IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

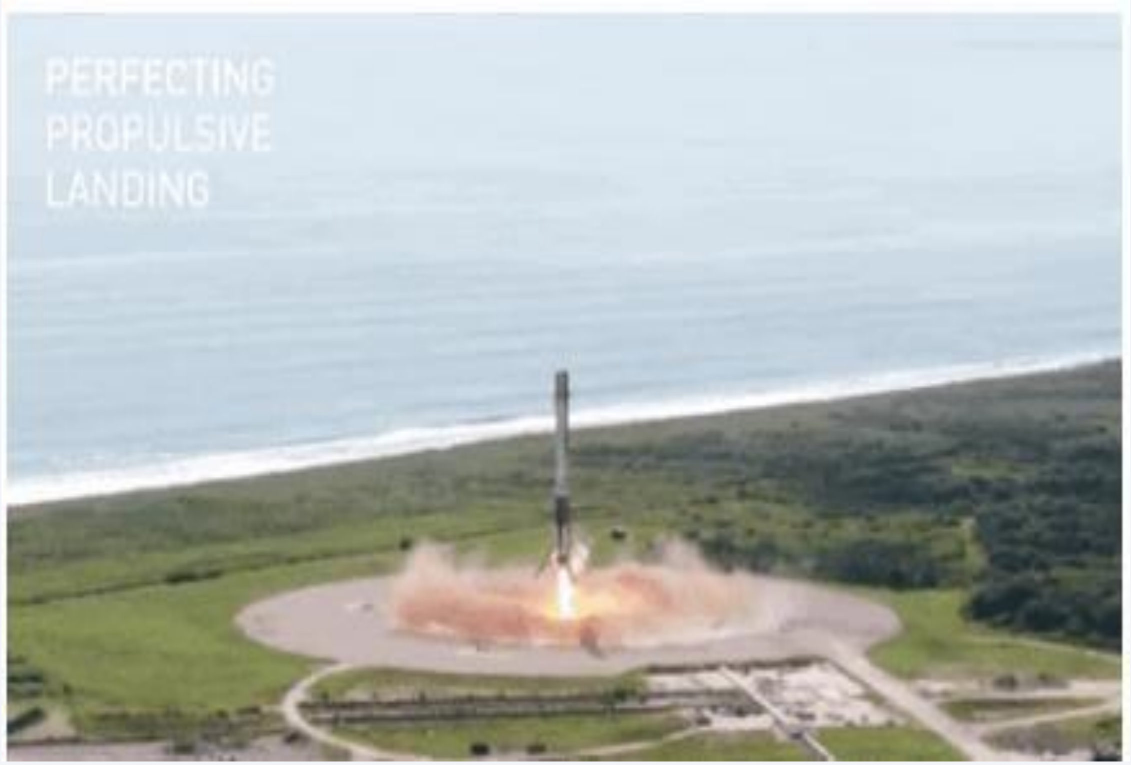
PRASHANTH B
9 November 2025

Outline

Executive Summary

- Summary of methodologies
  - SpaceX Data Collection using SpaceX API
  - SpaceX Data Collection with Web Scraping
  - SpaceX Data Wrangling
  - SpaceX Exploratory Data Analysis using SQL
  - Space-X EDA DataViz Using Python Pandas and Matplotlib
  - Space-X Launch Sites Analysis with Folium-Interactive Visual Analytics and Ploty Dash
  - SpaceX Machine Learning Landing Prediction
- Summary of all results
  - EDA results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis(Classification)

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 if an alternate company wants to bid against SpaceX for a rocket launch.

- Problems you want to find answers

  In this capstone, we will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describes how data sets were collected
- Perform data wrangling
  - Describes how data were 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

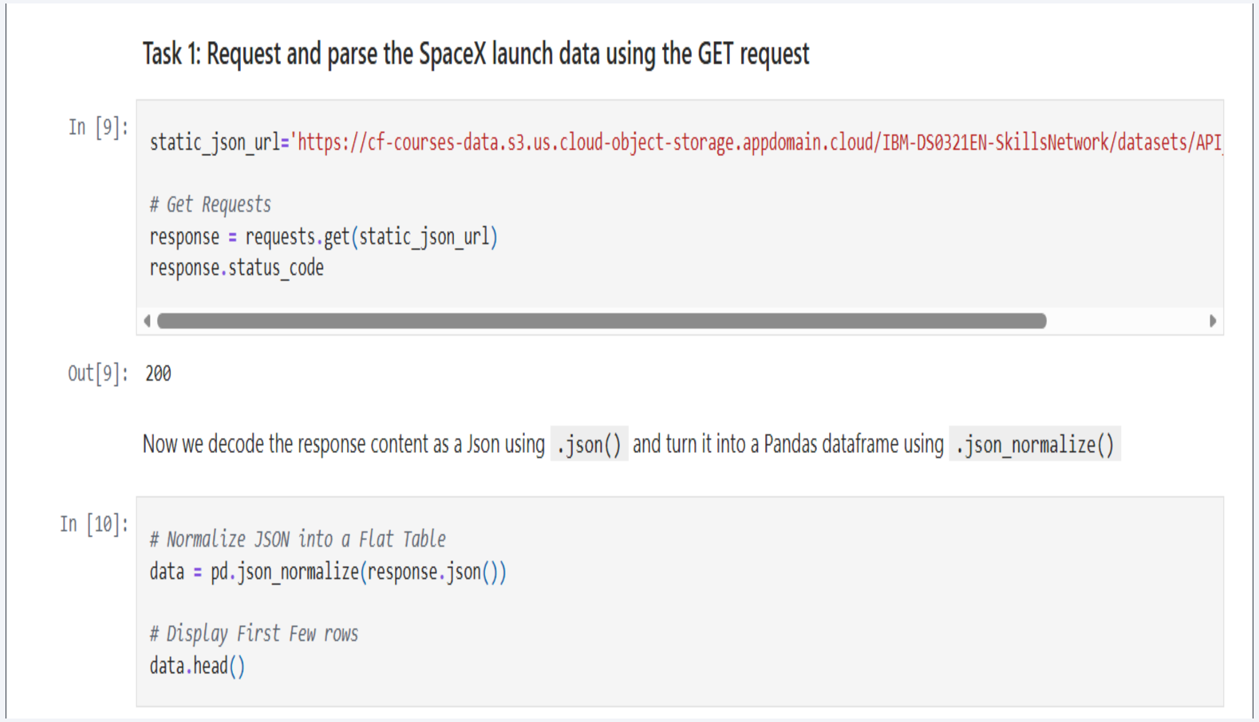
Data Collection

- Description of how SpaceX Falcon9 data was collected.
  - Data was first 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.
  - Finally to make the requested JSON results more consistent, the SpaceX launch data was requested and parsed using the GET request and then decoded the response content as a Json result which was then converted into a Pandas data frame.
  - Also 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, I extract the Falcon 9 launch HTML table records from the Wikipedia page, Parsed the table and converted it into a Pandas data frame

Data Collection – SpaceX API

- 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

- Here is the GitHub URL of the completed SpaceX API calls notebook [Click here]


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

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_

# Get Requests
response = requests.get(static_json_url)
response.status_code
```

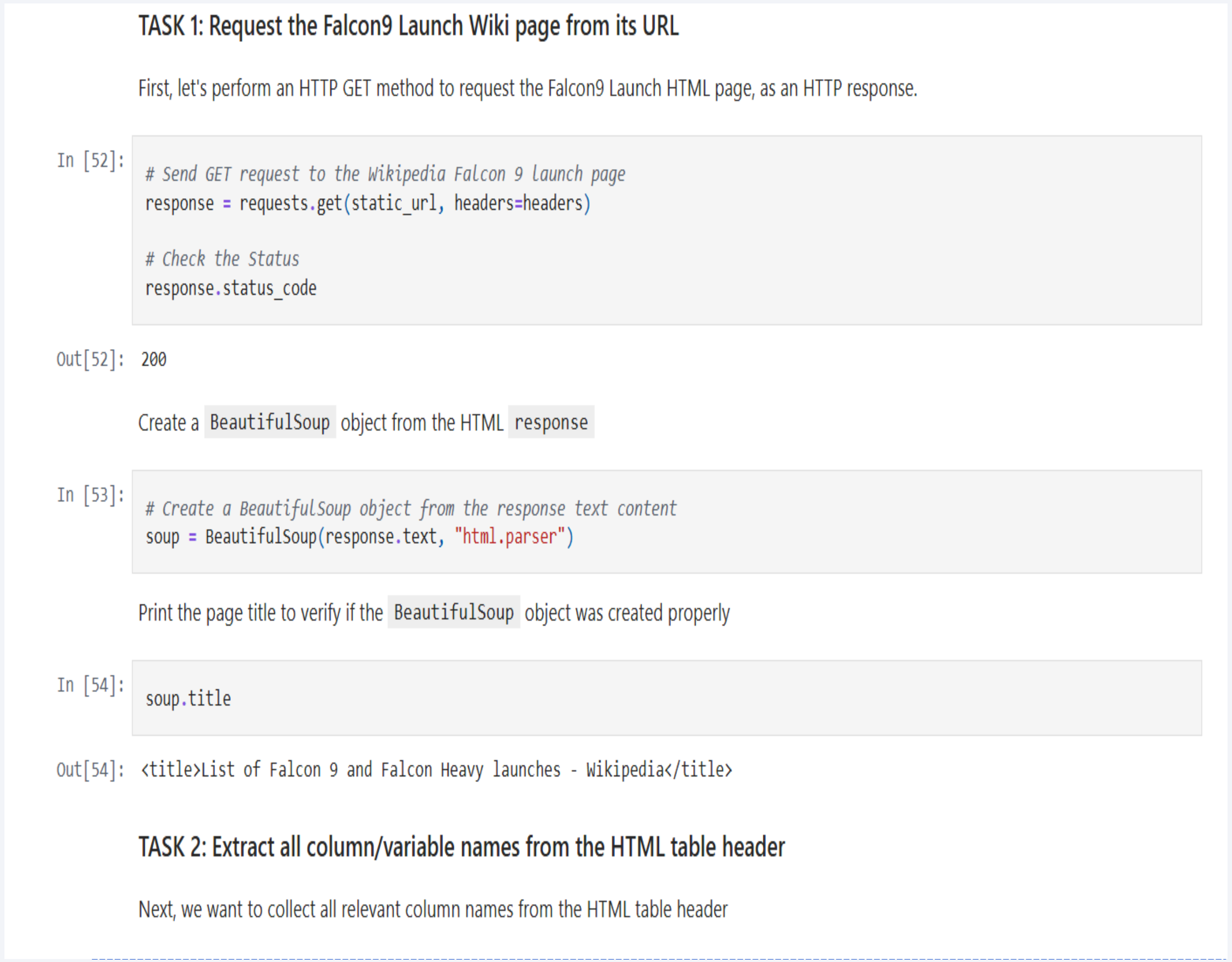
Out[9]:
```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

In [10]:
```
# Normalize JSON into a Flat Table
data = pd.json_normalize(response.json())

# Display First Few rows
data.head()
```

Data Collection - WebScraping

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



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.

```
In [52]: # Send GET request to the Wikipedia Falcon 9 launch page
         response = requests.get(static_url, headers=headers)

         # Check the Status
         response.status_code

Out[52]: 200
```

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

```
In [53]: # Create a BeautifulSoup object from the response text content
         soup = BeautifulSoup(response.text, "html.parser")
```

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

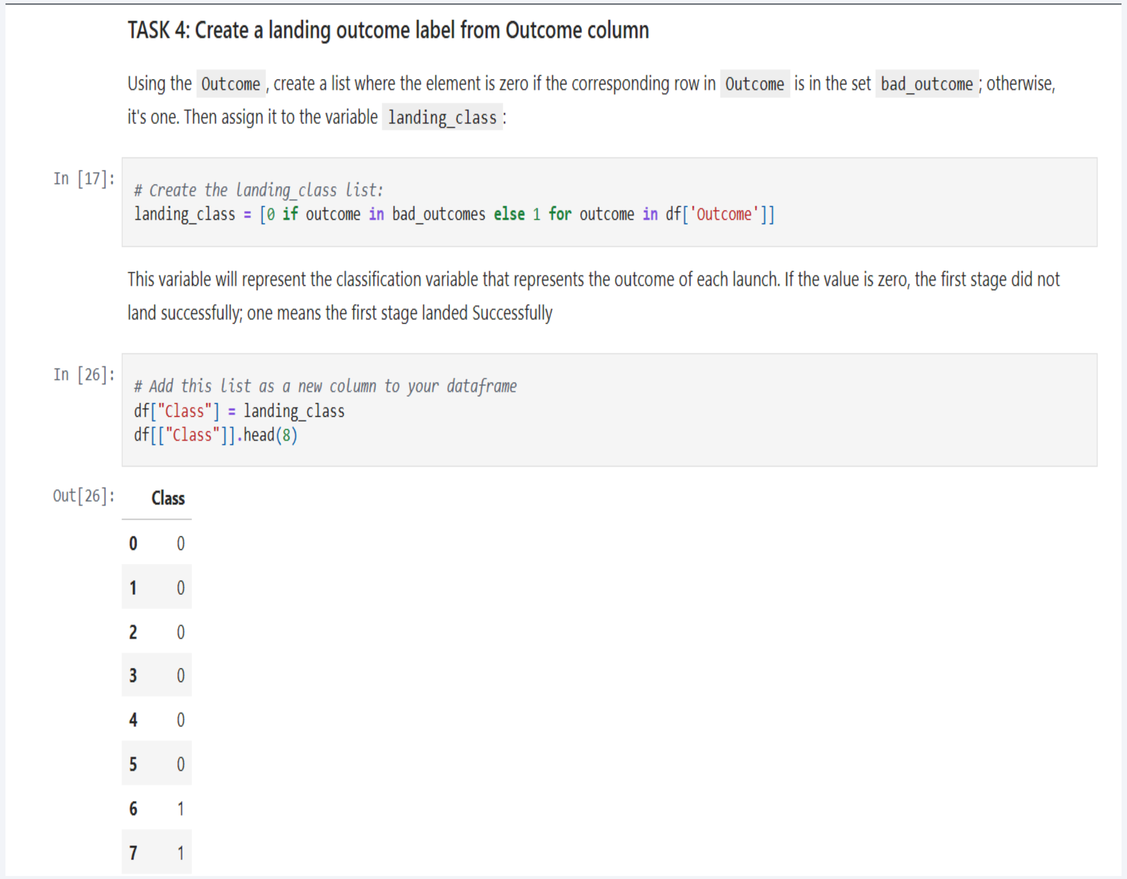
```
In [54]: soup.title

Out[54]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Data Wrangling

- After obtaining and creating a Pandas DF from the collected data, data was filtered using the ***BoosterVersion*** column to only keep the Falcon 9 launches, then dealt with the missing data values in the ***LandingPad*** and ***PayloadMass*** columns. For the ***PayloadMass*** , missing data values were replaced using mean value of column.
- Also performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models
- Here is the GitHub URL of the completed data wrangling related notebooks.



TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

```
In [17]:
# Create the landing_class list:
landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome']]
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

```
In [26]:
# Add this list as a new column to your dataframe
df["Class"] = landing_class
df[["Class"]].head(8)
```

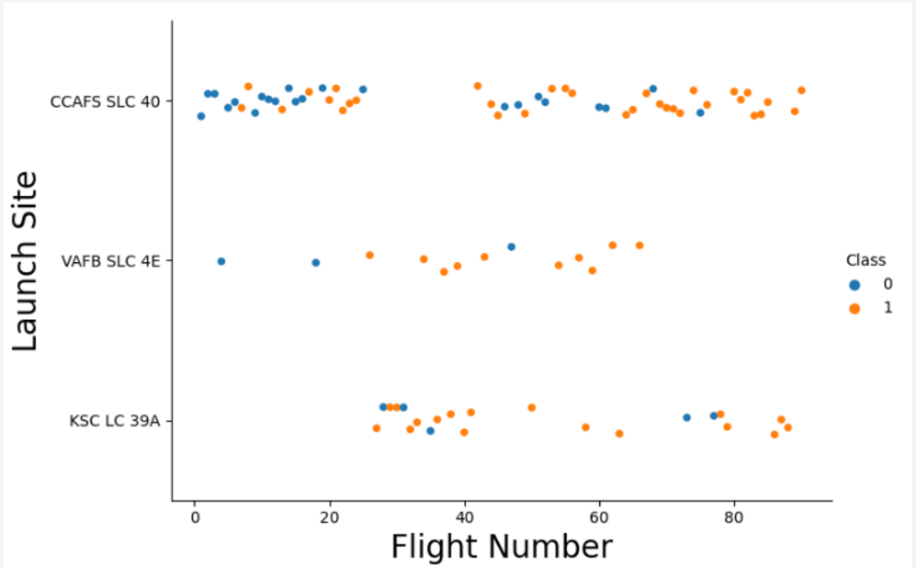
Out[26]:

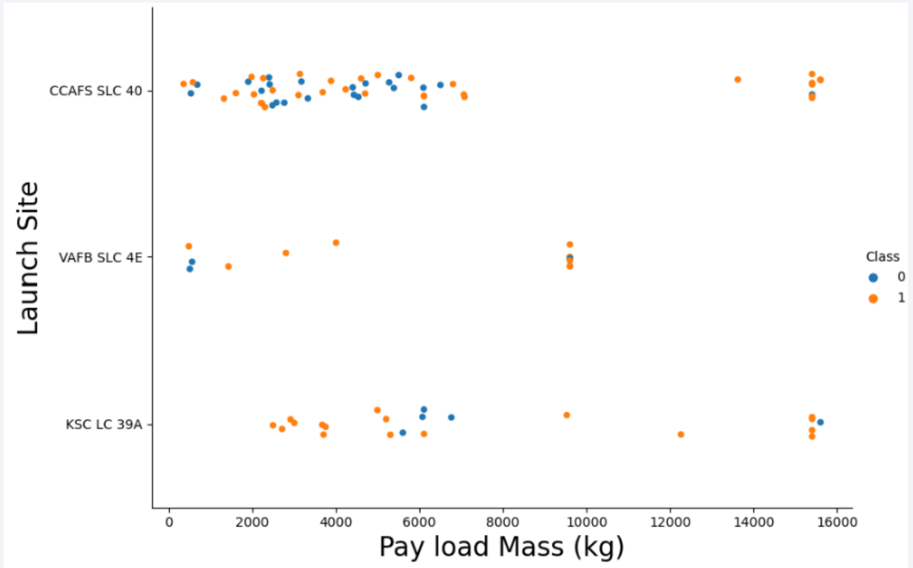
	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

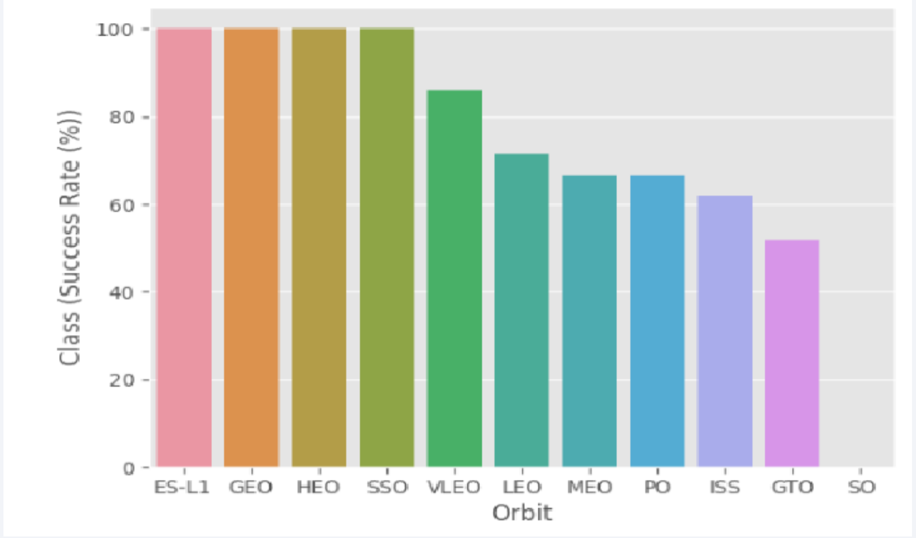
EDA with Data Visualization

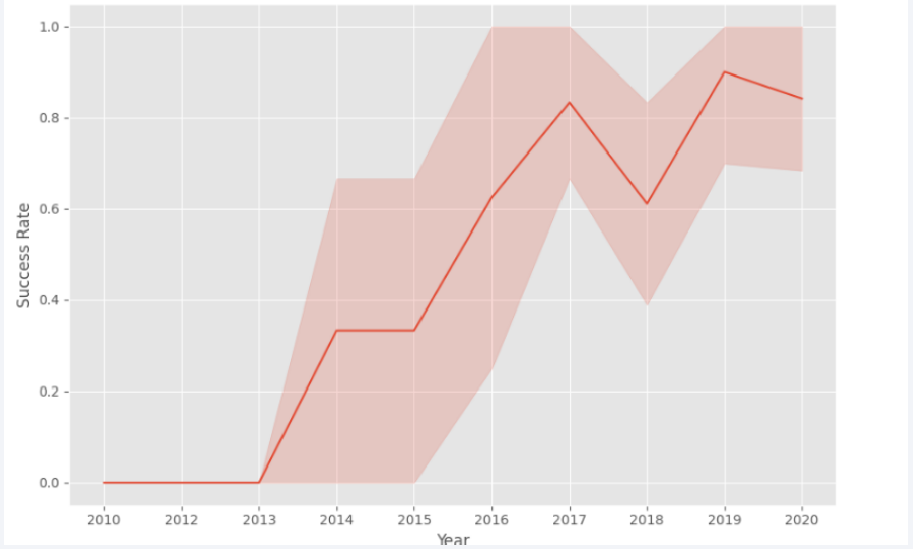
- Performed data Analysis and Feature Engineering using Pandas and Matplotlib.i.e.
  - Exploratory Data Analysis
  - Preparing Data Feature Engineering
- Used scatter plots to Visualize the relationship between Flight Number and Launch Site, Payload and Launch Site, FlightNumber and Orbit type, Payload and Orbit type.
- Used Bar chart to Visualize the relationship between success rate of each orbit type
- Line plot to Visualize the launch success yearly trend.
- Here is the GitHub URL of your completed EDA with data visualization notebook, (Falcon-9-First-Stage-Landing-Success-Prediction-for-Launch-Cost-Optimization/5. Space-X EDA DataViz Using Pandas and Matplotlib - SpaceX.ipynb at main · Prashanthbnaik/Falcon-9-First-Stage-Landing-Success-Prediction-for-Launch-Cost-Optimization)

EDA with Data Visualization (Plots Cont….)



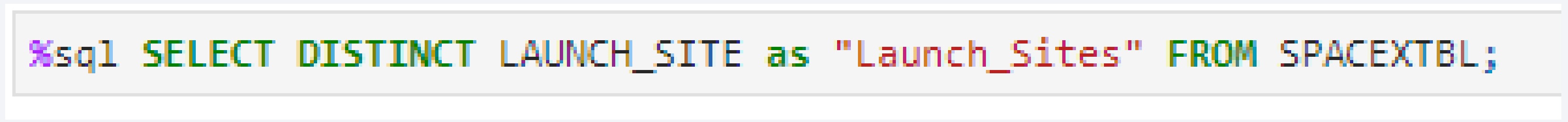






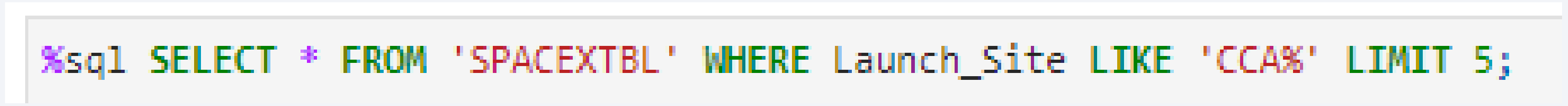
EDA with SQL

- The following SQL queries were performed for EDA
  - Display the names of the unique launch sites in the space mission



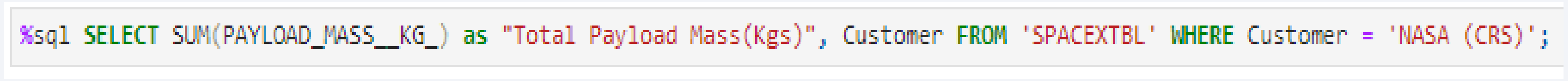
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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



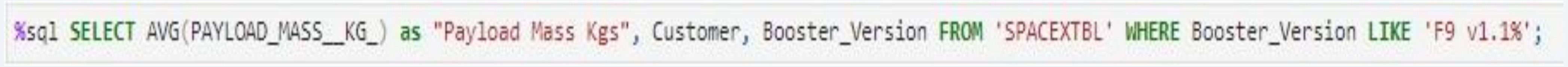
```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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



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

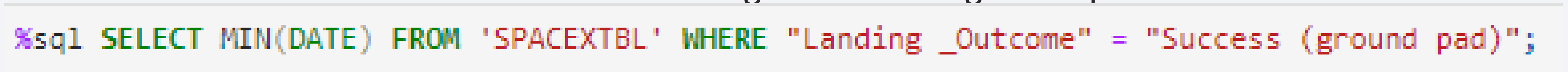
  - Display average payload mass carried by booster version F9 v1.1



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

EDA with SQL (Cont….)

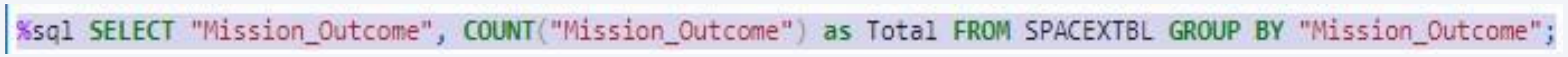
- List the date when the first successful landing outcome in ground pad was achieved



```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
```

- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000 *(%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;)*

- List the total number of successful and failure mission outcomes



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

- Here is the GitHub URL of your completed EDA with SQL notebook.

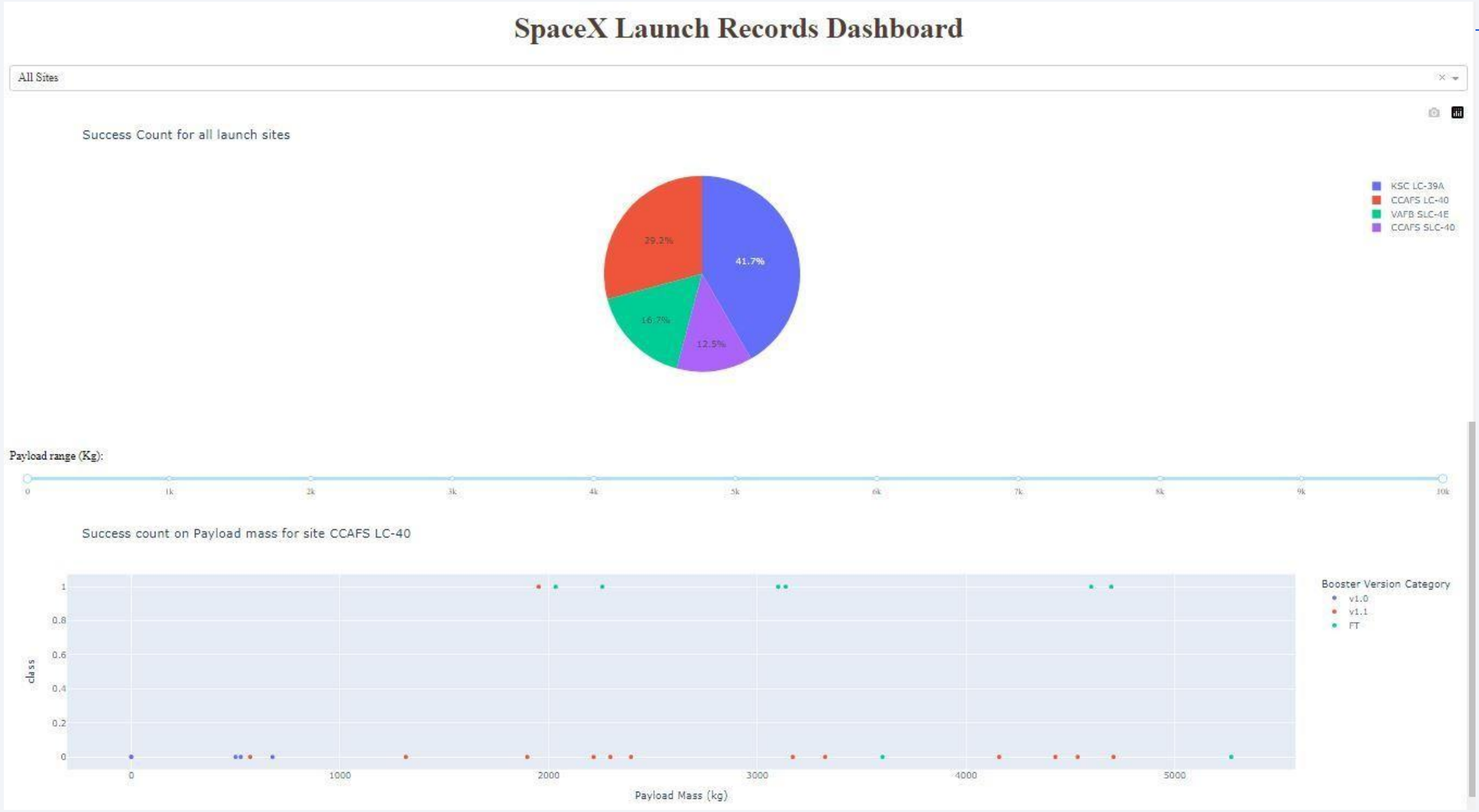
Build an Interactive Map with Folium

- Created folium map to marked all the launch sites, and created map objects such as markers, circles, lines to mark the success or failure of launches for each launch site.
- Created a launch set outcomes (failure=0 or success=1).
- Here is the GitHub URL of the completed interactive map with Folium map, as an external reference and peer-review purpose (Falcon-9-First-Stage-Landing-Success-Prediction-for-Launch-Cost-Optimization/6.Space-X Launch Sites Locations Analysis with Folium-Interactive Visual Analytics.ipynb at main · Prashanthbnaik/Falcon-9-First-Stage-Landing-Success-Prediction-for-Launch-Cost-Optimization)

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
  - Addeng a callback function to render the success-payload-scatter-chart scatter plot
- Here is the GitHub URL of your completed Plotly Dash lab (Falcon-9-First-Stage-Landing-Success-Prediction-for-Launch-Cost-Optimization/7. Build an Interactive Dashboard with Ploty Dash - spacex_dash_app.ipynb at main · Prashanthbnaik/Falcon-9-First-Stage-Landing-Success-Prediction-for-Launch-Cost-Optimization)

SpaceX Dash App


SpaceX Launch Records Dashboard

All Sites

Success Count for all launch sites

Payload range (Kg):

Success count on Payload mass for site CCAFS LC-40

Predictive Analysis (Classification)

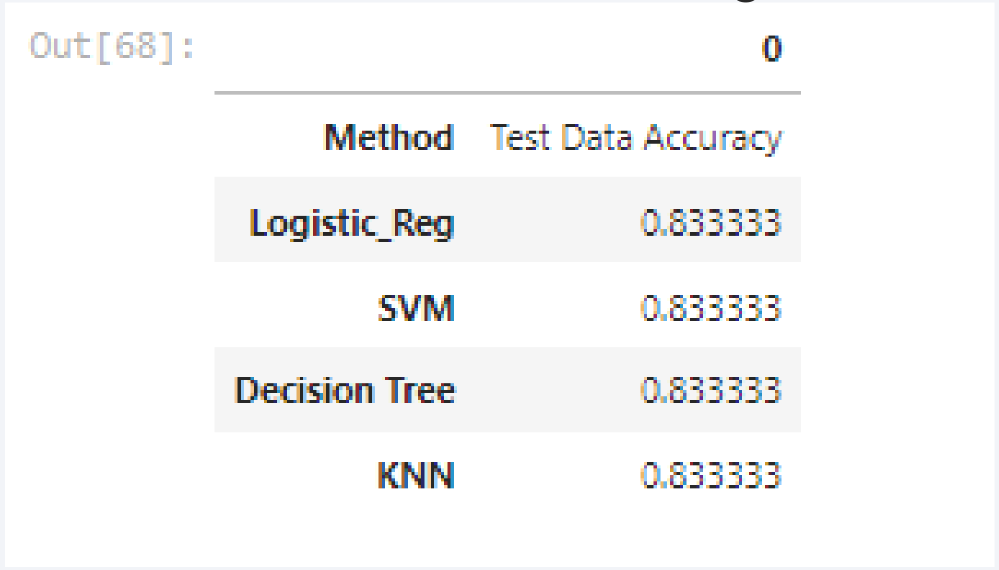
- Summary of how I built, evaluated, improved, and found the best performing classification model
- After loading the data as a Pandas Dataframe, I set out to perform exploratory Data Analysis and determine Training Labels by;
  - creating a NumPy array from the column Class in data, by applying the method to_numpy() then assigned it to the variable Y as the outcome variable.
  - Then standardized the feature dataset (x) by transforming it using preprocessing.StandardScaler() function from Sklearn.
  - After which the data was split into training and testing sets using the function train_test_split from sklearn.model_selection with the test_size parameter set to 0.2 and random_state to 2.

Predictive Analysis (Classification)

- In order to find the best ML model/ method that would performs best using the test data between SVM, Classification Trees, k nearest neighbors and Logistic Regression;
  - First created an object for each of the algorithms then created a GridSearchCV object and assigned them a set of parameters for each model.
  - For each of the models under evaluation, the GridsearchCV object was created with cv=10, then fit the training data into the GridSearch object for each to Find best Hyperparameter.
  - After fitting the training set, we output GridSearchCV object for each of the models, then displayed the best parameters using the data attribute best_params_ and the accuracy on the validation data using the data attribute best_score_.
  - Finally using the method score to calculate the accuracy on the test data for each model and plotted a confussion matrix for each using the test and predicted outcomes.

Predictive Analysis (Classification)

- The table below shows the test data accuracy score for each of the methods comparing them to show which performed best using the test data between SVM, Classification Trees, k nearest neighbors and Logistic Regression;



```
Out[68]:
```

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

- GitHub URL of the completed predictive analysis lab (Falcon-9-First-Stage-Landing-Success-Prediction-for-Launch-Cost-Optimization/8. SpaceX Machine Learning Prediction.ipynb at main · Prashanthbnaik/Falcon-9-First-Stage-Landing-Success-Prediction-for-Launch-Cost-Optimization)

Results

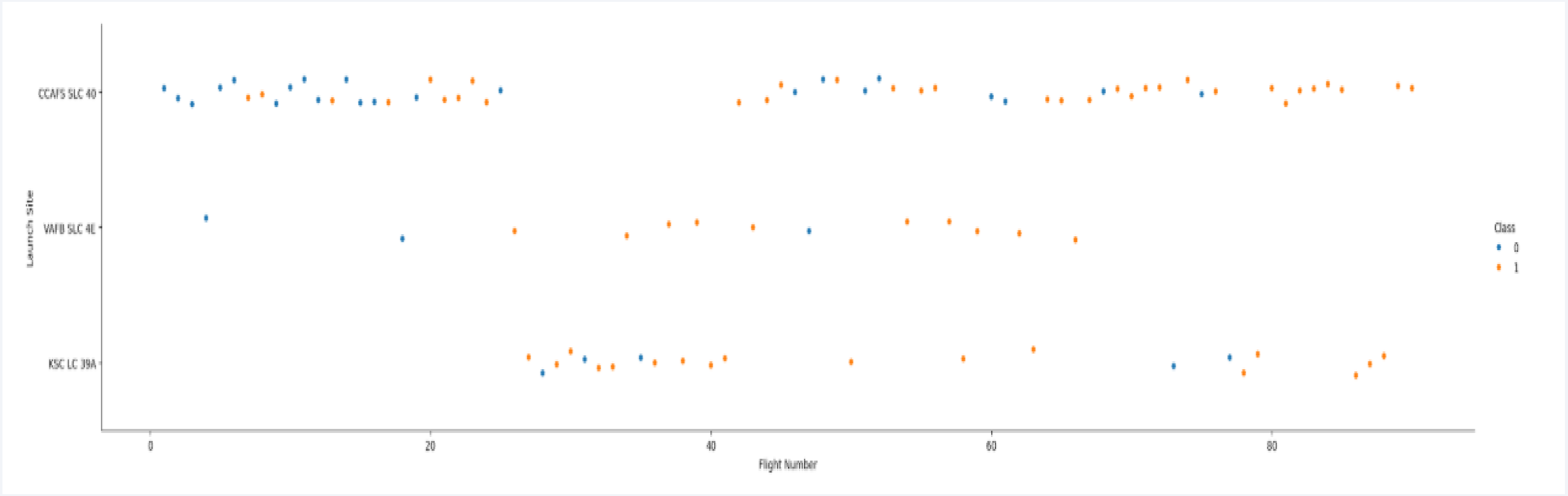
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

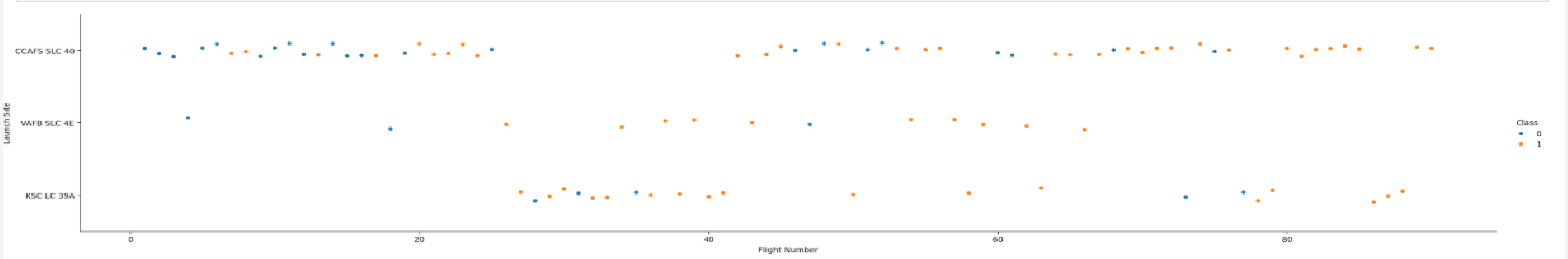
Flight Number vs. Launch Site

A scatter plot of Flight Number vs. Launch Site



Flight Number vs. Launch Site with explanations

A scatter plot with explanations Flight Number vs. Launch Site

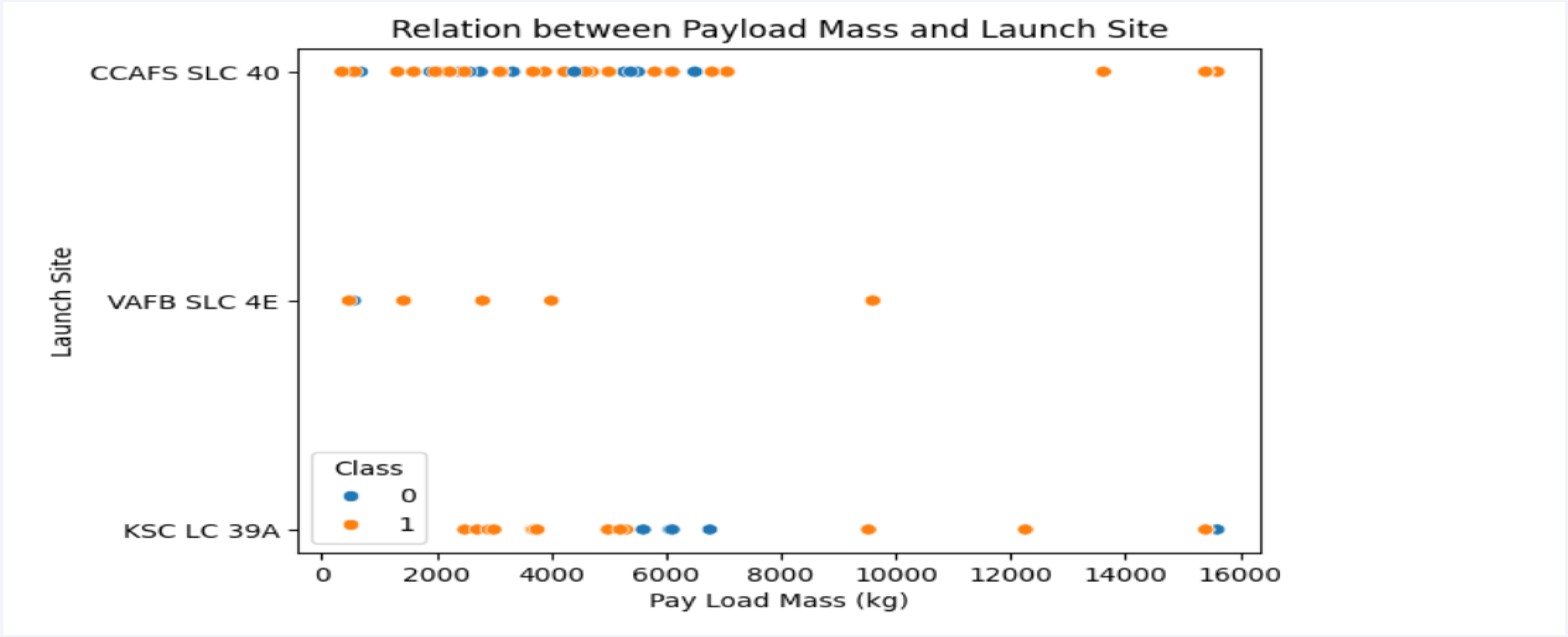


Insights and Patterns: Flight Number vs. Launch Site

- Certain launch sites show a **high concentration of flight numbers**, indicating frequent usage and operational reliability.
- **Class distribution varies by site**: some launch sites predominantly host successful missions, while others show a mix of outcomes, suggesting experimental or less optimized operations.
- **Sequential flight numbers** at specific sites hint at temporal trends—older sites hosting early missions, newer sites handling recent launches.
- **Sparse or isolated points** may represent special missions, anomalies, or temporary site inactivity.
- Sites with **diverse class outcomes** could be testing grounds or shared facilities for multiple mission types.

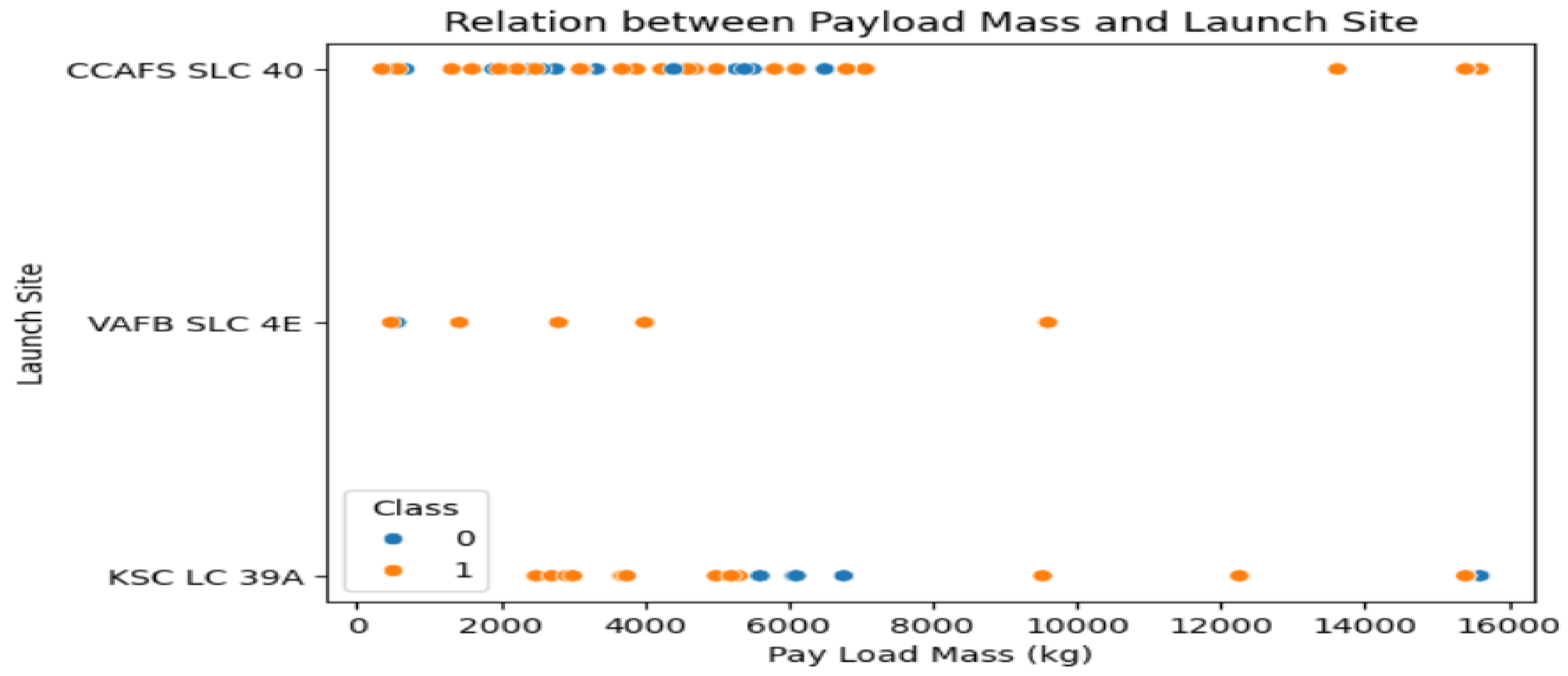
Payload vs. Launch Site

A scatter plot of Payload vs. Launch Site



Payload vs. Launch Site with explanations

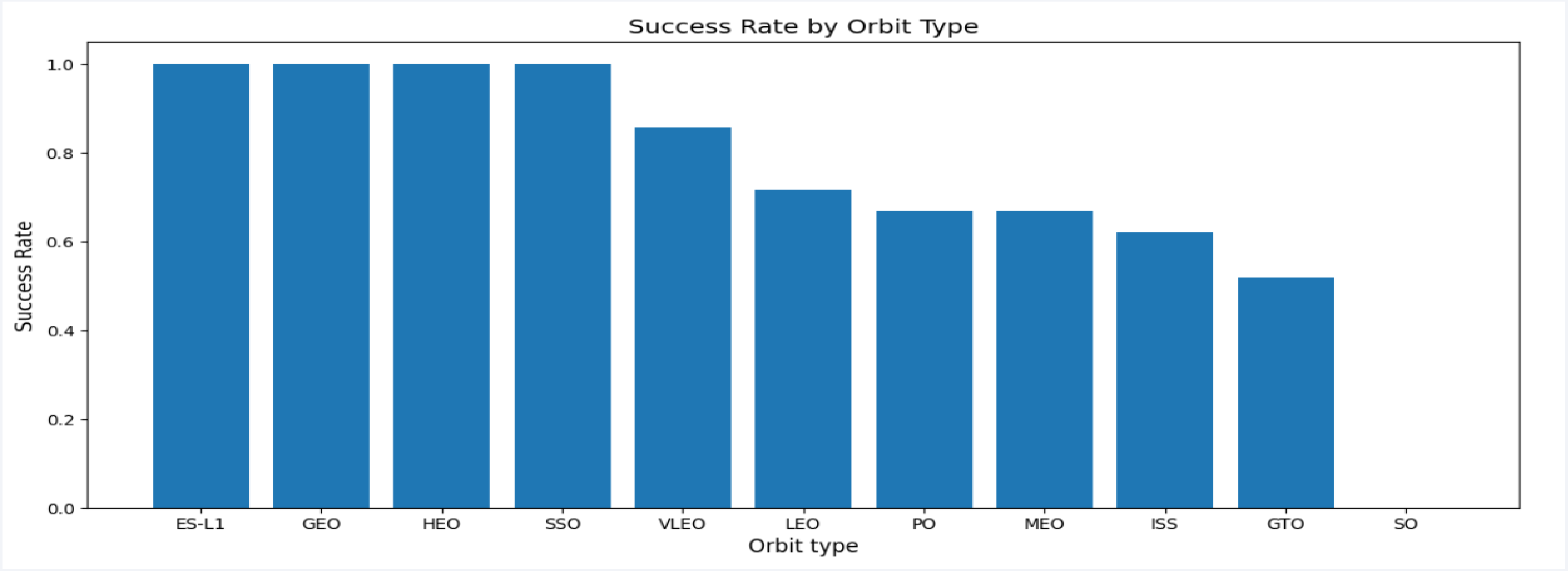
A scatter plot of Payload vs. Launch Site



Now if you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

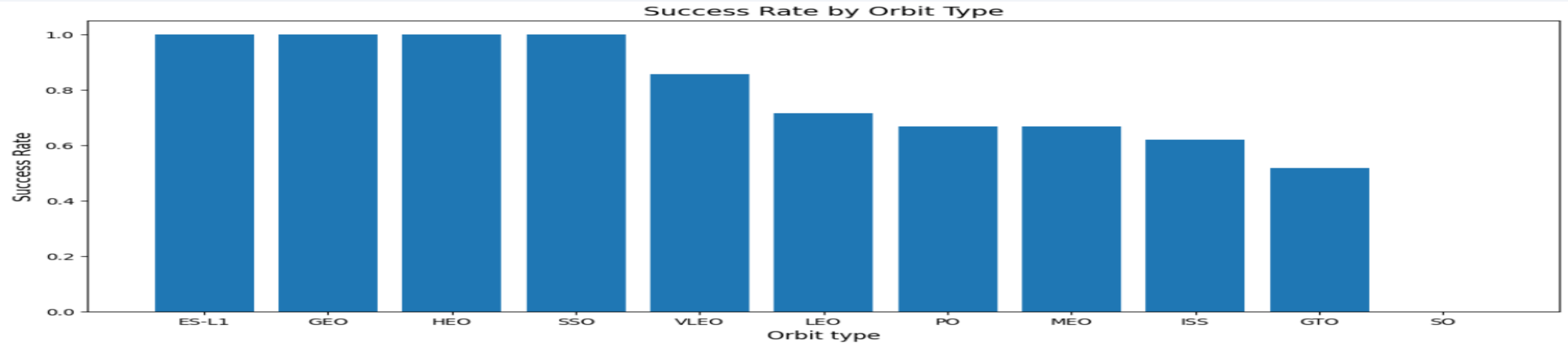
Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type



Success Rate vs. Orbit Type with explanations

- Show the screenshot of the bar chart with explanations

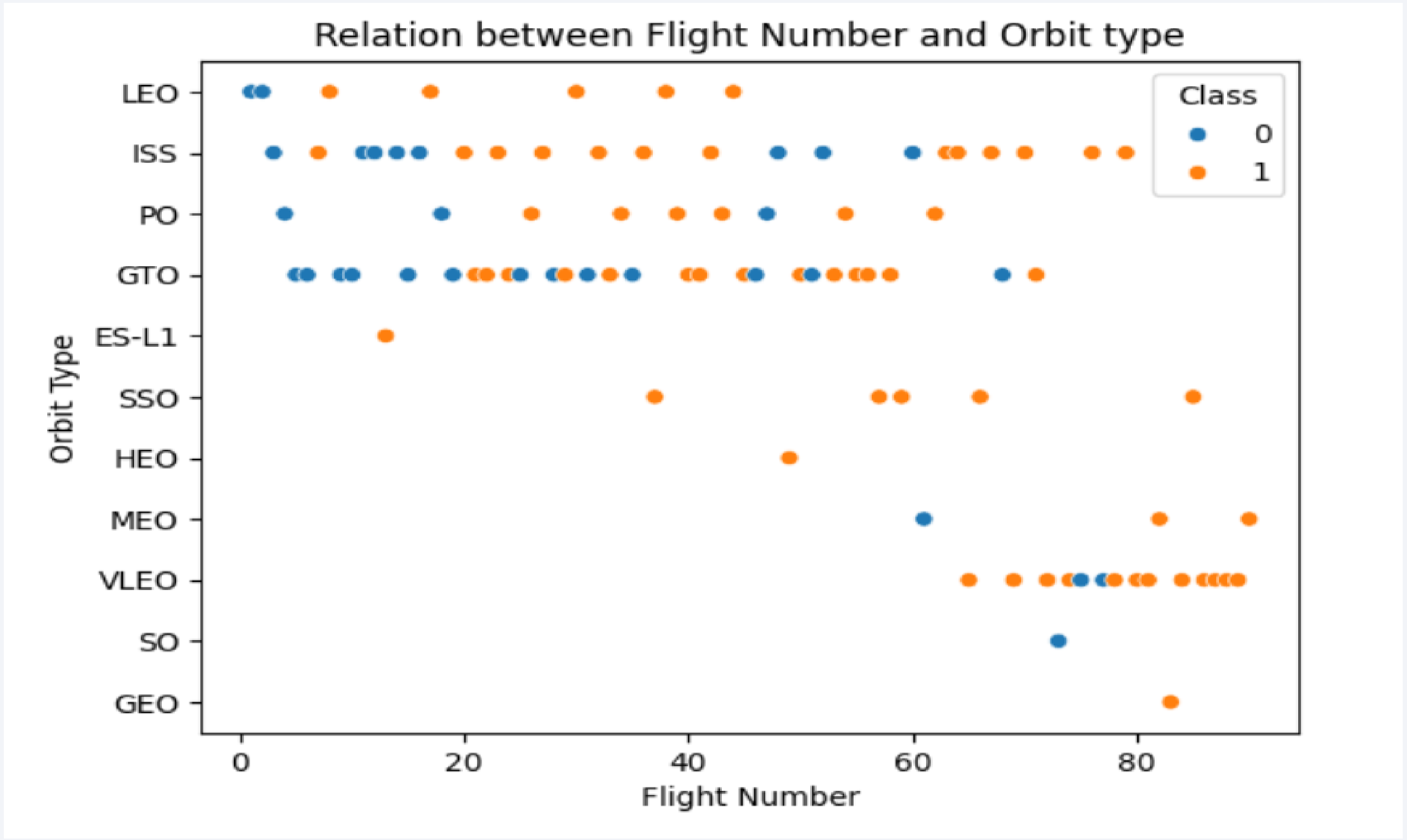


Insights and Patterns: Success Rate by Orbit Type

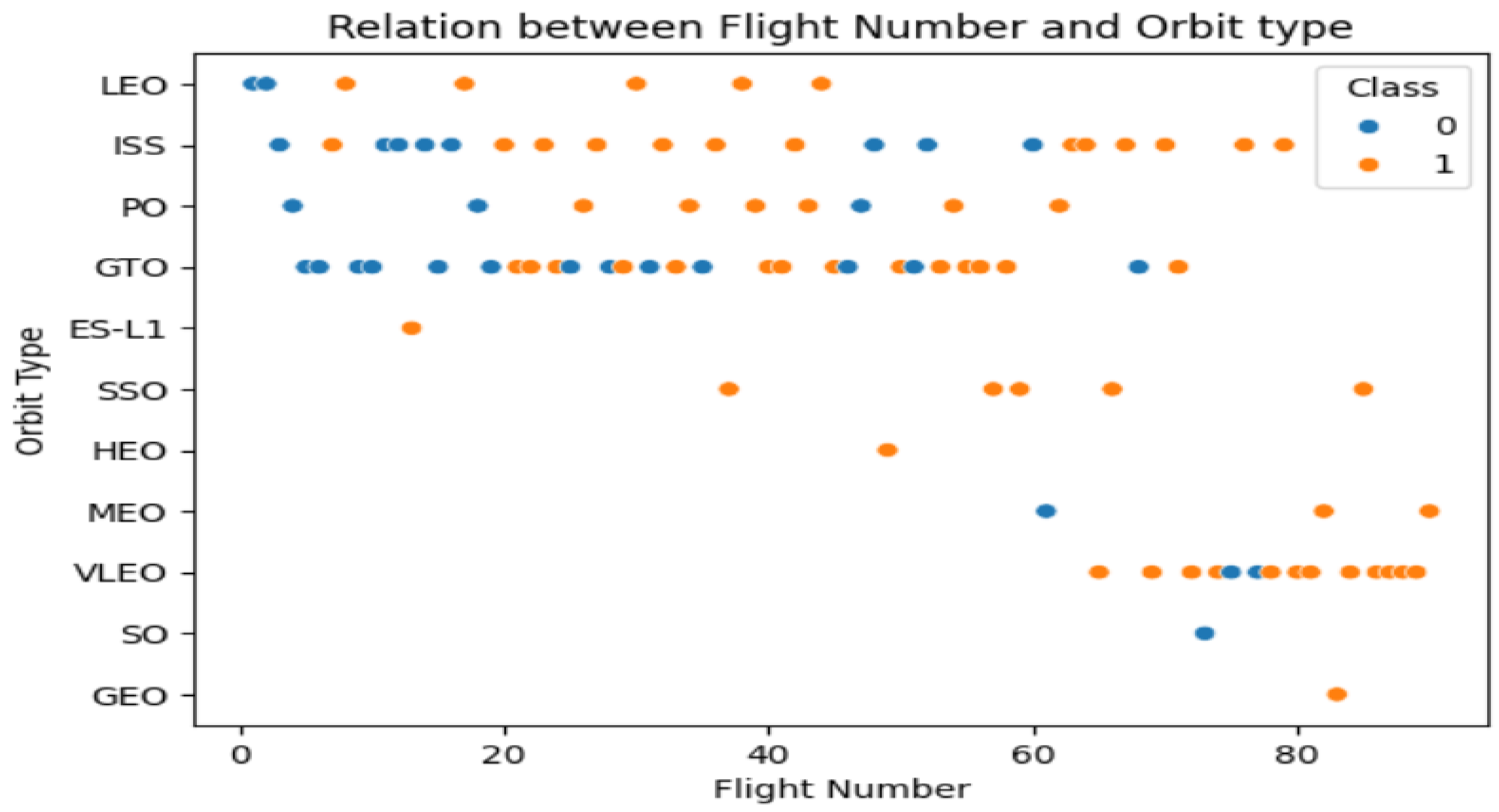
- **ES-L1, GEO, HEO, and SSO** exhibit the **highest success rates**, indicating strong reliability and mature mission infrastructure for these orbit types.
- **VLEO and LEO** show slightly lower success rates, possibly due to higher launch frequency or more experimental missions.
- **PO, MEO, and ISS** have moderate success rates, suggesting mixed mission outcomes or operational challenges.
- **GTO and SO** stand out with the **lowest success rates**, which may reflect technical complexity, higher risk profiles, or limited mission experience.
- Overall, the chart highlights a clear **performance gradient** across orbit types, useful for strategic planning and risk assessment in future missions.

Flight Number vs. Orbit Type

- A scatter point of Flight number vs. Orbit type



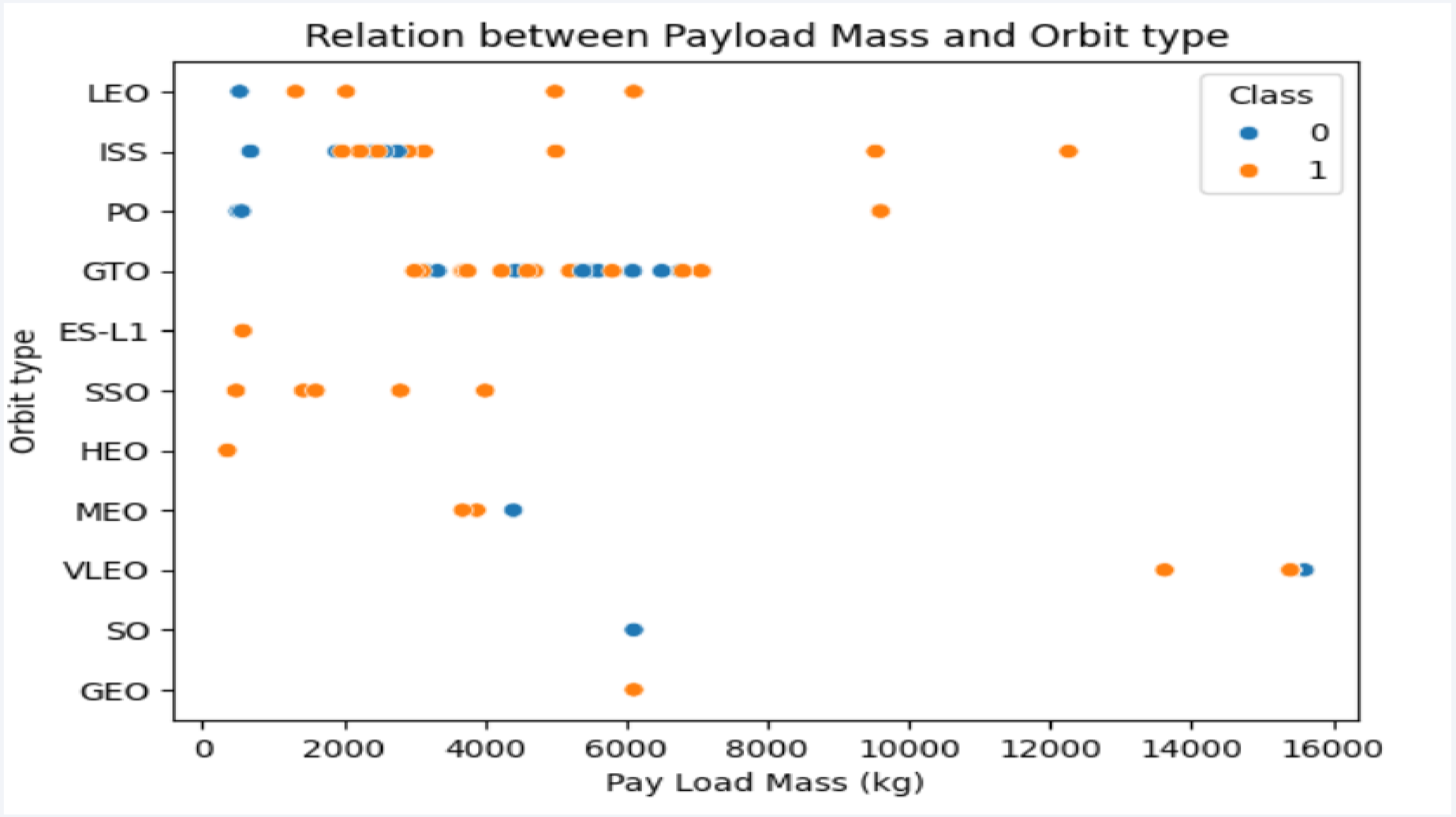
Flight Number vs. Orbit Typewith explanations



You can observe 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

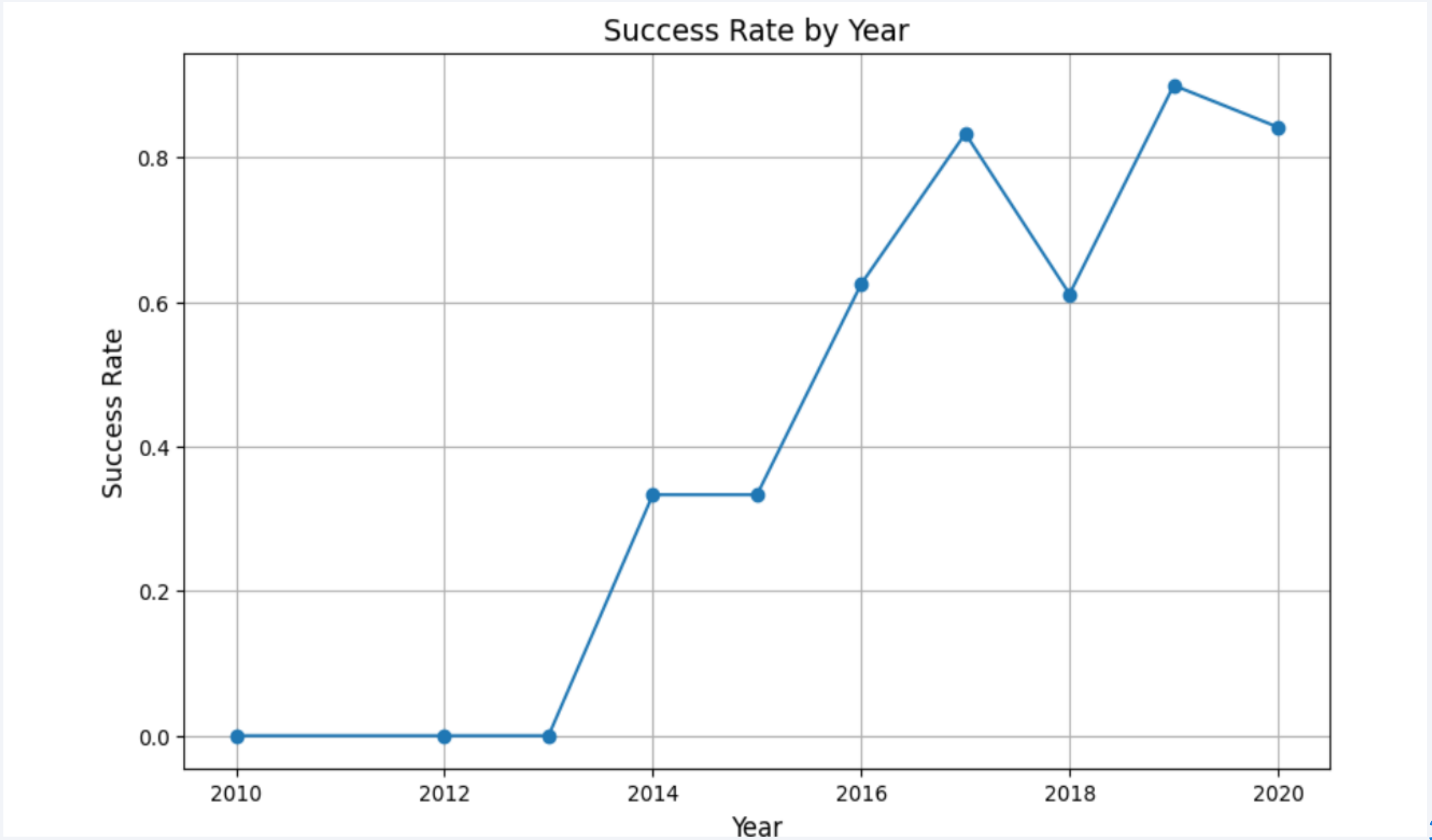
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) both have near equal chances.



Launch Success Yearly Trend

A line chart of yearly average success rate

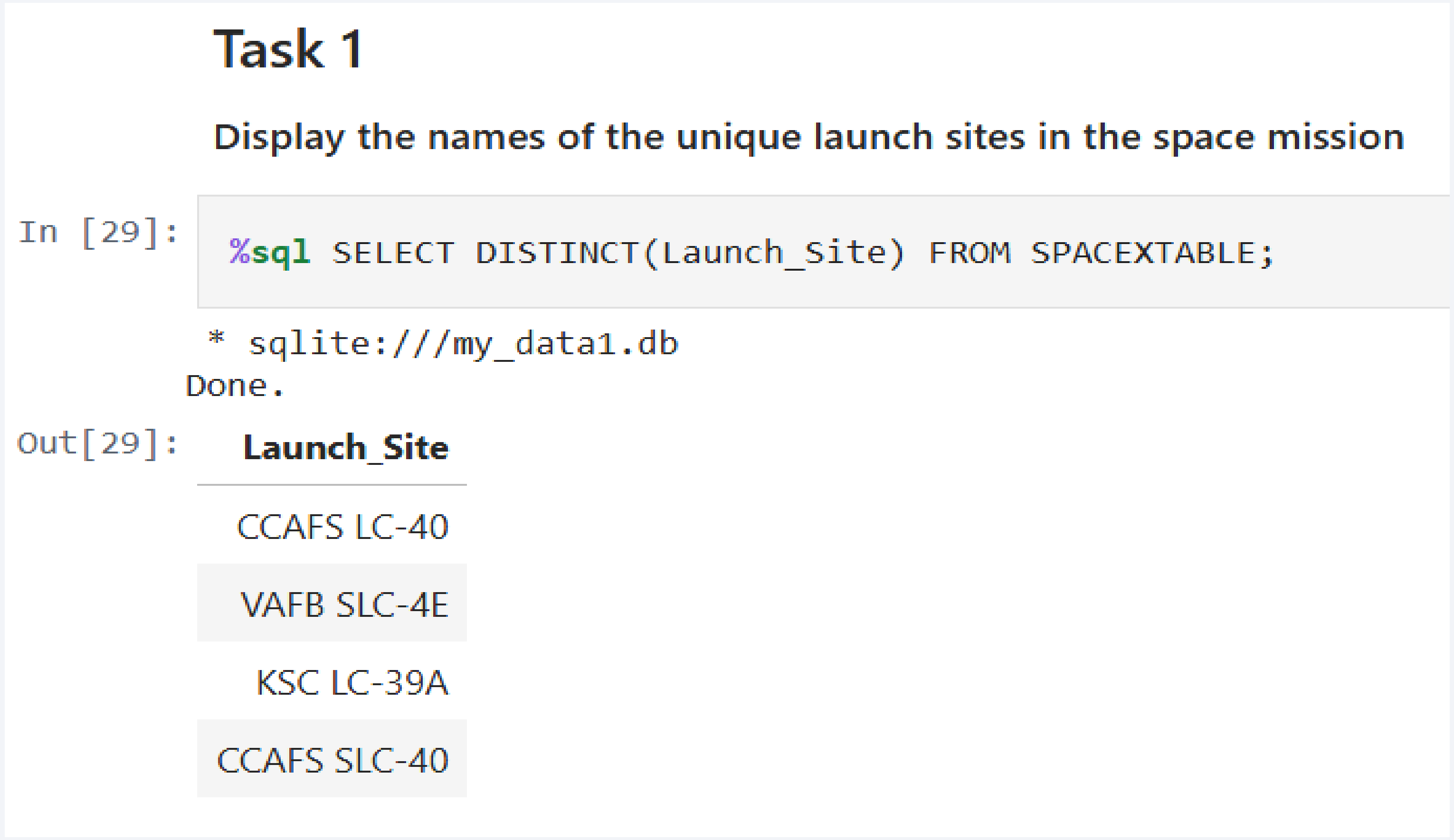
- Since 2013, the success rate kept going up till 2020



All Launch Site Names

- Find the names of the unique launch sites

- Used 'SELECT DISTINCT' statement to return only the unique launch sites from the 'LAUNCH_SITE' column of the SPACEXTBL table



Task 1

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

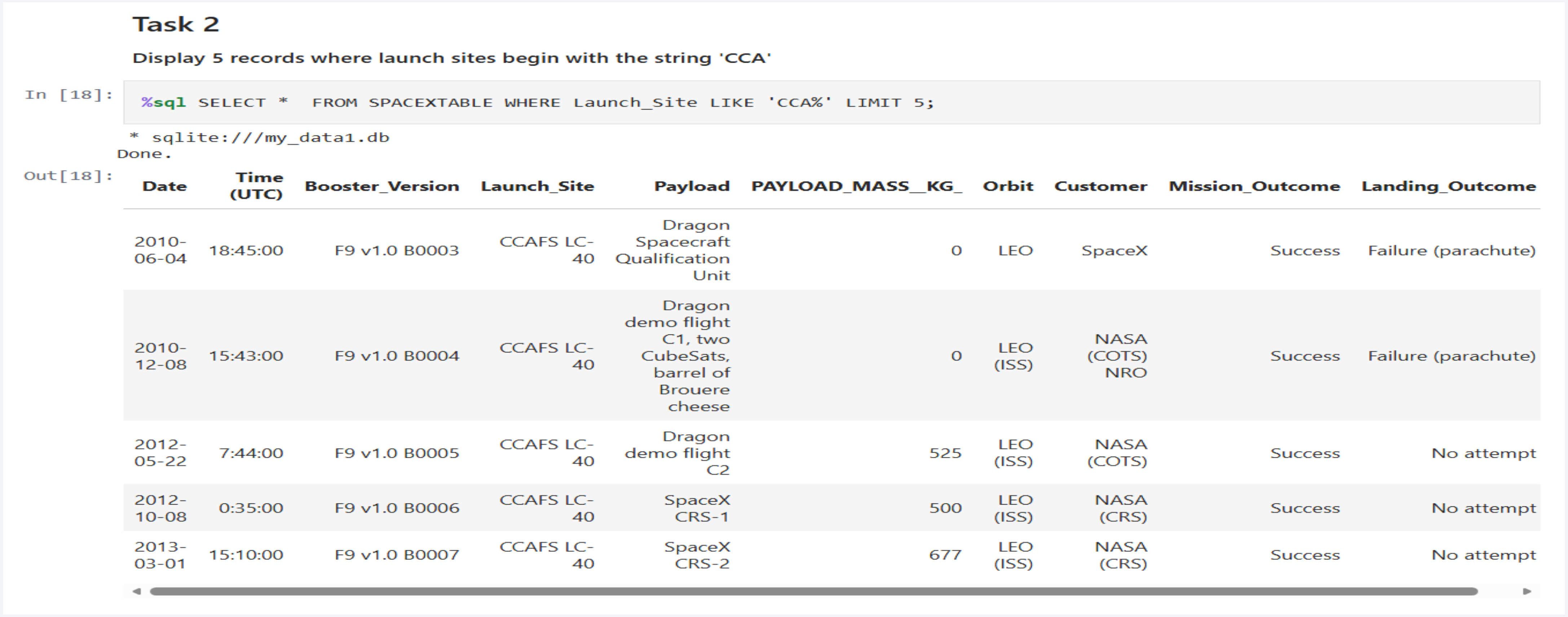
```
In [29]: %sql SELECT DISTINCT(Launch_Site) FROM SPACEXTABLE;

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

Out[29]:

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

Launch Site Names Begin with 'CCA'

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



Task 2

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

```
In [18]:  %sql SELECT *  FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;

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

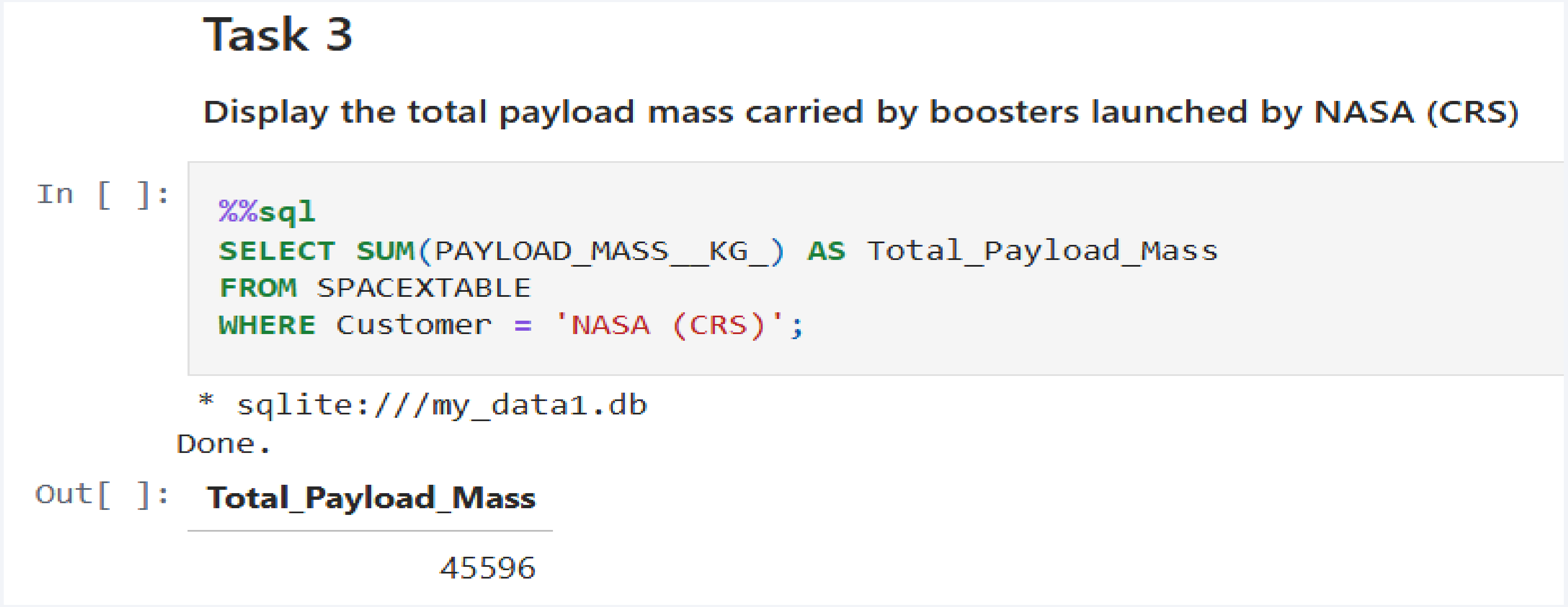
Out[18]:

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

- Used ‘LIKE’ command with ‘%’ wildcard in ‘WHERE’ clause to select and dispay a table of all records where **launch sites begin with the string 'CCA'**

Total Payload Mass

- Calculate and Display the total payload carried by boosters from NASA



Task 3

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

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

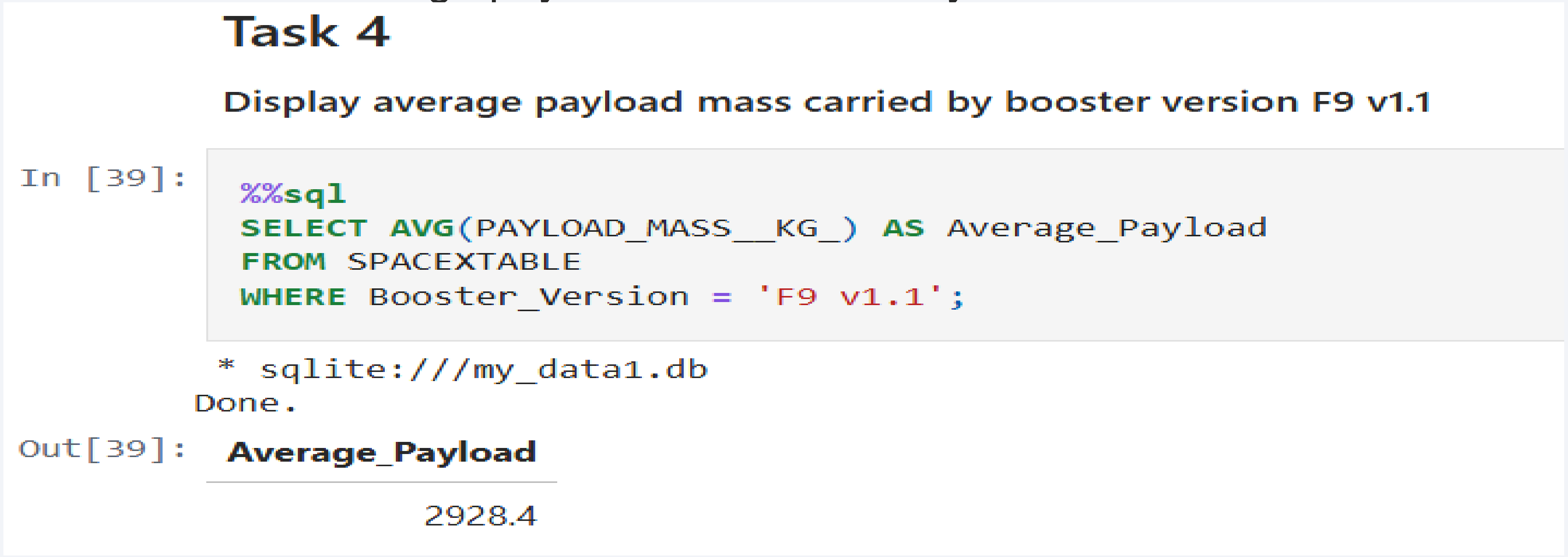
```
 * sqlite:///my_data1.db
Done.
```

```
Out[ ]:  Total_Payload_Mass

                      45596
```

- Used the ‘SUM()’ function to return and dispaly the total sum of ‘**PAYLOAD_MASS_KG**’ column for Customer ‘**NASA(CRS**’

Average Payload Mass by F9 v1.1

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



Task 4

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

```
In [39]:  %%sql
          SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload
          FROM SPACEXTABLE
          WHERE Booster_Version = 'F9 v1.1';
```

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

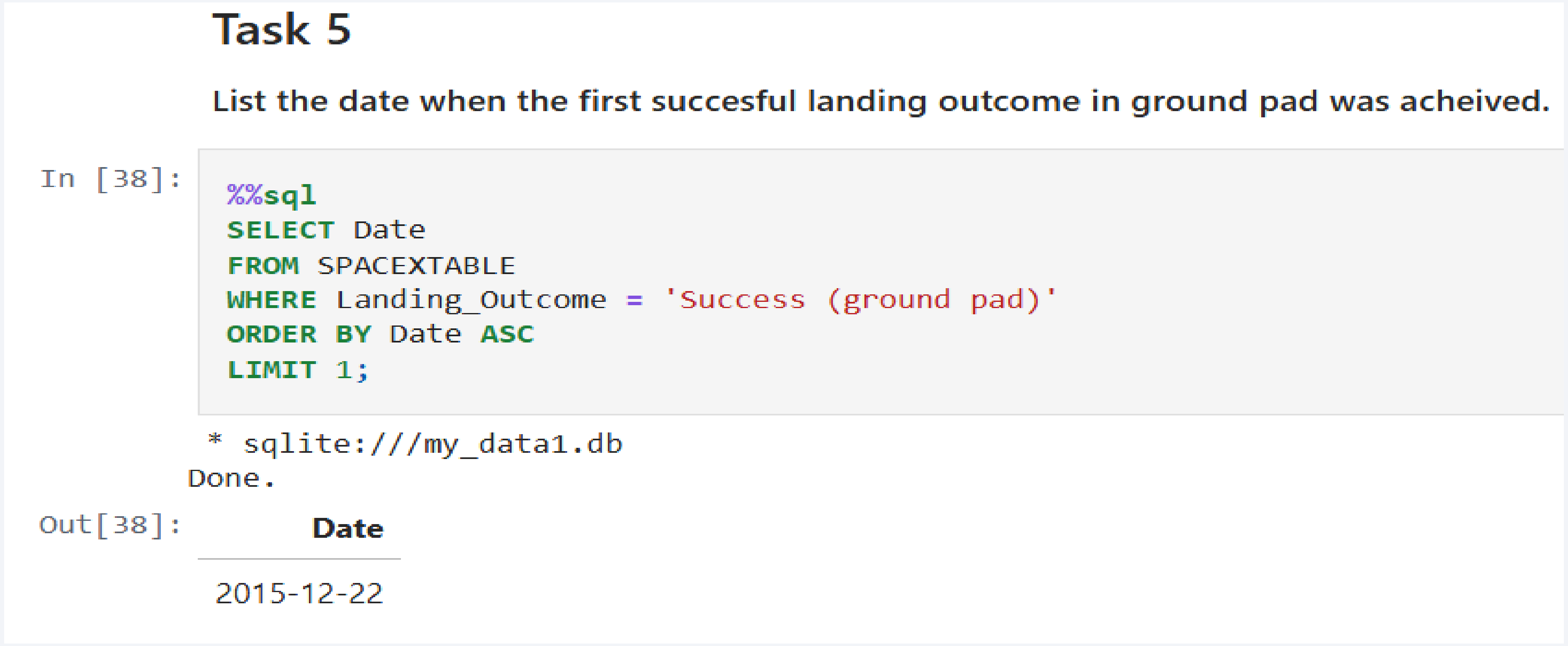
Out[39]:

Average_Payload
2928.4

- Used the 'AVG()' function to return and dispaly the average payload mass carried by booster version F9 v1.1

First Successful Ground Landing Date

- Find the dates 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.

```
In [38]:  %%sql
          SELECT Date
          FROM SPACEXTABLE
          WHERE Landing_Outcome = 'Success (ground pad)'
          ORDER BY Date ASC
          LIMIT 1;
```

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

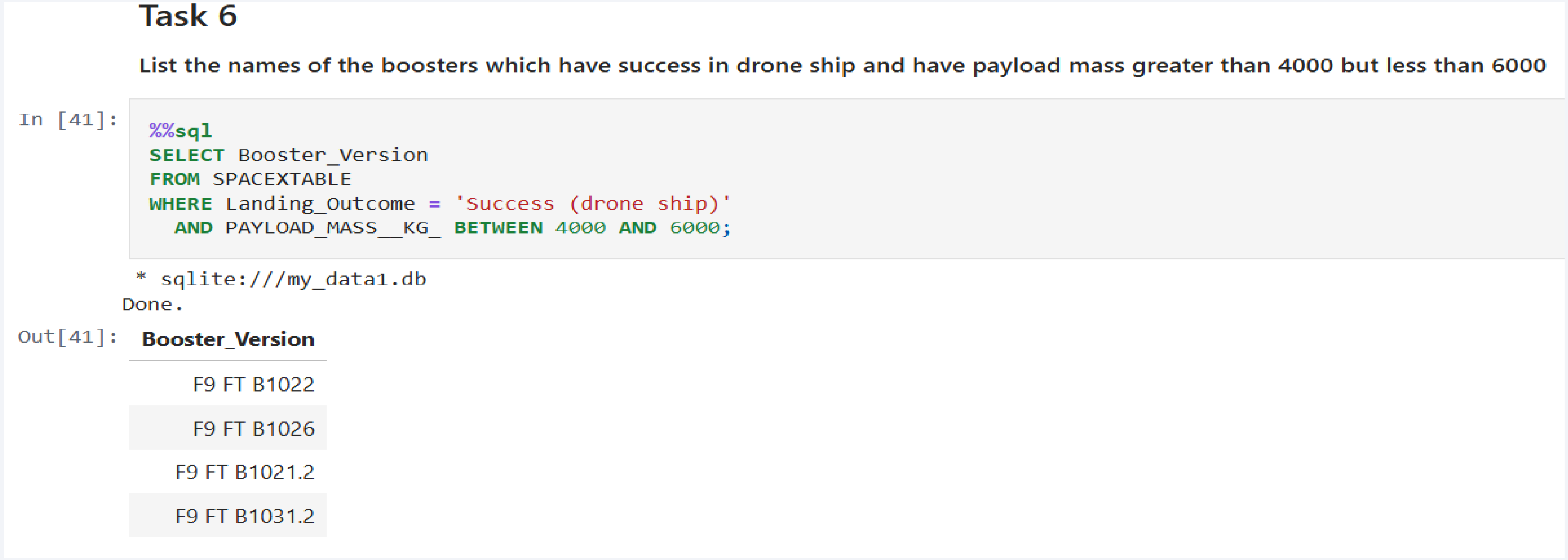
Out[38]:

Date
2015-12-22

- Used the ‘MIN()’ function to return and dispaly the first (oldest) date when first successful landing outcome on ground pad ‘***Success (ground pad)***’happened.

Successful Drone Ship Landing with Payload between 4000 and 6000

- List of Boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000



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

```
In [41]:
%%sql
SELECT Booster_Version
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (drone ship)'
  AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

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

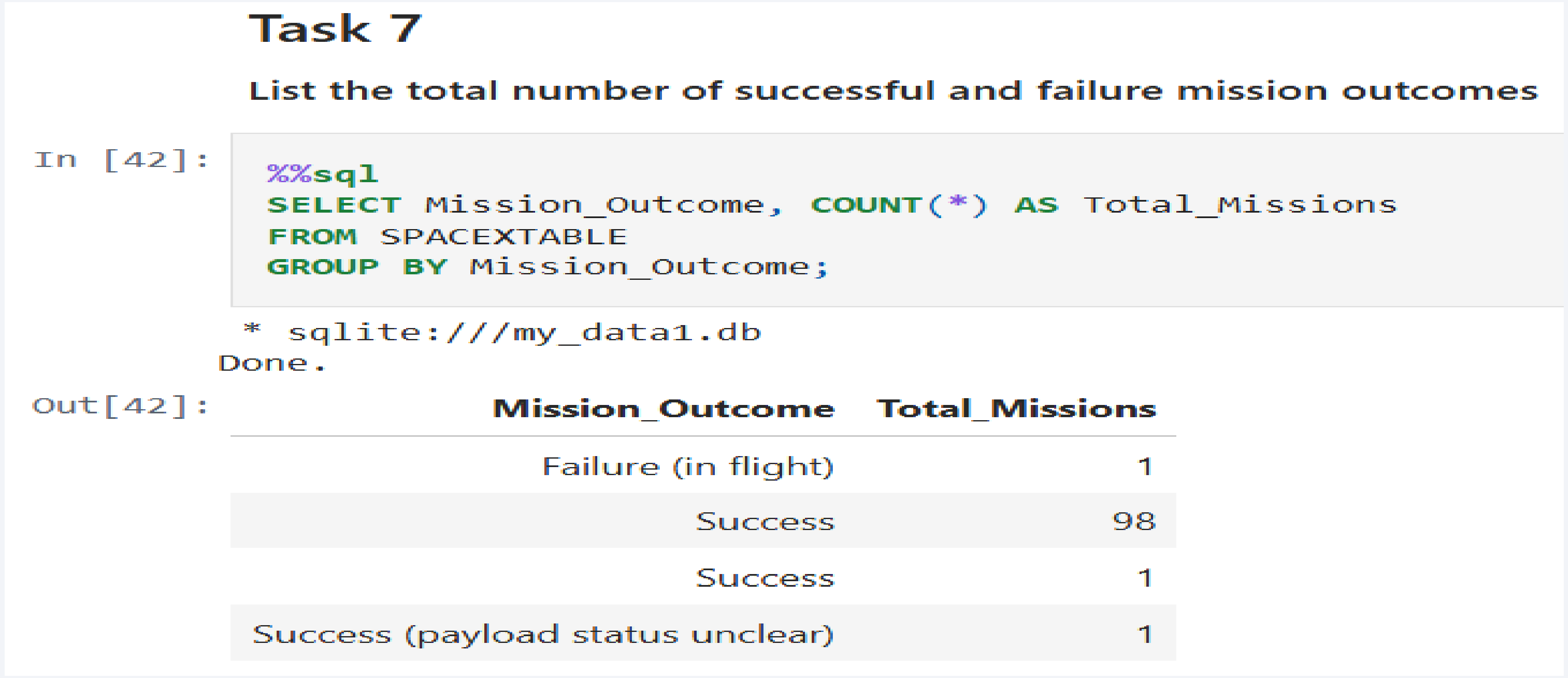
Out[41]:

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

- Used 'Select Distinct' statement to return and list the 'unique' names of boosters with operators >4000 and <6000 to only list booster with payloads btween 4000-6000 with landing outcome of 'Success (drone ship)'.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes



Task 7

List the total number of successful and failure mission outcomes

```
In [42]:
%%sql
SELECT Mission_Outcome, COUNT(*) AS Total_Missions
FROM SPACEXTABLE
GROUP BY Mission_Outcome;
```

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

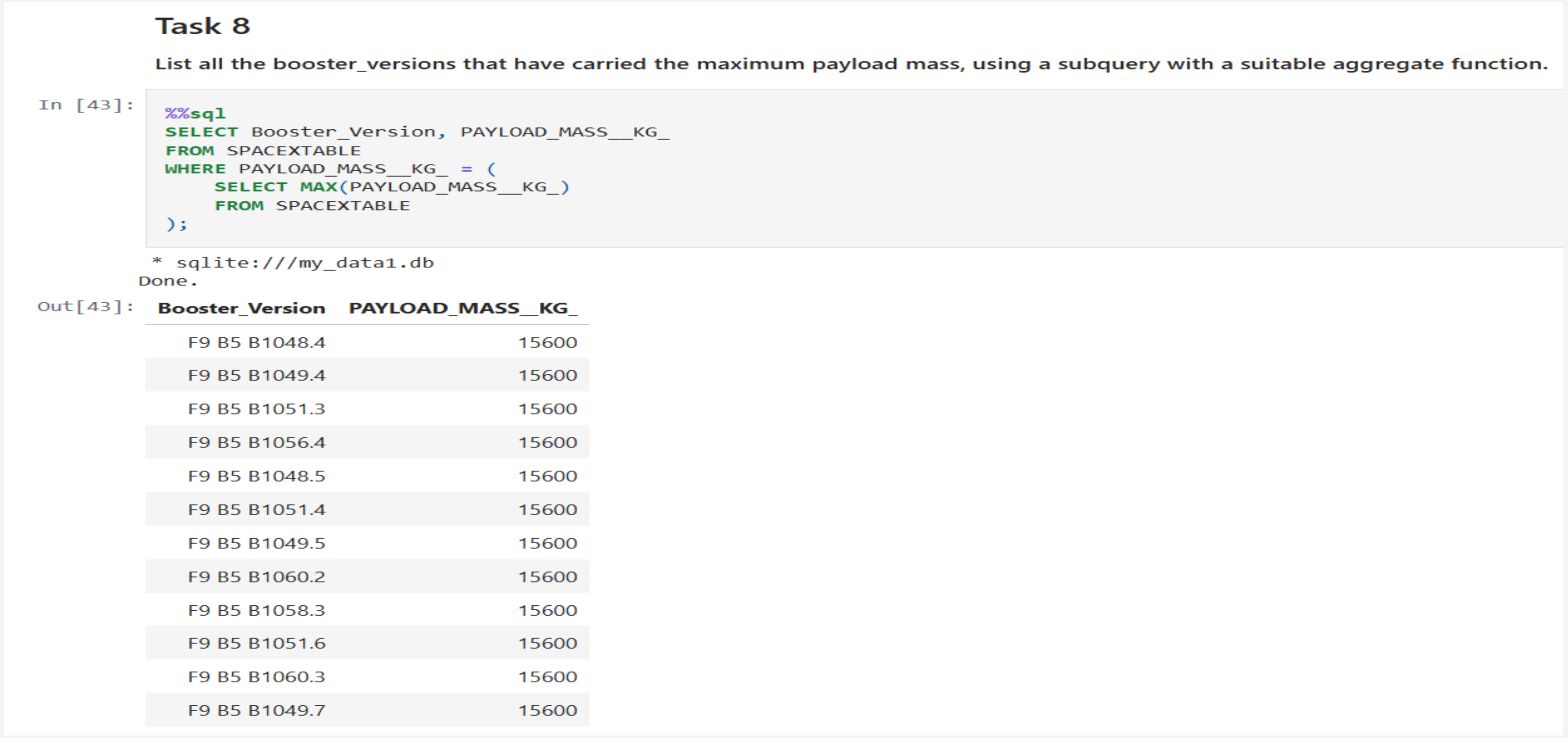
Out[42]:

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

- Used the ‘COUNT()’ together with the ‘GROUP BY’ statement to return total number of missions outcomes

Boosters Carried Maximum Payload

- List of the boosters which have carried the maximum payload mass



Task 8

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

```
In [43]: %%sql
         SELECT Booster_Version, PAYLOAD_MASS__KG_
         FROM SPACEXTABLE
         WHERE PAYLOAD_MASS__KG_ = (
             SELECT MAX(PAYLOAD_MASS__KG_)
             FROM SPACEXTABLE
         );
```

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

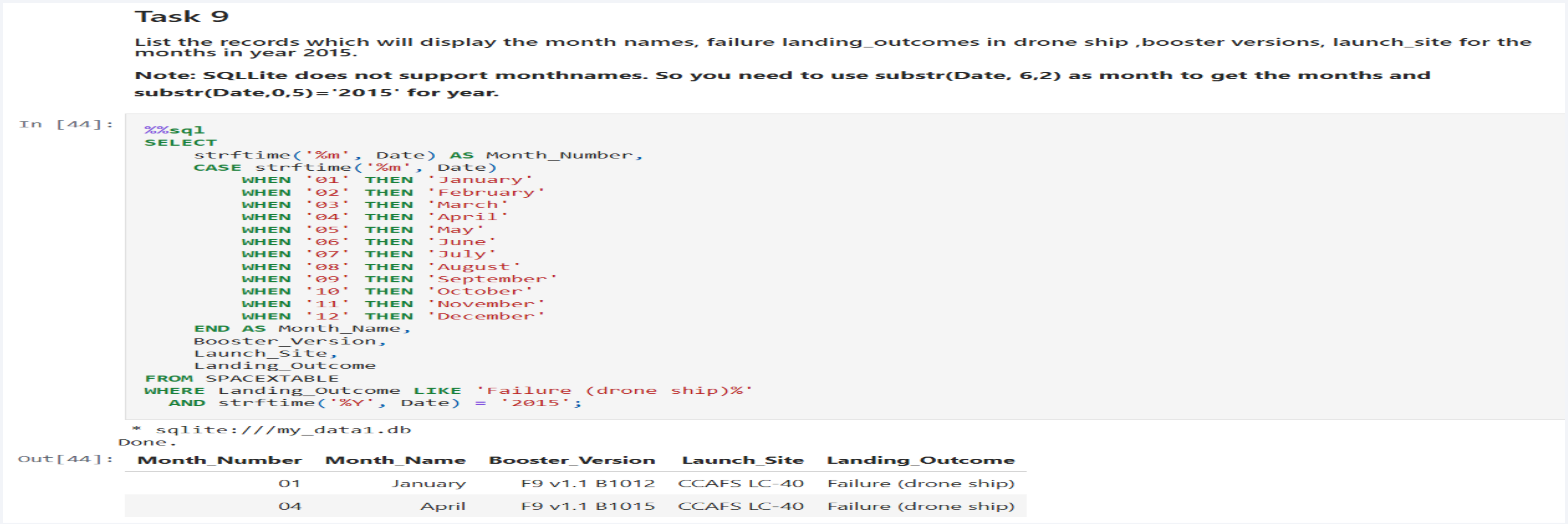
Out[43]:

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

- Using a Subquerry to return and pass the Max payload and used it list all the boosters that have carried the Max payload of 15600kgs

2015 Launch Records

- List of failed landing outcomes in drone ship, with their booster versions, and launch site names in 2015



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.

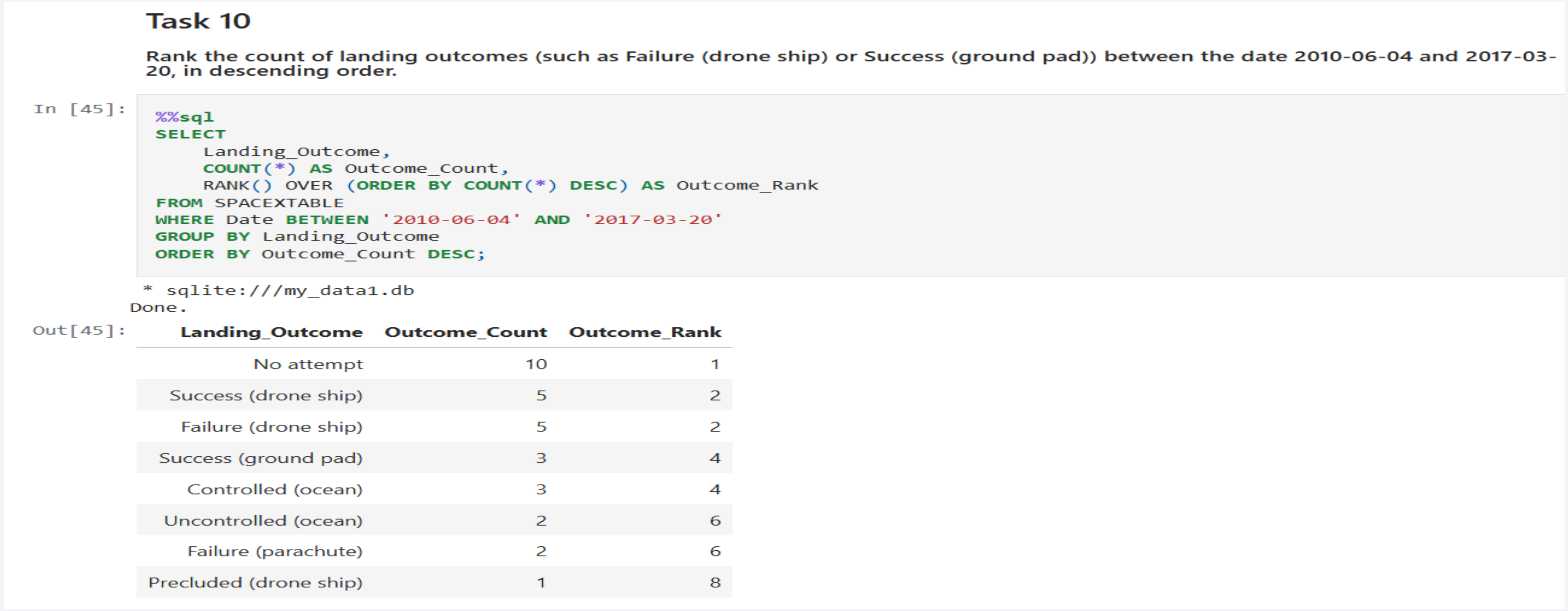
```
In [44]:  %%sql
          SELECT
              strftime('%m', Date) AS Month_Number,
              CASE strftime('%m', Date)
                  WHEN '01' THEN 'January'
                  WHEN '02' THEN 'February'
                  WHEN '03' THEN 'March'
                  WHEN '04' THEN 'April'
                  WHEN '05' THEN 'May'
                  WHEN '06' THEN 'June'
                  WHEN '07' THEN 'July'
                  WHEN '08' THEN 'August'
                  WHEN '09' THEN 'September'
                  WHEN '10' THEN 'October'
                  WHEN '11' THEN 'November'
                  WHEN '12' THEN 'December'
              END AS Month_Name,
              Booster_Version,
              Launch_Site,
              Landing_Outcome
          FROM SPACEXTABLE
          WHERE Landing_Outcome LIKE 'Failure (drone ship)%'
            AND strftime('%Y', Date) = '2015';

           * sqlite:///my_data1.db
          Done.
Out[44]:
```

Month_Number	Month_Name	Booster_Version	Launch_Site	Landing_Outcome
01	January	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	April	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- Used the 'subsrt()' in the select statement to get the month and year from the date column where substr(Date,7,4)='2015' for year and Landing_outcome was 'Failure (drone ship')
and return the records nmatching the filter.

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



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.

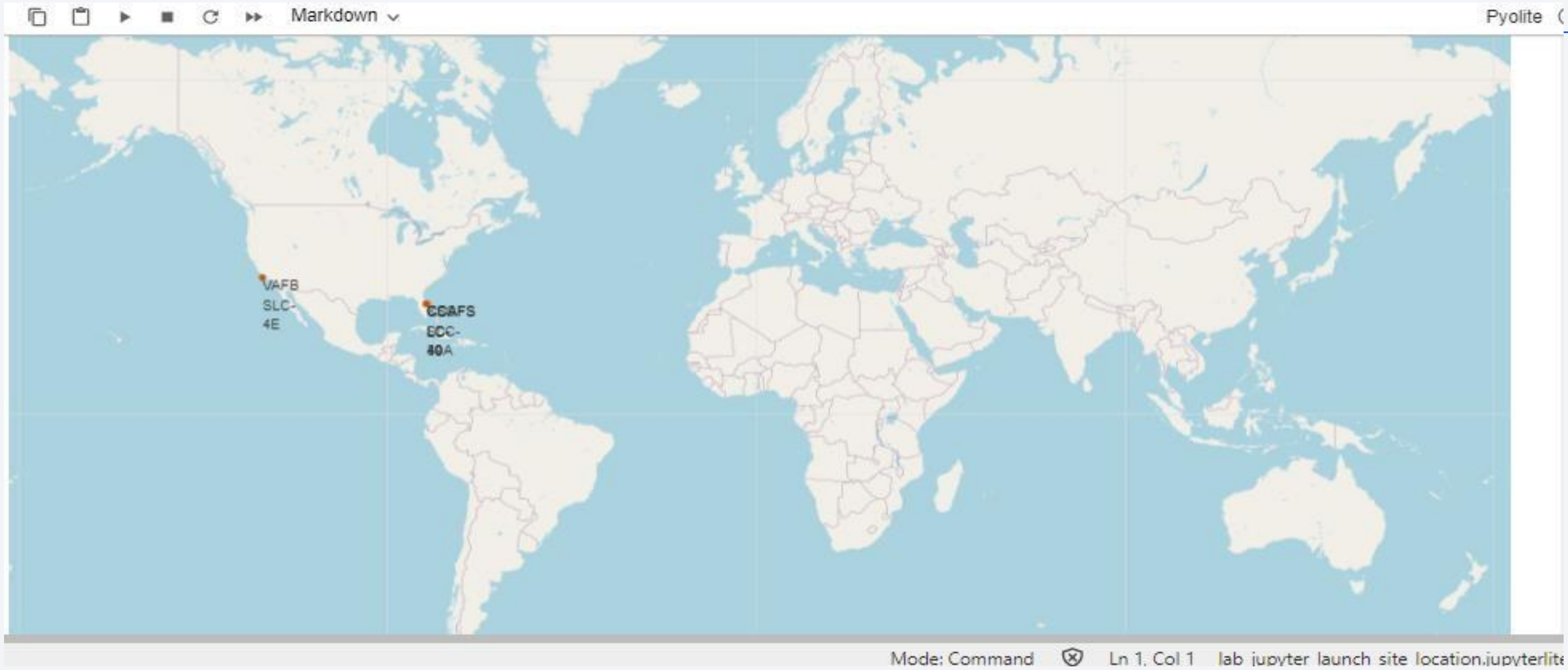
```
In [45]: %%sql
         SELECT
             Landing_Outcome,
             COUNT(*) AS Outcome_Count,
             RANK() OVER (ORDER BY COUNT(*) DESC) AS Outcome_Rank
         FROM SPACEXTABLE
         WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
         GROUP BY Landing_Outcome
         ORDER BY Outcome_Count DESC;

          * sqlite:///my_data1.db
         Done.
Out[45]:
```

Landing_Outcome	Outcome_Count	Outcome_Rank
No attempt	10	1
Success (drone ship)	5	2
Failure (drone ship)	5	2
Success (ground pad)	3	4
Controlled (ocean)	3	4
Uncontrolled (ocean)	2	6
Failure (parachute)	2	6
Precluded (drone ship)	1	8

Section 3

Launch Sites Proximities Analysis

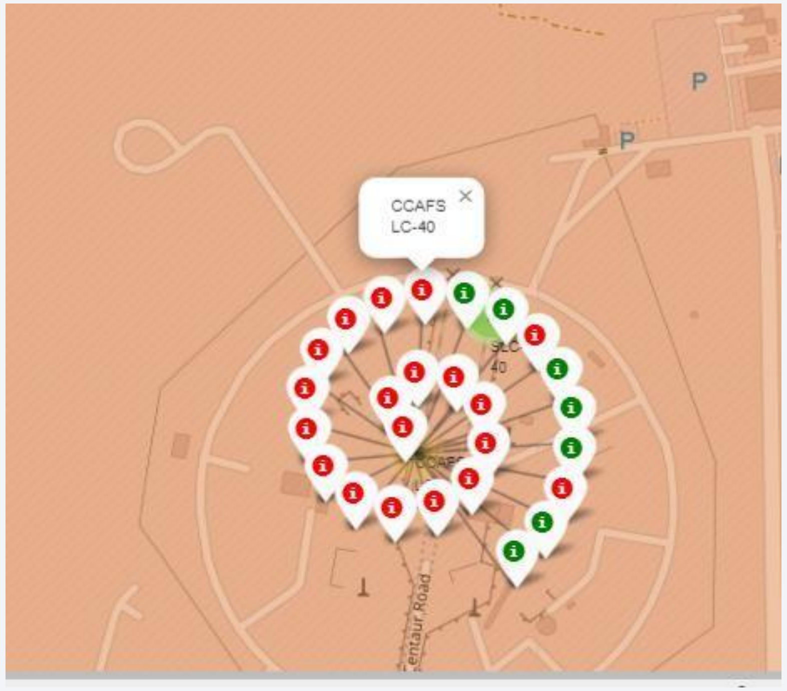
Markers of all launch sites on global map

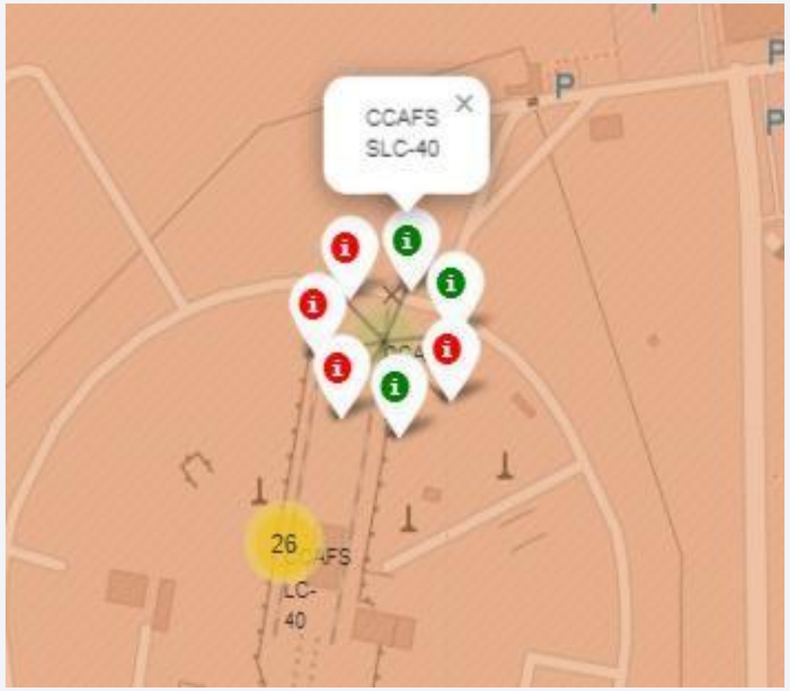


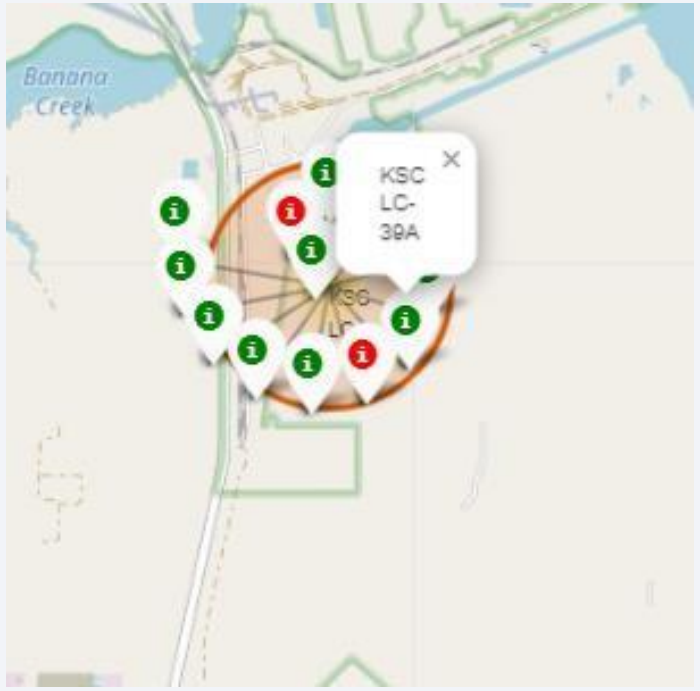
- All launch sites are in proximity to the Equator, (located southwards of the US map). Also all the laumch sites are in very close proximity to the coast.

Launch outcomes for each site on the map With Color Markers

Florida Sites



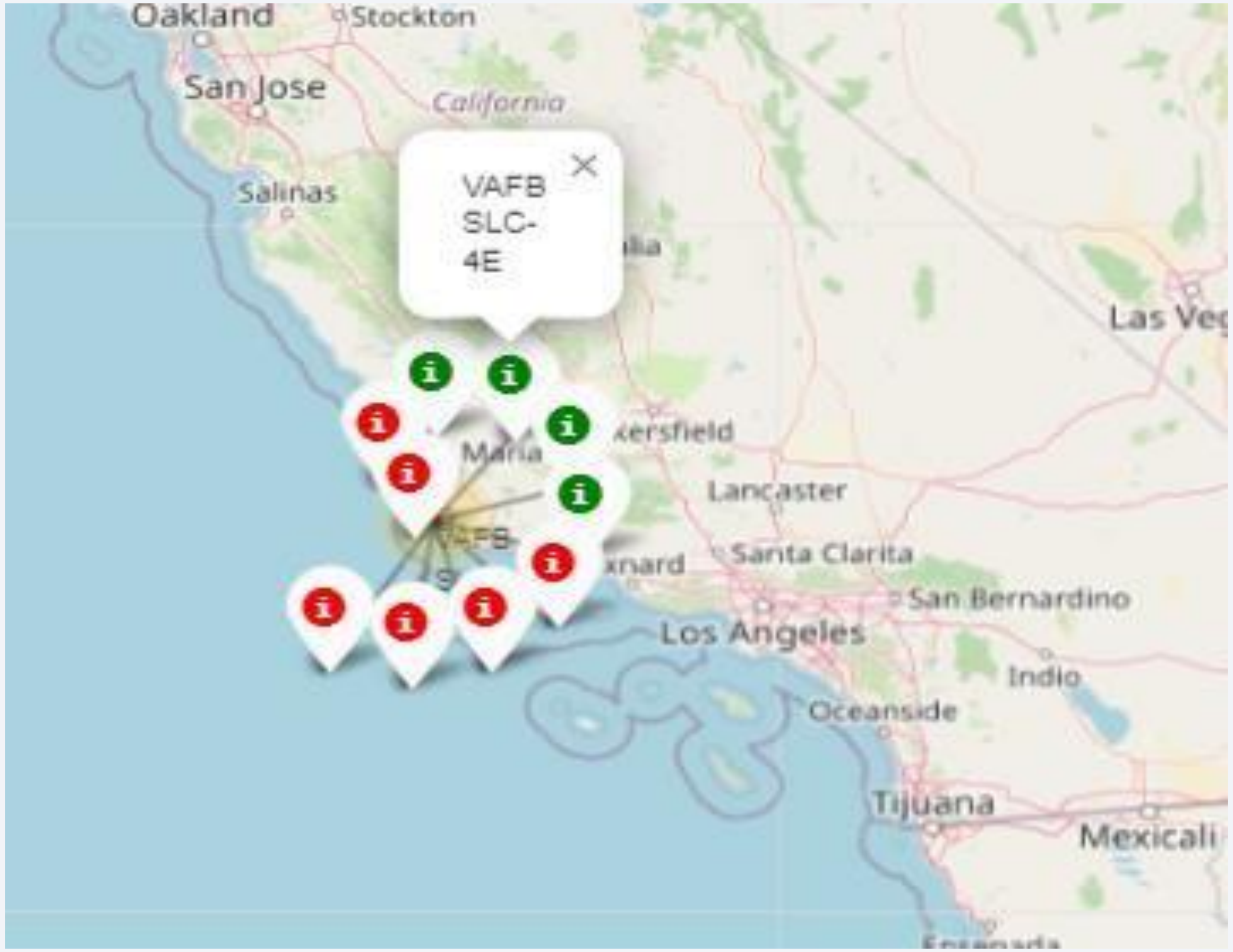




- In the Eastern coast (Florida) Launch site KSC LC-39A has relatively high success rates compared to CCAFS SLC-40 & CCAFS LC-40.

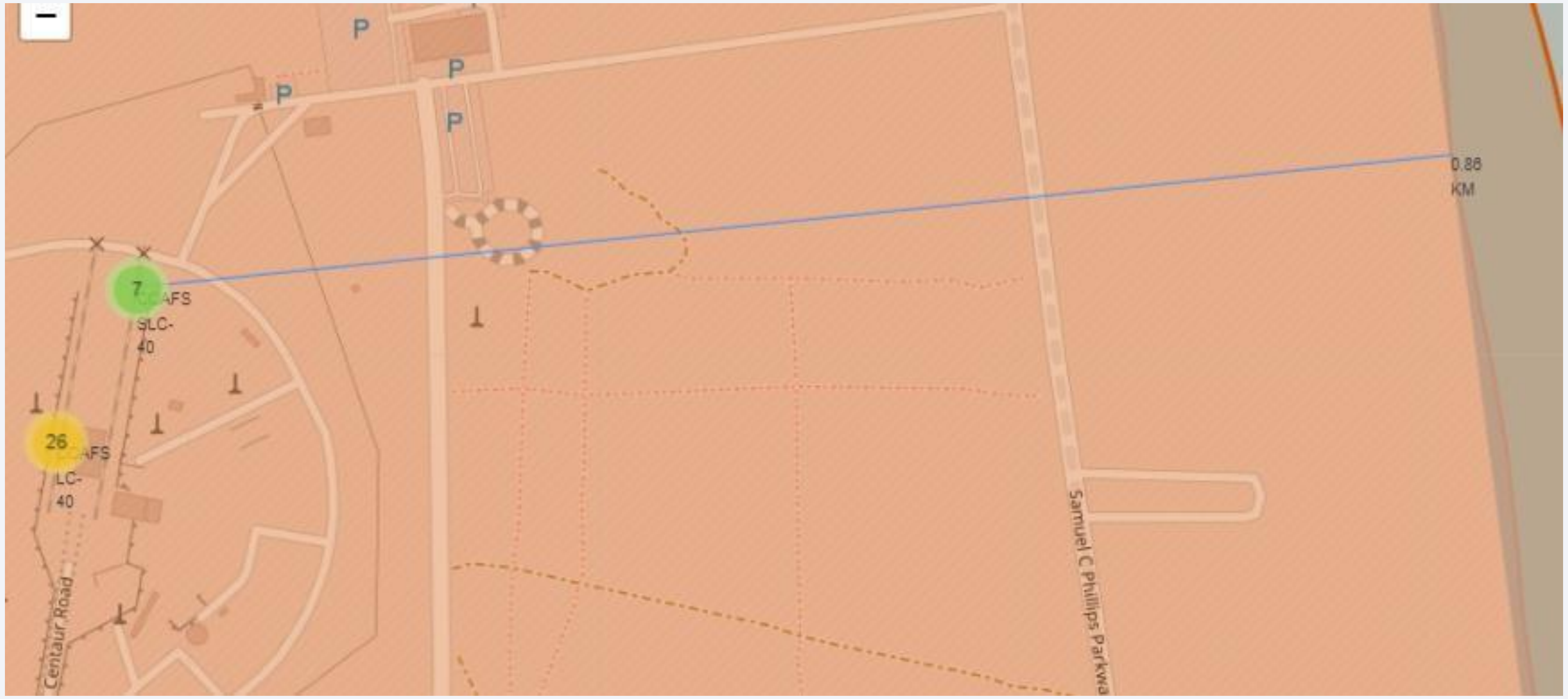
Launch outcomes for each site on the map With Color Markers

West Coast/ Carlifonia



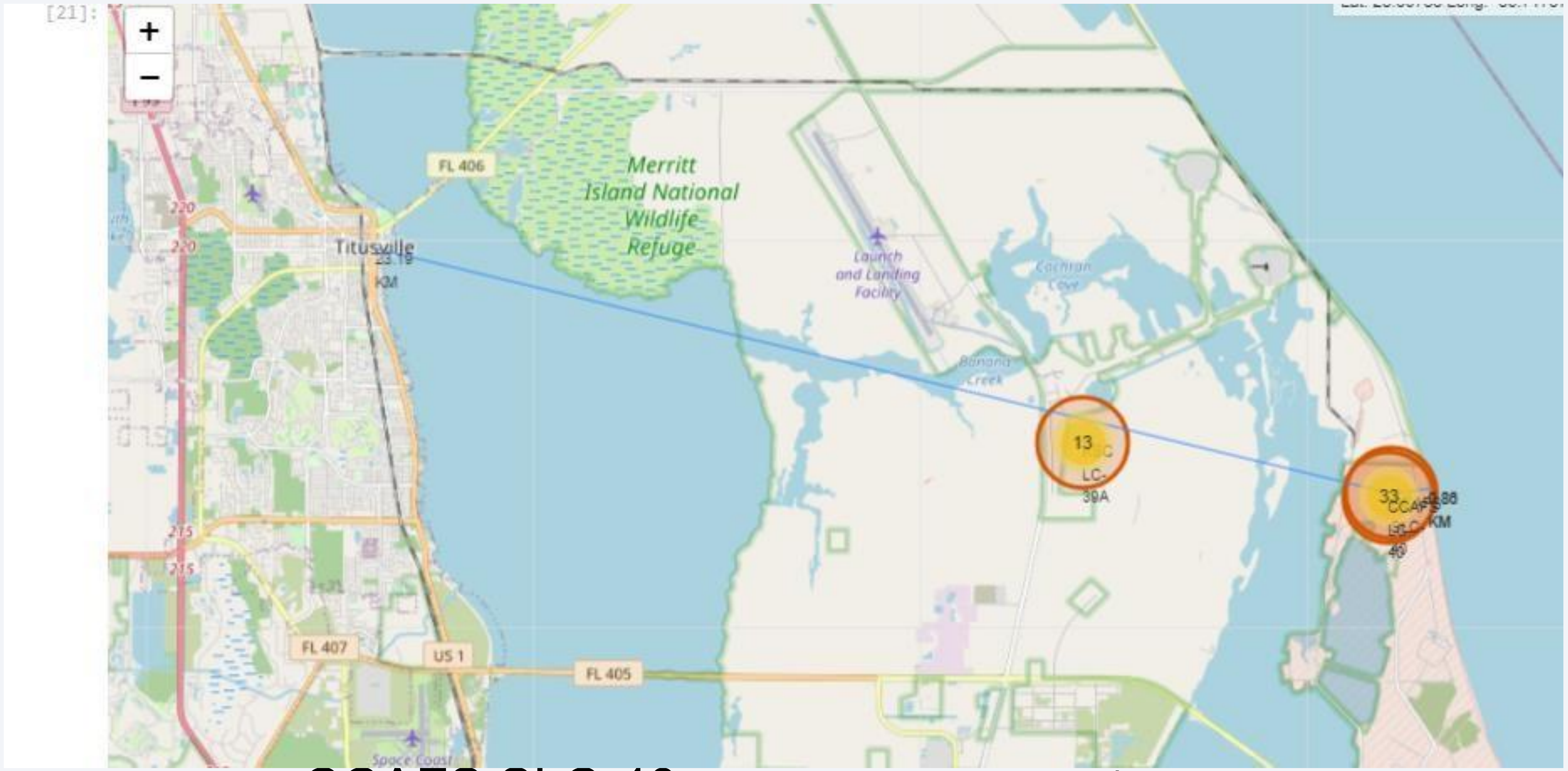
- In the West Coast (Californai) Launch site VAFB SLC-4E has relatively lower success rates 4/10 compared to KSC LC-39A launch site in the Eastern Coast of Florida.

Distances between a launch site to its proximities

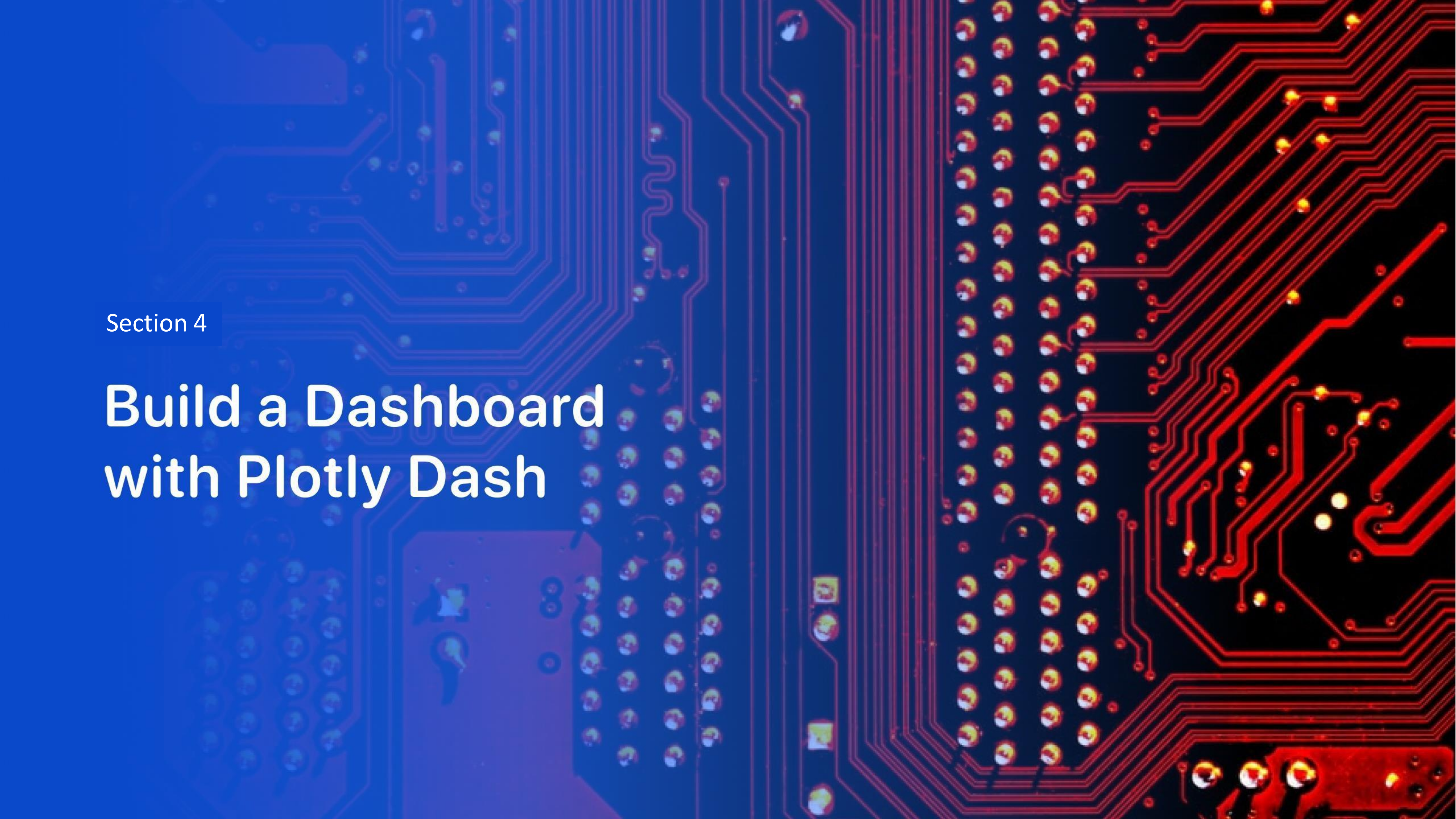


- Launch site CCAFS SLC-40 proximity to coastline is 0.86km

Distances between a launch site to its proximities



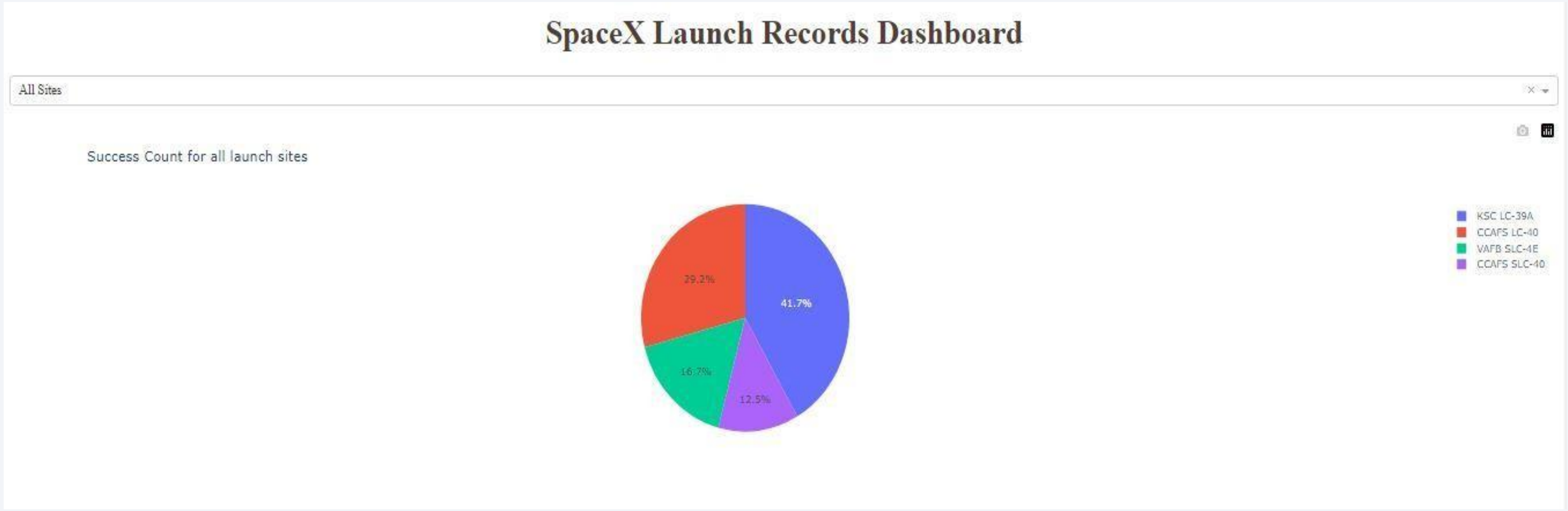
- Launch site CCAFS SLC-40 closest to highway (Washington Avenue) is 23.19km



Section 4

Build a Dashboard with Plotly Dash

Pie-Chart for launch success count for all sites



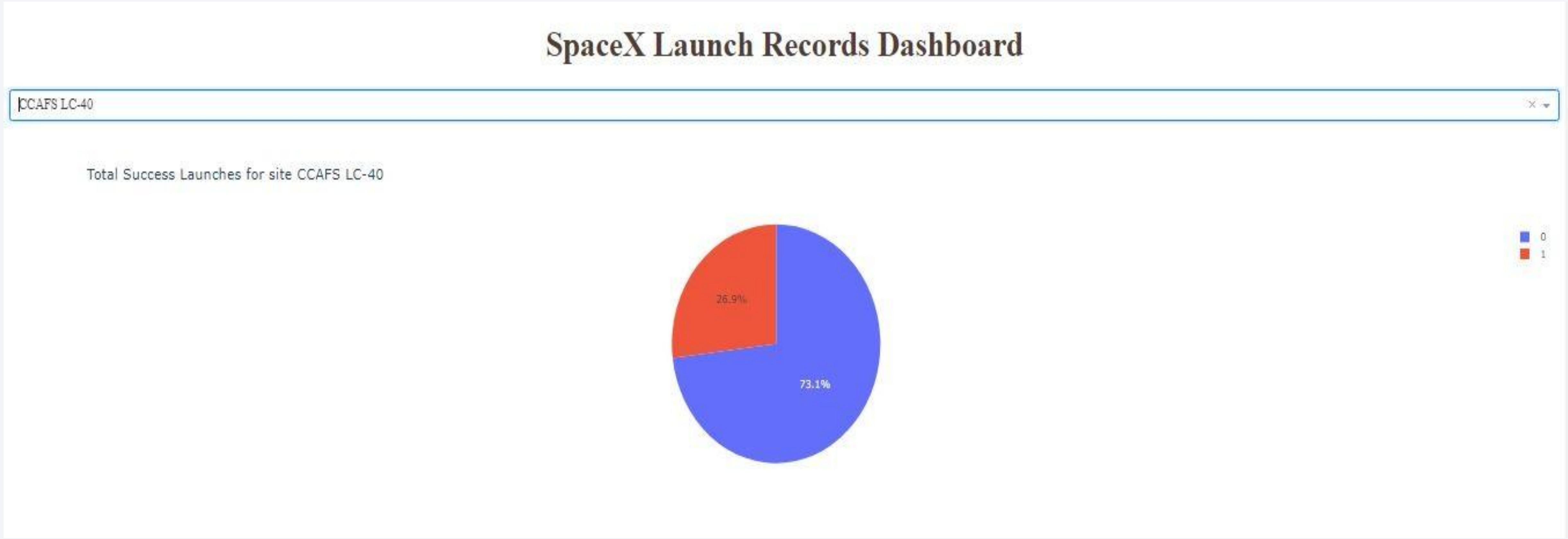
SpaceX Launch Records Dashboard

All Sites

Success Count for all launch sites

- Launch site KSC LC-39A has the highest launch success rate at 42% followed by CCAFS LC-40 at 29%, VAFB SLC-4E at 17% and lastly launch site CCAFS SLC-40 with a success rate of 13%

Pie chart for the launch site with 2nd highest launch success ratio

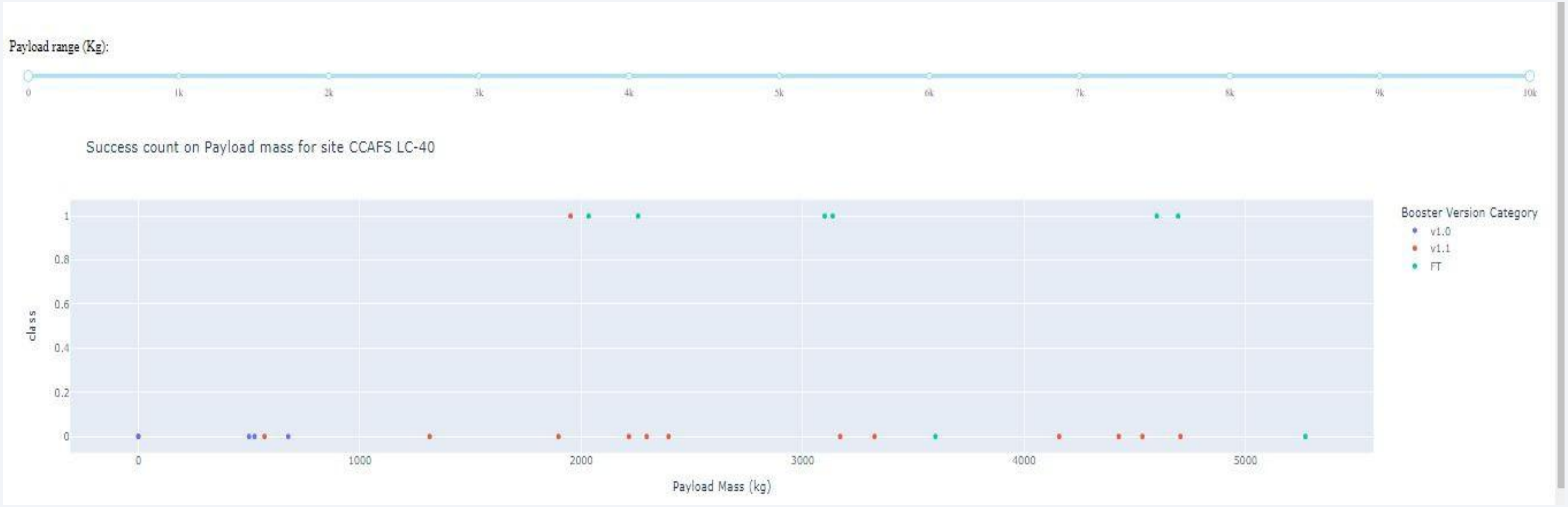

SpaceX Launch Records Dashboard

CCAFS LC-40

Total Success Launches for site CCAFS LC-40

- Launch site CCAFS LC-40 had the 2nd highest success ratio of 73% success against 27% failed launches

Payload vs. Launch Outcome scatter plot for all sites

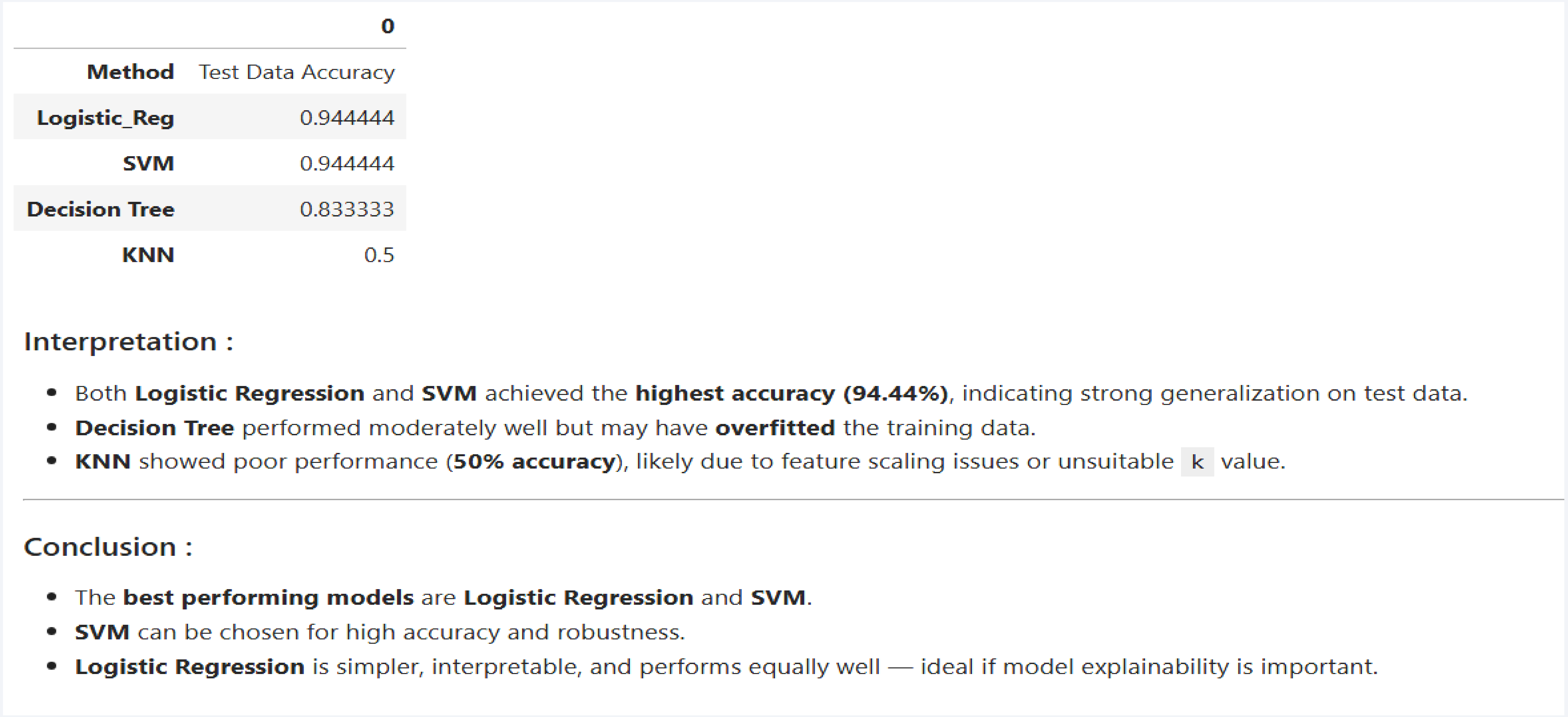


- For Launch site CCAFS LC-40 the booster version FT has the largest success rate from a payload mass of >2000kg

Section 5

Predictive Analysis (Classification)

Classification Models Accuracy



	0
Method	Test Data Accuracy
Logistic_Reg	0.944444
SVM	0.944444
Decision Tree	0.833333
KNN	0.5

Interpretation :

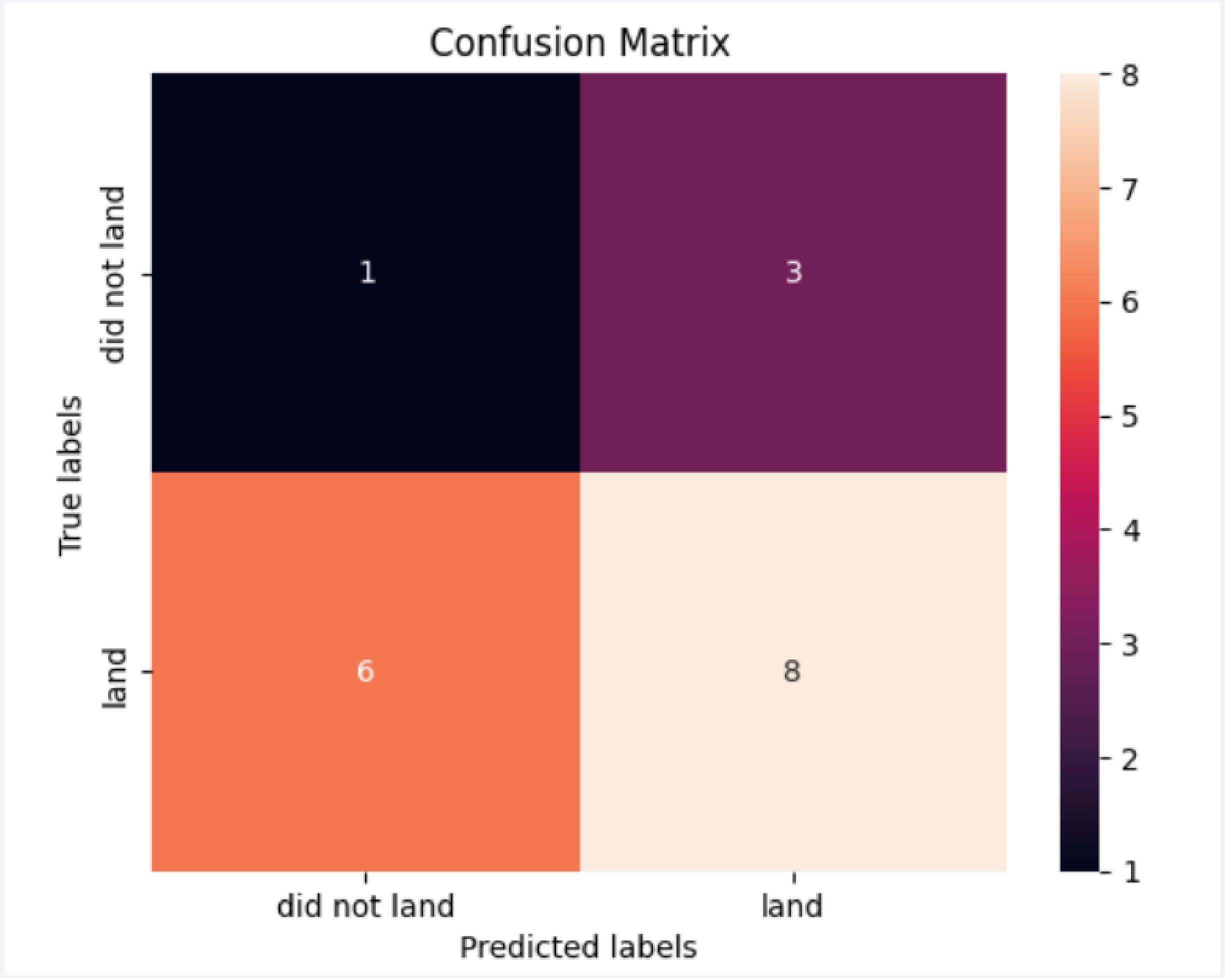
- Both **Logistic Regression** and **SVM** achieved the **highest accuracy (94.44%)**, indicating strong generalization on test data.
- **Decision Tree** performed moderately well but may have **overfitted** the training data.
- **KNN** showed poor performance (**50% accuracy**), likely due to feature scaling issues or unsuitable `k` value.

Conclusion :

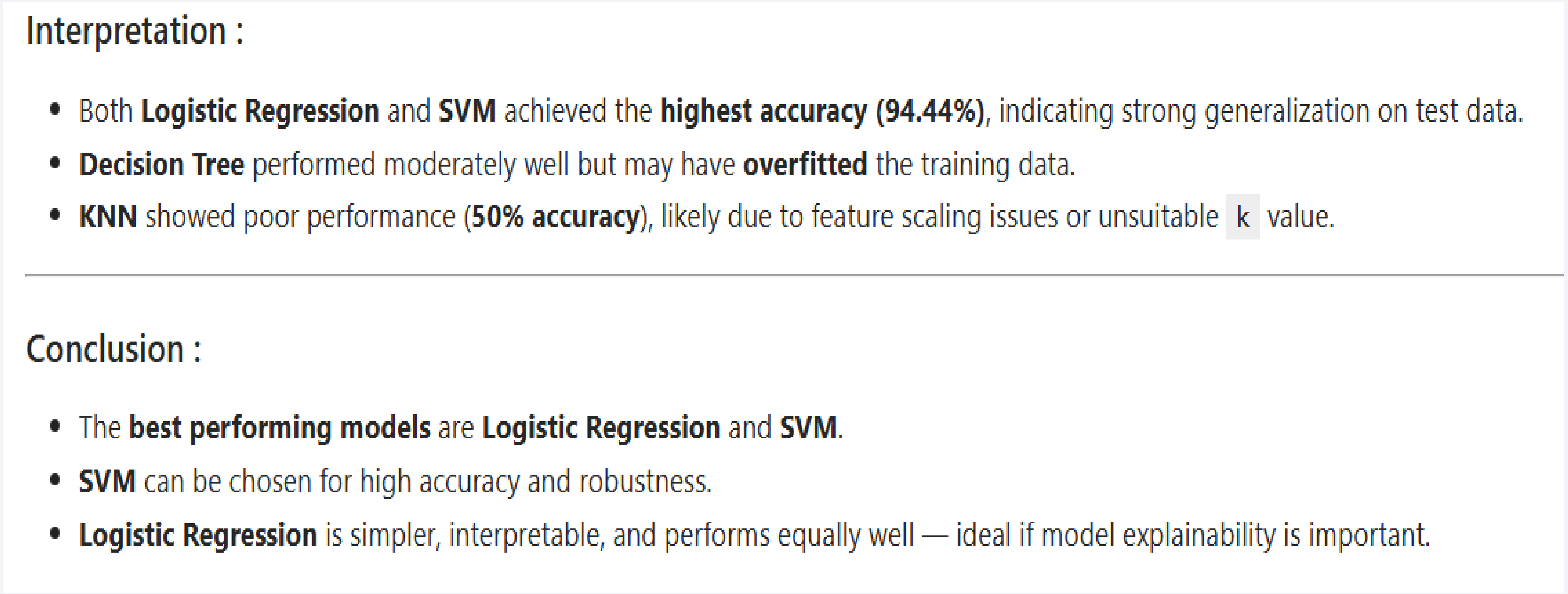
- The **best performing models** are **Logistic Regression** and **SVM**.
- **SVM** can be chosen for high accuracy and robustness.
- **Logistic Regression** is simpler, interpretable, and performs equally well — ideal if model explainability is important.

Confusion Matrix

- All the 4 classification model had the same confusion matrixes and were able equally distinguish between the different classes. The major problem is false positives for all the models.



Conclusions



Interpretation :

- Both **Logistic Regression** and **SVM** achieved the **highest accuracy (94.44%)**, indicating strong generalization on test data.
- **Decision Tree** performed moderately well but may have **overfitted** the training data.
- **KNN** showed poor performance (**50% accuracy**), likely due to feature scaling issues or unsuitable `k` value.

Conclusion :

- The **best performing models** are **Logistic Regression** and **SVM**.
- **SVM** can be chosen for high accuracy and robustness.
- **Logistic Regression** is simpler, interpretable, and performs equally well — ideal if model explainability is important.

Conclusions Cont.…

- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here
- Anf finally the sucess rate since 2013 kept increasing till 2020.

Thank you!