IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Lorenzo Anastasi

17 July 2025

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection with API;
  - Data collection with Web Scraping;
  - Data Wrangling;
  - Exploratory Data Analysis with Visualization;
  - Exploratory Data Analysis with SQL;
  - Interactive Visual Analytics with Folium;
  - Interactive Visual Analytics with Plotly Dash;
  - Predictive Analysis.
- Summary of all results
  - Exploratory Data Analysis results;
  - Interactive Visual Analytics results
  - Predictive Analysis results.

Introduction

- Project background and context
    - SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used by an alternate company who wants to bid against SpaceX for a rocket launch.
- Problems you want to find answers
    - Which data, publicly available on SpaceX site, can determine the successful of a landing;
    - What are the inter-relations between the gathered data;
    - What is the best model to predict the result of a landing.

Section 1

Methodology

Methodology

Executive Summary

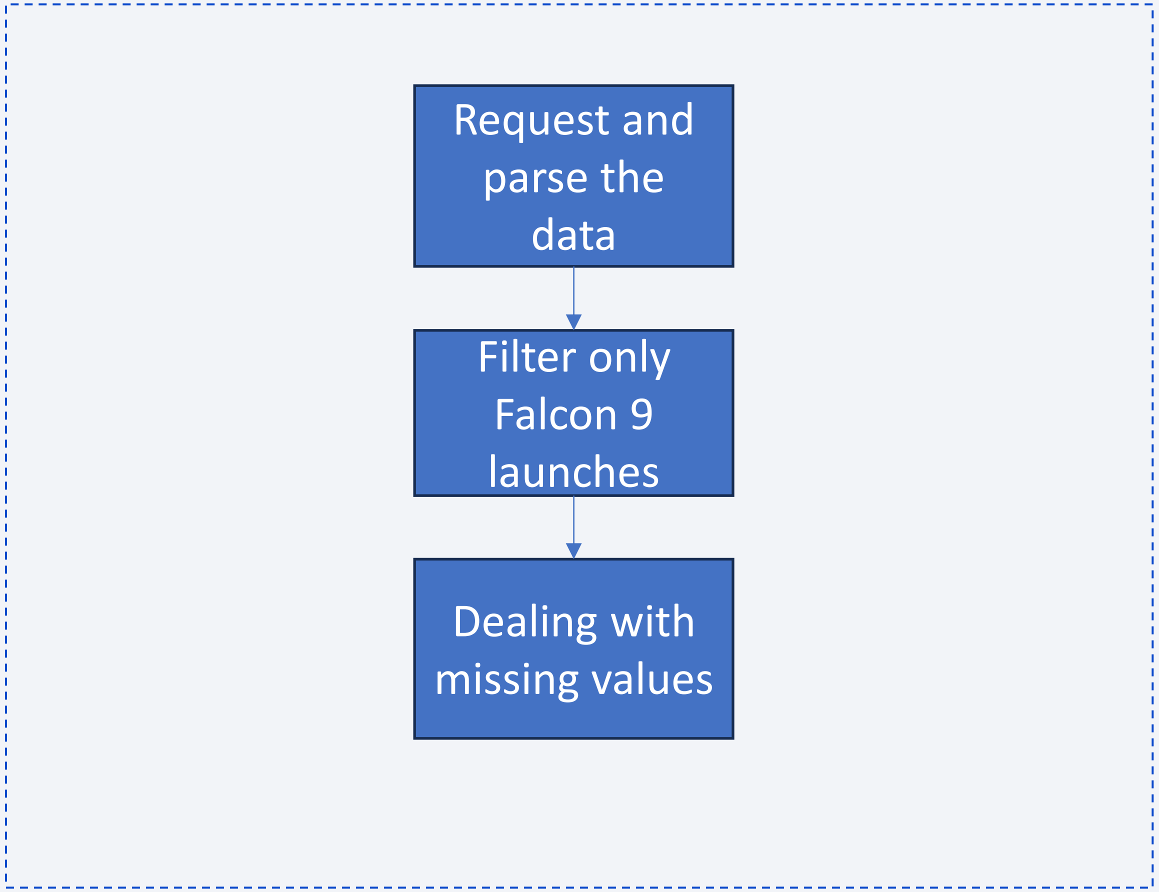
- Data collection methodology:
  - Data is gathered from an API, specifically the SpaceX REST API, available at: api.spacexdata.com/v4/
  - Data is also gathered scraping Wiki pages related to Falcon 9 Launch data.
- Perform data wrangling
  - The mainly objective of this methodology is to convert the outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - We have built a machine learning pipeline to predict if the first stage of the Falcon 9 lands successfully. This have included: Preprocessing, allowing us to standardize our data, and Train_test_split, allowing us to split our data into training and testing data, We will train the model and perform Grid Search, allowing us to find the hyperparameters that allow a given algorithm to perform best. Using the best hyperparameter values, we will determine the model with the best accuracy using the training data.

Data Collection

- Describe how data sets were collected.
    - Data is gathered from an API, specifically the SpaceX REST API, available at:
        - api.spacexdata.com/v4/
    - Data is also gathered scraping Wiki pages related to Falcon 9 Launch data at this URL:
        - https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

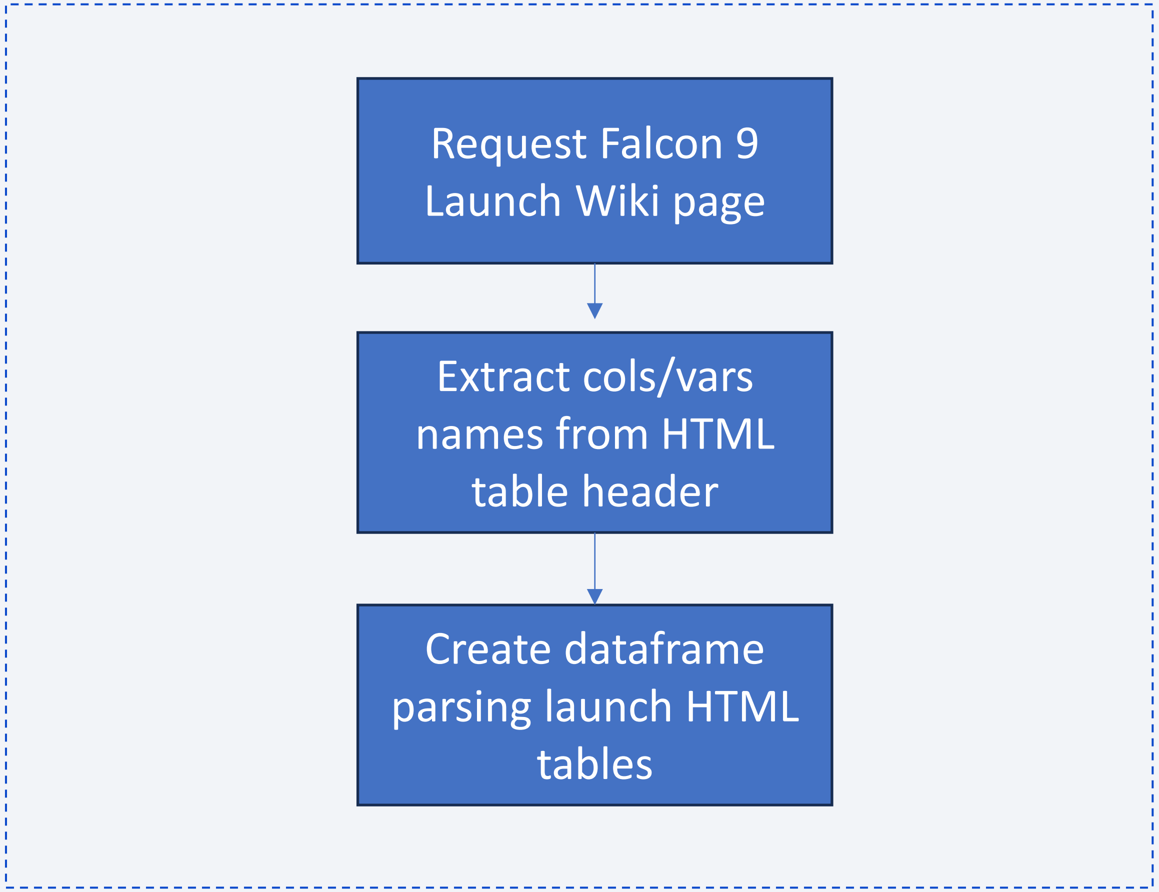
Data Collection – SpaceX API

- SpaceX offers publicly available data via REST API.
- The flowchart on the right shows the tasks of the Data Collection methodology.
- Source code available publicly at:
    - https://github.com/lodovico65/spacey/blob/main/spacex-data-collection-api.ipynb


Request and parse the data
Filter only Falcon 9 launches
Dealing with missing values

Data Collection - Scraping

- Data is also gathered scraping Wiki pages related to Falcon 9 Launch data.
- The flowchart on the right shows the tasks of the Data Collection methodology.
- Source code publicly available at:
    - https://github.com/lodovico65/spacey/blob/main/webscraping.ipynb



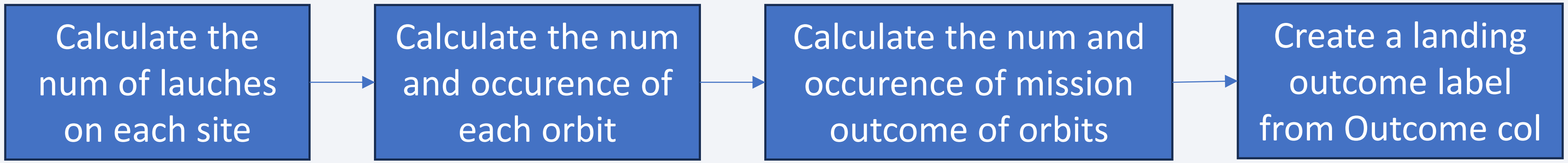
Request Falcon 9 Launch Wiki page

Extract cols/vars names from HTML table header

Create dataframe parsing launch HTML tables

Data Wrangling

- We have performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
- The following flowchart shows the tasks of the Data Wrangling methodology:



Calculate the num of lauches on each site

Calculate the num and occurence of each orbit

Calculate the num and occurence of mission outcome of orbits

Create a landing outcome label from Outcome col

- Source code publicly available at:
  - https://github.com/lodovico65/spacey/blob/main/spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- We have performed EDA with Data Visualization using Pandas and Matplolib. The following charts have been plotted:
  - Visualize the relationship between Flight Number and Launch Site with scatter plot;
  - Visualize the relationship between Payload Mass and Launch Site with scatter plot;
  - Visualize the relationship between success rate of each orbit type with bar chart;
  - Visualize the relationship between FlightNumber and Orbit type with scatter plot;
  - Visualize the relationship between Payload Mass and Orbit type with scatter plot;
  - Visualize the launch success yearly trend with line chart.
- This is the URL of the completed EDA with data visualization notebook:
  - https://github.com/lodovico65/spacey/blob/main/edadataviz.ipynb

EDA with SQL

- We have performed EDA with SQL using SQLAlchemy, prettytable and sqlite3. The following SQL queries have been 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;
  - 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;
  - 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;
- This is the URL of the completed EDA with SQL:
  - https://github.com/lodovico65/spacey/blob/main/eda-sql.ipynb

Build an Interactive Map with Folium

- Firstly, we have to use the Map() function to create a map.
  - The Circle() function permit to circle the coordinates by entering parameters such as radius and color. We have used the function to draw a small circle near the city of Houston;
  - The Marker() function permit to mark the location of the coordinates and set the location name by using the popup parameter. We have used the function to mark launch sites like ‘NASA JSC’;
  - The MarkerCluster() function permit to handle many markers and can help to declutter the map;
  - The Line() function can be used to show the distance between two points on the map.
- The GitHub URL of the completed interactive map with Folium map is available at:
  - https://github.com/lodovico65/spacey/blob/main/launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- The Plotly Dashboard shows the total success launch for all sites and for a specific launch site;
- The Dashboard also shows a scatter plot with the correlation between Payload Mass in Kg for all sites and for a specific launch site;
- The URL of the completed Plotly Dash lab is publicly available at:
    - https://github.com/lodovico65/spacey/blob/main/spacex-dash-app.py

Predictive Analysis (Classification)

- We have built a machine learning pipeline to predict if the first stage of the Falcon 9 lands successfully.
- This has included:
  - Preprocessing, allowing us to standardize our data;
  - Train_test_split, allowing us to split our data into training and testing data,
  - Train the model and perform Grid Search, allowing us to find the hyperparameters that allow a given algorithm to perform best.
  - Using the best hyperparameter values for determine the model with the best accuracy using the training data.
  - We have tested Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbors.
  - Finally, we output the confusion matrix.
- The URL of the completed predictive analysis lab:
  - https://github.com/lodovico65/spacey/blob/main/Machine%20Learning%20Prediction.ipynb

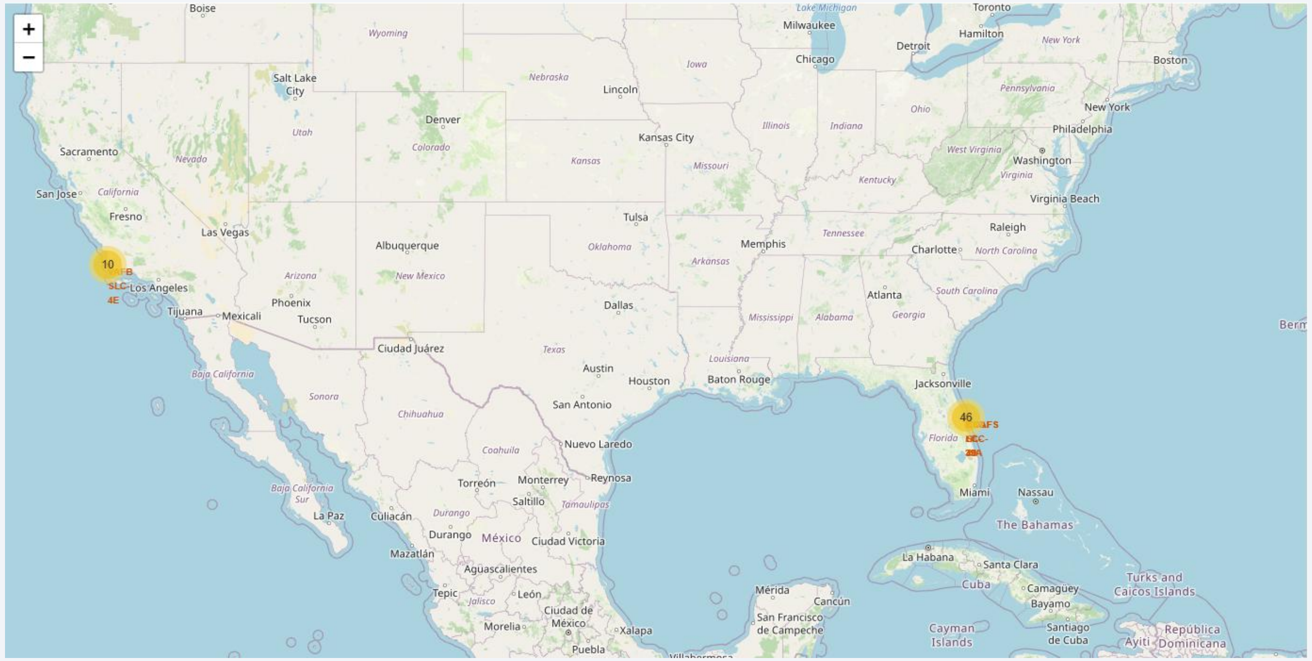
Results

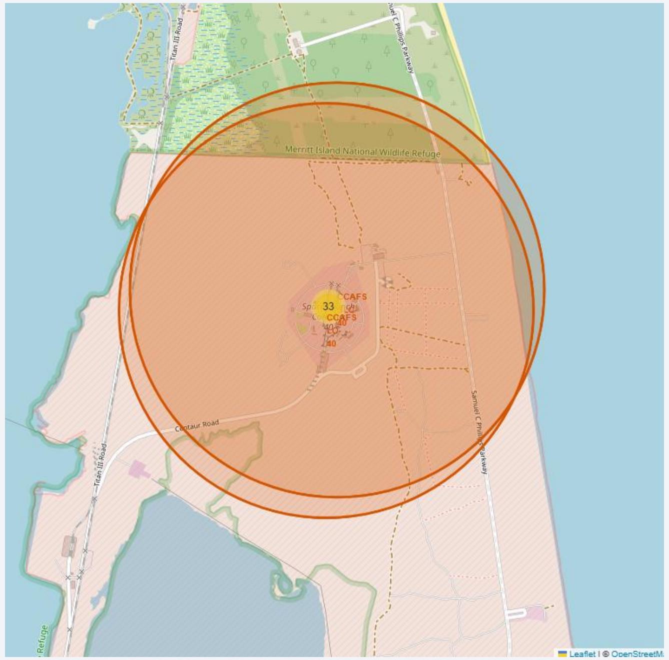
- Exploratory data analysis results
  - There are four launch sites in the SpaceX missions;
  - The average of Payload Mass in kg carried by booster version F9 v1.1 is 2928.4
  - The date of the first successful landing outcome in ground was achieved on December 22, 2015;
  - Many F9 booster with payload mass greater than the average have nevertheless been successful in drone ship;
  - The success rate is very high with a count of 100 success vs 1 failure;
  - The success rate since 2013 kept increasing till 2020.

Results

Interactive analytics in screenshots:

- Three launch sites are in close proximity to Equator line. One VAFB-SLC-4E is less close to the Equator line. All launch sites are in very close proximity to the coast;
- Most of the launches site are on the East coast;
- In the launch site proximity is easy to find railway, highway, coastline.



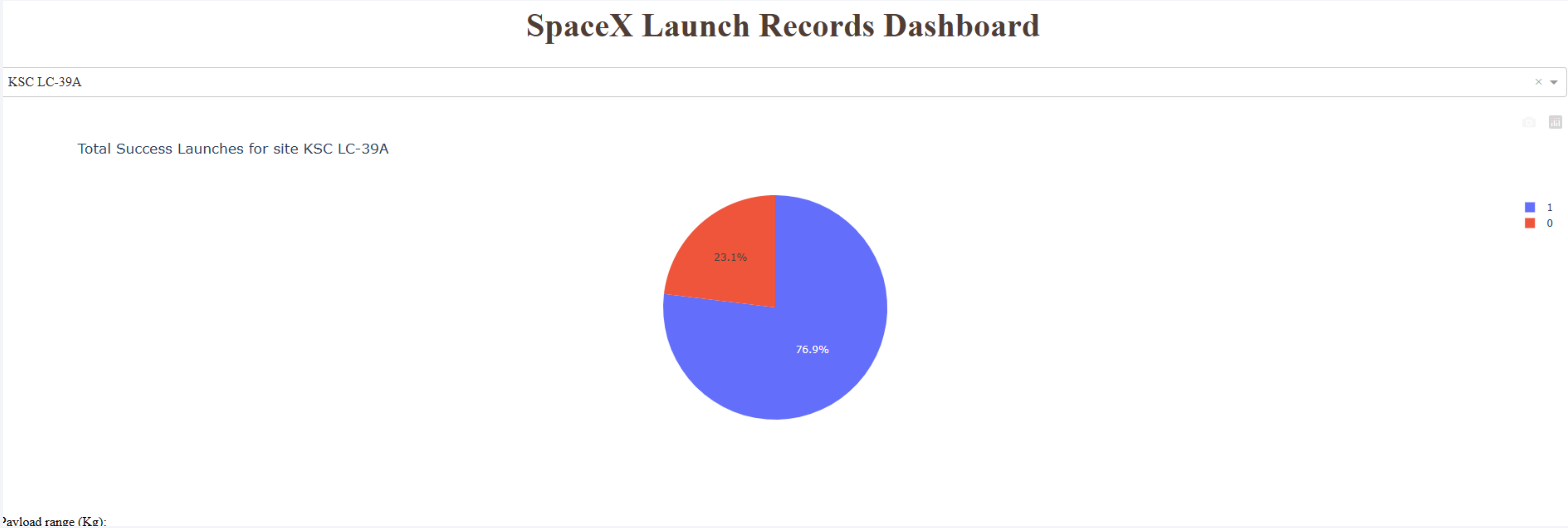


Merritt Island National Wildlife Refuge
33
CAFS
CCAFS
40
Leaflet | © OpenStreetMap

Results

Interactive analytics in screenshots:

- We found that the site with the greater success rate 76.9% is KSC LC-39A



SpaceX Launch Records Dashboard

KSC LC-39A

Total Success Launches for site KSC LC-39A

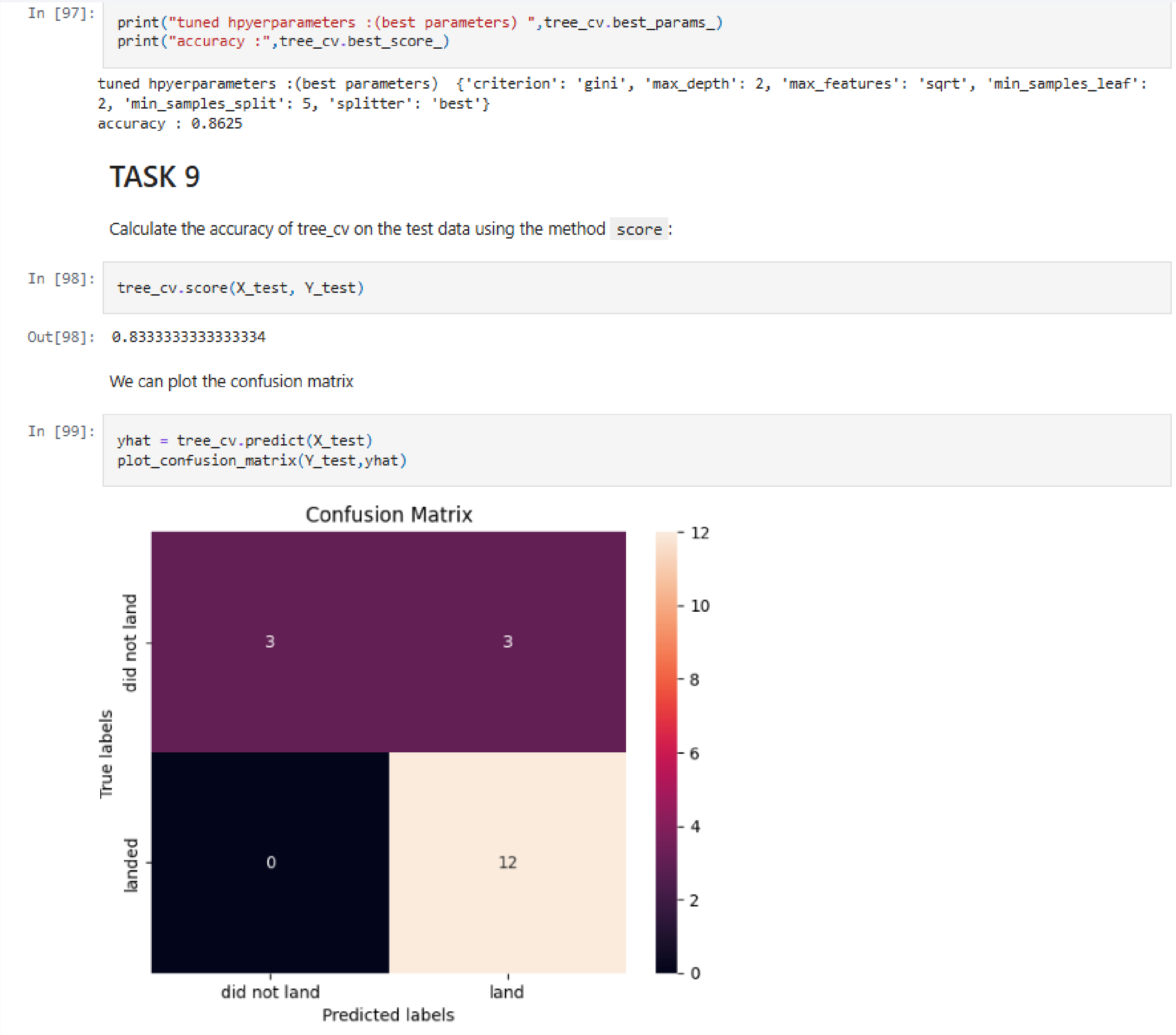
Payload range (Kg):

Results

Predictive analysis results:

In the predictive analysis phase, we have:

- Created a column for the class;
- Standardized the data;
- Split into training data and test data;
- Find best Hyperparameter for SVM, Classification Trees and Logistic Regression.
- Find that the Decision Tree is the method that performs best;



```
In [97]:  print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
          print("accuracy :",tree_cv.best_score_)

tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 2, 'max_features': 'sqrt', 'min_samples_leaf':
2, 'min_samples_split': 5, 'splitter': 'best'}
accuracy : 0.8625
```

TASK 9

Calculate the accuracy of tree_cv on the test data using the method `score`:

```
In [98]:  tree_cv.score(X_test, Y_test)

Out[98]:  0.8333333333333334
```

We can plot the confusion matrix

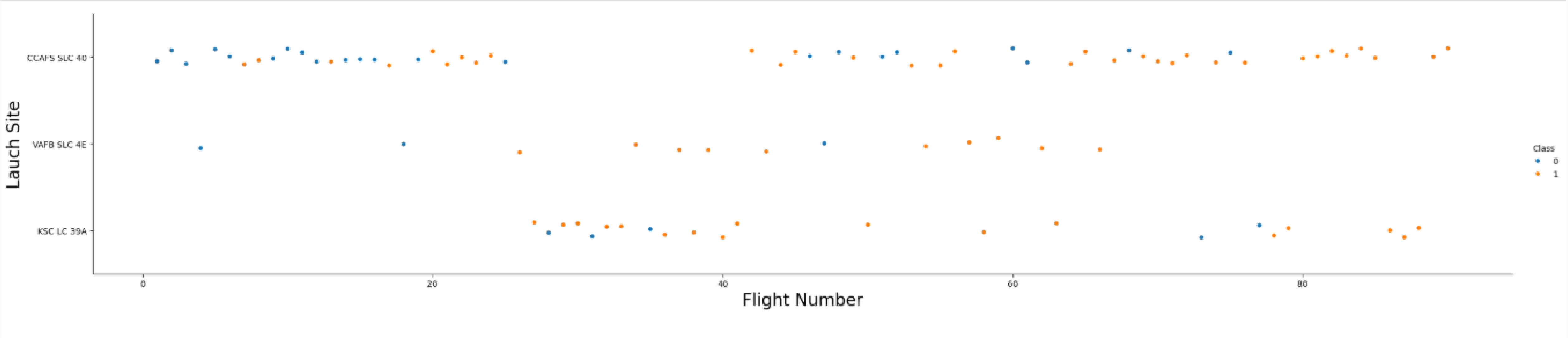
```
In [99]:  yhat = tree_cv.predict(X_test)
          plot_confusion_matrix(Y_test,yhat)
```

Section 2

Insights drawn from EDA

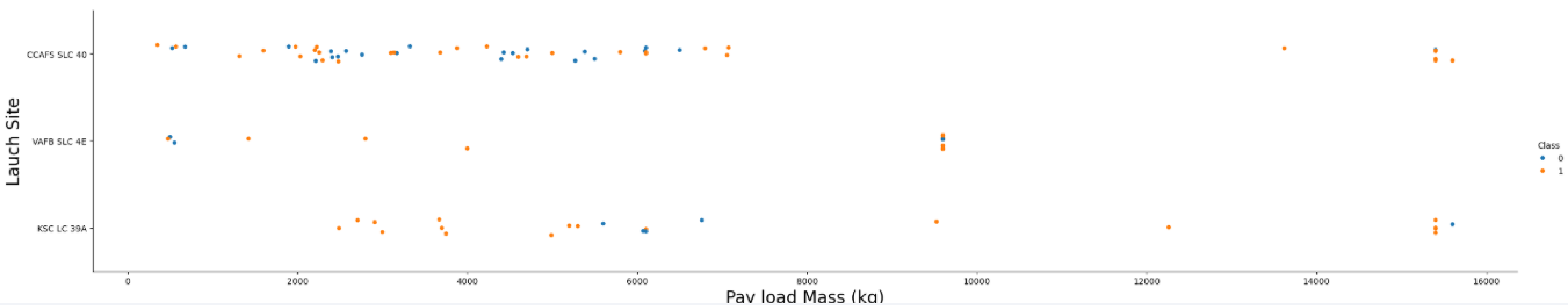
Flight Number vs. Launch Site

The chart shows that the success rate increases with the number of flights:



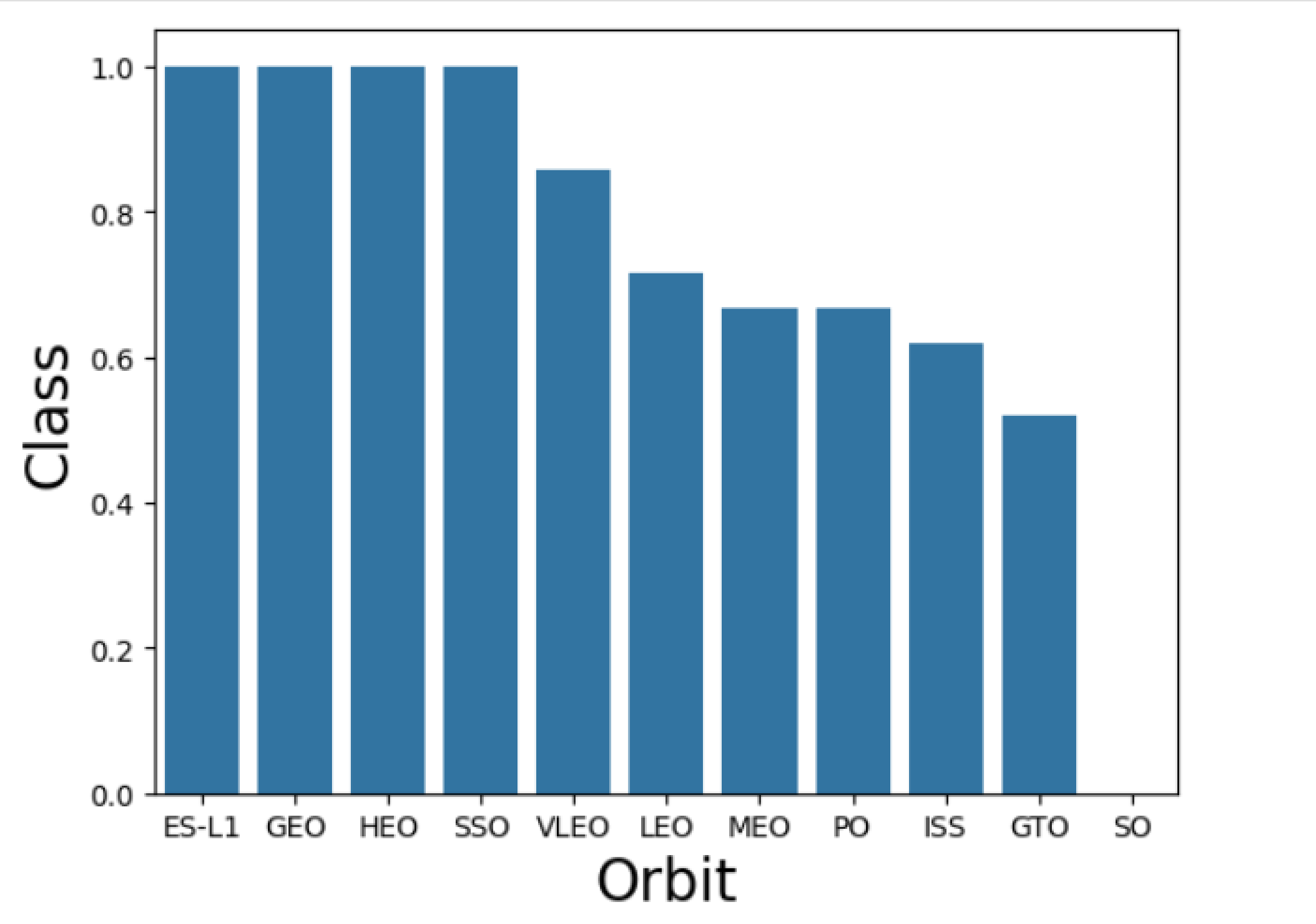
Payload vs. Launch Site

Examining Payload Mass Vs. Launch Site scatter point chart you will find that for the VAFB-SLC launch site there are no rockets launched for heavy payload mass (greater than 10000):



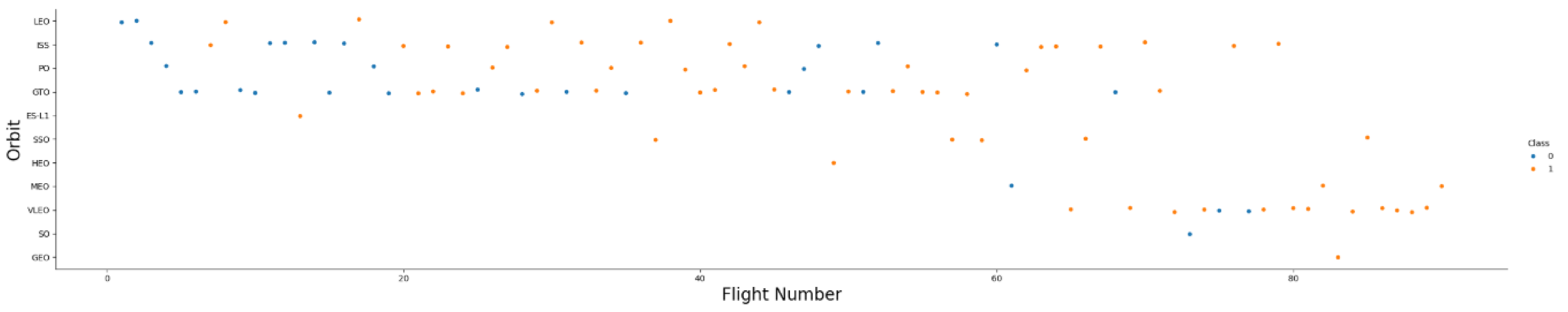
Success Rate vs. Orbit Type

The chart shows that the Orbit types with the highest success rate are: ES-L1, GEO, HEO, SSO.



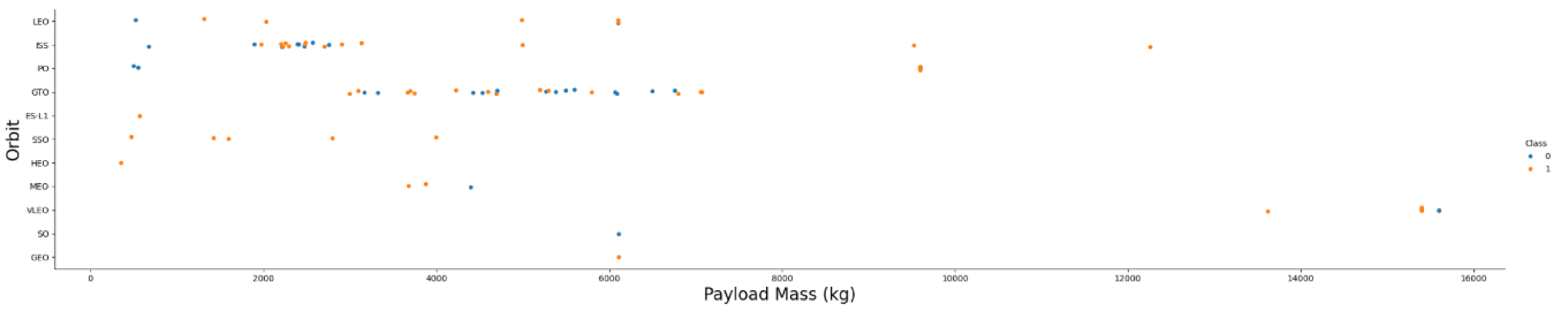
Flight Number vs. Orbit Type

The chart shows that 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.



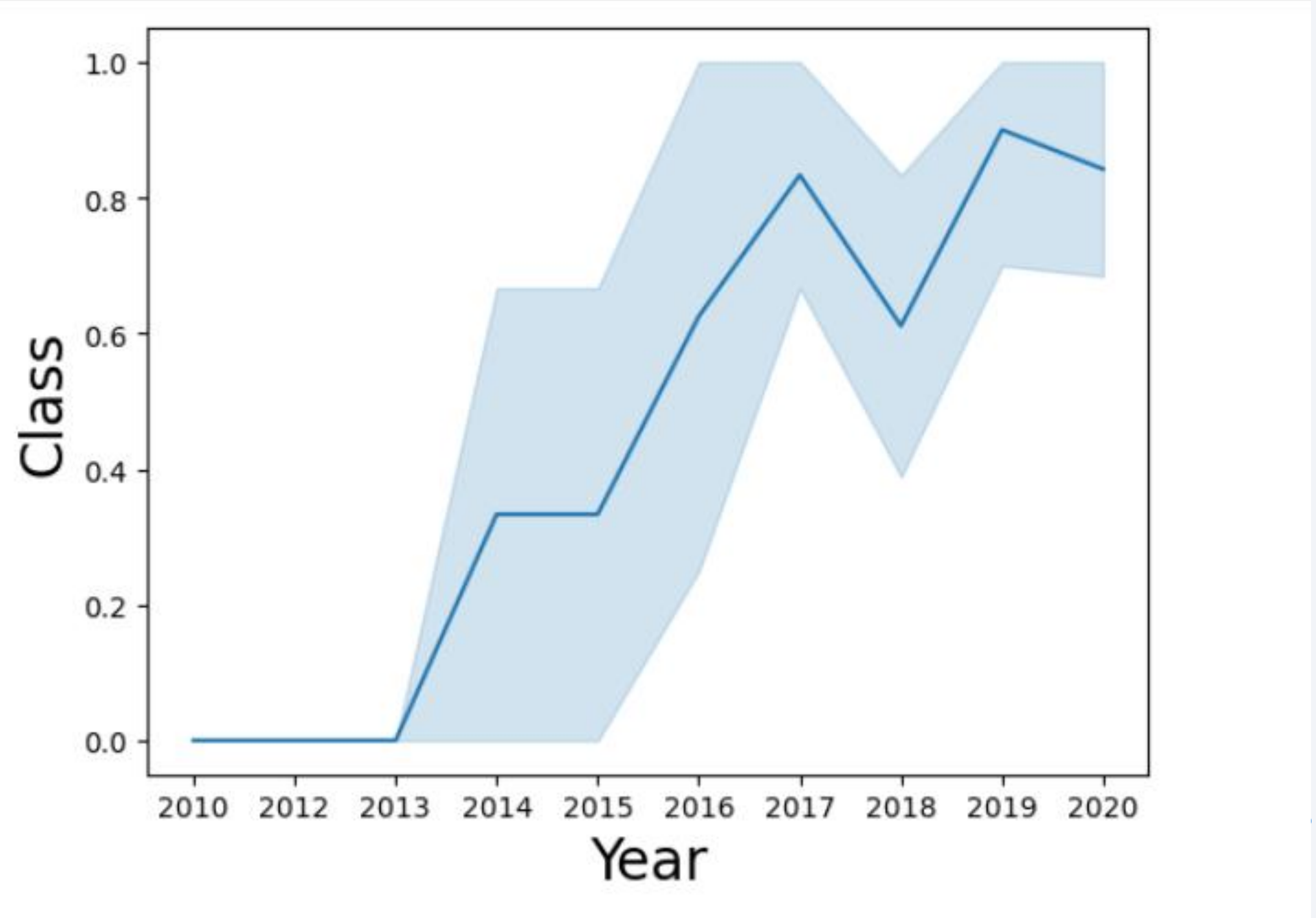
Payload vs. Orbit Type

The chart shows that with heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.



Launch Success Yearly Trend

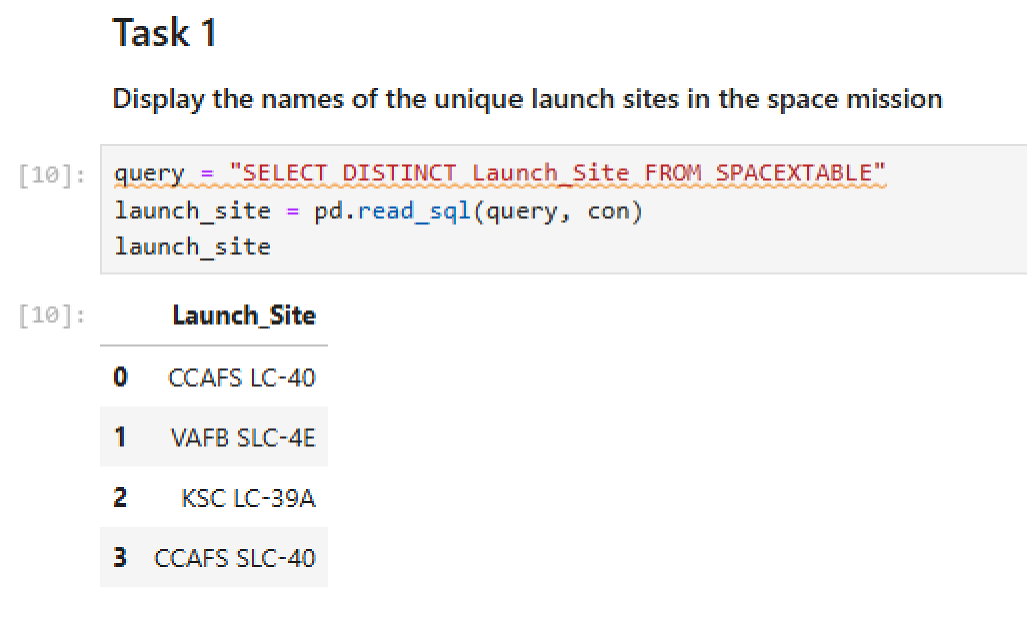
The chart shows that the success rate since 2013 kept increasing till 2020.



All Launch Site Names

Find the names of the unique launch sites:

- We have used the “DISTINCT” clause to retrieve a list of unique launch sites;



Task 1

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

```
[10]: query = "SELECT DISTINCT Launch_Site FROM SPACEXTABLE"
      launch_site = pd.read_sql(query, con)
      launch_site
```

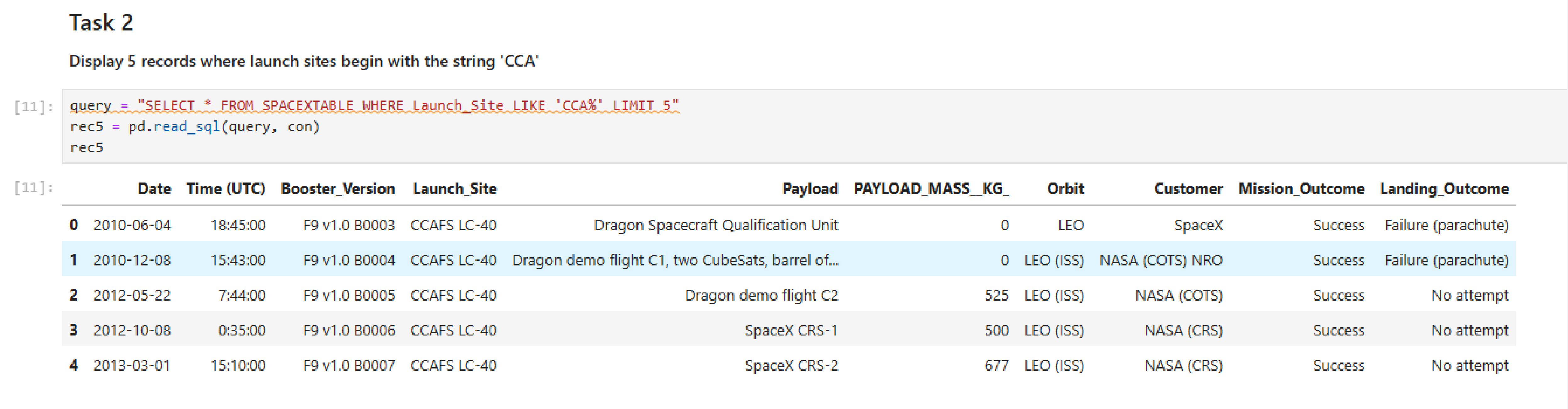
[10]:

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

Launch Site Names Begin with 'CCA'

Find 5 records where launch sites begin with `CCA`:

- We have used the LIKE operator for search based on a specific pattern in a column. We have also restricted the retrieve to 5 records using the LIMIT clause;



Task 2

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

```
[11]: query = "SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5"
      rec5 = pd.read_sql(query, con)
      rec5
```

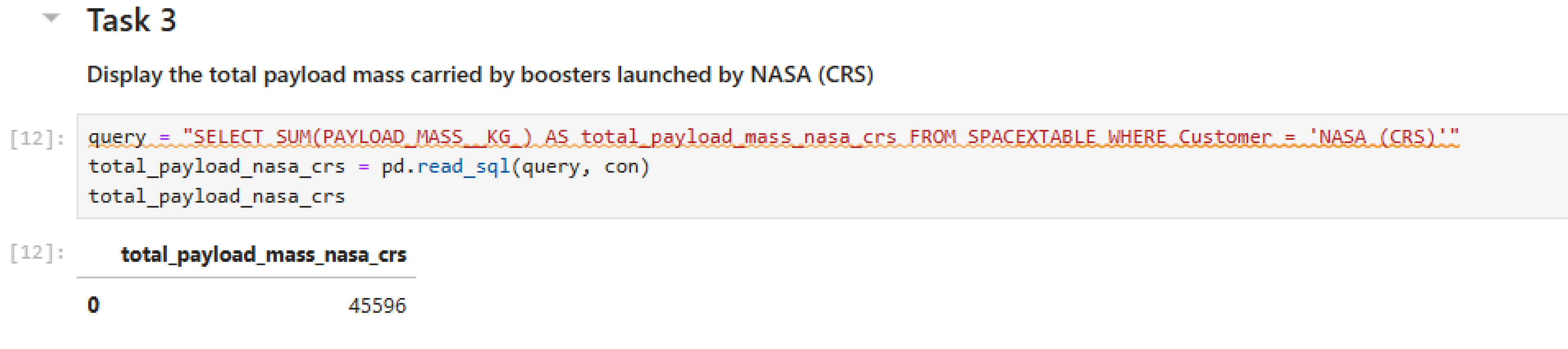
[11]:

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-12-08	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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	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

Calculate the total payload carried by boosters from NASA:

- We have used the SUM() function to calculate the sum of the Payload Mass in kg (equal to 45596) for the Customer ‘NASA (CRS);



Task 3

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

[12]:

```
query = "SELECT SUM(PAYLOAD_MASS__KG_) AS total_payload_mass_nasa_crs FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)'"
total_payload_nasa_crs = pd.read_sql(query, con)
total_payload_nasa_crs
```

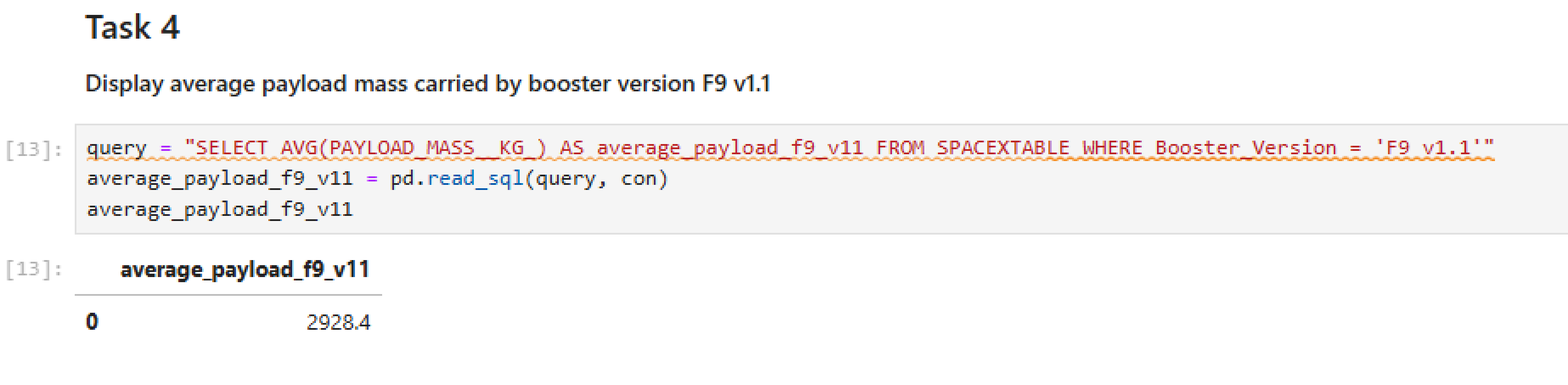
[12]:

	total_payload_mass_nasa_crs
0	45596

Average Payload Mass by F9 v1.1

Calculate the average payload mass carried by booster version F9 v1.1:

- We have used the AVG() function to calculate the mean of the Payload Mass in kg (equal to 2928.4) for the F9 v1.1 booster version;



Task 4

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

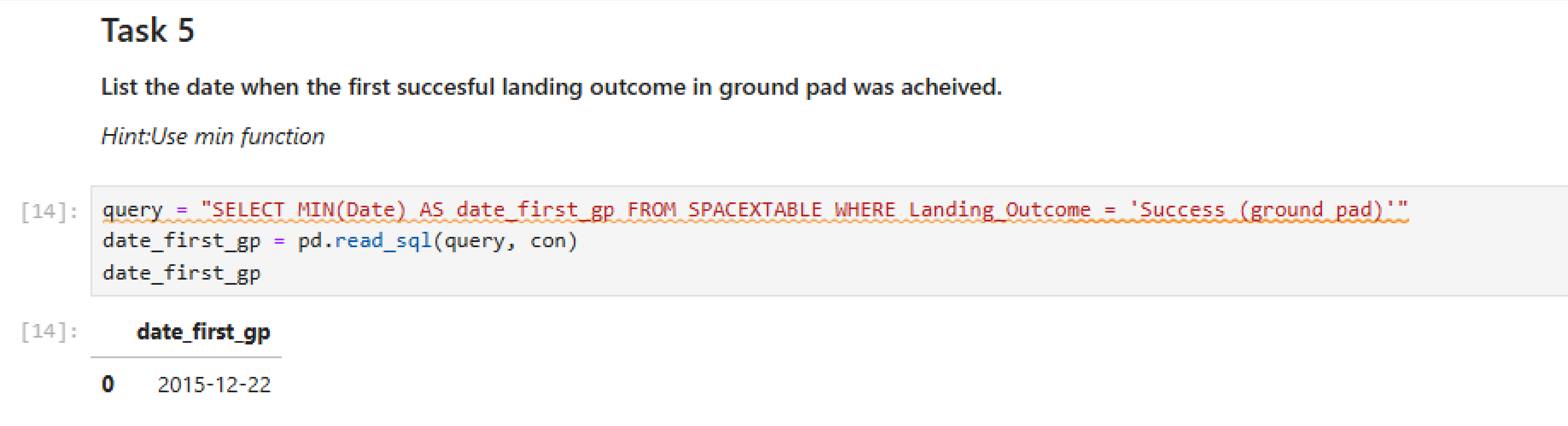
```
[13]: query = "SELECT AVG(PAYLOAD_MASS__KG_) AS average_payload_f9_v11 FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'"
      average_payload_f9_v11 = pd.read_sql(query, con)
      average_payload_f9_v11
```

```
[13]:    average_payload_f9_v11
      0                  2928.4
```

First Successful Ground Landing Date

Find the dates of the first successful landing outcome on ground pad:

- We have used the MIN() function to calculate the date (equal to 22 December 2015) of the first successful landing outcome on ground pad;



Task 5

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

Hint:Use min function

```
[14]: query = "SELECT MIN(Date) AS date_first_gp FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)'"
      date_first_gp = pd.read_sql(query, con)
      date_first_gp
```

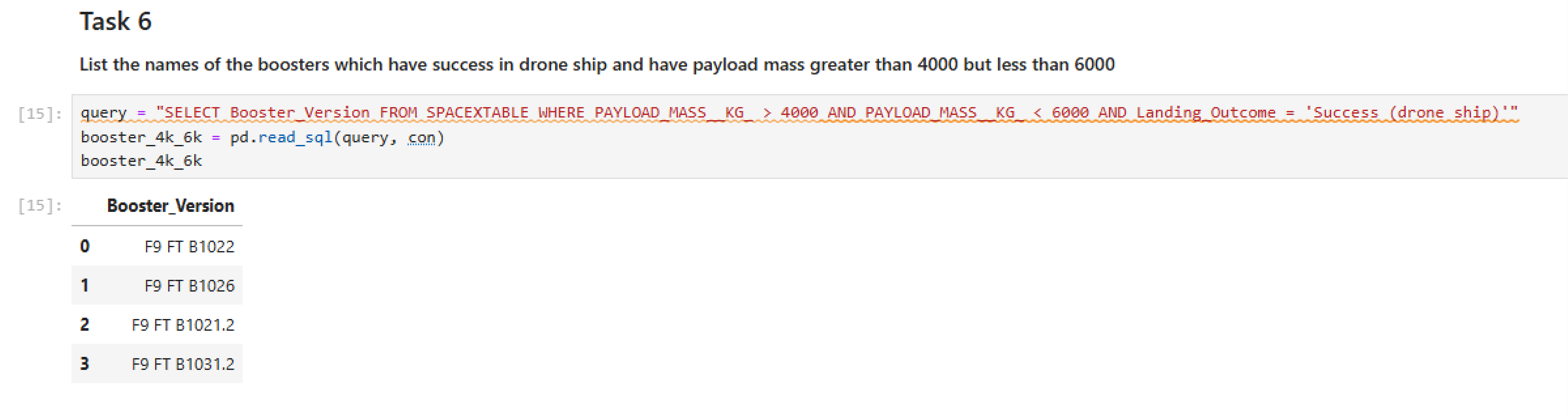
[14]:

	date_first_gp
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000:

- We have used the comparison operator ‘>’ (greater than) and ‘<‘ (less than) to retrieve the records with Payload Mass within the specified interval;



Task 6

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

```
[15]: query = "SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000 AND Landing_Outcome = 'Success (drone ship)'"
      booster_4k_6k = pd.read_sql(query, con)
      booster_4k_6k
```

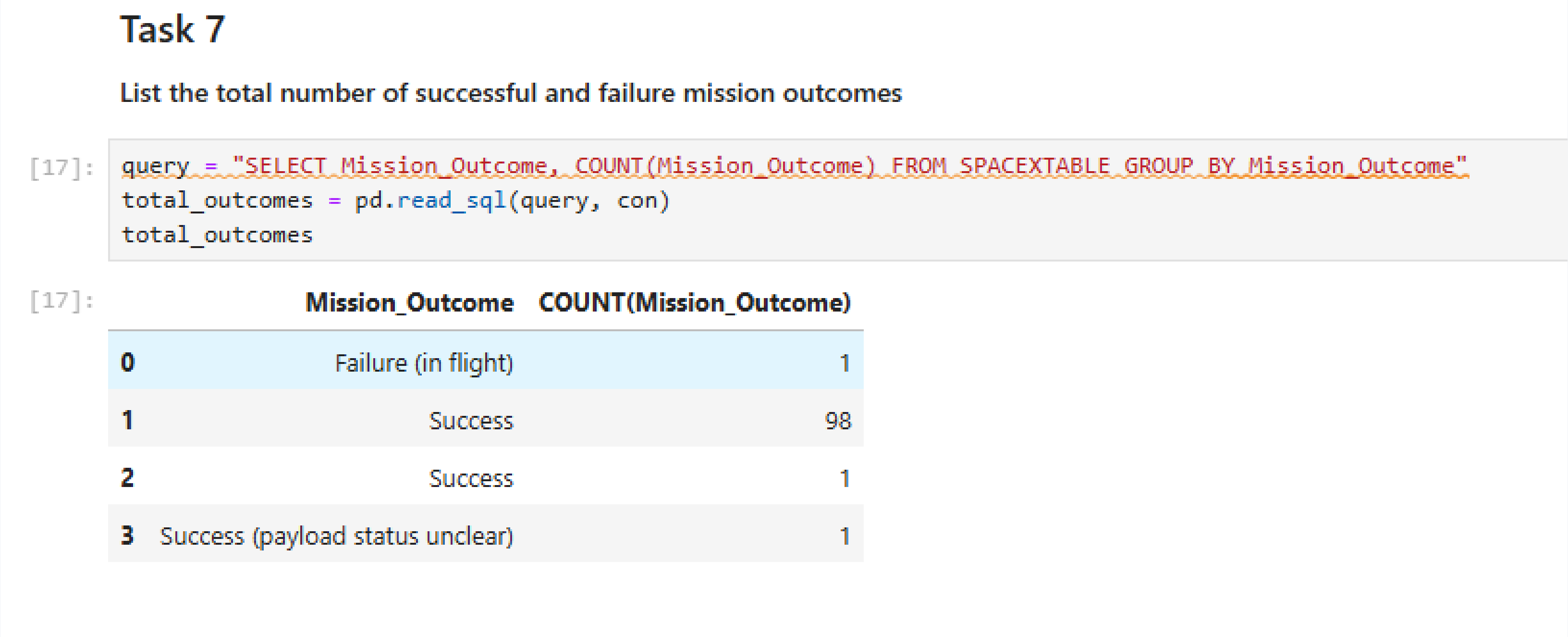
[15]:

	Booster_Version
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

Calculate the total number of successful and failure mission outcomes:

- We have used the aggregate statement GROUP BY to counts the occurrences of each Mission Outcome;



Task 7

List the total number of successful and failure mission outcomes

[17]:
```
query = "SELECT Mission_Outcome, COUNT(Mission_Outcome) FROM SPACEXTABLE GROUP BY Mission_Outcome"
total_outcomes = pd.read_sql(query, con)
total_outcomes
```

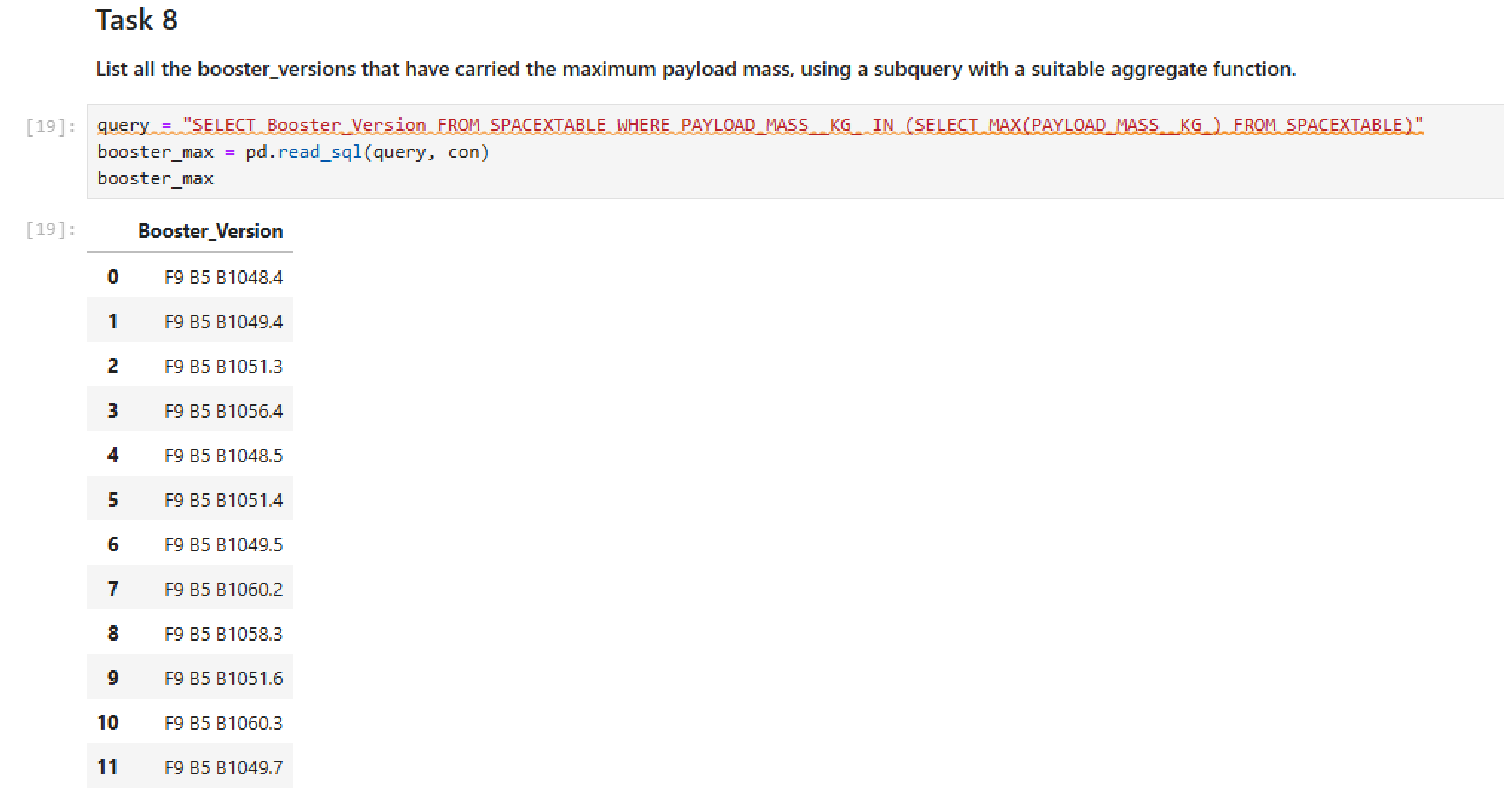
[17]:

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

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass:

- We have used a subquery with the MAX() function to retrieve the booster with the specified condition;



Task 8

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

```
[19]: query = "SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ IN (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)"
      booster_max = pd.read_sql(query, con)
      booster_max
```

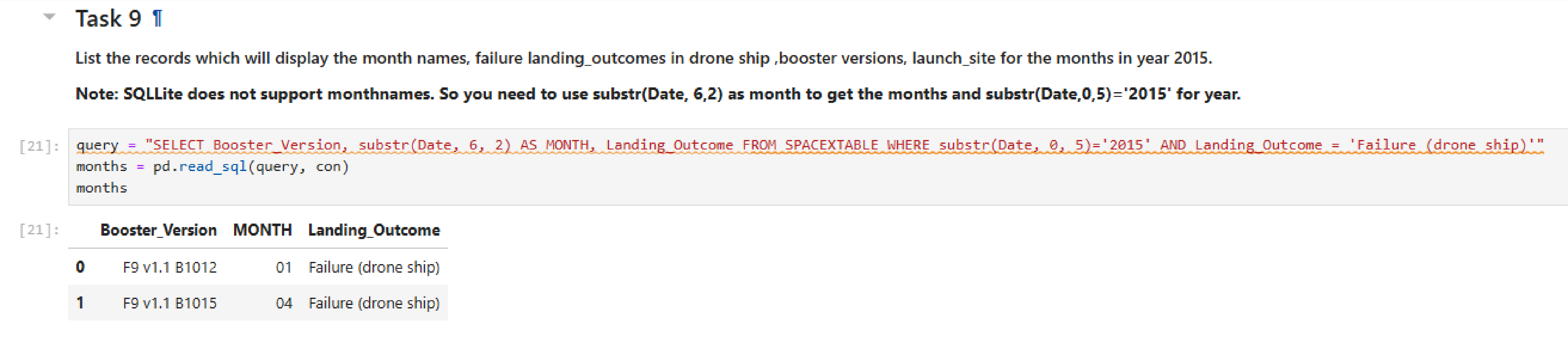
[19]:

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

2015 Launch Records

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

- We have used the substr() function to return the month portion of the Date;



Task 9 ¶

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.

[21]:
```
query = "SELECT Booster_Version, substr(Date, 6, 2) AS MONTH, Landing_Outcome FROM SPACEXTABLE WHERE substr(Date, 0, 5)='2015' AND Landing_Outcome = 'Failure (drone ship)'"
months = pd.read_sql(query, con)
months
```

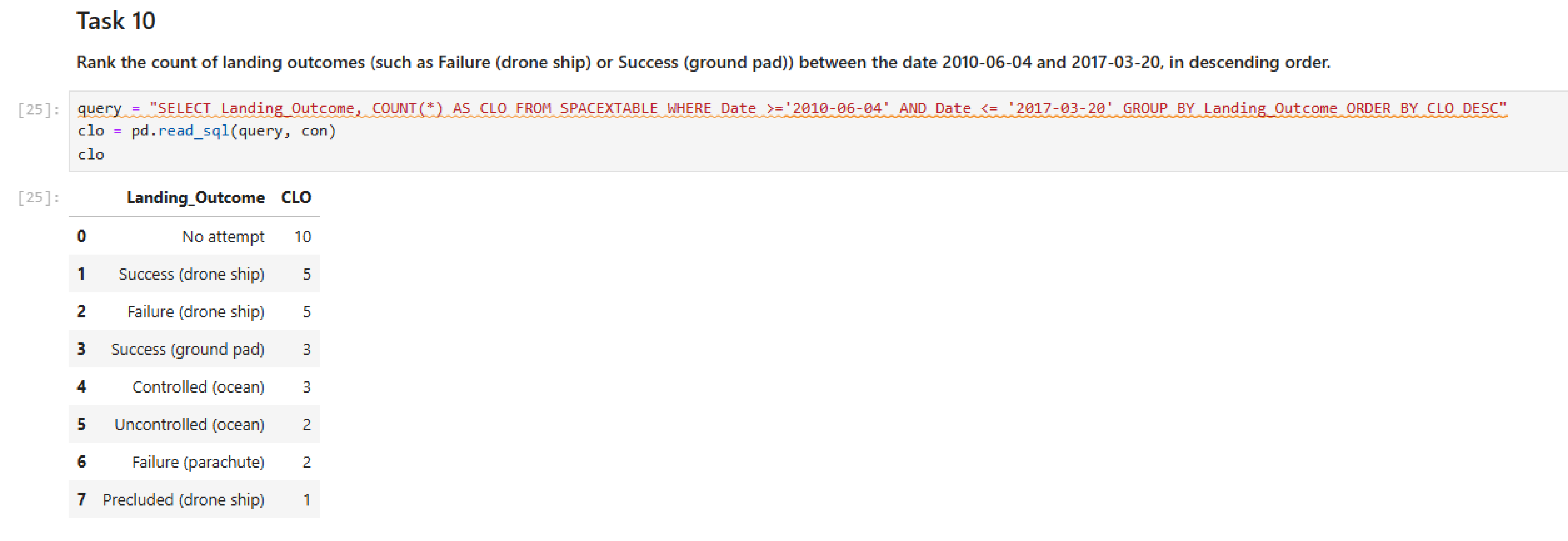
[21]:

	Booster_Version	MONTH	Landing_Outcome
0	F9 v1.1 B1012	01	Failure (drone ship)
1	F9 v1.1 B1015	04	Failure (drone ship)

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

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:

- We have used the aggregate statement GROUP BY and the COUNT() function to retrieve the count of landing outcomes;



Task 10

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.

[25]:

```
query = "SELECT Landing_Outcome, COUNT(*) AS CLO FROM SPACEXTABLE WHERE Date >='2010-06-04' AND Date <= '2017-03-20' GROUP BY Landing_Outcome ORDER BY CLO DESC"
clo = pd.read_sql(query, con)
clo
```

[25]:

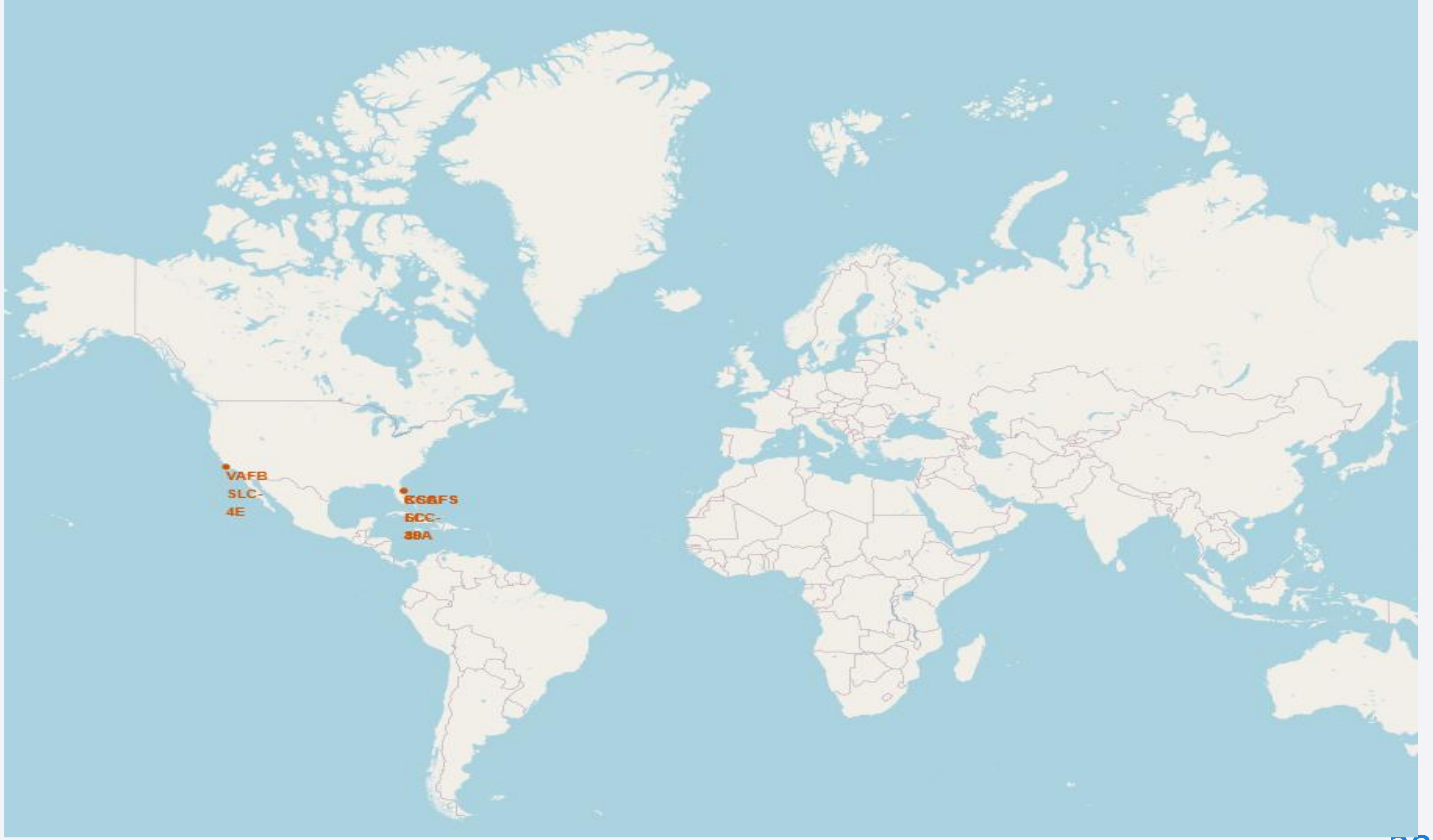
	Landing_Outcome	CLO
0	No attempt	10
1	Success (drone ship)	5
2	Failure (drone ship)	5
3	Success (ground pad)	3
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Failure (parachute)	2
7	Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

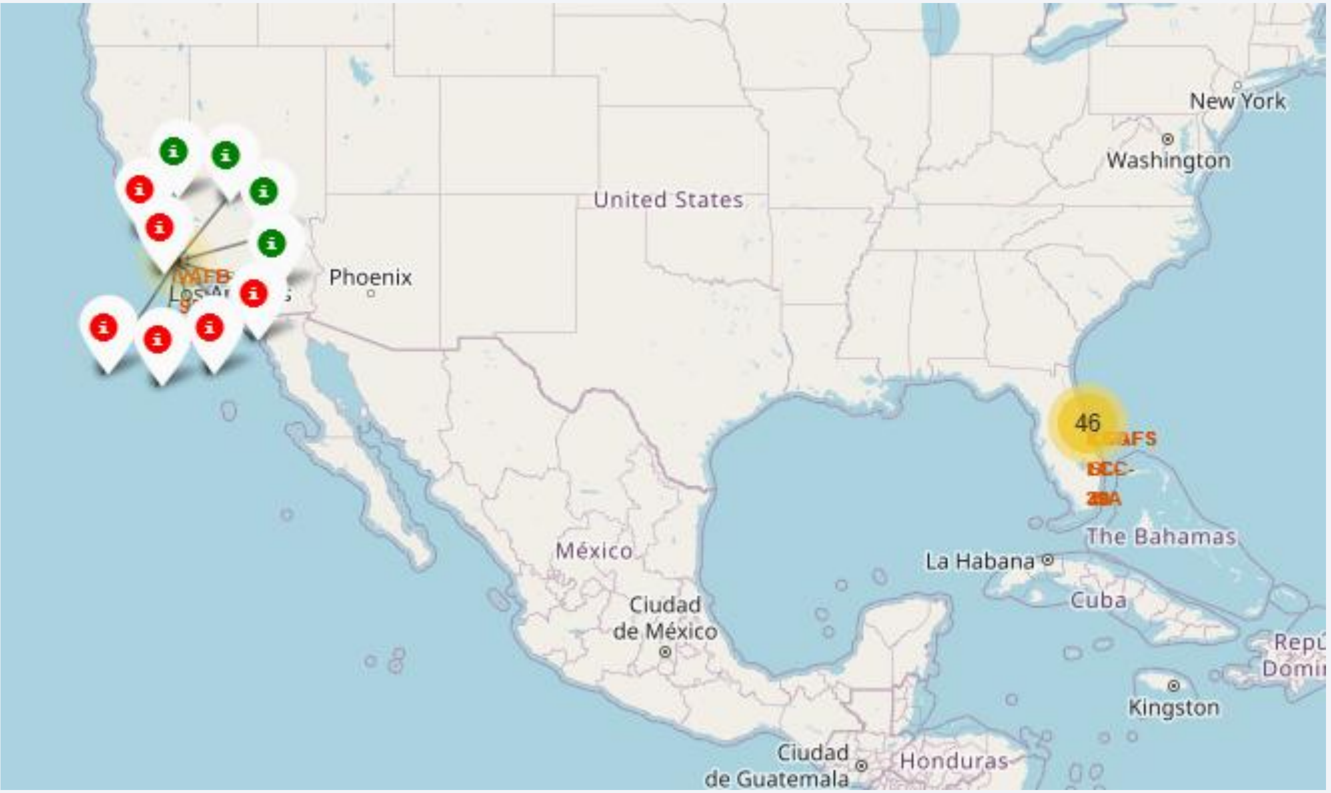
Launch sites location markers on global map

Three launch sites are in close proximity to Equator line. One VAFB-SLC-4E is less close to the Equator line. All launch sites are in very close proximity to the east and west coast.



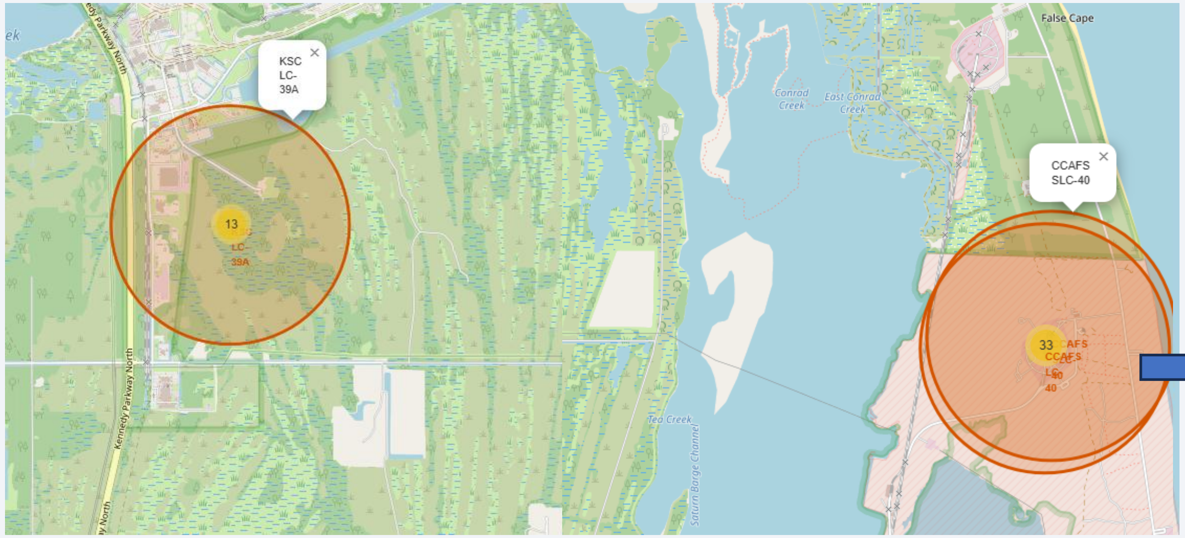
Map with color-labeled launch outcomes

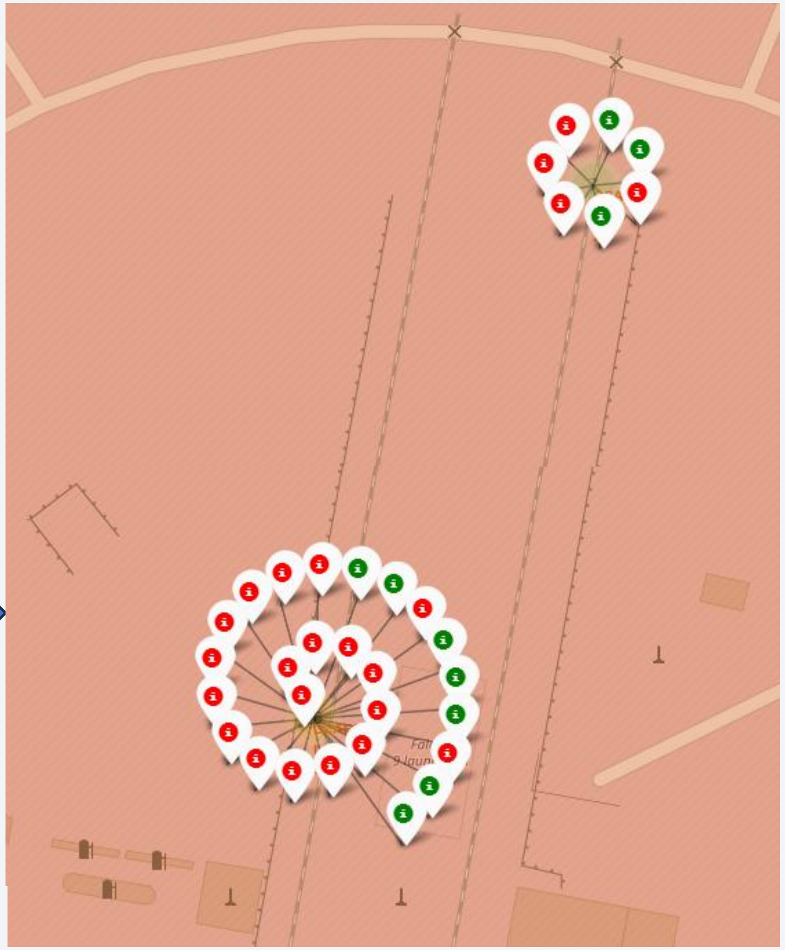
California launch sites. The green markers refers to successful landing, the red markers to unsuccessful landing.



Map with color-labeled launch outcomes

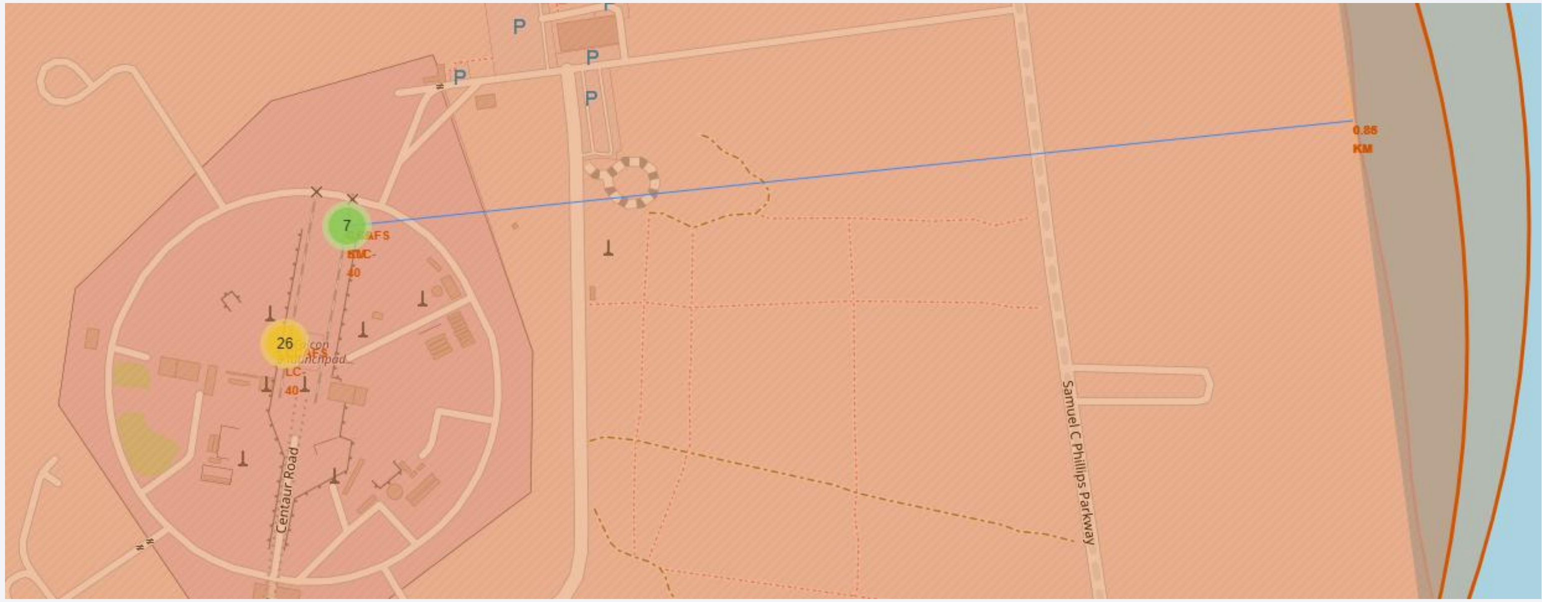
Florida launch sites. The green markers refers to successful landing, the red markers to unsuccessful landing.

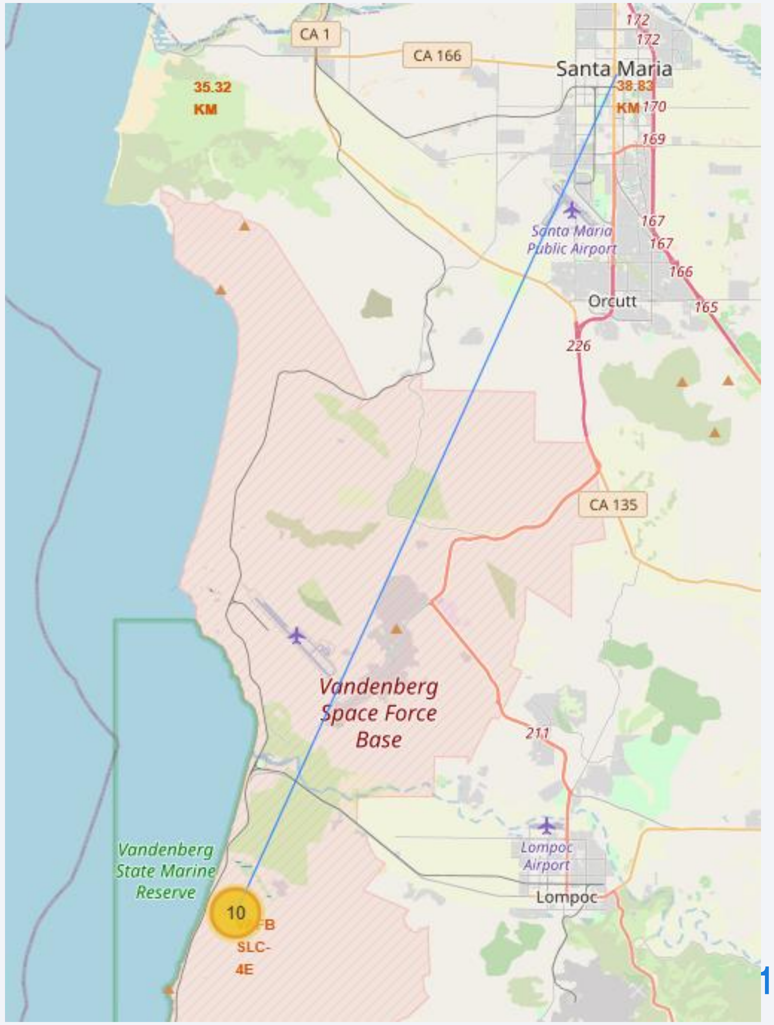




Launch sites proximities

The launches site are in close proximities to the coastline and keep
a certain distance away from the cities.

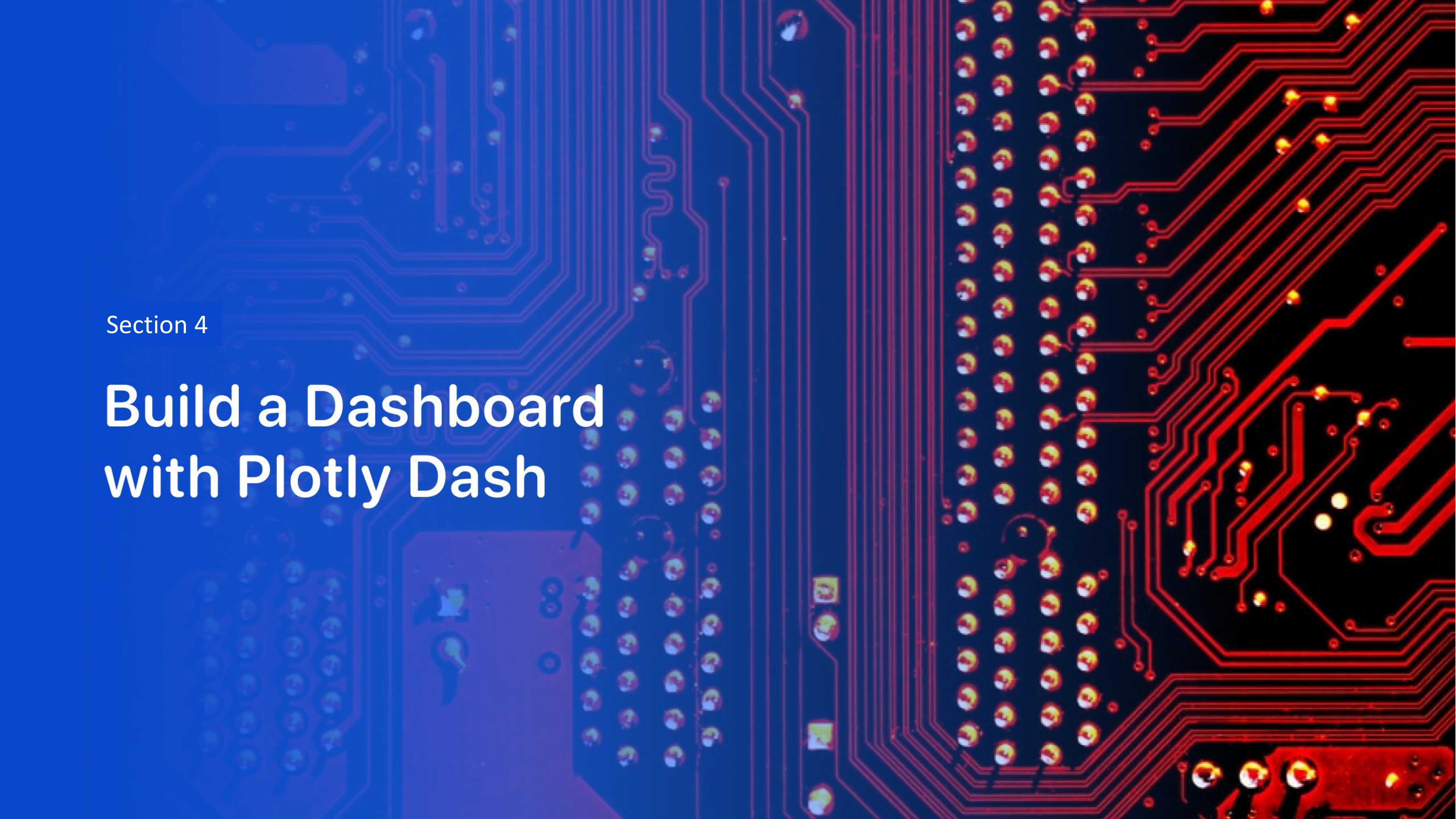




CA 1
CA 166
Santa Maria
35.32
KM
38.83
KM
Santa Maria Public Airport
Orcutt
CA 135
Vandenberg Space Force Base
Vandenberg State Marine Reserve
Lompoc Airport
Lompoc
10
SLC-
4E

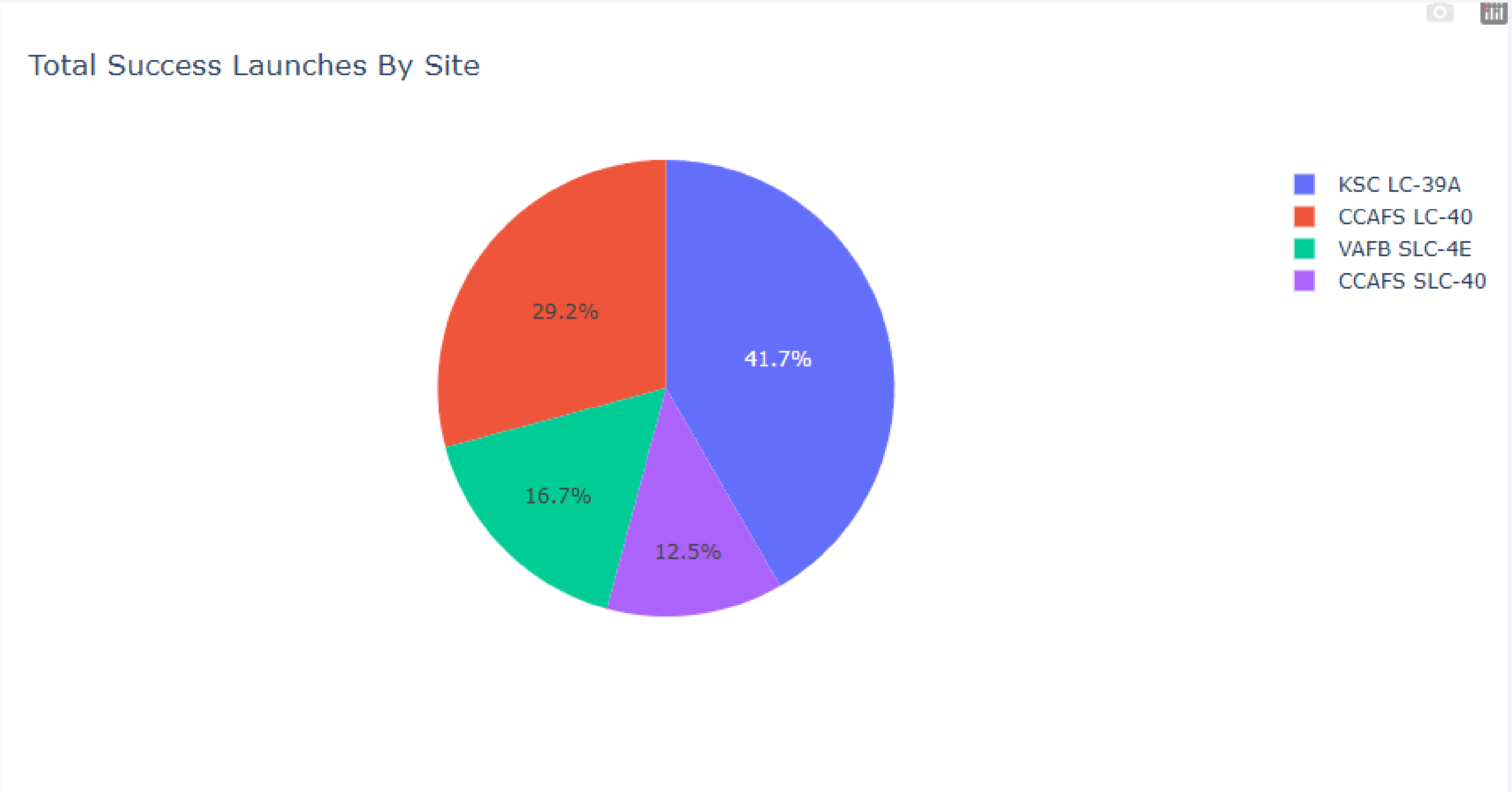
Section 4

Build a Dashboard with Plotly Dash



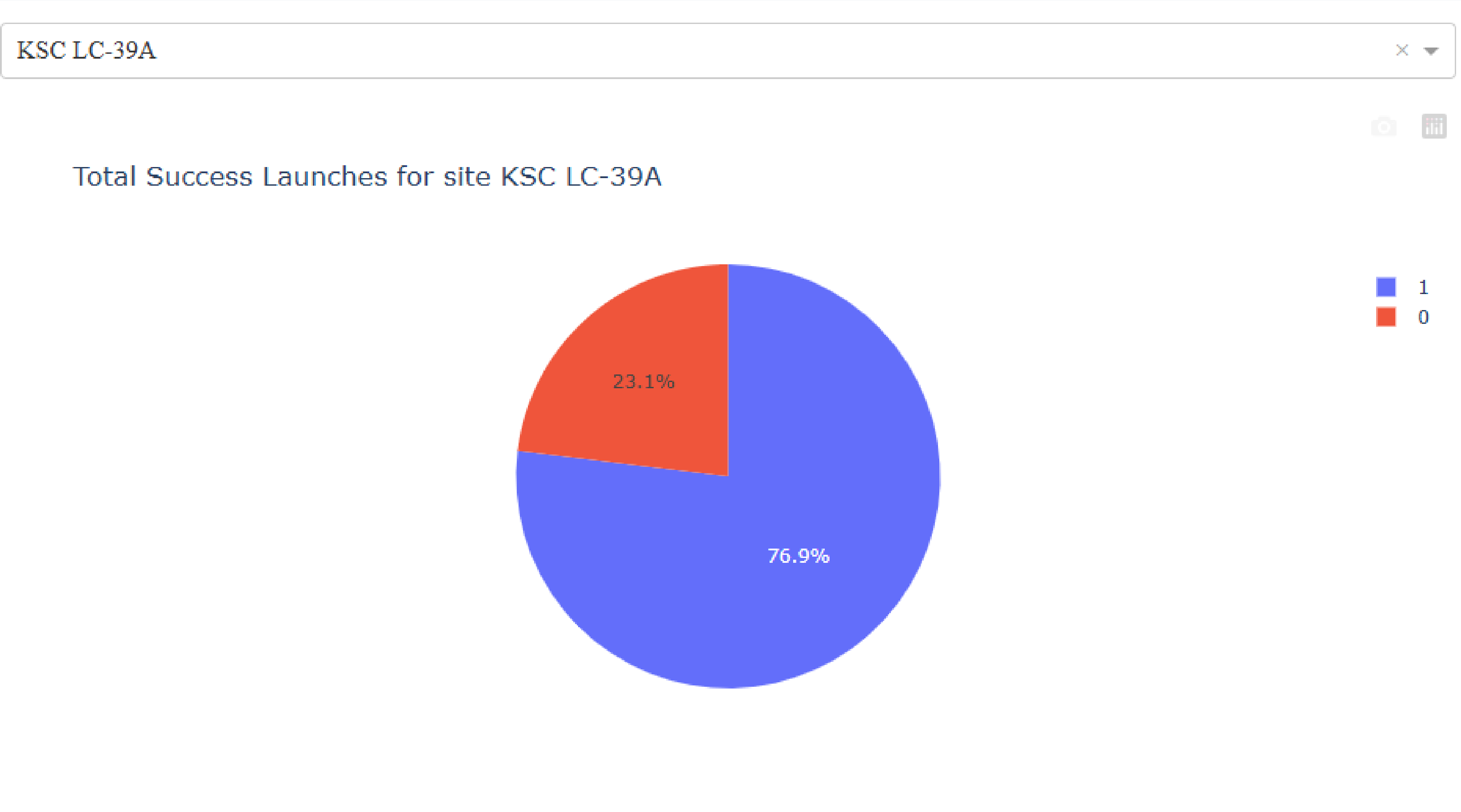
Pie chart with the launch success ratio for all sites

The pie chart shows that site KSC LC-39A has the higher successful rate.



Pie Chart with the highest successful rate

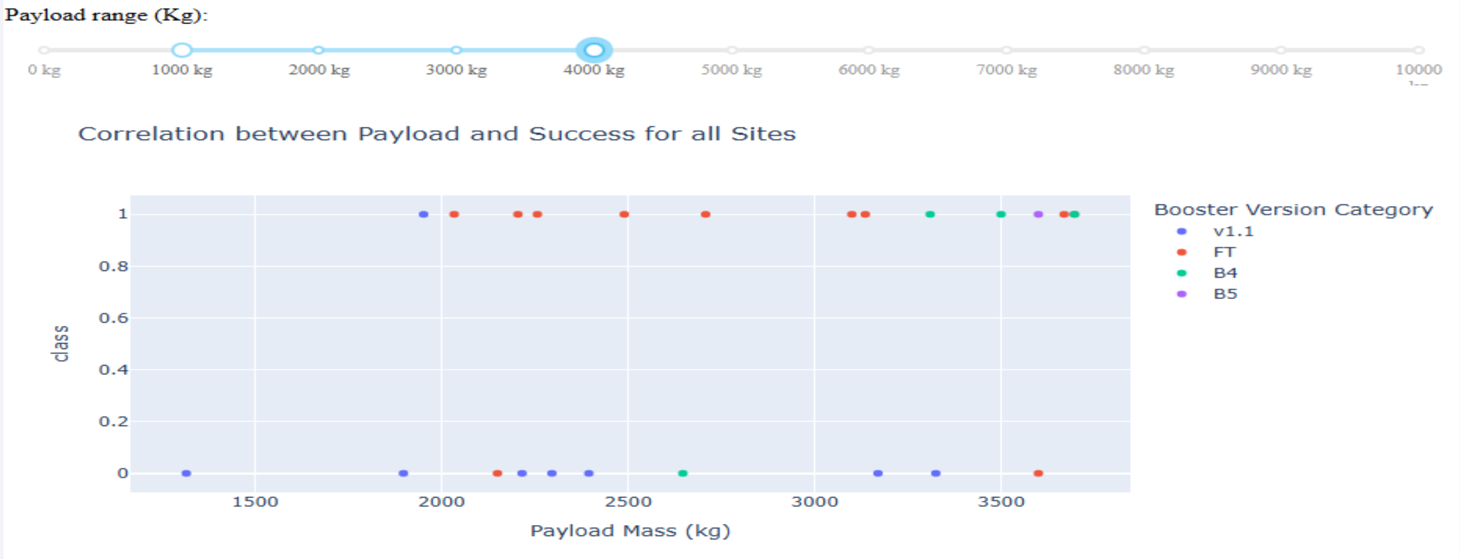
The Pie Chart shows that KSC LC39A site has the highest successful rate.

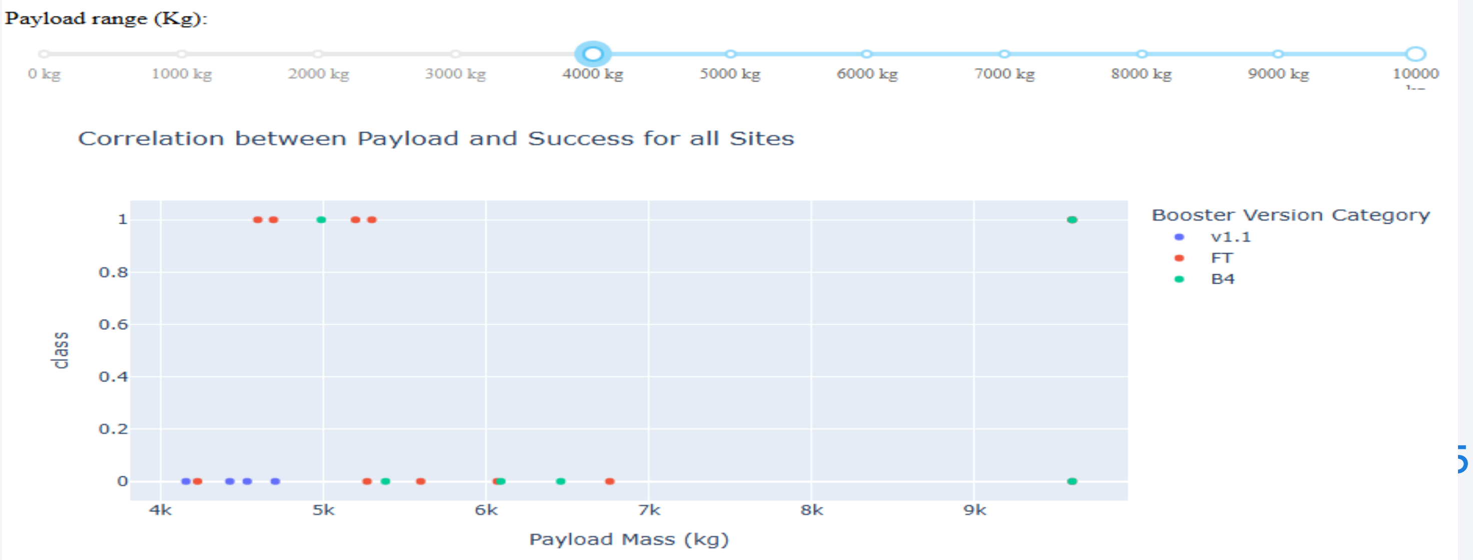


KSC LC-39A

Scatter plot for all sites with different payload selected in the range slider

The scatter plot for different range of payload mass shows that the success rate is higher for low weighted payload.

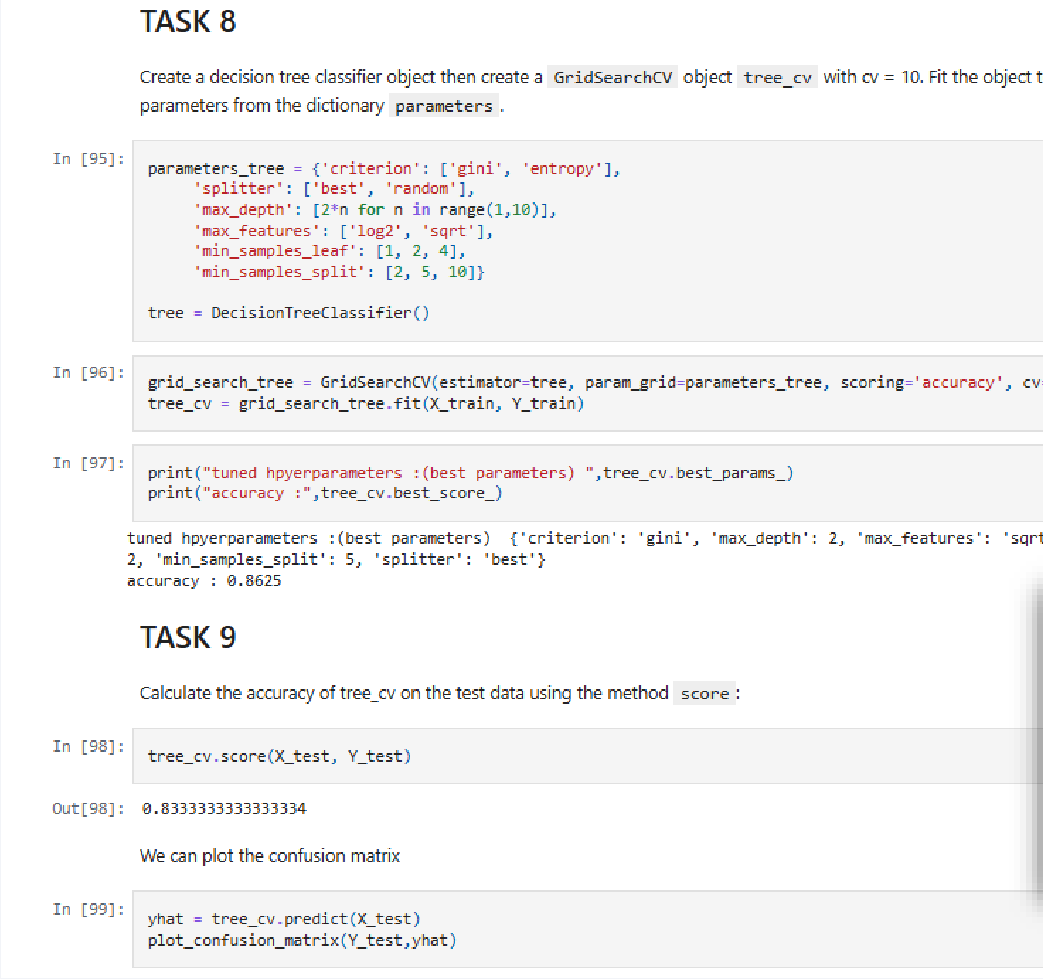




Section 5

Predictive Analysis (Classification)

Classification Accuracy



TASK 8

Create a decision tree classifier object then create a `GridSearchCV` object `tree_cv` with cv = 10. Fit the object to find the best parameters from the dictionary `parameters` .

```
In [95]:  parameters_tree = {'criterion': ['gini', 'entropy'],
               'splitter': ['best', 'random'],
               'max_depth': [2*n for n in range(1,10)],
               'max_features': ['log2', 'sqrt'],
               'min_samples_leaf': [1, 2, 4],
               'min_samples_split': [2, 5, 10]}

          tree = DecisionTreeClassifier()
```

```
In [96]:  grid_search_tree = GridSearchCV(estimator=tree, param_grid=parameters_tree, scoring='accuracy', cv=10)
          tree_cv = grid_search_tree.fit(X_train, Y_train)
```

```
In [97]:  print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
          print("accuracy :",tree_cv.best_score_)

         tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 2, 'max_features': 'sqrt', 'min_samples_leaf':
         2, 'min_samples_split': 5, 'splitter': 'best'}
         accuracy : 0.8625
```

TASK 9

Calculate the accuracy of tree_cv on the test data using the method `score` :

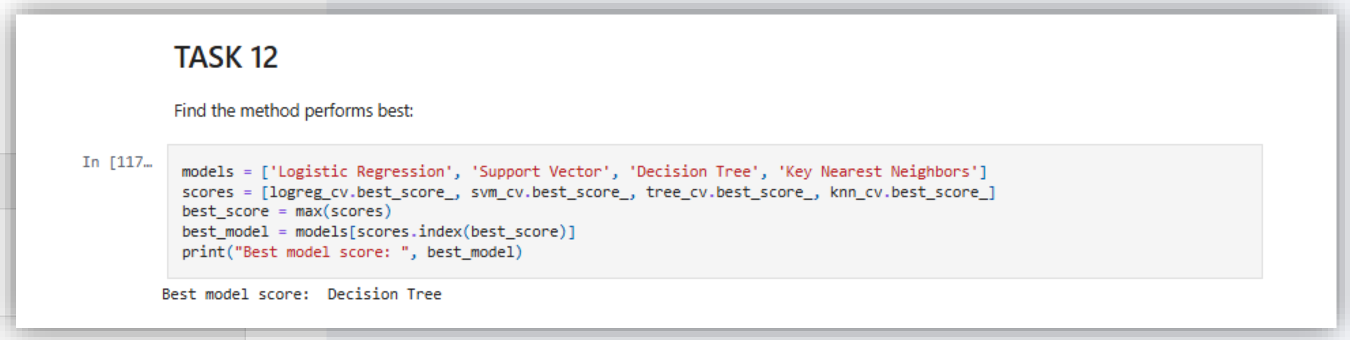
```
In [98]:  tree_cv.score(X_test, Y_test)

Out[98]:  0.8333333333333334
```

We can plot the confusion matrix

```
In [99]:  yhat = tree_cv.predict(X_test)
          plot_confusion_matrix(Y_test,yhat)
```

- On the left the accuracy for the Decision Tree model;
- The Decision Tree model is the model that perform better.



TASK 12

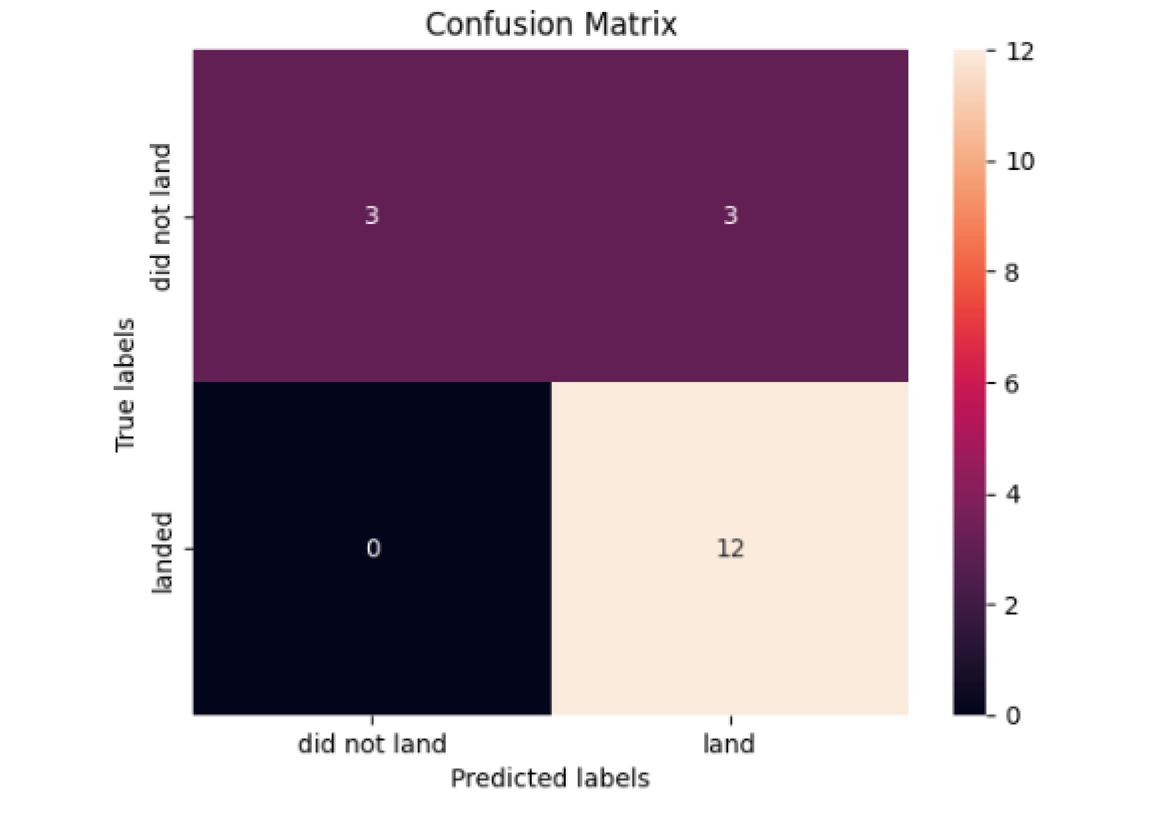
Find the method performs best:

```
In [117…  models = ['Logistic Regression', 'Support Vector', 'Decision Tree', 'Key Nearest Neighbors']
          scores = [logreg_cv.best_score_, svm_cv.best_score_, tree_cv.best_score_, knn_cv.best_score_]
          best_score = max(scores)
          best_model = models[scores.index(best_score)]
          print("Best model score: ", best_model)

         Best model score:  Decision Tree
```

Confusion Matrix

The confusion matrix of the best performing model with. The major drawback, like all the other models, are the false positives.



Conclusions

We can conclude that:

- The success rate is very high with a count of 100 success vs 1 failure;
- The Orbit types with the highest success rate are ES-L1, GEO, HEO, SSO;
- The success rate since 2013 kept increasing till 2020;
- All launch sites are in very close proximity to the east and west coast;
- The launches site are in close proximities to the coastline and keep a certain distance away from the cities;
- KSC LC39A site has the highest successful rate;
- The success rate is higher for low weighted payload;
- The Decision Tree is the method that performs best.

Thank you!