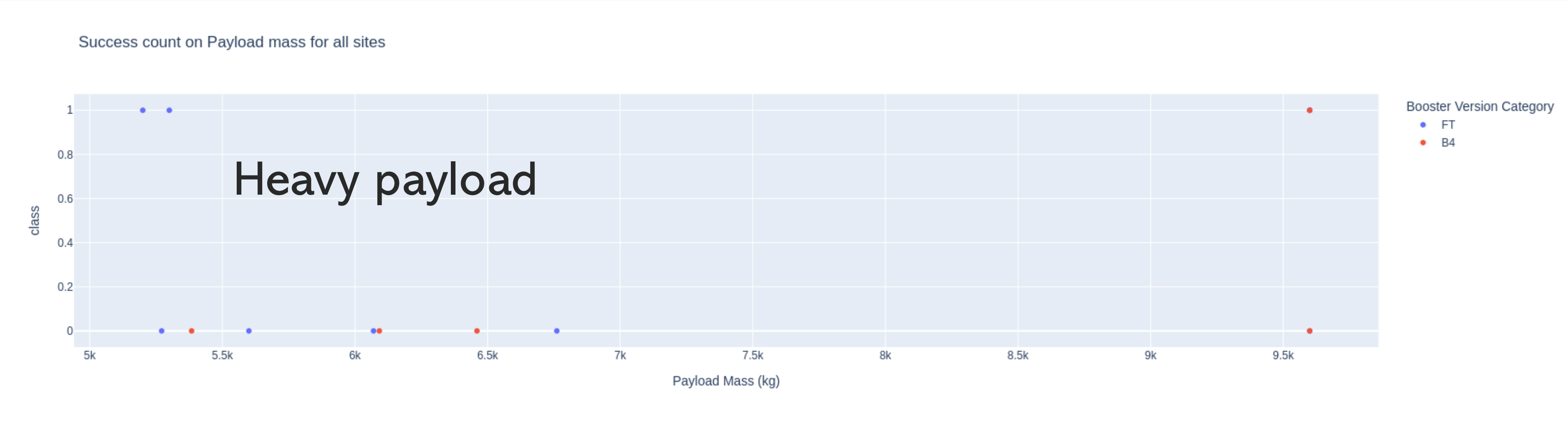
It is KSC LC-39A with a launch success ratio of 76.9%



Payload vs. Launch Outcome for all sites

- Light payloads have a higher success rate than the Heavy payloads
- FT booster has the largest success rate





Section 5

Predictive Analysis (Classification)

Classification Accuracy

We find that all the models are equivalent in terms of performance.

Find the method performs best:

In [45]:

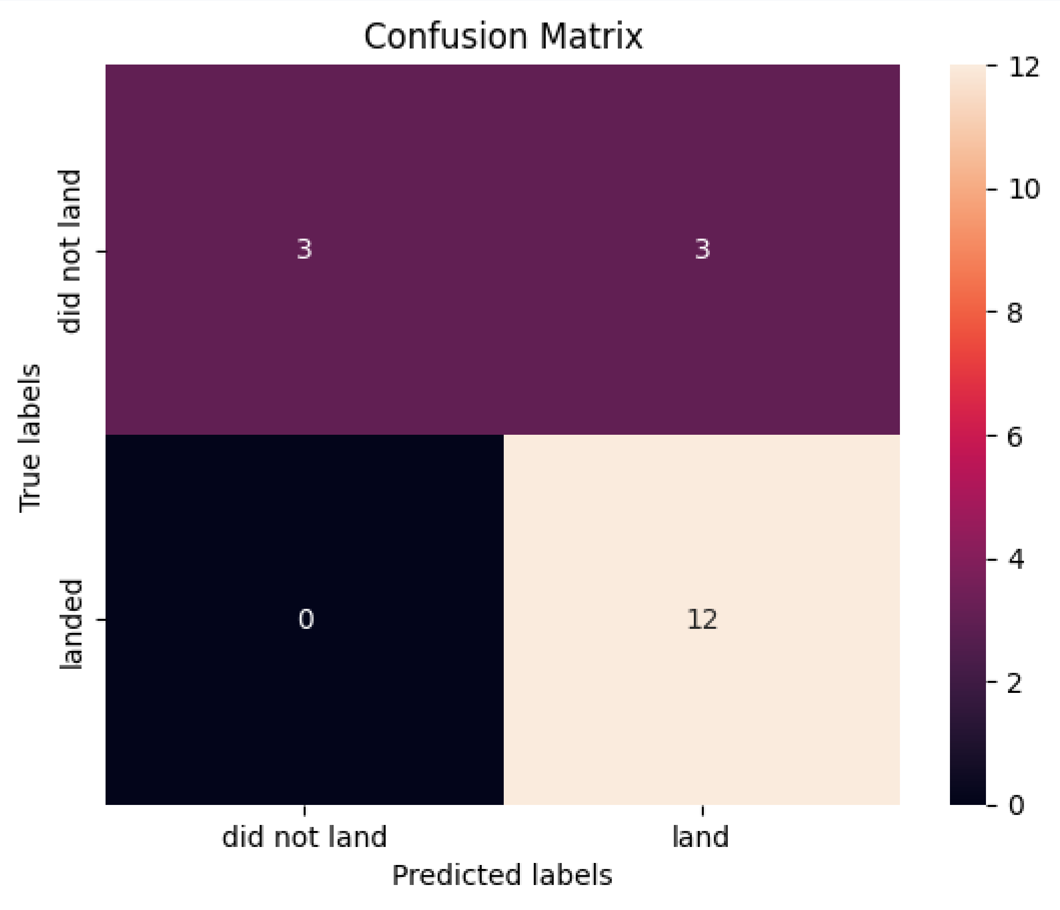
```python
Report = pd.DataFrame({'Method' : ['Test Data Accuracy']})

knn_accuracy=knn_cv.score(X_test, Y_test)
Decision_tree_accuracy=tree_cv.score(X_test, Y_test)
SVM_accuracy=svm_cv.score(X_test, Y_test)
Logistic_Regression=logreg_cv.score(X_test, Y_test)

Report['Logistic_Reg'] = [Logistic_Regression]
Report['SVM'] = [SVM_accuracy]
Report['Decision Tree'] = [Decision_tree_accuracy]
Report['KNN'] = [knn_accuracy]

Report.transpose()
```

Out[45]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

Confusion Matrix

- We have the exact same confusion matrix for the four models
- We see that the models can distinguish between the different classes. The problem is false positives, i.e. unsuccessful landing marked as successful landing by the classifier.



Conclusions

- The higher the number of flights from a launch site, the higher the success rate from that site.
- The orbits with the highest success rates are ES-L1, GEO, HEO, SSO, and VLEO.
- The sucess rate since 2013 kept increasing till 2020
- The site KSC LC-39A has recorded the highest number of successful launches
- The FT booster has recorded the highest number of successful launches
- All classifiers have the same performance

Thank you!