IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Fco. Alejandro Castilla Donis
20 de Mayo de 2025

OUTLINE

1.- EXECUTIVE SUMMARY

- 1.1 Data Collection API Lab
- 1.2 Data Collection with Web Scraping lab
- 1.3 Data Wrangling
- 1.4 Data Analysis with SQL
- 1.5 Data Analysis with Visualization Lab
- 1.6 Interactive Visual Analytics with Folium lab
- 1.7 Build an Interactive Dashboard with Ploty Dash
- 1.8 Machine Learning Prediction lab

INTRODUCTIO N

PROJECT FIRST STAGE LANDING PREDICTION PROJECT

USING DATA FROM FALCON 9 ROCKET LAUNCHES ADVERTISED ON SPACEX`S WEBSITE

- Project background and context
  - SpaceX advertises Falcon 9 rocket launches at 62 $ million
  - Other providers cost upwards of 165 $ million
  - SpaceX`s savings are due to the reuse of the first stage
  - Determining if the first stage lands allows for launch cost determination
  - This information is valuable for aleternative compaies wanting to bid against SapceX
- objective
  - To predict if The Falcon 9 first stage will land successfully

Section 1

Methodology

METHODOLOGY

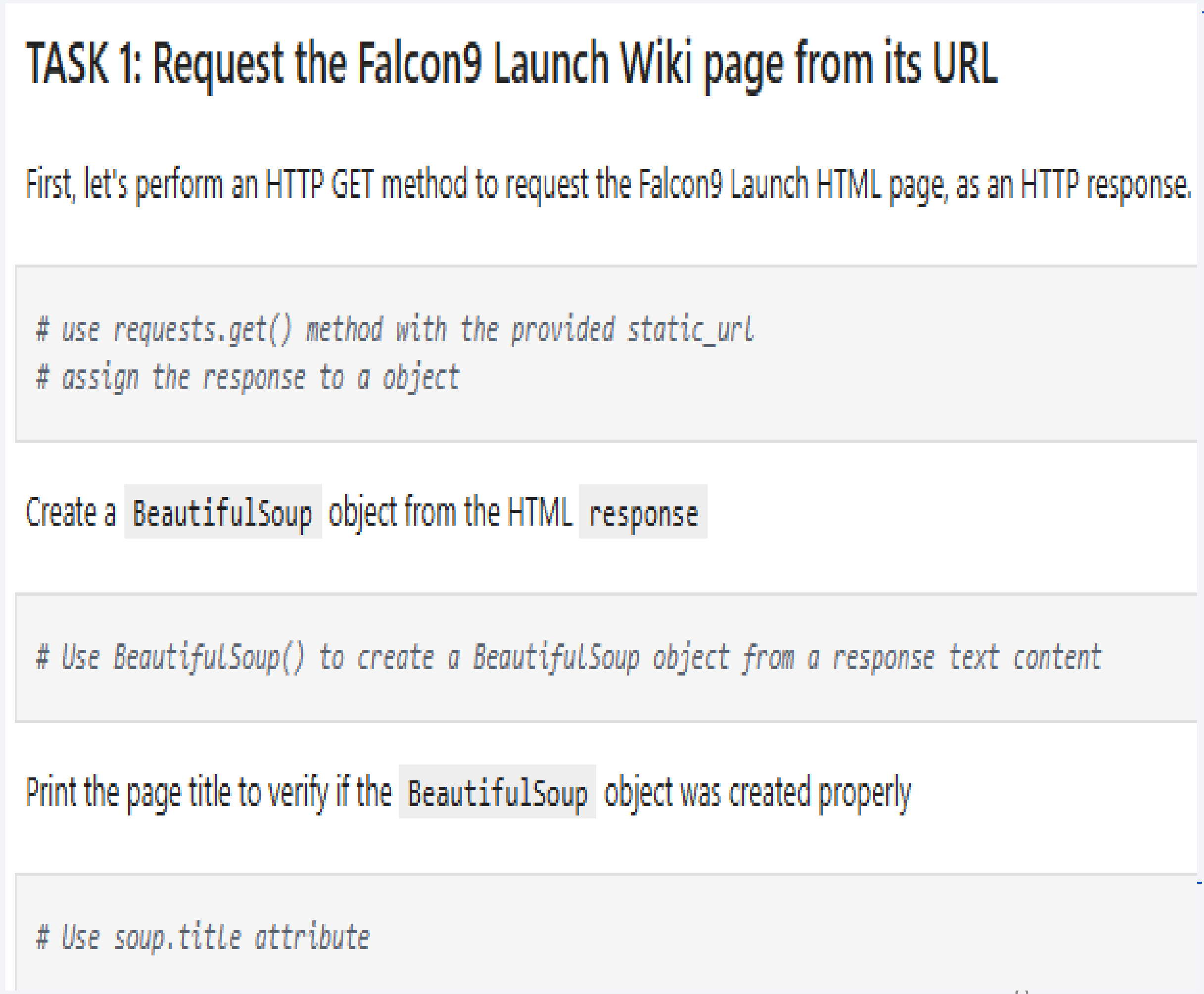
- Executive Summary
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

DATA COLLECTION

- DESCRIPTION OF HOW SPACEX FALCON9 DATA WAS COLLECTED.
    - DATA: COLLECTED USING SPACEX API (A RESTFUL API) BY MAKING A GET REQUEST TO THE SPACEX API. THIS WAS DONE BY FIRST DEFINING A SERIES HELPER FUNCTIONS THAT WOULD HELP IN THE USE OF THE API TO EXTRACT INFORMATION USING IDENTIFICATION NUMBERS IN THE LAUNCH DATA AND THEN REQUESTING ROCKET LAUNCH DATA FROM THE SPACEX API URL.
    - USING THE GET REQUEST AND THEN DECODED THE RESPONSE CONTENT AS A JSON RESULT WHICH WAS THEN CONVERTED INTO A PANDAS DATA FRAME.
- PERFORMED WEB SCRAPING TO COLLECT FALCON 9 HISTORICAL LAUNCH RECORDS FROM A WIKIPEDIA PAGE TITLED LIST OF FALCON 9 AND FALCON HEAVY LAUNCHES OF THE LAUNCH RECORDS ARE STORED IN A HTML. USING BEAUTIFULSOUP AND REQUEST LIBRARIES.
- EXTRACT THE FALCON 9 LAUNCH HTML TABLE RECORDS FROM THE WIKIPEDIA PAGE
- CONVERTED IT INTO A PANDAS DATA FRAME.

DATA COLLECTION

- Data collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API then requested and parsed the SpaceX launch data using the GET request and decoded the response content as a Json result which was then converted into a Pandas data frame
- *https://github.com/AlejandroCastilla72/Proyecto_Capstone_SpaceX/blob/main/3_Data_Collection_with_Web_Scraping_lab.ipynb*



TASK 1: Request the Falcon9 Launch Wiki page from its URL

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

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

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
```

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

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

Data Collection - Scraping

- Performed web scraping to collect Falcon 9 historical launch records from a Wikipedia page using BeautifulSoup and requests, to extract the Falcon 9 launch records from the HTML table of the Wikipedia page, then created a data frame by parsing the launch HTML.

- *https://github.com/AlejandroCastilla72/Proyecto_Capstone_SpaceX/blob/main/3_Data_Collection_with_Web_Scraping_lab.ipynb*

DATA WRANGLING

- After getting the data into a Pandas DataFrame:
- We filtered to keep only Falcon 9 launches (using the BoosterVersion column)
- Fixed missing values:
  - Left LandingPad as-is
  - Filled PayloadMass gaps with the column average
- Did some EDA to spot patterns and figure out what to use as our training label"
- *https://github.com/AlejandroCastilla72/Proyecto_Capstone_SpaceX/blob/main/4_Data_Wrangling.ipynb*

EDA WITH DATA VISUALIZATION

- Alright, so here's what I did with the SpaceX data:
- First, dumped all the raw data into a Pandas DF - easy peasy
- Filtered that bad boy to only show Falcon 9 launches (thanks BoosterVersion column!)
- Then dealt with those annoying missing values:
  - For LandingPad? Meh, left 'em blank
  - PayloadMass? Just slapped in the average value
- Did my usual EDA magic to:
  - Spot any cool patterns
  - Figure out what to use as our 'did it land?' label
  - Basically just poked around until the data made sense

- *https://github.com/AlejandroCastilla72/Proyecto_Capstone_SpaceX/blob/main/6_EDA_with_Visualization_Lab.ipynb*

EDA with SQL

- **Key Activities:**
- **Exploratory Data Analysis**
    - Performed initial data assessment and cleaning
    - Identified patterns and relationships in the dataset
- **Feature Engineering**
    - Created derived features to enhance predictive power
    - Handled missing values and data quality issues
    - **Visual Analysis Performed:**
- Scatter plots analyzing:
  • Flight Number vs Launch Site relationships
  • Payload mass distribution across launch sites
  • Flight patterns by orbit type
  • Payload capacity by orbit type
- Success rate analysis:
  • Bar charts comparing success rates across orbit types
  • Line charts tracking yearly success trends
- **Documentation:**
  The complete analysis with code is available in the Jupyter notebook here:
- https://github.com/AlejandroCastilla72/Proyecto_Capstone_SpaceX/blob/main/5_EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- I developed an interactive geographical analysis using Folium to:
- Map all SpaceX launch sites with precise location markers
- Visualize launch outcomes through:
    - Color-coded markers (success/failure)
    - Custom radius circles
    - Connecting lines for spatial relationships
- Implement a binary classification system:
    - Success = 1
    - Failure = 0
    - **Documentation:**
      The complete interactive map with all implementation details is available for review:

    - *https://github.com/AlejandroCastilla72/Proyecto_Capstone_SpaceX/blob/main/7_Interactive_Visual_Analytics_with_Folium_lab.ipynb*

Build a Dashboard with Plotly Dash

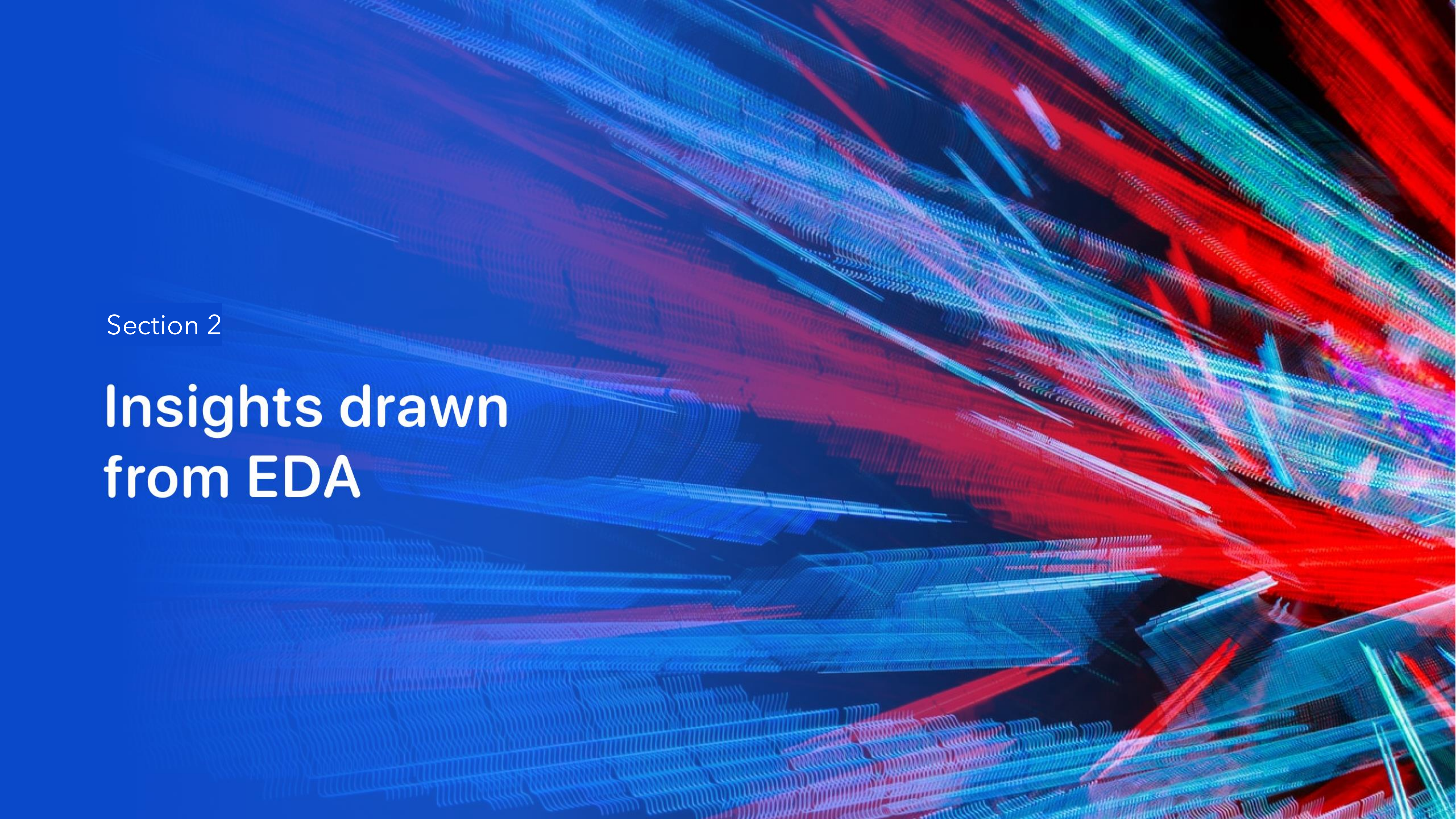
- Built an interactive dashboard application with Plotly Dash by:
  - Adding a Launch Site Drop-down Input Component
  - Adding a callback function to render success-pie-chart based on selected site dropdown
  - Adding a Range Slider to Select Payload
  - Adding a callback function to render the success-payload-scatter-chart scatter plot
- Here is the GitHub URL of your completed Plotly Dash lab: (*https://github.com/p.......................y*)

PREDICTIVE ANALYSIS (Classification)

- "I've covered everything from EDA to model implementation, identifying the most effective method for predicting rocket reusability."
- **Models Evaluated:**
- **Logistic Regression:** Optimized hyperparameters using GridSearchCV (accuracy ~83%)
- **SVM:** Tuned with kernels and C/gamma parameters (accuracy ~83%)
- **Decision Trees:** Depth and criteria optimization (accuracy ~94%)
- **KNN:** Neighbor and algorithm adjustment (accuracy ~83%)
- **Key Steps:**
- Confusion matrices: Identified false positives/negatives for each model
- Evaluation: Accuracy comparison on test data, with decision trees performing best (94%)

Results

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

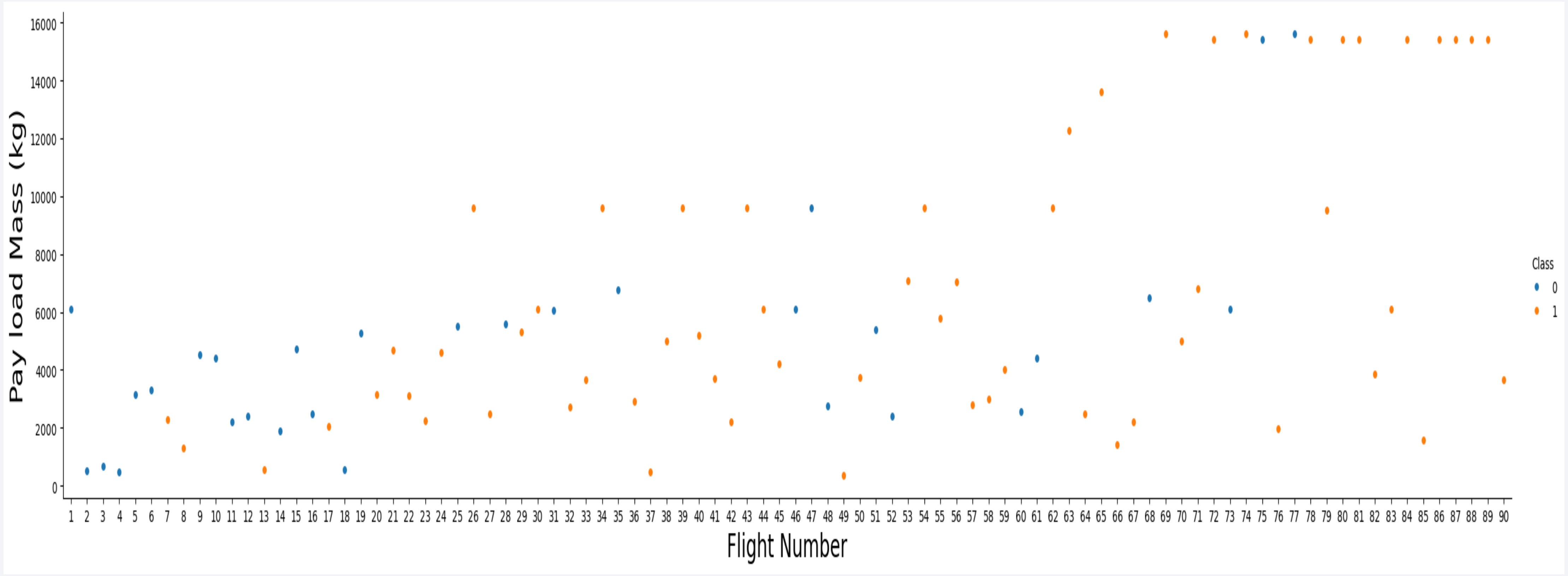


Section 2

Insights drawn from EDA

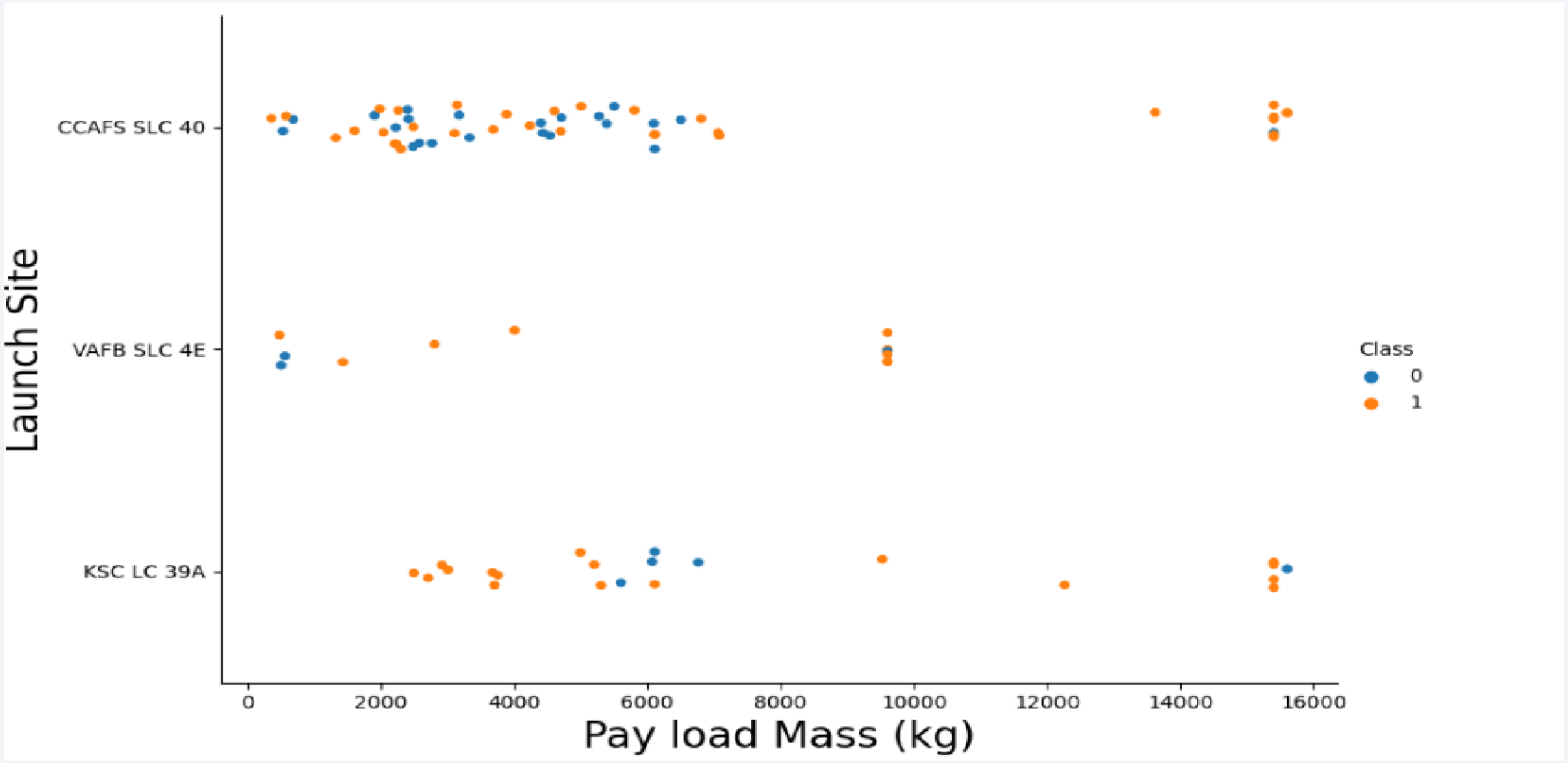
Flight Number vs. Launch Site

A scatter plot of Flight Number vs. Launch Site



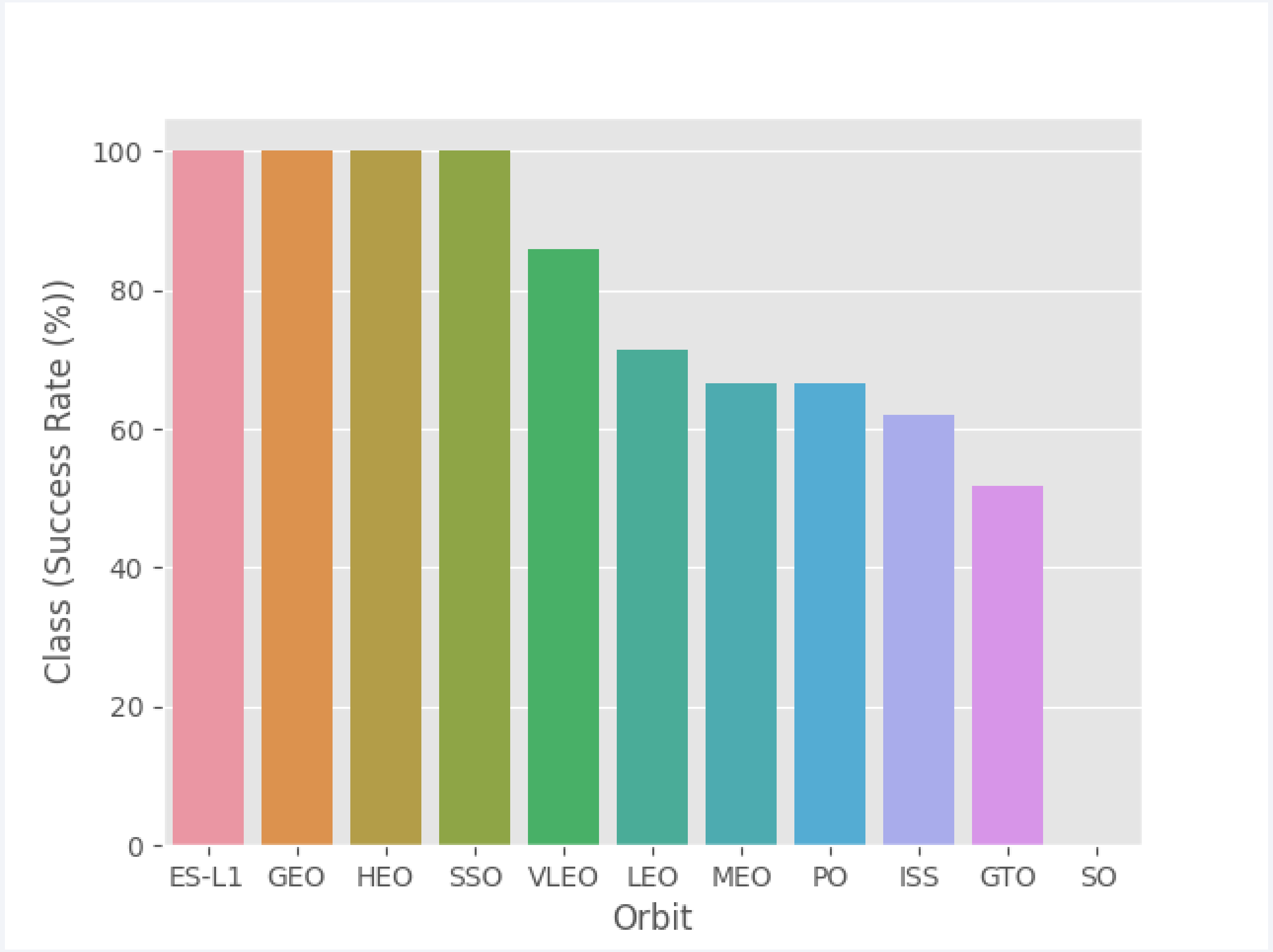
Payload vs. Launch Site

A scatter plot of Payload vs. Launch Site



Success Rate vs. Orbit Type

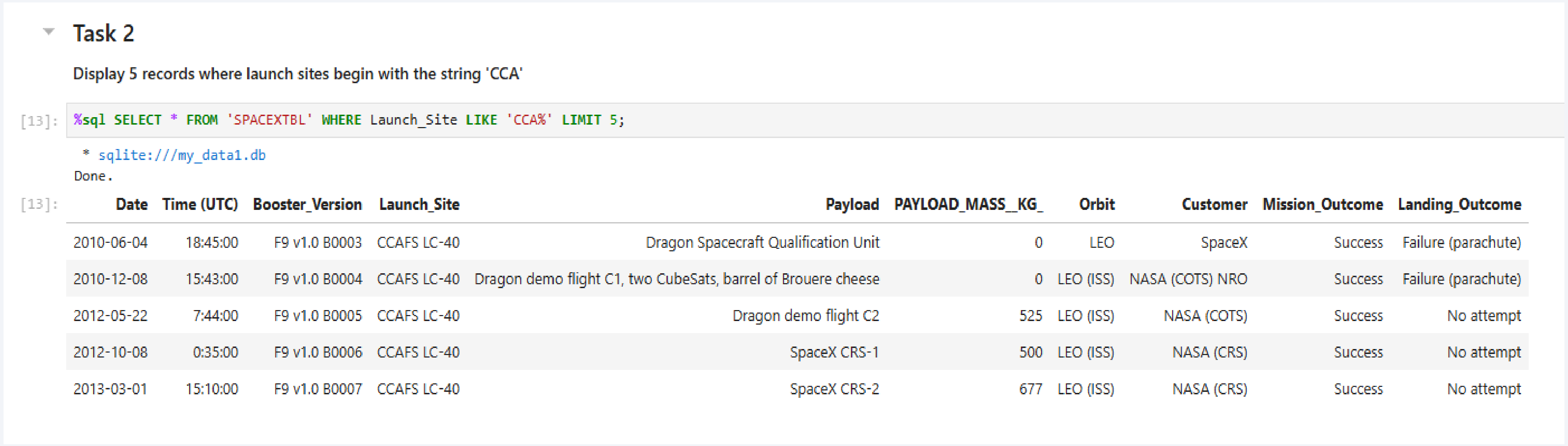
- bar chart for the success rate of each orbit type



All Launch Site Names

- **All Launch Site Names**
- Given the data, these are the names of the launch sites where different rocket landings were attempted:
- **CCAFS LC-40**
- **CCAFS SLC-40**
- **KSC LC-39A**
- **VAFB SLC-4E**

Launch Site Names Begin with 'CCA'



Task 2

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

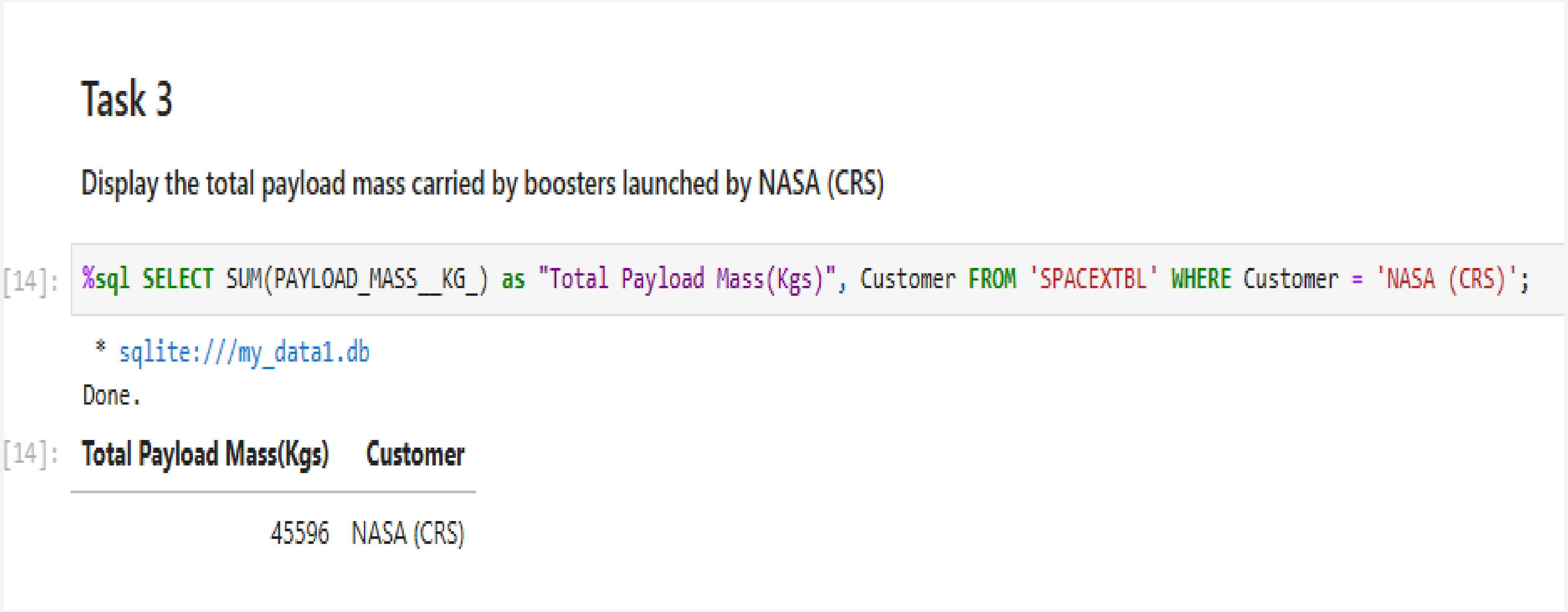
```
[13]: %sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

[13]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass



Task 3

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

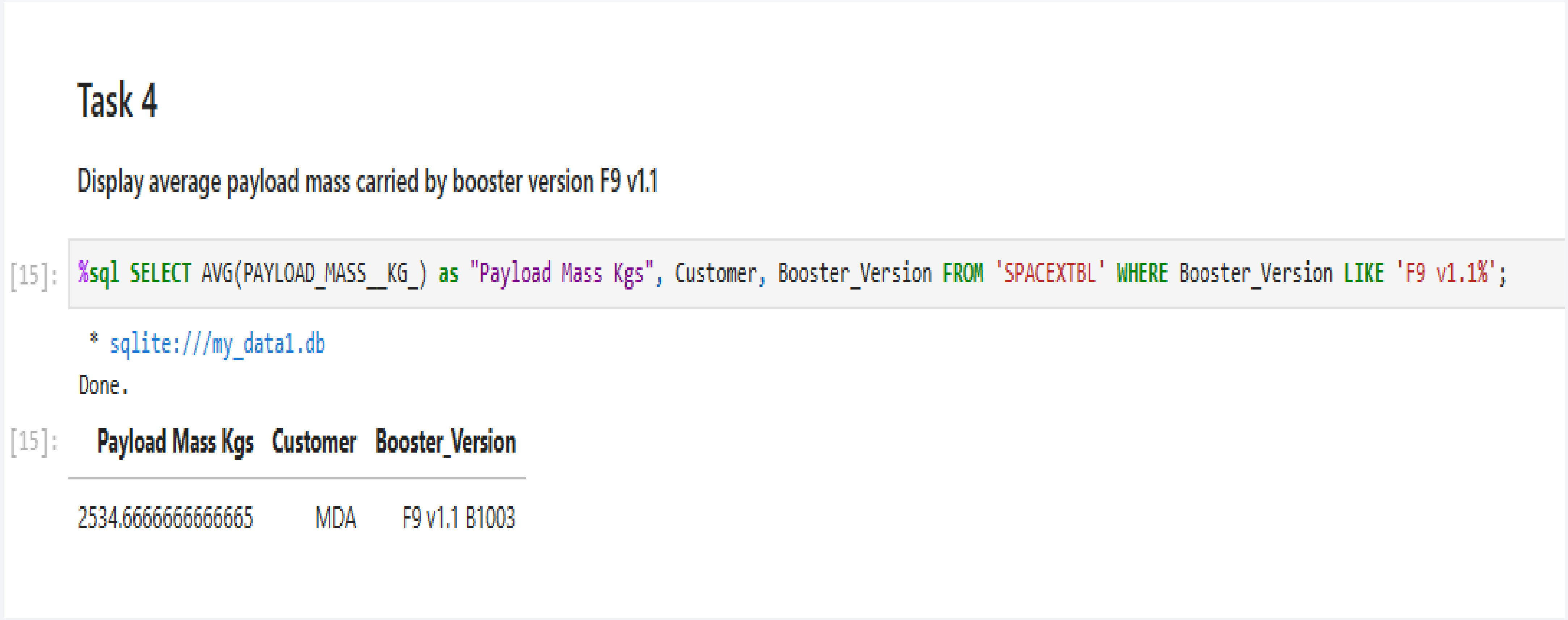
```
[14]: %sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

[14]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1



Task 4

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

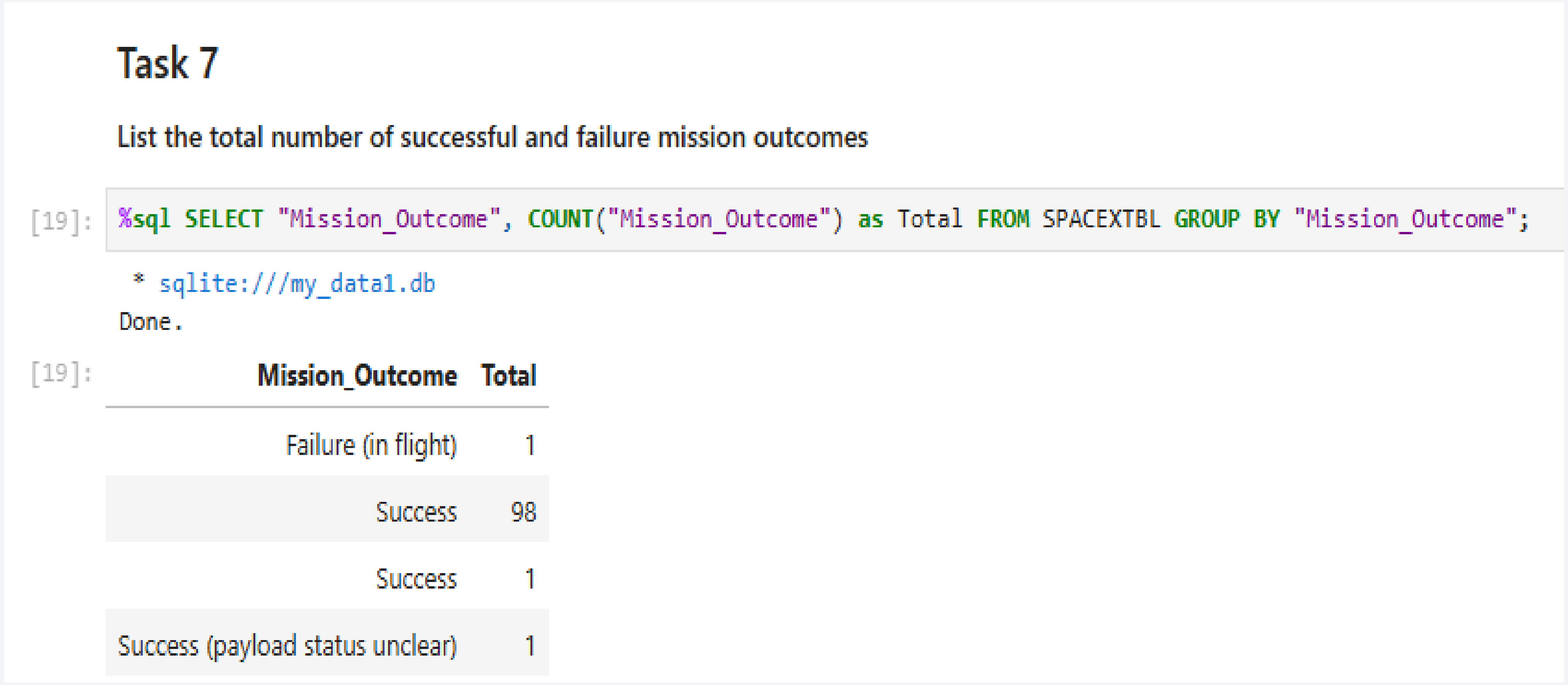
```
[15]: %sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

[15]:

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

Total Number of Successful and Failure Mission Outcomes



Task 7

List the total number of successful and failure mission outcomes

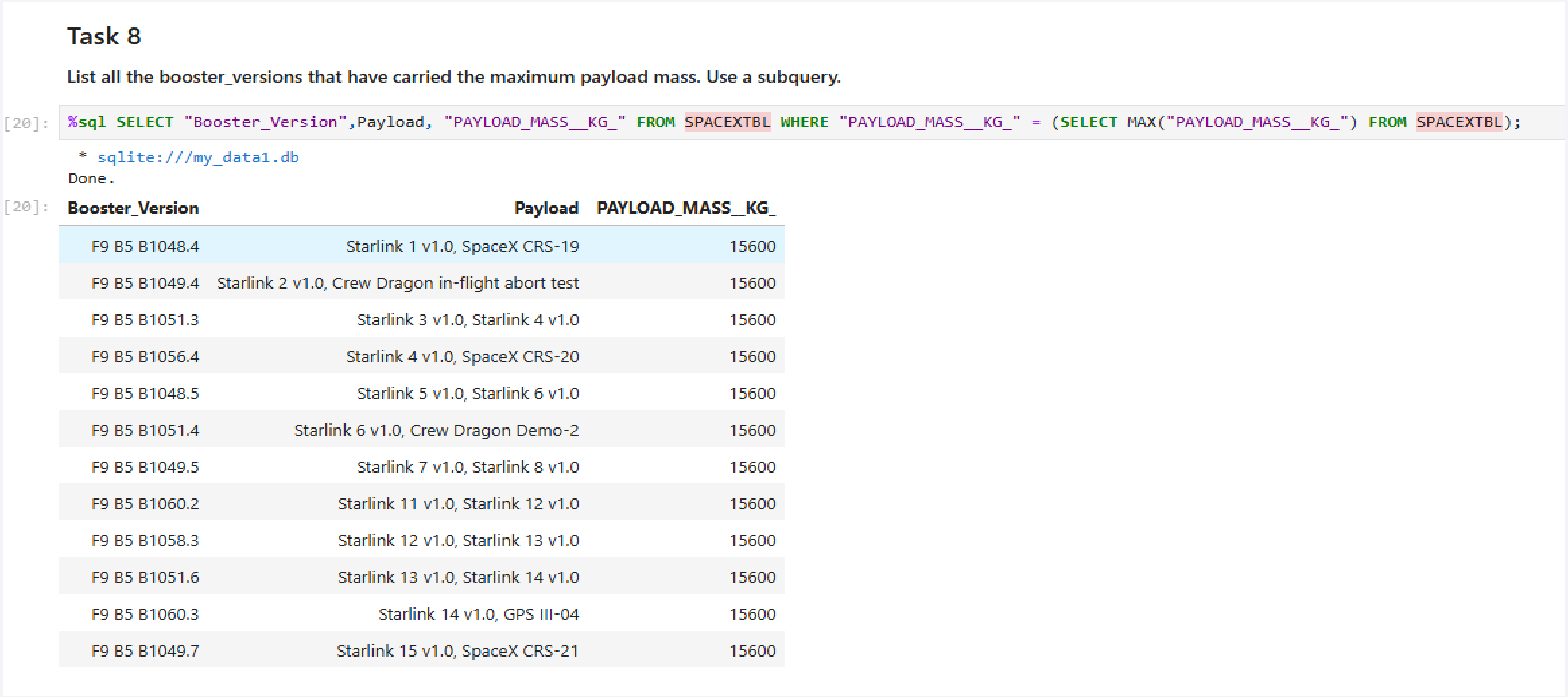
```
[19]: %sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

[19]:

Mission_Outcome	Total
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload



Task 8

List all the booster_versions that have carried the maximum payload mass. Use a subquery.

```
[20]: %sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);
```

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

[20]:

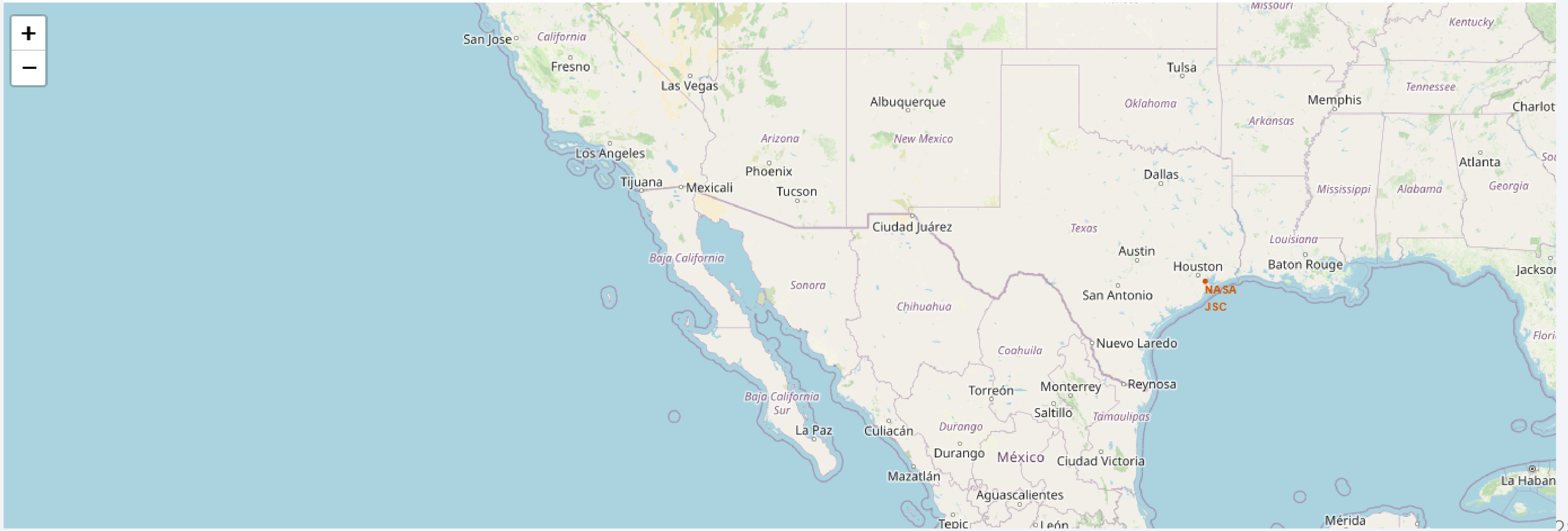
Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

Section 3

Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

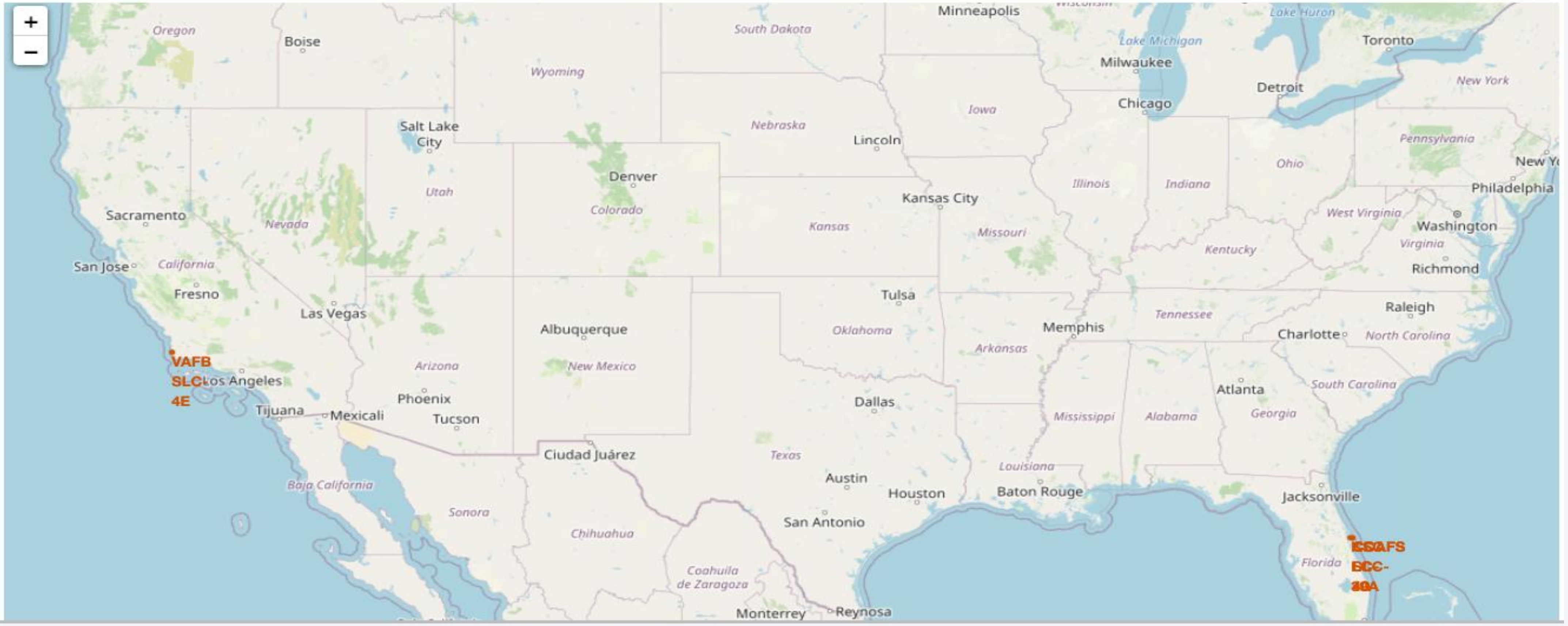
```
# Create a blue circle at NASA Johnson Space Center's coordinate with a popup label showing its name
circle = folium.Circle(nasa_coordinate, radius=1000, color='#d35400', fill=True).add_child(folium.Popup('NASA Johnson Space Center'))
# Create a blue circle at NASA Johnson Space Center's coordinate with a icon showing its name
marker = folium.map.Marker(
    nasa_coordinate,
    # Create an icon as a text label
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % 'NASA JSC',
        )
    )
site_map.add_child(circle)
site_map.add_child(marker)
```



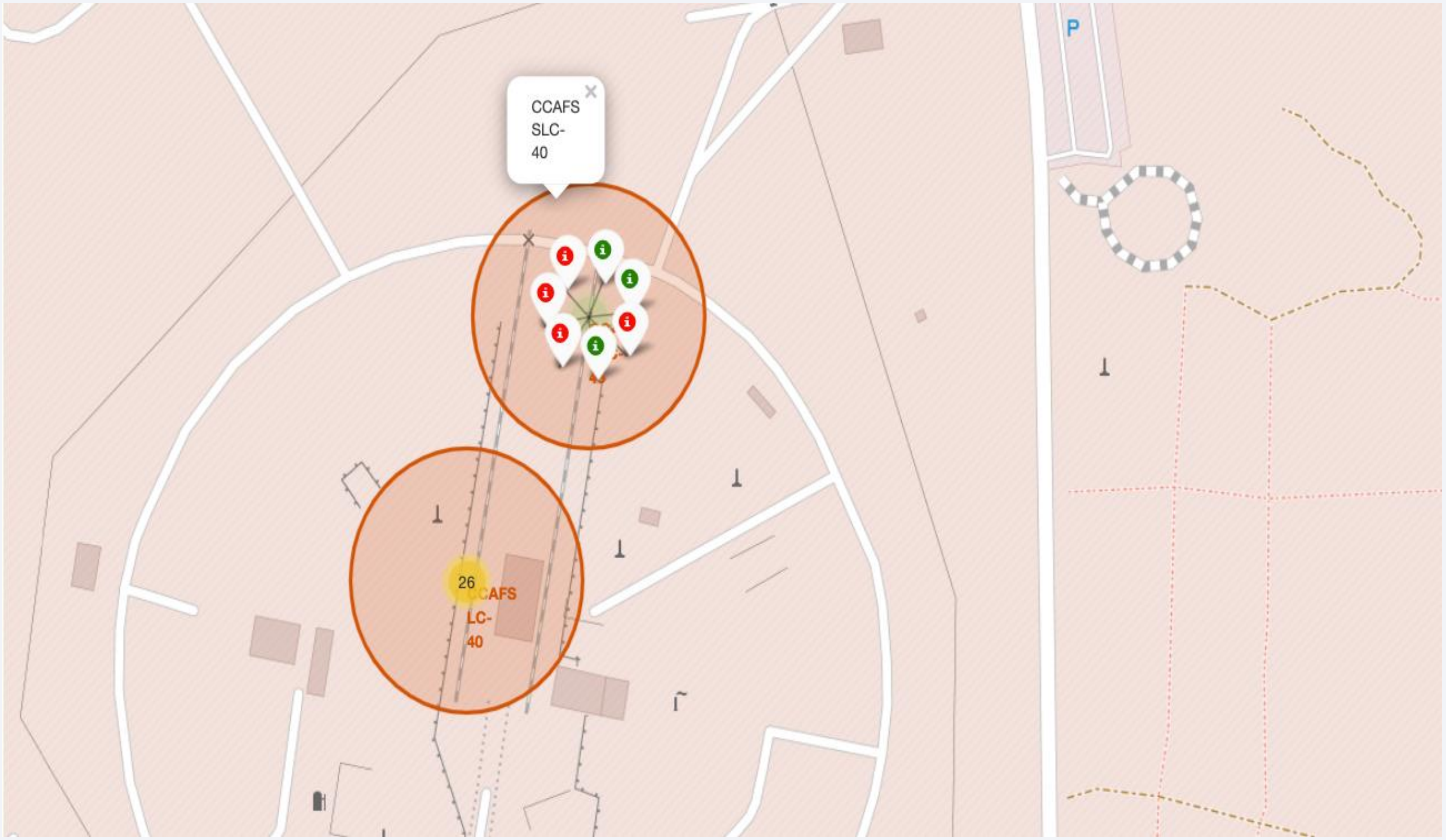
<Folium Map Screenshot 2>

```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch site name as a popup label
```

The generated map with marked launch sites should look similar to the following:



<Folium Map Screenshot 3>


CCAFS
SLC-
40
26
CCAFS
LC-
40
P

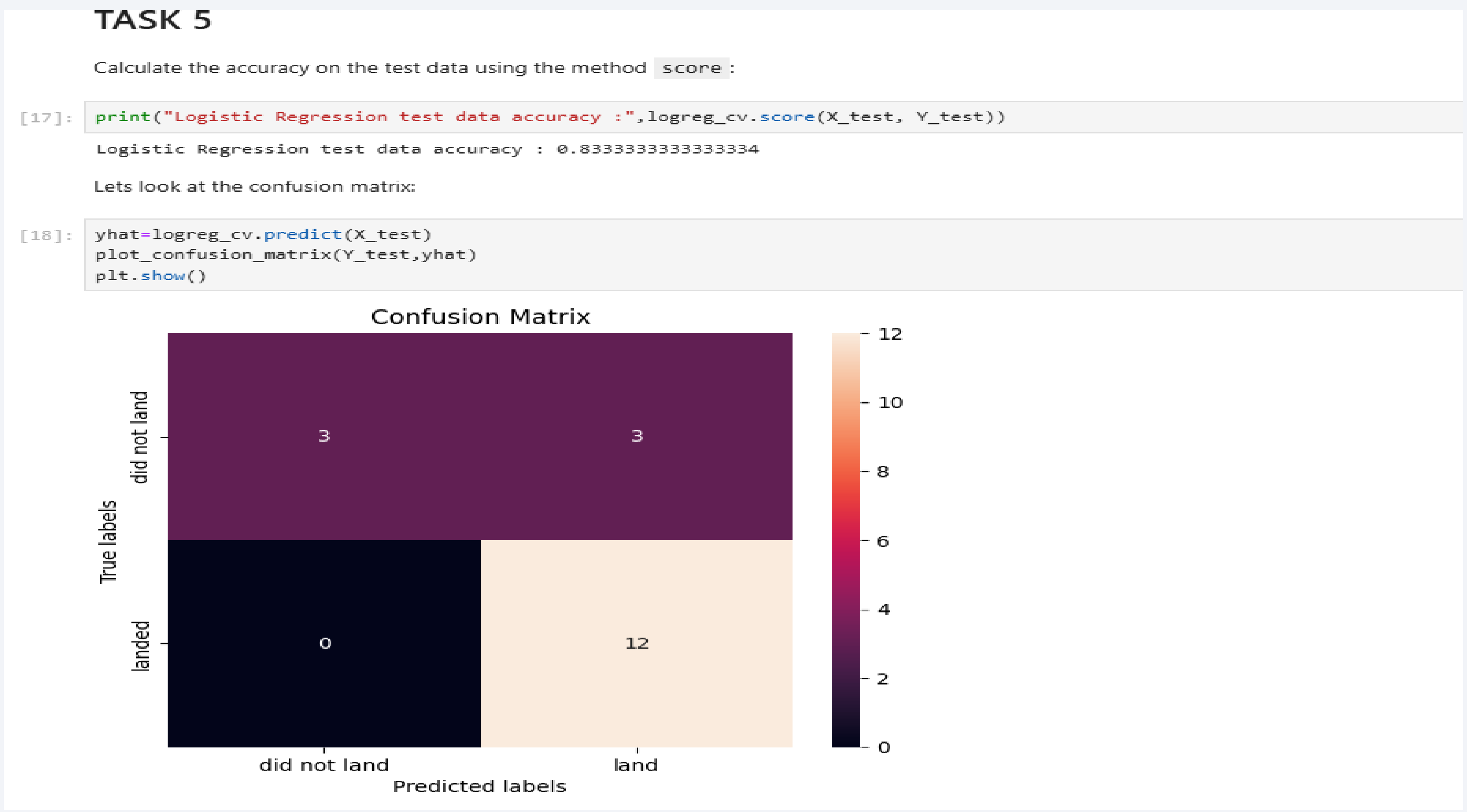
Section 4

Build a Dashboard with Plotly Dash

Section 5

Predictive Analysis (Classification)

Classification Accuracy



TASK 5

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

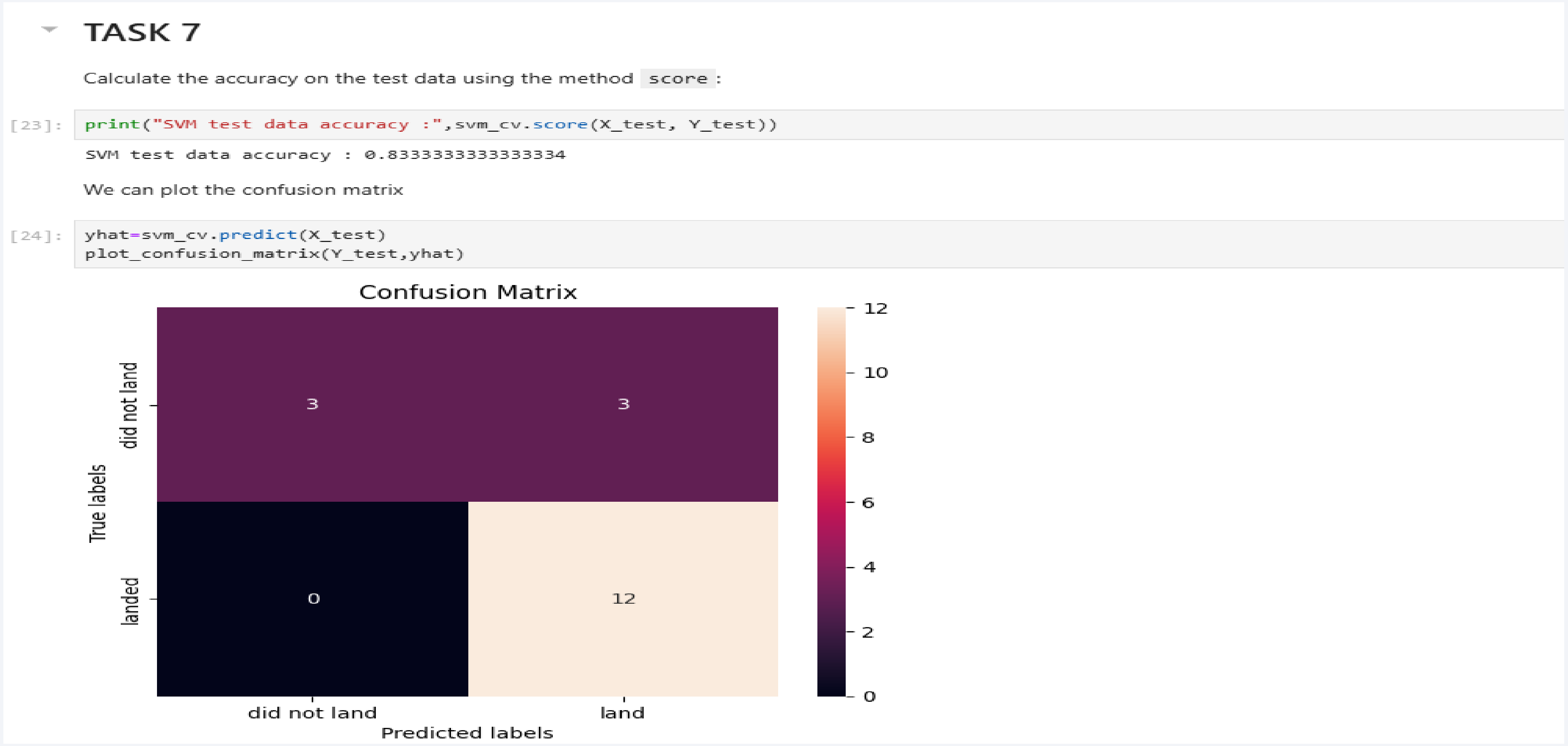
```
[17]: print("Logistic Regression test data accuracy :",logreg_cv.score(X_test, Y_test))
```

```
Logistic Regression test data accuracy : 0.8333333333333334
```

Lets look at the confusion matrix:

```
[18]: yhat=logreg_cv.predict(X_test)
      plot_confusion_matrix(Y_test,yhat)
      plt.show()
```

Confusion Matrix



TASK 7

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

```
[23]: print("SVM test data accuracy :",svm_cv.score(X_test, Y_test))
```

```
SVM test data accuracy : 0.8333333333333334
```

We can plot the confusion matrix

```
[24]: yhat=svm_cv.predict(X_test)
      plot_confusion_matrix(Y_test,yhat)
```

Conclusions

* The more flights we have from the CCAFS SLC 40 launch site, the more successful they are. And the same goes for the payload they carry, that also gets better!

* We have a 100% landing success rate for Orbit types ES-L1, SSO, HEO, and GEO! That's awesome!

* The Falcon 9 has been landing better and better since 2010 to 2020. It's really picking up steam.

* And get this, our model is super accurate! With 88.89% precision, we know if it's going to land successfully or not. It's almost like having a crystal ball!

Appendix

Everything you see here, like the Python code bits, SQL queries, charts, what came out of the Notebooks, and the data, you can find it all on my GitHub. So if you want to check it out or use it, you know where to look!

Thank you!